Offline and Online Optimization with Applications in Online Advertising

by

Alfonso Andres Lobos Ruiz

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Industrial Engineering and Operations Research

in the

Graduate Division

of the

University of California, Berkeley

Spring 2021

Abstract

Offline and Online Optimization with Applications in Online Advertising

by

Alfonso Andres Lobos Ruiz

Doctor of Philosophy in Engineering - Industrial Engineering and Operations Research

University of California, Berkeley

Professor Paul Grigas, Chair

In the last couple of decades, focus on speed and personalization has been a topic of major importance for optimization systems. The internet and big data have made users expect results immediately or with an unperceivable delay. For example, in online advertising a user is shown an ad a few milliseconds after entering a website, app, or other media. This new online environment has force optimization systems to evolve from operating in an offline fashion, *i.e.*, assuming all the information is available a priori, to an online one. In an online setting, information arrives sequentially, and systems need to make decisions as information arrives. This thesis is composed of three main chapters. The first studies an online advertising problem that serves as a motivation for the thesis as a whole. Though motivated by the leading online advertising problem, the second and third chapters make broad contributions to optimization theory and machine learning, respectively.

In the first chapter of this thesis, we develop an optimization model and corresponding algorithm to manage a demand-side platform (DSP), whereby the DSP acquires valuable ad space for its advertiser clients in a real-time bidding environment. In particular, we focus on how a DSP should bid in real-time auctions to acquire valuable ad space to allocate between its advertiser's clients. We propose a highly flexible model for the DSP to maximize its profit while maintaining acceptable budget spending levels for its advertisers' clients. We prove that a dual formulation attains a zero-duality gap under practical settings for DSPs. Using a primal-dual scheme, we derive a bidding and allocation policy that DSPs can apply in practice.

In the second chapter of this thesis, we propose a joint online optimization and learning algorithm through dual mirror descent. Part of the motivation for this topic comes from developing an online solution/policy to solve the DSP problem mentioned above. An online policy in the sense that it gets updated using simple steps after each user arrival. We achieved this goal for DSPs who buy ad space in real-time bidding environments which use second-price auction mechanisms. No complicated optimization problem needs to be solved in advance.

The contribution of this chapter of the thesis extends broadly beyond its original motivation on online advertising, making contributions in the online optimization field. In particular, we propose a new algorithm that mixes an online dual mirror descent scheme with a generic parameter learning process and a novel offline benchmark for this setup. Bounds on regret and worst possible constraints violation are studied.

In the third chapter of this thesis, we propose training neural networks jointly using subgradient-descent, Frank-Wolfe, and Frank-Wolfe variants called in-face directions. An important motivation for this chapter is how to add structure to neural networks in a principled manner. In particular, if we can promote sparsity in some layers of a neural network, we can make the overall inference step faster/cheaper. The latter is a major concern in production systems in online advertising in which the inference step of a neural network may be called billions of times per day.

To my parents and sister, for their unconditional love and support.

Contents

List of Figures

List of Tables

Acknowledgments

This thesis and associated research would not have been possible without the help and support of my family, friends, advisor, and great people who guide me during these years.

I start by thanking my parents, Julio and Veronica, and my sister, Daniela, for their support. I started my PhD odyssey in 2015, and my family has always been there to support me. I hope that once this pandemic has settled, we can celebrate the closure of this step together.

I would also like to thank Paul Grigas, my advisor, for this process that we have experienced together. Through Paul, I have learned many exciting problems in optimization, machine learning, and many other areas. Also, as his first student ever, we have learned together about many different topics both inside and outside academia. I hope we can continue collaborating or sharing experiences in years to come.

I want to thank my bosses, mentors, and collaborators I have enjoyed interacting with during these years. From Yahoo, I would like to thank Aaron Flores and Kuang-Chih Lee for introducing me to the world of online advertising and many work and non-work related topics. Also, I would like to thank professor Jose Blanchet for the great conversations and how you have advised me when I have felt lost. Finally, I can not forget my friends, mentors, and collaborators Junwei Pan and Zheng Wen. With both of you, I have been fortunate to share friendship, work, and academia.

From professors and staff from IEOR, I would like to thank Ilan Adler for allowing me to be his GSI twice and for the great discussions we had about optimization and academia in general. I want to thank professor Lee Fleming for allowing me to help him in his entrepreneurial endeavours. And want to thank professor Dorit Hochbaum who join me in her research group when I started my PhD. Finally, I want to thank the great IEOR staff who helped me on many occasions Anayancy, Keith, Heather, Diana, and Rebecca. It has been countless times in which I have relied on you.

Finally, I want to thank my friends from life as well. My amazing DAFTS group Daniel, Felipe, Tomas, Sofia, and "Lab Mala Ondi" including Nicolas, Pascal, Jorge, Rene T., Rene M., Claudio, Mario, Yoram (and Fernando). "Los Hey Para" Pablo, Emilio, Parli, Alvaro, Rucio, Kani, Ignacio, Parra and other friends from high school Alexis A., Gerald F., Jose H. Good friends from undergrad Sebastian E., Patricio N., Matias V., Matias L., Cesar H., Lopy, Carlos M., Rodrigo B., Ernesto S., Felipe C., Jose T., my neighbours Rodrigo and Jaime C., and many friends from life Katy W., Martin M., Steph S., Cyril T., Anna M., Tina A., Camila F. and others that I may be forgetting. My housemates, through these years Vivek A., Chao Ju, Matt T., Clark S., Cristian D., Dietrich D., Alex K., Paty H., Katya C., Jacob F., Galia M. and others. Finally, I would like to thank my friends from Berkeley and IEOR Salar F., Pedro H., Dean G., Cristobal P., Han F., Mark V., Erik B., Junyu C., Carlos D., Haoyang C., Xu R., Matt O., Georgios P., Yonatan M., Pelaggie U., Quico S., Jiung L., Kevin L., Yuhao D., Cedric J., Arman J., Igor M., Sang W., Jose L., Valeri V. Special shoutout to the musketeers Nishant L. and Kilian S. Kilian, you left early, but I hope we can meet again when the time is due.

Chapter 1

Introduction

In the last couple of decades, focus on speed and personalization has been a major topic for optimization systems. The internet and big data have made users expect results immediately or with an unperceivable delay. For example, when a user enters an app or website, the website layout or app visuals may need to adapt depending on user characteristics. From an optimization point of view, we think of scenarios in which orders/requests/users arrive sequentially, and the system needs to respond in a few milliseconds. Of particular importance for this thesis is the case of online advertising. In the latter case, an ad is shown to a user a few milliseconds after the user enters a website, start using an app, etc. The information of a user arrival to a website or app is sold in a real-time auction in which different companies called Demand-Side Platforms (DSPs) participates. How a DSP should bid in these real-time auctions to later allocate the obtained ad space between its advertisers' clients is the motivating problem for this thesis.

This thesis makes contributions in the areas of optimization theory, machine learning, and online advertising. The general outline of how the different chapters and contributions are related is as follows. Chapter 2 studies in detail the bidding and allocation problem for a DSP. Using a primal-dual scheme, we prove that our methodology is the best possible in a strong sense for many practical settings, in our case, auction types used in practice. Also, we derive a policy for DSPs that links our theoretical results to practical implementation. Feedback both from industry and academia taught us that some parameters we assumed known could be not. Either a DSP may not have good estimates of these parameters or these parameters may be observed only in a real-time fashion. For example, an adequate way to classify or partition users depending on their information, *e.g.*, cookie history, may not be available. Also, a DSP may not have a good forecast of the expected number of users with different characteristics that will arrive in a given period. The latter issues motivate Chapter 3, where we study 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. In terms of the motivating DSP problem, users entering to websites, using apps, etc. are represented as sequential context vector arrivals. As an application of the mathematics used and studied on Chapter 3, we were able to derive an online optimization policy when real-time auctions use second-price auction mechanisms. The latter meaning that no offline problem needs to be solved in advance. The derived bidding and allocation policy is updated as users arrive without the need to have estimates of future user arrivals types and other quantities.

Chapter 4 was motivated by a somehow tangential topic to the DSP problem. A key input for the DSP problem and in general for the online advertising field is the probability of a user converting after being shown an ad, *i.e.*, making an action such as a click, purchase a product, fill a form, etc. Neural networks are typically to obtain estimates of conversions probabilities. Using user characteristics and other attributes as inputs for a pre-trained neural network, the neural network's output would be the probability of a user converting. In practice, a DSP would need to run the inference step of a neural network to estimate the likelihood of conversion for an incoming user. A DSP may need these probabilities in real-time for each ad that it wants to show to an incoming user. (The thesis author has collaborated on a couple of papers on neural networks architectures for conversion prediction [98, 99].) Since a DSP may need to estimate conversion probabilities up to hundreds of billions of times per day, these neural networks can not have expensive inference calls. In Chapter 4 we explore how to train neural networks promoting sparsity and potentially making inference calls cheaper to run. We propose training neural networks jointly using subgradient-descent, Frank-Wolfe, and Frank-Wolfe variants called in-face directions. The layers of a neural network that we would like to sparsify are trained using Frank-Wolf schemes. We can also use the hybrid training scheme mentioned above to promote other structural properties on a neural network layer, such as low-rank.

For the rest of this chapter, we first provide an overview of the optimization issues of interest in this thesis. Second, we comment on the actors of importance in the online advertising field. We make emphasis on the most relevant actors for the bidding and allocation problem faced by DSPs. Third, we provide a summary of the contributions of the chapters with links to the relevant papers. Finally, we provide a common notation for all chapters, though specific chapter notation is defined in each chapter too.

1.1 Optimization Issues of Interest

Calling $\mathbb{R}$ the set of real numbers, the optimization problems studied in this thesis are of the form:

$$\begin{aligned} \mathcal{P}(\theta) := & \min_{x \in \mathbb{R}^n} f(x;\theta) \\ & \text{subject to } x \in \mathcal{X}, \end{aligned} \tag{1.1}$$

where x is the decision variable which belongs to a domain $\mathcal{X} \subseteq \mathbb{R}^n$ and θ represents problem parameters which may not be known *a priori*. The objective is to minimize the function $f(\cdot) : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ over the feasible space $\mathcal{X}$. The set $\mathcal{X}$ is typically defined by a set of constraints. Depending on the problem structure, we may need to content ourselves with finding a local minimum or stationary point instead of a global optimum. Hidden in Problem (1.1) is the case when decisions need to be taken sequentially as information gets observed. In the latter case, we care about the difference between the value obtained by an adaptive algorithm with respect to a setting in which all information is observed from the beginning (this difference is called the regret of an algorithm). Issues of interest for this work are:

- Size of the optimization problem. Many problems of interest today are too big to be fully stored in memory. Even if they could be stored, many global solution techniques do not scale adequately for huge datasets. For example, to solve semi-definite programming problems [119] is common to use Newton type of methods which require inverting a Hessian matrix. Inverting a Hessian matrix scales cubically on the size of the variables (there is a plethora of methods on how to make this matrix inversion step less expensive [85, 48]). Methods particularly well suited for high dimensional problems are first-order methods. These methods require only a gradient or subgradient estimate per algorithm iteration to work [114], and they are typically used for neural network training and other machine learning problems [22]. The three main chapters of this thesis utilize first-order methods.

- Structure of the optimization problem. Some algorithms work only for certain problem structures. For example, the famous Simplex method, which is said to have started the Operations Research field in the 1940's [35], assumes $f(\cdot;\theta)$ to be a linear function and $\mathcal{X}$ to be composed only of linear constraints. Also, an algorithm performance may heavily depend, at least on theory, on the underlying problem structure. In the case of first-order methods, theoretical guarantees vary if a problem is convex or not. For convex problems, best possible worst-convergence rates has been achieved under different assumptions, such as differentiability, strong convexity, and others [114, 17]. For non-convex problems, except on known good cases [45, 44, 71], only convergence to a stationary point can be proven (though, some recent works argue that non-convexity may not be so bad [70]). In Chapters 2, 3, 4 we tackle complicated non-convex problems using a methodology which involves using first-order methods. In the case of Chapters 2 and 3, to solve a complicated non-convex optimization problem we used a simpler convex problem called the dual. A main task on these chapters was to prove that by "operating" on the dual problem we could obtain good performance guarantees and solutions for the original optimization problem.

- Stochasticity. Many optimization problems have an inherent stochastic nature, for example, a production problem may depend on future demand for which only estimates may be available. In other cases, stochasticity occurs by algorithm design. The best example of the latter occurs in machine learning, where typically only a batch of rows from a huge dataset are used at each iteration on training time. In the machine learning case the stochasticity appears because of the solution technique, while in the production example stochasticity appears as we can not "observe" the future. Even though different on nature, both of these cases can be thought as having $f(x;\theta) = \mathbb{E}_{w\sim\mathcal{W}}[f(x;\theta,w)]$, *i.e.*, the value of $f(x;\theta)$ is an expectation over a distribution which may not be known. (Here we simplify the analysis assuming that only the objective function may be subject to random quantities, but in a general case they may affect the constraints as well.) In Chapters 2 and 4 stochasticity appears as the first-order methods tried use batches of information to solve static optimization problems. There is no *a priori* limit on how many times we can sample batches of data to solve the optimization problems therein. Differently, Chapter 3 assumes that an agent receives information sequentially for $T > 0$ periods. At each period, a context vector is sampled i.i.d. from a possibly unknown distribution and the agent needs to take a decision adaptively. This sequential environment motivates the next topic.

- Static or sequential decision-making setup. As written, to solve Problem (1.1) we need to find only one decision variable 'x' regardless if a stochastic setting is used or not. Different is the case in which an agent receives objective functions in a sequential manner and needs to take decisions sequentially as well (we assume an unconstrained setup for now). In terms of Problem (1.1), we can represent the sequential setting mentioned above as having $f(x;\theta) = \sum_{t=1}^{T} f^t(x^t;\theta)$ and $x = (x^1, \dots, x^T)$ where $T > 0$ represents a total number of periods. A function $f^t(\cdot;\theta)$ is received on period $t \leq T$ and a decision x^t needs to be taken before observing $f^{t+1}(\cdot;\theta)$. In this setup an agent tries to derive an adaptive decision-making policy, *i.e.*, how to decide or choose x^{t+1} in function of its previous decisions $\{x^s\}_{s=1}^t$ and the set $\{f^s(x^s;\theta)\}_{s=1}^t$. This type of problems are studied in the online optimization literature [59] under different assumptions over the nature and type of the objective function arrivals. The problem becomes harder when the constraints are also observed in a sequential manner. For example, the case when $\mathcal{X}$ equals $\sum_{t=1}^{T} c^t(x^t) \leq a$ in which $c^t(\cdot)$ and $f^t(\cdot;\theta)$ are observed together and a is a known number or vector (more complicated forms for $\mathcal{X}$ could be used). Then, any adaptive algorithm needs also to consider if it will produce a feasible solution, *i.e.*, one that satisfies $\sum_{t=1}^{T} c^t(x^t;\theta) \leq a$, or how bad the worst constraint violation can be. An online optimization setting with both lower and upper constraints is studied in Chapter 3.

- Observability of problem parameters. It can be the case that there is a vector $\theta^* \in \Theta$ representing certain parameters of Problem (1.1), but θ^* may not be known *a priori*. Then, an agent may need to learn θ^* in a sequential manner as it takes decisions and observes new information. A trade-off between exploration and exploitation appears naturally. At iteration $t \in [T]$, the agent can either take a decision that would minimize Problem (1.1) with respect to what is learned/observed of θ^* so far, or take a decision that would help to better identify θ^*. These type of problems and trade-offs are core in the bandits literature, with methods such as Thompson-Sampling [112], Upper-Confidence Bound [6], or other simpler methods, such as epsilon-greedy approaches [78]. In the case of this thesis, we study a joint online learning and sequential decision-making setup in Chapter 3.

1.2 Main Actors in Online Advertising

Here we offer a simplified view of the main players in the online advertising market, with an emphasis towards those of interest for this thesis. A more complete overview of the field is done in [31]. The main players in online advertising are publishers, Supply-Side Platforms (SSPs), Demand-Side Platforms (DSPs), ad-exchanges, and users. Publishers generate content through their websites, apps, etc. Through this content publishers also generate inventory where ads can be shown (*e.g.*, banners, seconds on a video, etc.). A publisher's inventory is typically managed by an SSP. Ad-exchanges are in charge of performing real-time auctions to sell these inventories, *a.k.a.* ad spaces, to DSPs (and few big-enough advertisers which we ignore here). It can be the case that one entity/corporation may act as several roles. Alphabet is the most extreme example of the latter as it acts as an ad-exchange, a DSP, a SSP, and as a regulatory entity [125].

The entities of main interest for this thesis are DSPs. A DSP manages the budgets of hundreds to thousands of advertisers who set campaigns with it. Each advertiser precises a target audience, campaign's length, budget to spend and spending pacing, payment scheme, etc. Important DSPs are Criteo, MediaMath, Jampp, Verizon Media, and Amazon Ads (the latter has considerably bigger revenue than the rest, but many of their ads are shown in websites and properties owned by Amazon). Figure 1.1 shows a simplified scheme of how a DSP operates in the online advertising environment.

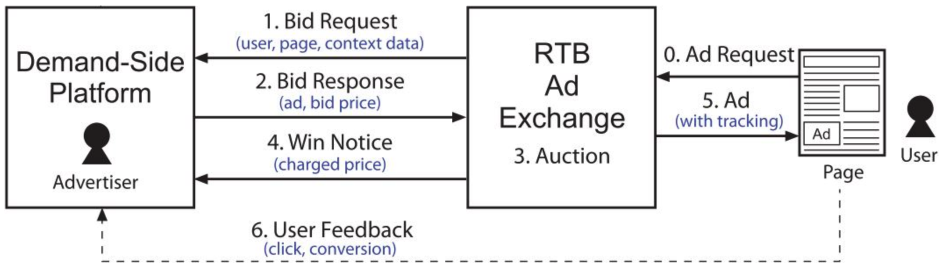


Figure 1.1: Modified figure from [129].

Following Figure 1.1, take the case in which a user enters a website, *e.g.*, `www.wired.com` which is owned by the Publisher Condé Nast. The website (or a SSP), informs an ad-exchange of the user arrival and the ad inventory. The ad-exchange then sets a real-time auction, informing all its DSP clients about the user arrival. DSPs call the latter event a 'bid request'; a bid request contains information about the user, website, and the type and characteristics of the real-time auction to be executed. Each DSP may submit at most one bid to the auction. Once the ad-exchange receives the bids, it executes the auction and the winner DSP, if any, shows an ad to the incoming user. The whole process from the time when the user arrives to an ad being shown takes, typically, 100 milliseconds or less. A medium DSP may receive hundreds of billions of bid requests per day.

Advertisers use conversions as their performance metric when setting a campaign with a DSP. Loosely speaking, conversions are actions an advertiser desires a user to perform after observing an ad. Examples are filling a form, making a click, buying a product, entering a website, or watching a video for a certain amount of seconds. Chapter 2 assumes a Cost-Per Action (CPA) payment scheme. In this scheme, an advertiser pays the DSP only when a conversion occurs. This makes the DSP, not the advertiser, to be the one leveraging the risk, as the DSP pays to ad-exchanges for each bid request it wins, regardless if conversions later occur or not.

Chapter 2 allows ad-exchanges to use arbitrary auction types, while in practice ad-exchanges today use either first or second-price auctions. In both first and second-price auctions, the winner of the auction is the highest bidder. The payment in first and second-price auctions is the highest and second-highest submitted bid, respectively. Second-price auctions generally were the only auction mechanism used by ad-exchanges until 2018 approximately. Today, both first and second-price auctions are used, with fast adoption of first-price auctions in the latter years. Both first and second-price auctions generate a censored data problem from a DSP point of view. A DSP observes the winning bid and the amount charged by ad-exchanges only for the auctions it won. When a DSP loses an auction, it only observes that its bid was smaller than the highest submitted bid.

1.3 Summary of Contributions

Here we divide the contributions of Chapters 2, 3, and 4 between "Bidding and Allocation for Demand-Side Platforms" and "Optimization Theory and Machine Learning". Chapter 2 makes contributions mostly on the former topic and Chapters 3 and 4 on the latter.

Bidding and Allocation for Demand-Side Platforms

- **Chapter 2:** Online advertising is an industry with an estimated $125 billion in global revenue in 2019 [102]. The online advertising environment offers advertising sellers the unique capability to deliver ads targeted to specific customer segments. A demand-side platform (DSP) serves as a type of intermediary between buyers and sellers of advertising inventory within targeted online advertising. A DSP typically manages the marketing campaigns of hundreds or thousands of advertisers. A campaign can be thought as of a plan for delivering advertisements, which defines a budget, pacing details, a target audience, and the ad to be shown. DSPs can charge its advertisers' clients using different schemes. Usual payment schemes are Cost-Per Mille/Clicks/Actions (CPM/CPC/CPA) in which advertisers are charged per thousands of ads shown to users or per clicks or actions. These clicks or actions are only counted as valid if users execute them after observing an advertiser's ad.

  We propose a static optimization model which tries to maximize the profit for a DSP while promoting an adequate budget spending for its advertisers clients in a

CPM/CPC/CPA scheme. The trade-off between maximizing the DSP profit and the budget spending is obtained by using a utility function on the budgets spending which offers a high degree of flexibility on how this trade-off is performed. Our model applies to any auction type by relying on machine learning techniques to predict conversions rates and auction outcomes. Our non-convex model is solved using a convex dual model used inside a primal-dual scheme. We show several no duality gap and convergence results of our primal-dual algorithm. Our no duality gap results show that the optimal bidding price to be submitted is the maximizer of a dual profit expression, which extend known results for second-price auctions. We show how to derive a policy that can be used by a DSP in a real operation from the solution of our optimization problem.

Optimization And Machine Learning Theory

- **Chapter 3:** Systems in which information arrives in a sequential manner and decisions need to be taken with slight delay are innocuous today. Examples of this are common in revenue management, such as pricing problems for airlines or hotels in which customers arrive sequentially or in online resource allocation problems. They also occur in online advertising as in the bidding and allocation problem for DSPs studied in Chapter 2. These general type of problems can be analyzed through the lens of online optimization. At each iteration or period, an agent observes a set of revenue and cost functions and needs to take a decision adaptively. The agent operates over a certain amount of periods, and the cost constraints may need to hold at the end of all periods (or their violation be bounded). Also, some problem parameters may need to be learned as the agent takes decisions and time progresses.

  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.

- **Chapter 4** When training neural networks, statistics like accuracy, f1-score, and Area Under the Curve (AUC) may not be adequate for all business needs. Depending on the use of a neural network, we may care about memory used to store it (think on mobile applications), training time, or resources needed to make predictions (inferencing). An example for the latter case occurs in conversion/click prediction in online advertising. An already trained neural network is used to predict if a user will click or perform an action of interest after observing an ad. The pre-trained neural network may make predictions up to hundreds of billions of times per day. Then, server costs become a concern. One way to reduce server costs is to add structure to the neural network such that making predictions, *i.e.*, running its inference step, consume fewer resources.

The Frank-Wolfe method and its extensions are well-suited for delivering solutions with desirable structural properties, such as sparsity or low-rank structure. We introduce a new variant of the Frank-Wolfe method that combines Frank-Wolfe steps and steepest descent steps, as well as a novel modification of the "Frank-Wolfe gap" to measure convergence in the non-convex case. We further extend this method to incorporate in-face directions for preserving structured solutions as well as block coordinate steps, and we demonstrate computational guarantees in terms of the modified Frank-Wolfe gap for all of these variants. We are particularly motivated by the application of this methodology to the training of neural networks with sparse properties, and we apply our block coordinate method to the problem of ℓ_1 regularized neural network training. We present the results of several numerical experiments on both artificial and real datasets demonstrating significant improvements of our method in training sparse neural networks.

Related Publications

- **Chapter 2**
  Main paper:

  1. Paul Grigas, Alfonso Lobos, Zheng Wen and Kuang-chih Lee, "Optimal Bidding, Allocation, and Budget Spending for a Demand-Side Platform with Generic Auctions", *submitted to Operations Research*, 2021

  Related papers:

  2. Alfonso Lobos, Paul Grigas, Zheng Wen and Kuang-chih Lee, "Optimal Bidding, Allocation and Budget Spending for a Demand Side Platform Under Many Auction Types", *AdKDD & TargetAd workshop of the conference Knowledge Data Discovery (KDD)*, 2018. Best student paper award.
  3. Paul Grigas, Alfonso Lobos, Zheng Wen and Kuang-chih Lee, "Profit Maximization for online advertising Demand-Side Platforms", *AdKDD & TargetAd workshop of the conference Knowledge Data Discovery (KDD)*, 2017.

- **Chapter 3**
  Main paper:

  1. Alfonso Lobos, Paul Grigas, Zheng Wen, "Joint Online Learning and Decision-making via Dual Mirror Descent", *International Conference on Machine Learning (ICML)*, 2021

- **Chapter 4**
  Main paper:
  1. Paul Grigas, Alfonso Lobos, Nathan Vermeersch, "Stochastic in-face frank-wolfe methods for non-convex optimization and sparse neural network training" [50].

1.4 Common Notation

The notation below is common for all chapters of this thesis. Chapters 2, 3, and 4 also use unique notations defined in each respective chapter.

For any n integer number we define $\mathbb{R}^n$, $\mathbb{R}^n_+$, and $\mathbb{R}^n_-$ as the sets of real, real non-negative, and real non-positive numbers of dimension n, respectively. For any two vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$, the usual dot product between vectors is defined as $x^T y := \sum_{i=1}^N x_i y_i$. We use calligraphic capital letters to denote sets. For any $x \in \mathbb{R}^n$, $[x]_+ := (\max\{x_1, 0\}, \dots, \max\{x_N, 0\})$ and use $\|\cdot\|$ to represent a norm operator. 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 $\|x\|_\infty = \max_{i \in \{1,\}} |x_i|$. We use $\mathbb{E}[\cdot]$ and $\mathbb{E}[\cdot|\cdot]$ to denote the mathematical expectations and conditional mathematical expectation operators. When not clear from the context, we use $\mathbb{E}_{\cdot \sim \cdot}[\cdot]$ and $\mathbb{E}_{\cdot \sim \cdot}[\cdot|\cdot]$ to indicate the random variable used to take the expectation and conditional expectation operators, respectively.

For a given set $\mathcal{S}$ and function $f(\cdot) : \mathcal{S} \to \mathbb{R}$, let $\arg\max_{x \in \mathcal{S}} f(x)$ denote the (possibly empty) set of maximizers of the function $f(\cdot)$ over the set S; likewise $\arg\min_{x \in \mathcal{S}} f(x)$ is defined similarly. If $f(\cdot) : \mathbb{R}^n \to \mathbb{R} \cup \{-\infty, +\infty\}$ is an extended real valued function, then we let $\mathrm{dom}(f(\cdot)) := \{x \in \mathbb{R}^n : f(x) \text{ is finite}\}$ denote its domain. Note that when $f(\cdot)$ is convex then $\mathrm{dom}(f(\cdot)) := \{x \in \mathbb{R}^n : f(x) < +\infty\}$, and when $f(\cdot)$ is concave then $\mathrm{dom}(f(\cdot)) := \{x \in \mathbb{R}^n : f(x) > -\infty\}$. Furthermore when $f(\cdot)$ is convex, for a given $x \in \mathrm{dom}(f(\cdot))$, $\partial f(x)$ denotes the set of subgradients of $f(\cdot)$ at x, i.e., the set of vectors g such that $f(y) \geq f(x) + g^\top (y - x)$ for all $y \in \mathrm{dom}(f(\cdot))$.

Chapter 2

Optimal Bidding, Allocation, and Budget Spending for a Demand-Side Platform with Generic Auctions

Due to the tremendous growth of the internet and various digital mediums, online advertising is now a substantially sized industry, with an estimated \$125 billion in global revenue in 2019 [102]. The online advertising environment offers advertising sellers the unique capability to deliver ads that are directly targeted to specific customer segments. Within targeted online advertising, a demand-side platform (DSP) serves as a type of intermediary between buyers and sellers of advertising inventory. A DSP typically manages the marketing campaigns of hundreds or thousands of advertisers, each of which simultaneously runs multiple campaigns. A campaign can be thought as of a plan for delivering advertisements, which defines a budget, pacing details, a target audience, and the ad to be shown. For example, Nike might run a campaign at half a million dollars per month that targets males between 18 and 35 years old who reside in California and visit sports websites. Advertisers measure the success of their campaigns in terms of the rate at which one or more desired actions occur, such as a click, conversion, or product purchase. In the CPA pricing model considered herein, advertisers pay the DSP only when this desired action occurs. This paper is primarily concerned with the problem of optimally managing a portfolio of campaigns, under a CPA pricing model, across multiple advertisers within a demand-side platform.

While demand-side platforms interact with and manage the campaigns of advertisers interested in buying advertising inventory, those publishers interested in selling typically utilize a supply-side platform (SSP) to manage the sales of their inventory. An ad exchange then serves as a connection between DSPs and SSPs responsible for allocating impressions supplied by SSPs to campaigns managed by DSPs. An impression constitutes a single instance in which an ad is served – for example a particular user browsing the homepage of ESPN – and contains information about the user as well as the publisher or domain in which the advertisement is to be shown. Once an ad exchange receives an impression opportunity, it initiates a real-time auction and all interested parties submit a single bid for the opportunity. Then, depending on the particular auction mechanism, the ad exchanges allocates the impression opportunity to a winner who is charged some amount possibly depending on all of the bids. This process is truly “real-time" and typically occurs in 100 milliseconds or less on average. Moreover, a large DSP may participate in hundreds of billions of auctions for different impressions opportunities per day.

As mentioned previously, when advertisers run campaigns they specify a budget that they desire to spend over a certain time horizon. In reality, a campaign will never exactly spend its budget amount and may underspend or overspend. Most existing algorithms in online advertising optimization, either through the use of budget constraints or heuristic approaches, heavily penalize overspending and place essentially zero penalty on underspending. In reality, underspending may be as undesirable or even more undesirable than overspending, and in this paper we consider a highly flexible utility function framework that allows campaigns to have greater control over their budget spending preferences.

Historically ad exchanges have usually used second-price auctions, also called Vickrey auctions, in which the highest bidder wins the auction and pays the amount of the second-highest bid submitted. Second-price auctions are truth revealing [77], i.e., is optimal for a bidder to bid their truthful valuation, although this property does not hold in the case of repeated auctions or when bidders have limited budgets [53, 13]. As of 2020, many ad exchanges including DoubleClick (Google), AppNexus, Rubicon, OpenX, Rubicon have increasingly used first-price auctions to sell impression opportunities. In any case, whenever an incoming impression opportunity arrives, a DSP is tasked with determining how much to bid in the corresponding auction, and traditional approaches either bid truthfully or otherwise make an assumption that the auction is second-price. In contrast, in this paper we directly model the interactions with ad exchanges and thus we can accommodate *arbitrary* auction types.

In this paper, we develop an optimization model and corresponding efficient algorithm to guide a DSP in determining how to bid for impressions and subsequently how to allocate acquired impressions to its portfolio of campaigns. Henceforth, we refer to “the DSP" as a generic demand-side platform implementing our proposed algorithm. We propose a formulation for the DSP to maximize its profit while ensuring that its advertisers are satisfied with the levels of budget spending across their campaigns. Due to the joint optimization over both bid price and impression allocation decisions, our model is non-convex. Nevertheless, we propose a dual formulation which is convex and can be efficiently solved with any subgradient based algorithm, and has strong theoretical guarantees and empirical performance.

Our contributions may be summarized as follows:

1. We propose a fixed time horizon optimization model for simultaneously determining both bid price and impression allocation decisions. Our model assumes a CPA pricing model, and its objective function allows the DSP to trade-off between maximizing its profit and maintaining acceptable levels of budget spending for its advertisers.

2. We propose the use of generic concave utility function over budget spending levels, such that the objective function of our optimization model is a combination of the

profit of the DSP and the total budget spending utility of its advertisers. This budget utility approach is highly flexible, includes standard budget constraints as a special case, and allows each advertiser to specify their preferences on how both underspending and overspending should be penalized.

3. We directly model the (stochastic) dynamics of how the DSP interacts with the ad exchanges. Such bid landscape models can be estimated directly from historical data (e.g., using machine learning), and as a consequence our modeling approach and results apply to arbitrary auction mechanisms (including, e.g., both first and second-price auctions).

4. Our optimization model is non-convex due to the joint optimization over both bidding and allocation decisions. Using Fenchel duality theory (see, e.g., [24]), we obtain a convex dual problem that can be efficiently solved with subgradient based algorithms – methods whose convergence rates and properties have been heavily studied (see, e.g., [95, 100] and the references therein). We also show how to efficiently obtain a primal solution and a corresponding real-time policy for the DSP directly from the dual solution. Our overall primal-dual algorithm is very general, and makes no assumptions about the specific functional form of the utility function over budget spending levels or the bid landscape models.

5. Under an intuitive “increasing marginal cost" condition, as well as under a more general condition concerning the uniqueness of bid prices, we demonstrate that there is zero duality gap in our formulation. We also extend this result, under the same conditions, to show convergence of our primal-dual algorithm to an optimal solution of the original primal problem.

6. We experimentally test our methodology on synthetic as well as a dataset from Criteo [39] (a real DSP). Our experimental results demonstrate that our methodology allows the DSP to effectively manage the trade off between maximizing its profit and satisfying the budget spending preferences of its advertisers. The synthetic data experiment employs both first and second-price auctions, while the dataset from Criteo assumes that second-price auctions are used. We show that our methodology dominates a greedy heuristic that is popular in practice (and is optimal in the case of infinite budgets).

Related Literature

Several prior works have considered topics related to optimization for a DSP although, to the best of our knowledge, none have proposed a model as general, flexible, and theoretically sound as ours. The works of [129] and [107] propose a Lagrangian relaxation approach to choose an adequate bidding strategy to maximize the profit of a DSP. Importantly, these works do not consider the simultaneous optimization of an allocation strategy as we do herein. Furthermore, they focus only on budget constraints, second-price auctions, and compared to our approach they either lack generic computational tractability (without strong assumptions) and/or strong theoretical guarantees (i.e., our zero duality gap results).

Assuming second-price auctions, the works of [14] and [10] study the special case of how to optimally bid on behalf of only a single advertiser. [10] obtain regret bounds under certain assumptions, and [14] show Nash equilibrium results. [8] study a different but related problem of determining how a single advertiser should target portfolios of impressions; they cast this problem as a multi-armed bandit problem with periodic budgets and propose an optimistic-robust learning algorithm. Returning to the problem of optimization for a DSP, and again assuming second-price auctions, [51] as well as [86] propose primal-dual schemes to find allocation and bidding strategies for a DSP. In addition to assuming second-price auctions, both papers rely on applying Lagrangian duality to budget constraints to formulate their dual problems. Thus, our model is more general in that we can accommodate arbitrary auction types as well as a more flexible utility function framework for accommodating the budget spending preferences of the advertisers. Note that our utility function framework generalizes budgets constraints, which have been considered in many prior works including [129, 51, 13, 14, 10] and [107], for example. Moreover, the model in our earlier paper [51] is an exact special case of the model we consider herein. There are no strong theoretical guarantees in [51], but it is important to note that the guarantees developed herein apply to that model. The model in [86] is substantially different from the model considered herein and can be regarded as a type of multidimensional knapsack problem; the authors prove a zero duality gap result and their proof is highly specialized to the particular structure of their multidimensional knapsack problem. [87] is an unpublished preliminary version of this paper.

Key inputs to our optimization model include forecasts of conversion rate and bid landscape information. Estimating the click through rate (CTR) or more generally the conversion through rate (CVR) is a central topic in online advertising, as advertisers typically see the amount of conversions as a key performance metric. Logistic regression is widely used to estimate CTR and CVR rates [27, 109], but many other methodologies have been used such as boosted trees [62], and Bayesian probit regression [49]. More recently, factorization machines [108] and derived methods [73] have gained popularity and importance and, for example, have been used by the winning teams on important CTR prediction challenges [7, 33] and in industry [72, 127, 54, 98]. Bid landscape forecasting refers to the estimation of the probability of winning an auction given that a bid was submitted [34], and the expected charge in case the auction is won. For both first and second-price auctions, estimating the bid landscape functions translate to estimating the cumulative distribution function (c.d.f.) of the highest competing bid [34, 129], a quantity that is also referred to as the market price. For DSPs, the estimation of bid landscapes is a biased or censored problem as a DSP observes the market price only when it wins an auction, i.e., only when its bid is higher than the market price [126, 130, 122]. Bid landscape forecasting is also essential for ad exchanges as it is typically used both for auction design and reserve price estimation [15, 28].

Optimization for a DSP – which involves jointly optimizing over both bidding and allocation decisions – is closely related to pure ad allocation problems, which have been primarily studied in the contexts of guaranteed contracts for display advertising and in the search space. In the guaranteed contracts setting, a publisher who owns inventory (e.g., a website) directly enters into contracts with advertisers. Each contract stipulates that the publisher will show a minimum amount of ads of a given advertiser to incoming impressions satisfying certain targeting constraints. The seminal work of [118] proposes a general model for guaranteed target display advertising as a quadratic network flow problem (see also the references therein for earlier approaches to ad allocation). [64] presents a more recent approach in the guaranteed contracts setting that also considers reach and frequency requirements. In the search space, Adwords is a preeminent problem in which a search provider has to allocate incoming search queries to advertisers who previously set budgets for different query types [92, 37]. The problem that a DSP faces and studied herein may be considered an extension of an ad allocation problem in which bidding strategies should also be derived. Recent extensions of ad allocation problems have considered other objectives/settings besides profitability and guaranteed contracts. For example, [16] propose a dynamic resource allocation scheme that emphasizes optimizing for allocation fairness, fully extracts the advertisers' budgets, and has strong performance guarantees. On the supply side, [15] and [28] studied how to optimally allocate impressions between guaranteed contracts and those to be sold in a real-time bidding environment. Appearing after our preliminary paper [87], [11] study a regularization approach to address fairness and under-delivery concerns in pure ad allocation problems. Their approach is similar in spirit to our utility function approach for promoting adequate budget spending. Finally, we refer readers to [31] for a more complete literature review of general topics in online advertising.

The problem of ad allocation may be considered as a special case of the broader topic of online resource allocation studied in the revenue management literature. Online resource allocation problems have been studied and applied in industries such as airlines, hotels, car rentals, and others [20, 23, 116]. Online resource allocation problems (and the closely related network revenue management problem) are commonly solved by searching for a policy maximizing the ex-ante expected revenue, which may be obtained using dynamic programming (e.g. [20, 23, 116]). Dynamic programming formulations tend to suffer from the "curse of dimensionality," and therefore approximations need to be done. Among others, [90] and [121] and the references therein present recent approaches based on approximate dynamic programming ideas. More closely related to this paper is the idea of formulating and solving a deterministic optimization problem that replaces random quantities with their expected values and then using the result of this optimization to determine an online policy. This idea at least dates to [113] and [124]. In the case of pure allocation problems, the deterministic optimization problem is simply a linear program and dual solutions can be used to derive bid price control techniques with theoretical guarantees [115, 116]. The effect of resolving such deterministic linear programs and using the resolves to update a policy, e.g., with model predictive control ideas has been studied theoretically as well [67, 32]. The main optimization problem proposed in this paper is also based on the idea of deterministic approximation, but it is important to emphasize that we obtain a much more difficult (non-convex) and general deterministic optimization problem due to the difficulites that arise from optimizing for both allocation and bidding decisions as well as the flexibility of our modeling framework. A key contribution of this paper is that we demonstrate how to efficiently solve this more general and complex deterministic optimization problem.

In the context of pure allocation problems in online advertising, it has been shown that seminal algorithms for the Adwords and online bipartite matching problems [92, 74] can be seen as solutions of randomized primal-dual schemes based on a deterministic optimization problem [37]. Here a bid price-price control technique assigns an opportunity cost to allocate an extra impression to an advertiser. This opportunity cost is usually related to the amount of inventory used by an advertiser. Examples of this occur in [92] for the Adwords problem, but also when deriving reserve prices for ad exchanges [15]. In the case of DSPs with budget constraints and second-price auctions, the dual variables change how valuable or costly bidding on behalf of an advertiser is [129, 51, 86, 107]. An intuition gained in this work is that when *arbitrary* auctions are used, the DSP's perception of the revenue value coming from different advertisers is linearly modified by the dual variables. These modified revenues help to derive the bidding prices used in this work, which are the maximizers of a well-defined profit function. In the case of the heavily studied second-price auctions, this profit function can be ignored as bidding truthfully is optimal [120].

2.1 Model Preliminaries and High-Level Overview

In this section, we first describe the basic assumptions and notation of our model. Then we discuss how the output of our optimization model (and, moreover, any feasible solution) may be used to derive an online/real-time bidding and allocation policy for the DSP. At a high-level, our general algorithm consists of the three parts: *(i)* parameter estimation/prediction, *(ii)* an offline optimization model, and *(iii)* an online policy that possibly incorporates model predictive control. We highlight these three aspects of our algorithm throughout this section.

Model Notation and Parameters

The set of campaigns managed by the DSP is denoted by $\mathcal{K}$, and the set of possible impression types is denoted by $\mathcal{I}$. For a given campaign $k \in \mathcal{K}$, the set of impression types that campaign k has elected to target is denoted by $\mathcal{I}_k$. Let $\mathcal{E} \subseteq \mathcal{I} \times \mathcal{K}$ denote the edges of an undirected bipartite graph between $\mathcal{I}$ and $\mathcal{K}$, whereby there is an edge $e = (i, k) \in \mathcal{E}$ whenever campaign k targets impression type i, i.e, $\mathcal{E} := \{(i, k) : k \in \mathcal{K}, i \in \mathcal{I}_k\}$. Let $\mathcal{K}_i := \{k \in \mathcal{K} : (i, k) \in \mathcal{E}\}$ denote the set of campaigns that are interested in impressions of type i. We sometimes slightly abuse the notation by referring to the k^{th} campaign (resp. the i^{th} impression type), whereby we implicitly assume a one-to-one correspondence between $\mathcal{K}$ and $\{1, \ldots, |\mathcal{K}|\}$.

Our optimization model depends on several different sets of parameters. Some of these parameters are completely known to the DSP (such as the budget of each campaign), whereas others (such as the probability of winning an auction) must be estimated based on historical data or by other means. We first describe the parameters that are known and then describe those which must be estimated.

Known Parameters:

- $\bar{b}_i$ = the maximum allowed bid for an impression of type i
- m_k = the target spending level (or budget) of campaign k
- ℓ_k = the amount campaign k is charged each time an action of interest occurs

Note that $\bar{b}_i$ is an upper limit on the allowable bid amount set either by the ad-exchange or the DSP, m_k is a target spending level set by the advertiser corresponding to campaign k, and ℓ_k is a price per action negotiated between the DSP and the corresponding advertiser.

Estimated Parameters and Functions:

- s_i = the expected number of arrivals of impressions of type i during the planning horizon
- θ_{ik} = the probability that an action of interest occurs when an ad of campaign k is shown to an impression of type i
- $r_{ik} := \ell_k \theta_{ik}$ = the expected revenue the DSP earns each time an ad of campaign k is shown to an impression of type i
- $\rho_i(b)$ = the probability of winning an impression of type i with a bid amount of $b \in [0, \bar{b}_i]$
- $\beta_i(b)$ = the expected amount that the DSP pays the ad exchange whenever the DSP wins an auction for an impression of type i with a submitted bid of $b \in [0, \bar{b}_i]$.

For each $i \in \mathcal{I}$, the functions $\rho_i(\cdot) : [0, \bar{b}_i] \to [0, 1]$ and $\beta_i(\cdot) : [0, \bar{b}_i] \to [0, \infty)$ encompass all of the information needed for our model regarding the auctions for impressions of type i. Note that $\rho_i(\cdot)$ takes as input a possible bid amount and returns a probability value, and $\beta_i(\cdot)$ takes as input a possible bid amount and returns a nonnegative expected payment amount. The functions $\rho_i(\cdot)$ and $\beta_i(\cdot)$ represent the DSP's forecast regarding the bid landscape for impression type i (see, e.g., [34]) and we refer to these functions as the bid landscape functions. Note that it is possible to have $\rho_i(0) > 0$ and/or $\rho_i(\bar{b}_i) < 1$. Although $\rho_i(0) > 0$ does not make much sense in practice, $\rho_i(\bar{b}_i) < 1$ allows for the possibility that the DSP may lose the auction even if they bid the maximum amount $\bar{b}_i$.

For convenience, we list all of the previously defined sets and parameters, as well as the decision variables that we discuss in Section 2.1, in Table A.1 that can be found in the Appendix.

Parameter Estimation Procedures

As mentioned, there are three types of parameters that the DSP needs to be estimate: *(i)* s_i, the expected number of impressions of type i, *(ii)* θ_{ik}, the conversion rate or action probability for each pair $(i, k) \in \mathcal{E}$, and *(iii)* the bid landscape functions $\rho_i(\cdot)$ and $\beta_i(\cdot)$. The

s_i parameters are relatively simple to estimate based on historical data as well as timing information such as whether the planning horizon intersects with a weekend or holiday. As mentioned in Section 2, estimating the conversion rate parameters θ_{ik} is usually done with machine learning models such as logistic regression or factorization machines (see, e.g., [27, 109, 73]), and a considerable amount of engineering effort is usually employed to develop high quality models. Estimation of the bid landscape functions $\rho_i(\cdot)$ and $\beta_i(\cdot)$ (such as in [34]) usually involves some combination of using knowledge of the structure of the specific auction as well as learning from historical data (which is typically censored). Examples 2.1.1 and 2.1.2 below discuss how two widely popular auction types – first and second-price auctions – affect the structure and estimation of the bid landscape functions.

Example 2.1.1 (First-Price Auction). In a first-price auction, the winner is the bidder who placed the highest bid and the winner is required to pay the bid that they submitted. This implies that the amount that the DSP is required to pay the ad exchange whenever the DSP wins an auction for an impression of type i is deterministically equal to the amount that they bid, i.e., $\beta_i(b) = b$. Moreover, whether the DSP wins the auction is completely determined by the highest competing bid, which we can model as a non-negative random variable C_i. Using this notation, we have that $\rho_i(b) = \mathbb{P}(C_i < b)$. (In the case of a tie, we assume that no one wins the auction). Note that, in the case of a first-price auction, there is no need to estimate $\beta_i(\cdot)$ as this function is completely determined and estimation of $\rho_i(\cdot)$ reduces to the problem of estimating the CDF of C_i. We can also naturally extend this case to the case of a *scaled* first-price auction, whereby $\beta_i(b) = \alpha b$ for some $\alpha \in (0, 1]$. In the last two years, first-price auctions have gained importance in practice as several prominent ad-exchanges including Google's DoubleClick, OpenX, AppNexus, and Rubicon have officially transitioned from second to first-price auctions.

Example 2.1.2 (Second-Price Auction). In a second-price auction, the winner is again the bidder who placed the highest bid, however now the winner is required to pay an amount equal to the bid of the *second* highest bidder. Again if C_i is a random variable representing the highest competing bid, then we have that $\rho_i(b) = \mathbb{P}(C_i < b)$. Now the amount that the DSP is required to pay the ad exchange whenever the DSP wins an auction for an impression of type i is a random variable with mean $\beta_i(b) = \mathbb{E}[C_i \mid C_i < b]$. Note that if C_i is a continuous random variable with density $f_{C_i}(\cdot)$, then it holds that $\beta_i(b) = \frac{1}{\rho_i(b)} \int_0^b z f_{C_i}(z) dz$. In the case of a second-price auction, the DSP needs to estimate both $\rho_i(\cdot)$ and $\beta_i(\cdot)$, which both reduce to estimating properties of the distribution of C_i. Second-price auctions are a common assumption in the literature on optimization related to DSPs and ad-exchanges, such as in [129] and [109], as they were the industry standard until 2017 when several ad-exchanges started to transition from second to first-price auctions. Second-price auctions have the desirable truth-revealing property which makes it optimal for advertisers to bid their real valuations. However, it is important to emphasize that this truth revealing property is no longer valid when bidders have budget constraints and/or in the case of repeated auctions.

Decision Variables and Real Time Dynamics

In this section, we describe the decision variables of our model and discuss how a feasible solution of our model leads to a real-time bidding and allocation policy for the DSP. Let us first review the basic flow of events in the model. When an impression of type $i \in \mathcal{I}$ is submitted to the ad exchange, a real-time auction is held for which the DSP has an opportunity to bid. Thus the DSP has an opportunity to make two operational decisions related to each real-time auction: *(i)* how to select a campaign $k \in \mathcal{K}$ to bid on behalf of in the auction, and *(ii)* how to set the corresponding bid amount b_{ik}. If the DSP wins the auction on behalf of campaign k, then the DSP must pay a certain amount to the ad-exchange and an ad from campaign k is displayed. Finally, the advertiser corresponding to campaign k is charged only if an action of interest occurs, such as when a user clicks on the ad. We define decision variable to model these two operational decisions as follows.
Decision Variables:

- x_{ik} = the probability of bidding on behalf of campaign k when an impression of type i arrives
- b_{ik} = the corresponding bid amount to be submitted for an impression of type i on behalf of campaign k

Interpreted differently, x_{ik} represents a proportional allocation, i.e., the long run average fraction of impressions of type i that are allocated to campaign k. In this application domain, proportional allocation variables are a reasonable modeling decision, as opposed to integer variables, due to the vast number of impressions that arrive during the planning horizon. Note that b_{ik} represents the bid that the DSP submits to an auction for impression type i *conditional* on the fact that the DSP has selected campaign k for the auction. Related approaches (e.g., as in [30]) also use bids to rank advertisers – in our approach, the selection of which campaign to bid on behalf of is completely captured by the x_{ik} decision variables and thus the b_{ik} decision variables only determine the actual bid price decisions. Let $\mathbf{x}, \mathbf{b} \in \mathbb{R}^{|\mathcal{E}|}$ denote vectors of these quantities, which will represent decision variables in our model. We defer the discussion of the offline optimization problem (2.1) used to solve for $\mathbf{x}$ and $\mathbf{b}$ to Section 2.2, as developing and studying the properties of problem (2.1) constitutes a major part of our contribution.

Policy 1 below summarizes the real-time bidding and allocation policy defined by a particular specification of the allocation and bidding variables ($\mathbf{x}$,$\mathbf{b}$), and we now describe the flow of events in more detail. Each time an impression of type $i \in \mathcal{I}$ arrives, the DSP first decides if it will bid or not. If the DSP decides to bid, then the campaign for which the DSP will bid on behalf of needs to be determined. In our model, both of these decisions are determined jointly in a randomized fashion based on the vector of allocation probabilities $\mathbf{x}$, whereby campaign $k \in \mathcal{K}_i$ is selected with probability x_{ik} and the option of not bidding at all is selected with probability $1 - \sum_{k \in \mathcal{K}_i} x_{ik}$. Given the selection of campaign k, the DSP then enters a bid amount equal to b_{ik} and wins the auction with probability $\rho_i(b_{ik})$. If the

DSP wins the auction, then the impression is acquired and the DSP pays the ad-exchange a certain amount that is modeled as a random variable equal to $\beta_i(b_{ik})$ in expectation. Finally, the advertisement of campaign k is shown and if the action of interest occurs, which happens with probability θ_{ik}, then campaign k is charged an amount ℓ_k by the DSP, which results in the budget of campaign k being depleted by ℓ_k while the DSP earns a revenue of ℓ_k.

Policy 1 Online Policy Implied by $(\mathbf{x},\mathbf{b})$

```
Input: Allocation and bidding variables $(\mathbf{x}, \mathbf{b})$ and a new impression arrival of type $i \in \mathcal{I}$.

1. Sample a campaign $\tilde{k} \in \mathcal{K}_i$ according to the probabilities $x_{ik}$, where $1 - \sum_{k \in \mathcal{K}_i} x_{ik}$ captures the probability of not participating in the auction.
2. Enter bid price $b_{i\tilde{k}}$. If the auction is won, then pay the ad exchange an amount equal to $\beta_i(b_{i\tilde{k}})$ in expectation. If the auction is not won, then break.
3. Show an ad for campaign $\tilde{k}$. If an action of interest happens, then deduct $\ell_{\tilde{k}}$ from the budget of campaign $\tilde{k}$ and earn revenue $\ell_{\tilde{k}}$.
```

As mentioned previously, our overall algorithm consists of three major components: parameter estimation as discussed in Section 2.1, offline optimization with problem (2.1), and the online Policy 1 applied with an optimal solution of (2.1). It is important to emphasize that these three components of our proposed algorithm can all be wrapped inside a model predictive control (MPC) framework. The simplest way to use an MPC scheme in our setting is to fix a certain time interval, e.g., one hour. At each passing of this time interval, the DSP should first re-estimate the relevant parameters based on the addition of new data that has been collected. Then, the DSP should re-solve the optimization problem (2.1) with the updated parameters and correspondingly update the bidding and allocation variables used in Policy 1. Moreover, the DSP should also employ this re-estimation and re-solve procedure whenever an important change in the state of the system occurs, such as when a campaign completely depletes its budget, or new advertisers begin working with the DSP, etc. This MPC scheme can be viewed as a heuristic for the complex control problem that the DSP faces. As our primary contributions concern the offline optimization model used in this MPC scheme, further investigation of the theoretical and/or practical performance of this MPC scheme in a control context is left for future work.

Additional Notation and Quantities of Interest

Given values of x_{ik} and b_{ik}, recall that x_{ik} is the probability that a new impression of type i is allocated to campaign $k \in \mathcal{K}_i$, $\rho_i(b_{ik})$ is the probability that the auction is won, and r_{ik} is expected amount that campaign k spends in the event that the auction is won. Therefore, for all $(i,k) \in \mathcal{E}$, we define $v_{ik}(x_{ik}, b_{ik}) := r_{ik}\rho_i(b_{ik})s_i x_{ik}$, which may be interpreted as the expected amount that campaign k spends on impressions of type i during the time horizon as a function of x_{ik} and b_{ik}. Furthermore, in the event that the auction is won, $\beta_i(b_{ik})$ is the expected amount that the DSP pays to the ad exchange. Therefore, for all $(i,k) \in \mathcal{E}$, we similarly define $\pi_{ik}(x_{ik}, b_{ik}) := [r_{ik} - \beta_i(b_{ik})]\rho_i(b_{ik})s_i x_{ik}$, which may be interpreted as the expected profit that the DSP earns from campaign k for impressions of type i. Indeed, note that $[r_{ik} - \beta_i(b_{ik})]\rho_i(b_{ik})$ is the expected profit the DSP earns when the DSP bids an amount b_{ik} for an impression of type i on behalf of campaign k. Presuming that the amount that the DSP earns per bid is independent of the number of bids, Wald's equation implies that the the total expected profit that the DSP earns from campaign k for impressions of type i is $[r_{ik} - \beta_i(b_{ik})]\rho_i(b_{ik})s_i x_{ik}$. A similar argument verifies the validity of the definition of $v_{ik}(x_{ik}, b_{ik})$.

