Robust Budget Allocation via Continuous Submodular Functions

Matthew Staib and Stefanie Jegelka

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
{mstaib, stefje}@mit.edu

Abstract

The optimal allocation of resources for maximizing the influence, the spread of information, or coverage, has gained attention in the past years, in particular in machine learning and data mining. However, in applications, the exact parameters of the problem are rarely known. We hence revisit a continuous version of the Budget Allocation or Bipartite Influence Maximization problem introduced by Alon et al. [3] from a robust optimization perspective, where an adversary may choose the least favorable parameters within a confidence set. The resulting problem is a nonconvex-concave saddle point problem. We show that this nonconvex problem can be solved by leveraging connections to continuous submodular optimization, and by solving a constrained submodular minimization problem. Although constrained submodular minimization is hard in general, here, we establish conditions under which a continuous submodular optimization problem can be solved to arbitrary (additive) precision ϵ.

1 Introduction

The optimal allocation of resources for maximizing the influence, the spread of information, or coverage, has gained attention in the past years, in particular in machine learning and data mining [24; 44; 19; 36; 15].

In the Budget Allocation Problem, one is given a bipartite influence graph between channels S and people T, and the task is to assign a budget $y(s)$ to each channel s in S with the goal of maximizing the expected number of influenced people $\mathcal{I}(y)$. The edge $(s,t) \in E$ between channel s and person t is weighted with a probability p_{st} which represents the likelihood that e.g. an advertisement on the radio station s will influence person t to buy some product. The budget $y(s)$ controls how many independent attempts are made via the channel s to influence the people in T. The probability that a customer t is influenced when the advertising budget is y is given by

$$I_t(y) = 1 - \prod_{(s,t)\in E} [1 - p_{st}]^{y(s)}, \tag{1}$$

and hence the expected number of influenced people is $\mathcal{I}(y) = \sum_{t\in T} I_t(y)$. We will write $\mathcal{I}(y;p) = \mathcal{I}(y)$ to make the dependence on the probabilities p_{st} explicit. The total budget y must remain within some feasible set $\mathcal{Y}$ which may encode e.g. a total budget limit $\sum_{s\in S} y(s) \le C$. We allow the budgets y to be continuous, as in Bian et al. [13].

Since its introduction in [3], several works have extended the formulation of the Budget Allocation problem, and provided efficient algorithms [13; 40; 52; 64; 63]. Budget Allocation may also be viewed as a version of influence maximization on a bipartite graph, where information spreads like the Independent Cascade model. In influence maximization [44], we are to select a set of k seed nodes that maximizes the number of informed nodes after a (pre-specified) diffusion process. If y is binary or integer, both Budget Allocation and Influence maximization are NP-hard. Yet, approximations within a constant factor of $(1 - \frac{1}{e})$ are possible, and build on the fact that the influence function has the property of submodularity in the binary case, and lattice submodularity and diminishing returns in the integer case [64; 40], i.e., it belongs to the class of *DR-submodular* functions. If y is continuous, the problem is a concave maximization problem over linear constraints.

The formulation of Budget Allocation assumes that the transmission probabilities are known exactly. This is however rarely the case in practice. In influence maximization, the probabilities p_{st} (and possibly also the graph itself) need to be inferred from observations, either directly or indirectly via learning the influence function [37; 25; 55; 26; 56]. For Budget Allocation, at best we will know confidence intervals for the p_{st}. Realizing this deficiency, recent work studied robust versions of Influence Maximization, where a budget y must be chosen that maximizes the worst-case approximation ratio over a set of possible influence functions [41; 21; 50]. The resulting optimization problem is difficult but admits bicriteria approximation algorithms.

In this work, we study a formulation of robustness that is corresponds to robust optimization [11; 10]. We maximize the worst-case influence – not approximation ratio – for p in a confidence set centered around the "best guess" (e.g., posterior mean). This avoids a pitfall of maximizing the approximation ratio (which can be misled to return poor worst-case budgets, as demonstrated in Appendix A) while also allowing us to formulate the problem as a max-min game:

$$\max_{y \in \mathcal{B}} \min_{p \in \mathcal{P}} \mathcal{I}(y; p), \tag{2}$$

where an "adversary" can arbitrarily manipulate p within the confidence set $\mathcal{B}$. With p fixed, $\mathcal{I}(y; p)$ is concave in y. However, the influence function $\mathcal{I}(y; p)$ is not convex in the adversary's variables p_{st}.

The insight we exploit in this work is that $\mathcal{I}(y; p)$ has the property of *continuous submodularity*, and can hence be minimized by generalizing techniques from discrete submodular optimization [5]. The techniques in [5] do not apply to our confidence sets, however, they are restricted to box constraints. In fact, general submodular minimization under constraints is hard [65; 33; 42]. In this work, we make the following contributions:

1. We present an algorithm for Robust Budget Allocation in the adversarial scenario (2).
2. We provide conditions and the first results for continuous submodular minimization with box constraints and one more well-behaved constraint.

1.1 Background and Related Work

Before going into details about our robust maximization problem, we discuss background material and, along the way, related work.

1.1.1 Submodularity over the integer lattice and continuous domains

Submodularity is perhaps best known as a property of set functions. A function $F : 2^V \to \mathbb{R}$ defined on subsets $S \subseteq V$ of a ground set V is *submodular* if for all sets $S, T \subseteq V$, it holds that

$$F(S) + F(T) \geq F(S \cap T) + F(S \cup T). \tag{3}$$

A similar definition extends to functions defined over a distributive lattice $\mathcal{L}$, e.g. the integer lattice. Such a function f is submodular if for all $x, y \in \mathcal{L}$, it holds that [31]

$$f(x) + f(y) \geq f(x \vee y) + f(x \wedge y). \tag{4}$$

For the integer lattice and vectors x, y, $x \vee y$ denotes the coordinate-wise maximum and $x \wedge y$ the coordinate-wise minimum. More recently, submodularity has been considered on continuous domains $\mathcal{X} \subset \mathbb{R}^d$, where, if f is also differentiable, the property of submodularity means that all off-diagonal entries of the the Hessian are nonpositive, i.e., $\frac{\partial f(x)}{\partial x_i \partial x_j} \leq 0$ for all $1 \leq i, j \leq d$ [5]. These functions may be convex, concave, or neither.