To further simplify notation, define $\mathbf{x}_i \in \mathbb{R}^{|\mathcal{K}_i|}$ and $\mathbf{b}_i \in \mathbb{R}^{|\mathcal{K}_i|}$ as subvectors of the allocation vector $\mathbf{x} \in \mathbb{R}^{|\mathcal{E}|}$ and the bidding vector $\mathbf{b} \in \mathbb{R}^{|\mathcal{E}|}$, respectively, consisting of the variables associated with impression type $i \in \mathcal{I}$. Let $\mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i) \in \mathbb{R}^{|\mathcal{K}|}$ denote a vector of the expected spending values associated with impression type i, whereby the k^{th} component of $\mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$ is equal to $v_{ik}(x_{ik}, b_{ik})$ if $(i,k) \in \mathcal{E}$ and 0 otherwise. Also, let $\mathbf{v}(\mathbf{x}, \mathbf{b}) \in \mathbb{R}^{|\mathcal{K}|}$ denote a vector of expected total spending values, whereby $\mathbf{v}(\mathbf{x}, \mathbf{b}) := \sum_{i \in \mathcal{I}} \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$. Likewise, let $\pi_i(\mathbf{x}_i, \mathbf{b}_i) := \sum_{k \in \mathcal{K}_i} \pi_{ik}(x_{ik}, b_{ik})$ denote the total expected profit earned from impressions of type i and let $\pi(\mathbf{x}, \mathbf{b}) := \sum_{i \in \mathcal{I}} \pi_i(\mathbf{x}_i, \mathbf{b}_i)$ denote the overall total expected profit.

We use boldfaced font, e.g., $\mathbf{x}$, $\boldsymbol{\lambda}$, and $\mathbf{0}$ (which denotes the vector of all 0s) to denote vectors, regular font, e.g., x_{ik} and λ_k to denote components of vectors, and calligraphic font, e.g., $\mathcal{K}$ and $\mathcal{S}$, to denote sets. For a given set $\mathcal{S}$ and function $f(\cdot) : \mathcal{S} \to \mathbb{R}$, let $\arg\max_{x \in \mathcal{S}} f(x)$ denote the (possibly empty) set of maximizers of the function $f(\cdot)$ over the set S; likewise $\arg\min_{x \in \mathcal{S}} f(x)$ is defined similarly. If $f(\cdot) : \mathbb{R}^n \to \mathbb{R} \cup \{-\infty, +\infty\}$ is an extended real valued function, then we let $\mathrm{dom}(f(\cdot)) := \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) \text{ is finite}\}$ denote its domain. Note that when $f(\cdot)$ is convex then $\mathrm{dom}(f(\cdot)) := \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) < +\infty\}$, and when $f(\cdot)$ is concave then $\mathrm{dom}(f(\cdot)) := \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) > -\infty\}$. Furthermore when $f(\cdot)$ is convex, for a given $\mathbf{x} \in \mathrm{dom}(f(\cdot))$, $\partial f(\mathbf{x})$ denotes the set of subgradients of $f(\cdot)$ at $\mathbf{x}$, i.e., the set of vectors $\mathbf{g}$ such that $f(\mathbf{y}) \geq f(\mathbf{x}) + \mathbf{g}^\top(\mathbf{y} - \mathbf{x})$ for all $\mathbf{y} \in \mathrm{dom}(f(\cdot))$. For any function $f(\cdot) : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, the convex conjugate of $f(\cdot)$, denoted by $f^*(\cdot)$, is defined by $f^*(\boldsymbol{\lambda}) := \sup_{\mathbf{x} \in \mathbb{R}^n} \left\{\boldsymbol{\lambda}^\top \mathbf{x} - f(\mathbf{x})\right\}$. For a given finite set $\mathcal{S}$, let $|\mathcal{S}|$ denote its cardinality. For a given set $\mathcal{S} \subseteq \mathbb{R}^n$, let $\mathrm{int}(S)$ denote its interior, let $\mathrm{conv}(\mathcal{S})$ denote the closure of the convex hull of $\mathcal{S}$. For $a \in \mathbb{R}$ define $[a]_+ := \max\{a, 0\}$, and for $a, b, c \in \mathbb{R}$ with $b \leq c$ define $\mathrm{clip}(a; [b,c]) := \min\{\max\{a, b\}, c\}$. Finally, we use $'$ to denote a scalar derivative in the right context.

2.2 Optimization Formulation: Primal and Dual Problems

We now introduce our main optimization problem of interest, which models the DSP's goal of maximizing profit while maintaining a desired spending level of its advertisers' budgets, as well as our corresponding two phase primal-dual procedure. A central ingredient of our proposed formulation is the use of a "utility" function to model the budget spending preferences of the advertisers. Formally, the budget spending utility function $u(\cdot) : \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R} \cup \{-\infty\}$ is an extended real valued concave function that receives the vector of the expected spending of the campaigns $\mathbf{v}(\mathbf{x}, \mathbf{b})$ as its input and outputs a value quantifying how satisfied the advertisers are with the spending levels $\mathbf{v}(\mathbf{x}, \mathbf{b})$. In addition to concavity, we also make a few mild technical assumptions about the budget spending utility function, which are summarized below in Assumption 2.2.1.

Assumption 2.2.1. *The advertisers' budget spending utility function* $u(\cdot) : \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R} \cup \{-\infty\}$ *is a concave function satisfying:*

(i) $\mathrm{dom}(u(\cdot)) = \{\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|} : u(\mathbf{v}) > -\infty\}$ *is a non-empty closed set, and*

(ii) $u(\cdot)$ *is continuous on* $\mathrm{dom}(u(\cdot))$.

Notice that part *(i)* of Assumption 2.2.1 implies that that $u(\cdot)$ is proper (meaning that $u(\mathbf{v}) > -\infty$ for at least one vector $\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|}$) and closed (meaning that the hypograph of $u(\cdot)$ is a closed set). In addition, throughout the rest of the paper we assume that the bid landscape functions are also continuous, which is formalized below in Assumption 2.2.2.

Assumption 2.2.2. *For each* $i \in \mathcal{I}$, *the bid landscape functions* $\rho_i(\cdot) : [0, \bar{b}_i] \to [0, 1]$ *and* $\beta_i(\cdot) : [0, \bar{b}_i] \to [0, \infty)$ *are continuous on* $[0, \bar{b}_i]$.

Notice that Assumption 2.2.2 implies that $\mathbf{v}(\mathbf{x}, \mathbf{b})$ and $\pi(\mathbf{x}, \mathbf{b})$ are continuous in $(\mathbf{x}, \mathbf{b})$ on the feasible domain of $(\mathbf{x}, \mathbf{b})$. Strictly speaking, Assumption 2.2.2 is not needed to develop the primal-dual procedure in this section. However, Assumption 2.2.2 simplifies the discussion and is also necessary for the theoretical results that we later develop in Section 2.3.

Problem (2.1) presented below is our main optimization problem of interest, which maximizes the sum of the total expected profit of the DSP and the total utility of the advertisers, subject to feasibility constraints on the allocation variables $\mathbf{x}$ and the bidding variables $\mathbf{b}$. Thus problem (2.1) directly trades-off between profitability for the DSP and the budget spending preferences of the advertisers, and in Section 2.2 we present several examples of utility functions $u(\cdot)$ that highlight the modeling flexibility of our framework.

$$\begin{array}{rlll} F^* := & \underset{\mathbf{x},\mathbf{b}}{\text{maximize}} & \displaystyle\sum_{(i,k)\in\mathcal{E}} [r_{ik} - \beta_i(b_{ik})]\rho_i(b_{ik})s_i x_{ik} + & u\left(\begin{bmatrix} \displaystyle\sum_{i\in\mathcal{I}_1} r_{i1}\rho_i(b_{i1})s_i x_{i1} \\ \vdots \\ \displaystyle\sum_{i\in\mathcal{I}_{|\mathcal{K}|}} r_{i|\mathcal{K}|}\rho_i(b_{i|\mathcal{K}|})s_i x_{i|\mathcal{K}|} \end{bmatrix}\right) \\ & \text{subject to} & \displaystyle\sum_{k\in\mathcal{K}_i} x_{ik} \leq 1 & \text{for all } i \in \mathcal{I} \\ & & 0 \leq b_{ik} \leq \bar{b}_i & \text{for all } (i,k)\in\mathcal{E} \\ & & x_{ik} \geq 0 & \text{for all } (i,k)\in\mathcal{E} \end{array} \tag{2.1}$$

Our high level approach for solving problem (2.1) is based on a two phase procedure. In the first phase, we construct a suitable dual of (2.1), which turns out to be a convex optimization problem that can be efficiently solved with subgradient based algorithms. A solution of the dual problem naturally suggests a way to set the bid prices $\mathbf{b}$. In the second phase, we set the bid prices using the previously computed dual solution and then we solve a convex optimization problem that results when $\mathbf{b}$ is fixed in order to recover allocation probabilities $\mathbf{x}$.

Towards developing our two phase procedure, let us start by deriving the dual problem of (2.1). For each $i \in \mathcal{I}$, define the feasible set of $\mathbf{x}_i$ by $\mathcal{X}_i := \{\mathbf{x}_i \in \mathbb{R}^{|\mathcal{K}_i|} : \sum_{k\in\mathcal{K}_i} x_{ik} \leq 1, \text{ and } x_{ik} \geq 0 \text{ for all } k \in \mathcal{K}_i\}$, the feasible set of $\mathbf{b}_i$ by $\mathcal{B}_i := [0, \bar{b}_i]^{|\mathcal{K}_i|}$, and the feasible set of $(\mathbf{x}_i, \mathbf{b}_i)$ by $\mathcal{S}_i := \mathcal{X}_i \times \mathcal{B}_i$. Then the feasible region of (2.1), denoted by $\mathcal{S}$, decomposes as $\mathcal{S} = \mathcal{X} \times \mathcal{B}$, where $\mathcal{X} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_{|\mathcal{I}|}$ is the feasible set of $\mathbf{x}$ and $\mathcal{B} := \mathcal{B}_1 \times \cdots \times \mathcal{B}_{|\mathcal{I}|}$ is the feasible set of $\mathbf{b}$. In a slight abuse of notation, we may also remark that $\mathcal{S}$ decomposes across the impression types as $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_{|\mathcal{I}|}$. Using the notation defined in Section 2.1, in particular the definition of the total profit $\pi(\mathbf{x}, \mathbf{b})$ and the vector of expected spending values $\mathbf{v}(\mathbf{x}, \mathbf{b})$, we can more compactly write problem (2.1) as:

$$F^* = \max_{(\mathbf{x},\mathbf{b})\in\mathcal{S}} \{F(\mathbf{x}, \mathbf{b}) := \pi(\mathbf{x}, \mathbf{b}) + u(\mathbf{v}(\mathbf{x}, \mathbf{b}))\} . \tag{2.2}$$

Let $\mathcal{V} := \{\mathbf{v}(\mathbf{x}, \mathbf{b}) : (\mathbf{x}, \mathbf{b}) \in \mathcal{S}\}$ denote the set of feasible expected spending values, and note that $\mathcal{V}$ is a compact set since $\mathcal{S}$ is compact and $\mathbf{v}(\cdot, \cdot)$ is a continuous function. We say that problem (2.2) is feasible whenever $\text{dom}(u(\cdot)) \cap \mathcal{V}$ is non-empty. Proposition 2.2.1 justifies the use of the max (instead of sup) in (2.2) above by demonstrating that F^* is finite and attained whenever problem (2.2) is feasible.

Proposition 2.2.1. *Suppose that problem* (2.2) *is feasible, i.e., it holds that* $\mathrm{dom}(u(\cdot)) \cap \mathcal{V} \neq \emptyset$*. Then, the optimal value* F^* *of* (2.2) *is finite and attained.*

The proof of Proposition 2.2.1 as well as all omitted proofs for this section are contained in Appendix A.2. Notice that problem (2.1) is quite general and possibly *non-convex* due to the generic form of $\rho_i(\cdot)$ and $\beta_i(\cdot)$ as well as the multiplications of these functions with the allocation variables $\mathbf{x}$. On the other hand, as we now demonstrate, the concavity of the utility function $u(\cdot)$ enables us to construct a dual problem that is *always* convex and amenable to efficient solution methods. Since $-u(\cdot)$ is a convex function, it is useful to consider its convex conjugate $p(\cdot) := (-u(\cdot))^*$. Recall that $p(\cdot) : \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R} \cup \{+\infty\}$ is defined by $p(\boldsymbol{\lambda}) := \sup_{\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|}} \left\{\boldsymbol{\lambda}^\top \mathbf{v} + u(\mathbf{v})\right\}$ for any $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$. Since $-u(\cdot)$ is a proper and closed convex function, the Fenchel-Moreau Theorem (see, e.g., [24] p. 91) ensures that $-u(\cdot) = p(\cdot)^*$ and hence

$$u(\mathbf{v}) = \inf_{\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}} \left\{-\boldsymbol{\lambda}^\top \mathbf{v} + p(\boldsymbol{\lambda})\right\} \text{ for all } \mathbf{v} \in \mathbb{R}^{|\mathcal{K}|} \ .$$

Now we can simply substitute the above equality into (2.2) to obtain:

$$F^* \ = \ \max_{(\mathbf{x},\mathbf{b}) \in \mathcal{S}} \left\{ \pi(\mathbf{x},\mathbf{b}) \ + \ \inf_{\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}} \left\{-\boldsymbol{\lambda}^\top \mathbf{v}(\mathbf{x},\mathbf{b}) + p(\boldsymbol{\lambda})\right\} \right\} \ . \tag{2.3}$$

Let us define the function $q(\cdot) : \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R}$ by:

$$q(\boldsymbol{\lambda}) := \max_{(\mathbf{x},\mathbf{b}) \in \mathcal{S}} \left\{\pi(\mathbf{x},\mathbf{b}) - \boldsymbol{\lambda}^\top \mathbf{v}(\mathbf{x},\mathbf{b})\right\} \ . \tag{2.4}$$

The Weierstrass Theorem ensures that $\mathrm{dom}(q(\cdot)) = \mathbb{R}^{|\mathcal{K}|}$ and that the maximum is attained above. Note also that $q(\cdot)$ is a convex function since it is a maximum of linear functions of $\boldsymbol{\lambda}$. Furthermore, $q(\cdot)$ is non-negative everywhere since $(\mathbf{0},\mathbf{0}) \in \mathcal{S}$, and $\pi(\mathbf{0},\mathbf{0}) = 0$ and $\mathbf{v}(\mathbf{0},\mathbf{0}) = \mathbf{0}$. Recalling the notation defined in Section 2.1, for each $i \in \mathcal{I}$, let us also define the function $q_i(\cdot) : \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R}$ by $q_i(\boldsymbol{\lambda}) := \max_{(\mathbf{x}_i,\mathbf{b}_i) \in \mathcal{S}_i} \left\{\pi_i(\mathbf{x}_i,\mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i,\mathbf{b}_i)\right\}$, and likewise note that $\mathrm{dom}(q_i(\cdot)) = \mathbb{R}^{|\mathcal{K}|}$, $q_i(\cdot)$ is convex, and $q_i(\cdot) \geq 0$. Then it holds that:

$$q(\boldsymbol{\lambda}) \ = \ \max_{(\mathbf{x},\mathbf{b}) \in \mathcal{S}_1 \times \cdots \times \mathcal{S}_{|\mathcal{I}|}} \left\{ \sum_{i \in \mathcal{I}} [\pi_i(\mathbf{x}_i,\mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i,\mathbf{b}_i)] \right\} \ = \ \sum_{i \in \mathcal{I}} q_i(\boldsymbol{\lambda}) \ , \tag{2.5}$$

hence $q(\cdot)$ is decomposable across the impression types.

Furthermore, let $Q(\cdot) := q(\cdot) + p(\cdot)$ denote the dual function, which is also convex. Then the dual problem of (2.1), which is obtained by exchanging the supremum and infimum in (2.3), is

$$\begin{aligned} Q^* := \quad & \underset{\boldsymbol{\lambda}}{\text{minimize}} & & Q(\boldsymbol{\lambda}) \\ & \text{subject to} & & \boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|} \ , \end{aligned} \tag{2.6}$$

which is a convex optimization problem. Notice that the dual objective function decomposes as the sum of two terms: *(i)* $q(\cdot)$ which depends on the particular form of the functions $\rho_i(\cdot), \beta_i(\cdot)$ and other problem parameters but *does not* depend on the utility function $u(\cdot)$, and *(ii)* $p(\cdot)$ which directly depends on the utility function $u(\cdot)$ as it is the convex conjugate of $-u(\cdot)$. Recall the definition of the primal function $F(\mathbf{x}, \mathbf{b}) := \pi(\mathbf{x}, \mathbf{b}) + u\left(\mathbf{v}(\mathbf{x}, \mathbf{b})\right)$. Basic weak duality immediately yields:

$$Q(\boldsymbol{\lambda}) \ \geq \ Q^* \ \geq \ F^* \ \geq \ F(\mathbf{x}, \mathbf{b}) \ \text{ for all } \boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|} \text{ and } (\mathbf{x}, \mathbf{b}) \in \mathcal{S} \ .$$

Also, since $u(\cdot)$ is proper by Assumption 2.2.1, we have that $p(\cdot)$ is proper (i.e., $\operatorname{dom}(p(\cdot))$ is non-empty) and therefore Q^* is finite (although possibly not attained) whenever (2.1) is feasible. Later, in Section 2.3, we study conditions under which there is zero duality gap above, i.e., it holds that $Q^* = F^*$.

Examples of Utility Functions.

Let us now give several examples of utility functions to demonstrate the modeling flexibility of our approach. For each example, we define the utility function $u(\cdot)$ and discuss the conjugate function $p(\cdot)$ that appears in the dual problem (2.6). All of our examples are *separable* across the different campaigns, i.e., it holds that $u(\mathbf{v}) = \sum_{k \in \mathcal{K}} u_k(v_k)$ where, for each campaign $k \in \mathcal{K}$, $u_k(\cdot) : \mathbb{R} \rightarrow \mathbb{R} \cup \{-\infty\}$ is a scalar concave function (that also satisfies Assumption 2.2.1) representing the utility of campaign k as a function of its expected spending level v_k. This separable structure is quite natural as we do not expect the spending of other advertisers to affect a given advertiser's utility and it allows each campaign to use their own preferred type of utility function. However, this is not a requirement as it is also possible for our framework to accommodate more general concave utility functions $u(\cdot)$ that do incorporate such interactions. Note that the separable structure of $u(\cdot)$ also implies that $p(\cdot)$ is similarly separable, i.e., it holds that $p(\boldsymbol{\lambda}) = \sum_{k \in \mathcal{K}} p_k(\lambda_k)$ where $p_k(\cdot) : \mathbb{R} \rightarrow \mathbb{R} \cup \{+\infty\}$ is the convex conjugate of $-u_k(\cdot)$. In the remainder of this section, we present examples of scalar utility functions $u_k(\cdot)$, as well as each corresponding conjugate function $p_k(\cdot)$. The derivations of the corresponding conjugate functions $p_k(\cdot)$ may be found in Appendix A.3.

Example 2.2.1 (Budget Constraint). Consider the case where campaign k is only concerned with ensuring that its expected spending level does not exceed the budget value m_k. This is the well-studied case of a budget constraint, which has been examined in several different contexts in the advertising optimization literature including, for example, in [30, 15, 129, 83], as well as [51]. In this case, the utility and conjugate function pair is given by:

$$u_k(v_k) = \begin{cases} 0 & \text{if } v_k \leq m_k \\ -\infty & \text{o/w} \end{cases} , \quad p_k(\lambda_k) = \begin{cases} m_k \lambda_k & \text{if } \lambda_k \geq 0 \\ +\infty & \text{o/w} \end{cases} .$$

Example 2.2.2 (Target Spending). Notice that the previous example does not directly encourage the DSP to spend the budget of campaign k and may in fact lead to budget spending values that are considerably smaller than the target budget value m_k. As mentioned previously, poor or insufficient budget spending may motivate advertising companies to stop working with the DSP and thus hurt the DSP's business in the long run. We can address this issue in our model by considering a utility function that directly encourages the expected spending level of campaign k to be close to the target budget value m_k. One such utility and conjugate function pair that uses a squared quadratic penalty, for a given parameter $\tau_k > 0$, is:

$$u_k(v_k) = -\frac{\tau_k}{2}(m_k - v_k)^2 \ , \quad p_k(\lambda_k) = m_k\lambda_k + \frac{1}{2\tau_k}\lambda_k^2 \ .$$

Example 2.2.3 (Budget Constraint and Target Spending). We can combine Examples 2.2.1 and 2.2.2 in order to encourage spending at, but not more than, the target value m_k:

$$u_k(v_k) = \begin{cases} -\frac{\tau_k}{2}(m_k - v_k)^2 & \text{if } v_k \le m_k \\ -\infty & \text{o/w} \end{cases} \ , \quad p_k(\lambda_k) = \begin{cases} m_k\lambda_k & \text{if } \lambda_k \ge 0 \\ m_k\lambda_k + \frac{1}{2\tau_k}\lambda_k^2 & \text{o/w} \end{cases} \ .$$

Example 2.2.4 (Budget Constraint and Hard Minimum Spending). A stricter way to guarantee a minimum budget spending is to require that campaign $k \in \mathcal{K}$ spends at least a fixed percentage of, and no more than, its target value m_k. In this case, for a given parameter $\alpha_k \in [0, 1]$, the utility and conjugate function pair is:

$$u_k(v_k) = \begin{cases} 0 & \text{if } v_k \in [\alpha_k m_k, m_k] \\ -\infty & \text{o/w} \end{cases} \ , \quad p_k(\lambda_k) = \begin{cases} m_k\lambda_k & \text{if } \lambda_k \ge 0 \\ \alpha_k m_k\lambda_k & \text{o/w} \end{cases} \ .$$

Note that $p_k(\cdot)$ can be equivalently expressed as $p_k(\lambda_k) = m_k[\lambda_k]_+ - \alpha_k m_k[-\lambda_k]_+$.

If the utility functions for each campaign $k \in \mathcal{K}$ are constructed from Examples 2.2.1, 2.2.2, or 2.2.3, then problem (2.1) is guaranteed to be feasible since $\mathbf{v}(0, 0) \in \text{dom}(u(\cdot))$. On the other hand, if Example 2.2.4 is used for one of the campaigns, then problem (2.1) may or may not be feasible depending on the choices of m_k and α_k.

Further Properties of the Dual Function

Let us now describe some additional properties of the dual function $q(\cdot)$ that provide further intuition and will also be useful in our two phase primal-dual procedure described in Section 2.2. Herein we demonstrate how to efficiently compute the function value $q(\boldsymbol{\lambda})$ as well as a subgradient of $q(\cdot)$ at any given $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$. It turns out that these computations rely on the ability to solve a simple one dimensional optimization problem to obtain the bidding variables, which we formalize below as an assumption.

Assumption 2.2.3. *For each $i \in \mathcal{I}$, define $h_i(b; r) := [r - \beta_i(b)]\rho_i(b)$, which represents the expected profit that the DSP earns each time it bids an amount b for an impression of type i, given that the expected revenue earned whenever the corresponding ad is displayed is equal to r. Then, for all values of $r \in \mathbb{R}$, we can efficiently compute an optimal bid price $b_i^*(r) \in \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r)$.*

Assumption 2.2.3 is not very restrictive as, under mild conditions, we can efficiently compute $b_i^*(r) \in \arg\max_{b\in[0,\bar{b}_i]} h_i(b;r)$ (at least approximately) using bisection or other methods. However, as is often the case, if the structure of the auction is well understood then $b_i^* \in \arg\max_{b\in[0,\bar{b}_i]} h_i(b;r)$ can be simply determined in closed form, as demonstrated by the following two examples. Formal derivations of $b_i^*(r)$ for these as well as additional examples are included in Appendix A.3.

Example 2.2.5 (Second-Price Auction cont.). Consider the case of a second-price auction for impressions of type $i \in \mathcal{I}$, as in Example 2.1.2. It is well known that bidding truthfully is optimal for second-price auctions [120]. In our context, truthful bidding implies that, for any $r \in \mathbb{R}$, $b_i^*(r) = \text{clip}(r; [0, \bar{b}_i])$, where recall that $\text{clip}(r; [0, \bar{b}_i]) := \min\left\{\max\{r, 0\}, \bar{b}_i\right\}$. Note importantly that this simple expression for $b_i^*(r)$ does not depend on the specific forms of $\rho_i(\cdot)$ or $\beta_i(\cdot)$ due to the special properties of second-price auctions.

Example 2.2.6 (First-Price Auction cont.). Consider the case of a first-price auction for impressions of type $i \in \mathcal{I}$, as in Example 2.1.1. In this case, we have $\beta_i(b) = b$ and $b_i^*(r)$ will depend on the functional form of $\rho_i(\cdot)$. As a simple example, consider the case where the highest competing bid C_i is equal to the maximum of n i.i.d. random variables uniformly distributed on $[0, \bar{b}_i]$. Then, it holds that $b_i^*(r) = \text{clip}(\frac{nr}{n+1}; [0, \bar{b}_i])$.

Given the separability property (2.5) of $q(\cdot)$, whereby $q(\cdot) = \sum_{i\in\mathcal{I}} q_i(\cdot)$, in order to compute $q(\boldsymbol{\lambda})$ and a subgradient of $q(\cdot)$ at any given $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$, by additivity of subgradients it suffices to demonstrate that we can efficiently compute $q_i(\boldsymbol{\lambda})$ as well as a subgradient of each of the individual functions $q_i(\cdot)$ at $\boldsymbol{\lambda}$ (which may be done in parallel). Recall that $q_i(\cdot)$ is defined by a certain maximization problem with respect to the variables $(\mathbf{x}_i, \mathbf{b}_i)$, whereby $q_i(\boldsymbol{\lambda}) := \max_{(\mathbf{x}_i,\mathbf{b}_i)\in\mathcal{S}_i} \left\{\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)\right\}$. Hence, in order to compute a subgradient of $q_i(\cdot)$ at $\boldsymbol{\lambda}$, it suffices to solve this maximization problem, which we now equivalently rewrite using the collection of functions $h_i(b;r) := [r - \beta_i(b)]\rho_i(b)$ parameterized by r defined in Assumption 2.2.3. Indeed, observe that:

$$
\begin{aligned}
q_i(\boldsymbol{\lambda}) &= \max_{(\mathbf{x}_i,\mathbf{b}_i)\in\mathcal{S}_i} \left\{ \sum_{k\in\mathcal{K}_i} [r_{ik} - \beta_i(b_{ik})]\rho_i(b_{ik})s_i x_{ik} \;-\; \sum_{k\in\mathcal{K}_i} \lambda_k r_{ik}\rho_i(b_{ik})s_i x_{ik} \right\} \qquad (2.7)\\
&= \max_{(\mathbf{x}_i,\mathbf{b}_i)\in\mathcal{S}_i} \left\{ \sum_{k\in\mathcal{K}_i} [r_{ik}(1-\lambda_k) - \beta_i(b_{ik})]\rho_i(b_{ik})s_i x_{ik} \right\}\\
&= \max_{(\mathbf{x}_i,\mathbf{b}_i)\in\mathcal{S}_i} \left\{ \sum_{k\in\mathcal{K}_i} h_i(b_{ik}; r_{ik}(1-\lambda_k))s_i x_{ik} \right\} .
\end{aligned}
$$

The above demonstrates that, for any given dual vector $\boldsymbol{\lambda}$, computing $q_i(\boldsymbol{\lambda})$ is equivalent to finding the allocation and bidding vectors $(\mathbf{x}_i, \mathbf{b}_i)$ for impression type i that maximize the profit of the DSP, but where the expected revenue quantities r_{ik} are each scaled by $(1 - \lambda_k)$. Hence, the dual variables $\boldsymbol{\lambda}$ provide a natural mechanism to account for the budget spending preferences of the campaigns within the dual functions $q_i(\cdot)$, whereby there is "extra spending" on campaign k if $\lambda_k < 0$, “normal spending" if $\lambda_k = 0$, “reduced spending" if $\lambda_k \in (0, 1)$, and “no spending" if $\lambda_k \geq 1$. The derivation in (2.7) also suggests a natural algorithm to compute $q_i(\boldsymbol{\lambda})$ and a corresponding subgradient: first maximize the scalar function $h_i(b_{ik}; r_{ik}(1 - \lambda_k))$ with respect to b_{ik} for each $k \in \mathcal{K}_i$, then chose the campaign $k_i^* \in \mathcal{K}_i$ with the largest corresponding optimal value. Algorithm 2 below formalizes this algorithm for computing $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda}))$ solving the maximization problem in (2.7) as well as the corresponding subgradient $\mathbf{g}_i \in \partial q_i(\boldsymbol{\lambda})$. Proposition 2.2.2 formally demonstrates that the output of Algorithm 2 is valid. Note that the computational complexity of Algorithm 2 is $O(|\mathcal{K}|)$ if one presumes that the bid prices $b_{ik}^*(\boldsymbol{\lambda})$ in Step (1.) can be computed in $O(1)$ time.

Algorithm 2 Computing $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda}))$ and a subgradient $\mathbf{g}_i \in \partial q_i(\boldsymbol{\lambda})$

```
  **Input:** $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$.
1. For each $k \in \mathcal{K}_i$, define:
   $b_{ik}^*(\boldsymbol{\lambda}) := b_i^*(r_{ik}(1-\lambda_k)) \in \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r_{ik}(1-\lambda_k)), \quad \tilde{\pi}_{ik}(\boldsymbol{\lambda}) := h_i(b_{ik}^*(\boldsymbol{\lambda}); r_{ik}(1-\lambda_k))$
2. Choose $k_i^* \in \arg\max_{k \in \mathcal{K}_i} \tilde{\pi}_{ik}(\boldsymbol{\lambda})$ arbitrarily and define:
   $x_{ik_i^*}^*(\boldsymbol{\lambda}) := \begin{cases} 1 & \text{if } \tilde{\pi}_{ik_i^*}(\boldsymbol{\lambda}) > 0 \\ 0 & \text{o/w} \end{cases}, \quad x_{ik}^*(\boldsymbol{\lambda}) := 0 \text{ for all } k \in \mathcal{K}_i, k \neq k_i^*.$
3. Set $\mathbf{g}_i := -\mathbf{v}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda}))$
  **Output:** $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda}))$ and $\mathbf{g}_i$.
```

Proposition 2.2.2. *For each $i \in \mathcal{I}$ and for all $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$, the output $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda}))$ and $\mathbf{g}_i$ of Algorithm 2 satisfies $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \arg\max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \left\{\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)\right\}$ and $\mathbf{g}_i \in \partial q_i(\boldsymbol{\lambda})$.*

Two-Phase Primal-Dual Procedure

We now describe in detail the two phase primal-dual procedure, in which we first solve the dual problem (2.6) to near optimality, and then use the dual variables to recover the allocation and bidding variables by solving an auxiliary convex optimization problem. Specifically, we first solve for (near) optimal dual variables $\hat{\boldsymbol{\lambda}}$, use these dual variables to construct biding variables $\hat{\mathbf{b}}$ via Assumption 2.2.3, and then solve for the allocation variables $\hat{\mathbf{x}}$ by solving a restricted version of problem (2.1) that results after fixing the bidding variables at $\hat{\mathbf{b}}$. For a given fixed vector $\mathbf{b}$, this restricted problem that maximizes the objective function of (2.1) with respect to the allocation variables $\mathbf{x}$ is presented below in (2.8).

$$\begin{aligned} F^*(\mathbf{b}) := \quad \underset{\mathbf{x}}{\text{maximize}} \quad & \pi(\mathbf{x},\mathbf{b}) + u\left(\mathbf{v}(\mathbf{x},\mathbf{b})\right) \\ \text{subject to} \quad & \sum_{k\in\mathcal{K}_i} x_{ik} \leq 1 && \text{for all } i \in \mathcal{I} \\ & x_{ik} \geq 0 && \text{for all } (i,k) \in \mathcal{E} \end{aligned} \tag{2.8}$$

Note that (2.8) is a concave maximization problem (hence a convex optimization problem) since, for fixed $\mathbf{b}$, $\pi(\cdot,\mathbf{b})$ and $\mathbf{v}(\cdot,\mathbf{b})$ are both linear functions in $\mathbf{x}$ (which implies that $u(\mathbf{v}(\cdot,\mathbf{b}))$ is a concave function). Let $\mathcal{X}$ denote the set of feasible allocation vectors $\mathbf{x}$ in (2.8) and let $\mathcal{V}(\mathbf{b}) := \{\mathbf{v}(\mathbf{x},\mathbf{b}) : \mathbf{x} \in \mathcal{X}\}$ denote the set of possible expected spending values given $\mathbf{b}$. The following proposition demonstrates that if problem (2.8) is feasible, then it is also finite and attained.

Proposition 2.2.3. *Suppose that problem* (2.8) *is feasible, i.e., it holds that* $\mathrm{dom}(u(\cdot)) \cap \mathcal{V}(\mathbf{b}) \neq \emptyset$. *Then, the optimal value* $F^*(\mathbf{b})$ *of* (2.8) *is finite and attained at some* $\mathbf{x}^*(\mathbf{b}) \in \mathcal{X}$.

The proof of Proposition 2.2.3 exactly follows that of Proposition 2.2.1 and is omitted for brevity. Note that for many choices of utility functions, such as Examples 2.2.1, 2.2.2, and 2.2.3 from Section 2.2, the feasibility condition is guaranteed to hold for all $\mathbf{b}$ since, in these examples, we have that $\mathbf{0} \in \mathrm{dom}(u(\cdot)) \cap \mathcal{V}(\mathbf{b})$ for all $\mathbf{b}$.

Algorithm 3 below formally presents our two-phase primal-dual solution procedure. Note that both phases involve solving a particular convex optimization problem. Phase 1 involves solving the dual problem (2.6). Due to the generic form of the bid landscape functions and other problem parameters, we generally need to treat the dual problem in a black-box format that can be tackled efficiently with subgradient based algorithms, the simplest of which is the projected subgradient descent method (see Section 2.4 for details), using Algorithm 2 as a subroutine to compute subgradients of $q(\cdot)$. On the other hand, problem (2.8), which is solved in Phase 2, is often highly structured depending on the choice of utility function. For example, any combination of the four examples given in Section 2.2 results in a convex problem with a quadratic objective and linear constraints (and a possibly a linear objective if only Examples 2.2.1 and 2.2.4 are used), which is a highly structured problem that can be tackled with commercial solvers such as Gurobi [55].

Algorithm 3 Two-Phase Primal-Dual Procedure

```
**Phase 1: Solve the Dual Problem**
  Solve the dual problem (2.6), obtaining (approximately) optimal dual variables $\hat{\boldsymbol{\lambda}}$ and dual
  objective function value $Q(\hat{\boldsymbol{\lambda}})$.
**Phase 2: Primal Recovery**
  1. For all $(i,k) \in \mathcal{E}$, set $\hat{b}_{ik} = b_i^*(r_{ik}(1-\hat{\lambda}_k)) \in \arg\max_{b\in[0,\bar{b}_i]} h_i(b; r_{ik}(1-\hat{\lambda}_k))$.
  2. Fixing $\hat{\mathbf{b}}$ as computed previously, solve the restricted primal problem (2.8) to obtain the
  allocation variables $\hat{\mathbf{x}}$ and primal objective function value $F(\hat{\mathbf{x}}, \hat{\mathbf{b}})$.
**Output:** Feasible primal solution $(\hat{\mathbf{x}}, \hat{\mathbf{b}})$ and suboptimality bound $Q(\hat{\boldsymbol{\lambda}}) - F(\hat{\mathbf{x}}, \hat{\mathbf{b}})$.
```

2.3 Zero Duality Gap Results

In Section 2.2, we introduced our primal optimization problem of interest (2.1) that balances profitability for the DSP with satisfying the advertisers' budget spending goals, its dual problem (2.6), as well as a two-phase primal-dual approach for solving (2.1), Algorithm 3. Despite the non-convexity of problem (2.1), we demonstrate herein that certain conditions on the problem parameters (all defined in Section 2.1) imply that: *(i)* there is zero duality gap between the primal problem (2.1) and its dual (2.6), and *(ii)* Algorithm 3 recovers an optimal solution of the primal problem (2.1). In particular, we define a type of "increasing marginal cost" condition on the bid landscape functions $\rho_i(\cdot)$ and $\beta_i(\cdot)$ that, when satisfied by all impression types $i \in \mathcal{I}$, provides a sufficient condition for the zero duality gap result. This condition is formally defined below in Definition 2.3.1.

Definition 2.3.1 (Increasing Marginal Cost (IMC) Condition)**.** *Impression type $i \in \mathcal{I}$ satisfies the increasing marginal cost (IMC) condition if the bid landscape functions $\rho_i(\cdot) : [0, \bar{b}_i] \to [0,1]$ and $\beta_i(\cdot) : [0, \bar{b}_i] \to [0, \infty)$ are differentiable on $(0, \bar{b}_i)$ and satisfy:*

1. *$\rho_i'(b) > 0$ for all $b \in (0, \bar{b}_i)$.*
2. *$g_i(b) := \frac{c_i'(b)}{\rho_i'(b)}$ is strictly increasing on the interval $b \in (0, \bar{b}_i)$, where $c_i(b) := \rho_i(b)\beta_i(b)$ is the expected cost associated with impression type i.*

The function value $g_i(b) = \frac{c_i'(b)}{\rho_i'(b)}$, defined as part of Definition 2.3.1, may be interpreted as the "expected marginal cost" of winning another auction for impressions of type i when bidding an amount b. Indeed, $c_i'(b)$ represents the rate of change of the expected cost function $c_i(\cdot)$ at b, whereas $\rho_i'(b)$, which is assumed positive, represents the rate of change of the probability of winning the auction $\rho_i(\cdot)$ at b. Hence, the ratio $g_i(b) := \frac{c_i'(b)}{\rho_i'(b)}$ may be interpreted as the "change in expected cost per bid" divided by the "change in quantity of auctions won per bid," i.e., the marginal cost of winning another auction. The IMC Condition then simply states that this "expected marginal cost" function $g_i(\cdot)$ is increasing. In Section 2.3, we discuss several examples of popular and important cases, including first and second-price auctions, that satisfy the IMC condition. It also turns out that the IMC Condition is a special case of a more general condition which states that, for all values of the expected revenue term r, there is a unique optimal bid price that maximizes the expected profit function $h_i(b;r) = [r - \beta_i(b)]\rho_i(b)$. This condition also implies the zero duality gap result, and is stated formally in Definition 2.3.2 below (Lemma A.2.2 in the appendix proves that condition IMC is a special case of condition UBP defined below).

Definition 2.3.2 (Unique Bid Price (UBP) Condition). *Impression type $i \in \mathcal{I}$ satisfies the unique bid price (UBP) condition if, for all $r \in \mathbb{R}$, the expected profit function $h_i(b;r) := [r - \beta_i(b)]\rho_i(b)$ has a unique maximizer $b_i^*(r)$ on $[0, \bar{b}_i]$, i.e., it holds that $b_i^*(r) = \arg\max_{b \in [0,\bar{b}_i]} h_i(b;r)$.*

Theorem 2.3.1 below presents our main zero duality gap result under either the IMC or UBC Conditions. Recall that we defined feasibility of problem (2.1) by the condition $\mathrm{dom}(u(\cdot)) \cap \mathcal{V} \neq \emptyset$, where $\mathcal{V} = \{\mathbf{v}(\mathbf{x}, \mathbf{b}) : (\mathbf{x}, \mathbf{b}) \in \mathcal{S}\}$ is the set of feasible expected spending values. The statement of Theorem 2.3.1 strengthens this feasibility condition to a version of Slater's condition for our problem, which states that $\mathrm{int}(\mathrm{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$ and ensures the existence of a dual optimal solution $\boldsymbol{\lambda}^*$. Indeed, this condition is correctly interpreted as a version of Slater's condition since $\mathrm{int}(\mathrm{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$ is equivalent to the existence of a feasible point $(\mathbf{x}, \mathbf{b}) \in \mathcal{S}$ such that $\mathbf{v}(\mathbf{x}, \mathbf{b}) \in \mathrm{int}(\mathrm{dom}(u(\cdot)))$. Note that Slater's Condition always holds for the utility functions in Examples 2.2.1, 2.2.2, and 2.2.3 from Section 2.2 since in all three cases we have $\mathbf{v}(\mathbf{0}, \mathbf{0}) = \mathbf{0} \in \mathrm{int}(\mathrm{dom}(u(\cdot)))$. For Example 2.2.4, Slater's Condition may not always hold but such cases are inherently degenerate.

Theorem 2.3.1. *Suppose that problem* (2.1) *satisfies Slater's Condition* $\mathrm{int}(\mathrm{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$ *and that, for all impression types $i \in \mathcal{I}$, either the IMC Condition (Definition 2.3.1) or the more general UBP Condition (Definition 2.3.2) holds. Then, the following statements hold:*

1. *There exists an optimal solution $\boldsymbol{\lambda}^*$ of the dual problem* (2.6).
2. *There is zero duality gap between the primal problem* (2.1) *and its dual* (2.6)*, i.e., it holds that $Q^* = F^*$.*
3. *When both the Phase 1 and Phase 2 subproblems of Algorithm 3 are solved to exact optimality, then Algorithm 3 returns an optimal solution* $(\mathbf{x}^*, \mathbf{b}^*)$ *of the primal problem* (2.1).

The proof of Theorem 2.3.1 is contained in Section A.2 of the Appendix. Appendix A.3 also contains examples showing that the conclusions of Theorem 2.3.1 may not hold if one or more of the assumptions are violated. Notice that the IMC and UBP Conditions depend on each impression type $i \in \mathcal{I}$ independently; in particular they only depend on the properties of the bid landscape functions $\rho_i(\cdot)$, $\beta_i(\cdot)$ on the interval $[0, \bar{b}_i]$. Thus, Theorem 2.3.1 may hold even in cases when different auction types and/or structurally different bid landscape functions are used for different impression types. Under the stated conditions, in addition to demonstrating the existence of a dual optimal solution $\boldsymbol{\lambda}^*$ as well as zero duality gap, Theorem 2.3.1 demonstrates that an optimal solution of the original problem of interest (2.1) may be efficiently computed using Algorithm 3. In particular, as long as the stated conditions of Theorem 2.3.1 hold and as long as the optimization problems in the two phases of the algorithm are solved to exact optimality, then Algorithm 3 returns an optimal solution $(\mathbf{x}^*, \mathbf{b}^*)$ of the primal problem (2.1). It is worth pointing out that the proof of Theorem 2.3.1 reveals that, under the stated conditions, there also exists a particular optimal solution $(\mathbf{x}^*, \mathbf{b}^*)$ with the property that $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \arg\max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \left\{\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)\right\}$ for all $i \in \mathcal{I}$. Although this is the same computation that is performed by Algorithm 2 when given input $\boldsymbol{\lambda}^*$, note that the output of Algorithm 2 is not in general unique. Therefore, Algorithm 2 with input $\boldsymbol{\lambda}^*$ is not guaranteed to return a primal optimal solution and the primal recovery phase of Algorithm 3 is necessary. In Section 2.3, we extend item (3.) of Theorem 2.3.1 to demonstrate that Algorithm 3 returns an approximately optimal primal solution when the optimization problems in the two phases of the algorithm are solved only to approximate optimality.

Examples Satisfying the IMC Condition

In this Section, we give several examples of bid landscape functions, arising from auction types that are widely used in practice, that satisfy the IMC Condition. Hence, if any combination of these bid landscape functions are used, and if the mild Slater's Condition holds, then Theorem 2.3.1 may be applied. The proofs of Propositions 2.3.1 and 2.3.2 are contained in Appendix A.2.

Proposition 2.3.1 (Second-Price Auction cont.). *Suppose that impression type $i \in \mathcal{I}$ corresponds to a second-price auction, as in Examples 2.1.2 and 2.2.5, and let C_i be a non-negative random variable representing the highest competing bid. If C_i is a continuous random variable with probability density function $f_{C_i}(\cdot)$ satisfying $f_{C_i}(b) > 0$ for all $b \in [0, \bar{b}_i]$, then $\rho_i(b) = \mathbb{P}(C_i < b)$ and $\beta_i(b) = \mathbb{E}[C_i \mid C_i < b]$ satisfy the IMC Condition.*

Proposition 2.3.2 (First-Price Auction cont.). *Suppose that impression type $i \in \mathcal{I}$ corresponds to a first-price auction, as in Examples 2.1.1 and 2.2.6, with $\beta_i(b) = b$ for all $b \in [0, \bar{b}_i]$. If $\rho_i(\cdot)$ is differentiable, strictly increasing, and concave on $(0, \bar{b}_i)$, then $\rho_i(\cdot)$ and $\beta_i(\cdot)$ satisfy the IMC condition. (Several examples of functional forms of $\rho_i(\cdot)$ satisfying the previous conditions are given in Appendix A.3.) Alternatively, if $\rho_i(\cdot)$ is the CDF of the maximum of n i.i.d. uniform random variables on the interval $[0, \bar{c}_i]$ with $\bar{c}_i \geq \bar{b}_i$, then $\rho_i(\cdot)$ and $\beta_i(\cdot)$ satisfy the IMC condition.*

It is worth mentioning that the IMC Condition, as well as results relying on the IMC condition such as Propositions 2.3.1 and 2.3.2, may be readily extended to allow for the possibility a fixed and known reserve price $\underline{b}_i \geq 0$. In such a case, a bid in an auction for impression type i is only considered valid if it is above $\underline{b}_i$ and the DSP has complete knowledge of the value $\underline{b}_i$. With a reserve price, the rules corresponding to first and second-price auctions remain the same but only valid bids above $\underline{b}_i$ are considered. For simplicity, we assume that $\underline{b}_i = 0$ in this Definition 2.3.1 and Propositions 2.3.1 and 2.3.2, but these results may be easily modified to allow for the case of $\underline{b}_i > 0$.

Algorithmic Convergence

In this section, we establish continuity results in order to demonstrate that, in the limit, Algorithm 3 returns an approximately optimal primal solution when the optimization problems in the two phases of the algorithm are solved only to approximate optimality. We assume throughout that the UBP Condition holds for all impression types $i \in \mathcal{I}$, which, as formalized in Lemma A.2.2 of the Appendix, is a more general condition than the IMC Condition. For any $i \in \mathcal{I}$ and $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$ we use the notation $\mathbf{b}_i^*(\boldsymbol{\lambda})$ to denote the vector in $\mathcal{B}_i := [0, \bar{b}_i]^{|\mathcal{K}_i|} \subseteq \mathbb{R}^{|\mathcal{K}_i|}$ whose component corresponding to campaign $k \in \mathcal{K}_i$ is equal to $b_{ik}^*(\boldsymbol{\lambda}) = \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r_{ik}(1 - \lambda_k))$, the unique maximizer of the profit expression $h_i(b; r_{ik}(1 - \lambda_k)) = [r_{ik}(1 - \lambda_k) - \beta_i(b)]\rho_i(b)$ on $[0, \bar{b}_i]$. We also use the notation $\mathbf{b}^*(\boldsymbol{\lambda}) \in \mathcal{B} \subseteq \mathbb{R}^{|\mathcal{E}|}$ to denote the enlarged vector of all of the $\mathbf{b}_i^*(\boldsymbol{\lambda})$ subvectors concatenated together. Recall also that $\mathcal{V}(\mathbf{b}) := \{\mathbf{v}(\mathbf{x}, \mathbf{b}) : \mathbf{x} \in \mathcal{X}\}$ denotes the set of possible expected spending values given $\mathbf{b}$.

Recall that Phase 1 of Algorithm 3 returns an approximately optimal vector of dual variables $\hat{\boldsymbol{\lambda}}$, which is then used to determine the vector of associated bid prices $\mathbf{b}^*(\hat{\boldsymbol{\lambda}})$ in Phase 2. The second part of Phase 2 then involves solving the restricted primal problem (2.8) given $\mathbf{b}^*(\hat{\boldsymbol{\lambda}})$ with optimal value $F^*(\mathbf{b}^*(\hat{\boldsymbol{\lambda}}))$. Thus, in order to properly consider the effect of only approximately solving problems (2.6) and (2.8), we must first establish some basic continuity properties of $\mathbf{b}^*(\cdot)$ and $F^*(\cdot)$. Proposition 2.3.3 below first establishes that $\mathbf{b}_i^*(\cdot)$ is continuous if impression type i satisfies the UBP Condition.

Proposition 2.3.3. *If impression type $i \in \mathcal{I}$ satisfies the UBP Condition, then $\mathbf{b}_i^*(\cdot)$ is a continuous function on $\mathbb{R}^{|\mathcal{K}|}$.*

Note that a simple consequence of Proposition 2.3.3 is that $\mathbf{b}^*(\cdot)$ is continuous if all impression types $i \in \mathcal{I}$ satisfy the UBP Condition. Now, in order to demonstrate continuity of $F^*(\cdot)$, we need a slightly stronger variant of Slater's Condition. In particular, Proposition 2.3.4 below states that $F^*(\cdot)$ is continuous at $\bar{\mathbf{b}} \in \mathcal{B}$ whenever it holds that $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V}(\bar{\mathbf{b}}) \neq \emptyset$. Note that for many choices of utility functions, such as Examples 2.2.1, 2.2.2, and 2.2.3 from Section 2.2, we have that $\mathbf{0} \in \text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V}(\bar{\mathbf{b}})$ for all $\bar{\mathbf{b}} \in \mathcal{B}$; hence in these cases $F^*(\cdot)$ is continuous on $\mathcal{B}$.

Proposition 2.3.4. *Let $\bar{\mathbf{b}} \in \mathcal{B}$ be given and suppose it holds that $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V}(\bar{\mathbf{b}}) \neq \emptyset$. Then, $F^*(\cdot) : \mathcal{B} \to \mathbb{R} \cup \{-\infty\}$ defined in (2.8) is continuous at $\bar{\mathbf{b}}$.*

We are now ready to state the following result, which uses the previous continuity results to establish an "approximate version" of Theorem 2.3.1. That is, suppose that, in Phase

1 of Algorithm 3, we apply an algorithm that converges to a dual optimal solution. Then, Theorem 2.3.2 states that the corresponding sequence of primal solutions obtained in Phase 2 of Algorithm 3 also converges (in objective function values) to a primal optimal solution.

Theorem 2.3.2. *Suppose that problem* (2.1) *satisfies the following stronger variant of Slater's Condition:* $\operatorname{int}(\operatorname{dom}(u(\cdot))) \cap \mathcal{V}(\mathbf{b}) \neq \emptyset$ *for all* $\mathbf{b} \in \mathcal{B}$. *Furthermore, suppose that for all impression types* $i \in \mathcal{I}$, *either the IMC Condition (Definition 2.3.1) or the more general UBP Condition (Definition 2.3.2) holds. Let* $\{\boldsymbol{\lambda}^t\}$ *be a sequence of dual variables (i.e., the iterates of an algorithm) converging to a dual optimal solution* $\boldsymbol{\lambda}^*$, *i.e.,* $\boldsymbol{\lambda}^t \to \boldsymbol{\lambda}^*$. *Then, in addition to statements (1.)-(3.) of Theorem 2.3.1, the following statement holds:*

1. *The corresponding sequence of restricted primal values* $\{F^*(\mathbf{b}^*(\boldsymbol{\lambda}^t))\}$ *converges to the optimal primal value* F^* *of* (2.1), *i.e.,* $F^*(\mathbf{b}^*(\boldsymbol{\lambda}^t)) \to F^*$.

The proof of Theorem 2.3.2, as well as the proofs of Propositions 2.3.3 and 2.3.4, are included in Section A.2 of the appendix.

2.4 Numerical Experiments

In this section, we present results wherein we applied Algorithm 3 (our two-phase solution procedure) on both synthetic and real data examples. For the real data case, we used bidding logs from Criteo ([39]). Our synthetic experiments simulate scenarios using first and second-price auctions, while the Criteo experiment assumes that impressions are sold using second-price auctions. Our experiments demonstrate the following results. First, we can obtain a family of solutions that trade off DSP profitability for higher budget spending rates for the DSP campaigns. This trade-off is obtained by using different utility functions with different parameters. Second, our methodology outperforms a natural baseline that is optimal when budgets go to infinity as well as a natural modification of this baseline that also trades off between profitability and budget spending rates in a more heuristic manner.

The remainder of this section is organized as follows. We first describe the implementation details and the evaluation procedure for our experiments. Then, we describe the experimental settings and results for the synthetic and real data experiments. In our experiments we use the utility functions described in Examples 2.2.1, 2.2.2, and 2.2.3. For ease of exposition, we will refer to them as UF 3, UF 4, and UF 5, respectively, in this section. Finally, please refer to Appendix A.4 for additional results for both the synthetic and real data based experiments.

Experimental Setup

At a high level, we first run Algorithm 3 during a training phase and then Policy 1 in a testing phase. To find an approximately optimal dual solution for Phase 1 of Algorithm 3 we use the basic projected subgradient descent method, which has the form $\boldsymbol{\lambda}^{t+1} := \Pi_{\operatorname{dom}(p(\cdot))}(\boldsymbol{\lambda}^t - \alpha_t \mathbf{g}^t)$, where $\mathbf{g}^t \in \partial Q(\boldsymbol{\lambda}^t)$ and $\Pi_{\operatorname{dom}(p(\cdot))}(\cdot)$ is the Euclidean projection operator to the domain of $p(\cdot)$. This projection operator can be removed for UF 4 and UF 5, while for UF 3 the operator function is simply $\Pi_{\text{dom}(p(\cdot))}(\boldsymbol{\lambda}) = [\boldsymbol{\lambda}]_+$, the element-wise maximum between the vector $\boldsymbol{\lambda}$ and zero. We use step sizes of the form $\alpha_t = C/\sqrt{t}$ where the constant C was selected by running Algorithm 3 for 5000 iterations using $C = 10^{2,\dots,-6}$ and selecting the one with highest primal value $F\left(\mathbf{b}^*\left(\frac{1}{2500}\cdot\sum_{i=2501}^{5000}\boldsymbol{\lambda}^t\right)\right)$. The candidate dual solution in Algoritm 3 is taken as $\hat{\boldsymbol{\lambda}} := \sum_{i=2501}^{5000}\boldsymbol{\lambda}^t$.

The testing phase consists of evaluating the performance of one or more bidding and allocation policies by simulating the dynamics described in Section 2.1. In particular, we are primarily interested in evaluating the performance of Policy 1 with input $(\hat{\mathbf{x}}, \hat{\mathbf{b}})$ generated by Algorithm 3 during the training phase. Notice that the dynamics described in Section 2.1 and Policy 1 depend on the parameters described in Section 2.1. Since some of these parameters need to be estimated, as described in Section 2.1, it is possible that there are errors in the estimation procedures that lead to a discrepancy between the parameters used in the training and testing phases. Additionally, the order in which different impression types arrive in the testing phase is not known during the training phase. In our synthetic experiments, we assume that there are no errors in parameter estimation, i.e., all of the parameters are the same in both the training and testing phases. In the real data experiments, the conversion probabilities θ_{ik} may be different in the training and testing phases, but all other parameters are the same. Full details of our simulation and testing procedures are given in Appendix A.4.

A major characteristic of our proposed methodology is that it efficiently and effectively trades off between profitability for the DSP and the level of budget spending for its campaigns. Thus, we present empirical results that explore this trade off. In particular, we consider two performance metrics herein: *(i)* total profit, and *(ii)* budget utilization. The total profit is simply the sum of all profits earned by the DSP over the entire course of the simulation. Budget utilization is defined as the ratio of the total amount spent by all campaigns (empirically observed at the end of the simulation) divided by $\sum_{k\mathcal{K}} m_k$, the sum of all target budget values. For both synthetic and real data-based experiments, we present "Pareto frontier-like" graphs that show achievable ranges for these two performance metrics. In particular, for UF 4 and 5, we obtain non-trivial graphs by varying the penalization parameter τ parameter used in the definition of these utility functions. We compare different configurations of our algorithm, i.e., different utility function choices and varying values of τ, against a family of baseline policies referred to as the generalized greedy policy shown in Policy 4. For all policies considered, we run 100 simulations to achieve better statistical significance. To make the results more comparable, on a particular experiment, each of the 100 runs use the same data and random number seeds for all policies (i.e., all configurations of our algorithm and of Policy 4) considered.

Policy 4 is optimal in the case of $\gamma = 1$ with infinite budgets (in the model with budget constraints, i.e., UF 3) since it chooses the campaign and bid that maximizes the expected profit for each impression. Put another way, Policy 4 with $\gamma = 1$ is optimal when $u(\cdot) = 0$. In general, the "pure" greedy policy, i.e., Policy 4 with $\gamma = 1$, may not lead to adequate spending

Policy 4 Generalized Greedy Policy

```
**Input:** Fixed scaling parameter $\gamma > 0$ and a new impression arrival of type $i \in \mathcal{I}$.
1. Choose a campaign $\tilde{k} \in \mathcal{K}_i$ among those campaigns whose budget is not currently depleted
   and that maximizes $h_i(r_{ik}; b_i^*(r_{ik}))$, with $b_i^*(r_{ik}) \in \arg\max_{b\in[0,\bar{b}_i]} h_i(r_{ik}; b) = (r_{ik} - \beta_i(b))\,\rho_i(b)$.
2. Bid $\gamma \cdot b_i^*(r_{ik})$ on behalf of the chosen campaign.
```

levels for the campaigns. The parameter γ therefore provides a simple mechanism to promote alternative rates of spending. In particular, choices of $\gamma \in (0, 1)$ represent a simple and naive way of promoting under-spending and choices of $\gamma > 1$ represent a simple and naive way to spend aggressively. As we vary the γ parameter we also obtain non-trivial graphs in the space of observed total profit and budget utilization values, just as we do when we vary the penalization parameter τ for UF 4 and UF 5.

Before presenting our results, let us make a few more remarks with respect to Policies 1 and 4 and our general experimental scheme. First, in all of our experiments we assume that each campaign $k \in \mathcal{K}$ can not spend more than its target budget m_k. A campaign will stop spending once its remaining budget is less than ℓ_k (the price it would pay for the next click or action of interest). Second, in practice a DSP would run Algorithm 3 inside a model predictive control (MPC) or resolving scheme. In this scheme, Algorithm 3 would be re-run as the campaigns spend their budgets (or under other events). Here we run Algorithm 3 once and then consider the allocation and bidding vectors $(\hat{\mathbf{x}}, \hat{\mathbf{b}})$ as fixed for our simulations. Generally speaking, an MPC version of any configuration of our algorithm is expected to perform better than the fixed policy version. Thus, the results presented herein are a conservative *lower bound* on the performance of an MPC scheme. Finally, Policy 4 adjusts its strategy as campaigns completely deplete their budgets, which suggests it may have a possible advantage over our methodology which does not. Policy 1 can be easily adapted to adjust its strategy in the same way. However, in order to be more faithful to the original interpretation of the variables $(\hat{\mathbf{x}}, \hat{\mathbf{b}})$, we do not perform this adjustment in Policy 1. We again expect this type of adjustment to only improve the performance of our methodology, and therefore the results presented herein again represent a conservative lower bound on the performance of a more practical version of Policy 1.

Synthetic Experiments

In this section, we discuss the results of two synthetic experiments. In the first experiment, we test how our proposed methodology performs when we fix all parameters, except for the target budget values m_k. In the second experiment, we consider the trade off between profitability and budget utilization by fixing all parameters except for the penalization parameter τ in UF 4 and UF 5 as well as the parameter γ in Policy 4. Both experiments use $|\mathcal{K}| = |\mathcal{I}| = 50$ with $s_i = 1000$ for all $i \in \mathcal{I}$ and $\ell_k = 1$ for all $k \in \mathcal{K}$. Each of the of the 100 runs per configuration is composed of $\sum_{i\in\mathcal{I}} s_i = 50000$ impression arrivals, and the type of each impression arrival

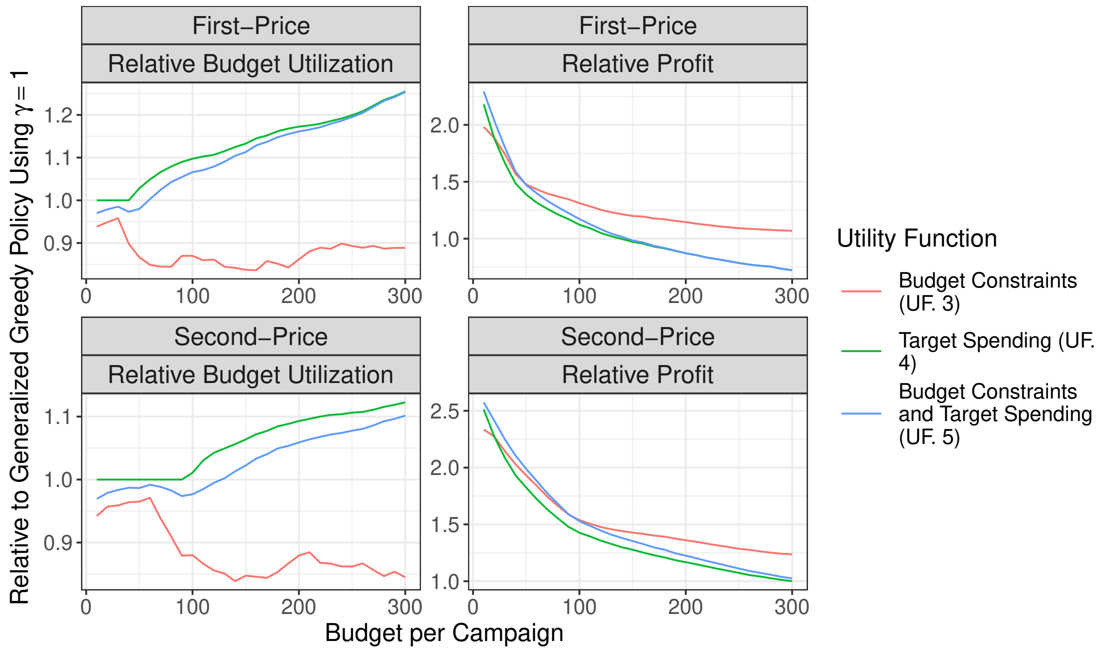


Figure 2.1: Plots showing the observed profit and budget utilizations of our proposed formulations, relative to the standard greedy heuristic (Policy 4 with $\gamma = 1$), versus the target budget values. The first row shows results assuming first-price auctions, and the second row shows results assuming second-price auctions.

is obtained by sampling uniformly from the set $\{1, \ldots, |\mathcal{I}|\}$. The highest competing bid for each impression arrival of type i is obtained by sampling from a maximum of adv_i i.i.d. Uniform$[0, 1]$ random variables, where adv_i is a random integer value defined in Appendix A.4. Note that, given the value of adv_i, in the cases of both first and second-price auctions we obtain closed formulas for the bid landscape functions and the optimal bidding function $b_i^*(\cdot)$ as described in Appendix A.3. For brevity, we defer the remainder of the discussion of how the impression-campaign graph, the bid landscapes, the conversion probabilities, and the revenue terms are generated to Appendix A.4.

Sensitivity With Respect to Budgets

In this experiment we assume that every campaign has an identical target budget value m_k and we vary these values in the range $m_k \in \{10, \ldots, 300\}$. We compare our methodology using UF 3, UF 4, and UF 5 against the “standard" greedy heuristic, i.e., Policy 4 with $\gamma = 1$. We used $\tau_k = 1/m_k$ as the penalization parameters in UF 4, and 5 for all $k \in \mathcal{K}$. Any choice of τ_k could have been used, but the choice of $\tau_k = 1/m_k$ places the utility function term in the objective on roughly the same scale as the DSP’s profit term.

Figure 2.1 shows the observed relative profit and budget utilization values of our methodology as compared to the standard greedy heuristic for both first and second-price auctions. In other words, these plots show the observed budget utilization and total profit values of our policies divided by their respective values for the standard greedy policy with $\gamma = 1$. The plots in Figure 2.1 demonstrate that, for small budget values, our methodology more than doubles the profit of the standard greedy heuristic without losing much in budget utilization. In fact, for second-price auctions UF 4 strictly dominates the greedy heuristic (UF 4 has both larger budget utilization and larger profit than the greedy heuristic), and UF 5 does as well for budget values above 100 or so. In the case of first-price auctions, UF 4 and 5 dominate the greedy heuristic for small to medium size budget values, while for high budget values they trade profitability for higher budget utilization. UF 3 consistently obtains more profit than the greedy heuristic, but sacrifices budget utilization. Generally speaking, the budget utilization of the greedy heuristic and UF 4 are the same for small budget values as they spend the advertisers' budgets entirely. UF 5 in general is not able to spend the advertisers budgets fully since using m_k as hard a constraint does not give enough slack to account for the randomness of the actual impression arrivals.

As we increase the target budget values m_k, the relative profit decreases faster for first-price auctions in comparison to second-price auctions. This makes sense as the cost that the DSP pays for each impression is higher for first-price auctions. Also, notice that – again in the first-price case – for high budget values UF 4 and UF 5 obtain worse profitability than the standard greedy heuristic, but better budget utilization. This occurs since, when we increase the budget, the ℓ_2 penalization part in the objective function of problem 2.2 becomes more important, which leads to the DSP bidding higher overall. In the case of first-price auctions, the cost of bidding higher is immediately reflected in all auctions won by the DSP. For second-price auctions, the cost of bidding higher is reflected only in those auctions with high market price which were won given the increase in the bid values.

Pareto Frontier-like Analysis

In this experiment we fixed the budgets to be $m_k = 100$ for all $k \in \mathcal{K}$, but changed the ℓ_2 penalization parameter τ ($\tau = \tau_k$ for all $k \in \mathcal{K}$) used in UF 4 and UF 5. For each penalization parameter value, we recorded the total profit and budget utilization during the testing phase for each of the 100 runs. The same was done for multiple values of γ in Policy 4. UF 3 was also run, but there is no analogous parameter needed to be explored.

Figure 2.2 shows the results of this experiment and demonstrates that by changing the penalization parameter τ of UF 4 and 5 we can obtain better profit and budget utilization than Policy 4. In fact, the performance of our methodology almost always dominates Policy 4 except for a few values of τ in UF 4 in the first-price case. Figure 2.2 highlights a central premise of this work – our framework offers flexibility for a DSP to tune its desired budget utilization level for their advertisers in a smart way in multiple auction environments. Notice that for the same level of budget utilization, the average total profit is lower in first-price auctions compared to second-price auctions. This is expected since the probability of winning

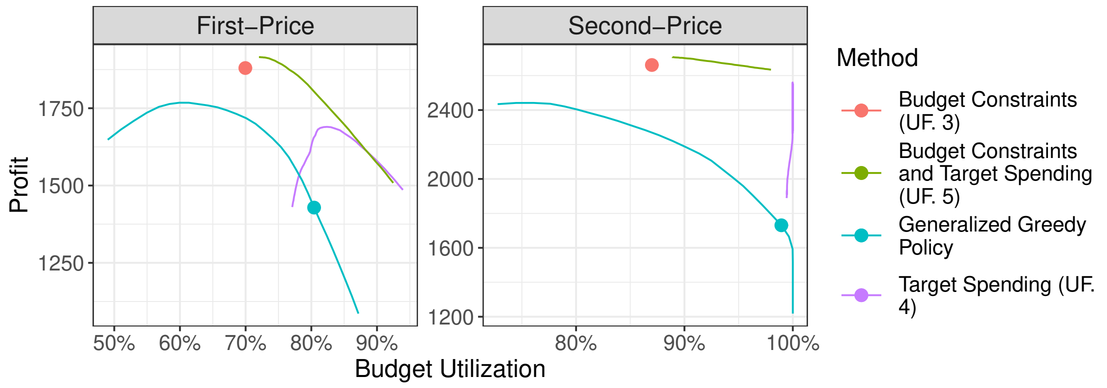


Figure 2.2: Profit and budget utilization as we change the penalization parameter τ of UF 4 and UF 5, and the scaling parameter γ of Policy 4. The dots in the graph represent UF 3, which has no parameters, and Policy 4 without scaling ($\gamma = 1$).

an auction is the same under both auction types. Furthermore, under the same highest competing bid, the price paid by the winning DSP is always lower in a second-price auction compared to a first-price (we used the same highest competing bid structure for both auction mechanisms).

Let us compare the performance of UF 4 versus UF 5. In the first-price case, UF 5 is able to total profit greater than or equal to that of UF 4 for the same level of budget utilization. In the second-price case, UF 4 does a better job at fully utilizing the budget, while 5 trades off a slightly lower budget utilization for more total profit. The latter makes sense as UF 4 is naive in the sense that it does not consider that once a campaign depletes its budget, the DSP does not continue bidding for it. The latter also explains why UF 4 shows a somehow erratic behaviour. On the other hand, very high budget utilization levels near 100 percent are not achievable for UF 5. The mismatch between impressions received at "real-time" and expected impressions when running Algorithm 3 does not match well with having the exact target budgets as upper bounds in its budget spending constraints. In comparison, UF 4 suffers less of this mismatch as it allows overspending. A helpful trick to improve the behaviour of UF 5 under high budget utilization levels would be to use a slightly higher target budget value m_k in the constraints than the corresponding value used in the ℓ_2 term when running Algorithm 3.

Real Data Experiments

In this subsection, we present results of an experiment based on a real world dataset from the DSP Criteo [39]. The dataset was generated during one month of Criteo's operation, and is composed of 16.5M impression logs in which Criteo bid on behalf of 700 advertisers. This dataset was released to address the problem of conversion attribution modelling, while here adapt it for our purposes. In particular, we consider the "action of interest" in our model to be a click and ignore columns in the data corresponding to conversions. Criteo was bidding in a second-price market and the 16.5M logs are all logs in which Criteo won the auction. Importantly, each impression log records the market price for that impression (the highest competing bid), nine categorical features of that impression, which campaign showed an ad, and if a click occurred or not. An ideal DSP dataset should also include the impressions lost by the DSP, but unfortunately a DSP has no access to the market price for those impressions. As is often standard in the literature (see [129] and others such as [107]), we assume that the distribution of winning logs is reasonably representative of the overall distribution of impression logs. In other words, we ignore the censored data problem. As long as Criteo's bidding algorithm did a reasonably good job at acquiring impressions – which we believe it did – then the censored data problem is not a major issue for our purposes since our main interest is in showing the benefits of our framework for better trading off between profitability and budget utilization. Put another way, if we can demonstrate benefits of our methodology on Criteo's winning logs, then such results provide evidence that our methodology would lead to similar benefits in a real operational environment for Criteo.

Let us now describe how we constructed the problem instance from the dataset. We split the data so that the first three weeks of data are used for training and the last week for testing. The (impression type, campaign) graph $\mathcal{E}$ was constructed by first creating the impression types, then selecting the campaigns, and finally choosing the edges. To create impression types, we used the CART algorithm [25] with user clicks as labels and the Gini index as the impurity function. CART allows us to map each combination of the nine categorical features to a leaf in the CART tree, and each leaf corresponds to an impression type. Thus, CART provides a data-driven way to identify impression types such that impressions are grouped together in these impression types according to their click-through rates. Running CART requires tuning a "complexity parameter" which impacts the number of leaves of the tree. We used cross-validation to search for a complexity parameter with nearly optimal validation error while also producing a computationally treatable number of impression types (more about how CART was run in Appendix A.4). We then filtered the campaigns to those that appeared in at least two hundred impressions logs in both the train and test logs. We added an edge (i, k) to $\mathcal{E}$ only if there were at least thirty logs in which a selected campaign k bid for an impression of type $i \in \mathcal{I}$. Following the procedure described above, the constructed graph has 84 impression types, 649 campaigns, and 9903 edges. We leave the details on how to obtain the campaigns' budgets, bid landscapes, price paid per click, click-through rates and other parameters to Appendix A.4. In Appendix A.4 we also explain how the simulator scheme shown in Appendix A.4 was implemented for this experiment.

We run three experiments, each comparing our methodology with UF 3, UF 4 and UF 5 to the generalized greedy policy described in Policy 4. In order to test robustness of our methodology, we tried three configurations for the price paid per click values ℓ_k. In the first experiment, for each campaign we multiplied the original values of ℓ_k by 0.5, in the second by 0.75, and in the third by 1.0. In all cases we use the same budget values m_k. Changing the price paid per click affects the spending of campaigns, thereby Algorithm 3 needs to

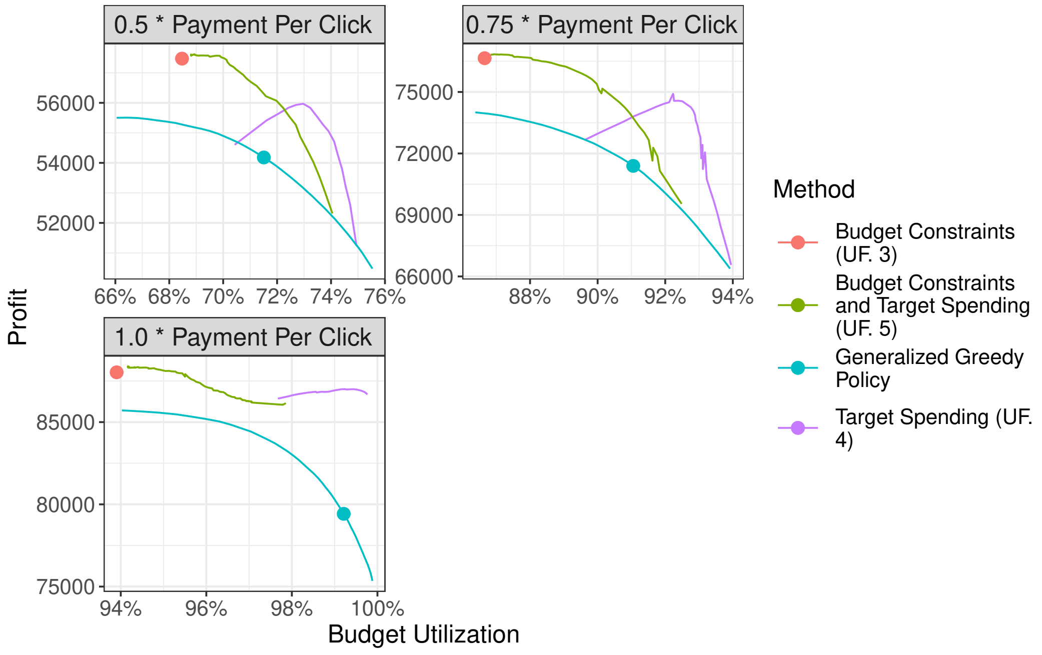


Figure 2.3: Profit and budget utilization as we change the penalization parameter τ of UF 4 and UF 5, and the scaling parameter γ of Policy 4. The dots in the graph represent UF 3, which has no parameters, and Policy 4 without scaling ($\gamma = 1$).

re-calculate the allocation and bid vector $(\mathbf{x}, \mathbf{b})$ to achieve desired trade off between profit and budget utilization. To obtain the curves shown in Figure 2.3 we tried different τ parameter values for UF 4, UF 5 and different γ values for Policy 4. Appendix A.4 shows confidence band plots showing results for all parameters tried.

Figure 2.3 again plots the total profit versus budget utilization for the four different methods. Figure 2.3 shows that UF 4 and UF 5 strictly dominate Policy 4 by achieving a higher profit at any budget utilization level. Furthermore, we see that this relationship holds across different budget utilization levels in the three plots. A primary reason why our methodology can dominate the generalized greedy policy is that it allows campaigns to wait for more favourable impressions types. The latter is especially important for campaigns with relative high r_{ik} revenue terms. When an impression arrives, our methodology can decide to allocate the impression to campaigns which do not maximize the profit of it, saving the budget of "better" of campaigns for future opportunities.

As in the synthetic case, UF 4 and UF 5 perform differently from each other. Since UF 4 only considers the target budget as a penalization term, it performs poorly for small penalization levels, which correspond to the left side of the curves for UF 4 in the first two panels of Figure 2.3. In this case, the implied policies for UF 4 overspend heavily for some campaigns, while insufficiently spending for many others. These policies obtain both low profit and low budget utilization levels as campaigns are not allowed to bid more than their target budgets. As the penalization value increases, UF 4 improves in terms of both budget utilization and profit until the profit begins to go down, at which point the objective is simply placing too much weight on the ℓ_2 penalty term. On the other hand, for small penalization values, UF 5 behaves similarly to UF 3 (they match when no penalization is present), and a clear trade off between profitability and budget utilization. Comparing UF 4 and UF 5 directly to each other, we see in the first two graphs that 5 is better at lower budget utilization levels and 4 is better at higher budget utilization levels. The fact that UF 5 does not allow overspending, while UF 4 does is the reason why the latter tends to work better for higher budget utilization levels. As in the synthetic case, employing different target budget values m_k in the constraints than those used in the ℓ_2 term in the objective should improve the performance of UF 5.

2.5 Conclusion

We proposed an optimization formulation for the joint bidding and allocation problem faced by a demand-side platform. Our approach allows the DSP to efficiently and effectively trade off between profitability and budget utilization for its advertisers and works for arbitrary auction types. Our optimization formulation is non-convex, but we employ a dual approach that leads to an efficient two phase algorithm. We study general conditions under which there is zero duality gap and the two phase algorithm converges to an optimal solution of the non-convex problem. Experimentally, we observe that our methodology allows a DSP to better trade off profitability and budget spending on both synthetic problems and on a problem using data from a real DSP.

Our paper focuses heavily on the properties of a static optimization model, and we provide a provably efficient algorithm for solving this model. A clear and important extension is to study the theoretical properties (e.g., a regret bound type of analysis) of employing such a model in an online, real-time environment. Another related and important extension is to consider the effect of parameter learning during the real-time decision making process, and to develop an algorithm that balances exploitation (e.g, via our optimization model) with exploration.

Chapter 3

Joint Online Learning and Decision-making via Dual Mirror Descent

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., [59]), 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 [20, 23]. 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.

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 $\mathcal{O}(\sqrt{T})$ 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 $\mathcal{O}(\sqrt{T})$ 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 [84].

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.

Related Work

The problem of online revenue maximization under feasibility constraints has been mostly studied under the lens of online convex optimization [59]. While first studied on resource allocation problems under linear constraints [92, 38], 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 [38, 29] or an i.i.d. manner [123, 10], with the data being sampled from an unknown distribution in the latter case. Subgradient methods based on primal-dual schemes have gained attraction [38, 68, 29, 128] 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 [91, 68, 29, 128]. On the i.i.d. setting and under linear constraints, [10] 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, [10] 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 [10]. Note that our first contribution, the incorporation of arbitrary revenue and cost functions, was simultaneously obtained by the same set of authors on [12].

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 [65] for the adversarial case). Assuming concave reward functions, [3] proposes an Upper-Confidence Bound type of algorithms which achieves sublinear rates of regret and constraint violations. [9] proposes a primal-dual algorithm to solve BwK with has a sublinear regret. By problem construction, their cost constraints are 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.

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 $\mathbb{1}(x \in A) := 1$ 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.

3.1 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 $\sup_{z\in\mathcal{Z},w\in\mathcal{W}} f(z;\theta^*,w) \le \bar{f}$ with $\bar{f} > 0$ and for the cost functions it holds that $\sup_{z\in\mathcal{Z},\theta\in\Theta,w\in\mathcal{W}} \|c(z;\theta,w)\|_\infty \le \bar{C}$ 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 θ^* 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:

$$\begin{aligned}(O):\quad \max_{(z^1,\dots,z^T)\in\mathcal{Z}^T} \quad & \sum_{t=1}^T f(z^t;\theta^*,w^t)\\ \text{s.t.}\quad & T\alpha\odot b \le \sum_{t=1}^T c(z^t;\theta^*,w^t) \le Tb \end{aligned} \tag{3.1}$$

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, (3.1) is a revenue maximization problem over the time horizon T with lower and upper cost constraints. Setting -1 as the lower bound for α_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 (3.1) 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 (3.1). However, as long as there is any positive probability of (3.1) 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 (3.1) 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:

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

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

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

and let us further define

$$\operatorname{OPT}(\mathcal{P}) := \max_{\gamma\in[0,1]} \operatorname{OPT}(\mathcal{P},\gamma). \tag{3.2}$$

Note that $\operatorname{OPT}(\mathcal{P},0)$ is exactly the expected optimal value of the hindsight problem (3.1). On the other hand, $\operatorname{OPT}(\mathcal{P},1)$ corresponds to a deterministic approximation of (3.1) 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 $\operatorname{OPT}(\mathcal{P},0)$ and the expectation problem $\operatorname{OPT}(\mathcal{P},1)$. Importantly, the benchmark we consider is $\operatorname{OPT}(\mathcal{P})$, which considers the *worst case* between these two extremes. It is possible to have $\operatorname{OPT}(\mathcal{P}) = \operatorname{OPT}(\mathcal{P},0)$, $\operatorname{OPT}(\mathcal{P}) = \operatorname{OPT}(\mathcal{P},1)$, $\operatorname{OPT}(\mathcal{P}) = \operatorname{OPT}(\mathcal{P},\gamma)$ for some $\gamma \in (0,1)$, and $\operatorname{OPT}(\mathcal{P}) = -\infty$. It is also possible to have a unique γ that maximizes $\operatorname{OPT}(\mathcal{P},\gamma)$ as well as infinitely many such maximizers. Examples of all of these possibilities are included in the supplementary materials.

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 3.1, 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 ξ_t is a mean zero random variable that is allowed to depend on w^t but is independent of everything else.

Let A_θ 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

$$\begin{aligned} \theta^t &= A_\theta\left(\mathcal{H}^{t-1}\right), \\ z^t &= A_z\left(f(\cdot;\theta^t,w^t), c(\cdot;\theta^t,w^t), \mathcal{H}^{t-1}\right) \end{aligned} \tag{3.3}$$

We refer to (A_z, A_θ) as A when no confusion is possible. Note that an important special case is when A_θ outputs θ^* for all inputs, which is the case where θ^* is known. Algorithm 5, which alternates between an online learning step using A_θ 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.

Algorithm 5 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 $t = 1, \dots, T$ do
  1. Update $\theta^t \leftarrow A_\theta(\mathcal{H}^{t-1})$.
  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 $b_k^{t+1} < \bar{C}$ for any $k \in [K]$, break.
end for
```

Note that Steps 4. and 5. of Algorithm 5 ensure that the total cost incurred is always less than or equal to bT, which ensures that the upper bound constraints in (3.1) 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 (3.3). 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.

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 θ, which is a key component of the analysis and algorithms proposed in this work. In particular, we first demonstrate in Proposition 3.1.1 that $D(\cdot; \theta^*)$ can be used to obtain an upper bound on $\text{OPT}(\mathcal{P})$.

Proposition 3.1.1. *For any $\lambda \in \Lambda$, it holds that* $\text{OPT}(\mathcal{P}) \leq T D(\lambda; \theta^*)$.

Next, Proposition 3.1.2 demonstrates that a stochastic estimate of a subgradient of $D(\cdot; \theta)$ can be easily obtained during the sequence of events described in Algorithm 5.

Proposition 3.1.2. *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 λ, 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 σ_1-strongly convex in the ℓ_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 λ, λ' in Λ. 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 6 presents the main algorithm of this work. Algorithm 6 is a specific instance of the more general algorithmic scheme, presented in Algorithm 5, where we fill in the generic decision making subroutine A_z with a dual stochastic mirror descent [59, 18] 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_θ is left as a generic subroutine; the regret bounds that we prove in Section 3.2 hold for any learning algorithm A_θ and naturally get better when A_θ has better convergence properties.

Algorithm 6 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 $\theta^t \leftarrow A_\theta(\mathcal{H}^{t-1})$.
    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. 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 $\tilde{g}_k^t \leftarrow -c_k(z^t; \theta^t, w^t) + b_k(\mathbb{1}(\lambda_k \geq 0) + \alpha_k \mathbb{1}(\lambda_k < 0))$ 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**
```

Note that Proposition 3.1.2 ensures that $\tilde{g}^t$ from Step 6. of Algorithm 6 is a stochastic subgradient of $D(\cdot; \theta^t)$ at λ^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 $\lambda_k^t \leftarrow \lambda^t \exp(-\eta_t \tilde{g}_k^t)$ [5]. 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 6 fills in the gap for A_z using mirror descent, the learning algorithm A_θ 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 $f' : \mathcal{Z} \times \mathcal{W} \to \mathbb{R}^p$ such that $f(z; \theta, w) = f'(z; w)^T \theta$ for $(z, \theta, w) \in \mathcal{Z} \times \Theta \times \mathcal{W}$ and we observe both $f(z^t; \theta^*, w^t)$ and $f'(z^t; w^t)$ at time t. Then, one could use $(f^s(z^s; \theta^*, w^s), f'(z^s; w^s))_{s=1}^{t-1}$ to fit a linear model (possibly with regularization) for implementing A_θ at time t. Depending on the underlying structure of the problem and randomness of the data arrivals, the previous methods may not converge to θ^*. Different ways of applying Step 1. are shown for a linear contextual bandits problem in Section 3.3. The performance of the different implementations vary drastically depending on the underlying randomness of the data arrivals.

3.2 Regret Bound and Related Results

In this section, we present our main theoretical result, Theorem 3.2.1, which shows regret bounds for Algorithm 6. In particular, the regret of Algorithm 6 can be decomposed as the summation of two parts: *(i)* the terms that appear when θ^* 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 θ^*. We also discuss the proof strategy for Theorem 3.2.1. Finally, for each lower bound constraint in (3.1), we prove that our algorithm may violate this lower bound by at most $\mathcal{O}(\sqrt{T})$ plus terms that depend on how θ^t converges to θ^*.

Regret Bound

Before presenting our main theorem, we need to establish a few more ingredients of the regret bound. First, we present Assumption 3.2.1, which can be thought of as a boundedness assumption on the dual iterates.

Assumption 3.2.1 (Bounded Dual Iterates)**.** *There is an absolute constant $C_h > 0$ such that the dual iterates $\{\lambda^t\}$ of Algorithm 6 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 3.2.1 may be thought of as a type of boundedness condition. After stating our regret bound, we present a sufficient condition for Assumption 3.2.1, 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 6 up to period t. Then, Step 4. of Algorithm 6 is intrinsically related to the following stopping time with respect to $\mathcal{H}^{t-1}$.

Definition 3.2.1 (Stopping time)**.** *Define τ_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, σ_1 refers to the strong convexity constant of $h(\cdot)$. We are now ready to state Theorem 3.2.1, which presents our main regret bound.

Theorem 3.2.1. *Let A denote Algorithm 6 with a constant "step-size" rule $\eta_t \leftarrow \eta$ for all $t \geq 1$ where $\eta > 0$. Suppose that Assumption 3.2.1 holds. Then, for any distribution $\mathcal{P}$ over $w \in \mathcal{W}$, it holds that* $\mathrm{Regret}(A|\mathcal{P}) \leq \Delta_{\mathrm{DM}} + \Delta_{\mathrm{Learn}}$ *where*

$$\begin{aligned}\Delta_{\mathrm{DM}} &:= \frac{2(\bar{C}^2+\bar{b}^2)}{\sigma_1}\eta\mathbb{E}[\tau_A] + \frac{1}{\eta}V_h(0,\lambda^1) + \frac{\bar{f}}{\underline{b}}\left(\bar{C} + \frac{C_h + \|\nabla h(\lambda^1)\|_\infty}{\eta}\right)\\ \Delta_{\mathrm{Learn}} &:= \mathbb{E}\left[\sum_{t=1}^{\tau_A}(c(z^t;\theta^*,w^t) - c(z^t;\theta^t,w^t))^T\lambda^t\right]\\ &\quad + \frac{\bar{f}}{\underline{b}}\left\|\mathbb{E}\left[\sum_{t=1}^{\tau_A} c(z^t;\theta^*,w^t) - c(z^t;\theta^t,w^t)\right]\right\|_\infty .\end{aligned}$$

Theorem 3.2.1 states that the regret of Algorithm 6 can be upper bounded by the sum of two terms: *(i)* a quantity Δ_{DM} that relates to the properties of the decision-making algorithm, dual mirror descent, and *(ii)* a quantity Δ_{Learn} that relates to the convergence of the learning algorithm A_θ. It is straightforward to see that setting $\eta \leftarrow \gamma/\sqrt{T}$ for some constant parameter $\gamma > 0$ implies that Δ_{DM} is $O(\sqrt{T})$. In the pure online optimization case, θ^* is known and hence $\theta^t = \theta^*$ for all $t \in [T]$ yielding $\Delta_{\mathrm{Learn}} = 0$. Thus, using $\eta \leftarrow \gamma/\sqrt{T}$ in the pure online optimization case yields $\mathrm{Regret}(A|\mathcal{P}) \leq O(\sqrt{T})$ and extends results presented by [10]. More generally, Δ_{Learn} depends on the convergence of θ^t to θ^*. Under a stricter version of Assumption 3.2.1 and assuming the cost functions are Lipschitz in θ, we demonstrate in the supplementary materials that Δ_{Learn} is $O(\mathbb{E}[\sum_{t=1}^{\tau_A}\|\theta^t - \theta^*\|_\theta])$.

Let us now return to Assumption 3.2.1 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 3.2.2. *For a given $\theta \in \Theta$, 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 3.2.1.

Proposition 3.2.1. *Assume that there exists $\delta > 0$ such that $\delta_\theta \geq \delta$ for all $\theta \in \Theta$, and let $C^{\rhd} := 2(\eta\frac{(\bar{C}^2+\bar{b}^2)}{\sigma_1} + \bar{f})$. 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 $C_h \leq \max\{\|\lambda^1\|_\infty, \sqrt{2}\sqrt{0.5(C^{\rhd}/\delta)^2 + \eta C^{\rhd}}\}$.*

Proof Sketch and Cost Feasibility

The proof sketch for Theorem 3.2.1 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 θ^t converges to θ^* appear. Second, we bound the expected regret up to period τ_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 τ_A plus the trivial bound $\bar{f}\mathbb{E}[T - \tau_A]$. (Note that the two major steps of our proof mimic those of [10] 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 θ^*). A solution obtained using Algorithm 6 can not overspend, but may underspend. Proposition 3.2.2 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 (3.1) grow. In the general case, terms depending on how θ^t converges to θ^* again appear, analogously to Theorem 3.2.1.

Proposition 3.2.2. *Assume we run Algorithm 6 under Assumption 3.2.1 using $\eta_t = \eta$ for all $t \geq 1$. For any $k \in [K]$ with $\alpha_k \neq -\infty$ it holds:*

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

3.3 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 [39]. 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 θ^*, *i.e.*, implementations of Step 1. of Algorithm 6, may be more or less effective depending on the inherent randomness of the data arrivals.

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 '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 (3.1)):

$$\begin{aligned} &\max_{(z^t,x^t)\in\mathcal{R}_+\times\mathcal{X}:t\in[T]} \quad \sum_{t=1}^T \sum_{k=1}^K (r_k(w^t) - \mathrm{mp}^t) x_k^t \mathbb{1}(z^t \geq \mathrm{mp}^t) \\ &\text{s.t.} \;\; T\alpha \odot b \leq \sum_{t=1}^T r(w^t) \odot x^t \mathbb{1}(z^t \geq \mathrm{mp}^t) \leq Tb. \end{aligned}$$

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 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 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 7 shows.

Algorithm 7 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 [39]. 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

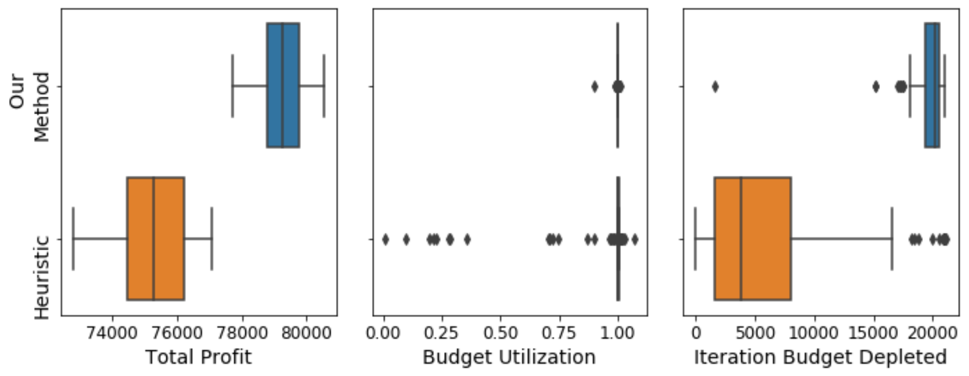


Figure 3.1: Box plots of the total profit obtained, and average budget utilization and budget depletion iteration per advertiser over 100 simulations. Budget utilization corresponds to the percentage of the total budget that an advertiser spent. If an advertiser never depleted its budget, its depletion time equals the simulation length.

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 [98]. 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 6 (more experimental details in the supplement).

Figure 3.1 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.

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 ρ 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 θ^* is an unknown parameter. We assume that $r^t = (W^t_{i(t)})^T \theta^* + \epsilon$ with ϵ 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, $f(z; \theta, W^t) = ((W^t_1)^T\theta, \ldots, (W^t_d)^T\theta)^T z$, and $c(z; \theta, W^t) = (\rho, \ldots, \rho) \odot z$. Even when $\mathcal{Z}$ allows probabilistic allocations, there is always a solution of Step 3. of Algorithm 6 which takes at most one action per period.

Experiment Details. We tried eight combinations of $d \times n$, run Algorithm 6 using $T = 1000$, 5000, 10000, use $\rho = 4$, and run 100 simulations of each experiment setting. Each simulation uses a unique seed to create θ^* and the mean matrix W by sampling i.i.d. Uniform$(-0.5, 0.5)$ random variables. Both θ^* 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 ξ^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 6 using subgradient descent.

We implemented Step 1. of Algorithm 6 in the following ways. 1. Gaussian Thompson-Sampling as in [4]. 2. Least-squares estimation. 3. Ridge regression estimation. 4. Ridge regression estimation plus a decaying randomized perturbation. 5. 'Known θ^*'. The last method represents the case of a pure online optimization problem. We also solve (3.1) 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 3.1 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^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 θ^*' method matches $\text{OPT}(\mathcal{P})$ as the optimal action is always the same.)

Table 3.1 shows interesting patterns. First, Thompson Sampling implemented as in [4]

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 θ^*	100	100	99.9	96.7	96.7	96.8

Table 3.1: 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 $t \in [T]$.

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 6 should depend on the application and randomness. Finally, the results shown in Table 3.1 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.

Chapter 4

Stochastic In-Face Frank-Wolfe Methods for Non-Convex Optimization and Sparse Neural Network Training

The Frank-Wolfe method (also called the conditional gradient method) and its extensions are often especially applicable in several areas of machine learning due to their low iteration costs and convenient structural properties. The Frank-Wolfe method has classically been applied and analyzed in the setting of smooth, constrained convex optimization problems; for a partial list of references in this setting, see [41, 36, 40] for older references and see [66, 57, 42, 43] and the references therein for more recent work. At each iteration, the basic Frank-Wolfe method relies only on a single gradient evaluation and a single call to a linear optimization subroutine, wherein the method computes a minimizer of the linear approximation of the objective function over the feasible region and then updates the next iterate as a convex combination of this minimizer and the current iterate.

In this paper, we consider variants of the Frank-Wolfe method for non-convex stochastic optimization problems with mixed constrained and unconstrained variables. Our problem of interest is:

$$\begin{array}{rll} F^* := & \min\limits_{x,y} & F(x,y) := \mathbb{E}_{z\sim\mathcal{D}}[f(x,y,z)] \\ & \text{s.t.} & x \in S, y \in \mathbb{R}^q \ , \end{array} \tag{4.1}$$

where $S \subseteq \mathbb{R}^p$ is a compact and convex set, z is a random variable in a probability space $\mathcal{Z}$ with (possibly unknown) distribution $\mathcal{D}$, and $f(\cdot,\cdot,\cdot) : S \times \mathbb{R}^q \times \mathcal{Z} \to \mathbb{R}$ is differentiable in (x,y) for each fixed $z \in \mathcal{Z}$. Note that we allow for the possibility of either $p = 0$ or $q = 0$, in which case only one of the two sets of variables x and y would be present in (4.1). Herein, we develop and analyze several stochastic gradient algorithms that utilize Frank-Wolfe "style" steps in the x variables and standard steepest descent steps in the y variables.

In many core machine learning methodologies, such as the setting of training neural networks, non-convexity is ubiquitous. Moreover, due to large training set sizes, stochastic algorithms (or related strategies) are also a necessity. For several reasons, including increased interpretability, memory efficiency, and improved computation at prediction/inference time, *structured models* (such as sparse networks, low-rank models, etc.) are often highly desirable. In order to induce a structured model, one might consider a strategy such as pruning [1, 93, 56] that modifies the model after the training procedure or one might consider a strategy that induces structured models throughout the training procedure. In this paper, we consider a method that falls into the latter approach based on extending the Frank-Wolfe method to problem (4.1). The Frank-Wolfe method, which falls into the more general class of "structure-enhancing" algorithms, is particularly attractive because the dynamics of the algorithm directly helps to promote near-optimal well-structured (e.g, sparse, low-rank) solutions. In some optimization formulations, such well-structured solutions also lie on low-dimensional faces of the feasible region, which was a key motivation for the development of "in-face" directions (also referred to as alternative directions herein), including away steps [52, 80], and the in-face extended Frank-Wolfe method developed in [43] for the case of deterministic, smooth convex optimization and particularly matrix completion.

In this paper, we extend the methodology of the Frank-Wolfe method with in-face directions to the setting of stochastic non-convex optimization, and we also allow for the possibility of mixed structured and unstructured variables. In other words, the x variables in (4.1) represent the variables that we would like to be well-structured (e.g., sparse edges in a neural network) and the y variables are completely "free." In Section 4.1, we develop a "hybrid" Frank-Wolfe steepest descent (FW-SD) method with alternative in-face direction steps that promote structured solutions in the x variables. Although we refer to this as a single method, we prove computational guarantees for two versions: the simple version without alternative direction steps, and the version with alternative direction steps. In the non-convex setting, the "Frank-Wolfe gap" function is often used to measure convergence (see, e.g., [79, 106]). We introduce a novel modification of the Frank-Wolfe gap that accounts for the mixed constrained and unconstrained variable structure, and all of our theoretical computational guarantees are stated in terms of the modified Frank-Wolfe gap. In particular, if K denotes the total number of iterations and when the number of samples per iteration is $O(K)$, we demonstrate $O(1/K)$ convergence in terms of the expected squared modified Frank-Wolfe gap (i.e., its second moment) for the methods developed herein. In Section 4.2, we extend our method to problems with block coordinate structure. The computational guarantees for the block version is in the worst-case the same as the guarantee for the non-block method, however in practice the block method is effective due to its ability to use different step-sizes for each block. Section 4.3 presents the results of some numerical experiments on the MNIST and CIFAR-10 datasets demonstrating the viability of our algorithm.

Stochastic gradient methods (also stochastic approximation) dates back to [110]. For recent works related to stochastic gradient descent and its variants see, e.g., [94], [21], [82], [22], and the references therein. Mostly closely related to our work, at least in terms of the theoretical computational guarantees developed herein, is perhaps [106] who study stochastic Frank-Wolfe methods with and without variance reduction in the non-convex setting. In Section 4.1 we comment on how our results relate to [106]. [46] also studies Frank-Wolfe type methods for stochastic non-convex problems with a composite structure. [61] also studies, in the case of convex and related variational inequality problems, Frank-Wolfe type methods with a related "semi-proximal" decomposable structure. Block coordinate Frank-Wolfe type methods have been studied in several contexts beginning with [81]. Some other related references examining variants and extensions of Frank-Wolfe method in the deterministic setting are [79, 69, 76, 96, 105, 80], and in the stochastic convex setting are [60, 47, 89].

Illustrative application: sparse neural network training. Let us conclude the introduction by describing an illustrative application to sparse neural network training. In general, it has been observed that structured neural networks are practically advantageous for several reasons. Networks with desirable structural properties, that are often tailored to the application in mind, are more efficient – both computationally and statistically – than general purpose feedforward networks. A general approach for conceptualizing and training well-structured networks is through edge weight sparsity. Indeed, a feedforward network with sparse edges offers a number of benefits including increased interpretability, reduced memory footprint, and reduced computation at prediction/inference time. In fact, in practical applications, inference time and memory footprint are sometimes major bottlenecks that are often overlooked during the training/designing of deep neural networks [26]. Finally, sparse networks offer a conceptual advantage in that they encompass several popular representations, such as the widely popular convolutional layers. For more discussion on the benefits of sparse networks and approaches for constructing them, see [117, 88], for example.

The formulation for training a sparse neural network considered herein is based on ℓ_1 regularization, which is a natural and widely popular idea for promoting sparsity as well as other benefits of regularization [58]. For simplicity, let us consider training a fully connected feedforward network in the general setting of supervised learning. (This model can easily be extended to an arbitrary directed acylic graph.) We consider a per-node regularization model where including an ℓ_1 regularization constraint is optional for each node, hence we define the set $\mathcal{N}_R := \{(t,i) : \text{ node } i \text{ in layer } t \text{ imposes a regularization constraint}\}$ and we let $\delta_{t,i}$ denote the corresponding regularization parameter. The optimization model we consider is:

$$\min_w \frac{1}{n}\sum_{i=1}^n \ell(\hat{y}_w(x_i), y_i) \ , \quad \text{s.t. } \ \|w_i^{t-1}\|_1 \leq \delta_{i,t} \ \text{ for all } (t,i) \in \mathcal{N}_R \ , \tag{4.2}$$

where $(x_1, y_1), \ldots, (x_n, y_n)$ is the training data (in this section only we use x and y to refer to data, in later sections they refer to the optimization variables), $\hat{y}_w(\cdot) : \mathbb{R}^d \to \mathbb{R}^l$ denotes the prediction function of the model parameterized by the collection of weights w, and $\ell(\cdot,\cdot) : \mathbb{R}^l \times \mathbb{R}^l \to \mathbb{R}$ is a differentiable loss function. Here w_i^{t-1} is the vector of incoming edge weights at node i in layer t.

We note that a very closely related model, albeit with less flexibility, has been studied in the improper learning setting by [131]. Let us now point out a few salient features of the optimization model (4.2). First, note that the variables of (4.2) can be partitioned into constrained variables and unconstrained variables, corresponding to x and y in (4.1), respectively. Furthermore, the constrained variables in (4.2) have a block coordinate decomposable structure, wherein each vector $w_{t-1,i}$ for $(t,i) \in \mathcal{N}_R$ is constrained only to lie in its own

ℓ_1 ball of radius $\delta_{i,t}$. This type of block coordinate decomposable structure is considered herein in Section 4.2. Note that the ℓ_1 ball constraints are intended to promote sparsity, and therefore algorithmic schemes that also promote sparsity, such as in-face directions, are highly desirable in this context. It is possible to also model additional types of network structures with different types of convex constraints that are amenable to the Frank-Wolfe method and its extensions. For example, node level sparsity can be modeled with group ℓ_1 constraints and low-rank weight matrices between can be modeled with nuclear norm constraints.

4.1 Stochastic Frank-Wolfe steepest descent method with in-face directions

Let us now return to studying the generic non-convex stochastic optimization problem (4.1) where x is constrained to lie in a compact and convex set S and y is unconstrained. As mentioned, our algorithm is based on using Frank-Wolfe steps in the x variables and steepest descent steps in the y variables (both with stochastic versions of the partial gradients). Let us first review some useful notation.

Notation. Let $\|\cdot\|_X$ be a given norm on the variables $x \in \mathbb{R}^p$, and let $\|\cdot\|_Y$ be a given norm on the variables $y \in \mathbb{R}^q$. The diameter of S is $\text{diam}(S) := \max_{x,\bar{x}\in S} \|x - \bar{x}\|_X$, and recall that $\text{diam}(S) < +\infty$ since S is bounded. The dual norms associated with $\|\cdot\|_X$ and $\|\cdot\|_Y$ are denoted by $\|\cdot\|_{X*}$ and $\|\cdot\|_{Y*}$, respectively. Recall that $\|\cdot\|_{X*}$ is defined by $\|s\|_{X*} := \max_{x:\|x\|\le 1} s^T x$ and $\|\cdot\|_{Y*}$ is defined analogously. We also use $\|\cdot\|$ to denote the "Euclidean combination" of the two norms $\|\cdot\|_X$ and $\|\cdot\|_Y$ as the norm on $(x, y) \in \mathbb{R}^p \times \mathbb{R}^q$, whereby $\|(x, y)\| := \sqrt{\|x\|_X^2 + \|y\|_Y^2}$. Note that the dual norm of $\|\cdot\|$ is also the "Euclidean combination" of $\|\cdot\|_{X*}$ and $\|\cdot\|_{Y*}$, whereby $\|(s, t)\|_* = \sqrt{\|s\|_{X*}^2 + \|t\|_{Y*}^2}$. The standard inner product between $s \in \mathbb{R}^p$ and $x \in \mathbb{R}^p$ is denoted by $s^T x$, and that the inner product on $\mathbb{R}^p \times \mathbb{R}^q$ is the sum of the inner products on the two spaces, i.e, $(s, t)^T (x, y) := s^T x + t^T y$. The notation ∇ refers to gradients with respect to (x, y), and ∇_x and ∇_y refers to partial gradients with respect to x and y, respectively. For a scalar α, $\text{sgn}(\alpha)$ is the sign of α, which is equal to -1 if $\alpha < 0$, $+1$ if $\alpha > 0$ and 0 if $\alpha = 0$. The notation "$\tilde{v} \leftarrow \arg\max_{v\in S}\{f(v)\}$" denotes assigning $\tilde{v}$ to be an arbitrary optimal solution of the problem $\max_{v\in S}\{f(v)\}$.

Assumptions. Note that the choice of the norm $\|\cdot\|_Y$ directly affects the form of the steepest descent step. For example, if $\|\cdot\|_Y$ is the ℓ_2 norm then the steepest descent step becomes a standard stochastic gradient step. Another relevant example is when $\|\cdot\|_Y$ is the ℓ_1 norm, in which case the steepest descent step becomes a stochastic variant of a greedy coordinate descent step (see, e.g., [97]). On the other hand, the choice of the norm $\|\cdot\|_X$ does not affect the direction of the Frank-Wolfe step in the x variables but it does affect the step-size strategy employed herein.

We make the following assumptions regarding problem (4.1):

(A1) The objective function $F(\cdot,\cdot)$ is smooth, i.e., there is a constant $L_\nabla > 0$ such that $\|\nabla F(x, y) - \nabla F(\bar{x}, \bar{y})\|_* \le L_\nabla \|(x, y) - (\bar{x}, \bar{y})\|$ for all $x, \bar{x} \in S$ and $y, \bar{y} \in \mathbb{R}^q$.

(A2) The partial gradient with respect to x is uniformly bounded, i.e., there is a constant $L_f > 0$ such that $\|\nabla_x f(x, y, z)\|_{X*} \leq L_f$ for all $x \in S, y \in \mathbb{R}^q$, and $z \in \mathcal{Z}$.

(A3) The stochastic gradient has bounded variance, i.e., there is a constant $\sigma \geq 0$ such that $\mathbb{E}_{z\sim D}\left[\|\nabla f(x, y, z) - \nabla F(x, y)\|_*^2\right] \leq \sigma^2$ for all $x \in S$ and $y \in \mathbb{R}^q$.

(A4) We have knowledge of the constant L_∇ as well as a constant $\bar{C} > 0$ satisfying $\bar{C} \geq \max\{2L_\nabla \cdot \text{diam}(S)^2, L_f \cdot \text{diam}(S)\}$.

Modified Frank-Wolfe gap. Since (4.1) is generally a non-convex problem, we measure convergence in terms of a modified version of the "Frank-Wolfe" gap function. Let us first define the function $\tilde{G}(\cdot, \cdot) : S \times \mathbb{R}^q \to \mathbb{R}_+$ by $\tilde{G}(\bar{x}, \bar{y}) := \max_{x\in S}\left\{\nabla_x F(\bar{x}, \bar{y})^T(\bar{x} - x)\right\}$. Note that when the y variables are not present, this is exactly the definition of the "gap function" due to [63] and studied in the recent literature on Frank-Wolfe. Definition 4.1.1 presents our modified gap function that accounts for both the x and y variables.

Definition 4.1.1. *The modified Frank-Wolfe gap function $G(\cdot, \cdot) : S \times \mathbb{R}^q \to \mathbb{R}_+$ is the function given by*

$$G(\bar{x}, \bar{y}) := \tilde{G}(\bar{x}, \bar{y})\sqrt{\frac{2L_\nabla}{\bar{C}}} \;+\; \|\nabla_y F(\bar{x}, \bar{y})\|_{Y*} \;,$$

where $\tilde{G}(\bar{x}, \bar{y}) := \max_{x\in S}\left\{\nabla_x F(\bar{x}, \bar{y})^T(\bar{x} - x)\right\}$.

Note that Definition 4.1.1 depends on the particular specification of the parameters L_∇ and $\bar{C}$, which is a slightly undesirable property. However, in the case when $\bar{C} = 2L_\nabla \cdot \text{diam}(S)^2$, then we have that $\sqrt{\frac{2L_\nabla}{\bar{C}}} = \frac{1}{\text{diam}(S)}$, which is a natural way to normalize the function $\tilde{G}(\cdot, \cdot)$. Note again that when the y variables are not present then the modified Frank-Wolfe gap reduces to a scaled version of the standard Frank-Wolfe gap, and when the x variables are not present then it reduces to the norm of the gradient (which is also a standard metric in unconstrained non-convex optimization). The use of the modified Frank-Wolfe gap is justified by Proposition 4.1.1 below, which states that $G(\bar{x}, \bar{y}) = 0$ is a *necessary condition* for any locally optimal solution $(\bar{x}, \bar{y})$.

Proposition 4.1.1. *Suppose that $(\bar{x}, \bar{y})$ is a locally optimal solution of problem* (4.1)*. Then, it holds that $G(\bar{x}, \bar{y}) = 0$.*

The proof of Proposition 4.1.1, as well as all other omitted proofs, is included in the supplementary materials. In the convex case, we can also use the modified Frank-Wolfe gap to bound the objective function value optimality gap, as demonstrated by Proposition 4.1.2 below.