Submodular functions over discrete domains with box constraints can be minimized by a reduction to the set function (binary) case, more precisely, ring families [14]. More recently, Bach [5] extended results based on the convex Lovász extension, by building on connections to optimal transport. The subclass of $L^\natural$-convex Functions admits steepest descent algorithms [53; 45; 54]. A function on the integer lattice is $L^\natural$-convex if it satisfies a discrete version of midpoint convexity, i.e., for all x, y it holds that $f(x) + f(y) \geq f(\lceil \frac{x+y}{2} \rceil) + f(\lfloor \frac{x+y}{2} \rfloor)$. An $L^\natural$-convex function with domain $\prod_{i=1}^n [k]$, where $[k] = \{0, 1, \ldots, k-1\}$, can be minimized in strongly polynomial time $O(n^4 \log^2 n \cdot k \cdot EO + n^5 \log^{O(1)} n \cdot k)$ by combining the steepest descent algorithm [53] with the submodular minimization algorithm of Lee et al. [49].

Similarly, results for submodular maximization extend to integer lattices. For example, Gottschalk and Peis [38] extend the bidirectional greedy algorithm, and obtain a $\frac{1}{3}$-approximation. Stronger results are possible if the submodular function also satisfies *diminishing returns*: for all $x \leq y$ (coordinate-wise) and i such that $y + e_i \in \mathcal{X}$, it holds that $f(x + e_i) - f(x) \geq f(y + e_i) - f(y)$, i.e., increasing the coordinate of the smaller vector x results in a larger increase of the function value. For such DR-submodular functions, many approximation results for the set function case extend [13; 63; 64]. In particular, Ene and Nguyen [29] show a generic reduction to set function optimization that applies to both maximization and minimization.

The following observations are our own.

Proposition 1.1. *A DR-submodular function f defined on $\prod_{i=1}^n [k_i]$ can be minimized in strongly polynomial time $O(n^4 \log^4 k \cdot \log^2(n \log k) \cdot EO + n^4 \log^4 k \cdot \log^{O(1)}(n \log k))$, where $k = \max_i k_i$ and EO is the time complexity of evaluating f.*

In particular, the time complexity is logarithmic in k. For general lattice submodular function minimization, this is not possible without further assumptions.

Proof. The function f can be reduced to a submodular set function $g : 2^V \to \mathbb{R}$ via [29], where $|V| = O(n \log k)$. The function g can be evaluated via mapping from 2^V to the domain of f, then evaluating f, in time $O(n \log k \cdot EO)$. We can then directly substitute these complexities into the runtime bound given in [49]. $\square$

Remark 1.1. An $L^\natural$-convex function need not be DR-submodular, and vice-versa. Hence, algorithms for optimizing one type may not apply for the other.

Consider $f_1(x_1, x_2) = -x_1^2 - 2x_1x_2$ and $f_2(x_1, x_2) = x_1^2 + x_2^2$, both defined on $\{0, 1, 2\} \times \{0, 1, 2\}$. The function f_1 is DR-submodular but violates discrete midpoint convexity for the pair of points $(0, 0)$ and $(2, 2)$, while f_2 is $L^\natural$-convex but does not have diminishing returns in either dimension.

1.1.2 Robust (Submodular) Optimization

In practice, many optimization problems have parameters that are unknown, and we would like to be 'robust' to any likely instantiations of these parameters. If the distribution of the parameters is known (stochastic optimization), formulations such as value-at-risk and conditional value-at-risk [61; 62] apply. Submodular objectives do not in general admit submodular value-at-risk and conditional value-at-risk, and in fact hardness results exist for optimizing these quantities [51]. Despite this, Zhang et al. [67] give an algorithm with an approximation guarantee for the seed set selection problem which is a variant of the submodular coverage problem. The brittleness of submodular optimization under noise has been studied in [7; 6; 39].

Often, an exact distribution is not known. Robust optimization [10; 11] assumes that the parameters (of the cost function and constraints) can vary arbitrarily within a known confidence set U, and the aim is to optimize the worst-case setting, i.e., $\min_y \sup_{u,A,b \in \mathcal{U}} \{g(y; u) \text{ s.t. } Ay \leq b\}$. Here, we will only have uncertainty in the cost function.

A few robust versions of submodular maximization have been studied. Krause et al. [46] show a bicriterion approximation for maximizing over the point-wise minimum of a finite set of monotone submodular functions F_i, i.e., $\max_{|S| \leq k} \min_{1 \leq i \leq m} F_i(S)$. He and Kempe [41] use this formulation to optimize a worst-case approximation ratio for influence maximization. Chen et al. [21] use two calls to the greedy algorithm, one optimistic and one pessimistic, to optimize this worst-case ratio in the confidence interval setting. Lowalekar et al. [50] also use a parameterized form of the greedy algorithm.

Instead of allowing an arbitrary set of submodular functions, Orlin et al. [58] study the removal of τ elements from the selected set: $\max_{|S| \leq k} \min_{Z, |Z| \leq \tau} F(S \setminus Z)$. Their algorithms achieve a $(1 - 1/e) - \epsilon$-approximation for constant τ and $O(n^{1/\epsilon^3})$ queries, and a 0.387-approximation for $\tau = o(\sqrt{k})$.

1.1.3 Related Problems

The Budget Allocation or Bipartite Influence Maximization Problem is closely related to Influence Maximization on general graphs, and the stochastic covering problem.