Proposition 4.1.2. *Suppose that $F(\cdot, \cdot)$ is convex on $S \times \mathbb{R}^q$, and let (x^*, y^*) denote an optimal solution of* (4.1)*. Consider a given feasible solution $(\bar{x}, \bar{y}) \in S \times \mathbb{R}^q$, and let $R \geq 0$ be a constant such that $\|\bar{y} - y^*\|_Y \leq R$. Then, it holds that:*

$$F(\bar{x}, \bar{y}) - F^* \;\leq\; \max\left\{\sqrt{\tfrac{\bar{C}}{2L_\nabla}}, R\right\} \cdot G(\bar{x}, \bar{y}) \;.$$

Note that Proposition 4.1.2 requires existence of a constant $R \geq 0$ such that $\|\bar{y} - y^*\|_Y \leq R$. In the case that $(\bar{x}, \bar{y})$ corresponds to the iterate of some algorithm that is guaranteed to lie in a bounded initial level set of the function $F(\cdot, \cdot)$, then this constant R is guaranteed to exist. For example, this is always the case for deterministic steepest descent. This level set condition is not guaranteed to hold for the stochastic algorithms that we study herein but we would expect this condition to hold in practice (with high probability) after sufficiently many iterations. We also utilize Lemma 4.1.1 below, which relates a stochastic estimate of the modified gap function to the above definition.

Lemma 4.1.1. *Let $(\bar{x}, \bar{y}) \in S \times \mathbb{R}^q$ be given and let $(\hat{g}, \hat{h})$ denote an unbiased stochastic estimate of $\nabla F(\bar{x}, \bar{y})$ such that $\mathbb{E}[\hat{g}] = \nabla_x F(\bar{x}, \bar{y})$ and $\mathbb{E}[\hat{h}] = \nabla_y F(\bar{x}, \bar{y})$. Define the random variables:*

$$\tilde{G} := \max_{x \in S} \left\{ \hat{g}^T (\bar{x} - x) \right\} \ , \ \textit{and} \ \ \hat{G} := \tilde{G} \sqrt{\frac{2L_\nabla}{\bar{C}}} \ + \ \|\hat{h}\|_{Y*} \ .$$

Then, it holds that $\mathbb{E}[\hat{G}] \geq G(\bar{x}, \bar{y})$.

Stochastic Frank-Wolfe steepest descent (FW-SD) method with in-face directions

We are now ready to present our stochastic Frank-Wolfe steepest descent method for problem (4.1), which possibly incorporates "alternative directions" and is formally presented below in Algorithm 8. Algorithm 8 includes a true/false variable, called AlternativeDirections, which indicates whether to use the alternative direction step in the x variables or not. Each iteration of Algorithm first uses a stochastic estimate of the gradient $\nabla F(x_k, y_k)$ based on b_k i.i.d. samples to perform standard Frank-Wolfe and steepest descent steps in the variables x and y, respectively. Note that the step-sizes $\bar{\alpha}_k$ and α_k are dynamic random variables depending on the stochastic gradients $\hat{g}_k$ and $\hat{h}_k$, which is in contrast to the step-sizes (such as constant step-sizes in [106]) that have been previously considered in the literature on stochastic Frank-Wolfe methods in the non-convex setting. The dynamic step-sizes employed by Algorithm 8 are more "adaptive" than constant step-sizes and hence can have better practical performance. If alternative directions are not used, then x_{k+1} is simply determined by the stochastic Frank-Wolfe step. Otherwise, for the version with alternative directions, Step (4.) is the computation of the stochastic alternative direction step, which is based on a fresh stochastic gradient estimate in Step (4a.). Then, Step (4b.) represents the computation of a generic alternative direction d_k and the corresponding step, which we elaborate on further below. Finally, note that the output of Algorithm 8 is chosen uniformly at random from all past iterates, which is also equivalent to randomly sampling the total number of iterations prior to starting the algorithm.

Theorem 4.1.1 below presents our main computational guarantee for the stochastic Frank-Wolfe steepest descent method for the non-convex optimization problem (4.1). The statement of the theorem involves the maximum ratios between the Euclidean norm $\|\cdot\|_2$ and the given

Algorithm 8 Stochastic Frank-Wolfe steepest descent (FW-SD) Method with alternative directions

```
**Initialize** at $x_0 \in S$, $y_0 \in \mathbb{R}^q$ $k \leftarrow 0$, set AlternativeDirections $\in \{\text{TRUE}, \text{FALSE}\}$.
**At iteration $k$:**
1. Choose number of samples $b_k$, sample $z_{k,1}, \ldots, z_{k,b_k}$ i.i.d. from $\mathcal{D}$ and compute:
    $\hat{g}_k \leftarrow \frac{1}{b_k} \sum_{i=1}^{b_k} \nabla_x f(x_k, y_k, z_{k,i})$
    $\hat{h}_k \leftarrow \frac{1}{b_k} \sum_{i=1}^{b_k} \nabla_y f(x_k, y_k, z_{k,i})$
2. Do Stochastic Frank-Wolfe Step:
    $\tilde{x}_k \leftarrow \arg\min_{x \in S} \{\hat{g}_k^T x\}$
    $\tilde{G}_k \leftarrow \hat{g}_k^T (x_k - \tilde{x}_k)$
    $\bar{x}_k \leftarrow x_k + \bar{\alpha}_k(\tilde{x}_k - x_k)$ where $\bar{\alpha}_k := \tilde{G}_k / \bar{C}$
    **If AlternativeDirections = FALSE, then set** $x_{k+1} \leftarrow \bar{x}_k$
3. Do Stochastic steepest descent Step:
    $\tilde{y}_k \leftarrow \arg\max_{y \in \mathbb{R}^q} \{\hat{h}_k^T y : \|y\|_Y \leq 1\}$
    $y_{k+1} \leftarrow y_k - \alpha_k \tilde{y}_k$ where $\alpha_k := \|\hat{h}_k\|_{Y*} / 2L_\nabla$
4. **If AlternativeDirections = TRUE, then do** Stochastic Alternative Direction Step:
    4a. Sample $\check{z}_{k,1}, \ldots, \check{z}_{k,b_k}$ i.i.d. from $\mathcal{D}$ and compute
        $\check{g}_k \leftarrow \frac{1}{b_k} \sum_{i=1}^{b_k} \nabla_x f(\bar{x}_k, y_{k+1}, \check{z}_{k,i})$
    4b. Compute a stochastic alternative direction $d_k$ (formally a measurable function of $\check{g}_k$)
        satisfying $\check{g}_k^T d_k < 0$ and $\|d_k\|_X \leq \text{diam}(S)$, and set:
        $A_k := -\check{g}_k^T d_k$
        $\alpha_k^{\text{stop}} := \arg\max_{\alpha \geq 0} \{\alpha : \bar{x}_k + \alpha d_k \in S\}$
        $x_{k+1} \leftarrow \bar{x}_k + \bar{\beta}_k d_k$ where $\bar{\beta}_k := \min\left\{A_k/\bar{C}, \alpha_k^{\text{stop}}\right\}$
**After $K$ total iterations:**
**Output:** $(\hat{x}_k, \hat{y}_k)$ chosen uniformly at random from $(x_0, y_0), \ldots, (x_K, y_K)$
```

norm $\|\cdot\|$, defined by:

$$\kappa_1 := \max_{(x,y)\neq 0} \|(x,y)\|_2/\|(x,y)\| \ , \ \ \kappa_2 := \max_{(x,y)\neq 0} \|(x,y)\|/\|(x,y)\|_2 \ .$$

(Note that $\|(x,y)\|_2$ is simply defined by $\|(x,y)\|_2 := \sqrt{\|x\|_2^2 + \|y\|_2^2}$.) Let us also define $\kappa := \kappa_1\kappa_2$. Note that norm equivalence on finite dimensional vector spaces ensures that κ is finite, and in the case that $\|\cdot\|_X$ and $\|\cdot\|_Y$ are both the ℓ_2 norm then $\kappa = 1$. We also define a constant α_{AD} that is useful in the statement of the theorem as well as later results. Specifically, we define $\alpha_{\text{AD}} := 1$ if AlternativeDirections = FALSE and $\alpha_{\text{AD}} := 2$ if AlternativeDirections = TRUE.

Theorem 4.1.1. *Consider the Stochastic FW-SD Method, possibly with alternative directions (Algorithm 8). Under assumptions (A1)-(A4), it holds for all $K \geq 0$ that:*

$$\mathbb{E}[G(\hat{x}_K, \hat{y}_K)^2] \ \leq \ \frac{8L_\nabla(F(x_0,y_0) - F^*)}{K+1} \ + \ \frac{4\alpha_{\text{AD}}\kappa^2\sigma^2}{K+1}\sum_{k=0}^{K}\frac{1}{b_k} \ .$$

Based on Theorem 4.1.1, setting $b_k = K$ at each iteration of Algorithm 8 leads to an $O(1/K)$ convergence bound whereas setting $b_k = k$ would lead to an $O(\ln(K)/K)$ bound. Note that, as compared to previous related results for the Frank-Wolfe method in the non-convex case developed in [106], Theorem 4.1.1 obtains a similar bound but has a few differences. In addition to the novel extensions herein of including steepest descent steps in the y variables and possibly incorporating alternative direction steps in x variables, note that Theorem 4.1.1 holds for the dynamic step-size rule of Algorithm 8 whereas [106] studies a constant step-size rule. Also note that Theorem 4.1.1 bounds the second moment of the modified Frank-Wolfe gap whereas [106] bounds the first moment of the Frank-Wolfe gap. The proof of Theorem 4.1.1 is included in the supplement.

Examples of alternative "in-face" directions. Step (4b.) of Algorithm 8 is written in a purposefully generic way that does not specify precisely how to compute the alternative direction d_k so that we may consider a wide framework that accommodates several computationally advantageous choices. At the same time, the intuitive idea of the role of alternative directions in the convergence analysis of Algorithm 8 is that alternative directions should do no harm in terms of the modified Frank-Wolfe gap convergence. Let us now present several concrete examples of alternative directions, which all have the property of also being in-face directions. Given any feasible $x \in S$, we denote $\mathcal{F}_S(x)$ as the minimal face of S that contains the point x. Given $\bar{x}_k$ computed in Step (2.) of Algorithm 8, d_k is an in-face direction if $\bar{x}_k + \alpha d_k \in \mathcal{F}_S(\bar{x}_k)$ for all $\alpha \in [0, \alpha_k^{\text{stop}}]$. One possible in-face direction is the "away step" direction introduced in [52] obtained by choosing $d_k \leftarrow x_k - \check{x}_k$, where $\check{x}_k \leftarrow \arg\max_{x\in\mathcal{F}_S(\bar{x}_k)}\{\check{g}_k^T x\}$. Due to the facial structure of S, in-face directions often preserve certain types of solution structures. For example, when S is an ℓ_1 ball, then an in-face step preserves sparsity so that x_{k+1} has the the same signed sparsity pattern as $\bar{x}_k$. In-face directions, including away steps, are also often as simple to compute or even simpler to compute than the Frank-Wolfe directions. Another example of an in-face directions pertinent to the non-convex setting include a regular Frank-Wolfe direction inside $\mathcal{F}_S(\bar{x}_k)$. Additional discussion of how to compute in-face directions in the case where S is an ℓ_1 ball is included in the supplementary materials.

4.2 Block coordinate extension

In this section, we extend the previously developed stochastic FW-SD method with alternative directions to the block coordinate setting. We consider an extension of problem (4.1) where the variables x as well as the feasible region S for x have a block coordinate structure. Specifically, we presume that $x \in S \subseteq \mathbb{R}^p$ has a decomposable block coordinate structure across $N \geq 1$ total blocks, whereby $x = (x^{(1)}, \ldots, x^{(N)})$, $S = S_1 \times \cdots \times S_N$, and each $x^{(i)} \in S_i \subseteq \mathbb{R}^{p_i}$ where S_i is a compact and convex set and with $\sum_{i=1}^N p_i = p$. For each $i \in \{1, \ldots, N\}$, let $\|\cdot\|_{X,i}$ denote the given norm on the space of variables $x^{(i)} \in \mathbb{R}^{p_i}$ with dual norm denoted by $\|\cdot\|_{X*,i}$. The norm $\|\cdot\|_X$ on the overall space of $x = (x^{(1)}, \ldots, x^{(N)})$ variables is now taken to be the Euclidean combination of all of the block norms, i.e., we define $\|x\|_X := \sqrt{\sum_{i=1}^N \|x^{(i)}\|_{X,i}^2}$. Note that the overall norm on the entire space of variables (x, y) is the same as before, i.e., $\|(x, y)\| := \sqrt{\|x\|_X^2 + \|y\|_Y^2}$. Furthermore, recall that $\operatorname{diam}(S_i) := \max_{x^{(i)}, \bar{x}^{(i)} \in S} \|x^{(i)} - \bar{x}^{(i)}\|_{X,i}$. We use the notation $\nabla_x^{(i)}$ to refer to partial gradients with respect to $x^{(i)}$ for each $i \in \{1, \ldots, N\}$.

In this block coordinate setting, we retain the earlier assumptions (A1) and (A3) and modify assumptions (A2) and (A4) as follows:

(A2- B) For each block $i \in \{1, \ldots, N\}$, the partial gradient with respect to $x^{(i)}$ is uniformly bounded, i.e., there is a constant $L_{f,i} > 0$ such that $\|\nabla_x^{(i)} f(x, y, z)\|_{X*,i} \leq L_{f,i}$ for all $x \in S, y \in \mathbb{R}^q$, and $z \in \mathcal{Z}$.

(A4- B) We have knowledge of the constant L_∇ as well as constants $\bar{C}_i > 0$ satisfying $\bar{C}_i \geq \max\{2L_\nabla \cdot \operatorname{diam}(S_i)^2, L_{f,i} \cdot \operatorname{diam}(S_i)\}$ for each block $i \in \{1, \ldots, N\}$.

Notice that the decomposable structure of S, i.e., $S = S_1 \times \cdots \times S_N$ implies that linear optimization problems are completely separable across the N different blocks and that the modified Frank-Wolfe gap also has a similar decomposable structure. For each block $i \in \{1, \ldots, N\}$, let us define $\tilde{G}_i(\cdot, \cdot) : S \times \mathbb{R}^q \to \mathbb{R}_+$ by $\tilde{G}_i(\bar{x}, \bar{y}) := \max_{x^{(i)} \in S_i} \left\{ \nabla_x^{(i)} F(\bar{x}, \bar{y})^T (\bar{x}^{(i)} - x^{(i)}) \right\}$. Then, the function $\tilde{G}(\cdot, \cdot)$ defined in Section 4.1 satisfies $\tilde{G}(\bar{x}, \bar{y}) = \sum_{i=1}^N \tilde{G}_i(\bar{x}, \bar{y})$ for all $x \in S$ and $y \in \mathbb{R}^q$. Moreover, in light of Assumption (A4B), we have that $\sum_{i=1}^N \bar{C}_i \geq 2L_\nabla \sum_{i=1}^N \operatorname{diam}(S)_i^2 = 2L_\nabla \operatorname{diam}(S)^2$. Hence, we define $\bar{C} := \sum_{i=1}^N \bar{C}_i$, which is needed to specify the modified Frank-Wolfe gap function.

The main idea of the block coordinate version of Algorithm 8 is to replace the stochastic Frank-Wolfe step in Step (2.) and the alternative direction step in Step (4.) with block coordinate versions that use different step-sizes in each of the different blocks. Subroutines 9 and 10 below precisely describe how the block variants of these two steps work.

```
Subroutine 9 Block Coordinate Stochastic Frank-Wolfe Step
  For each $i = 1, \dots, N$, set:
    $\tilde{x}_k^{(i)} \leftarrow \arg\min_{x^{(i)} \in S_i} \{(\hat{g}_k^{(i)})^T x^{(i)}\}$
    $\tilde{G}_k^i \leftarrow (\tilde{g}_k^{(i)})^T (x_k^{(i)} - \tilde{x}_k^{(i)})$
    $\bar{x}_k^{(i)} \leftarrow x_k^{(i)} + \bar{\alpha}_k^i (\tilde{x}_k^{(i)} - x_k^{(i)})$ where $\bar{\alpha}_k^i := \tilde{G}_k^i / \bar{C}_i$.
```

```
Subroutine 10 Block Coordinate Stochastic Alternative Direction Step
  4a. Sample $\check{z}_{k,1}, \dots, \check{z}_{k,b_k}$ i.i.d. from $\mathcal{D}$ and compute
    $\check{g}_k \leftarrow \frac{1}{b_k} \sum_{i=1}^{b_k} \nabla_x f(\bar{x}_k, y_{k+1}, \check{z}_{k,i})$
  4b. Compute a stochastic alternative direction $d_k$ (formally a measurable function of
  $\check{g}_k$) satisfying $(\check{g}_k^{(i)})^T d_k^{(i)} < 0$ and $\|d_k^{(i)}\|_{X,i} \leq \mathrm{diam}(S_i)$ for all $i \in \{1, \dots, N\}$. For each
  $i \in \{1, \dots, N\}$, set:
    $A_k^i := -(\check{g}_k^{(i)})^T d_k^{(i)}$
    $\alpha_k^{\mathrm{stop},i} := \arg\max_{\alpha \geq 0} \{\alpha : \bar{x}_k^{(i)} + \alpha d_k^{(i)} \in S_i\}$
    $x_{k+1}^{(i)} \leftarrow \bar{x}_k^{(i)} + \bar{\beta}_k^i d_k^{(i)}$ , $\bar{\beta}_k^i := \min\{A_k^i / \bar{C}_i, \alpha_k^{\mathrm{stop},i}\}$
```

Theorem 4.2.1. *Consider the Block Coordinate Stochastic FW-SD Method, possibly with alternative directions, i.e., Algorithm 8 with Step (2.) replaced with Subroutine 9 and Step (4.) replaced with Subroutine 10. Under assumptions (A1), (A2B), (A3), and (A4B), it holds for all $K \geq 0$ that:*

$$\mathbb{E}[G(\hat{x}_K, \hat{y}_K)^2] \;\leq\; \frac{8L_\nabla(F(x_0, y_0) - F^*)}{K+1} \;+\; \frac{4\alpha_{\mathrm{AD}}\kappa^2\sigma^2}{K+1} \sum_{k=0}^{K} \frac{1}{b_k} \;,$$

where the modified Frank-Wolfe gap $G(\cdot, \cdot)$ (Definition 4.1.1) is defined using $\bar{C} := \sum_{i=1}^N \bar{C}_i$.

4.3 Numerical Experiments

Let us now discuss our illustrative numerical experiments wherein we applied the block coordinate version Algorithm 8 studied in Section 4.2 to the ℓ_1 regularized neural network training problem (4.2) on both synthetic and real datasets. We used PyTorch [101] to write an optimizer that partitions the layers into: *(i)* Frank-Wolfe layers whose weights correspond to the x variables in (4.1), and *(ii)* SGD layers whose weights correspond to the y variables in (4.1) and with the ℓ_2 norm used for the steepest descent steps. For the type of in-face direction in the Frank-Wolfe layers, we used away steps on the ℓ_1 ball as described earlier and elaborated on further in the supplementary materials. We initialize the weights of the Frank-Wolfe layers in such a way that each node has at least one non-zero edge coming in and another coming out. Since the Lipschitz constant may not be known in practice, we used cross validation on a held out validation set to tune the parameter L_∇ over the range $L_\nabla = 4^i$ with $i \in \{-1, 0, \ldots, 6\}$. Finally, since our method is not much more complex than SGD and is supported by rigorous computational guarantees, our experiments are intended to be illustrative. In particular, we would like to illustrate the potential advantages of incorporating Frank-Wolfe layers on top of layers that use standard SGD or SGD variants (e.g., momentum). Therefore, we only perform comparisons with the basic SGD method which uses the standard PyTorch initialization. We also try both variants of the block coordinate version of Algorithm 8, referred to as SFW (Stochastic Frank-Wolfe Steepest Descent without alternative in-face directions) and SFW-IF (Stochastic Frank-Wolfe Steepest Descent with alternative in-face directions) herein. (Note that, out of fairness with respect to the number of stochastic gradient calls, we allow SFW to have twice as many iterations as SFW-IF by counting each iteration of Algorithm 8 as two iterations in the case when AlternativeDirections = TRUE.) Finally, note that all methods were run for 25 epochs using a batch size of 250 data points.

We experimented with a multilayer perceptron and a convolutional network for MNIST and a convolutional network for CIFAR-10. The convolutional networks for MNIST and CIFAR-10 were taken from PyTorch tutorials [104, 103], while the multilayer perceptron is simply a three layer network taken from a Keras tutorial [75]. For the multilayer perceptron MNIST example, we treat the first two layers as Frank-Wolfe layers. For the convolutional CIFAR-10 and MNIST examples, we treated the convolutional layers as SGD layers and the next two dense layers after the convolutional layers as Frank-Wolfe layers. The bias terms are always incorporated into the SGD variables. For SFW and SFW-IF, we cross validated δ separately for each layer on a grid of values $\{1, 5, 10, 50, 100\}$. Each of the examples has two Frank-Wolfe layers, and for each of these layers we report the average percent of non-zero edges going into each node (we consider any value less than 0.001 to be 0) of the solutions returned after 25 epochs by the three methods. For the same solutions returned by the three methods, we examined how the test accuracy is affected when we do hard thresholding to retain only the top $\theta\%$ of largest magnitude edges in the Frank-Wolfe layers. The results are displayed in Table 4.1, which shows that SFW-IF and SFW outperform SGD in these two metrics. All experiments show that SFW and SFW-IF are more robust to hard-thresholding than SGD. For example, when we zero out 95% of the entries in MNIST-MLP the solution found by SFW-IF only losses 0.4% of accuracy, while SGD loses more than 17%. Interestingly, on the convolutional networks, SFW can be more robust than SFW-IF for very small values of the hard-thresholding parameter θ. Also, the results show that both SFW and SFW-IF find solutions in which most of the weight entries have very small values (< 0.001), while SGD simply does not promote this behaviour. We also performed experiments on synthetically generated data which are described in detail in the supplementary materials.

	MNIST and CIFAR-10 Results								
	MNIST-MLP			**MNIST-Conv**			**CIFAR-10**		
Metric	**SFW-IF**	**SFW**	**SGD**	**SFW-IF**	**SFW**	**SGD**	**SFW-IF**	**SFW**	**SGD**
Layer 1 Avg. NNZ (%)	10.05	7.26	97.28	9.71	1.69	97.23	29.27	15.12	98.08
Layer 2 Avg. NNZ (%)	1.55	0.73	97.79	27.34	13.08	98.78	7.27	13.26	98.90
Accuracy (%) w/ Top 100%	96.88	96.49	98.25	98.79	98.50	99.11	54.72	53.12	57.76
Accuracy (%) w/ Top 50%	96.88	96.49	98.12	98.79	98.50	99.07	54.71	53.12	55.69
Accuracy (%) w/ Top 25%	96.88	96.46	97.73	98.82	98.50	98.63	54.33	53.12	49.04
Accuracy (%) w/ Top 10%	96.73	96.22	96.80	98.58	98.49	96.44	44.82	52.57	37.08
Accuracy (%) w/ Top 5%	96.49	94.52	81.12	90.8	98.06	84.38	32.68	49.5	23.63

Table 4.1: Comparison in terms of accuracy and percentage of non-zero terms between our proposed Stochastic Frank-Wolfe (SFW) method both with and without In-Face directions (IF) with respect to a traditional subgradient descent approach using the MNIST and CIFAR-10 datasets.

Appendix A

Optimal Bidding, Allocation, and Budget Spending for a Demand-Side Platform with Generic Auctions

A.1 Summary of Notation

Table A.1 provides a summary of the notation used herein.

A.2 Omitted Proofs

Proofs for Section 2.2

Proof of Proposition 2.2.1:

Proof. Let $\mathcal{P} := \{(\mathbf{x}, \mathbf{b}) \in \mathcal{S} : u(\mathbf{v}(\mathbf{x}, \mathbf{b})) > -\infty\}$. By the assumption that $\mathrm{dom}(u(\cdot)) \cap \mathcal{V} \neq \emptyset$, we have that $\mathcal{P} \neq \emptyset$ as well. Note also that we have $\mathcal{P} = \{(\mathbf{x}, \mathbf{b}) \in \mathcal{S} : \mathbf{v}(\mathbf{x}, \mathbf{b}) \in \mathrm{dom}(u(\cdot))\}$, which is the inverse image of a closed set $\mathrm{dom}(u(\cdot))$ (by part *(i)* of Assumption 2.2.1) under a continuous function $\mathbf{v}(\cdot, \cdot)$. By a standard result in real analysis this implies that $\mathcal{P}$ is closed, and hence compact since $\mathcal{S}$ is bounded. Note also that $u(\mathbf{v}(\cdot, \cdot))$ is continuous on $\mathcal{P}$ by part *(ii)* of Assumption 2.2.1. Now, we have that:

$$F^* = \sup_{(\mathbf{x},\mathbf{b}) \in \mathcal{S}} \{\pi(\mathbf{x}, \mathbf{b}) + u(\mathbf{v}(\mathbf{x}, \mathbf{b}))\} = \sup_{(\mathbf{x},\mathbf{b}) \in \mathcal{P}} \{\pi(\mathbf{x}, \mathbf{b}) + u(\mathbf{v}(\mathbf{x}, \mathbf{b}))\} .$$

By the previous discussion, the expression on the right side of the above is a supremum of a continuous function over a compact set and therefore the Weierstrass Theorem ensures that F^* is finite and attained. □

Proof of Proposition 2.2.2:

Proof. Let $i \in \mathcal{I}$ and $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$ be given. For ease of notation, define the function $\phi_i(\cdot,\cdot;\boldsymbol{\lambda}) : \mathcal{X}_i \times \mathcal{B}_i \to \mathbb{R}$ by $\phi_i(\mathbf{x}_i, \mathbf{b}_i; \boldsymbol{\lambda}) := \pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$. By the reasoning in (2.7), we have that

$$\phi_i(\mathbf{x}_i, \mathbf{b}_i; \boldsymbol{\lambda}) \;=\; \sum_{k \in \mathcal{K}_i} h_i(b_{ik}; r_{ik}(1-\lambda_k)) s_i x_{ik} \;\text{ for all } \mathbf{x}_i \in \mathcal{X}_i \text{ and } \mathbf{b}_i \in \mathcal{B}_i \;. \tag{A.1}$$

By the definition of $b_i^*(\cdot)$ in Assumption 2.2.3, we have that

$$\tilde{\pi}_{ik}(\boldsymbol{\lambda}) = h_i(b_{ik}^*(\boldsymbol{\lambda}); r_{ik}(1-\lambda_k)) = h_i(b_i^*(r_{ik}(1-\lambda_k)); r_{ik}(1-\lambda_k)) \geq h_i(b_{ik}; r_{ik}(1-\lambda_k)) \;\text{ for all } b_{ik} \in [0, \bar{b}_i] \;.$$

Let us now fix $\mathbf{x}_i \in \mathcal{X}_i$ arbitrarily. Since $x_{ik} \geq 0$ for all $k \in \mathcal{K}_i$ and since $s_i \geq 0$, the above inequality implies that that $\tilde{\pi}_{ik}(\boldsymbol{\lambda}) s_i x_{ik} \geq h_i(b_{ik}; r_{ik}(1-\lambda_k)) s_i x_{ik}$ for all $b_{ik} \in [0, \bar{b}_i]$; summing these inequalities over $k \in \mathcal{K}_i$ and using (A.1) as well as the fact that $\mathbf{x}_i$ was selected arbitrarily yields:

$$\phi_i(\mathbf{x}_i, \mathbf{b}_i^*(\boldsymbol{\lambda}); \boldsymbol{\lambda}) \;=\; \sum_{k \in \mathcal{K}_i} \tilde{\pi}_{ik}(\boldsymbol{\lambda}) s_i x_{ik} \;\geq\; \phi_i(\mathbf{x}_i, \mathbf{b}_i; \boldsymbol{\lambda}) \;\text{ for all } \mathbf{x}_i \in \mathcal{X}_i \text{ and } \mathbf{b}_i \in \mathcal{B}_i \;. \tag{A.2}$$

It is straightforward to see that the greedy selection in Step (2.) of Algorithm 2 leads to an optimal solution of the corresponding linear optimization problem over the simplex-like set $\mathcal{X}_i$ with coefficients $\tilde{\pi}_{ik}(\boldsymbol{\lambda}) s_i x_{ik}$, i.e., it holds that $\mathbf{x}_i^*(\boldsymbol{\lambda}) \in \arg\max_{\mathbf{x}_i \in \mathcal{X}_i} \left\{ \sum_{k \in \mathcal{K}_i} \tilde{\pi}_{ik}(\boldsymbol{\lambda}) s_i x_{ik} \right\}$. Therefore (A.2) implies that

$$\phi_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda}); \boldsymbol{\lambda}) \;\geq\; \phi_i(\mathbf{x}_i, \mathbf{b}_i^*(\boldsymbol{\lambda}); \boldsymbol{\lambda}) \;\geq\; \phi_i(\mathbf{x}_i, \mathbf{b}_i; \boldsymbol{\lambda}) \;\text{ for all } \mathbf{x}_i \in \mathcal{X}_i \text{ and } \mathbf{b}_i \in \mathcal{B}_i \;, \tag{A.3}$$

from which we conclude that $(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \arg\max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \left\{ \pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i) \right\}$.

Finally, to see that $\mathbf{g}_i := -\mathbf{v}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \partial q_i(\boldsymbol{\lambda})$, let $\bar{\boldsymbol{\lambda}} \in \mathbb{R}^{|\mathcal{K}|}$ be given and note that:

$$\begin{aligned}
q_i(\bar{\boldsymbol{\lambda}}) &= \max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \left\{ \pi_i(\mathbf{x}_i, \mathbf{b}_i) - \bar{\boldsymbol{\lambda}}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i) \right\} \\
&\geq \pi_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) - \bar{\boldsymbol{\lambda}}^\top \mathbf{v}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) \\
&= \pi_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) + (\boldsymbol{\lambda} - \bar{\boldsymbol{\lambda}})^\top \mathbf{v}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}), \mathbf{b}_i^*(\boldsymbol{\lambda})) \\
&= q_i(\boldsymbol{\lambda}) + \mathbf{g}_i^\top (\bar{\boldsymbol{\lambda}} - \boldsymbol{\lambda}) \;.
\end{aligned}$$

□

Proof of Theorem 2.3.1

The proof of Theorem 2.3.1 involves several intermediate results. Recall the notation $\mathcal{V} = \{\mathbf{v}(\mathbf{x}, \mathbf{b}) : (\mathbf{x}, \mathbf{b}) \in \mathcal{S}\}$ and $\text{dom}(u(\cdot)) = \{\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|} : u(\mathbf{v}) > -\infty\}$. We start with Lemma A.2.1, which demonstrates that Slater's Condition, $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$, is a sufficient condition to guarantee the existence of a dual optimal solution $\boldsymbol{\lambda}^*$. Throughout this subsection, $\|\cdot\|$ denotes the ℓ_2-norm $\|\cdot\|_2$.

Lemma A.2.1. *Suppose that problem* (2.1) *satisfies Slater's Condition,* $\mathrm{int}(\mathrm{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$*. Then, the dual objective function* $Q(\cdot) = q(\cdot) + p(\cdot)$ *has compact level sets and there exists an optimal solution* $\boldsymbol{\lambda}^*$ *of the dual problem* (2.6)*.*

Proof. Let us first argue that $Q(\cdot) = q(\cdot) + p(\cdot)$ is a proper, lower semi-continuous, convex function. Equivalently, this means that $Q(\cdot)$ is a proper and closed convex function. Recall that $q(\cdot)$ is defined by $q(\boldsymbol{\lambda}) := \max_{(\mathbf{x},\mathbf{b})\in\mathcal{S}} \left\{\pi(\mathbf{x},\mathbf{b}) - \boldsymbol{\lambda}^\top \mathbf{v}(\mathbf{x},\mathbf{b})\right\}$ for any $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$. It follows that $q(\cdot)$ is convex and $\mathrm{dom}(q(\cdot)) = \mathbb{R}^{|\mathcal{K}|}$, hence $q(\cdot)$ is globally continuous. Moreover $p(\cdot)$, defined by $p(\boldsymbol{\lambda}) := \sup_{\mathbf{v}\in\mathbb{R}^{|\mathcal{K}|}} \left\{\boldsymbol{\lambda}^\top \mathbf{v} + u(\mathbf{v})\right\}$ for any $\lambda \in \mathbb{R}^{|\mathcal{K}|}$, is the convex conjugate of $-u(\cdot)$ and as such is lower semi-continuous. Additionally, by part *(i)* of Assumption 2.2.1, we have that $\mathrm{dom}(u(\cdot))$ is non-empty and therefore it must be the case that $\mathrm{dom}(p(\cdot))$ is non-empty as well. Therefore, we have that $Q(\cdot) = q(\cdot) + p(\cdot)$ is lower semi-continuous and $\mathrm{dom}(Q(\cdot)) = \mathrm{dom}(p(\cdot))$ is non-empty, which means that $Q(\cdot)$ is also proper.

Now let $\alpha \in \mathbb{R}$ be given and consider the level set $\mathcal{L}_\alpha := \{\boldsymbol{\lambda} : Q(\boldsymbol{\lambda}) \leq \alpha\}$. Since, as argued above, $Q(\cdot)$ is a closed convex function, we have that $\mathcal{L}_\alpha$ is a closed set. It remains to demonstrate that $\mathcal{L}_\alpha$ is also bounded. By way of contradiction, suppose that there exists a sequence $\{\boldsymbol{\lambda}^t\}_{t=0}^{\infty}$ such that $\boldsymbol{\lambda}^t \in \mathcal{L}_\alpha$ for all $t \geq 0$ and $\|\boldsymbol{\lambda}^t\| \to \infty$. By Slater's Condition, $\mathrm{int}(\mathrm{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$, we have that there exists $(\bar{\mathbf{x}}, \bar{\mathbf{b}}) \in \mathcal{S}$ and $\varepsilon > 0$ such that, for all $\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|}$, it holds that:

$$\|\mathbf{v} - \mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}})\| < \varepsilon \ \Rightarrow \ u(\mathbf{v}) > -\infty \ . \tag{A.4}$$

Let us define $\mathbf{v}^t := \mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + \frac{(\varepsilon/2)\boldsymbol{\lambda}^t}{\|\boldsymbol{\lambda}^t\|}$ for all $t \geq 0$. Then, by (A.4), we have that $u(\mathbf{v}^t) > -\infty$. Additionally, we have that:

$$\begin{aligned}
\pi(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + u(\mathbf{v}^t) \ &\leq \ \pi(\bar{\mathbf{x}}, \bar{\mathbf{b}}) - {\boldsymbol{\lambda}^t}^\top \mathbf{v}^t + p(\boldsymbol{\lambda}^t) \\
&= \ \pi(\bar{\mathbf{x}}, \bar{\mathbf{b}}) - {\boldsymbol{\lambda}^t}^\top \mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}}) - {\boldsymbol{\lambda}^t}^\top \left(\frac{(\varepsilon/2)\boldsymbol{\lambda}^t}{\|\boldsymbol{\lambda}^t\|}\right) + p(\boldsymbol{\lambda}^t) \\
&= \ \pi(\bar{\mathbf{x}}, \bar{\mathbf{b}}) - {\boldsymbol{\lambda}^t}^\top \mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + p(\boldsymbol{\lambda}^t) - (\varepsilon/2)\|\boldsymbol{\lambda}^t\| \\
&\leq \ q(\boldsymbol{\lambda}^t) + p(\boldsymbol{\lambda}^t) - (\varepsilon/2)\|\boldsymbol{\lambda}^t\| \\
&= \ Q(\boldsymbol{\lambda}^t) - (\varepsilon/2)\|\boldsymbol{\lambda}^t\| \\
&\leq \ \alpha - (\varepsilon/2)\|\boldsymbol{\lambda}^t\| \ ,
\end{aligned}$$

where the first inequality uses $u(\mathbf{v}^t) = \inf_{\boldsymbol{\lambda}\in\mathbb{R}^{|\mathcal{K}|}} \left\{-\boldsymbol{\lambda}^\top \mathbf{v}^t + p(\boldsymbol{\lambda})\right\}$, the second inequality uses $q(\boldsymbol{\lambda}^t) = \max_{(\mathbf{x},\mathbf{b})\in\mathcal{S}} \left\{\pi(\mathbf{x},\mathbf{b}) - {\boldsymbol{\lambda}^t}^\top \mathbf{v}(\mathbf{x},\mathbf{b})\right\}$, and the third inequality uses $\boldsymbol{\lambda}^t \in \mathcal{L}_\alpha$. In particular, since $\|\boldsymbol{\lambda}^t\| \to \infty$, the above chain of inequalities implies that $\{u(\mathbf{v}^t)\}_{i=0}^{\infty}$ is unbounded below.

Now, note that $\|\mathbf{v}^t - \mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}})\| = \varepsilon/2$ for all $t \geq 0$. Hence, by the Bolzano-Weierstrass Theorem, we assume without loss of generality that there exists $\tilde{\mathbf{v}} \in \mathbb{R}^{|\mathcal{K}|}$ with $\|\tilde{\mathbf{v}} - \mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}})\| = \varepsilon/2$ such that $\mathbf{v}^t \to \tilde{\mathbf{v}}$. Then, (A.4) implies that $u(\tilde{\mathbf{v}}) > -\infty$. Moreover, by the continuity of $u(\cdot)$ on $\mathrm{dom}(u(\cdot))$ stated in Assumption 2.2.1, we have that $u(\mathbf{v}^t) \to u(\tilde{\mathbf{v}})$. However,

$u(\mathbf{v}^t) \to u(\tilde{\mathbf{v}})$ contradicts the previous conclusion that $\{u(\mathbf{v}^t)\}_{i=0}^{\infty}$ is unbounded below. Therefore, it must be the case that $\mathcal{L}_\alpha$ is a bounded set.

Finally, letting $\bar{\boldsymbol{\lambda}} \in \mathrm{dom}(Q(\cdot))$ be given, we have that the level set $\mathcal{L}_{Q(\bar{\boldsymbol{\lambda}})}$ is a non-empty and compact set. By the extension of the Weierstrass Extreme Value Theorem to lower semi-continuous functions, we have that $Q(\cdot)$ attains its minimum value on $\mathcal{L}_{Q(\bar{\boldsymbol{\lambda}})}$ at some $\boldsymbol{\lambda}^* \in \mathcal{L}_{Q(\bar{\boldsymbol{\lambda}})}$. By the definition of $\mathcal{L}_{Q(\bar{\boldsymbol{\lambda}})}$, $\boldsymbol{\lambda}^*$ must also then be the global minimizer of $Q(\cdot)$ and hence is an optimal solution of the dual problem (2.6). □

The next lemma demonstrates that the IMC Condition (Definition 2.3.1) is a special case of the UBP Condition (Definition 2.3.2).

Lemma A.2.2. *Suppose that impression type $i \in \mathcal{I}$ satisfies the Increasing Marginal Cost (IMC) Condition. Then, impression type i also satisfies the Unique Bid Price (UBP) Condition.*

Proof. Let $r \in \mathbb{R}$ be given and recall the definition of the expected profit function is given by $h_i(b; r) := [r - \beta_i(b)]\rho_i(b) = r\rho_i(b) - c_i(b)$, where $c_i(b) = \rho_i(b)\beta_i(b)$. By Assumption 2.2.2 we have that $h_i(b; r)$ is continuous in b on $[0, \bar{b}_i]$, and by the IMC Condition we have that $h_i(b; r)$ is differentiable on $(0, \bar{b}_i)$. Taking the derivative with respect to b yields $h_i'(b; r) = \rho_i'(b)(r - g_i(b))$ for all $b \in (0, \bar{b}_i)$, where recall that $g_i(b) := \frac{c_i'(b)}{\rho_i'(b)}$. The second part of the IMC Condition states that $g_i(b)$ is strictly increasing on $(0, \bar{b}_i)$, which implies that $r - g_i(b)$ is strictly decreasing on $(0, \bar{b}_i)$. Moreover, the first part of IMC Condition states that $\rho_i'(b) > 0$ on $(0, \bar{b}_i)$. Thus, these two properties imply that $h_i'(b; r)$ must satisfy exactly one of the following three possibilities: *(i)* $h_i'(b; r) > 0$ for all $b \in (0, \bar{b}_i)$, *(ii)* $h_i'(b; r) < 0$ for all $b \in (0, \bar{b}_i)$, or *(iii)* there exists $\hat{b} \in (0, \bar{b}_i)$ such that $h_i'(b; r) > 0$ for all $b \in (0, \hat{b})$, $h_i'(\hat{b}; r) = 0$, and $h_i'(b; r) < 0$ for all $b \in (\hat{b}, \bar{b}_i)$. By continuity of $h_i(b; r)$ on $[0, \bar{b}_i]$, case *(i)* implies that $\bar{b}_i$ is the unique maximizer of $h_i(b; r)$ on $[0, \bar{b}_i]$, while case *(ii)* implies that 0 is the unique such maximizer. Finally, in case *(iii)* it is clear that $\hat{b}$ is the unique such maximizer. In all cases, there is a unique maximizer of $h_i(b; r)$ on $[0, \bar{b}_i]$ and thus the UBP condition is satisfied. □

Recall that the part of the dual function associated with impression type $i \in \mathcal{I}$ is $q_i(\cdot) : \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R}$, which is defined by $q_i(\boldsymbol{\lambda}) := \max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \left\{\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)\right\}$, where we also define $\mathcal{X}_i := \{\mathbf{x}_i \in \mathbb{R}^{|\mathcal{K}_i|} \ : \ \sum_{k \in \mathcal{K}_i} x_{ik} \leq 1, \text{ and } x_{ik} \geq 0 \text{ for all } k \in \mathcal{K}_i\}$, $\mathcal{B}_i := [0, \bar{b}_i]^{|\mathcal{K}_i|}$, and $\mathcal{S}_i := \mathcal{X}_i \times \mathcal{B}_i$. Let us additionally define $\mathcal{S}_i^*(\boldsymbol{\lambda}) := \arg\max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \left\{\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)\right\}$. The next lemma is simply a statement of Danskin's Theorem [19, p. 737] as well as the subdifferential sum rule [111, p. 223] in our setting.

Lemma A.2.3. *For any $i \in \mathcal{I}$ and $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$, it holds that:*

$$\partial q_i(\boldsymbol{\lambda}) = \mathrm{conv}\left\{-\mathbf{v}_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \ : \ (\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \in \mathcal{S}_i^*(\boldsymbol{\lambda})\right\} \ . \tag{A.5}$$

Note that $\partial q_i(\boldsymbol{\lambda})$ is non-empty for all $i \in \mathcal{I}$ and $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$. Moreover, for any $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$, it holds that $\partial Q(\boldsymbol{\lambda})$ is non-empty if and only if $\partial p(\boldsymbol{\lambda})$ is non-empty and:

$$\partial Q(\boldsymbol{\lambda}) = \sum_{i \in \mathcal{I}} \partial q_i(\boldsymbol{\lambda}) \ + \partial p(\boldsymbol{\lambda}) \ . \tag{A.6}$$

Proof. For ease of notation, recall the function $\phi_i(\cdot,\cdot,\cdot) : \mathcal{X}_i \times \mathcal{B}_i \times \mathbb{R}^{|\mathcal{K}|} \to \mathbb{R}$ defined by $\phi_i(\mathbf{x}_i, \mathbf{b}_i, \boldsymbol{\lambda}) := \pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$, and note that $\mathcal{S}_i^*(\boldsymbol{\lambda}) := \arg\max_{(\mathbf{x}_i, \mathbf{b}_i) \in \mathcal{S}_i} \{\phi_i(\mathbf{x}_i, \mathbf{b}_i; \boldsymbol{\lambda})\}$. Observe that $\phi_i(\cdot,\cdot,\cdot)$ is jointly continuous in all of its arguments by Assumption 2.2.2, and is linear (hence convex and differentiable) in $\boldsymbol{\lambda}$ for any fixed $(\mathbf{x}_i, \mathbf{b}_i)$. Moreover, $\nabla_{\boldsymbol{\lambda}} \phi_i(\mathbf{x}_i, \mathbf{b}_i, \boldsymbol{\lambda}) = -\mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$ is also continuous in $(\mathbf{x}_i, \mathbf{b}_i)$ by Assumption 2.2.2. Thus, the conditions of Danskin's Theorem [19, p. 737] are satisfied and we may conclude that (A.5) holds.

The fact that $\partial q_i(\boldsymbol{\lambda})$ is non-empty follows form continuity of $\phi_i(\cdot,\cdot,\cdot)$ and compactness of $\mathcal{S}_i$. Since $\mathrm{dom}(p(\cdot))$ is guaranteed to be non-empty and $\mathrm{dom}(q_i(\cdot)) = \mathbb{R}^{|\mathcal{K}|}$ for all $i \in \mathcal{I}$, we may apply Theorem 23.8 of [111, p. 223] to conclude that (A.6) holds for any $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$. Thus, it must be the case that $\partial Q(\boldsymbol{\lambda})$ is non-empty if and only if $\partial p(\boldsymbol{\lambda})$ is non-empty. □

We will now strengthen the above lemma when the UBP Condition holds. As defined in Section 2.3, recall that whenever the UBP Condition holds we use the notation $\mathbf{b}_i^*(\boldsymbol{\lambda})$ to denote the vector in $\mathbb{R}^{|\mathcal{K}_i|}$ whose kth component is equal to $b_i^*(r_{ik}(1-\lambda_k)) = \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r_{ik}(1-\lambda_k))$, the unique maximizer of the profit expression $h_i(b; r_{ik}(1-\lambda_k)) = [r_{ik}(1-\lambda_k) - \beta_i(b)]\rho_i(b)$ on $[0, \bar{b}_i]$. Let us also define $\mathcal{X}_i^*(\boldsymbol{\lambda}; \mathbf{b}_i) := \arg\max_{\mathbf{x}_i \in \mathcal{X}_i} \left\{\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)\right\}$ as the maximizing set of allocation variables $\mathbf{x}_i$ in the definition of $q_i(\boldsymbol{\lambda})$ given that the bid price variables are fixed at $\mathbf{b}_i$. The following lemma provides a more precise characterization of $\partial q_i(\boldsymbol{\lambda})$ under the UBP Condition.

Lemma A.2.4. *Suppose that impression type $i \in \mathcal{I}$ satisfies the Unique Bid Price (UBP) Condition. Then, for any $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$, it holds that:*

$$\partial q_i(\boldsymbol{\lambda}) = \{-\mathbf{v}_i(\check{\mathbf{x}}_i, \mathbf{b}_i^*(\boldsymbol{\lambda})) \ : \ \check{\mathbf{x}}_i \in \mathcal{X}_i^*(\boldsymbol{\lambda}; \mathbf{b}_i^*(\boldsymbol{\lambda}))\} \ . \tag{A.7}$$

Furthermore, for any $\check{\mathbf{x}}_i \in \mathcal{X}_i^*(\boldsymbol{\lambda}; \mathbf{b}_i^*(\boldsymbol{\lambda}))$, it holds that $(\check{\mathbf{x}}_i, \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \mathcal{S}_i^*(\boldsymbol{\lambda})$.

Proof. Let us define two sets of interest: $\check{\mathcal{V}}_i(\boldsymbol{\lambda}) := \{-\mathbf{v}_i(\check{\mathbf{x}}_i, \mathbf{b}_i^*(\boldsymbol{\lambda})) \ : \ \check{\mathbf{x}}_i \in \mathcal{X}_i^*(\boldsymbol{\lambda}; \mathbf{b}_i^*(\boldsymbol{\lambda}))\}$, which appears on the right-hand side of (A.7), and $\tilde{\mathcal{V}}_i(\boldsymbol{\lambda}) := \left\{-\mathbf{v}_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \ : \ (\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \in \mathcal{S}_i^*(\boldsymbol{\lambda})\right\}$, which appears inside the convex hull operation on the right-hand side of (A.5). We will now show that, under the UBP Condition, we have $\check{\mathcal{V}}_i(\boldsymbol{\lambda}) = \tilde{\mathcal{V}}_i(\boldsymbol{\lambda})$. First, let $\check{\mathbf{x}}_i \in \mathcal{X}_i^*(\boldsymbol{\lambda}; \mathbf{b}_i^*(\boldsymbol{\lambda}))$ be given. Regardless of the whether the UBP Condition holds or not, following exactly the same logic as in the proof of Proposition 2.2.2, particularly (A.3), yields that $(\check{\mathbf{x}}_i, \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \mathcal{S}_i^*(\boldsymbol{\lambda})$. Notice that this then implies that $\check{\mathcal{V}}_i(\boldsymbol{\lambda}) \subseteq \tilde{\mathcal{V}}_i(\boldsymbol{\lambda})$.

To see that $\tilde{\mathcal{V}}_i(\boldsymbol{\lambda}) \subseteq \check{\mathcal{V}}_i(\boldsymbol{\lambda})$, let $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \in \mathcal{S}_i^*(\boldsymbol{\lambda})$ be given. By the reasoning in (2.7), we have that

$$\pi_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \ = \ \sum_{k \in \mathcal{K}_i} h_i(\tilde{b}_{ik}; r_{ik}(1-\lambda_k)) s_i \tilde{x}_{ik} \ . \tag{A.8}$$

Let $k \in \mathcal{K}_i$ be given and let us consider two possible cases: *(i)* $\tilde{x}_{ik} > 0$ and *(ii)* $\tilde{x}_{ik} = 0$. In case *(i)*, since $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i)$ maximizes the expression in (A.8) and since $s_i > 0$ by assumption, it must be the case that $\tilde{b}_{ik} \in \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r_{ik}(1 - \lambda_k))$. Since the UBP Condition holds, this implies that $\tilde{b}_{ik} = b^*_{ik}(\boldsymbol{\lambda})$, where $b^*_{ik}(\boldsymbol{\lambda})$ denotes the k^{th} component of $\mathbf{b}^*_i(\boldsymbol{\lambda})$. Moreover, we therefore conclude that $v_{ik}(\tilde{x}_{ik}, \tilde{b}_{ik}) = v_{ik}(\tilde{x}_{ik}, b^*_{ik}(\boldsymbol{\lambda}))$ for all $k \in \mathcal{K}_i$ such that $\tilde{x}_{ik} > 0$. Furthermore, from (A.8), we then have:

$$\pi_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \;=\; \pi_i(\tilde{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda})) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\tilde{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda}))\ ,$$

which also implies that $\tilde{\mathbf{x}}_i \in \mathcal{X}^*_i(\boldsymbol{\lambda}; \mathbf{b}^*_i(\boldsymbol{\lambda}))$ since $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) \in \mathcal{S}^*_i(\boldsymbol{\lambda})$. In case *(ii)*, since $\tilde{x}_{ik} = 0$ we have that $v_{ik}(\tilde{x}_{ik}, \tilde{b}_{ik}) = r_{ik}\rho_i(\tilde{b}_{ik}) s_i \tilde{x}_{ik} = 0 = v_{ik}(\tilde{x}_{ik}, b^*_{ik}(\boldsymbol{\lambda}))$. Also, whenever $k \notin \mathcal{K}_i$, we have that $v_{ik}(\tilde{x}_{ik}, \tilde{b}_{ik}) = 0 = v_{ik}(\tilde{x}_{ik}, b^*_{ik}(\boldsymbol{\lambda}))$ by definition. Thus, we conclude that $\mathbf{v}_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) = \mathbf{v}_i(\tilde{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda}))$. Since we have already argued that $\tilde{\mathbf{x}}_i \in \mathcal{X}^*_i(\boldsymbol{\lambda}; \mathbf{b}^*_i(\boldsymbol{\lambda}))$, this implies that $-\mathbf{v}_i(\tilde{\mathbf{x}}_i, \tilde{\mathbf{b}}_i) = -\mathbf{v}_i(\tilde{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda})) \in \check{\mathcal{V}}_i(\boldsymbol{\lambda})$.

Now, since the expression $\pi_i(\mathbf{x}_i, \mathbf{b}_i) - \boldsymbol{\lambda}^\top \mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$ is linear in $\mathbf{x}_i$ given fixed values of $\mathbf{b}_i$ and $\boldsymbol{\lambda}$ and since $\mathcal{X}_i$ is a polytope, we have that $\mathcal{X}^*_i(\boldsymbol{\lambda}; \mathbf{b}^*_i(\boldsymbol{\lambda}))$ is a polytope (hence convex). Moreover since $\mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$ is a linear map in $\mathbf{x}_i$ given fixed values of $\mathbf{b}_i$, we have that $\check{\mathcal{V}}_i(\boldsymbol{\lambda}) := \{-\mathbf{v}_i(\check{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda}))\ :\ \check{\mathbf{x}}_i \in \mathcal{X}^*_i(\boldsymbol{\lambda}; \mathbf{b}^*_i(\boldsymbol{\lambda}))\}$ is a polytope and therefore convex as well. Thus, since $\check{\mathcal{V}}_i(\boldsymbol{\lambda}) = \tilde{\mathcal{V}}_i(\boldsymbol{\lambda})$ and by (A.5) we have that $\partial q_i(\boldsymbol{\lambda}) = \text{conv}(\tilde{\mathcal{V}}_i(\boldsymbol{\lambda})) = \check{\mathcal{V}}_i(\boldsymbol{\lambda})$, which is exactly what is stated in (A.7). □

We are now ready to complete the proof of Theorem 2.3.1.

Proof of Theorem 2.3.1:

Proof. By Lemma A.2.1, there exists an optimal solution $\boldsymbol{\lambda}^*$ of the dual problem (2.6) and, by the general optimality conditions for convex problems, we have that $\mathbf{0} \in \partial Q(\boldsymbol{\lambda}^*)$. Since Lemma A.2.2 implies that the UBP Condition is guaranteed to hold for all impression types $i \in \mathcal{I}$, we can apply Lemma A.2.4. In particular, by (A.6) and (A.7), there exist $\check{\mathbf{x}}_i \in \mathcal{X}^*_i(\boldsymbol{\lambda}^*; \mathbf{b}^*_i(\boldsymbol{\lambda}^*))$ for each $i \in \mathcal{I}$ and $\mathbf{y} \in \partial p(\boldsymbol{\lambda}^*)$ such that:

$$\mathbf{0} \;=\; \sum_{i \in \mathcal{I}} -\mathbf{v}_i(\check{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda}^*)) \;+\; \mathbf{y}\ .$$

Let $\check{\mathbf{x}} \in \mathbb{R}^{|\mathcal{E}|}$ and $\mathbf{b}^*(\boldsymbol{\lambda}^*) \in \mathbb{R}^{|\mathcal{E}|}$ denote enlarged vectors of all of the $\check{\mathbf{x}}_i$ and $\mathbf{b}^*_i(\boldsymbol{\lambda}^*)$ subvectors, respectively. Then by definition we have that $\mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) = \sum_{i \in \mathcal{I}} \mathbf{v}_i(\check{\mathbf{x}}_i, \mathbf{b}^*_i(\boldsymbol{\lambda}^*))$, and the above equality implies that $\mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) \in \partial p(\boldsymbol{\lambda}^*)$. The subgradient inequality then states that:

$$p(\boldsymbol{\lambda}) \;\geq\; p(\boldsymbol{\lambda}^*) + \mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*))^\top (\boldsymbol{\lambda} - \boldsymbol{\lambda}^*) \;\text{ for all } \boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}\ .$$

Rearranging the above inequality and using the conjugate representation of $u(\cdot)$ given by the Fenchel-Moreau Theorem yields:

$$u(\mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*))) \;=\; \inf_{\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}} \left\{-\boldsymbol{\lambda}^\top \mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) + p(\boldsymbol{\lambda})\right\} \;=\; -\,(\boldsymbol{\lambda}^*)^\top \mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) + p(\boldsymbol{\lambda}^*)\ .$$

Adding $\pi(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*))$ to both sides of the above and using $(\check{\mathbf{x}}_i, \mathbf{b}_i^*(\boldsymbol{\lambda})) \in \mathcal{S}_i^*(\boldsymbol{\lambda})$ from Lemma A.2.4 yields:

$$\begin{aligned} F(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) &= \pi(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) + u(\mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*))) \\ &= \pi(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) - (\boldsymbol{\lambda}^*)^\top \mathbf{v}(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) + p(\boldsymbol{\lambda}^*) \\ &= q(\boldsymbol{\lambda}^*) + p(\boldsymbol{\lambda}^*) \\ &= Q(\boldsymbol{\lambda}^*) \ . \end{aligned}$$

Therefore, we have shown that $F(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) = Q(\boldsymbol{\lambda}^*) = Q^*$, which, by weak duality, implies that $(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*))$ is an optimal solution of the primal problem (2.1) and that we have $F^* = Q^*$, i.e., there is zero duality gap.