Influence Maximization In Influence Maximization, a contagion spreads stochastically through a graph through some diffusion process. The goal is to select an initial set S of nodes to infect, subject to the constraint $|S| \leq k$, in order to maximize the expected number of infected nodes $\sigma(S)$ at the end of the process. For many diffusion models, $\sigma(S)$ is monotone submodular and can be maximized up to a constant factor via the greedy algorithm [44]. However, the influence function $\sigma(S)$ may be #P-hard to compute exactly [20] and must be approximated. This has spurred the development of computationally efficient heuristics [19; 20] as well as efficient (provable) randomized algorithms [15].

Stochastic Coverage The Stochastic Coverage problem introduced by Goemans and Vondrák [34] is a stochastic generalization of Set Cover where the covering sets $S_i \subset V$ are random. A coverage variant of Budget Allocation can be written as stochastic coverage with multiplicity, in which case there is an $\Omega(|V|)$ gap in the number of sets needed between the adaptive case where new sets are chosen one-by-one and the non-adaptive case [34]. Golovin and Krause [35] generalize Stochastic Coverage to the Stochastic Submodular Coverage problem, by replacing the coverage function with a submodular function, and they provide an adaptive version of the usual greedy algorithm . Deshpande et al. [23] build on this to develop an adaptive double greedy algorithm, which yields the first constant factor approximation to the stochastic min-knapsack problem. In a related setting, Adamczyk et al. [2] provide the first constant factor approximation for maximizing monotone submodular functions over a matroid in the stochastic probing framework.

Signomial Optimization *Signomials* are functions which are linear combinations of *monomials* of the form $\prod_i x_i^{c_i}$. General signomial optimization is NP-hard [22], but some subclasses are tractable. Signomials whose monomial coefficients are all positive are called *posynomials*, and can be efficiently minimized subject to posynomial constraints via Geometric Programming [17]. Signomials with only one negative coefficient can be minimized efficiently and this has led to sum of squares-like relaxations for general signomial optimization [18]. In this paper, the adversary's optimization problem is equivalent to constrained posynomial maximization which is not in general a Geometric Program. There is some work on duality and optimality conditions for constrained posynomial maximization based on approximating the posynomial with a monomial [59; 28], but to our knowledge we are the first to give an algorithm which solves such a problem to arbitrary suboptimality.

2 Robust and Stochastic Budget Allocation

2.1 The Influence Function

The Budget Allocation and Influence Maximization problems rely on parameters – transmission probabilities or edge weights in a graph – that are assumed to be known. However, often, these parameters are not known exactly and must be inferred from observations. In such cases, we may have posterior distributions or, a weaker assumption, confidence sets for the parameters. For the specific case of Budget Allocation, the unknown parameters are the transmission probabilities p_{st}, or equivalently, the failure probabilities $x_{st} = 1 - p_{st}$. These failure probabilities are slightly easier to work with, and we write $\mathcal{I}(x, y) = \mathcal{I}(y; p) = \sum_{t \in T}(1 - \prod_{s \in S} x_{st}^{y(s)})$.

2.2 Stochastic Optimization.

The simplest strategy to handle uncertainty, if a (posterior) distribution of the parameters is known, is to use expectations. For example, we may place a uniform prior on x_{st}, and observe observe n_{st} independent observations drawn from $\mathrm{Ber}(x_{st})$. If we observe α_{st} failures and and β_{st} successes, the resulting posterior distribution is $\mathrm{Beta}(1 + \alpha_{st}, 1 + \beta_{st})$.

Given such a posterior, we may optimize

$$\max_{y \in \mathcal{Y}} \mathcal{I}(\mathbb{E}[X], y), \text{ or} \tag{5}$$

$$\max_{y \in \mathcal{Y}} \mathbb{E}[\mathcal{I}(X, y)]. \tag{6}$$

These stochastic problems can be solved exactly:

Proposition 2.1. *Problems* (5) *and* (6) *are concave maximization problems over the (convex) set* $\mathcal{Y}$.

Concavity of (6) follows since it is an expectation over concave functions. Problem (6) can be solved e.g. by stochastic gradient ascent, or by explicitly computing gradients of the expectation. In our case of independent Beta posteriors, the gradient can be computed by noting that each term $I_t(X, y)$ takes the form

$$\mathbb{E}\left[\prod_{i=1}^{n} X_i^{y_i}\right] = \prod_{i=1}^{n} \frac{B(\alpha_i + y_i, \beta_i)}{B(\alpha_i, \beta_i)} \tag{7}$$

and differentiating. Here, $y_i \sim \text{Beta}(\alpha_i, \beta_i)$ and $B(\alpha, \beta)$ is the Beta function. Note that by Lemma 3.1, $\mathbb{E}[\mathcal{I}(X, y)]$ with beta posteriors is also continuous submodular in y.

Merely maximizing expectation does not explicitly account for volatility and hence risk. There are techniques which explicitly encode risk, e.g. value-at-risk (VaR) and conditional value-at-risk (CVaR), as well as some techniques which are either good heuristics or lead to probabilistic guarantees in both continuous [9; 11] as well as discrete [57; 4] settings. When we want to robustly minimize a function $f(x, \omega)$ of x parameterized by a random variable ω, a tried-and-true technique is to solve:

$$\min_{x} \ \mathbb{E}[f(x, \omega)] + \varepsilon\sqrt{\text{Var}(f(x, \omega))} \tag{8}$$

for appropriately chosen ε [9; 11; 57; 4]. Typically f is linear in each of x and ω, and so this is a convex problem. For Budget Allocation, f is nonlinear but as stated above, $\mathbb{E}[f(x, \omega)]$ is still well-behaved. However, for Budget Allocation, we have the following negative result:

Fact 2.1. For y in the nonnegative orthant, the square root of the variance $\sqrt{\text{Var}(\mathcal{I}(X, y))}$ need not be convex or concave, and need not be submodular or supermodular.

We have not ruled out other potential properties which would make Problem 8 tractable, nor have we ruled out other ways of getting probabilistic guarantees such as VaR/CVaR. However, the difficulties here further motivate the Robust Optimization version of Budget Allocation which is our primary focus.