Notice that that above reasoning applies to any given dual optimal solution $\boldsymbol{\lambda}^*$. In particular, suppose that $\boldsymbol{\lambda}^*$ has been computed by solving the dual problem (2.6) to exact optimality during Phase 1 of Algorithm 3. Let $\mathbf{x}^*$ denote the allocation vector computed during Phase 2 of Algorithm 3. Then, presuming that we solve the restricted primal problem (2.8) given $\mathbf{b}^*(\boldsymbol{\lambda}^*)$ to exact optimality, we have by definition that $F(\mathbf{x}^*, \mathbf{b}^*(\boldsymbol{\lambda}^*)) \geq F(\check{\mathbf{x}}, \mathbf{b}^*(\boldsymbol{\lambda}^*)) = F^*$, which verifies that Algorithm 3 computes an optimal solution $(\mathbf{x}^*, \mathbf{b}^*(\boldsymbol{\lambda}^*))$ of the primal problem (2.1). $\square$

Proofs for Section 2.3

Proof of Proposition 2.3.1:

Proof. Without loss of generality, we presume throughout this proof that $\bar{b}_i > 0$. By assumption C_i is a continuous random variable with CDF $\rho_i(\cdot)$, hence $\rho_i'(b) = f_{C_i}(b)$ for all $b \in (0, \bar{b}_i)$. Thus, the assumption that $f_{C_i}(b) > 0$ for all $b \in [0, \bar{b}_i]$ implies that item (1.) of the IMC Condition holds. To demonstrate that item (2.) of the IMC Condition holds, notice that for any $b \in (0, \bar{b}_i)$ we have that $\beta_i(b) = \mathbb{E}[C_i \mid C_i < b] = \frac{1}{\rho_i(b)} \int_0^b x f_{C_i}(x)dx$, hence $c_i(b) = \int_0^b x f_{C_i}(x)dx$ and

$$g_i(b) = \frac{c_i'(b)}{\rho_i'(b)} = \frac{b f_{C_i}(b)}{f_{C_i}(b)} = b \ .$$

Hence, $g_i(\cdot)$ is strictly increasing on $(0, \bar{b}_i)$. $\square$

Proof of Proposition 2.3.2:

Proof. Without loss of generality, we presume throughout this proof that $\bar{b}_i > 0$. Let us first consider the case where $\rho_i(\cdot)$ is differentiable, strictly increasing, and concave on $(0, \bar{b}_i)$. Since $\rho_i(\cdot)$ is differentiable and strictly increasing, we immediately have that item (1.) of the IMC Condition holds. To demonstrate that item (2.) of the IMC Condition holds, recall that $\beta_i(b) = b$ for all $b \in [0, \bar{b}_i]$ and hence, for any $b \in (0, \bar{b}_i)$, the product rule yields

$$g_i(b) = \frac{c_i'(b)}{\rho_i'(b)} = \frac{(\rho_i(b)b)'}{\rho_i'(b)} = \frac{\rho_i'(b)b + \rho_i(b)}{\rho_i'(b)} = b + \frac{\rho_i(b)}{\rho_i'(b)} \ . \tag{A.9}$$

Now since $\rho_i(\cdot)$ is concave and strictly increasing on $(0, \bar{b}_i)$ we have that $\rho_i'(\cdot)$ is non-increasing on $(0, \bar{b}_i)$, therefore $g_i(\cdot)$ is strictly increasing on $(0, \bar{b}_i)$.

Let us now consider the case where $\rho_i(\cdot)$ is the CDF of the maximum of $n \geq 1$ i.i.d. uniform random variables on the interval $[0, \bar{c}_i]$ with $\bar{c}_i \geq \bar{b}_i > 0$. A quick calculation yields that $\rho_i(b) = \left(\frac{b}{\bar{c}_i}\right)^n$ for $b \in [0, \bar{b}_i]$. Hence, we have that $\rho_i'(b) = \frac{nb^{n-1}}{\bar{c}_i^n} > 0$ for $b \in (0, \bar{b}_i)$, which proves that item (1.) of the IMC Condition holds. To demonstrate that item (2.) of the IMC Condition holds note that, by equation (A.9), $g_i(b) = b(1 + 1/n)$ which is strictly increasing on $(0, \bar{b}_i)$. □

Proofs for Section 2.3

Proof of Proposition 2.3.3:

Proof. Recall that $\mathbf{b}_i^*(\boldsymbol{\lambda}) \in \mathcal{B}_i := [0, \bar{b}_i]^{|\mathcal{K}_i|} \subseteq \mathbb{R}^{|\mathcal{K}_i|}$ is defined component-wise for each campaign $k \in \mathcal{K}_i$ by $b_{ik}^*(\boldsymbol{\lambda}) := b_i^*(r_{ik}(1 - \lambda_k)) = \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r_{ik}(1 - \lambda_k))$, which is the unique maximizer of the profit expression $h_i(b; r_{ik}(1-\lambda_k)) = [r_{ik}(1-\lambda_k) - \beta_i(b)]\rho_i(b)$ on $[0, \bar{b}_i]$ under the assumed UBP condition. Thus, it suffices to demonstrate that $b_i^*(\cdot)$ is continuous in its argument $r \in \mathbb{R}$, from which continuity of $\mathbf{b}_i^*(\cdot)$ on $\mathbb{R}^{|\mathcal{K}|}$ directly follows.

Let $\bar{r} \in \mathbb{R}$ be given and let $\{r^t\}_{t=0}^{\infty}$ be a sequence satisfying $r^t \to \bar{r}$ as $j \to \infty$. We wish to demonstrate that $b_i^*(r^t) \to b_i^*(\bar{r})$ as $j \to \infty$. Since $b_i^*(r^t) \in [0, \bar{b}_i]$, which is a compact set, a standard result in real analysis states that $b_i^*(r^t) \to b_i^*(\bar{r})$ if and only if all convergent subsequences of the sequence $\{b_i^*(r^t)\}_{j=0}^{\infty}$ converge to the (identical) value of $b_i^*(\bar{r})$. Therefore, without loss of generality, we assume that $b_i^*(r^t) \to \tilde{b}$ for some $\tilde{b} \in [0, \bar{b}_i]$, and we wish to demonstrate that $\tilde{b} = b_i^*(\bar{r})$. Let us first demonstrate that $h_i(\tilde{b}; \bar{r}) = h_i(b_i^*(\bar{r}); \bar{r})$. By the definition of $b_i^*(\bar{r})$, we clearly have that $h_i(\tilde{b}; \bar{r}) \leq h_i(b_i^*(\bar{r}); \bar{r})$. Now, suppose by way of contradiction that $h_i(\tilde{b}; \bar{r}) < h_i(b_i^*(\bar{r}); \bar{r})$. Since $h_i(b; r) := [r - \beta_i(b)]\rho_i(b)$ is jointly continuous in $r \in \mathbb{R}$ and $b \in [0, \bar{b}_i]$ and since $r^t \to \bar{r}$ and $b_i^*(r^t) \to \tilde{b}$, we have that there exists $\delta > 0$ such that:

$$h_i(b_i^*(r^t); r^t) \;\leq\; h_i(b_i^*(\bar{r}); \bar{r}) - \delta \;\text{ for all } t \text{ sufficiently large.}$$

Combining the above with the definition of $b_i^*(r^t)$ yields:

$$h_i(b_i^*(\bar{r}); r^t) \;\leq\; h_i(b_i^*(r^t), r^t) \;\leq\; h_i(b_i^*(\bar{r}); \bar{r}) - \delta \;\text{ for all } t \text{ sufficiently large.}$$

Now, taking the limit of the left side of the above as $t \to \infty$ yields $h_i(b_i^*(\bar{r}); \bar{r}) \leq h_i(b_i^*(\bar{r}); \bar{r}) - \delta$, which is a contradiction. Therefore, we conclude that $h_i(\tilde{b}; \bar{r}) = h_i(b_i^*(\bar{r}); \bar{r})$. Finally, by the UBP condition, it must be the case that $\tilde{b} = b_i^*(\bar{r})$. □

Proof of Proposition 2.3.4:

Proof. Let $\bar{\mathbf{b}} \in \mathcal{B}$ be given satisfying $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V}(\bar{\mathbf{b}}) \neq \emptyset$, and let $\{\mathbf{b}^t\}_{t=0}^{\infty}$ be a sequence, with $\mathbf{b}^t \in \mathcal{B}$ for all $t \geq 0$, such that $\mathbf{b}^t \to \bar{\mathbf{b}}$. We wish to demonstrate that $F^*(\mathbf{b}^t) \to F^*(\bar{\mathbf{b}})$.

The condition $\operatorname{int}(\operatorname{dom}(u(\cdot))) \cap \mathcal{V}(\bar{\mathbf{b}}) \neq \emptyset$ means that there exists $\hat{\mathbf{x}} \in \mathcal{X}$ and $\varepsilon > 0$ such that $\mathbf{v}(\hat{\mathbf{x}}, \bar{\mathbf{b}}) \in \operatorname{int}(\operatorname{dom}(u(\cdot)))$, i.e., for all $\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|}$, it holds that:

$$\|\mathbf{v} - \mathbf{v}(\hat{\mathbf{x}}, \bar{\mathbf{b}})\| < \varepsilon \;\Rightarrow\; u(\mathbf{v}) > -\infty \; . \tag{A.10}$$

By the continuity of $\mathbf{v}(\hat{\mathbf{x}}, \cdot)$ and without loss of generality, we assume that $\|\mathbf{v}(\hat{\mathbf{x}}, \mathbf{b}^t) - \mathbf{v}(\hat{\mathbf{x}}, \bar{\mathbf{b}})\| < \varepsilon$ for all $t \geq 0$. Thus, by (A.10), we have that $\mathbf{v}(\hat{\mathbf{x}}, \mathbf{b}^t) \in \operatorname{dom}(u(\cdot))$, which means that problem (2.8) given $\mathbf{b}^t$ is feasible, i.e., we have $\operatorname{dom}(u(\cdot)) \cap \mathcal{V}(\mathbf{b}^t) \neq \emptyset$. Therefore, by Proposition 2.2.3, $F^*(\mathbf{b}^t)$ is finite and there exists $\mathbf{x}^t \in \mathcal{X}$ such that:

$$F^*(\mathbf{b}^t) = \pi(\mathbf{x}^t, \mathbf{b}^t) + u(\mathbf{v}(\mathbf{x}^t, \mathbf{b}^t)) \;\; \text{for all } t \geq 0 \; . \tag{A.11}$$

Additionally, there exists $\bar{\mathbf{x}} \in \mathcal{X}$ such that:

$$F^*(\bar{\mathbf{b}}) = \pi(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + u(\mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}})) \; . \tag{A.12}$$

Let us now demonstrate that all of the $F^*(\mathbf{b}^t)$ values are bounded on an interval. Define the set $\mathcal{P} := \{(\mathbf{x}, \mathbf{b}) \in \mathcal{S} : u(\mathbf{v}(\mathbf{x}, \mathbf{b})) > -\infty\}$. As previously argued in the proof of Proposition 2.2.1, $\mathcal{P}$ is a compact set and the objective function $\pi(\cdot, \cdot) + u(\mathbf{v}(\cdot, \cdot))$ is continuous on $\mathcal{P}$. Therefore, the Weierstrass extreme value theorem implies that there exists finite values F_* and F^* such that $\pi(\mathbf{x}, \mathbf{b}) + u(\mathbf{v}(\mathbf{x}, \mathbf{b})) \in [F_*, F^*]$ for all $(\mathbf{x}, \mathbf{b}) \in \mathcal{P}$. Finally, note that we clearly have $(\mathbf{x}^t, \mathbf{b}^t) \in \mathcal{P}$, hence by (A.11) we have that $F^*(\mathbf{b}^t) \in [F_*, F^*]$ for all $t \geq 0$.

Now, since we have demonstrated that the $F^*(\mathbf{b}^t)$ values are bounded on a compact interval $[F_*, F^*]$, a standard result from real analysis states that $F^*(\mathbf{b}^t) \to F^*(\bar{\mathbf{b}})$ if and only if all convergent subsequences of the sequence $\{F^*(\mathbf{b}^t)\}$ converge to the (identical) value of $F^*(\bar{\mathbf{b}})$. Thus, let $\{F^*(\mathbf{b}^j)\}$ denote a convergent subsequence of $\{F^*(\mathbf{b}^t)\}$, whereby $F^*(\mathbf{b}^j) = F^*(\mathbf{b}^{t(j)})$ for all $j \geq 0$ and $F^*(\mathbf{b}^j) \to \tilde{F}$ as $j \to \infty$. Consider the subsequence $\{\mathbf{x}^j\}$ from (A.11) corresponding to the $\{F^*(\mathbf{b}^j)\}$ subsequence. Then, since $\mathbf{x}^j \in \mathcal{X}$, which is a compact set, there is another subsequence of $\{\mathbf{x}^j\}$ converging to some $\tilde{\mathbf{x}} \in \mathcal{X}$. By considering convergence along this additional subsequence, as well as by the correspondence in (A.11) and the continuity of $\pi(\cdot, \cdot) + u(\mathbf{v}(\cdot, \cdot))$ on $\mathcal{P}$, it holds that $\tilde{F} = \pi(\tilde{\mathbf{x}}, \bar{\mathbf{b}}) + u(\mathbf{v}(\tilde{\mathbf{x}}, \bar{\mathbf{b}}))$. Thus, we clearly have that $\tilde{F} \leq F^*(\bar{\mathbf{b}})$.

Suppose now that $\tilde{F} < F^*(\bar{\mathbf{b}})$. Then, since $F^*(\mathbf{b}^j) \to \tilde{F}$ and by (A.11) and (A.12), there exists $\delta > 0$ such that:

$$\pi(\mathbf{x}^j, \mathbf{b}^j) + u(\mathbf{v}(\mathbf{x}^j, \mathbf{b}^j)) \;\leq\; \pi(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + u(\mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}})) - \delta \;\; \text{for all } j \text{ sufficiently large.} \tag{A.13}$$

Consider an arbitrary $\gamma \in [0, 1]$ and let $\mathbf{x}(\gamma) := (1 - \gamma)\bar{\mathbf{x}} + \gamma\hat{\mathbf{x}}$. Then, by the linearity of $\mathbf{v}(\cdot, \bar{\mathbf{b}})$ in the first argument given fixed $\bar{\mathbf{b}}$, we have that $\mathbf{v}(\mathbf{x}(\gamma), \bar{\mathbf{b}}) = (1 - \gamma)\mathbf{v}(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + \gamma\mathbf{v}(\hat{\mathbf{x}}, \bar{\mathbf{b}})$. Then, by the convexity of $\operatorname{dom}(u(\cdot))$, note that $\mathbf{v}(\mathbf{x}(\gamma), \bar{\mathbf{b}}) \in \operatorname{int}(\operatorname{dom}(u(\cdot)))$ for all $\gamma > 0$. Additionally, by continuity and (A.13), there is a sufficiently small $\check{\gamma} > 0$ such that:

$$\pi(\mathbf{x}^j, \mathbf{b}^j) + u(\mathbf{v}(\mathbf{x}^j, \mathbf{b}^j)) \;\leq\; \pi(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}}) + u(\mathbf{v}(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}})) - \delta/2 \;\; \text{for all } j \text{ sufficiently large.} \tag{A.14}$$

Now, since $\mathbf{v}(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}}) \in \text{int}(\text{dom}(u(\cdot)))$ and $\mathbf{b}^j \to \bar{\mathbf{b}}$, for sufficiently large j we have that $\mathbf{v}(\mathbf{x}(\check{\gamma}), \mathbf{b}^j) \in \text{int}(\text{dom}(u(\cdot)))$, hence by (A.11) we have that:

$$-\infty \ < \ \pi(\mathbf{x}(\check{\gamma}), \mathbf{b}^j) + u(\mathbf{v}(\mathbf{x}(\check{\gamma}), \mathbf{b}^j)) \ \leq \ \pi(\mathbf{x}^j, \mathbf{b}^j) + u(\mathbf{v}(\mathbf{x}^j, \mathbf{b}^j)) \ \text{ for all } j \text{ sufficiently large.}$$

Combining the above with (A.14) and taking the limit of the left side as $j \to \infty$ yields:

$$\pi(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}}) + u(\mathbf{v}(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}})) \ \leq \ \pi(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}}) + u(\mathbf{v}(\mathbf{x}(\check{\gamma}), \bar{\mathbf{b}})) - \delta/2 \ ,$$

which is a contradiction. Thus, we have that $\tilde{F} = F^*(\bar{\mathbf{b}})$, which completes the proof. □

Proof of Theorem 2.3.2:

Proof. First, note that statements (1.)-(3.) of Theorem 2.3.1 are immediate. Indeed, $\mathcal{V}(\mathbf{b}) \subseteq \mathcal{V}$ for all $\mathbf{b} \in \mathcal{B}$ implies that the the condition $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V}(\mathbf{b}) \neq \emptyset$ for all $\mathbf{b} \in \mathcal{B}$ is stronger than $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$. Then, statements (1.)-(3.) of Theorem 2.3.1 hold immediately as the rest of the assumptions of Theorem 2.3.1 also hold for Theorem 2.3.2.

Statement (1.) of Theorem 2.3.2 follows from Propositions 2.3.3 and 2.3.4. In particular, we have that $F(\mathbf{b}^*(\cdot))$ is a continuous function at $\boldsymbol{\lambda}^*$. Indeed, since all impression types satisfy either the IMC Condition or the more general UBP Condition, Proposition 2.3.3 implies that $\mathbf{b}^*(\cdot)$ is continuous. Thus, $\boldsymbol{\lambda}^t \to \boldsymbol{\lambda}^*$ implies that $\mathbf{b}^*(\boldsymbol{\lambda}^t) \to \mathbf{b}^*(\boldsymbol{\lambda}^*)$. Now, by our stronger assumption, we have that $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V}(\mathbf{b}^*(\boldsymbol{\lambda}^*)) \neq \emptyset$, and thus Proposition 2.3.4 yields that $F^*(\cdot)$ is continuous at $\mathbf{b}^*(\boldsymbol{\lambda}^*)$. Therefore, it holds that $F^*(\mathbf{b}^*(\boldsymbol{\lambda}^t)) \to F^*(\mathbf{b}^*(\boldsymbol{\lambda}^*))$. Finally, by statement (3.) of Theorem 2.3.1, it holds that $F^* = F^*(\mathbf{b}^*(\boldsymbol{\lambda}^*))$, hence $F^*(\mathbf{b}^*(\boldsymbol{\lambda}^t)) \to F^*$. (Note that the formula for the last row, the exponential case, is not easy to state but can be solved numerically.) □

A.3 Examples and Derivations

Convex Conjugate Examples

Here we show how to obtain the convex conjugates $p_k(\cdot)$ of the utility functions $-u_k(\cdot)$ considered in Section 2.2. Before doing so, let us first prove that a utility function being separable on its campaigns, i.e., $u(\mathbf{v}) = \sum_{k \in \mathcal{K}} u_k(v_k)$, implies that $p(\cdot)$ is similarly separable. Define $p_k(\cdot) : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ as the convex conjugate of $-u_k(\cdot)$, then for any $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{K}|}$ it holds that:

$$p(\boldsymbol{\lambda}) = \sup_{\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|}} \left\{ \boldsymbol{\lambda}^\top \mathbf{v} + u(\mathbf{v}) \right\} = \sup_{\mathbf{v} \in \mathbb{R}^{|\mathcal{K}|}} \left\{ \boldsymbol{\lambda}^\top \mathbf{v} + \sum_{k \in \mathcal{K}} u_k(v_k) \right\} = \sum_{k \in \mathcal{K}} \sup_{v_k \in \mathbb{R}} \{ \lambda_k v_k + u_k(v_k) \}$$
$$= \sum_{k \in \mathcal{K}} p_k(\lambda_k) \ .$$

All the results use $k \in \mathcal{K}$ as an arbitrarily chosen campaign.

- Derivation for Example 2.2.1. For $u_k(v_k)$ equal to 0 if $v_k \leq m_k$ and $-\infty$ o.w. we have

  $$p_k(\lambda_k) = \sup_{v_k \in \mathbb{R}} \quad \{\lambda_k v_k + u_k(v_k)\} = \sup_{v_k \leq m_k} \quad \lambda_k v_k$$

  Then, $p_k(\cdot)$ takes the value $\lambda_k m_k$ if $\lambda_k \geq 0$ and $+\infty$ otherwise.

- Derivation for Example 2.2.2. For $u_k(v_k) = -\frac{\tau_k}{2}(m_k - v_k)^2$ we have

  $$p_k(\lambda_k) = \sup_{v_k \in \mathbb{R}} \quad \left\{\lambda_k v_k - \frac{\tau_k}{2}(m_k - v_k)^2\right\}$$

  Then, for a fixed λ_k we have that $p_k(\lambda_k)$ is the supremum of a quadratic function with negative second derivative, then the first order optimality conditions (derivative equal to zero) finds an optimal solution which is obtained at $v_k^* = \frac{\lambda_k}{\tau_k} + m_k$. Plugging v_k^* in $p_k(\lambda_k)$ we obtain $p_k(\lambda_k) = m_k \lambda_k + \frac{1}{2\tau_k}\lambda_k^2$.

- Derivation for Example 2.2.3. For $u_k(v_k) = -\frac{\tau_k}{2}(m_k - v_k)^2$ if $v_k \leq m_k$ and $-\infty$ o.w. we have

  $$p_k(\lambda_k) = \sup_{v_k \leq m_k} \quad \left\{\lambda_k v_k - \frac{\tau_k}{2}(m_k - v_k)^2\right\}$$

  If $\lambda \geq 0$ an optimal solution for $p_k(\lambda_k)$ is $v_k^* = m_k$ as the term $-\frac{\tau_k}{2}(m_k - v_k)^2$ is non-positive, with which we obtain $p_k(\lambda_k) = \lambda_k m_k$. If $\lambda_k < 0$, then the optimal solution of the previous example, $v_k^* = \frac{\lambda_k}{\tau_k} + m_k$, is feasible and obtains the same objective value as in the previous example. An unconstrained problem can only have a higher optimal solution than a constrained one. Then, we have shown $p_k(\lambda_k) = m_k \lambda_k$ if $\lambda_k \geq 0$ and $p_k(\lambda_k) = m_k \lambda_k + \frac{1}{2\tau_k}\lambda_k^2$ o.w.

- Derivation for Example 2.2.4. For $\alpha_k \in [0, 1]$ let $u_k(v_k)$ equal to 0 if $v_k \in [\alpha_k m_k, m_k]$ and $-\infty$ o.w and we have

  $$p_k(\lambda_k) = \sup_{v_k \in [\alpha_k m_k, m_k]} \quad \{\lambda_k m_k\}$$

  Then, an optimal v_k^* is m_k when $\lambda_k \geq 0$ and $\alpha_k m_k$ when $\lambda_k < 0$, thus obtaining $p_k(\lambda_k) = \lambda_k m_k$ if $\lambda_k \geq 0$ and $p_k(\lambda_k) = \alpha_k \lambda_k m_k$ o.w.

Optimal Bidding Forms

Let $i \in \mathcal{I}$ be an arbitrary impression type and let $b_i^*(r) \in \arg\max_{b \in [0, \bar{b}_i]} \; h_i(b; r)$ (defined in Assumption 2.2.3). In this section, we calculate $b_i^*(r)$ for first and second-price auctions for some forms of $\rho_i(\cdot)$, and mention when $b_i^*(r)$ is the unique maximum of $\arg\max_{b \in [0, \bar{b}_i]} \; h_i(r; b)$. Our results are summarized in Table A.2. Recall that $\rho_i(\cdot)$ is defined on $[0, \bar{b}_i]$ in Section 2.1. Herein we extend the definition of $\rho_i(\cdot)$ to an interval $[0, c_i]$ where $c_i \geq \bar{b}_i$ and we may possibly have $c_i = +\infty$. Note again that $\rho_i(\bar{b}_i) < 1$ allows for the possibility that the DSP may lose the auction even if they bid the maximum amount $\bar{b}_i$.

We only need to directly derive the optimal bidding form for the example in the first row of Table A.2. The proofs of the uniqueness and the optimal bidding forms of the last five rows can be considered simultaneously as they all follow from the Increasing Marginal Cost (IMC) condition (Definition 2.3.1). Finally, note that the examples of $\rho_i(\cdot)$ for the first-price case correspond, in order, to the following: (i) the c.d.f. of $n \geq 1$ i.i.d. Uniform$(0, c_i)$ random variables with $c_i \geq \bar{b}_i$, (ii) a form used in [129], (iii) a square root function, (iv) the c.d.f. of an Exponential(λ) random variable. For the first of these four examples c_i is finite, and for the last three examples we have that $c_i = +\infty$.

First Row of Table A.2: Let $r \in \mathbb{R}$ be given. We will show that for arbitrary second-price auctions it holds that $b_i^*(r) = \text{clip}(r; [0, \bar{b}_i])$. Let C_i be a non-negative random variable denoting the maximum competing bid for an impression of type i. Also, denote $\mathbb{1}_{\cdot \in A}$ as the indicator function that takes the value of 1 if the input belongs to a set A or satisfies a given condition, and 0 otherwise. Given that this is a second-price auction, we have that $\rho_i(b) = \mathbb{P}(C_i < b)$ and $\beta_i(b) = \mathbb{E}[C_i | C_i < b]$. Note that the derivation to be done here does not necessarily assume that $\rho_i(\cdot)$ is a continuous function as in Assumption 2.2.2. Notice that we have $\beta_i(b) = \mathbb{E}[C_i | C_i < b] = \frac{\mathbb{E}[C_i \mathbb{1}_{C_i < b}]}{\mathbb{P}(C_i < b)} = \frac{\mathbb{E}[C_i \mathbb{1}_{C_i < b}]}{\rho_i(b)}$ whenever $\rho_i(b) > 0$. Let $\check{b} := \sup\{b : \rho_i(b) = 0\}$. Then, $\check{b}$ is finite and $\rho_i(b) = 0$ for all $b \leq \check{b}$ and $\rho_i(b) > 0$ for all $b > \check{b}$. Then, we have that $h_i(r; b) := (r - \beta_i(b))\, \rho_i(b)$ satisfies

$$h_i(r; b) = \begin{cases} 0 & \text{if } b \leq \check{b} \\ r\mathbb{P}(C_i < b) - \mathbb{E}[C_i \mathbb{1}_{C_i < b}] & \text{if } b \geq \check{b} \end{cases} \tag{A.15}$$

Clearly then, for any $r \leq \check{b}$ we have $\text{clip}(r; [0, \bar{b}_i]) \in \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r)$.

Suppose now that $r > \check{b}$. We will prove that the function $h_i(r; \cdot)$ is monotone increasing on $[0, r]$ and monotone decreasing on $[r, \infty)$. Subsequently, we have that $\text{clip}(r; [0, \bar{b}_i]) \in \arg\max_{b \in [0, \bar{b}_i]} h_i(b; r)$.

Let $b \geq \check{b}$ and $\delta \geq 0$ such that $b + \delta \leq r$ be given. Then, by (A.15) we have that

$$\begin{aligned} h_i(r; b+\delta) - h_i(r; b) &= r\left(\mathbb{P}(C_i < b+\delta) - \mathbb{P}(C_i < b)\right) - \left(\mathbb{E}[C_i \mathbb{1}_{C_i < b+\delta}] - \mathbb{E}[C_i \mathbb{1}_{C_i < b}]\right) \\ &= r\mathbb{P}(b \leq C_i < b+\delta) - \mathbb{E}[C_i \mathbb{1}_{b \leq C_i < b+\delta}] \\ &\geq r\mathbb{P}(b \leq C_i < b+\delta) - (b+\delta)\mathbb{P}(b \leq C_i < b+\delta) \geq 0, \end{aligned}$$

hence $h_i(r; \cdot)$ is monotone increasing on $[0, r]$.

Now let $b \geq r$ and $\delta \geq 0$ be given. Then, again by (A.15) we have that

$$\begin{aligned} h_i(r; b) - h_i(r; b+\delta) &= \mathbb{E}[C_i \mathbb{1}_{b \leq C_i < b+\delta}] - r\mathbb{P}(b \leq C_i < b+\delta) \\ &\geq b\mathbb{P}(b \leq C_i < b+\delta) - r\mathbb{P}(b \leq C_i < b+\delta) \geq 0, \end{aligned}$$

hence $h_i(r; \cdot)$ is monotone decreasing on $[r, \infty)$.

Last Five Rows of Table A.2: By Proposition 2.3.1 the second row satisfies the IMC Condition (Definition 2.3.1), and by Proposition 2.3.2 the last four rows satisfy IMC too. Lemma A.2.2 shows that the IMC condition implies the UBP condition (Definition 2.3.2) which proves the uniqueness of the last five rows of Table A.2. We are only left to verify that the optimal bidding forms $b_i^*(r)$ for the last four rows of Table A.2 are correct (the second row of TableA.2 is a particular case of the first row). This can be easily done by following the proof of Lemma A.2.2. In the proof we use the function $g_i(b) = \frac{(\rho_i(b)\beta_i(b))'}{\rho_i'(b)}$ defined for $b \in [0, \bar{b}_i]$. In particular, in each case we need to calculate $\hat{b}$ such that $g_i(\hat{b}) = r$ and then take $b_i^*(r) = \text{clip}(\hat{b}, [0, \bar{b}_i])$. Using the particular formulas for $\rho_i(\cdot)$ and $\beta_i(b) = b$, the formulas for $b_i^*(r)$ in Table A.2 then follow easily.

Examples Concerning Theorem 2.3.1

We present two examples that demonstrate that the conclusions of Theorem 2.3.1 may not hold if one or more of the assumptions are violated. We first examine a case where Slater's Condition does not hold and then a case where the UBP Condition does not hold.

Slater's Condition Does Not Hold.

This example shows a case of dual non-attainment. In this example, there is zero duality gap between problems (2.1) and (2.6) – conclusion (2.) of Theorem 2.3.1 holds – but the dual optimal value is only approached asymptotically. Therefore there is no dual optimal solution and conclusion (1.) of Theorem 2.3.1 does not hold. Take $|\mathcal{I}| = |\mathcal{K}| = 1$ allowing us to drop the sub-indices i and k from all parameters and variables. Assume $s = 1$, $r = 0$, $\bar{b} = 1$, and $\rho(b) = b$ and $\beta(b) = 0.5b$ for all $b \in [0, 1.0]$. These bid landscape functions can be understood as a second-price mechanism where the highest competing bid is a Uniform$(0, 1)$ random variable. Here we use the utility function $u(v) = \sqrt{v}$ for $v \geq 0$ and $u(v) = -\infty$ for $v < 0$. For this utility function it can be easily verified that $p(\lambda) = +\infty$ when $\lambda \geq 0$ and $p(\lambda) = -0.25\lambda^{-1}$ when $\lambda < 0$. Problem (2.1) corresponds to

$$\begin{aligned} \max_{b\in[0,1], x\in[0,1]} \quad & (0 - 0.5b)\, bx + \sqrt{0 \cdot bx} \\ \text{s.t.} \quad & 0 \leq 0 \cdot bx \end{aligned} \tag{A.16}$$

(For this example, the campaign spending is the expression $0 \cdot bx$ which is not meaningful from a practical perspective, but nevertheless satisfies our assumptions.) Notice that the Slater condition $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V} \neq \emptyset$ does not hold as $\text{int}(\text{dom}(u(\cdot))) = \mathbb{R}_+/\{0\}$ and $\mathcal{V} = \{0\}$. All other conditions of Theorem 2.3.1 holds including the IMC Condition (Definition 2.3.1) by Proposition 2.3.1. The optimal value of the problem shown above is 0 with optimal solution $(x^*, b^*) = (0, 0)$. The dual function of problem (A.16) is

$$Q(\lambda) = \max_{b\in[0,1], x\in[0,1]} \quad -0.5b^2x - 0.25\lambda^{-1}$$

for $\lambda < 0$, and $Q(\lambda) = +\infty$ whenever $\lambda > 0$. Assuming $\lambda < 0$, notice that $(0,0)$ is an optimal solution of the inner problem for $Q(\lambda)$ yielding $Q(\lambda) = -0.25\lambda^{-1}$. Notice that $Q(\lambda) > 0$ for any $\lambda < 0$ and that $Q(\lambda) \to 0$ as $\lambda \to -\infty$. Then, the primal and dual problems match only in the limit as $\lambda \to -\infty$.

UBP Condition Does Not Hold.

This example shows a case of non-zero duality gap when the UBP Condition (Definition 2.3.2) does not hold. Take $|\mathcal{I}| = |\mathcal{K}| = 1$ allowing us to drop the sub-indexes i and k from all parameters and variables. Assume that $r = 1$, $s = 10$, $\bar{b} = 1$, $m = 10$, that the budget constraint and hard minimum spending utility function (Example 2.2.4) is used with lower and upper bounds equal to 5.5 and 10.0 respectively, and that $\rho(\cdot)$ is defined as

$$\rho(b) = \begin{cases} 5b & \text{if } b \in [0, 0.1] \\ 0.5 & \text{if } b \in [0.1, 0.9] \\ -4 + 5b & \text{if } b \in [0.9, 1.0] \end{cases}$$

Assuming first-price auctions ($\beta(b) = b$), problem (2.1) for this case corresponds to:

$$\begin{aligned} \max_{b \in [0,1], x \in [0,1]} \quad & 10(1-b)\rho(b)x \\ \text{s.t.} \quad & 5.5 \le 10\rho(b)x \le 10 \end{aligned} \tag{A.17}$$

Notice that problem (A.17) satisfies Slater's Condition from Theorem 2.3.1 since $\text{int}(\text{dom}(u(\cdot))) \cap \mathcal{V} = (5.5, 10)$, and Assumptions 2.2.1 and 2.2.2 also hold. Below we show that the optimal value of problem (A.17) is smaller than the optimal value of its dual problem (0.495 vs 4.05 respectively). This gap does not contradict Theorem 2.3.1 as the UBP condition (Definition 2.3.2) does not hold for this example, a fact that is also proven below. Recall that the IMC (Definition 2.3.1) condition is a particular case of the UBP Condition by Lemma A.2.2. Nevertheless, it can be immediately observed that IMC does not hold as $\rho(\cdot)$ is not a differentiable function.

Primal Problem. We argue that $x^* = 1$ and $b^* = 0.91$ is an optimal solution of problem (A.17) with an optimal objective value of 0.495. First note that we can fix $x = 1$ as $(1-b)\rho(b) \ge 0$ for all $b \in [0,1]$. Subsequently, only the segment $b \in [0.91, 1.0]$ is feasible as $5.5 > 10\rho(b)$ for all $b \in [0, 0.91)$. For any $b \in (0.91, 1.0)$ the derivative of the objective function is negative as $((1-b)\rho(b))' = (1-b)\rho'(b) - \rho(b) < (0.09)5 - 0.55 = -0.1$, therefore proving that $b^* = 0.91$ is optimal.

Dual Problem. Let us first formulate the dual of problem (A.17). Recalling the notation $h(r; b) = (r - b)\rho(b)$ for any $r \in \mathbb{R}$, $[a]_+ = \max\{a, 0\}$ the dual function is

$$Q(\lambda) = \max_{b \in [0,1], x \in [0,1]} 10 \cdot h\left((1-\lambda), b\right) x + 10 \cdot [\lambda]_+ - 5.5[-\lambda]_+,$$

and as usual, the dual problem is $\min_{\lambda \in \mathbb{R}} Q(\lambda)$. The goal of this part is to show that $\min_{\lambda \in \mathbb{R}} Q(\lambda) = Q(-0.9) = 4.05$, and that the UBP condition does not hold. It is straightforward to calculate:

$$\arg\max_{b \in [0,1]} h(r;b) = \begin{cases} \{0\} & \text{if } r \in (-\infty, 0] \\ \{0.5r\} & \text{if } r \in (0, 0.2] \\ \{0.1\} & \text{if } r \in (0.2, 1.9) \\ \{0.1, 1.0\} & \text{if } r = 1.9 \\ \{1.0\} & \text{if } r \in (1.9, \infty) \end{cases}$$

Is immediate to see that UBP Condition does not hold as, for $r = 1.9$, both $b = 0.1$ and $b = 1.0$ are maximizers of the profit expression. We now prove that $Q(\lambda) > Q(-0.9) = 4.05$ for all $\lambda \neq -0.9$ by first showing that this is the case for any $\lambda > 0$, and then for any $\lambda < 0$, $\lambda \neq -0.9$. For any $\lambda > 0$ we have $Q(\lambda) > 4.5$ as

$$\begin{aligned} Q(\lambda) - 4.5 &= \max_{b \in [0,1], x \in [0,1]} 10h\left((1-\lambda); b\right) x + 10\lambda - 4.5 \\ &\geq 10((1-\lambda) - 0.1)0.5 + 10\lambda - 4.5 = 5\lambda > 0, \end{aligned}$$

where the inequality is obtained by using $b = 0.1$ and $x = 1$ in the maximum. Finally, let us prove that $4.05 = Q(-0.9) < Q(\lambda)$ for all $\lambda < 0$, $\lambda \neq -0.9$ by parts. First, for any $a < 0$ we have $Q(-0.9 + a) = 4.05 - 4.5a$ using that $1.0 \in \arg\max_{b \in [0,1]} h(r, b)$ for any $r \in (1.9, \infty)$. Finally, for any $a \in (0, 0.9]$ we have $Q(-0.9 + a) = 4.05 + 0.5a$ using that $0.1 \in \arg\max_{b \in [0,1]} h(r, b)$ for any $r \in (0.2, 1.9)$.

A.4 Additional Experimental Details

Additional Experimental Results

Herein we present additional experimental results that demonstrate the robustness of the results presented in Section 2.4.

To show the dispersion of our results we use confidence band plots. In this type of plots, the y-axis has a band in which the upper and lower lines show the maximum and minimum obtained for a given quantity of interest on the 100 simulations tried. The line on the center represents the mean value. We preferred this type of plots over box plots as the number of parameters used made box-plots unreadable. Importantly, we tried more γ parameters for Policy 4 and τ parameters for Utility Functions 2.2.2 and 2.2.3 than those shown in the plots on the paper.

For the synthetic experiment, Figure A.1 has confidence bands for the sensitivity to budget experiment. Figure A.2 shows how the γ parameter in the Generalized Greedy heuristic (Policy 4) and the τ parameter in Utility Functions 2.2.2 and 2.2.3 affect the Budget

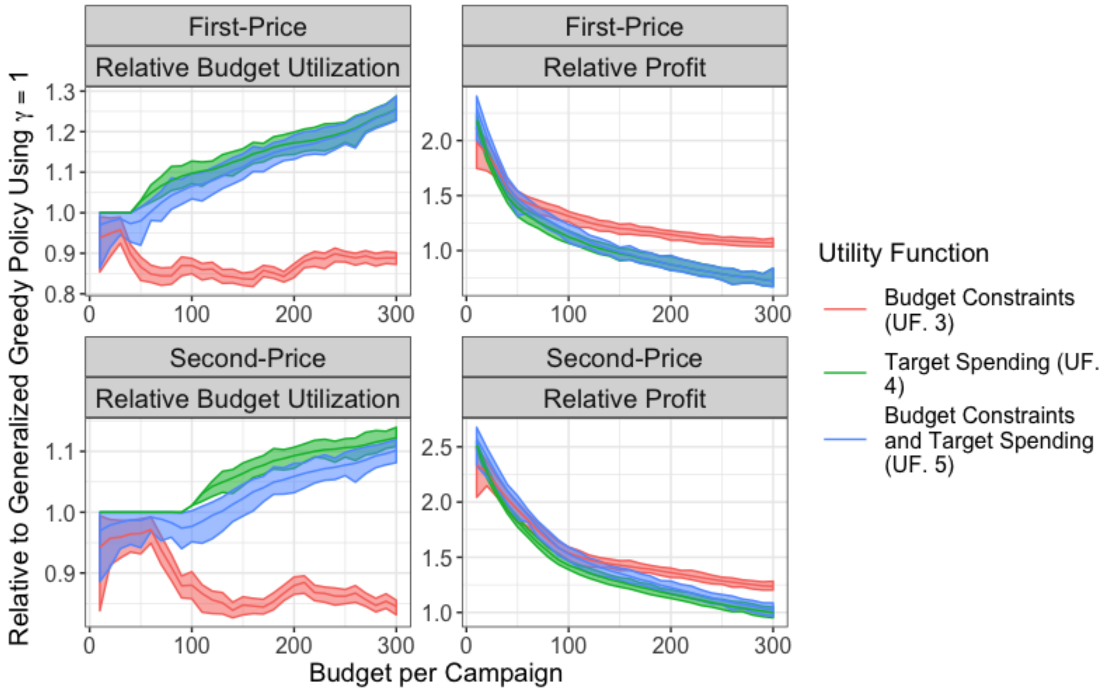


Figure A.1: Sensitivity to budget experiment using synthetic data. The upper and lower curves represent the minimum and maximum relative profit or budget utilization of 100 simulations depending on the graph. The x-axis represent the budgets levels used.

Utilization and Profit of the different methods. Figure A.3 is an equivalent of Figure A.2 for the Criteo experiment. In Figure A.3 we analyze the results depending on the multiplier used for ℓ_k terms. Notice that both Figures A.2 and A.3 show the non-relative profit and budget utilization obtained by the different methods, while Figure A.1 shows results relative to the the Generalized Greedy Policy using $\gamma = 1$.

Simulator Scheme

Here we describe in further detail how we simulate the performance of an allocation and bidding vector $(\mathbf{x}, \mathbf{b})$ on a "real" DSP operation. The simulation uses the following parameters. 1. imp^{test}: Vector containing in sequential order the type of each impression to be received in testing time. 2. mp^{test}: Vector containing in sequential order the highest competing bid associated to each impression (this quantity is also called market price and therefore the 'mp' in its name). 3. θ_{ik}^{test} for $(i,k) \in \mathcal{E}$: Probability of conversion when an ad of campaign 'k' is shown to an impression of type 'i' in testing time. For the synthetic experiment the latter values coincide with those used to run Algorithm 3, *i.e.* $\theta_{ik}^{test} = \theta_{ik}$ for all $(i,k) \in \mathcal{E}$, while for the real-based experiment these values can differ significantly. The latter occurs as we use

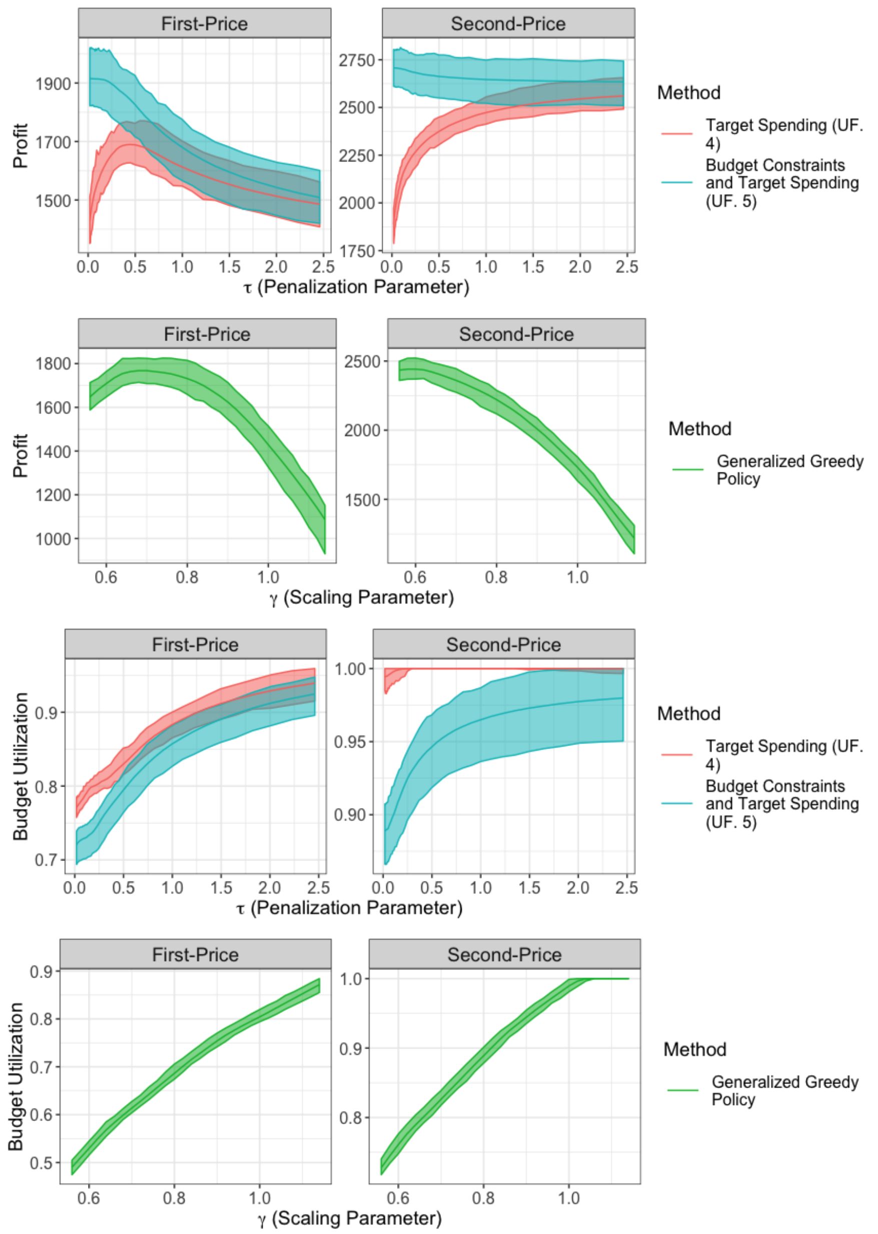


Figure A.2: Pareto curve experiment using synthetic data. The upper and lower curves represent the minimum and maximum profit or budget utilization of 100 simulations depending on the graph. The x-axis represents the parameters used for each methodology.

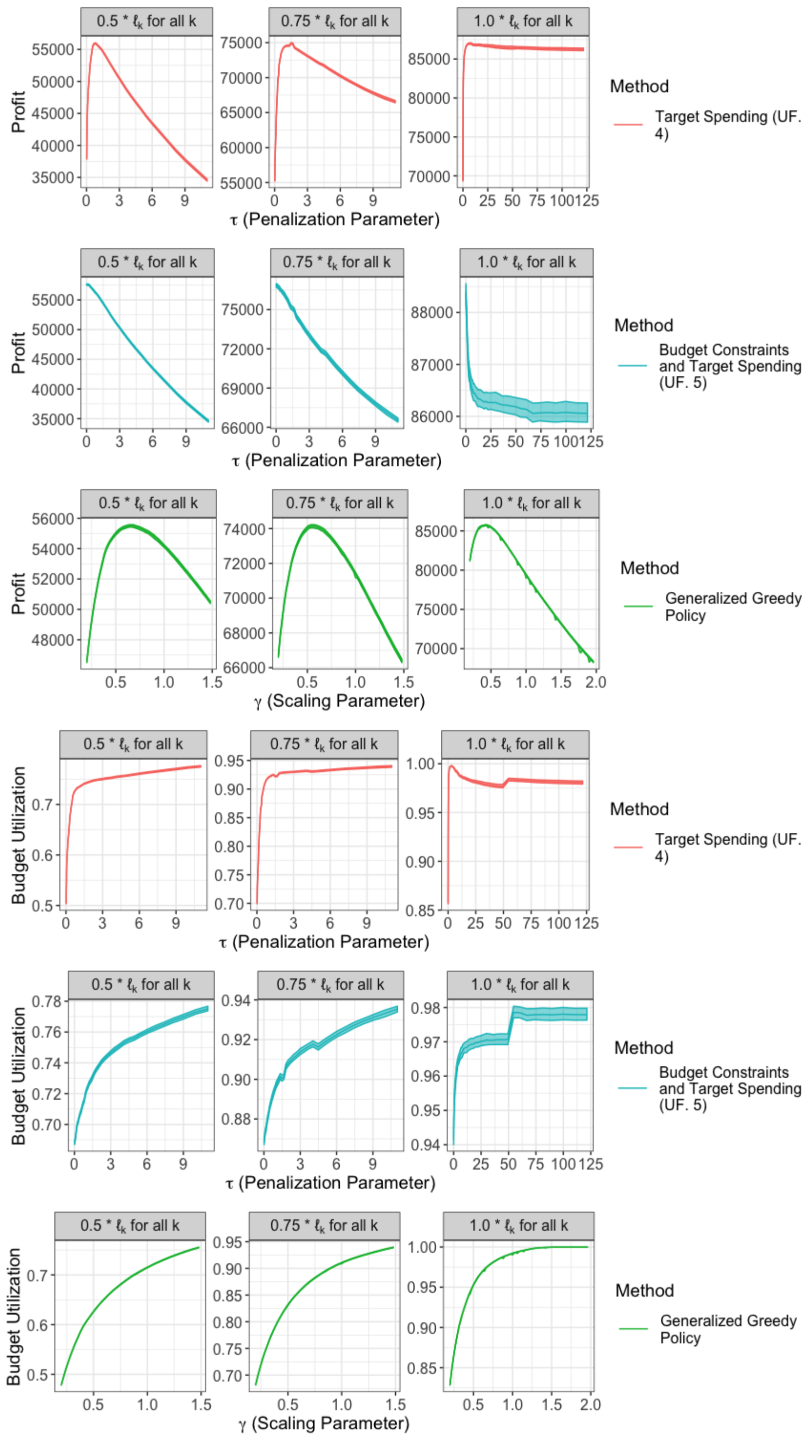


Figure A.3: Pareto curve experiment using the Criteo data. The upper and lower curves represent the minimum and maximum profit or budget utilization, depending on the graph, of 100 simulations using a given method and parameters as shown in the x-axis.

empirical click-through rates in the train and test sets to create the parameters θ_{ik} and θ_{ik}^{test} for all $(i, k) \in \mathcal{E}$ respectively. Procedure 11 shows the mechanism used to simulate how a pair $(\mathbf{x}, \mathbf{b})$ would perform in a "real" DSP operation for both first and second-price auctions.

Procedure 11 Implementation of Policy 1 given $(\mathbf{x},\mathbf{b})$

Input: Allocation and bidding variables $(\mathbf{x}, \mathbf{b})$ and vectors imp^{test}, mp^{test}, and parameter θ_{ik}^{test} for all $(i, k) \in \mathcal{E}$.

1. For each pair of impression type $\tilde{i}$ and highest competing bid $\tilde{\text{mp}}$ from vectors imp^{test} and mp^{test} do (the pairs of impression types and highest competing bids are read in strict order as they appear in their vectors):
2. Sample $u \sim Uniform(0, 1)$ and remember that $\mathcal{K}_{\tilde{i}} = \{k \in \mathcal{K} : (\tilde{i}, k) \in \mathcal{E}\}$. If $u \leq 1 - \sum_{k \in \mathcal{K}_{\tilde{i}}} x_{\tilde{i}k}$, or if the remaining budget for all campaigns $k \in \mathcal{K}_{\tilde{i}}$ is smaller than ℓ_k, the DSP does not bid and return to Step 1. Otherwise, the DSP bids on behalf of a campaign $\tilde{k} \in \mathcal{K}_{\tilde{i}}$ with probability proportional to its $x_{\tilde{i}\tilde{k}}$ value. Only campaigns with a remaining budget higher or equal to $\ell_{\tilde{k}}$ can be selected.
3. If $b_{\tilde{i}\tilde{k}} \geq \tilde{\text{mp}}$, then the DSP wins the auction and pays to the ad-exchange, *i.e.* gets its profit discounted, the amount $b_{\tilde{i}\tilde{k}}$ ($\tilde{\text{mp}}$) if first (second) price auctions are used. If $b_{\tilde{i}\tilde{k}} < \tilde{\text{mp}}$ go to Step 1 and the DSP incurs in no cost.
4. Sample $u \sim Uniform(0, 1)$. If $u > \theta_{\tilde{i}\tilde{k}}^{test}$ return to Step 1. If $u \leq \theta_{\tilde{i}\tilde{k}}^{test}$ then we assume a conversion of interest occur for campaign $\tilde{k}$. In this case add $\ell_{\tilde{k}}$ as profit for the DSP, and discount $\ell_{\tilde{k}}$ from campaign's $\tilde{k}$ budget. Return to Step 1.

Additional Details for the Synthetic Experiments

Here we describe how the impression-campaign graph, the bid landscapes, the conversion probabilities, and the revenue terms were generated for the synthetic experiments. To each impression type i and campaign k we associate a quality score number QS_i and QS_k respectively, which are independently sampled $\text{Uniform}(0, 1)$. The number of campaigns interested in impressions of type i is taken as $\max\{\text{Binomial}(|\mathcal{K}|, \text{QS}_\text{i}), 1\}$ where $\text{Binomial}(|\mathcal{K}|, \text{QS}_\text{i})$ represents a binomial random variable with $|\mathcal{K}|$ tries and probability of success QS_i. To select the particular campaigns interested in an impression type we sample uniformly without replacement from the set $\mathcal{K}$ until the total number of interested campaigns is reached.

We assume that each impression type $i \in \mathcal{I}$ has a number adv_i of other DSPs interested on it who bid *i.i.d.* $\text{Uniform}(0, 1)$ for each impression of type i (these other DSPs are adversaries from a DSP point of view and therefore the name adv_i). The latter implies $\rho_i(b) = b^{\text{adv}_\text{i}}$ for all $b \in [0, 1]$ and $i \in \mathcal{I}$. adv_i is sampled from $\max\{\text{Binomial}(4, \text{QS}_\text{i}), 1\}$ for each $i \in \mathcal{I}$. The probability that an impression of type i converts when an ad of campaign k is shown to it is $\theta_{ik} = 0.5 \cdot (QS_i + QS_k)$ for all valid pairs $(i, k) \in \mathcal{E}$. We use $\mathcal{B}_i = [0, 1]$ for all $i \in \mathcal{I}$.

Additional Details for the Criteo Experiment

Here we give further details on how we obtained and created the parameters for the Criteo experiments, and how these experiments were run. As mentioned in Section 2.4, once we clean the data and run the CART methodology, we can associate each of the Criteo's bidding logs to an impression type (further details on how we run CART are in Appendix A.4). Each of Criteo's bidding logs has its market price, *i.e.*, the highest bid between the other DSPs that competed with Criteo for that impression. Given that Criteo bid in second-price auctions, and that all impression logs are from impressions that Criteo acquired, the market price in each log corresponds to how much Criteo paid for that impression.