2.3 Robust Optimization

The focus of this work is the robust version of Budget Allocation, where we allow an adversary to arbitrarily set the parameters x within an uncertainty set $\mathcal{X}$. This uncertainty set may result, for instance, from a known distribution, or simply assumed bounds. Formally, we solve

$$\max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \mathcal{I}(x, y), \tag{9}$$

where $\mathcal{Y} \subset \mathbb{R}_+^S$ is a convex set with an efficient projection oracle, and $\mathcal{X}$ is an uncertainty set containing an estimate $\hat{x}$. In the sequel, we will use uncertainty sets of the form $\mathcal{X} = \{x \in \text{Box}(l, u) : R(x) \leq B\}$, where R is a distance (or divergence) from the estimate $\hat{x}$, and $\text{Box}(l, u)$ is the box $\prod_{(s,t)\in E}[l_{st}, u_{st}]$. The intervals $[l_{st}, u_{st}]$ can be thought of as either confidence intervals around $\hat{x}$, or in the case $[0, 1]$ as the constraint that each x_{st} is a valid probability.

Common examples of uncertainty sets used in Robust Optimization are *Ellipsoidal* and *D-norm uncertainty sets* [11]. Our algorithm in Section 3.1 applies to both.

Ellipsoidal uncertainty. The ellipsoidal or quadratic uncertainty set is defined by

$$\mathcal{X}^Q(\gamma) = \{x \in \text{Box}(0, 1) : (x - \hat{x})^T \Sigma^{-1}(x - \hat{x}) \leq \gamma\}, \tag{10}$$

where Σ is the covariance of the random vector X of probabilities distributed according to our Beta posteriors. In our case, since the distributions on each x_{st} are independent, Σ^{-1} is actually diagonal. Writing $\Sigma = \text{diag}(\sigma^2)$, we have

$$\mathcal{X}^Q(\gamma) = \left\{ x \in \text{Box}(0, 1) : \sum_{(s,t)\in E} R_{st}(x_{st}) \leq \gamma \right\}, \tag{11}$$

where $R_{st}(x) = (x_{st} - \hat{x}_{st})^2 \sigma_{st}^{-2}$.

D-norm uncertainty: The D-norm uncertainty set is much like an ℓ_1-ball around $\hat{x}$, and is defined by

$$\mathcal{X}^D(\Gamma) = \left\{ x : \exists c \in \text{Box}(0, 1) \text{ s.t. } x_{st} = \hat{x}_{st} + (u_{st} - \hat{x}_{st})c_{st}, \sum_{(s,t)\in E} c_{st} \leq \Gamma \right\}. \tag{12}$$

Essentially, we allow an adversary to increase $\hat{x}_{st}$ up to some upper bound u_{st}[1], subject to some total budget Γ across all terms x_{st}. The set $\mathcal{X}^D(\Gamma)$ can be rewritten as

$$\mathcal{X}^D(\Gamma) = \left\{ x \in \text{Box}(\hat{x}, u) : \sum_{(s,t)\in E} R_{st}(x_{st}) \leq \Gamma \right\}, \tag{13}$$

where $R_{st}(x_{st}) = (x_{st} - \hat{x}_{st})/(u_{st} - \hat{x}_{st})$ is the fraction of the interval $[\hat{x}_{st}, u_{st}]$ we have used up in increasing x_{st}.

The min-max formulation $\max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{I}(x, y)$ has several merits: the model is not tied to a specific learning algorithm for the probabilities x as long as we can choose a "reasonable" distance function d and optionally compute confidence intervals. This formulation also encodes fully worst-case robustness. It only remains to be seen that we can actually solve the underlying optimization problem.

3 Optimization Algorithm and Submodularity

As noted above and also observed in [40], the function $\mathcal{I}(x, y)$ is concave as a function of y for fixed x. As a pointwise minimum of concave functions, $F(y) := \min_{x\in\mathcal{X}} \mathcal{I}(x, y)$ is concave. Hence, if we

[1]For example, the upper boundary of the confidence interval $[l_{st}, u_{st}]$.

can compute subgradients of $F(y)$, we can solve our max-min-problem via the subgradient method, as outlined in Algorithm 1.

A subgradient $g_y \in \partial F(y)$ at a given point y is given by the gradient of $\mathcal{I}(x^*, y)$ for the minimizing $x^* \in \arg\min_{x \in \mathcal{X}} \mathcal{I}(x, y)$, i.e., $g_y = \nabla_y \mathcal{I}(x^*, y)$. Hence, we can apply the subgradient method if we can compute x^* for any y. In this case, we also obtain a duality gap via

$$\min_{x\in\mathcal{X}} \mathcal{I}(x, y') \le \max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{I}(x, y) \le \max_{y\in\mathcal{Y}} \mathcal{I}(x', y). \tag{14}$$

This means we can estimate the optimal value $\mathcal{I}^*$, and as long as the estimate is good, we can use Polyak's stepsize rule for the subgradient method [16; 60].

Algorithm 1: Subgradient-Ascent

```
   Input: A suboptimality tolerance $\varepsilon > 0$ and initial feasible budget $y^{(0)} \in \mathcal{Y}$
   Output: An $\varepsilon$-optimal budget $y$ for Problem (9)
1  do
2      $x^{(k)} \leftarrow \arg\min_{x\in\mathcal{X}} \mathcal{I}(x, y^{(k)})$;
3      $g^{(k)} \leftarrow \nabla_y \mathcal{I}(x^{(k)}, y)$;
4      $L^{(k)} \leftarrow \mathcal{I}(x^{(k)}, y^{(k)})$;
5      $U^{(k)} \leftarrow \max_{y\in\mathcal{Y}} \mathcal{I}(x^{(k)}, y)$;
6      $\gamma^{(k)} \leftarrow (U^{(k)} - L^{(k)})/\|g^{(k)}\|_2^2$;
7      $y^{(k+1)} \leftarrow \mathrm{proj}_{\mathcal{Y}}(y^{(k)} + \gamma^{(k)} g^{(k)})$;
8      $k \leftarrow k + 1$;
9  while $U^{(k)} - L^{(k)} > \varepsilon$;
10 return $y^{(k)}$
```

But $\mathcal{I}(x, y)$ is not convex in x, and not even quasiconvex . So, it is not immediately clear that we can solve the inner optimization problem. But, while not convex, $\mathcal{I}(x, y)$ as a function of x is *continuous submodular.*

Lemma 3.1. *Suppose we have $n \ge 1$ differentiable functions $f_i : \mathbb{R} \to \mathbb{R}_+$, for $i = 1, \ldots, n$, which are either all nonincreasing or all nondecreasing. Then, $f(x) = \prod_{i=1}^n f_i(x_i)$ is a continuous supermodular function from $\mathbb{R}^n$ to $\mathbb{R}_+$.*

Proof. First, if the number of functions $n = 1$, the resulting function is modular and therefore supermodular. In the case $n \ge 2$, we simply need to compute derivatives. The mixed derivatives are given by

$$\frac{\partial f}{\partial x_i \partial x_j} = f_i'(x_i) f_j'(x_j) \cdot \prod_{k\ne i,j} f_k(x_k). \tag{15}$$

By monotonicity, f_i' and f_j' have the same sign, so their product is nonnegative, and since each f_k is nonnegative, the entire expression is nonnegative. Hence, $f(x)$ is continuous supermodular. □

Corollary 3.1. *The influence function $\mathcal{I}(x, y)$ defined in section 2 is continuous submodular in x over the nonnegative orthant, for each $y \ge 0$.*

Proof. Since submodularity is preserved under nonnegative weighted sums, it suffices to show that each function $I_t(y)$ is continuous submodular. By Lemma 3.1, since $f_s(z) = z^{y(s)}$ is nonnegative and monotone nondecreasing for $y(s) \geq 0$, the product $\prod_{(s,t)\in E} x_{st}^{y(s)}$ is continuous supermodular in x. Flipping the sign and adding a constant term yields $I_t(y)$, which is therefore continuous submodular, so we are done. $\square$

Conjecture 3.1. *Strong duality holds, i.e.*

$$\max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{I}(x, y) = \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{I}(x, y). \tag{16}$$

If strong duality holds, then the duality gap $\max_{y\in\mathcal{Y}} \mathcal{I}(x^*, y) - \min_{x\in\mathcal{X}} \mathcal{I}(x, y^*)$ in Equation (14) is zero at optimality, meaning our stepsize choice is reasonable. If $\mathcal{I}(x, y)$ were quasiconvex in x, strong duality would hold by Sion's min-max theorem, however $\mathcal{I}(x, y)$ need not be quasiconvex: one can show using standard methods [66, chap. 12] that $f(x_1, x_2, x_3) = 1 - x_1x_2 - \sqrt{x_3}$ is not quasiconvex on $\mathbb{R}^3_+$, from which the claim follows. In practice the duality gap always converges to zero, meaning there is empirical evidence for strong duality.

Bach [5] demonstrates how to minimize a continuous submodular function $H(x)$ subject to box constraints $x \in \text{Box}(l, u)$, up to an arbitrary suboptimality gap $\varepsilon > 0$. The constraint set $\mathcal{X}$ in our Robust Budget Allocation problem, however, has box constraints with an additional constraint $R(x) \leq B$. This case is not addressed in [5], nor, to our knowledge in any other work. Fortunately, for a large class of functions R, there is still an efficient algorithm for continuous submodular minimization, which we present in the next section.

3.1 Constrained Continuous Submodular Function Minimization

We next address an algorithm for minimizing a monotone continuous submodular function $H(x)$ subject to box constraints $x \in \text{Box}(l, u)$ and a constraint $R(x) \leq B$, which corresponds to $d(x, \hat{x}) \leq B$ in Robust Budget Allocation:

$$\begin{array}{ll} \text{minimize} & H(x) \\ \text{s.t.} & R(x) \leq B \\ & x \in \text{Box}(l, u). \end{array} \tag{17}$$

If H and R were convex, the constrained problem would be equivalent to solving the problem for the right Lagrange multipler $\lambda^* \geq 0$:

$$\begin{array}{ll} \text{minimize} & H(x) + \lambda^* R(x) \\ \text{s.t.} & x \in \text{Box}(l, u). \end{array} \tag{18}$$

Although H and R are not necessarily convex here, it turns out that a similar approach indeed applies here.

Following [5], we discretize the problem; for a sufficiently fine discretization, we will achieve arbitrary accuracy. Let A be an *interpolation mapping* that maps the discrete set $\prod_{i=1}^n [k_i]$ into $\text{Box}(l, u) = \prod_{i=1}^n [l_i, u_i]$ via the componentwise interpolation functions $A_i : [k_i] \to [l_i, u_i]$. We say A_i is *δ-fine* if $A_i(x_i + 1) - A_i(x_i) \leq \delta$ for all $x_i \in \{0, 1, \ldots, k_i - 2\}$, and we say the full interpolation function A is δ-fine if each A_i is δ-fine. Intuitively, the smaller the fineness δ, the more discrete points we use to approximate the original continuous domain $\mathcal{X}$.

This mapping yields functions $H^\delta : \prod_{i=1}^n [k_i] \to \mathbb{R}$ and $R^\delta : \prod_{i=1}^n [k_i] \to \mathbb{R}$ via $H^\delta(x) = H(A(x))$ and $R^\delta(x) = R(A(x))$. H^δ is lattice submodular (on the integer lattice). This construction leads to a reduction of Problem (17) to a submodular minimization problem over the integer lattice:

$$\begin{array}{ll} \text{minimize} & H^\delta(x) + \lambda R^\delta(x) \\ \text{s.t.} & x \in \prod_{i=1}^n [k_i]. \end{array} \tag{19}$$

Ideally, there should then exist a λ such that the associated minimizer $x(\lambda)$ yields a close to optimal solution for the constrained problem. Theorem 3.1 below states that this is indeed the case.

Moreover, a second benefit of submodularity is that we can find the entire solution path for Problem (19) by solving a single optimization problem.