Parameters

- $\rho_i(\cdot)$ and $\beta_i(\cdot)$. To create $\rho_i(\cdot)$ and $\beta_i(\cdot)$ for an impression type $i \in \mathcal{I}$, we first obtained the market price of all bidding logs containing impressions of type i (we removed the 2.5% lowest and highest of them). For a fixed $b \in \mathcal{B}_i$, we calculate $\rho_i(b)$ as the percentage of the market price values that are below b. To make the calculation computationally efficient, we saved the 300 values which leaves $(x/300) * 100\%$ of the data below them for $x \in \{0, 1, \ldots, 299\}$. Then, a simple bisection method is an efficient way of approximating the $\rho_i(\cdot)$ function. For $b \in \mathcal{B}_i$, we obtain $\beta_i(b)$ by taking the average between the 300 values mentioned before that are smaller than b (in practice, for $\beta_i(\cdot)$ we stored only 100 values and use interpolation).

- $\mathcal{B}_i$. For each $i \in \mathcal{I}$, we defined $\mathcal{B}_i = [0, \bar{b}_i]$ with $\bar{b}_i$ being the maximum of the 300 values used for calculating $\rho_i(\cdot)$ parameter (mentioned above).

- m_k. The train (test) budget for a campaign $k \in \mathcal{K}$ was taken as the sum of all the market prices in the train (test) logs in which Criteo bid on behalf of campaign k.

- ℓ_k. We obtained ℓ_k for each $k \in \mathcal{K}$ in two steps. First, we divided the train budget of campaign k by the number of clicks associated to campaign k in the train logs. Second, we multiplied the previous number by a $Uniform(0.5, 1.5)$ random variable. The random value per campaign remains constant for all simulations.

- s_i. For each impression type $i \in \mathcal{I}$, s_i equals the number of bidding logs containing impressions of type i in the test set.

- θ_{ik}. The train (test) click-through rate for an impression-campaign pair (i, k) was taken as the empirical click-through rate of this pair in the train (test) logs. In other words, for a pair (i, k) we counted the number of times Criteo bid on impressions of type i on behalf of campaign k, and divide this number by the number of clicks in those bidding logs.

Notice that our construction of the $\rho_i(\cdot)$ and $\beta_i(\cdot)$ parameter functions imply that they are not differentiable and therefore we do not satisfy the IMC condition (Definition 2.3.1). In practice, we did notice a small duality gap rates in our experiment once we run our primal-dual scheme (the duality gap increased as we used higher penalization values). A way of satisfying the IMC condition would be to fit a continuous differentiable function for each impression type. This would require us to call at least 9903 times the cumulative distributive function of the distributions to obtain $\rho_i(b_{ik})$ for all $(i, k) \in \mathcal{E}$ at each iteration of the subgradient step. We found that this amount of computation was prohibitively slow for our experimentation purposes.

Comments about the implementation of the Simulation Scheme 11.

- We run our primal-dual scheme using the s_i parameters obtained using the test set. DSPs usually have decently accurate forecast of the number of future impression opportunities, making this assumption acceptable.
- We read the impression logs in the order they appear in the test set, and used the test budgets both to simulate and to train. There is no error in the latter as a DSP knows the target budget that each campaign wants to spend.
- The randomness in each simulation comes from the probabilistic allocation procedure derived from a solution $(\mathbf{x}, \mathbf{b})$, and the randomness on the the click events in Procedure 11. This randomness explains why we use 100 runs for each pair of experiment and parameter configuration.

Creating Impression Types for Criteo

Criteo's data has nine anonymized categorical columns that we used for creating impression types. Our goal was to find a method that systematically could. 1) Partition the nine categorical columns in a way that each impression log could be associated with one and only one impression type. 2) That different impression types had distinctly different click-through rate probabilities. 3) That the different impression types would each be composed of a significant amount of logs. We achieved these three goals by using Classification and Regression Trees (CART). If an impression was click or not was used as the labels for CART. The impurity coefficient used was Gini, and each leaf in a tree represents an impression type. The number of leaves obtained when running CART depends on the complexity parameter used. An adequate amount of leaves is one that is computationally tractable, while also obtaining good validation accuracy and Gini coefficient. To create a validation set, we split the training logs in the first 3/4 of them as a 'small' train set, and the last 1/4 of them as the validation set. Then, to search for an adequate complexity parameter, we only used the 'small' training set to run CART and the validation set to validate. Figure A.4 shows the number of leaves, validation accuracy, and Gini coefficient under different complexity parameters.

In Figure A.4, we highlighted with a dot the complexity parameter used in our experiments. The complexity parameter chosen created a tree with 96 leaves (impression types) which was computationally tractable. Also, it obtained a lower Gini coefficient than other complexity parameters that would lead to trees with almost ten times the number of leaves. As a final

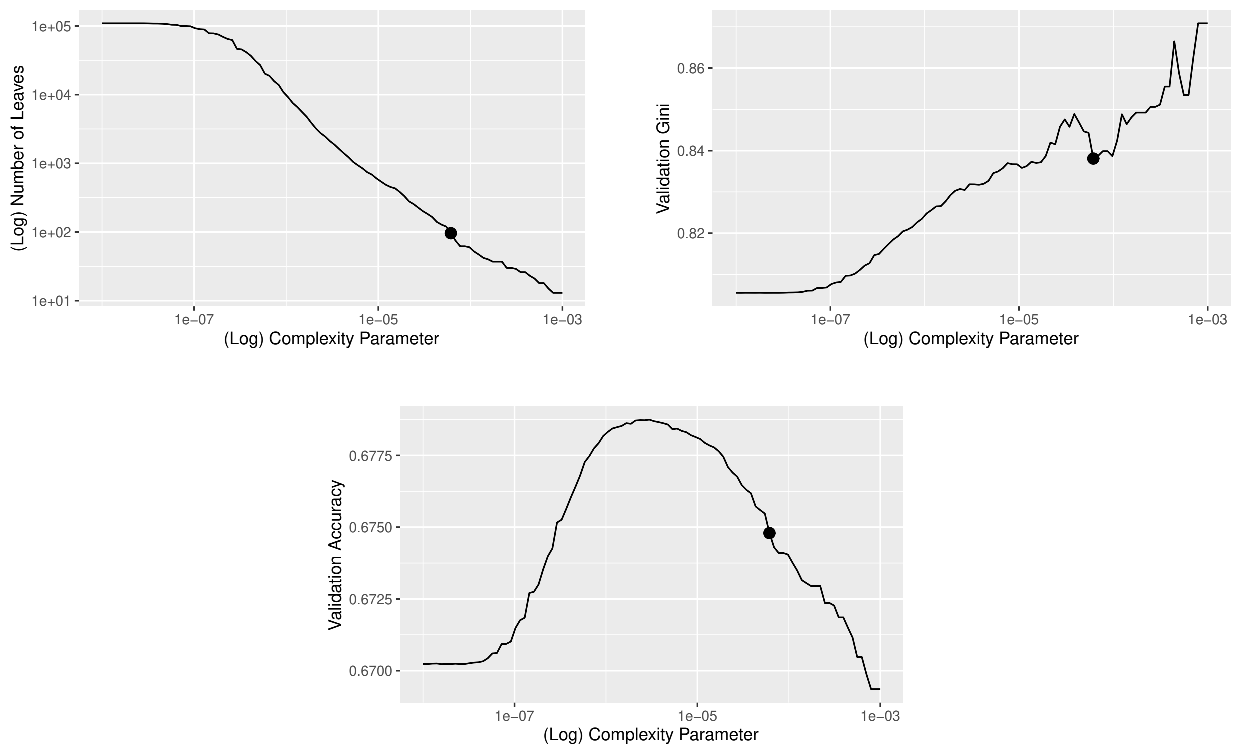


Figure A.4: Number of leaves, validation accuracy and Gini coefficient for different complexity parameter levels. The dot in the curves correspond to the complexity parameter chosen for the experiments.

remark, once we chose the complexity parameter, we re-run CART using the whole dataset, *i.e.*, including the validation set. By doing this, we obtained 84 leaves and not the 96 shown in the figure.

Sets	
$\mathcal{I}$	Impression types.
$\mathcal{K}$	Campaigns.
$\mathcal{I}_k$	Impression types that can be shown an ad of campaign $k \in \mathcal{K}$.
$\mathcal{K}_i$	Campaigns interested in impressions of type $i \in \mathcal{I}$.
$\mathcal{E}$	Graph of (impression type, campaign) pairs.
Known Parameters	
$\bar{b}_i$	Maximum allowed bid for an impression of type i.
m_k	Target spending level (or budget) of campaign k.
ℓ_k	Amount campaign k is charged each time an action of interest occurs.
Estimated Parameters	
s_i	Expected number of arrivals of impressions of type i.
θ_{ik}	Probability that an action of interest occurs when an ad of campaign k is shown to an impression of type i.
r_{ik}	Expected revenue the DSP earns each time an ad of campaign k is shown to an impression of type i, i.e., $r_{ik} := \ell_k \theta_{ik}$.
$\rho_i(b)$	Probability of winning an impression of type i with a bid $b \in [0, \bar{b}_i]$.
$\beta_i(b)$	Expected amount that the DSP pays the ad exchange whenever it wins an auction for an impression of type i with a submitted bid $b \in [0, \bar{b}_i]$.
Variables and Expressions	
x_{ik}	Probability of bidding on behalf of campaign k when an impression of type i arrives.
b_{ik}	Bid to be submitted for an impression of type i on behalf of campaign k.
$\mathbf{v}_i(\mathbf{x}_i, \mathbf{b}_i)$	Vector in $\mathbb{R}^{\lvert\mathcal{K}\rvert}$ of expected spending values associated with impression type i. Its components are $r_{ik}\rho_i(b_{ik})s_i x_{ik}$ for $k \in \mathcal{K}_i$ and 0 for $k \notin \mathcal{K}_i$.
$\pi_i(\mathbf{x}_i, \mathbf{b}_i)$	Total expected profit earned from impression of type i, i.e., $\pi_i(\mathbf{x}_i, \mathbf{b}_i) := \sum_{k \in \mathcal{K}_i} [r_{ik} - \beta_i(b_{ik})]\rho_i(b_{ik}) s_i x_{ik}$.
$h_i(b; r)$	Expected profit that the DSP earns each time it bids an amount b for an impression of type i, given that the expected revenue earned whenever the corresponding ad is displayed is equal to r. That is, $h_i(b; r) := [r - \beta_i(b)]\rho_i(b)$.

Table A.1: List of sets, parameters, and variables.

Auction Type	Condition/form of $\rho_i(\cdot)$	$b_i^*(r)$	Unique
Second-Price	No condition	$\text{clip}(r; [0, \bar{b}_i])$	No
Second-Price	$\rho_i(\cdot)$ strictly increasing in $(0, \bar{b}_i)$	$\text{clip}(r; [0, \bar{b}_i])$	Yes
First-Price	$\rho_i(b) = \left(\frac{b}{c_i}\right)^n$, $n \in \{1, 2, \dots\}$, $\quad c_i \geq \bar{b}_i$	$\text{clip}(\frac{n}{n+1} r; [0, \bar{b}_i])$	Yes
First-Price	$\rho_i(b) = \left(\frac{b}{c+b}\right)$, $c > 0$	$\text{clip}(-c + \sqrt{c^2 + rc}; [0, \bar{b}_i])$	Yes
First-Price	$\rho_i(b) = \sqrt{\frac{b}{c+\bar{b}_i}}$, $c > 0$	$\text{clip}(\frac{r}{3}; [0, \bar{b}_i])$	Yes
First-Price	$\rho_i(b) = 1 - \exp(-\lambda b)$, $\lambda > 0$	See Derivation	Yes

Table A.2: Optimal bidding forms for first and second-price auctions under different bid landscapes.

Appendix B

Joint Online Learning and Decision-making via Dual Mirror Descent

B.1 Additional Theoretical Results and Examples

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

Take the case of $T = 1$, $\mathcal{Z} = \{[0,1]\}$, $\mathcal{W} = \{w_1, w_2\}$ 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 $\arg\max_{\gamma\in[0,1]} \mathrm{OPT}(\mathcal{P},\gamma)$ 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

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

The equality comes directly from the definition of $\mathrm{OPT}(\mathcal{P},\gamma)$. Is direct to see that $z = 1$ maximizes both optimization problems and that $\mathrm{OPT}(\mathcal{P}) = \mathrm{OPT}(\mathcal{P},\gamma)$ 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

$$\begin{aligned}\mathrm{OPT}(\mathcal{P},\gamma) =&\frac{1}{2}\Bigg(\max_{z\in[0,1]}\left\{(1-\gamma)z + \Pi(\tfrac{1}{2}\leq \gamma z)\right\}\\ &+\max_{z\in[0,1]}\left\{-(1-\gamma)z + \Pi(\tfrac{1}{2}\leq (2-\gamma)z) + \Pi((2-\gamma)z\leq 1)\right\}\Bigg)\\ =&\frac{1}{2}\Bigg((1-\gamma) + \Pi(\tfrac{1}{2}\leq\gamma)\\ &+\max_{z\in[0,1]}\left\{-(1-\gamma)z + \Pi(\tfrac{1}{2}\leq (2-\gamma)z) + \Pi((2-\gamma)z\leq 1)\right\}\Bigg)\end{aligned}$$

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

$$\begin{aligned}\mathrm{OPT}(\mathcal{P}) =&\frac{1}{2}\left(\max_{z\in[0,1],\gamma\in[0.5,1)}(1-\gamma)-(1-\gamma)z+\Pi(\tfrac{1}{2}\leq(2-\gamma)z)+\Pi((2-\gamma)z\leq 1)\right)\\ =&\frac{1}{2}\left(\max_{\gamma\in[0.5,1)}(1-\gamma)-\tfrac{1-\gamma}{2(2-\gamma)}\right)\end{aligned}$$

The first equality uses the definition of $\mathrm{OPT}(\mathcal{P})$ and that we have restricted γ to be in $[0.5,1)$. The second equality uses that for any $\gamma\in[0.5,1)$ the unique optimal is $z(\gamma)=\frac{1}{2(2-\gamma)}$ as it maximizes the term $-(1-\gamma)z$ by taking the smallest feasible z value that satisfies the cost lower bound. Finally, the function $\xi(\gamma):=(1-\gamma)-\frac{1-\gamma}{2(2-\gamma)}$ 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

$$\begin{aligned}\mathrm{OPT}(\mathcal{P},\gamma) =&\frac{1}{2}\Bigg(\max_{z\in[0,1]}\left\{z^2(1-\tfrac{\gamma}{2})-z\tfrac{\gamma}{2}+\Pi(\tfrac{1}{2}\leq(1-\gamma)z+\tfrac{\gamma}{2})\right\}\\ &+\max_{z\in[0,1]}\left\{\tfrac{\gamma}{2}z^2-z(1-\tfrac{\gamma}{2})+\Pi(\tfrac{1}{2}\leq(1-\gamma)(1-z)+\tfrac{\gamma}{2})\right\}\Bigg)\end{aligned}$$

To understand why $\gamma=0$ is the unique solution let us analyze both maximization problems separately. The expression $\frac{\gamma}{2}z^2-z(1-\frac{\gamma}{2})$ in the second maximization problem is non-positive in $(z,\gamma)\in[0,1]^2$ as we can write it as $(\frac{\gamma}{2}z^2-\frac{1}{2}z)-z(\frac{1}{2}-\frac{\gamma}{2})$ 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 $z^2(1-\frac{\gamma}{2})-z\frac{\gamma}{2}$ 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 $\mathbb{E}[f(z;\theta^*, w)] = z$ and $\mathbb{E}[c(z;\theta^*, w)] = 0.5z$. Then, for any $\gamma \in [0,1]$ we have

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

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

Bound on Δ_{Learn}

Before stating this subsection result, we define an stricter version of Assumption 3.2.1

Assumption B.1.1 ((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 B.1.1. *Run Algorithm 6 with a constant "step-size" rule $\eta_t \leftarrow \eta$ for all $t \geq 1$ where $\eta > 0$. Suppose that Assumption B.1.1 holds and that $c(\cdot;\cdot,\cdot)$ is Lipschitz on its θ 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*

$$\Delta_{Learn} \leq L_c\,(1 + C_h')\,\mathbb{E}\left[\sum_{t=1}^{\tau_A} \|\theta^* - \theta^t\|_\theta \right].$$

Proof. The proof is obtained directly by bounding each term of Δ_{Learn} separately. First,

$$\begin{aligned}\mathbb{E}\left[\sum_{t=1}^{\tau_A} c(z^t;\theta^*,w^t) - c(z^t;\theta^t,w^t)\right] \leq& \mathbb{E}\left[\sum_{t=1}^{\tau_A} \|c(z^t;\theta^*,w^t) - c(z^t;\theta^t,w^t))\|_\infty\right] \\ \leq& L_c \mathbb{E}\left[\sum_{t=1}^{\tau_A} \|\theta^* - \theta^t\|_\theta\right],\end{aligned}$$

where we have used above that $c(\cdot;\cdot,\cdot)$ its Lipschitz on its θ argument. Now, for any pair x, y of real vectors of same dimension it holds $|x^T y| \leq \|x\|_\infty \|y\|_1$. Using the latter fact and again that $c(\cdot;\cdot,\cdot)$ is Lipschitz on its θ argument, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{\tau_A}(c(z^t;\theta^*,w^t)-c(z^t;\theta^t,w^t))^T\lambda^t\right] \leq & \mathbb{E}\left[\sum_{t=1}^{\tau_A}|(c(z^t;\theta^*,w^t)-c(z^t;\theta^t,w^t))^T\lambda^t|\right] \\
\leq & \mathbb{E}\left[\sum_{t=1}^{\tau_A}\|c(z^t;\theta^*,w^t)-c(z^t;\theta^t,w^t))\|_\infty\|\lambda^t\|_1\right] \\
\leq & L_c\mathbb{E}\left[\sum_{t=1}^{\tau_A}\|\lambda^t\|_1\|\theta^*-\theta^t\|_\theta\right] \\
\leq & L_cC_h'\mathbb{E}\left[\sum_{t=1}^{\tau_A}\|\theta^*-\theta^t\|_\theta\right].
\end{aligned}$$

□

Proof That $\mathrm{OPT}(\mathcal{P})=\mathrm{OPT}(\mathcal{P},0)$ 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 $\mathrm{OPT}(\mathcal{P},\gamma) > -\infty$ for all $\gamma\in[0,1]$. 2. That $\mathrm{OPT}(\mathcal{P},\gamma)\leq \mathrm{OPT}(\mathcal{P},0)$ for all $\gamma\in(0,1]$. 3. How to efficiently solve $\mathrm{OPT}(\mathcal{P},0)$. Take $\mathcal{Z}=\{z\in\mathbb{R}_+^K : \sum_{i=1}^K z_i\leq 1\}$ 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'^t to represent matrices of size $d\times n$. Also, to shorten notation, we write $\mathbf{W}$ to define a sequence $\{W^1,\ldots,W^T\}$ of W^t matrices (analogous for $\mathbf{W}'$). The traditional multiplication between a matrix A of size $d\times n$ and a vector x of size n is written as $Ax=((A_1)^Tx,\ldots,(A_d)^Tx)$. The term inside the outer expectation of $\mathrm{OPT}(\mathcal{P},\gamma)$ corresponds to (for $\gamma=1$ the outer expectation can be removed)

$$\begin{aligned}
O(\mathbf{W},\gamma) := \quad & \max_{z^t\in\mathcal{Z}:t\in[T]} \quad (1-\gamma)\sum_{t=1}^T (W^t\theta^*)^Tz^t+\gamma\mathbb{E}_{W'\sim\mathcal{P}}[(W'\theta^*)^Tz^t] \\
& \text{s.t.} \;\; 0.5*T\leq \rho\sum_{t=1}^T\sum_{i=1}^d z_i^t\leq T.
\end{aligned}$$

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

$$\begin{aligned}
\mathrm{OPT}(\mathcal{P},\gamma) &= \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[(1-\gamma)\sum_{t=1}^T (W^t\theta^*)^T z_\gamma^t(\mathbf{W}) + \gamma\mathbb{E}_{W'\sim\mathcal{P}}\left[(W'\theta^*)^T z_\gamma^t(\mathbf{W})\right]\right]\\
&= \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[(1-\gamma)\sum_{t=1}^T (W^t\theta^*)^T z_\gamma^t(\mathbf{W})\right] + \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[\gamma\sum_{t=1}^T \mathbb{E}_{W'\sim\mathcal{P}}[(W'\theta^*)^T z_\gamma^t(\mathbf{W})]\right]\\
&= \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[(1-\gamma)\sum_{t=1}^T (W^t\theta^*)^T z_\gamma^t(\mathbf{W})\right] + \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[\mathbb{E}_{\mathbf{W}'\sim\mathcal{P}^T}\left[\gamma\sum_{t=1}^T (W'^t\theta^*)^T z_\gamma^t(\mathbf{W})\right]\right]\\
&= \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[(1-\gamma)\sum_{t=1}^T (W^t\theta^*)^T z_\gamma^t(\mathbf{W})\right] + \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T,\mathbf{W}'\sim\mathcal{P}^T}\left[\gamma\sum_{t=1}^T ((W^t)^T\theta^*)^T z_\gamma^t(\mathbf{W}')\right]\\
&= \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[\sum_{t=1}^T (W^t\theta^*)^T\left((1-\gamma)z_\gamma^t(\mathbf{W}) + \gamma\mathbb{E}_{\mathbf{W}'\sim\mathcal{P}^T}[z_\gamma^t(\mathbf{W}')]\right)\right]\\
&\le \mathbb{E}_{\mathbf{W}\sim\mathcal{P}^T}\left[\sum_{t=1}^T (W^t\theta^*)^T z_0^t(\mathbf{W})\right] = \mathrm{OPT}(\mathcal{P},0).
\end{aligned}$$

The second equality uses the linearity of the expectation operator, the third uses that each W'^t is sampled i.i.d. from $\mathcal{P}$, the fourth that $\mathbf{W}$ and $\mathbf{W}'$ 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 $\mathbf{z}_0(\mathbf{W})$. In particular, the last inequality uses that $(1-\gamma)\mathbf{z}_\gamma(\mathbf{W}) + \gamma\mathbb{E}_{\mathbf{W}'\sim\mathcal{P}^T}[\mathbf{z}_\gamma(\mathbf{W}')]$ is a feasible solution of $O(\mathbf{W},0)$. Finally, notice that for any given $\mathbf{W}$ solving $O(\mathbf{W},0)$ is equivalent to solving the following knapsack problem

$$\begin{aligned}
O(\mathbf{W},0) = \max_{y^t\in[0,1]:t\in T} \quad & \sum_{t=1}^T \left(\max_{i\in[d]}(W_i^t)^T\theta^*\right) y^t\\
\text{s.t.} \quad & 0.5 * T \le \rho\sum_{t=1}^T y^t \le T.
\end{aligned}$$

Let $\{m_1,\ldots,m_T\}$ represent the sequence $\{\max_{i\in[d]}(W_i^t)^T\theta^*\}_{t=1}^T$ ordered from biggest to smallest value. Then, is not hard to see that

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

where $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ are the traditional ceiling and floor integer functions respectively.

B.2 Proofs

Proof of Proposition 3.1.1

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

$$
\begin{aligned}
&\mathrm{OPT}(\mathcal{P}, \gamma)\\
&= \mathbb{E}_{\mathcal{P}^T}\left[\begin{array}{c} \max_{z^t \in \mathcal{Z}: t \in [T]} \quad \sum_{t=1}^T (1-\gamma) f(z^t; \theta^*, w^t) + \gamma \mathbb{E}_{\mathcal{P}}[f(z^t; \theta^*, w)] \\ \text{s.t.} \;\; T\alpha_k b_k \le \sum_{t=1}^T (1-\gamma) c_k(z^t; \theta^*, w^t) + \gamma \mathbb{E}_{\mathcal{P}}[c_k(z^t; \theta^*, w)] \le T b_k \text{ for all } k \in [K] \end{array}\right]\\
&\le \mathbb{E}_{\mathcal{P}^T}\left[\max_{z^t \in \mathcal{Z}: t \in [T]} \left\{ \sum_{t=1}^T (1-\gamma)\left(f(z^t; \theta^*, w^t) - \lambda^T c(z^t; \theta^*, w^t)\right) + \gamma \mathbb{E}_{\mathcal{P}}[f(z^t; \theta^*, w) - \lambda^T c(z^t; \theta^*, w)] \right\} + Tp(\lambda)\right]\\
&= \mathbb{E}_{\mathcal{P}^T}\left[\sum_{t=1}^T \max_{z^t \in \mathcal{Z}: t \in T} (1-\gamma)\left(f(z^t; \theta^*, w^t) - \lambda^T c(z^t; \theta^*, w^t)\right) + \gamma \mathbb{E}_{\mathcal{P}}[f(z^t; \theta^*, w) - \lambda^T c(z^t; \theta^*, w)]\right] + Tp(\lambda)\\
&\le (1-\gamma)\mathbb{E}_{\mathcal{P}^T}\left[\sum_{t=1}^T \max_{z^t \in \mathcal{Z}: t \in T} f(z^t; \theta^*, w^t) - \lambda^T c(z^t; \theta^*, w^t)\right]\\
&+ \gamma \mathbb{E}_{\mathcal{P}^T}\left[\sum_{t=1}^T \max_{z^t \in \mathcal{Z}: t \in T} \mathbb{E}_{\mathcal{P}}[f(z^t; \theta^*, w) - \lambda^T c(z^t; \theta^*, w)]\right] + Tp(\lambda)\\
&\le (1-\gamma) T \mathbb{E}_{\mathcal{P}}\left[\max_{z \in \mathcal{Z}} f(z; \theta^*, w) - \lambda^T c(z; \theta^*, w)\right] + \gamma T \max_{z \in \mathcal{Z}} \mathbb{E}_{\mathcal{P}}\left[f(z; \theta^*, w) - \lambda^T c(z; \theta^*, w)\right] + Tp(\lambda)\\
&\le (1-\gamma) T \mathbb{E}_{\mathcal{P}}\left[\varphi(\lambda; \theta^*, w)\right] + \gamma T \mathbb{E}_{\mathcal{P}}\left[\max_{z \in \mathcal{Z}} f(z; \theta^*, w) - \lambda^T c(z; \theta^*, w)\right] + Tp(\lambda)\\
&= T \mathbb{E}_{\mathcal{P}}\left[\varphi(\lambda; \theta^*, w)\right] + Tp(\lambda)\\
&= TD(\lambda; \theta^*)
\end{aligned}
$$

The first equality is the definition of $\mathrm{OPT}(\mathcal{P}, \gamma)$, 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 $max_{z \in \mathcal{Z}} \{a(z) + b(z)\} \le max_{z \in \mathcal{Z}}\, a(z) + max_{z \in \mathcal{Z}}$ $b(z)$, 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 $\max_{z \in \mathcal{Z}} \mathbb{E}_{\mathcal{P}}[\cdot] \le \mathbb{E}_{\mathcal{P}}[\max_{z \in \mathcal{Z}} \cdot]$. Finally, we use the definition of $\varphi(\cdot; \cdot, \cdot)$ again and the fact that $\gamma + (1-\gamma) = 1$. □

Proof of Proposition 3.1.2

Proof. First note that the $p(\cdot)$ function used inside $D(\cdot; \cdot)$ is convex since $b \ge 0$ and $\alpha \in [-1, 1)^K$. We need to prove that $D(\lambda; \theta) + \mathbb{E}_{\mathcal{P}}[\tilde{g}(\lambda; \theta, w)]^T(\lambda' - \lambda) \le D(\lambda'; \theta)$ for any $\lambda \in \Lambda$

and $\lambda' \in \Lambda$. Let p' be any member of $\partial p(\lambda)$, we have

$$\begin{aligned} D(\lambda;\theta) + \mathbb{E}_{\mathcal{P}}[\tilde{g}(\lambda;\theta,w)]^T(\lambda'-\lambda) =& \mathbb{E}_{\mathcal{P}}[\varphi(\lambda;\theta,w) + p(\lambda) + \tilde{g}(\lambda;\theta,w)^T(\lambda'-\lambda)] \\ =& \mathbb{E}_{\mathcal{P}}[f(z(\lambda;\theta,w);\theta,w) - (\lambda')^T c(z(\lambda;\theta,w);\theta,w) \\ & + p(\lambda) + p'^T(\lambda'-\lambda)] \\ \le& \mathbb{E}_{\mathcal{P}}[f(z(\lambda;\theta,w);\theta,w) - (\lambda')^T c(z(\lambda;\theta,w);\theta,w) + p(\lambda')] \\ \le& D(\lambda';\theta). \end{aligned}$$

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 inequlity for $p(\cdot)$, and the second inequality uses the definition of $D(\lambda';\theta)$. ☐

Intermediate Results

The following propositions were not mentioned in the paper. Proposition B.2.1 shows an inequality that holds for Step 7. of Algorithm 6 under the conditions given for Λ and $h(\cdot)$ on the paper. Propositions B.2.2 and B.2.3 are intermediate steps to prove Theorem 3.2.1. Proposition B.2.2 bounds $T - \tau_A$ in expectation. Proposition B.2.3 shows an upper bound for the regret that Algorithm 6 up to period τ_A. Proposition B.2.4 is the key result needed to prove Proposition 3.2.1.

Proposition B.2.1. *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 $h(\cdot): \Lambda \to \mathbb{R}$ 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 $\lambda^+ = \arg\min_{\lambda \in \Lambda} \lambda^T \tilde{g}^t + \frac{1}{\eta} V_h(\lambda, \lambda')$. 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 $\dot{h}_k(\lambda_k^+) \ge \dot{h}_k(\lambda_k') - \eta \tilde{g}_k$ if $\lambda_k^+ = 0$.*

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

Proof. Notice that $\min_{\lambda \in \Lambda} \lambda^T \tilde{g}^t + \frac{1}{\eta} V_h(\lambda, \lambda') = \sum_{k \in [K]} \min_{\lambda_k \in \Lambda_k} \phi_k(\lambda_k; \lambda_k', \tilde{g}_k)$ with $\phi_k(\lambda_k; \lambda_k', \tilde{g}_k) := \tilde{g}_k \lambda_k + \frac{1}{\eta}(h_k(\lambda_k) - h_k(\lambda_k') - \dot{h}_k(\lambda_k')(\lambda_k - \lambda_k'))$ 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 λ_k^+ exists for each $k \in [K]$. Also, λ_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 $\Pi_+(\cdot): \mathbb{R} \to \{0\} \cup \{\infty\}$ as the convex function that takes the value of 0 if its input is non-negative and ∞ otherwise. Then, the minimization problem for dimension k can be re-written as $\min_{\lambda_k \in \Lambda_k} \phi_k(\lambda_k; \lambda_k', \tilde{g}_k) + \Pi_+(\lambda_k)$. First order necessary optimality conditions imply $0 \in \partial(\phi_k(\lambda_k^+; \lambda_k', \tilde{g}_k) + \Pi_+(\lambda_k^+))$. Then, there exists $y \in \partial(\Pi_+(\lambda_k^+))$, such that $\dot{h}_k(\lambda_k^+) = \dot{h}(\lambda_k') - \eta\tilde{g}_k - \eta y$. The result is obtained directly using that $\partial(\Pi_+(\lambda_k))$ is equal to $\{0\}$ when $\lambda_k > 0$ and equal to $\mathbb{R}_-$ when $\lambda_k = 0$. □

Proposition B.2.2. *Run Algorithm 6 with a constant "step-size" rule $\eta_t \leftarrow \eta$ for all $t \geq 1$ where $\eta > 0$. Suppose that Assumption B.1.1 holds and take τ_A as in Definition 3.2.1. Then,*

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

Proof. Let $k' \in [K]$ be the index of the first violated upper cost bound, *i.e.* the index which activates the stop time τ_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 6 defines $\tilde{g}_{k'}^t = -c_{k'}(z^t;\theta^t, w^t) + b_{k'}\left(\mathbb{1}(\lambda_{k'} \geq 0) + \alpha_{k'}\mathbb{1}(\lambda_{k'} < 0)\right)$, which can be upper bounded by $\tilde{g}_{k'}^t \leq -c_{k'}(z^t;\theta^t, w^t) + b_{k'}$. Using the definition of τ_A and $\tilde{g}_{k'}^t$ we have

$$\begin{aligned}
\sum_{t=1}^{\tau_A} \tilde{g}_{k'}^t \leq& b_{k'}\tau_A - \sum_{t=1}^{\tau_A} c_{k'}(z^t;\theta^*, w^t) + \left(\sum_{t=1}^{\tau_A}(c_{k'}(z^t;\theta^*, w^t) - c_{k'}(z^t;\theta^t, w^t))\right) \\
\leq& b_{k'}\tau_A - b_{k'}T + \bar{C} + \left(\sum_{t=1}^{\tau_A}(c_{k'}(z^t;\theta^*, w^t) - c_{k'}(z^t;\theta^t, w^t))\right) \\
\Rightarrow T - \tau_A \leq& \frac{1}{b_{k'}}\left(\bar{C} - \sum_{t=1}^{\tau_A}\tilde{g}_{k'}^t\right) + \frac{1}{b_{k'}}\left(\sum_{t=1}^{\tau_A}(c_{k'}(z^t;\theta^*, w^t) - c_{k'}(z^t;\theta^t, w^t))\right).
\end{aligned}$$

Using that our update rule satisfies $\dot{h}_{k'}(\lambda_{k'}^{t+1}) \geq \dot{h}_{k'}(\lambda_{k'}^t) - \eta\tilde{g}_{k'}^t$ for all $t \leq \tau_A$ and the definitions of $\underline{b}$ and C_h, we get

$$\begin{aligned}
-\sum_{t=1}^{\tau_A}\tilde{g}_{k'}^t \leq& \frac{1}{\eta}\left(\dot{h}_{k'}(\lambda_{k'}^{\tau_A+1}) - \dot{h}_{k'}(\lambda_{k'}^1)\right) \\
\Rightarrow T - \tau_A \leq& \frac{\bar{C}}{b_{k'}} + \frac{\dot{h}_{k'}(\lambda_{k'}^{\tau_A+1}) - \dot{h}_{k'}(\lambda_{k'}^1)}{\eta b_{k'}} + \left(\frac{\sum_{t=1}^{\tau_A}(c_{k'}(z^t;\theta^*, w^t) - c_{k'}(z^t;\theta^t, w^t))}{b_{k'}}\right) \\
\Rightarrow \mathbb{E}\left[T - \tau_A\right] \leq& \frac{\bar{C}}{\underline{b}} + \frac{C_h + \|\nabla h(\lambda^1)\|_\infty}{\eta\underline{b}} + \left(\frac{\|\mathbb{E}[\sum_{t=1}^{\tau_A} c(z^t;\theta^*, w^t) - c(z^t;\theta^t, w^t)]\|_\infty}{\underline{b}}\right)
\end{aligned}$$

□

Proposition B.2.3. *Run Algorithm 6 with a constant "step-size" rule $\eta_t \leftarrow \eta$ for all $t \geq 1$ where $\eta > 0$. Denote $\bar{\lambda}^{\tau_A} = \frac{\sum_{t=1}^{\tau_A}\lambda^t}{\tau_A}$ (τ_A as in Definition 3.2.1). It holds*

$$\begin{aligned}
\mathbb{E}\left[\tau_A D(\bar{\lambda}^{\tau_A};\theta^*) - \sum_{t=1}^{\tau_A} f(z^t;\theta^t, w^t)\right] \leq& \frac{2(\bar{C}^2 + \bar{b}^2)}{\sigma_1}\eta\mathbb{E}[\tau_A] + \frac{1}{\eta}V_h(\lambda, \lambda^1) \\
&+ \mathbb{E}\left[\sum_{t=1}^{\tau_A}(c(z^t;\theta^*, w^t) - c(z^t;\theta^t, w^t))^T\lambda^t\right].
\end{aligned}$$

Proof. For clarity, we sometimes use $\mathbb{E}_w[\cdot]$, $\mathbb{E}_{w^t}[\cdot]$, or $\mathbb{E}_{\mathcal{H}^{t-1}}[\cdot]$ 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 6. Call $\tilde{g}^t$ the vector obtained in Step 6. and define $\mathbb{E}[\tilde{g}^t] = g^t$. The proof is composed of three steps. 1. Bounding $\tilde{g}^t$. 2. Upper bounding $\mathbb{E}\left[\sum_{s=1}^{\tau_A}(g^s)^T(\lambda^s - \lambda)\right]$. 3. Lower bounding $\mathbb{E}\left[\sum_{s=1}^{\tau_A}(g^s)^T(\lambda^s - \lambda))\right]$. The upper and lower bounds match the left and right hand side of the terms in Proposition B.2.3.

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

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

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

$$\begin{aligned}
&\mathbb{E}_{w^t}[(\tilde{g}^t)^T(\lambda^t - \lambda)|\lambda^t,\theta^t] \\
\leq&\mathbb{E}_{w^t}\left[(\tilde{g}^t)^T(\lambda^t - \lambda^{t+1}) + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \frac{1}{\eta}V_h(\lambda,\lambda^{t+1}) - \frac{1}{\eta}V_h(\lambda^{t+1},\lambda^t)\big|\lambda^t,\theta^t\right] \\
\leq&\mathbb{E}_{w^t}\left[(\tilde{g}^t)^T(\lambda^t - \lambda^{t+1}) + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \frac{1}{\eta}V_h(\lambda,\lambda^{t+1}) - \frac{\sigma_1}{2\eta}\|\lambda^{t+1} - \lambda^t\|_1^2\big|\lambda^t,\theta^t\right] \\
\leq&\mathbb{E}_{w^t}\left[\frac{\eta}{\sigma_1}\|\tilde{g}^t\|_\infty^2 + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \frac{1}{\eta}V_h(\lambda,\lambda^{t+1})\big|\lambda^t,\theta^t\right] \\
\leq&\frac{2\eta}{\sigma_1}(\bar{C}^2 + \bar{b}^2) + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \mathbb{E}_{w^t}\left[\frac{1}{\eta}V_h(\lambda,\lambda^{t+1})\big|\lambda^t,\theta^t\right],
\end{aligned} \tag{B.1}$$

where the first inequality is due to the three point property (Lemma 4.1 of [18]), 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 σ_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 $(a^1)^Ta^2 + 0.5\|a^1\|_\infty^2 \geq -0.5\|a^2\|_1^2$, and the final inequality is just understanding which terms are constant under the conditional expectation. Taking $E_{\mathcal{H}^{t-1}}[\cdot]$ over both sides of equation (B.1) and using the law of total expectation we get

$$\mathbb{E}[\eta(g^t)^T(\lambda^t - \lambda)] \leq \frac{2(\bar{C}^2 + \bar{b}^2)}{\sigma_1}\eta^2 + \mathbb{E}\left[V_h(\lambda,\lambda^t)\right] - \mathbb{E}\left[V_h(\lambda,\lambda^{t+1})\right], \tag{B.2}$$

since the pair (λ^t,θ^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 τ_A, using the telescopic property and that $V_h(\cdot,\cdot)$ is non-negative we obtain

$$\mathbb{E}\left[\sum_{s=1}^{\tau_A}(g^s)^T(\lambda^s - \lambda)\right] \leq \frac{2(\bar{C}^2 + \bar{b}^2)}{\sigma_1}\eta\mathbb{E}[\tau_A] + \frac{V_h(\lambda,\lambda^1)}{\eta}.$$

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

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

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 $\varphi(\lambda^t;\theta^t,w)$ not of $\varphi(\lambda;\theta^t,w)$. Defining $\bar{\lambda}^{\tau_A}:=\frac{1}{\tau_A}\sum_{t=1}^{\tau_A}\lambda^t$, taking $\lambda=(0,0,\ldots,0)$, and summing from one to τ_A we get

$$\begin{aligned}&\sum_{t=1}^{\tau_A}(g^t)^T(\lambda^t-0)\\ \geq&\sum_{t=1}^{\tau_A} D(\lambda^t;\theta^t)-\mathbb{E}_w[f(z(\lambda^t;\theta^t,w);\theta^t,w)]\\ \geq&\sum_{t=1}^{\tau_A}\left(D(\lambda^t;\theta^*)-\mathbb{E}_w[f(z(\lambda^t;\theta^t,w);\theta^*,w)]\right)+\sum_{t=1}^{\tau_A}\left(D(\lambda^t;\theta^t)-D(\lambda^t;\theta^*)\right)\\ &+\sum_{t=1}^{\tau_A}\left(\mathbb{E}_w[f(z(\lambda^t;\theta^t,w);\theta^*,w)-f(z(\lambda^t;\theta^t,w);\theta^t,w)]\right)\\ \geq&\left(\tau_A D(\bar{\lambda}^{\tau_A};\theta^*)-\sum_{t=1}^{\tau_A}\mathbb{E}_w[f(z(\lambda^t;\theta^*,w);\theta^*,w)]\right)+\sum_{t=1}^{\tau_A}\left(D(\lambda^t;\theta^t)-D(\lambda^t;\theta^*)\right)\\ &+\sum_{t=1}^{\tau_A}\left(\mathbb{E}_w[f(z(\lambda^t;\theta^t,w);\theta^*,w)-f(z(\lambda^t;\theta^t,w);\theta^t,w)]\right). \qquad \text{(B.3)}\end{aligned}$$

Taking expectation over (B.3) and using the results from Step 2. we get

$$\begin{aligned}\mathbb{E}\left[\tau_A D(\bar{\lambda}^{\tau_A};\theta^*)-\sum_{t=1}^{\tau_A}\mathbb{E}_w\left[f(z(\lambda^t;\theta^t,w);\theta^*,w)\right]\right]&\leq\frac{2(\bar{C}^2+\bar{b}^2)}{\sigma_1}\eta\mathbb{E}[\tau_A]+\frac{1}{\eta}V_h(0,\lambda^1)\\ +\mathbb{E}\left[\sum_{t=1}^{\tau_A}\mathbb{E}_w[c(z(\lambda^t;\theta^t,w);\theta^t,w)]^T\lambda^t\right]-\mathbb{E}&\left[\sum_{t=1}^{\tau_A}\mathbb{E}_w[c(z(\lambda^t;\theta^t,w);\theta^*,w)]^T\lambda^t\right], \qquad \text{(B.4)}\end{aligned}$$

where we have used the definition of $D(\cdot,\cdot)$ to reduce the second line of (B.4) to use only the cost functions. Equation (B.4) almost matches the conclusion of Theorem 3.2.1 except that (B.4) uses a $\mathbb{E}[\sum_{t=1}^{\tau_A}\mathbb{E}_w[\cdot]]$ term, while the theorem uses $\mathbb{E}[\sum_{t=1}^{\tau_A}\cdot]$. The previous issue is solved using the Optional Stopping Theorem. We prove now that $\mathbb{E}\left[\sum_{t=1}^{\tau_A} f(z(\lambda^t;\theta^t,w^t);\theta^*,w^t)\right]$ equals $\mathbb{E}\left[\sum_{t=1}^{\tau_A}\mathbb{E}_w\left[f(z(\lambda^t;\theta^t,w);\theta^*,w)\right]\right]$ (the analysis for the cost terms appearing in the second line of (B.4) is analogous). First notice

$$\mathbb{E}_w\left[f(z(\lambda;\theta,w);\theta^*,w)|\lambda=\lambda^t,\theta=\theta^t\right]=\mathbb{E}_w\left[f(z(\lambda^t;\theta^t,w);\theta^*,w)|\mathcal{H}^{t-1}\right].$$

Define the martingale $M^t = \sum_{s=1}^t f(z(\lambda^s;\theta^s,w^s);\theta^*,w^s) - \mathbb{E}_w[f(z(\lambda^s;\theta^s,w);\theta^*,w)|\mathcal{H}^{s-1}]$ for all $t \le T$. Using that τ_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:

$$\mathbb{E}\left[\sum_{t=1}^{\tau_A} \mathbb{E}_w\left[f(z(\lambda^t;\theta^t,w);\theta^*,w)|\mathcal{H}^{t-1}\right]\right] = \mathbb{E}\left[\sum_{t=1}^{\tau_A} f(z(\lambda^t;\theta^t,w^t);\theta^*,w^t)\right]$$

concluding the proof. □

Proposition B.2.4. *Run Algorithm 6 with a constant "step-size" rule $\eta_t \leftarrow \eta$ for all $t \ge 1$ where $\eta > 0$. Using δ_θ as in Definition 3.2.2, for each $t \in [T-1]$ it holds (here we use 0 to refer to the zero-vector $(0,\dots,0)$ of dimension K):*

$$\mathbb{E}\left[V_h(0,\lambda^{t+1})\middle|\lambda^t,\theta^t\right] \le \eta\left(\frac{2\eta}{\sigma_1}(\bar{C}^2+\bar{b}^2) + 2\bar{f} - \delta_{\theta^t}\|\lambda^t\|_1\right) + V_h(0,\lambda^t).$$

Proof. Let $\tilde{g}^t$ be the λ^t stochastic subgradient obtained in Step 6. of Algorithm 6. Here we abuse notation and use, *e.g.*, $\mathbb{E}[\tilde{g}^t|\lambda^t,\theta^t]$ to represent that $\tilde{g}^t$ is a random variable on w given a fixed pair $(\lambda^t,\theta^t) \in (\Lambda\times\Theta)$. The following bound holds

$$\mathbb{E}_{\mathcal{P}}[\|\tilde{g}^t\|_\infty^2] \le \mathbb{E}[(\|c(z^t;\theta^t,w^t)\|_\infty + \|b\|_\infty)^2] \le 2\mathbb{E}[\|c(z^t;\theta^t,w^t)\|_\infty^2 + \|b\|_\infty^2] \le 2(\bar{C}^2+\bar{b}^2).$$

For any $\lambda\in\Lambda$ we have

$$\begin{aligned}
&\mathbb{E}[\tilde{g}^t|\lambda^t,\theta^t]^T(\lambda^t-\lambda)\\
=&\mathbb{E}[(\tilde{g}^t)^T(\lambda^t-\lambda)|\lambda^t,\theta^t]\\
\le&\mathbb{E}\left[(\tilde{g}^t)^T(\lambda^t-\lambda^{t+1}) + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \frac{1}{\eta}V_h(\lambda,\lambda^{t+1}) - \frac{1}{\eta}V_h(\lambda^{t+1},\lambda^t)\middle|\lambda^t,\theta^t\right]\\
\le&\mathbb{E}\left[(\tilde{g}^t)^T(\lambda^t-\lambda^{t+1}) + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \frac{1}{\eta}V_h(\lambda,\lambda^{t+1}) - \frac{\sigma_1}{2\eta}\|\lambda^{t+1}-\lambda^t\|_1^2\middle|\lambda^t,\theta^t\right]\\
\le&\mathbb{E}\left[\frac{\eta}{\sigma_1}\|\tilde{g}^t\|_\infty^2 + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \frac{1}{\eta}V_h(\lambda,\lambda^{t+1})\middle|\lambda^t,\theta^t\right]\\
\le&\frac{2\eta}{\sigma_1}(\bar{C}^2+\bar{b}^2) + \frac{1}{\eta}V_h(\lambda,\lambda^t) - \mathbb{E}\left[\frac{1}{\eta}V_h(\lambda,\lambda^{t+1})\middle|\lambda^t,\theta^t\right],
\end{aligned}$$

where we have used linearity of the expectation, the three point property, that $V_h(\cdot,\cdot)$ is σ_1 strongly convex on with respect to the $\|\cdot\|_1$ norm, Cauchy-Schwartz, and the bound for $\mathbb{E}[\|\tilde{g}^t\|_\infty^2]$ obtained before (same steps as in Step 1. and 2. of Proof B.2.3). Choosing $\lambda = (0,\dots,0)$ we get

$$\mathbb{E}\left[V_h(0,\lambda^{t+1})\middle|\lambda^t,\theta^t\right] \le \eta\left(\frac{2\eta}{\sigma_1}(\bar{C}^2+\bar{b}^2) - \mathbb{E}[\tilde{g}^t|\lambda^t,\theta^t]^T\lambda^t\right) + V_h(0,\lambda^t).$$

To finish the proof we now show that $\mathbb{E}[\tilde{g}^t|\lambda^t,\theta^t]^T\lambda^t \geq \|\lambda^t\|_1\delta_{\theta^t} - 2\bar{f}$. Notice first that for any $(\lambda^t,\theta^t)\in(\Lambda\times\Theta)$ we have $\mathbb{E}[\tilde{g}^t(w)]^T\lambda^t = -\mathbb{E}[c(z(\lambda^t;\theta^t,w);\theta^t,w)]^T\lambda^t + p(\lambda^t)$ using that by definition $p(\lambda)=\sum_{k\in[K]} b_k([\lambda_k]_+ - \alpha_k[-\lambda_k]_+)$. Let $\{z(w)\}_{w\in\mathcal{W}}$ be a series that satisfies $\delta_{\theta^t} = \mathbb{E}_{\mathcal{P}}[\min\{\|Tb_k - c_k(z(w);\theta^t,w)\|_\infty, \|c_k(z(w);\theta^t,w) - T\alpha_k b_k\|_\infty\}]$. Then,

$$\begin{aligned}
&\mathbb{E}[\tilde{g}^t|\lambda^t,\theta^t]^T\lambda^t\\
&= D(\lambda^t;\theta^t) - \mathbb{E}_{\mathcal{P}}[f(z(\lambda^t;\theta^t,w);\theta^t,w)]\\
&\geq \mathbb{E}_{\mathcal{P}}[\max_{z\in\mathcal{Z}} f(z;\theta^t,w) + \sum_{k\in[K]}\left([\lambda_k^t]_+(b_k - \mathbb{E}_{\mathcal{P}}[c_k(z;\theta^t,w)]) + [-\lambda_k^t]_+(\mathbb{E}_{\mathcal{P}}[c_k(z;\theta^t,w)] - \alpha_k b_k)\right)] - \bar{f}\\
&\geq \mathbb{E}_{\mathcal{P}}[f(z(w);\theta^t,w) + \sum_{k\in[K]}\left([\lambda_k^t]_+(b_k - \mathbb{E}_{\mathcal{P}}[c_k(z(w);\theta^t,w)]) + [-\lambda_k^t]_+(\mathbb{E}_{\mathcal{P}}[c_k(z(w);\theta^t,w)] - \alpha_k b_k)\right)] - \bar{f}\\
&\geq \mathbb{E}_{\mathcal{P}}[\sum_{k\in[K]}[\lambda_k^t]_+(b_k - \mathbb{E}_{\mathcal{P}}[c_k(z(w);\theta^t,w)]) + [-\lambda_k^t]_+(\mathbb{E}_{\mathcal{P}}[c_k(z(w);\theta^t,w)] - \alpha_k b_k)] - 2\bar{f}\\
&\geq \|\lambda^t\|_1\delta_{\theta^t} - 2\bar{f},
\end{aligned}$$

where we have used that $\|\lambda^t\|_1 = \sum_{k\in[K]}([\lambda_k^t]_+ + [-\lambda_k^t]_+)$ and the definition of $D(\lambda^t;\theta^t)$, $\bar{f}$, and δ_{θ^t}. □

Proof of Theorem 3.2.1

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

$$\begin{aligned}
OPT(\mathcal{P}) &\leq \frac{t'}{T}OPT(\mathcal{P}) + \frac{T-t'}{T}OPT(\mathcal{P})\\
&\leq t'D(\bar{\lambda}^{t'};\theta^*) + (T-t')\bar{f},
\end{aligned}$$

where we have used Proposition 3.1.1 and that a loose upper bound for $OPT(\mathcal{P})$ is $T\bar{f}$. Therefore,

$$\begin{aligned}
&Regret(A|\mathcal{P})\\
=&OPT(\mathcal{P}) - R(A|\mathcal{P})\\
\leq&\mathbb{E}\left[\tau_A D(\bar{\lambda}^{\tau_A};\theta^*) + (T-\tau_A)\bar{f} - \sum_{t=1}^{\tau_A} f(z^t;\theta^*,w^t)\right]\\
=&\mathbb{E}\left[\tau_A D(\bar{\lambda}^{\tau_A};\theta^*) - \sum_{t=1}^{\tau_A} f(z^t;\theta^*,w^t)\right] + \mathbb{E}[T-\tau_A]\bar{f}\\
\leq&\frac{2(\bar{C}^2+\bar{b}^2)}{\sigma_1}\eta\mathbb{E}[\tau_A] + \frac{1}{\eta}V_h(0,\lambda^1) + \frac{\bar{f}}{\underline{b}}\left(\bar{C} + \frac{C_h + \|\nabla h(\lambda^1)\|_\infty}{\eta}\right)\\
&+\mathbb{E}\left[\sum_{t=1}^{\tau_A}(c(z^t;\theta^*,w^t) - c(z^t;\theta^t,w^t))^T\lambda^t\right] + \frac{\bar{f}}{\underline{b}}\left\|\mathbb{E}\left[\sum_{t=1}^{\tau_A} c(z^t;\theta^*,w^t) - c(z^t;\theta^t,w^t)\right]\right\|_\infty,
\end{aligned}$$

where in the first inequality we have used the definition of $R(A|\mathcal{P})$ and the fact that Algorithm 6 runs for τ_A periods. The second inequality is obtained directly from Propositions B.2.2 and B.2.3. $\square$

Proof of Proposition 3.2.1

Proof. A direct application of Proposition B.2.4 shows that whenever $\|\lambda^t\|_1 \geq C^{\triangleright}/\delta$ we have $\mathbb{E}[V_h(0,\lambda^{t+1})|(\lambda^t,\theta^t)] \leq V_h(0,\lambda^t)$. Then, for any $(\lambda^t,\theta^t) \in \Lambda \times \Theta$ we have

$$\begin{aligned}
&\mathbb{E}[V_h(0,\lambda^{t+1})|(\lambda^t,\theta^t)] \leq \max\big\{ \max_{\|\lambda\|_1 \leq \delta^{-1}C^{\triangleright}} V_h(0,\lambda) + \eta C^{\triangleright},\, V_h(0,\lambda^1)\big\} \\
\Rightarrow &\mathbb{E}[V_h(0,\lambda^{t+1})] \leq \max\big\{ \max_{\|\lambda\|_1 \leq \delta^{-1}C^{\triangleright}} V_h(0,\lambda) + \eta C^{\triangleright},\, V_h(0,\lambda^1)\big\}
\end{aligned}$$

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 $\max_{\|\lambda\|_1 \leq \delta^{-1}C^{\triangleright}} 0.5\|\lambda\|_2^2 = 0.5(C^{\triangleright}/\delta)^2$. Using Jensen inequality and previous results we get

$$\mathbb{E}[\|\lambda^{t+1}\|_2] \leq \max\big\{\sqrt{(C^{\triangleright}/\delta)^2 + 2\eta C^{\triangleright}}, \|\lambda^1\|_2\big\}$$

Finally, since $\|\lambda\|_\infty \leq \|\lambda\|_2$ for any $\lambda \in \Lambda$ we conclude the proof as for any $t \in [T]$ we have $\mathbb{E}[\|\lambda^t\|_\infty] \leq \max\big\{\sqrt{(C^{\triangleright}/\delta)^2 + 2\eta C^{\triangleright}}, \|\lambda^1\|_\infty\big\}$.