Lemma 3.2. *Suppose H is continuous submodular, and suppose the regularizer R is strictly increasing and separable, i.e. $R(x) = \sum_{i=1}^n R_i(x)$. Then, we can recover a minimizer $x(\lambda)$ for the induced discrete Problem* (19) *for any $\lambda \in \mathbb{R}$ by solving a single convex optimization problem.*

The problem in question takes the form

$$\begin{array}{ll} \text{minimize} & h_\downarrow(\rho) + \sum_{i=1}^n \sum_{x_i=1}^{k_i-1} a_{ix_i}(\rho_i(x_i)) \\ \text{s.t.} & \rho \in \prod_{i=1}^n \mathbb{R}_\downarrow^{k_i-1} \end{array} \tag{20}$$

where $\mathbb{R}_\downarrow^k$ refers to the set of vectors in $z \in \mathbb{R}^k$ which satisfy $z_1 \geq z_2 \geq \cdots \geq z_k$, the function $h_\downarrow$ is the convex extension of H^δ as defined in [5], the notation $\rho_i(x_i)$ refers to the x_i-th coordinate of the vector ρ_i, and a_{ix_i} are strictly convex functions given by $a_{ix_i}(t) = \frac{1}{2}t^2 \cdot [R_i^\delta(x_i) - R_i^\delta(x_i - 1)]$. We detail the relationship between Problems (19) and (20), as well as how to solve Problem (20) using Frank-Wolfe methods [47; 30; 43; 27], in Appendix B,.

Let ρ^* be the optimal solution for Problem (20). For any λ, we obtain a rounded solution $x(\lambda)$ for Problem (19) from ρ^* by thresholding: we set $x(\lambda)_i = \max\{j \mid 1 \leq j \leq k_i - 1,\ \rho_i^*(j) \geq \lambda\}$, or zero if $\rho_i^*(j) < \lambda$ for all j. Each $x(\lambda')$ is the optimal solution for Problem (19) with $\lambda = \lambda'$. We use the largest parameterized solution $x(\lambda)$ which is still feasible, i.e. the solution $x(\lambda^*)$ where λ^* solves

$$\begin{array}{ll} \min & H^\delta(x(\lambda)) \\ \text{s.t.} & \lambda \geq 0 \\ & R^\delta(x(\lambda)) \leq B, \end{array} \tag{21}$$

which can be found efficiently via binary search, much like Fujishige et al. [32].

Theorem 3.1. *Let H be continuous submodular and monotone decreasing, with ℓ_∞-Lipschitz constant G, and R be strictly increasing and separable. Assume all entries $\rho_i^*(j)$ of the optimal solution ρ^* of Problem* (20) *are distinct. Let $x' = A(x(\lambda^*))$ be the thresholding corresponding to the optimal solution λ^* in Problem* (21)*, mapped back into the original continuous domain $\mathcal{X}$. Then x' is feasible for the continuous Problem* (17)*, and is a $2G\delta$-approximate solution:*

$$H(x') \leq 2G\delta + \min_{x \in \mathrm{Box}(l,u),\ R(x) \leq B} H(x).$$

Theorem 3.1 implies an algorithm for solving Problem (17) to ε-optimality: (1) set $\delta = \varepsilon/G$, (2) compute ρ^* which solves Problem (20), (3) find the optimal thresholding of ρ^* by determining the smallest λ^* for which $R^\delta(x(\lambda^*)) \leq B$, and (4) map $x(\lambda^*)$ back into continuous space via the interpolation mapping A.

Remark 3.1. Theorem 3.1 is proved by comparing x' and x^* to the optimal solution on the discretized mesh

$$x_d^* \in \operatorname*{argmin}_{x \in \prod_{i=1}^n [k_i] : R^\delta(x) \leq B} H^\delta(x).$$

Beyond the theoretical guarantee of Theorem 3.1, for any problem instance and candidate solution x', we can compute a bound on the gap between $H(x')$ and $H^\delta(x_d^*)$. In fact, there are two possible bounds, which we prove in the appendix.

1. We can generate another discete point $x(\lambda_+)$ which satisfies

$$H(x') \leq [H(x') - H^\delta(x(\lambda_+))] + H^\delta(x_d^*).$$

2. The Lagrangian also yields a bound:

$$H(x') \leq \lambda^*(B - R(x')) + H^\delta(x_d^*).$$

Remark 3.2. The requirement in Theorem 3.1 that the elements of ρ^* be distinct may seem somewhat restrictive, but as long as ρ^* has distinct elements in the neighborhood of our particular λ^*, this bound still holds. We see in Section 4.1.1 that in practice our problems seem well-behaved: ρ^* typically has distinct elements in the regime we care about, and the bounds of Remark 3.1 are very good.

3.2 Application to Robust Allocation

The above algorithm directly applies to Robust Allocation with the uncertainty sets in Section 2.3. The ellipsoidal uncertainty set $\mathcal{X}^Q$ corresponds to the constraint that $\sum_{(s,t)\in E} R_{st}(x_{st}) \leq \gamma$ with $R_{st}(x) = (x_{st} - \hat{x}_{st})^2 \sigma_{st}^{-2}$, and $x \in \text{Box}(0,1)$. By the monotonicity of $\mathcal{I}(x,y)$, there is never incentive to reduce any x_{st} below $\hat{x}_{st}$, so we can replace $\text{Box}(0,1)$ with $\text{Box}(\hat{x},1)$. On this interval, each R_{st} is strictly increasing, and Theorem 3.1 applies.

For D-norm uncertainty sets, we have $R_{st}(x_{st}) = (x_{st} - \hat{x}_{st})/(u_{st} - \hat{x}_{st})$. Since each R_{st} is monotone, Theorem 3.1 applies.