$\square$

Proof of Proposition 3.2.2

Proof. Since $\alpha_k \neq -\infty$ by statement, Proposition B.2.1 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 $\dot{h}_k(\lambda^{\tau_A+1}) - \dot{h}_k(\lambda^1) = -\eta\sum_{t=1}^{\tau_A}\tilde{g}_k^t$ regardless of the τ_A value. Then, using the definition of $\tilde{g}^t$ we get

$$\begin{aligned}
&\sum_{t=1}^{\tau_A}\big(b_k(\mathbb{1}(\lambda_k \geq 0) + \alpha_k\mathbb{1}(\lambda_k < 0)) - c_k(z^t;\theta^t,w^t)\big) = \frac{\dot{h}_k(\lambda^1) - \dot{h}_k(\lambda^{\tau_A+1})}{\eta} \\
\Rightarrow &\sum_{t=1}^{\tau_A}\big(b_k(\mathbb{1}(\lambda_k \geq 0) + \alpha_k\mathbb{1}(\lambda_k < 0)) - c_k(z^t;\theta^*,w^t)\big) \\
&= \frac{\dot{h}_k(\lambda^1) - \dot{h}_k(\lambda^{\tau_A+1})}{\eta} + \sum_{t=1}^{\tau_A} c_k(z^t;\theta^t,w^t) - c_k(z^t;\theta^*,w^t).
\end{aligned}$$

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

$$\sum_{t=1}^{\tau_A}(b_k(\mathbb{1}(\lambda_k \geq 0) + \alpha_k\mathbb{1}(\lambda_k < 0))) + (T - \tau_A)\alpha_k b_k \geq T\alpha_k b_k.$$

Combining the previous results and taking expectation we get

$$\begin{aligned} T\alpha_k b_k - \mathbb{E}[\sum_{t=1}^{\tau_A} c_k(z^t;\theta^*,w^t)] \leq \frac{\dot{h}_k(\lambda^1) - \mathbb{E}[\dot{h}_k(\lambda^{\tau_A+1})]}{\eta} + \mathbb{E}[T-\tau_A]\alpha_k b_k \\ + \mathbb{E}\left[\sum_{t=1}^{\tau_A} c_k(z^t;\theta^t,w^t) - c_k(z^t;\theta^*,w^t)\right]. \end{aligned}$$

Finally, we conclude the proof by using Proposition B.2.2 and the definition of C_h. □

B.3 Extra Experimental Details and Results

Bidding Experiment

This experiment is based on data from Criteo [39]. 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 [39] 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 millisecond 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 occur 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 [98] 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 12.

Algorithm 6 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 6, as we do not need to learn the distribution of the highest competing (mp) to solve Step 3. of Algorithm 6 (shown in Algorithm 7). We compare the performance of Algorithm 6 to using the Greedy Heuristic 13. When $\gamma = 1$, Algorithm 13 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

```
Algorithm 12 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, $\{\text{mp}^t\}_{t=1}^T$ the price Criteo paid for each ad spot in the test set in order.
  for $t = 1, \dots, T$ do
    1. Read the $t$ bidding test log and $\text{mp}^t$.
    2. Use model $\sigma$ to obtain estimated conversion probabilities conv_prob. Take $r_k^t = \text{conv\_prob}_k \cdot q_k$ 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 $z^t \geq \text{mp}^t$ then the auction is won, advertiser $k^t$ pays $r_{k^t}^t$ to the bidder (the DSP), the bidder pays $\text{mp}^t$ for the ad spot and obtains $r_{k^t}^t - \text{mp}^t$ as profit.
  end for
```

requirements. Then, we can think of γ as a way to increase/decrease the bids in order to take the budgets into account. (For this example, we can think of Algorithm 6 as an online algorithm for obtaining γ variables per advertiser.)

```
Algorithm 13 Greedy Heuristic($\gamma$)
  Input: Vector $r \in \mathbb{R}_+^K$ and $\gamma > 0$.
  Let $\mathcal{K}'$ be the set of advertisers with non depleted budgets. If $\mathcal{K}' = \emptyset$ do not bid, otherwise bid on behalf of $k^* \in \arg\max_{k \in \mathcal{K}'} r_k$ the amount $\gamma r_{k^*}$.
```

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 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 6 using traditional subgradient descent trying the fixed step sizes $\{1 * 10^{-i}\}_{i=0}^3 \cup \{0.5 * 10^{-i}\}_{i=0}^3$ and $\{0.25 + 0.05 * i\}_{i=0}^{25}$ as γ parameters for the Greedy Heuristic 13. 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 12. 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 6 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 6 and 13 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 B.1. 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

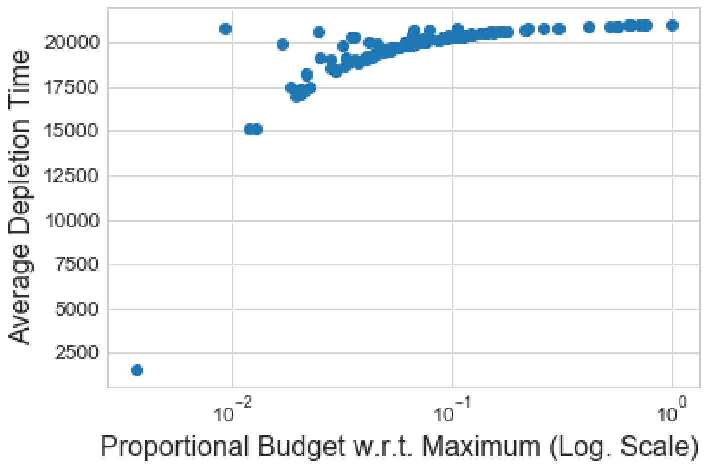


Figure B.1: The x-axis in the figure shows the proportion of an advertiser budget w.r.t. the highest budget between all advertisers (shown on a logarithmic scale).

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 6 and 26 γ values for 13, 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.

Linear Contextual Bandits Experiment

We now describe in detail the methods used to implement Step 1. of Algorithm 6. 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 6 in the following ways.

1. Gaussian Thompson Sampling as in [4]. 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 $B(t+1) = I_d + \sum_{s \in [t]: y^s = 1} W^s_{i(s)} (W^s_{i(s)})^T$ and

$\hat{\theta}^{t+1} = B(t+1)^{-1}(\sum_{s\in[t]:y^s=1} W^s_{i(s)} r^t)$, otherwise $B(t+1) = B(t)$ and $\hat{\theta}^{t+1} = \hat{\theta}^t$. After the prior update, we sample θ^t from $\mathcal{N}(\hat{\theta}^t, \nu^2 B(t)^{-1})$ where $\mathcal{N}(\cdot,\cdot)$ 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, we used $\nu = \frac{\text{rev\_err}}{10} * \sqrt{\log T * n}$ with 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 ν was inspired on [4] 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 6 uses $\theta^t = \hat{\theta}^t$. (This update is a core element of many learning approaches for linear contextual bandits [4, 2] 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 $\{W^s_{i(s)}, r^s\}_{s\in[t-1]:y^s=1}$ with an ℓ_2 penalization parameter of $\alpha = 0.001$.

4. Ridge regression plus error. Same method as above but adds noise to the θ^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 θ^t.

5. Known θ^*. Algorithm 6 using $\theta^t = \theta^*$ for all $t \in [T]$.

Figures B.2 and B.3 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 6 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 B.1, B.2, B.3 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 θ^*' 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 θ^*' 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.

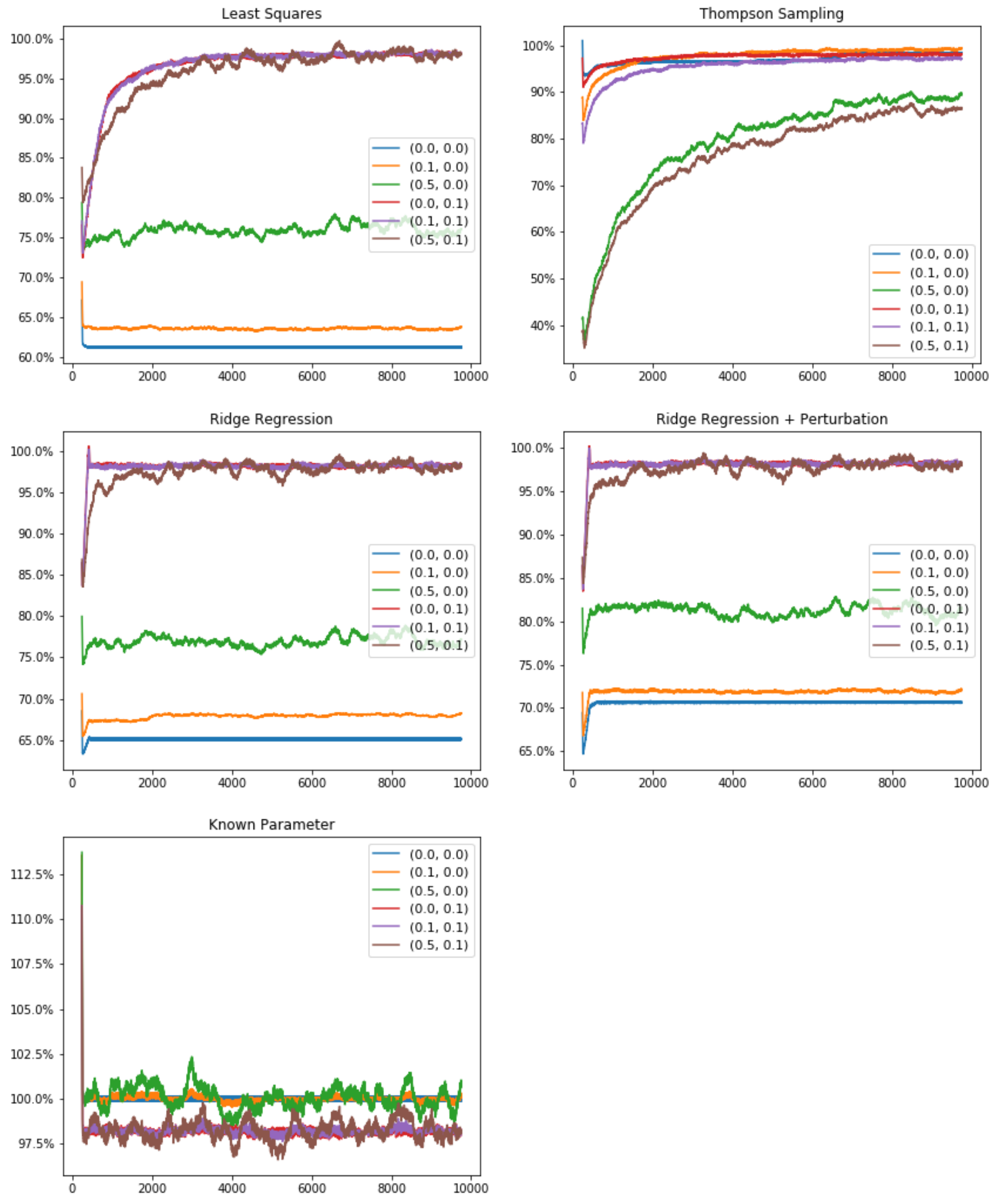


Figure B.2: Moving average revenue for windows of 250 iterations against the proportional best average revenue possible using $d = 5$, $n = 10$.

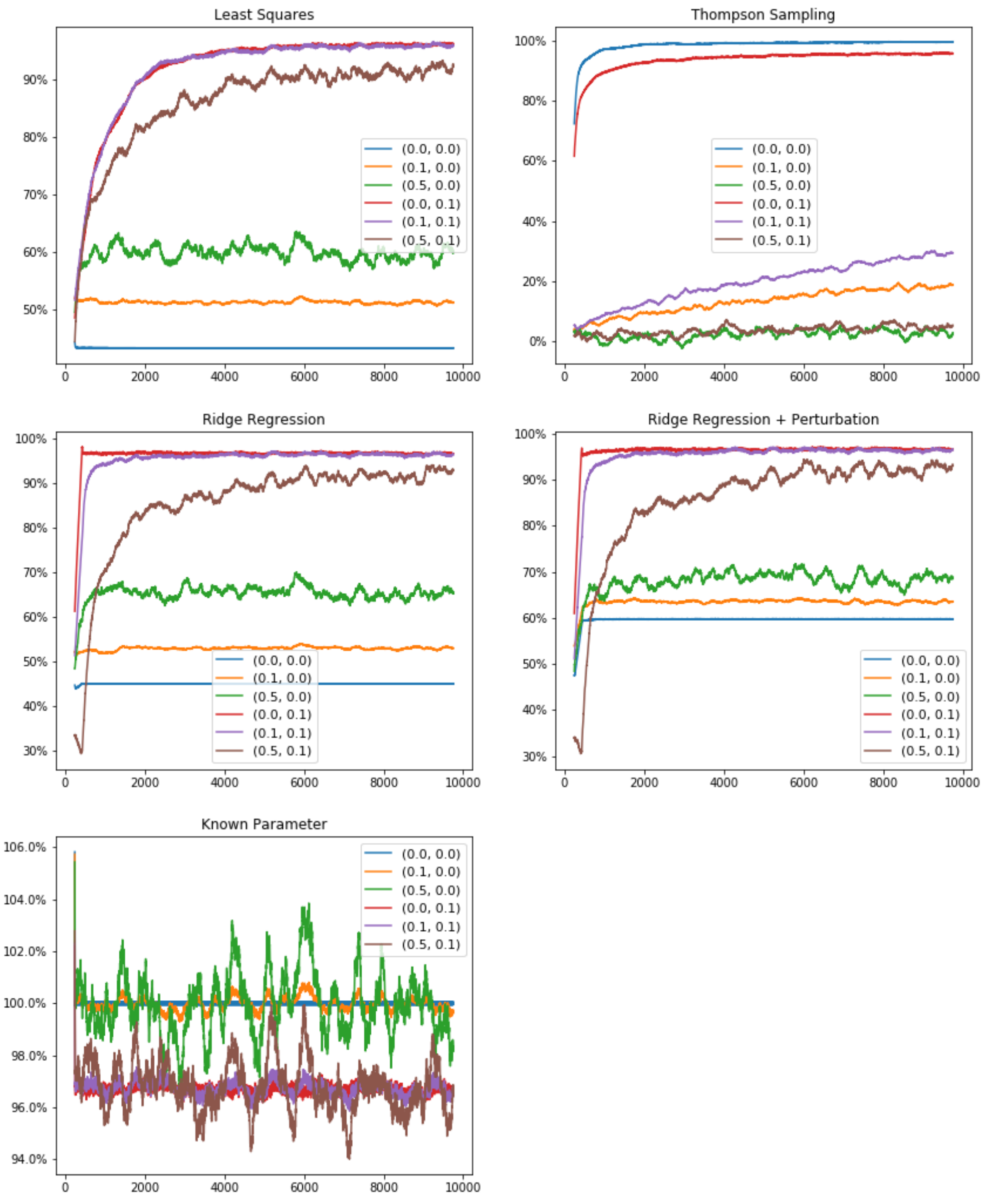


Figure B.3: Moving average revenue for windows of 250 iterations against the proportional best average revenue possible using $d = 50$, $n = 50$.

$T = 1,000$	$d \times n$	$(0.0, 0.0)$	$(0.1, 0.0)$	$(0.5, 0.0)$	$(0.0, 0.1)$	$(0.1, 0.1)$	$(0.5, 0.1)$
Least Squares	5×5	76.2%	77.6%	84.2%	78.9%	79.4%	79.3%
Thompson Sampling	5×5	95.2%	74.2%	21.9%	85.4%	65.4%	19.0%
Ridge Regression	5×5	77.6%	79.0%	85.4%	90.4%	89.8%	83.5%
Ridge Reg. + Perturbation	5×5	80.8%	80.9%	86.0%	90.3%	89.6%	83.5%
Known Parameter	5×5	99.9%	100.1%	100.8%	92.4%	92.4%	92.2%
Least Squares	5×10	60.9%	63.2%	73.4%	80.1%	80.5%	82.6%
Thompson Sampling	5×10	94.2%	90.3%	51.2%	89.4%	85.9%	48.3%
Ridge Regression	5×10	64.5%	67.3%	76.5%	93.0%	92.8%	90.0%
Ridge Reg. + Perturbation	5×10	73.9%	73.9%	81.1%	92.8%	92.6%	90.2%
Known Parameter	5×10	100.0%	100.0%	99.9%	95.5%	95.5%	95.4%
Least Squares	10×5	70.9%	74.6%	78.1%	83.7%	84.0%	82.9%
Thompson Sampling	10×5	94.5%	91.0%	50.6%	89.0%	84.6%	47.6%
Ridge Regression	10×5	71.0%	75.2%	78.8%	92.5%	92.5%	89.0%
Ridge Reg. + Perturbation	10×5	82.0%	84.3%	84.7%	92.3%	92.3%	89.7%
Known Parameter	10×5	99.9%	99.9%	99.7%	94.5%	94.4%	94.2%
Least Squares	10×10	58.5%	62.5%	72.0%	75.7%	75.3%	76.7%
Thompson Sampling	10×10	92.2%	66.6%	14.7%	86.1%	62.4%	15.1%
Ridge Regression	10×10	59.0%	63.4%	72.4%	91.2%	90.4%	84.0%
Ridge Reg. + Perturbation	10×10	72.3%	73.6%	77.2%	90.9%	90.2%	84.1%
Known Parameter	10×10	100.0%	99.9%	99.7%	93.9%	93.9%	93.9%
Least Squares	25×25	44.0%	49.7%	54.0%	64.5%	66.0%	58.9%
Thompson Sampling	25×25	89.1%	5.4%	0.3%	74.4%	6.1%	0.7%
Ridge Regression	25×25	44.1%	50.6%	56.0%	86.1%	78.4%	46.8%
Ridge Reg. + Perturbation	25×25	69.5%	66.7%	61.4%	85.4%	78.0%	46.3%
Known Parameter	25×25	100.0%	100.0%	99.7%	90.7%	90.8%	91.3%
Least Squares	25×50	41.4%	48.1%	56.1%	64.7%	65.1%	68.1%
Thompson Sampling	25×50	89.0%	19.6%	3.3%	82.4%	20.5%	3.7%
Ridge Regression	25×50	43.3%	50.3%	62.7%	90.0%	85.8%	69.7%
Ridge Reg. + Perturbation	25×50	62.8%	64.0%	68.8%	89.5%	85.5%	69.1%
Known Parameter	25×50	100.0%	100.1%	100.3%	93.7%	93.8%	94.1%
Least Squares	50×25	49.1%	53.7%	59.1%	67.7%	68.1%	68.3%
Thompson Sampling	50×25	92.2%	18.3%	2.6%	82.7%	19.5%	2.8%
Ridge Regression	50×25	51.9%	55.9%	64.6%	89.4%	85.3%	67.7%
Ridge Reg. + Perturbation	50×25	70.8%	69.7%	71.8%	89.1%	85.2%	67.6%
Known Parameter	50×25	100.0%	100.0%	100.0%	92.9%	92.9%	92.6%
Least Squares	50×50	42.0%	52.2%	55.7%	62.3%	63.7%	58.7%
Thompson Sampling	50×50	87.5%	5.4%	1.5%	76.0%	6.7%	1.5%
Ridge Regression	50×50	43.6%	54.5%	62.1%	86.8%	76.8%	47.7%
Ridge Reg. + Perturbation	50×50	67.2%	68.8%	66.7%	86.0%	76.6%	47.2%
Known Parameter	50×50	100.0%	100.0%	100.0%	92.0%	91.9%	91.6%

Table B.1: 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 × 5	76.7%	79.4%	87.1%	91.6%	91.5%	90.5%
Thompson Sampling	5 × 5	98.7%	88.6%	42.6%	93.2%	80.9%	36.7%
Ridge Regression	5 × 5	78.1%	79.4%	86.5%	95.1%	94.9%	92.4%
Ridge Reg. + Perturbation	5 × 5	80.0%	79.7%	87.2%	94.9%	94.8%	92.3%
Known Parameter	5 × 5	100.0%	100.0%	99.9%	95.9%	95.9%	96.0%
Least Squares	5 × 10	61.2%	63.5%	75.3%	93.1%	93.3%	92.6%
Thompson Sampling	5 × 10	97.3%	96.0%	71.7%	95.8%	93.0%	68.6%
Ridge Regression	5 × 10	64.9%	67.9%	79.6%	96.5%	96.5%	95.5%
Ridge Reg. + Perturbation	5 × 10	71.0%	71.9%	80.4%	96.4%	96.4%	95.3%
Known Parameter	5 × 10	100.0%	100.0%	100.0%	97.5%	97.5%	97.4%
Least Squares	10 × 5	71.3%	72.3%	80.9%	93.6%	93.4%	93.4%
Thompson Sampling	10 × 5	96.0%	96.4%	70.4%	95.2%	92.1%	67.1%
Ridge Regression	10 × 5	71.5%	73.7%	81.5%	96.3%	96.2%	95.5%
Ridge Reg. + Perturbation	10 × 5	77.0%	80.1%	83.0%	96.2%	96.1%	95.3%
Known Parameter	10 × 5	100.0%	100.0%	100.1%	97.0%	97.0%	97.0%
Least Squares	10 × 10	58.9%	63.3%	70.0%	91.0%	90.9%	91.3%
Thompson Sampling	10 × 10	96.2%	83.9%	29.5%	94.2%	80.7%	30.8%
Ridge Regression	10 × 10	59.4%	63.7%	70.4%	95.6%	95.4%	93.3%
Ridge Reg. + Perturbation	10 × 10	69.2%	69.8%	74.1%	95.5%	95.4%	93.1%
Known Parameter	10 × 10	100.0%	100.0%	100.1%	96.7%	96.6%	96.5%
Least Squares	25 × 25	44.6%	54.0%	58.6%	85.6%	85.6%	78.3%
Thompson Sampling	25 × 25	97.2%	12.6%	1.2%	88.6%	15.0%	1.9%
Ridge Regression	25 × 25	44.8%	54.7%	60.4%	93.4%	91.1%	76.4%
Ridge Reg. + Perturbation	25 × 25	64.9%	64.0%	66.2%	93.2%	90.9%	76.5%
Known Parameter	25 × 25	100.0%	100.1%	100.4%	95.0%	94.9%	94.7%
Least Squares	25 × 50	41.5%	48.1%	57.5%	87.7%	87.4%	84.4%
Thompson Sampling	25 × 50	94.6%	36.2%	7.3%	93.0%	39.9%	8.6%
Ridge Regression	25 × 50	43.5%	49.9%	68.0%	95.0%	94.2%	87.8%
Ridge Reg. + Perturbation	25 × 50	55.7%	58.0%	74.1%	94.9%	94.1%	87.0%
Known Parameter	25 × 50	100.0%	99.9%	99.6%	96.5%	96.5%	96.5%
Least Squares	50 × 25	49.3%	53.0%	57.8%	87.6%	87.9%	85.3%
Thompson Sampling	50 × 25	97.8%	34.3%	5.5%	92.3%	38.9%	7.1%
Ridge Regression	50 × 25	52.2%	55.3%	58.4%	94.6%	93.9%	86.8%
Ridge Reg. + Perturbation	50 × 25	66.0%	65.7%	67.8%	94.4%	93.7%	87.1%
Known Parameter	50 × 25	100.0%	100.0%	100.1%	96.0%	96.0%	96.0%
Least Squares	50 × 50	41.9%	52.7%	60.4%	85.8%	86.2%	79.6%
Thompson Sampling	50 × 50	96.4%	10.0%	1.8%	89.7%	14.3%	2.7%
Ridge Regression	50 × 50	43.6%	53.2%	68.2%	94.0%	91.5%	77.9%
Ridge Reg. + Perturbation	50 × 50	59.9%	61.3%	71.8%	93.7%	91.4%	77.8%
Known Parameter	50 × 50	100.0%	100.0%	100.2%	95.5%	95.5%	95.5%

Table B.2: All percentages shown are the average revenue over 100 simulations divided by the best average revenue achievable (OPT($\mathcal{P}$)).

$T = 10,000$	$d \times n$	$(0.0, 0.0)$	$(0.1, 0.0)$	$(0.5, 0.0)$	$(0.0, 0.1)$	$(0.1, 0.1)$	$(0.5, 0.1)$
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 B.3: All percentages shown are the average revenue over 100 simulations divided by the best average revenue achievable ($\text{OPT}(\mathcal{P})$).

Appendix C

Stochastic In-Face Frank-Wolfe Methods for Non-Convex Optimization and Sparse Neural Network Training

C.1 Proofs in Section 4.1

Proof of Proposition 4.1.1

Proof. It is easily verified that $\tilde{G}(\bar{x},\bar{y}) \geq 0$ and hence $G(\bar{x},\bar{y}) \geq 0$ for all $(\bar{x},\bar{y}) \in S \times \mathbb{R}^q$. Now suppose that $G(\bar{x},\bar{y}) > 0$. Then, either $\tilde{G}(\bar{x},\bar{y}) > 0$ or $\|\nabla_y F(\bar{x},\bar{y})\|_{Y*} > 0$. In the case that $\tilde{G}(\bar{x},\bar{y}) > 0$, let $\tilde{x} \in \arg\max_{x\in S} \left\{\nabla_x F(\bar{x},\bar{y})^T(\bar{x}-x)\right\}$ and define a direction $d \in \mathbb{R}^p \times \mathbb{R}^q$ by $d := (\tilde{x}-\bar{x},0)$. Then, d is a feasible descent direction for (1) and therefore $(\bar{x},\bar{y})$ is not locally optimal. Likewise, if $\|\nabla_y F(\bar{x},\bar{y})\|_{Y*} > 0$, let $\tilde{y} \in \arg\max\limits_{y\in\mathbb{R}^q}\{\nabla_y F(\bar{x},\bar{y})^T y : \|y\|_Y \leq 1\}$ and define $d := (0,-\tilde{y})$. Then d is also a descent direction and therefore $(\bar{x},\bar{y})$ is not locally optimal. □

Proof of Proposition 4.1.2

Proof. By the gradient inequality for differentiable convex functions, it holds that:

$$\begin{aligned} F(\bar{x},\bar{y}) - F^* &\leq \nabla_x F(\bar{x},\bar{y})^T(\bar{x}-x^*) + \nabla_y F(\bar{x},\bar{y})^T(\bar{y}-y^*) \\ &\leq \tilde{G}(\bar{x},\bar{y}) + \|\nabla_y F(\bar{x},\bar{y})\|_{Y*}\|\bar{y}-y^*\|_Y \\ &\leq \sqrt{\tfrac{\bar{C}}{2L_\nabla}} \cdot \tilde{G}(\bar{x},\bar{y})\sqrt{\tfrac{2\bar{L}_\nabla}{\bar{C}}} + R\|\nabla_y F(\bar{x},\bar{y})\|_{Y*} \\ &\leq \max\left\{\sqrt{\tfrac{\bar{C}}{2L_\nabla}}, R\right\} \cdot G(\bar{x},\bar{y}) \ , \end{aligned}$$

where the second inequality uses the definition of $\tilde{G}(\bar{x},\bar{y})$ as well as Hölder's inequality. □

Proof of Lemma 4.1.1

Proof. Let $\sigma_S(\cdot)$ denote the support function of the set S, i.e., $\sigma_S(g) = \max_{x\in S}\left\{g^T x\right\}$. Consider the function $\psi(\cdot,\cdot) : \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$ defined by $\psi(g,h) := (g^T\bar{x} + \sigma_S(-g))\sqrt{\frac{2L_\nabla}{C}} + \|h\|_{Y*}$, which is a convex function of (g,h). Note that $G(\bar{x},\bar{y}) = \psi(\nabla_x F(\bar{x},\bar{y}), \nabla_y F(\bar{x},\bar{y}))$. Finally, Jensen's inequality yields:

$$\mathbb{E}[\hat{G}] = \mathbb{E}[\psi(\hat{g},\hat{h})] \geq \psi(\nabla_x F(\bar{x},\bar{y}), \nabla_y F(\bar{x},\bar{y})) = G(\bar{x},\bar{y}) \ .$$

□

Useful Lemmas

We use the following Lemma to prove the results in this section.

Lemma C.1.1. *Suppose that $(g_1,h_1),\ldots,(g_b,h_b)$ are i.i.d. random vectors in $\mathbb{R}^p\times\mathbb{R}^q$ with mean 0 and satisfying $\mathbb{E}[\|(g_i,h_i)\|_*^2] \leq \sigma^2$ for all $i=1,\ldots,b$. Define $\hat{g} := \frac{1}{b}\sum_{i=1}^b g_i$ and $\hat{h} = \frac{1}{b}\sum_{i=1}^b h_i$. Then, it holds that:*

$$\mathbb{E}[\|(\hat{g},\hat{h})\|_*^2] \ \leq \ \frac{\kappa^2\sigma^2}{b} \ .$$

Proof. Recall that $\kappa_1 := \max_{(x,y)\neq 0} \|(x,y)\|_2/\|(x,y)\| = \max_{(s,t)\neq 0}\|(s,t)\|_*/\|(s,t)\|_2$ as well as $\kappa_2 := \max_{(x,y)\neq 0}\|(x,y)\|/\|(x,y)\|_2 = \max_{(s,t)\neq 0}\|(s,t)\|_2/\|(s,t)\|_*$. Hence, for any $(s,t)\in\mathbb{R}^p\times\mathbb{R}^q$, it holds that:

$$\|(s,t)\|_* \leq \kappa_1\|(s,t)\|_2 \leq \kappa_1\kappa_2\|(s,t)\|_* = \kappa\|(s,t)\|_* \ .$$

Now we have that:

$$\mathbb{E}[\|(\hat{g},\hat{h})\|_*^2] \leq \kappa_1^2\cdot\mathbb{E}[\|(\hat{g},\hat{h})\|_2^2] = \frac{\kappa_1^2}{b}\cdot\mathbb{E}[\|(g_1,h_1)\|_2^2] \leq \frac{\kappa^2}{b}\cdot\mathbb{E}[\|(g_1,h_1)\|_*^2] \leq \frac{\kappa^2\sigma^2}{b} \ ,$$

where the equality in the above chain uses the fact that $(g_1,h_1),\ldots,(g_b,h_b)$ are i.i.d. with mean 0. □

The proof of Theorem 2.1 is based on the following key lemma that bounds the expected progress per iteration.

Lemma C.1.2. *For each $k\geq 0$, let $\mathcal{F}_k$ denote the σ-field of all information gathered after completing iteration $k-1$ of Algorithm 1, i.e., right before starting iteration k, and define $\Delta_k := 8L_\nabla(F(x_k,y_k) - F(x_{k+1},y_{k+1}))$. Then, at every iteration $k\geq 0$, it holds that:*

$$\mathbb{E}[\Delta_k \mid \mathcal{F}_k] \ \geq \ G(x_k,y_k)^2 - \frac{4\alpha_{\text{AD}}\kappa^2\sigma^2}{b_k} \ .$$

Proof.

Case 1: AlternativeDirections = FALSE. Let us first consider the case of not using alternative directions, i.e., AlternativeDirections = FALSE. By Assumption (A1), it is well-known and follows easily from the fundamental theorem of calculus that:

$$F(x,y) \le F(\bar{x},\bar{y}) + \nabla F(\bar{x},\bar{y})^T((x,y)-(\bar{x},\bar{y})) + \tfrac{L_\nabla}{2}\|(x,y)-(\bar{x},\bar{y})\|^2 \ \text{ for all } (x,y),(\bar{x},\bar{y}) \in S\times\mathbb{R}^q \,. \tag{C.1}$$

In the case of AlternativeDirections = FALSE, we have that $x_{k+1} = \bar{x}_k = x_k + \bar{\alpha}_k(\tilde{x}_k - x_k)$. Applying the above inequality to the iterates of Algorithm 1 yields deterministically:

$$\begin{aligned}
F(x_{k+1},y_{k+1}) &\le F(x_k,y_k) + \nabla F(x_k,y_k)^T((x_{k+1},y_{k+1}) - (x_k,y_k)) \\
&\quad + \tfrac{L_\nabla}{2}\|(x_{k+1},y_{k+1}) - (x_k,y_k)\|^2 \\
&= F(x_k,y_k) + \nabla_x F(x_k,y_k)^T(x_{k+1}-x_k) + \tfrac{L_\nabla}{2}\|x_{k+1}-x_k\|_X^2 \\
&\quad + \nabla_y F(x_k,y_k)^T(y_{k+1}-y_k) + \tfrac{L_\nabla}{2}\|y_{k+1}-y_k\|_Y^2 \\
&= F(x_k,y_k) + \bar{\alpha}_k \nabla_x F(x_k,y_k)^T(\tilde{x}_k - x_k) + \tfrac{L_\nabla \bar{\alpha}_k^2}{2}\|\tilde{x}_k - x_k\|_X^2 \\
&\quad - \alpha_k \nabla_y F(x_k,y_k)^T \tilde{y}_k + \tfrac{L_\nabla \alpha_k^2}{2}\|\tilde{y}_k\|_Y^2 \\
&\le F(x_k,y_k) + \bar{\alpha}_k \nabla_x F(x_k,y_k)^T(\tilde{x}_k - x_k) + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\alpha}_k^2}{2} \\
&\quad - \alpha_k \nabla_y F(x_k,y_k)^T \tilde{y}_k + \tfrac{L_\nabla \alpha_k^2}{2} \\
&= F(x_k,y_k) + \bar{\alpha}_k \hat{g}_k^T(\tilde{x}_k - x_k) + \bar{\alpha}_k(\nabla_x F(x_k,y_k) - \hat{g}_k)^T(\tilde{x}_k - x_k) \\
&\quad + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\alpha}_k^2}{2} - \alpha_k \hat{h}_k^T \tilde{y}_k + \alpha_k(\hat{h}_k - \nabla_y F(x_k,y_k))^T \tilde{y}_k + \tfrac{L_\nabla \alpha_k^2}{2} \\
&= F(x_k,y_k) - \bar{\alpha}_k \tilde{G}_k + \bar{\alpha}_k(\nabla_x F(x_k,y_k) - \hat{g}_k)^T(\tilde{x}_k - x_k) + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\alpha}_k^2}{2} \\
&\quad - \alpha_k \|\hat{h}_k\|_{Y*} + \alpha_k(\hat{h}_k - \nabla_y F(x_k,y_k))^T \tilde{y}_k + \tfrac{L_\nabla \alpha_k^2}{2} \,.
\end{aligned}$$

Recall that for any $\gamma > 0$ and vectors $s, x \in \mathbb{R}^p$, it holds that $s^T x \le \frac{1}{2\gamma}\|s\|_{X*}^2 + \frac{\gamma}{2}\|x\|_X^2$. Applying this inequality with $\gamma \leftarrow L_\nabla$, $s \leftarrow \nabla_x F(x_k,y_k) - \hat{g}_k$ and $x \leftarrow \bar{\alpha}_k(\tilde{x}_k - x_k)$ yields:

$$\begin{aligned}
F(x_{k+1},y_{k+1}) &\le F(x_k,y_k) - \bar{\alpha}_k \tilde{G}_k + \tfrac{1}{2L_\nabla}\|\nabla_x F(x_k,y_k) - \hat{g}_k\|_{X*}^2 + \tfrac{L_\nabla \bar{\alpha}_k^2}{2}\|\tilde{x}_k - x_k\|_X^2 \\
&\quad + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\alpha}_k^2}{2} - \alpha_k \|\hat{h}_k\|_{Y*} + \alpha_k(\hat{h}_k - \nabla_y F(x_k,y_k))^T \tilde{y}_k + \tfrac{L_\nabla \alpha_k^2}{2} \\
&\le F(x_k,y_k) - \bar{\alpha}_k \tilde{G}_k + \tfrac{1}{2L_\nabla}\|\nabla_x F(x_k,y_k) - \hat{g}_k\|_{X*}^2 + \tfrac{\bar{C}\bar{\alpha}_k^2}{2} \\
&\quad - \alpha_k \|\hat{h}_k\|_{Y*} + \alpha_k(\hat{h}_k - \nabla_y F(x_k,y_k))^T \tilde{y}_k + \tfrac{L_\nabla \alpha_k^2}{2} \,,
\end{aligned}$$

where the second inequality uses $\bar{C} \ge 2L_\nabla \cdot \mathrm{diam}(S)^2$. Applying the same reasoning on the space of y variables with norms $\|\cdot\|_Y$ and $\|\cdot\|_{Y*}$ yields:

$$\begin{aligned}
F(x_{k+1},y_{k+1}) &\le F(x_k,y_k) - \bar{\alpha}_k \tilde{G}_k + \tfrac{1}{2L_\nabla}\|\nabla_x F(x_k,y_k) - \hat{g}_k\|_{X*}^2 + \tfrac{\bar{C}\bar{\alpha}_k^2}{2} \\
&\quad - \alpha_k \|\hat{h}_k\|_{Y*} + \tfrac{1}{2L_\nabla}\|\nabla_y F(x_k,y_k) - \hat{h}_k\|_{Y*}^2 + \tfrac{L_\nabla \alpha_k^2}{2}\|\tilde{y}_k\|_Y^2 + \tfrac{L_\nabla \alpha_k^2}{2} \\
F(x_{k+1},y_{k+1}) &\le F(x_k,y_k) - \bar{\alpha}_k \tilde{G}_k + \tfrac{1}{2L_\nabla}\|\nabla_x F(x_k,y_k) - \hat{g}_k\|_{X*}^2 + \tfrac{\bar{C}\bar{\alpha}_k^2}{2} \\
&\quad - \alpha_k \|\hat{h}_k\|_{Y*} + \tfrac{1}{2L_\nabla}\|\nabla_y F(x_k,y_k) - \hat{h}_k\|_{Y*}^2 + L_\nabla \alpha_k^2 \,,
\end{aligned}$$

where the second inequality uses $\|\tilde{y}_k\|_Y \leq 1$. Using $\bar{\alpha}_k = \tilde{G}_k/\bar{C}$, and $\alpha_k = \|\hat{h}_k\|_{Y*}/2L_\nabla$ yields:

$$\begin{aligned} F(x_{k+1}, y_{k+1}) \leq F(x_k, y_k) - \frac{\tilde{G}_k^2}{2\bar{C}} - \frac{\|\hat{h}_k\|_{Y*}^2}{4L_\nabla} + \frac{1}{2L_\nabla}\|\nabla_x F(x_k, y_k) - \hat{g}_k\|_{X*}^2 \\ + \frac{1}{2L_\nabla}\|\nabla_y F(x_k, y_k) - \hat{h}_k\|_{Y*}^2 \end{aligned}$$

Multiplying the above inequality by $8L_\nabla$ and rearranging terms yields:

$$\begin{aligned} \Delta_k &\geq \frac{4L_\nabla \tilde{G}_k^2}{\bar{C}} + 2\|\hat{h}_k\|_{Y*}^2 - 4\|\nabla_x F(x_k, y_k) - \hat{g}_k\|_{X*}^2 - 4\|\nabla_y F(x_k, y_k) - \hat{h}_k\|_{Y*}^2 \\ &= \frac{4L_\nabla \tilde{G}_k^2}{\bar{C}} + 2\|\hat{h}_k\|_{Y*}^2 - 4\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \\ &\geq \left(\tilde{G}_k\sqrt{\frac{2L_\nabla}{\bar{C}}} + \|\hat{h}\|_{Y*}\right)^2 - 4\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 , \end{aligned}$$

where the second inequality uses $(a + b)^2 \leq 2(a^2 + b^2)$. By combining assumption (A3) with Lemma C.1.1, we have that

$$\mathbb{E}\left[\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \mid \mathcal{F}_k\right] \leq \frac{\kappa^2\sigma^2}{b_k} .$$

Furthermore, by combining Lemma 2.1 with Jensen's inequality on $t \mapsto t^2$ we have:

$$G(x_k, y_k)^2 \leq \left(\mathbb{E}\left[\tilde{G}_k\sqrt{\tfrac{2L_\nabla}{\bar{C}}} + \|\hat{h}\|_{Y*} \mid \mathcal{F}_k\right]\right)^2 \leq \mathbb{E}\left[\left(\tilde{G}_k\sqrt{\tfrac{2L_\nabla}{\bar{C}}} + \|\hat{h}\|_{Y*}\right)^2 \mid \mathcal{F}_k\right] .$$

Combining the previous inequalities together yields:

$$\mathbb{E}[\Delta_k \mid \mathcal{F}_k] \;\geq\; G(x_k, y_k)^2 - \frac{4\kappa^2\sigma^2}{b_k} ,$$

which proves the result for Case 1.

Case 2: AlternativeDirections = TRUE. First notice that we can decompose Δ_k as:

$$\Delta_k \;=\; 8L_\nabla(F(x_k, y_k) - F(\bar{x}_k, y_{k+1})) \;+\; 8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1})) . \qquad \text{(C.2)}$$

By the exact same reasoning as above, we have that

$$\mathbb{E}[8L_\nabla(F(x_k, y_k) - F(\bar{x}_k, y_{k+1})) \mid \mathcal{F}_k] \;\geq\; G(x_k, y_k)^2 - \frac{4\kappa^2\sigma^2}{b_k} . \qquad \text{(C.3)}$$

Let $\mathcal{G}_k$ denote the σ-field of all information gathered after completing Step (3.) of iteration k of Algorithm 2, i.e., right before starting Step (4.) (the alternative direction step). Note that $\mathcal{F}_k \subset \mathcal{G}_k$. Applying (C.1) at Step (4.) of Algorithm 2, we have deterministically:

$$\begin{aligned}
F(x_{k+1}, y_{k+1}) &\leq F(\bar{x}_k, y_{k+1}) + \nabla F(\bar{x}_k, y_{k+1})^T((x_{k+1}, y_{k+1}) - (\bar{x}_k, y_{k+1})) \\
&\quad + \tfrac{L_\nabla}{2}\|(x_{k+1}, y_{k+1}) - (\bar{x}_k, y_{k+1})\|^2 \\
&= F(\bar{x}_k, y_{k+1}) + \nabla_x F(\bar{x}_k, y_{k+1})^T(x_{k+1} - \bar{x}_k) + \tfrac{L_\nabla}{2}\|x_{k+1} - \bar{x}_k\|_X^2 \\
&= F(\bar{x}_k, y_{k+1}) + \bar{\beta}_k \nabla_x F(\bar{x}_k, y_{k+1})^T d_k + \tfrac{L_\nabla \bar{\beta}_k^2}{2}\|d_k\|_X^2 \\
&\leq F(\bar{x}_k, y_{k+1}) + \bar{\beta}_k \nabla_x F(\bar{x}_k, y_{k+1})^T d_k + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\beta}_k^2}{2} \\
&= F(\bar{x}_k, y_{k+1}) + \bar{\beta}_k \check{g}_k^T d_k + \bar{\beta}_k(\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k)^T d_k + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\beta}_k^2}{2} \\
&= F(\bar{x}_k, y_{k+1}) - \bar{\beta}_k A_k + \bar{\beta}_k(\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k)^T d_k + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\beta}_k^2}{2}
\end{aligned}$$

Applying the inequality $s^T x \leq \frac{1}{2\gamma}\|s\|_{X*}^2 + \frac{\gamma}{2}\|x\|_X^2$ with $\gamma \leftarrow L_\nabla$, $s \leftarrow \nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k$ and $x \leftarrow \bar{\beta}_k d_k$ yields:

$$\begin{aligned}
F(x_{k+1}, y_{k+1}) &\leq F(\bar{x}_k, y_{k+1}) - \bar{\beta}_k A_k + \tfrac{1}{2L_\nabla}\|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 \\
&\quad + \tfrac{L_\nabla \bar{\beta}_k^2}{2}\|d_k\|_X^2 + \tfrac{L_\nabla \mathrm{diam}(S)^2 \bar{\beta}_k^2}{2} \\
&\leq F(\bar{x}_k, y_{k+1}) - \bar{\beta}_k A_k + \tfrac{1}{2L_\nabla}\|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 + \tfrac{\bar{C}\bar{\beta}_k^2}{2} \ ,
\end{aligned}$$

where the second inequality uses $\bar{C} \geq 2L_\nabla \cdot \mathrm{diam}(S)^2$. Notice that $\bar{\beta}_k = \min\left\{A_k/\bar{C}, \alpha_k^{\text{stop}}\right\}$ minimizes the quadratic function $\beta \mapsto -\beta A_k + \frac{\bar{C}\beta^2}{2}$ on the interval $[0, \alpha_k^{\text{stop}}]$. Hence, in particular we have that $-\bar{\beta}_k A_k + \frac{\bar{C}\bar{\beta}_k^2}{2} \leq 0$ and therefore:

$$F(x_{k+1}, y_{k+1}) \ \leq \ F(\bar{x}_k, y_{k+1}) + \tfrac{1}{2L_\nabla}\|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 \ . \tag{C.4}$$

Multiplying the above inequality by $8L_\nabla$ and rearranging terms yields:

$-4\|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 \leq 8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1}))$, and by definition:
$-4\|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 = -4\|(\nabla_x F(\bar{x}_k, y_{k+1}), \nabla_y F(\bar{x}_k, y_{k+1})) - (\check{g}_k, \nabla_y F(\bar{x}_k, y_{k+1}))\|_*^2$.

Using the definition of the dual norm $\|\cdot\|_*$ as well as assumption (A3), we have for all $(x, y) \in S \times \mathbb{R}^q$ that:

$$\begin{aligned}
&\mathbb{E}_{z\sim D}\left[\|(\nabla_x f(x, y, z), \nabla_y F(x, y)) - (\nabla_x F(x, y), \nabla_y F(x, y))\|_*^2\right] = \\
&\mathbb{E}_{z\sim D}\left[\|(\nabla_x f(x, y, z) - \nabla_x F(x, y)\|_{X*}^2 + \|0\|_{Y*}^2\right] \leq \\
&\mathbb{E}_{z\sim D}\left[\|(\nabla_x f(x, y, z) - \nabla_x F(x, y)\|_{X*}^2 + \|\nabla_y f(x, y, z) - \nabla_y F(x, y)\|_{Y*}^2\right] = \\
&\mathbb{E}_{z\sim D}\left[\|\nabla f(x, y, z) - \nabla F(x, y)\|_*^2\right] \leq \sigma^2
\end{aligned}$$

Hence, by combining the above with Lemma C.1.1, we have that

$$\mathbb{E}\left[\|(\nabla_x F(\bar{x}_k, y_{k+1}), \nabla_y F(\bar{x}_k, y_{k+1})) - (\breve{g}_k, \nabla_y F(\bar{x}_k, y_{k+1}))\|_*^2 \mid \mathcal{G}_k\right] \leq \frac{\kappa^2\sigma^2}{b_k} .$$

Combining the previous inequalities together yields:

$$\mathbb{E}[8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1})) \mid \mathcal{G}_k] \;\geq\; -\frac{4\kappa^2\sigma^2}{b_k} .$$

Using the tower property of conditional expectation we have that

$$\begin{aligned}\mathbb{E}[8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1})) \mid \mathcal{F}_k] \;&=\; \mathbb{E}\left[\mathbb{E}[8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1})) \mid \mathcal{G}_k] \mid \mathcal{F}_k\right] \\ &\geq\; -\frac{4\kappa^2\sigma^2}{b_k} .\end{aligned}$$

Finally combining the above with (C.3) and and (C.2) yields:

$$\mathbb{E}[\Delta_k \mid \mathcal{F}_k] \;\geq\; G(x_k, y_k)^2 - \frac{8\kappa^2\sigma^2}{b_k} ,$$

which proves the result in Case 2. □

Proof of Theorem 4.1.1

By combining Lemma C.1.2 with the law of iterated expectations, it holds for each $k \in \{0, \ldots, K\}$ that:

$$\mathbb{E}[\Delta_k] \;=\; \mathbb{E}\left[\mathbb{E}[\Delta_k \mid \mathcal{F}_k]\right] \;\geq\; \mathbb{E}[G(x_k, y_k)^2] - \frac{4\alpha_{\mathrm{AD}}\kappa^2\sigma^2}{b_k} .$$

Recalling that $\mathbb{E}[\Delta_k] = 8L_\nabla\mathbb{E}[F(x_k, y_k)] - 8L_\nabla\mathbb{E}[F(x_{k+1}, y_{k+1})]$ and summing the above inequality over all $k \in \{0, \ldots, K\}$ yields:

$$\sum_{k=0}^{K}\mathbb{E}[G(x_k, y_k)^2] \;\leq\; 8L_\nabla(F(x_0, y_0) - \mathbb{E}[F(x_{K+1}, y_{K+1})]) \;+\; 4\alpha_{\mathrm{AD}}\kappa^2\sigma^2\sum_{k=0}^{K}\frac{1}{b_k} .$$

Then, using $F^* \leq \mathbb{E}[F(x_{K+1}, y_{K+1})]$ and dividing by $K+1$ yields:

$$\frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}[G(x_k, y_k)^2] \;\leq\; \frac{8L_\nabla(F(x_0, y_0) - F^*)}{K+1} \;+\; \frac{4\alpha_{\mathrm{AD}}\kappa^2\sigma^2}{K+1}\sum_{k=0}^{K}\frac{1}{b_k} .$$

Finally, since $(\hat{x}_k, \hat{y}_k)$ is chosen uniformly at random from $(x_0, y_0), \ldots, (x_K, y_K)$, another iterated expectations argument implies that $\mathbb{E}[G(\hat{x}_K, \hat{y}_K)^2] = \frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}[G(x_k, y_k)^2]$, from which the desired result follows. □

C.2 Example of In-Face Direction Computation

In this section, we briefly describe how to compute an in-face direction in the case where $S = \{x : \|x\|_1 \leq \delta\}$ is an ℓ_1-ball. Let $\bar{x} \in S$ be a given point representing our current iterate. In particular, let us discuss the complexity of a solving a linear optimization problem $\min_{x \in \mathcal{F}(\bar{x})} c^T x$ over the minimal face $\mathcal{F}(\bar{x})$ containing $\bar{x}$ for some given $c \in \mathbb{R}^p$, which is required in the "away step" direction (3), for example.

Let us consider two cases: *(i)* $\bar{x} \in \operatorname{int}(S)$ and *(ii)* $\bar{x} \in \partial S$, where ∂S represents the boundary of S. In case *(i)*, we simply have that $\mathcal{F}(\bar{x}) = S$ and the linear optimization problem is simply that of minimizing $c^T x$ over S, which is the same subproblem as the Frank-Wolfe step as is equivalent to computing $\|c\|_\infty = \max_{j=1,\ldots,p} |c_j|$. Otherwise, if $\bar{x} \in \partial S$, then we have that $\|\bar{x}\|_1 = \delta$ and let $J_+(\bar{x}) = \{j : \bar{x}_j > 0\}$, $J_-(\bar{x}) = \{j : \bar{x}_j < 0\}$, $J_0(\bar{x}) = \{j : \bar{x}_j = 0\}$. Then, it is straightforward to see that

$$\begin{aligned}\mathcal{F}(\bar{x}) = \{x :\ & x_j > 0 \text{ if } j \in J_+(\bar{x}), \\ & x_j < 0 \text{ if } j \in J_-(\bar{x}), \\ & x_j = 0 \text{ if } j \in J_0(\bar{x}), \\ & \sum_{j \in J_+(\bar{x})} x_j - \sum_{j \in J_-(\bar{x})} x_j = \delta\} \ .\end{aligned}$$

(Note that we clearly have $\|x\|_1 = \sum_{j \in J_+(\bar{x})} x_j - \sum_{j \in J_-(\bar{x})} x_j$ in the above.) Then, in order to solve $\min_{x \in \mathcal{F}(\bar{x})} c^T x$, we can simply follow an argument that enumerates the extreme points of the above polytope, from which we obtain that:

$$j^* \in \operatorname*{arg\,min}_{j \in J_+(\bar{x}) \cup J_-(\bar{x})} \operatorname{sgn}(\bar{x}_j) c_j \implies \operatorname{sgn}(\bar{x}_{j^*}) \delta e_{j^*} \in \arg \min_{x \in \mathcal{F}(\bar{x})} c^T x \ .$$

Thus, as in the case when $\bar{x} \in \operatorname{int}(S)$, we can solve $\min_{x \in \mathcal{F}(\bar{x})} c^T x$ efficiently in time that is linear in p.

C.3 Proofs in Section 4.2

Let us first state and prove the following lemma, which is the "block coordinate" version of Lemma C.1.2 and will be critical proving Theorem 3.1.

Lemma C.3.1. *For each $k \geq 0$, let $\mathcal{F}_k$ denote the σ-field of all information gathered after completing iteration $k-1$ of the Block Coordinate variant of Algorithm 1, i.e., right before starting iteration k, and define $\Delta_k := 8L_\nabla(F(x_k, y_k) - F(x_{k+1}, y_{k+1}))$. Then, at every iteration $k \geq 0$, it holds that:*

$$\mathbb{E}[\Delta_k \mid \mathcal{F}_k] \ \geq \ G(x_k, y_k)^2 - \frac{4\alpha_{\mathrm{AD}} \kappa^2 \sigma^2}{b_k} \ .$$

where the modified Frank-Wolfe gap $G(\cdot, \cdot)$ (Definition 2.1) is defined using $\bar{C} := \sum_{i=1}^N \bar{C}_i$.

Proof. **Case 1: AlternativeDirections = FALSE.** Let us define $\Theta_k := 8L_\triangledown(F(x_k, y_k) - F(\bar{x}_k, y_{k+1}))$. We first bound Θ_k following the same general structure as in the proof of Lemma C.1.2 in Section C.1.