4 Experiments

We evaluate our algorithm for Robust Budget Allocation on both synthetic test data as well as a real-world bidding dataset from Yahoo! Webscope [1] to demonstrate that our method yields real improvements. For all experiments, we solved the Robust Budget Allocation problem using Algorithm 1 as the outer loop. For the inner submodular minimization step, we implemented the pairwise Frank-Wolfe algorithm of [48]. In all cases, the feasible set of budgets $\mathcal{Y}$ is $\{y \in \mathbb{R}_+^S : \sum_{s\in S} y(s) \leq C\}$ where the specific budget C depends on the experiment.

4.1 Synthetic

We wish to evaluate both the quality of our solutions to the inner constrained submodular minimization problem (17) as well as the performance of the resulting advertiser budget y. For both settings, we set $|S| = 6$ and $|T| = 2$, and drew each "true" transmission probability p_{st} between channel s and person t uniformly from $[0, 1]$. We then generated a Beta posterior for each p_{st} by uniformly choosing a number of observations n from $\{0, 1, \ldots, 5\}$ and then sampling n times from $\text{Ber}(p_{st})$. We then ran our Robust Budget Allocation code for both the ellipsoidal uncertainty set $\mathcal{X}^Q(\gamma)$ and the D-norm uncertainty set $\mathcal{X}^D(\Gamma)$. The upper bounds u_{st} for the D-norm set were chosen by computing the 97.5%-quantile of the posterior (we chose each interval $[l_{st}, u_{st}]$ to contain 95% of the probability mass). We discretized the resulting influence functions to a fineness of $\varepsilon = 0.001$. Each instance was allowed 200 iterations of the outer subgradient step and 50 iterations of the inner Frank-Wolfe method to select a robust budget y_{robust}.

4.1.1 Optimality

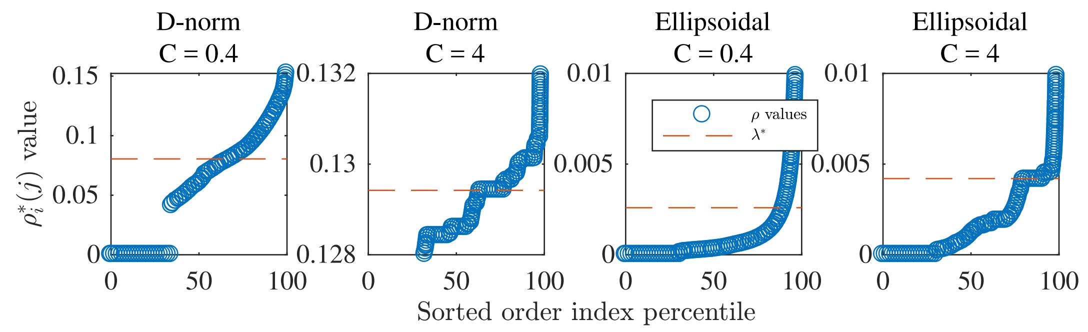
We solved the Robust Budget Allocation code for four total cases which used either an Ellipsoidal or a D-norm uncertainty set, and a total influence budget $C \in \{0.4, 4\}$. In Figure 1a we evaluate the requirement in Theorem 3.1 and Remark 3.2 that the values $\rho_i^*(j)$ be distinct in the neighborhood of our chosen Lagrange multiplier λ^*. For each case we compute ρ^* corresponding to the adversary's strategy against the computed optimal budget y_{robust}. We then plot all elements of ρ^* in sorted order, as well as a dashed line indicating our Lagrange multiplier λ^* which serves as a threshold. We observe that when the influencer is weaker (i.e. $C = 0.4$), nearly all elements of ρ^* are distinct. When the influencer is stronger ($C = 4$), there are some plateaus, meaning there are repeated ρ^* values, but most values are still distinct.

In the case that ρ^* does not have distinct values, we would like to know that our solution dependent suboptimality bounds from Remark 3.1 are good. In finding the optimal robust budget, we solve a constrained submodular minimization problem for each of our 200 outer iterations. For each solve, we compute the solution-dependent bounds, and combine the bounds from all 200 iterations into histograms in Figure 1b. The results are promising: never is the gap greater than 2×10^{-4}. The second, Lagrangian-based bound performs slightly better, but the difference is not huge, and both bounds seem to have practical benefit.

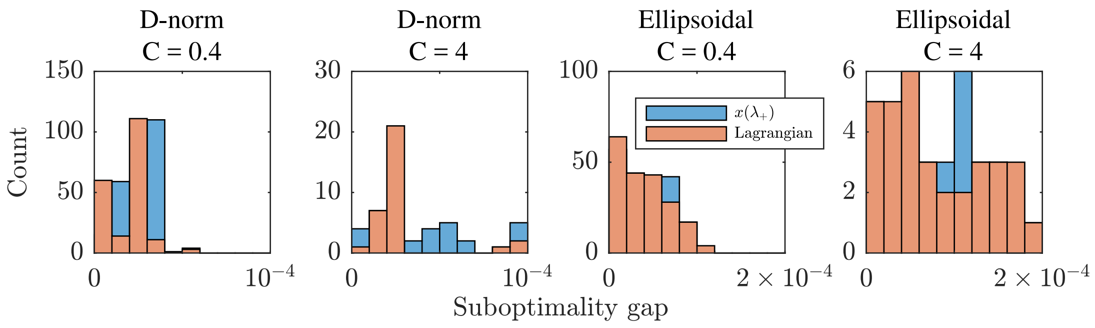
4.1.2 Robust Budget Quality

We use the same basic experimental setup as above. Since the problem instances are relatively small, we are able to compare the performance of the budgets y_{robust} across many different uncertainty sets, namely Ellipsoidal sets $\mathcal{X}^Q(\gamma)$ and D-norm sets $\mathcal{X}^D(\Gamma)$ for many parameter values γ and Γ. In all trials, the influencer was given a total budget of $C = 0.4$. To compare, we also computed a budget $y_{\text{nom}} \in \text{argmax}_{y \in \mathcal{Y}} \mathcal{I}(\hat{x}, y)$ which treated the estimates $\hat{x}$ as the true values, as well as a budget $y_{\text{expect}} \in \text{argmax}_{y \in \mathcal{Y}} \mathbb{E}[\mathcal{I}(X, y)]$ as per Section 2.2. These two optimization problems were solved via standard first-order methods using TFOCS [8].