Applying (C.1) at Steps (2.)/(3.) of the block coordinate version of Algorithm 2, we have deterministically:

$$\begin{aligned} F(\bar{x}_k, y_{k+1}) &\le F(x_k, y_k) + \nabla F(x_k, y_k)^T((\bar{x}_k, y_{k+1}) - (x_k, y_k)) + \tfrac{L_\triangledown}{2}\|(\bar{x}_k, y_{k+1}) - (x_k, y_k)\|^2 \\ &= F(x_k, y_k) + \nabla_x F(x_k, y_k)^T(\bar{x}_k - x_k) + \tfrac{L_\triangledown}{2}\|\bar{x}_k - x_k\|_X^2 \\ &\qquad + \nabla_y F(x_k, y_k)^T(y_{k+1} - y_k) + \tfrac{L_\triangledown}{2}\|y_{k+1} - y_k\|_Y^2 \end{aligned} \tag{C.5}$$

For ease of notation, define $\Gamma_k := \nabla_x F(x_k, y_k)^T(\bar{x}_k - x_k) + \frac{L_\triangledown}{2}\|\bar{x}_k - x_k\|_X^2$. Utilizing the block coordinate structure, we have that:

$$\begin{aligned} \Gamma_k &= \nabla_x F(x_k, y_k)^T(\bar{x}_k - x_k) + \tfrac{L_\triangledown}{2}\|\bar{x}_k - x_k\|_X^2 \\ &= \sum_{i=1}^N \nabla_x^{(i)} F(x_k, y_k)^T(\bar{x}_k^{(i)} - x_k^{(i)}) \ + \ \tfrac{L_\triangledown}{2}\sum_{i=1}^N \|\bar{x}_k^{(i)} - x_k^{(i)}\|_{X,i}^2 \\ &= \sum_{i=1}^N \left[\nabla_x^{(i)} F(x_k, y_k)^T(\bar{x}_k^{(i)} - x_k^{(i)}) + \tfrac{L_\triangledown}{2}\|\bar{x}_k^{(i)} - x_k^{(i)}\|_{X,i}^2\right] \\ &= \sum_{i=1}^N \left[\bar{\alpha}_k^i \nabla_x^{(i)} F(x_k, y_k)^T(\tilde{x}_k^{(i)} - x_k^{(i)}) + \tfrac{L_\triangledown(\bar{\alpha}_k^i)^2}{2}\|\tilde{x}_k^{(i)} - x_k^{(i)}\|_{X,i}^2\right] \\ &\le \sum_{i=1}^N \left[\bar{\alpha}_k^i \nabla_x^{(i)} F(x_k, y_k)^T(\tilde{x}_k^{(i)} - x_k^{(i)}) + \tfrac{L_\triangledown \mathrm{diam}(S_i)^2(\bar{\alpha}_k^i)^2}{2}\right] \\ &= \sum_{i=1}^N \left[\bar{\alpha}_k^i (\hat{g}_k^{(i)})^T(\tilde{x}_k^{(i)} - x_k^{(i)}) + \bar{\alpha}_k^i(\nabla_x^{(i)} F(x_k, y_k) - \hat{g}_k^{(i)})^T(\tilde{x}_k^{(i)} - x_k^{(i)}) + \tfrac{L_\triangledown \mathrm{diam}(S_i)^2(\bar{\alpha}_k^i)^2}{2}\right] \\ &= \sum_{i=1}^N \left[-\bar{\alpha}_k^i \tilde{G}_k^i + \bar{\alpha}_k^i(\nabla_x^{(i)} F(x_k, y_k) - \hat{g}_k^{(i)})^T(\tilde{x}_k^{(i)} - x_k^{(i)}) + \tfrac{L_\triangledown \mathrm{diam}(S_i)^2(\bar{\alpha}_k^i)^2}{2}\right] \\ &\le \sum_{i=1}^N \left[-\bar{\alpha}_k^i \tilde{G}_k^i + \tfrac{1}{2L_\triangledown}\|\nabla_x^{(i)} F(x_k, y_k) - \hat{g}_k^{(i)}\|_{X*,i}^2 + \tfrac{L_\triangledown(\bar{\alpha}_k^i)^2}{2}\|\tilde{x}_k^{(i)} - x_k^{(i)}\|_{X,i}^2 + \tfrac{L_\triangledown \mathrm{diam}(S_i)^2(\bar{\alpha}_k^i)^2}{2}\right] \\ &\le \sum_{i=1}^N \left[-\bar{\alpha}_k^i \tilde{G}_k^i + \tfrac{1}{2L_\triangledown}\|\nabla_x^{(i)} F(x_k, y_k) - \hat{g}_k^{(i)}\|_{X*,i}^2 + \tfrac{\bar{C}_i(\bar{\alpha}_k^i)^2}{2}\right] \\ &= \sum_{i=1}^N \left[-\bar{\alpha}_k^i \tilde{G}_k^i + \tfrac{\bar{C}_i(\bar{\alpha}_k^i)^2}{2}\right] \ + \ + \tfrac{1}{2L_\triangledown}\|\nabla_x F(x_k, y_k) - \hat{g}_k\|_{X*}^2 \ . \end{aligned}$$

The second as well as the final equality above uses the Euclidean structure of $\|\cdot\|_X$ and $\|\cdot\|_{X*}$, namely $\|x\|_X^2 := \sum_{i=1}^N \|x^{(i)}\|_{X,i}^2$ and $\|s\|_{X*}^2 := \sum_{i=1}^N \|s^{(i)}\|_{X*,i}^2$. The second inequality uses $(s^{(i)})^T x^{(i)} \leq \frac{1}{2\gamma}\|s^{(i)}\|_{X*,i}^2 + \frac{\gamma}{2}\|x^{(i)}\|_{X,i}^2$ with $\gamma \leftarrow L_\nabla$, $s \leftarrow \nabla_x^{(i)} F(x_k, y_k) - \hat{g}_k^{(i)}$ and $x \leftarrow \bar{\alpha}_k^i(\tilde{x}_k^{(i)} - x_k^{(i)})$ for each $i \in \{1, \ldots, N\}$, and the third inequality uses $\bar{C}_i \geq 2L_\nabla \mathrm{diam}(S_i)^2$. Now, combining (C.5) with the above as well as the same reasoning on the space of y variables that was used in the proof of Lemma 2.2 yields:

$$\begin{aligned} F(\bar{x}_k, y_{k+1}) \leq F(x_k, y_k) &+ \sum_{i=1}^N \left[-\bar{\alpha}_k^i \tilde{G}_k^i + \tfrac{\bar{C}_i(\bar{\alpha}_k^i)^2}{2}\right] + \tfrac{1}{2L_\nabla}\|\nabla_x F(x_k, y_k) - \hat{g}_k\|_{X*}^2 \\ &- \alpha_k \|\hat{h}_k\|_{Y*} + \tfrac{1}{2L_\nabla}\|\nabla_y F(x_k, y_k) - \hat{h}_k\|_{Y*}^2 + L_\nabla \alpha_k^2 \ . \end{aligned}$$

Using $\bar{\alpha}_k^i = \tilde{G}_k^i / \bar{C}_i$, and $\alpha_k = \|\hat{h}_k\|_{Y*}/2L_\nabla$ yields:

$$\begin{aligned} F(\bar{x}_k, y_{k+1}) &\leq F(x_k, y_k) \\ &\quad - \sum_{i=1}^N \frac{(\tilde{G}_k^i)^2}{2\bar{C}_i} - \frac{\|\hat{h}_k\|_{Y*}^2}{4L_\nabla} + \frac{1}{2L_\nabla}\left(\|\nabla_x F(x_k, y_k) - \hat{g}_k\|_{X*}^2 + \|\nabla_y F(x_k, y_k) - \hat{h}_k\|_{Y*}^2\right) \\ &= F(x_k, y_k) \\ &\quad - \sum_{i=1}^N \frac{(\tilde{G}_k^i)^2}{2\bar{C}_i} - \frac{\|\hat{h}_k\|_{Y*}^2}{4L_\nabla} + \frac{1}{2L_\nabla}\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \ . \end{aligned}$$

Letting $\Theta_k := 8L_\nabla(F(x_k, y_k) - F(\bar{x}_k, y_{k+1}))$ and multiplying the above inequality by $8L_\nabla$ and rearranging terms yields:

$$\Theta_k \geq 4L_\nabla \sum_{i=1}^N \frac{(\tilde{G}_k^i)^2}{\bar{C}_i} + 2\|\hat{h}_k\|_{Y*}^2 - 4\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \ . \tag{C.6}$$

Recall that for any two sequences $\{g_i\}_{i=1}^N$ and $\{c_i\}_{i=1}^N$ with $g_i \geq 0$ and $c_i > 0$, Cauchy-Schwartz yields:

$$\left(\sum_{i=1}^N g_i\right)^2 = \left(\sum_{i=1}^N \frac{g_i\sqrt{c_i}}{\sqrt{c_i}}\right)^2 \leq \left(\sum_{i=1}^N \frac{g_i^2}{c_i}\right)\left(\sum_{i=1}^N c_i\right) \ .$$

Recall that $\bar{C} = \sum_{i=1}^N \bar{C}_i$ and let us define $\tilde{G}_k := \sum_{i=1}^N \tilde{G}_k^i$. Then, applying the above to (C.6) with $g_i \leftarrow \tilde{G}_k^i$ and $c_i \leftarrow \bar{C}_i$ yields:

$$\begin{aligned} \Theta_k &\geq \frac{4L_\nabla \tilde{G}_k^2}{\bar{C}} + 2\|\hat{h}_k\|_{Y*}^2 - 4\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \\ &\geq \left(\tilde{G}_k\sqrt{\frac{2L_\nabla}{\bar{C}}} + \|\hat{h}_k\|_{Y*}\right)^2 - 4\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \ , \end{aligned}$$

where the second inequality uses $(a+b)^2 \leq 2(a^2+b^2)$. By combining assumption (A3) with Lemma C.1.1, we have that

$$\mathbb{E}\left[\|(\nabla_x F(x_k, y_k), \nabla_y F(x_k, y_k)) - (\hat{g}_k, \hat{h}_k)\|_*^2 \mid \mathcal{F}_k\right] \leq \frac{\kappa^2\sigma^2}{b_k} \ .$$

Furthermore, note that the decomposable structure of S implies that $\tilde{G}_k = \max_{x \in S} \left\{ \hat{g}_k^T(\bar{x} - x) \right\}$. Therefore, we may apply Lemma 2.1 along with Jensen's inequality on $t \mapsto t^2$ to yield:

$$G(x_k, y_k)^2 \leq \left(\mathbb{E} \left[\tilde{G}_k \sqrt{\tfrac{2L_\nabla}{\bar{C}}} + \|\hat{h}\|_{Y*} \mid \mathcal{F}_k \right] \right)^2 \leq \mathbb{E} \left[\left(\tilde{G}_k \sqrt{\tfrac{2L_\nabla}{\bar{C}}} + \|\hat{h}\|_{Y*} \right)^2 \mid \mathcal{F}_k \right] .$$

Combining the previous inequalities together yields:

$$\mathbb{E}[\Theta_k \mid \mathcal{F}_k] \ \geq \ G(x_k, y_k)^2 - \frac{4\kappa^2 \sigma^2}{b_k} \ ,$$

which proves the result in Case 1 since $x_{k+1} = \bar{x}_k$ in that case.

Case 2: AlternativeDirections = TRUE. Now notice that we can decompose Δ_k as:

$$\Delta_k \ = \ 8L_\nabla(F(x_k, y_k) - F(\bar{x}_k, y_{k+1})) \ + \ 8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1})) \ . \qquad \text{(C.7)}$$

Let us define $\Lambda_k := 8L_\nabla(F(\bar{x}_k, y_{k+1}) - F(x_{k+1}, y_{k+1}))$ so that $\Delta_k = \Theta_k + \Lambda_k$.

Let us now work on bounding Λ_k using the same general structure as that of Case 2 of Lemma C.1.2 in Section C.1. Let $\mathcal{G}_k$ denote the σ-field of all information gathered after completing Step (3.) of iteration k of the block coordinate version of Algorithm 2, i.e., right before starting Step (4.) (the alternative direction step). Note that $\mathcal{F}_k \subset \mathcal{G}_k$. Applying (C.1) at Step (4.), we have deterministically:

$$\begin{aligned}
F(x_{k+1}, y_{k+1}) &\leq F(\bar{x}_k, y_{k+1}) + \nabla F(\bar{x}_k, y_{k+1})^T((x_{k+1}, y_{k+1}) - (\bar{x}_k, y_{k+1})) \\
&\quad + \tfrac{L_\nabla}{2}\|(x_{k+1}, y_{k+1}) - (\bar{x}_k, y_{k+1})\|^2 \\
&= F(\bar{x}_k, y_{k+1}) + \nabla_x F(\bar{x}_k, y_{k+1})^T(x_{k+1} - \bar{x}_k) + \tfrac{L_\nabla}{2}\|x_{k+1} - \bar{x}_k\|_X^2 \\
&= F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^N \left[\nabla_x^{(i)} F(\bar{x}_k, y_{k+1})^T(x_{k+1}^{(i)} - \bar{x}_k^{(i)}) + \tfrac{L_\nabla}{2}\|x_{k+1}^{(i)} - \bar{x}_k^{(i)}\|_{X,i}^2 \right] \\
&= F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^N \left[\bar{\beta}_k^i \nabla_x^{(i)} F(\bar{x}_k, y_{k+1})^T d_k^{(i)} + \tfrac{L_\nabla (\bar{\beta}_k^i)^2}{2}\|d_k^{(i)}\|_{X,i}^2 \right] \\
&\leq F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^N \left[\bar{\beta}_k^i \nabla_x^{(i)} F(\bar{x}_k, y_{k+1})^T d_k^{(i)} + \tfrac{L_\nabla \text{diam}(S_i)^2 (\bar{\beta}_k^i)^2}{2} \right] \\
&= F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^N \left[\bar{\beta}_k^i (\breve{g}_k^{(i)})^T d_k^{(i)} + \bar{\beta}_k^i (\nabla_x^{(i)} F(\bar{x}_k, y_{k+1}) - \breve{g}_k^{(i)})^T d_k^{(i)} + \tfrac{L_\nabla \text{diam}(S_i)^2 (\bar{\beta}_k^i)^2}{2} \right] \\
&= F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^N \left[-\bar{\beta}_k^i A_k^i + \bar{\beta}_k^i (\nabla_x^{(i)} F(\bar{x}_k, y_{k+1}) - \breve{g}_k^{(i)})^T d_k^{(i)} + \tfrac{L_\nabla \text{diam}(S_i)^2 (\bar{\beta}_k^i)^2}{2} \right]
\end{aligned}$$

Applying the inequality $(s^{(i)})^T x^{(i)} \ \leq \ \frac{1}{2\gamma}\|s^{(i)}\|_{X*,i}^2 + \frac{\gamma}{2}\|x^{(i)}\|_{X,i}^2$ with $\gamma \leftarrow L_\nabla$, $s \leftarrow \nabla_x^{(i)} F(\bar{x}_k, y_{k+1}) - \breve{g}_k^{(i)}$ and $x \leftarrow \bar{\beta}_k^i d_k^{(i)}$ yields:

$$\begin{aligned}
F(x_{k+1}, y_{k+1}) &\le F(\bar{x}_k, y_{k+1}) \\
&\quad + \sum_{i=1}^{N} \left[-\bar{\beta}_k^i A_k^i + \tfrac{1}{2L_\nabla} \|\nabla_x^{(i)} F(\bar{x}_k, y_{k+1}) - \check{g}_k^{(i)}\|_{X*,i}^2 + \tfrac{L_\nabla (\bar{\beta}_k^i)^2}{2} \|d_k^{(i)}\|_{X,i}^2 + \tfrac{L_\nabla \mathrm{diam}(S_i)^2 (\bar{\beta}_k^i)^2}{2} \right] \\
&\le F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^{N} \left[-\bar{\beta}_k^i A_k^i + \tfrac{1}{2L_\nabla} \|\nabla_x^{(i)} F(\bar{x}_k, y_{k+1}) - \check{g}_k^{(i)}\|_{X*,i}^2 + \tfrac{\bar{C}_i (\bar{\beta}_k^i)^2}{2} \right] \\
&= F(\bar{x}_k, y_{k+1}) + \sum_{i=1}^{N} \left[-\bar{\beta}_k^i A_k^i + \tfrac{\bar{C}_i (\bar{\beta}_k^i)^2}{2} \right] \ + \tfrac{1}{2L_\nabla} \|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 \ ,
\end{aligned}$$

where the second inequality uses $\|d_k^{(i)}\|_{X,i} \le \mathrm{diam}(S_i)$ and $\bar{C}_i \ge 2L_\nabla \cdot \mathrm{diam}(S_i)^2$.

Notice that $\bar{\beta}_k^i = \min\left\{A_k^i/\bar{C}_i, \alpha_k^{\mathrm{stop},i}\right\}$ minimizes the quadratic function $\beta \mapsto -\beta A_k^i + \frac{\bar{C}_i \beta^2}{2}$ on the interval $[0, \alpha_k^{\mathrm{stop},i}]$. Hence, in particular we have that $-\bar{\beta}_k^i A_k^i + \frac{\bar{C}_i (\bar{\beta}_k^i)^2}{2} \le 0$ and therefore:

$$F(x_{k+1}, y_{k+1}) \ \le \ F(\bar{x}_k, y_{k+1}) + \tfrac{1}{2L_\nabla} \|\nabla_x F(\bar{x}_k, y_{k+1}) - \check{g}_k\|_{X*}^2 \ .$$

The remainder of the proof is now exactly the same as that of Lemma C.1.2 starting at (C.4). □

Proof of Theorem 4.2.1

Given Lemma C.3.1, the proof of Theorem 3.1 follows the exact same logic as that of Theorem 2.1 in Section C.1.

C.4 Additional Numerical Results on Synthetic Data

In this set of experiments, we generated artificial data from a model that is described by an artificially generated sparse network. In particular, the network is composed of three layers of sizes 50×50, 50×50 and 50×1, and the activation functions are either ReLU or sigmoid (notice that sigmoid is smooth, but ReLU is not). We did not add bias terms for this experiment. For the first two layers of the true network, we randomly generated m edges going into each node where $m \in \{5, 10, 15\}$ and the weight on each randomly sampled edge is either -1 or 1 with equal probability. All edges that are not selected have their weight equal to 0, and the last layer is fully connected. We treated this as a regression problem with mean squared error loss, the feature matrix is composed of elements that are sampled i.i.d. from a standard Gaussian distribution, and the dependent variable values y are chosen in a way to control the signal to noise ratio (SNR) in the set $SNR \in \{1, 5, 10\}$. Our train, validation, and test sets are composed of 100,000, 20,000, and 100,000 data points, respectively. For SFW and SFW-IF, the first two layers are treated as Frank-Wolfe layers and the last layer is fully dense and treated as an SGD layer.

For each combination of $m \in \{5, 10, 15\}$ and $SNR \in \{1, 5, 10\}$, we ran 30 trials over randomly generated true networks and datasets as described above. Figures 1 and 2 show box plots over these 30 trials, with three performance metrics of interest: average number of non-zero edges (here average number of non-zeros per edge can be at most 50) going into each node in layers 1 and 2 (every value below 0.001 is considered to be 0), and the test set mean squared error. All Figures show that SFW and SFW-IF recover solutions that are sparser than SGD, which does not not promote this behaviour. SFW-IF is also able to consistently recover a solution that is sparser than SFW due to the incorporation of in-face directions (except for layer 2 using ReLU). Interestingly, using the sigmoid activation, the gap between the sparsity of SFW-IF and SFW is larger for layer 2 than layer 1; however, using the ReLU activation, this pattern is reversed. Also, SFW-IF and SFW have comparable test set MSE to SGD, and for the sigmoid experiment they even have lower test MSE for the case of $m \in \{10, 15\}$. Figures 3 and 4 also consider a single instance with $m = 10$ and $SNR = 10$, using the sigmoid activiation, and display the evolution of the per layer average non-zeros and also the modified Frank-Wolfe gap. (Since we are only able to compute a stochastic estimate of the modified Frank-Wofle gap, we also display a smoothed version of this plot.) Notice that, throughout all iterations, SFW-IF consistently maintains a sparser solution than SFW. Interestingly, it appears that SFW-IF is also able to reduce the modified Frank-Wolfe gap faster than SFW.

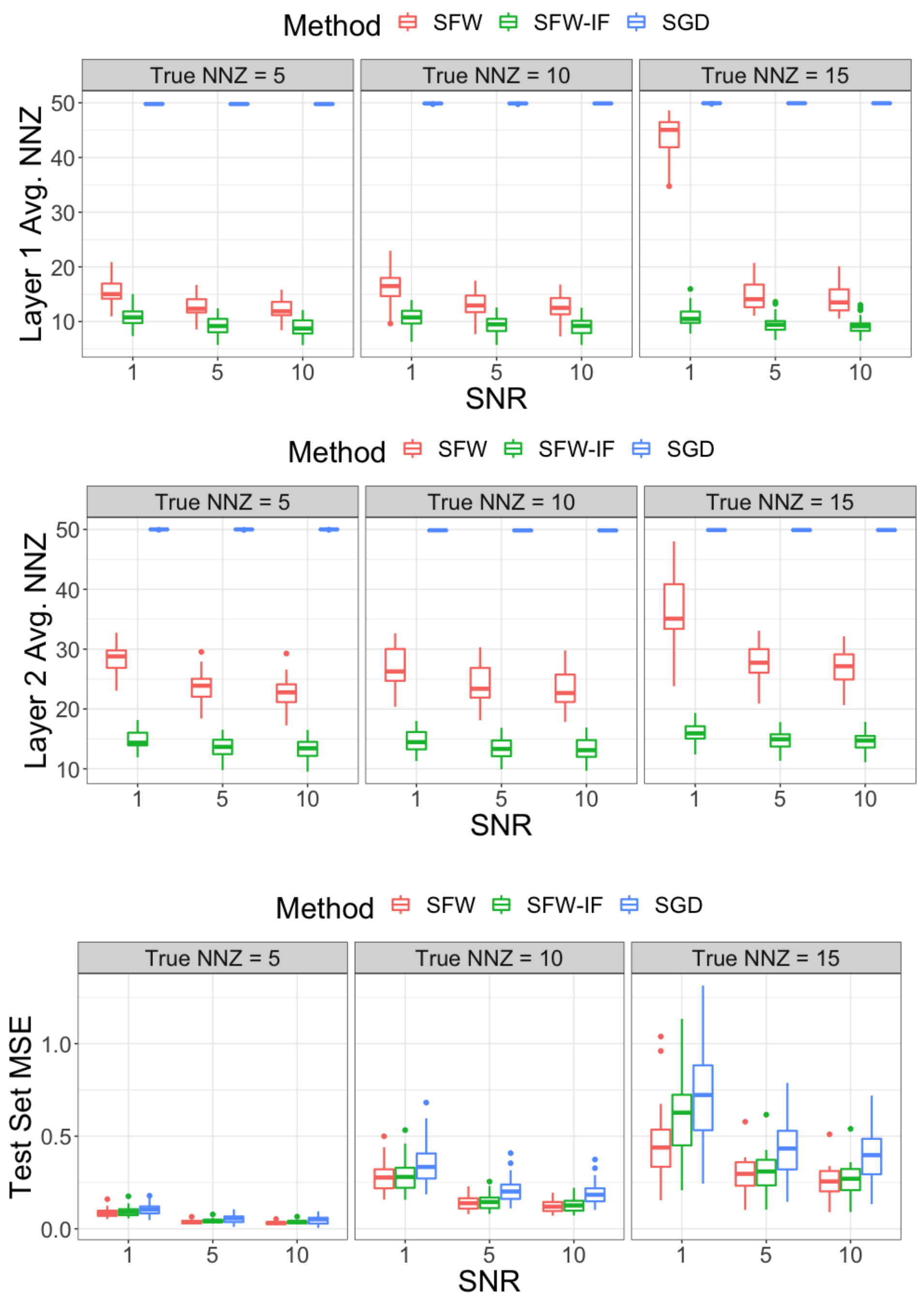


Figure C.1: Results using sigmoid activation function

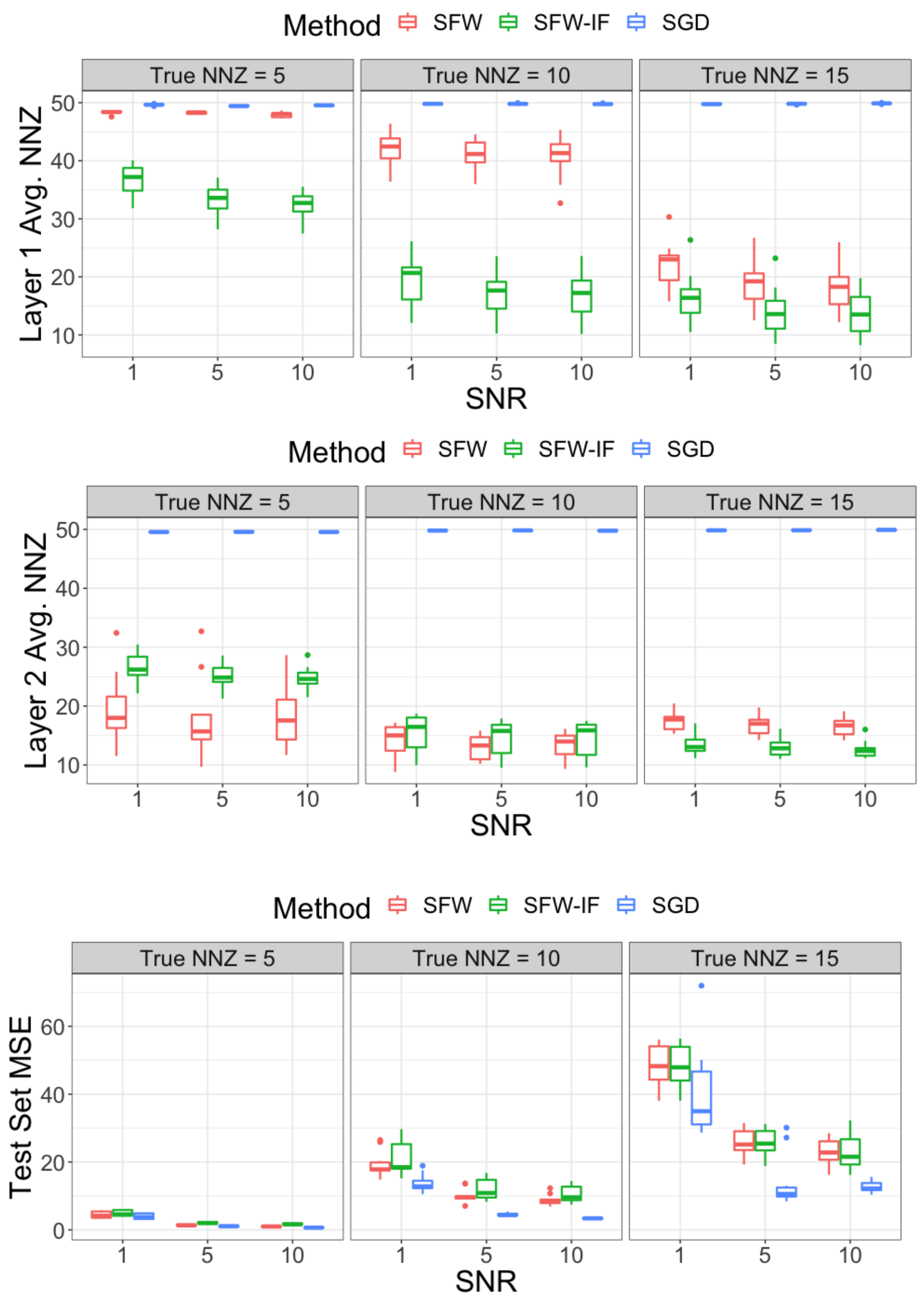
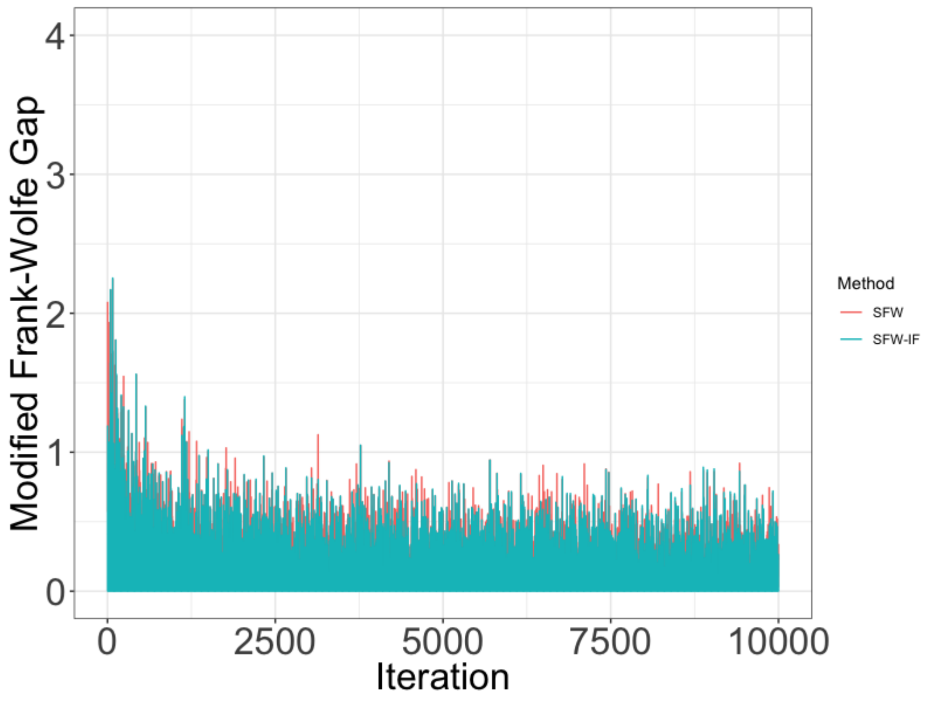


Figure C.2: Results using ReLU activiation function



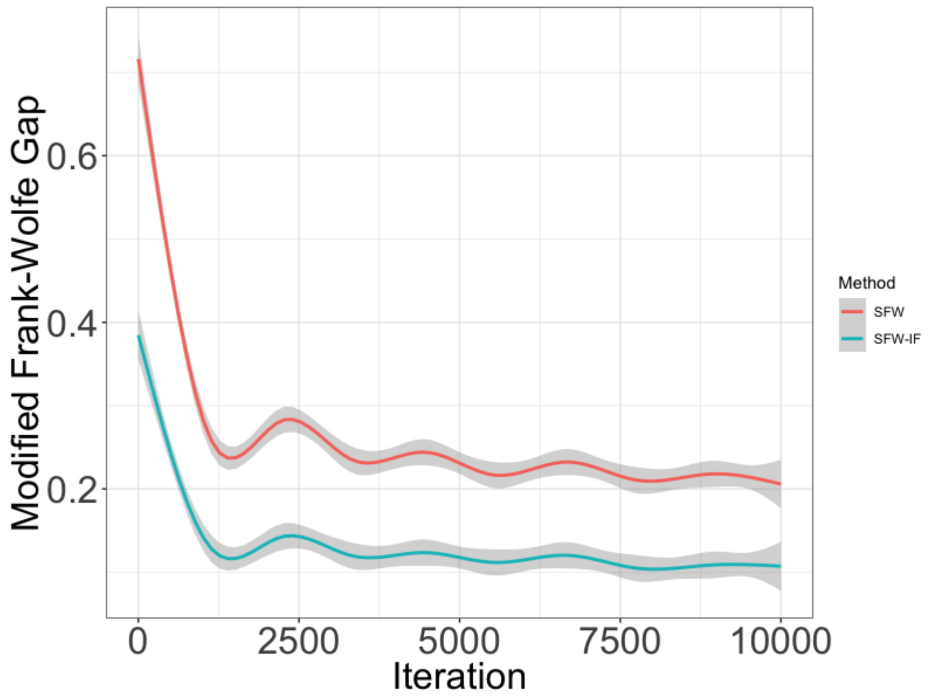
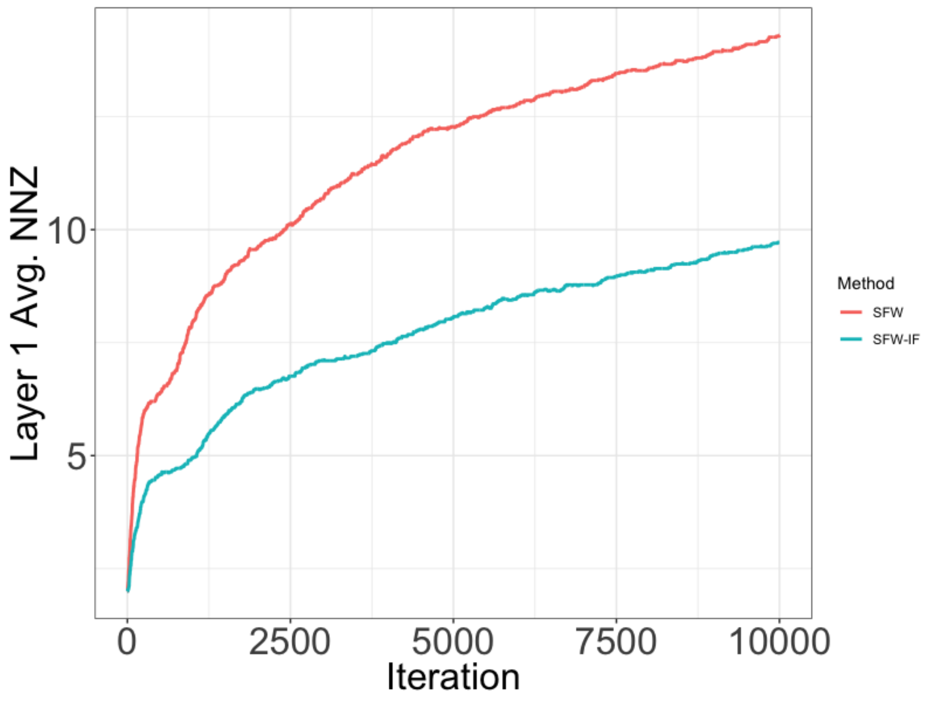


Figure C.3: Modified Frank-Wolfe gap vs. iterations for SFW and SFW-IF, on an instance from a network generated using the sigmoid activation function with $m = 10$ non-zeros and $SNR = 10$.



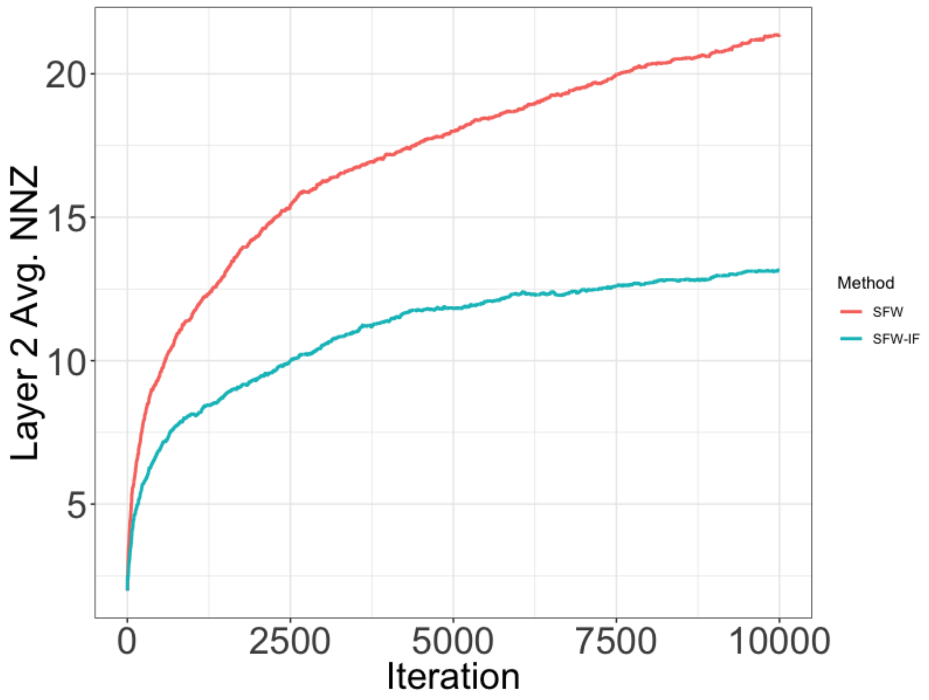


Figure C.4: Average number of non-zeros (NNZ) per layer for SFW and SFW-IF, on an instance from a network generated using the sigmoid activation function with $m = 10$ non-zeros and $SNR = 10$.

Bibliography

[1] Alireza Aghasi et al. “Net-trim: Convex pruning of deep neural networks with performance guarantee”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 3177–3186.

[2] Shipra Agrawal and Nikhil Devanur. “Linear contextual bandits with knapsacks”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 3450–3458.

[3] Shipra Agrawal and Nikhil R Devanur. “Bandits with concave rewards and convex knapsacks”. In: *Proceedings of the fifteenth ACM conference on Economics and computation*. 2014, pp. 989–1006.

[4] Shipra Agrawal and Navin Goyal. “Thompson sampling for contextual bandits with linear payoffs”. In: *International Conference on Machine Learning*. 2013, pp. 127–135.

[5] Sanjeev Arora, Elad Hazan, and Satyen Kale. “The multiplicative weights update method: a meta-algorithm and applications”. In: *Theory of Computing* 8.1 (2012), pp. 121–164.

[6] Peter Auer and Ronald Ortner. “UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem”. In: *Periodica Mathematica Hungarica* 61.1-2 (2010), pp. 55–65.

[7] Avazu. *Click-Through Rate Prediction: Predict whether a mobile ad will be clicked. (Contest on Kaggle.)* `https://www.kaggle.com/c/avazu-ctr-prediction`. Nov. 2014.

[8] Lennart Baardman et al. “Learning Optimal Online Advertising Portfolios with Periodic Budgets”. In: *Available at SSRN 3346642* (2019).

[9] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. “Bandits with knapsacks”. In: *Journal of the ACM (JACM)* 65.3 (2018), pp. 1–55.

[10] Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. “Dual Mirror Descent for Online Allocation Problems”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 613–628.

[11] Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. “Regularized Online Allocation Problems: Fairness and Beyond”. In: *arXiv preprint arXiv:2007.00514* (2020).

[12] Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. “The Best of Many Worlds: Dual Mirror Descent for Online Allocation Problems”. In: *arXiv preprint arXiv:2011.10124* (2020).

[13] Santiago R Balseiro, Omar Besbes, and Gabriel Y Weintraub. “Repeated auctions with budgets in ad exchanges: Approximations and design”. In: *Management Science* 61.4 (2015), pp. 864–884.

[14] Santiago R Balseiro and Yonatan Gur. “Learning in repeated auctions with budgets: Regret minimization and equilibrium”. In: *Management Science* 65.9 (2019), pp. 3952–3968.

[15] Santiago R Balseiro et al. “Yield optimization of display advertising with ad exchange”. In: *Management Science* 60.12 (2014), pp. 2886–2907.

[16] Mohammad Hossein Bateni et al. “Fair resource allocation in a volatile marketplace”. In: *Proceedings of the 2016 ACM Conference on Economics and Computation*. 2016, pp. 819–819.

[17] Amir Beck. *First-order methods in optimization*. SIAM, 2017.

[18] Amir Beck and Marc Teboulle. “Mirror descent and nonlinear projected subgradient methods for convex optimization”. In: *Operations Research Letters* 31.3 (2003), pp. 167–175.

[19] Dimitri P Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.

[20] Dimitris Bertsimas and Ioana Popescu. “Revenue management in a dynamic network environment”. In: *Transportation science* 37.3 (2003), pp. 257–277.

[21] Léon Bottou. “Large-scale machine learning with stochastic gradient descent”. In: *Proceedings of COMPSTAT’2010*. Springer, 2010, pp. 177–186.

[22] Léon Bottou, Frank E Curtis, and Jorge Nocedal. “Optimization methods for large-scale machine learning”. In: *Siam Review* 60.2 (2018), pp. 223–311.

[23] E Andrew Boyd and Ioana C Bilegan. “Revenue management and e-commerce”. In: *Management science* 49.10 (2003), pp. 1363–1386.

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

[25] Leo Breiman et al. *Classification and regression trees*. CRC press, 1984.

[26] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. “An analysis of deep neural network models for practical applications”. In: *arXiv preprint arXiv:1605.07678* (2016).

[27] Olivier Chapelle, Eren Manavoglu, and Romer Rosales. “Simple and scalable response prediction for display advertising”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 5.4 (2015), p. 61.

[28] Bowei Chen, Shuai Yuan, and Jun Wang. "A dynamic pricing model for unifying programmatic guarantee and real-time bidding in display advertising". In: *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. 2014, pp. 1–9.

[29] Tianyi Chen, Qing Ling, and Georgios B Giannakis. "An online convex optimization approach to proactive network resource allocation". In: *IEEE Transactions on Signal Processing* 65.24 (2017), pp. 6350–6364.

[30] Ye Chen et al. "Real-time bidding algorithms for performance-based display ad allocation". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2011, pp. 1307–1315.

[31] Hana Choi et al. "Online display advertising markets: A literature review and future directions". In: *Information Systems Research* (2020).

[32] Dragos Florin Ciocan and Vivek Farias. "Model predictive control for dynamic resource allocation". In: *Mathematics of Operations Research* 37.3 (2012), pp. 501–525.

[33] Criteo. *Display Advertising Challenge: Predict click-through rates on display ads. (Contest on Kaggle.)* `https://www.kaggle.com/c/criteo-display-ad-challenge`. June 2014.

[34] Ying Cui et al. "Bid landscape forecasting in online ad exchange marketplace". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2011, pp. 265–273.

[35] George B Dantzig. "Linear programming". In: *Operations research* 50.1 (2002), pp. 42–47.

[36] V. Demyanov and A. Rubinov. *Approximate Methods in Optimization Problems*. New York: American Elsevier Publishing Co., 1970.

[37] Nikhil R Devanur, Kamal Jain, and Robert D Kleinberg. "Randomized primal-dual analysis of ranking for online bipartite matching". In: *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*. SIAM. 2013, pp. 101–107.

[38] Nikhil R Devanur et al. "Near optimal online algorithms and fast approximation algorithms for resource allocation problems". In: *Proceedings of the 12th ACM conference on Electronic commerce*. 2011, pp. 29–38.

[39] Eustache Diemert et al. "Attribution modeling increases efficiency of bidding in display advertising". In: *arXiv preprint arXiv:1707.06409* (2017).

[40] J. Dunn and S. Harshbarger. "Conditional gradient algorithms with open loop step size rules". In: *Journal of Mathematical Analysis and Applications* 62 (1978), pp. 432–444.

[41] M. Frank and P. Wolfe. "An algorithm for quadratic programming". In: *Naval Research Logistics Quarterly* 3 (1956), pp. 95–110.

[42] Robert M Freund and Paul Grigas. "New analysis and results for the Frank–Wolfe method". In: *Mathematical Programming* 155.1-2 (2016), pp. 199–230.

[43] Robert M Freund, Paul Grigas, and Rahul Mazumder. "An Extended Frank–Wolfe Method with "In-Face" Directions, and Its Application to Low-Rank Matrix Completion". In: *SIAM Journal on optimization* 27.1 (2017), pp. 319–346.

[44] Rong Ge, Chi Jin, and Yi Zheng. "No spurious local minima in nonconvex low rank problems: A unified geometric analysis". In: *arXiv preprint arXiv:1704.00708* (2017).

[45] Rong Ge, Jason D Lee, and Tengyu Ma. "Matrix completion has no spurious local minimum". In: *Advances in Neural Information Processing Systems*. 2016, pp. 2973–2981.

[46] Saeed Ghadimi. "Conditional gradient type methods for composite nonlinear and stochastic optimization". In: *Mathematical Programming* (2016), pp. 1–34.

[47] Donald Goldfarb, Garud Iyengar, and Chaoxu Zhou. "Linear convergence of stochastic frank wolfe variants". In: *arXiv preprint arXiv:1703.07269* (2017).

[48] Robert M Gower and Peter Richtárik. "Randomized quasi-Newton updates are linearly convergent matrix inversion algorithms". In: *SIAM Journal on Matrix Analysis and Applications* 38.4 (2017), pp. 1380–1409.

[49] Thore Graepel et al. "Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine". In: Omnipress. 2010.

[50] Paul Grigas, Alfonso Lobos, and Nathan Vermeersch. "Stochastic in-face frank-wolfe methods for non-convex optimization and sparse neural network training". In: *arXiv preprint arXiv:1906.03580* (2019).

[51] Paul Grigas et al. "Profit maximization for online advertising demand-side platforms". In: *Proceedings of the ADKDD'17*. 2017, pp. 1–7.

[52] J. Guélat and P. Marcotte. "Some comments on Wolfe's 'away step'". In: *Mathematical Programming* 35 (1986), pp. 110–119.

[53] Ramakrishna Gummadi, Peter Key, and Alexandre Proutiere. "Repeated auctions under budget constraints: Optimal bidding strategies and equilibria". In: *the Eighth Ad Auction Workshop*. Citeseer. 2012.

[54] Huifeng Guo et al. "DeepFM: a factorization-machine based neural network for CTR prediction". In: *arXiv preprint arXiv:1703.04247* (2017).

[55] LLC Gurobi Optimization. *Gurobi Optimizer Reference Manual*. 2020. URL: `http://www.gurobi.com`.

[56] Song Han et al. "Learning both weights and connections for efficient neural network". In: *Advances in neural information processing systems*. 2015, pp. 1135–1143.

[57] Z. Harchaoui, A. Juditsky, and A. Nemirovski. *Conditional Gradient Algorithms for Norm-Regularized Smooth Convex Optimization*. Technical Report. 2013.

[58] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

[59] Elad Hazan. “Introduction to online convex optimization”. In: *arXiv preprint arXiv:1909.05207* (2019).

[60] Elad Hazan and Haipeng Luo. “Variance-reduced and projection-free stochastic optimization”. In: *International Conference on Machine Learning*. 2016, pp. 1263–1271.

[61] Niao He and Zaid Harchaoui. “Semi-proximal mirror-prox for nonsmooth composite minimization”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 3411–3419.

[62] Xinran He et al. “Practical lessons from predicting clicks on ads at facebook”. In: *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ACM. 2014, pp. 1–9.

[63] Donald W Hearn. “The gap function of a convex program”. In: *Operations Research Letters* 1.2 (1982), pp. 67–71.

[64] Ali Hojjat et al. “A unified framework for the scheduling of guaranteed targeted display advertising under reach and frequency requirements”. In: *Operations Research* 65.2 (2017), pp. 289–313.

[65] Nicole Immorlica et al. “Adversarial bandits with knapsacks”. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2019, pp. 202–219.

[66] Martin Jaggi. “Revisiting Frank-Wolfe: Projection-free sparse convex optimization”. In: *Proceedings of the 30th international conference on machine learning*. CONF. 2013, pp. 427–435.

[67] Stefanus Jasin and Sunil Kumar. “A re-solving heuristic with bounded revenue loss for network revenue management with customer choice”. In: *Mathematics of Operations Research* 37.2 (2012), pp. 313–345.

[68] Rodolphe Jenatton, Jim Huang, and Cédric Archambeau. “Adaptive algorithms for online convex optimization with long-term constraints”. In: *International Conference on Machine Learning*. PMLR. 2016, pp. 402–411.

[69] Bo Jiang et al. “Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis”. In: *arXiv preprint arXiv:1605.02408* (2016).

[70] Chi Jin et al. “How to escape saddle points efficiently”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1724–1732.

[71] Cedric Josz et al. “A theory on the absence of spurious solutions for nonconvex and nonsmooth optimization”. In: *Advances in neural information processing systems*. 2018, pp. 2441–2449.

[72] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. “Field-aware factorization machines in a real-world online advertising system”. In: *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee. 2017, pp. 680–688.

[73] Yuchin Juan et al. "Field-aware factorization machines for CTR prediction". In: *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM. 2016, pp. 43–50.

[74] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. "An optimal algorithm for on-line bipartite matching". In: *Proceedings of the twenty-second annual ACM symposium on Theory of computing*. 1990, pp. 352–358.

[75] *Keras MNIST Tutorial*. `https://github.com/keras-team/keras/blob/master/examples/mnist_mlp.py`. Accessed: 2019-05-22.

[76] Koulik Khamaru and Martin J Wainwright. "Convergence guarantees for a class of non-convex and non-smooth optimization problems". In: *arXiv preprint arXiv:1804.09629* (2018).

[77] Paul Klemperer. "Auction theory: A guide to the literature". In: *Journal of economic surveys* 13.3 (1999), pp. 227–286.

[78] Volodymyr Kuleshov and Doina Precup. "Algorithms for multi-armed bandit problems". In: *arXiv preprint arXiv:1402.6028* (2014).

[79] Simon Lacoste-Julien. "Convergence rate of frank-wolfe for non-convex objectives". In: *arXiv preprint arXiv:1607.00345* (2016).

[80] Simon Lacoste-Julien and Martin Jaggi. "On the global linear convergence of Frank-Wolfe optimization variants". In: *Advances in neural information processing systems*. 2015, pp. 496–504.

[81] Simon Lacoste-Julien et al. "Block-coordinate Frank-Wolfe optimization for structural SVMs". In: *arXiv preprint arXiv:1207.4747* (2012).

[82] Guanghui Lan. "An optimal method for stochastic composite optimization". In: *Mathematical Programming* 133.1-2 (2012), pp. 365–397.

[83] Kuang-Chih Lee, Ali Jalali, and Ali Dasdan. "Real time bid optimization with smooth budget delivery in online advertising". In: *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*. ACM. 2013, p. 1.

[84] Nikolaos Liakopoulos et al. "Cautious regret minimization: Online optimization with long-term budget constraints". In: *International Conference on Machine Learning*. 2019, pp. 3944–3952.

[85] Dong C Liu and Jorge Nocedal. "On the limited memory BFGS method for large scale optimization". In: *Mathematical programming* 45.1 (1989), pp. 503–528.

[86] Huahui Liu et al. "Dual Based DSP Bidding Strategy and its Application". In: *arXiv preprint arXiv:1705.09416* (2017).

[87] Alfonso Lobos et al. "Optimal bidding, allocation and budget spending for a demand side platform under many auction types". In: *arXiv preprint arXiv:1805.11645* (2018).

[88] Christos Louizos, Max Welling, and Diederik P Kingma. "Learning Sparse Neural Networks through L_0 Regularization". In: *arXiv preprint arXiv:1712.01312* (2017).

[89] Haihao Lu and Robert M Freund. "Generalized Stochastic Frank-Wolfe Algorithm with Stochastic" Substitute"Gradient for Structured Convex Optimization". In: *arXiv preprint arXiv:1807.07680* (2018).

[90] Yuhang Ma et al. "An approximation algorithm for network revenue management under nonstationary arrivals". In: *Operations Research* 68.3 (2020), pp. 834–855.

[91] Mehrdad Mahdavi, Rong Jin, and Tianbao Yang. "Trading regret for efficiency: online convex optimization with long term constraints". In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 2503–2528.

[92] Aranyak Mehta et al. "Adwords and generalized online matching". In: *Journal of the ACM (JACM)* 54.5 (2007), p. 22.

[93] Pavlo Molchanov et al. "Pruning convolutional neural networks for resource efficient inference". In: *arXiv preprint arXiv:1611.06440* (2016).

[94] Arkadi Nemirovski et al. "Robust stochastic approximation approach to stochastic programming". In: *SIAM Journal on optimization* 19.4 (2009), pp. 1574–1609.

[95] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science & Business Media, 2013.

[96] Maher Nouiehed, Jason D Lee, and Meisam Razaviyayn. "Convergence to Second-Order Stationarity for Constrained Non-Convex Optimization". In: *arXiv preprint arXiv:1810.02024* (2018).

[97] Julie Nutini et al. "Coordinate descent converges faster with the Gauss-Southwell rule than random selection". In: *International Conference on Machine Learning*. 2015, pp. 1632–1641.

[98] Junwei Pan et al. "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*. International World Wide Web Conferences Steering Committee. 2018, pp. 1349–1357.

[99] Junwei Pan et al. "Predicting different types of conversions with multi-task learning in online advertising". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 2689–2697.

[100] Neal Parikh, Stephen Boyd, et al. "Proximal algorithms". In: *Foundations and Trends® in Optimization* 1.3 (2014), pp. 127–239.

[101] Adam Paszke et al. "Automatic differentiation in PyTorch". In: (2017).

[102] PricewaterhouseCoopers. *IAB internet advertising revenue report*. Tech. rep. 2020.

[103] *Pytorch CIFAR-10 Tutorial*. `https://github.com/pytorch/tutorials/blob/master/beginner_source/blitz/cifar10_tutorial.py`. Accessed: 2019-05-22.

[104] *Pytorch MNIST Tutorial*. `https://github.com/pytorch/examples/tree/master/mnist`. Accessed: 2019-05-22.

[105] Nikhil Rao, Parikshit Shah, and Stephen Wright. "Forward–backward greedy algorithms for atomic norm regularization". In: *IEEE Transactions on Signal Processing* 63.21 (2015), pp. 5798–5811.

[106] Sashank J Reddi et al. "Stochastic frank-wolfe methods for nonconvex optimization". In: *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE. 2016, pp. 1244–1251.

[107] Kan Ren et al. "Bidding Machine: Learning to Bid for Directly Optimizing Profits in Display Advertising". In: *IEEE Transactions on Knowledge and Data Engineering* 30.4 (2018), pp. 645–659.

[108] Steffen Rendle. "Factorization machines". In: *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE. 2010, pp. 995–1000.

[109] Matthew Richardson, Ewa Dominowska, and Robert Ragno. "Predicting clicks: estimating the click-through rate for new ads". In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 521–530.

[110] Herbert Robbins and Sutton Monro. "A Stochastic Approximation Method". In: *Ann. Math. Statist.* 22.3 (Sept. 1951), pp. 400–407. DOI: `10.1214/aoms/1177729586`. URL: `https://doi.org/10.1214/aoms/1177729586`.

[111] Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.

[112] Daniel J Russo et al. "A tutorial on thompson sampling". In: *Foundations and Trends® in Machine Learning* 11.1 (2018), pp. 1–96.

[113] Robert Warren Simpson. *Using network flow techniques to find shadow prices for market demands and seat inventory control*. MIT, Department of Aeronautics and Astronautics, Flight Transportation ..., 1989.

[114] Suvrit Sra, Sebastian Nowozin, and Stephen J Wright. *Optimization for machine learning*. Mit Press, 2012.

[115] Kalyan Talluri and Garrett Van Ryzin. "An analysis of bid-price controls for network revenue management". In: *Management science* 44.11-part-1 (1998), pp. 1577–1593.

[116] Kalyan Talluri and Garrett Van Ryzin. "Revenue management under a general discrete choice model of consumer behavior". In: *Management Science* 50.1 (2004), pp. 15–33.

[117] Markus Thom and Günther Palm. "Sparse activity and sparse connectivity in supervised learning". In: *Journal of Machine Learning Research* 14.Apr (2013), pp. 1091–1143.

[118] John Turner. "The planning of guaranteed targeted display advertising". In: *Operations research* 60.1 (2012), pp. 18–33.

[119] Lieven Vandenberghe and Stephen Boyd. "Semidefinite programming". In: *SIAM review* 38.1 (1996), pp. 49–95.

[120] William Vickrey. "Counterspeculation, auctions, and competitive sealed tenders". In: *The Journal of finance* 16.1 (1961), pp. 8–37.

[121] Thomas WM Vossen and Dan Zhang. "Reductions of approximate linear programs for network revenue management". In: *Operations Research* 63.6 (2015), pp. 1352–1371.

[122] Yuchen Wang et al. "Functional bid landscape forecasting for display advertising". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2016, pp. 115–131.

[123] Xiaohan Wei, Hao Yu, and Michael J Neely. "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*. 2020, pp. 3–4.

[124] Elizabeth Louise Williamson. "Airline network seat inventory control: Methodologies and revenue impacts". PhD thesis. Massachusetts Institute of Technology, 1992.

[125] *Wired Google Roadmap*. `https://www.wired.com/story/antitrust-case-against-google-roadmap-paper/`. Accessed: 07-16-2020.

[126] Wush Chi-Hsuan Wu, Mi-Yen Yeh, and Ming-Syan Chen. "Predicting winning price in real time bidding with censored data". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2015, pp. 1305–1314.

[127] Jun Xiao et al. "Attentional factorization machines: Learning the weight of feature interactions via attention networks". In: *arXiv preprint arXiv:1708.04617* (2017).

[128] Jianjun Yuan and Andrew Lamperski. "Online convex optimization for cumulative constraints". In: *Advances in Neural Information Processing Systems*. 2018, pp. 6137–6146.

[129] Weinan Zhang, Shuai Yuan, and Jun Wang. "Optimal real-time bidding for display advertising". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 1077–1086.

[130] Weinan Zhang et al. "Bid-aware gradient descent for unbiased learning with censored data in display advertising". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2016, pp. 665–674.

[131] Yuchen Zhang, Jason D Lee, and Michael I Jordan. "l1-regularized neural networks are improperly learnable in polynomial time". In: *International Conference on Machine Learning*. 2016, pp. 993–1001.