Figure 2 demonstrates that the robustly-chosen budget y_{robust} indeed beats the other two budgets when an adversary intervenes. This is true for both types of uncertainty sets, and uniformly across the entire range of parameter choices. The robustly chosen y_{robust} attains significant expected influence even in some cases where the noise-oblivious y_{nom} manages zero expected influence. It is worth pointing out that y_{expect} performs nearly as well as y_{robust} in all instances. Maximizing the expectation $\mathbb{E}[\mathcal{I}(X, y)]$ may therefore be a useful surrogate in practice, since computing y_{expect} is more efficient for large instances than computing y_{robust}.



(a) Visualization of the sorted values of $\rho_i^*(j)$ with comparison to the particular Lagrange multiplier λ^*. Note that in most regimes there are no duplicate values, so that Theorem 3.1 usually applies.



(b) Comparison of the two problem-dependent suboptimality gaps from Remark 3.1 which each bound $H^\delta(x(\lambda^*)) - H^\delta(x_d^*)$. The Lagrangian-based bound is slightly tighter, but both methods yield gaps on the order of 10^{-4}.

Figure 1: Experimental evaluation of the applicability of Theorem 3.1 and general bounds on optimality.

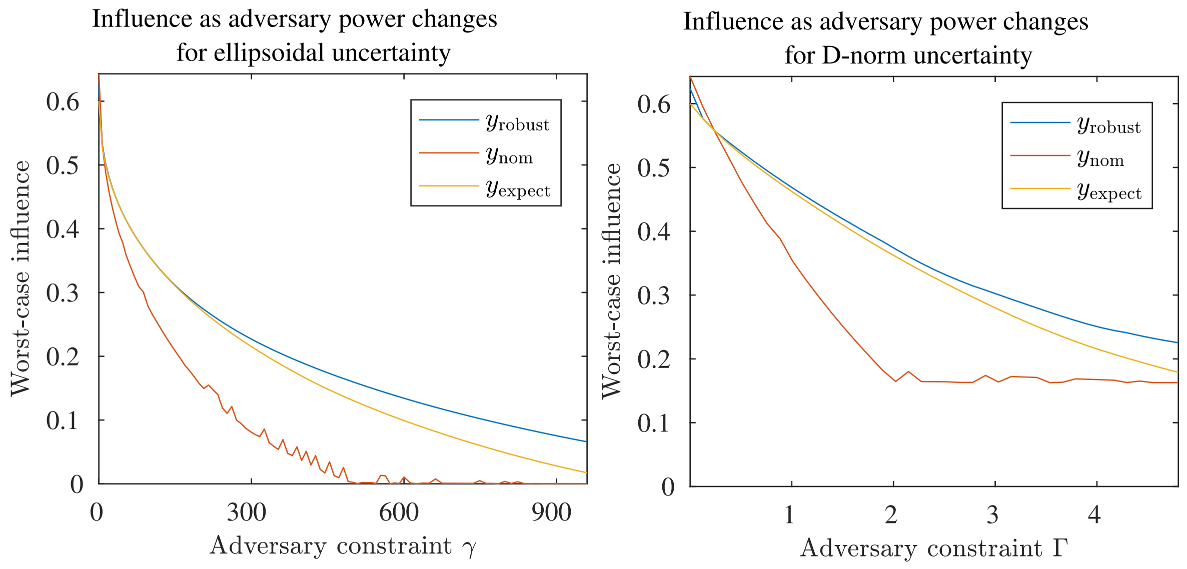


Figure 2: A comparison of the worst-case expected influences for ellipsoidal uncertainty sets $\mathcal{X}^Q(\gamma)$ (left) and D-norm uncertainty sets $\mathcal{X}^D(\Gamma)$ (right). How to read these graphs: for a particular γ, we compare $\min_{x\in\mathcal{X}(\gamma)} \mathcal{I}(x, y)$ for each candidate budget y.

4.2 Yahoo! data

To evaluate our method on real-world data, we formulate a Budget Allocation problem instance from advertiser bidding data from Yahoo! Webscope [1]. This dataset logs bids on 1000 different phrases by advertising accounts. We map the phrases to channels S and the accounts to customers T, with an edge between s and t if a corresponding bid was made. For each pair (s, t), we draw the associated transmission probability p_{st} uniformly from $[0, 0.4]$. We bias these towards zero because we expect people are not so easily influenced by advertising in the real world. We then generate an estimate $\hat{p}_{st}$ and build up a posterior by generating n_{st} samples from $\text{Ber}(p_{st})$, where n_{st} is the number of bids in the dataset between s and t.

To keep our experiments timely, we subsample 200 nodes from S, which induces a bipartite graph with $|T| = 2095$ and 3701 edges, and use this as the graph for the Budget Allocation problem. We use the D-norm uncertainty set $\mathcal{X}^D(\Gamma)$, again with bounds provided by 95% confidence intervals, and adversary power $\Gamma = 185.05$. The constraint on the total budget $\sum_{s\in S} y(s)$ is 6. We discretize with parameter $\varepsilon = 0.05$. We ran 200 outer subgradient iterations and allowed up to 100 inner Frank-Wolfe iterations. These experiments took about 19 hours to run on a Xubuntu system with an i3-6100 and 16GB of RAM. Our implementation used only one thread, but the most computationally intensive part, the repeated evaluation of $\mathcal{I}$ within the greedy algorithm, can be trivially parallelized.

Our results are presented in Figures 3 and 4. Figure 3 shows the convergence properties of our algorithm. In particular, the worst-case influence of the budget at each iteration clearly converges to the upper bound provided by Equation 14. Moreover, it takes only 4 outer iterations (only six minutes of CPU time) before our budget is better in the worst case than the noise-oblivious budget y_{nom}. Figure 4 compares the budgets chosen by optimizing the noiseless, expected-value,

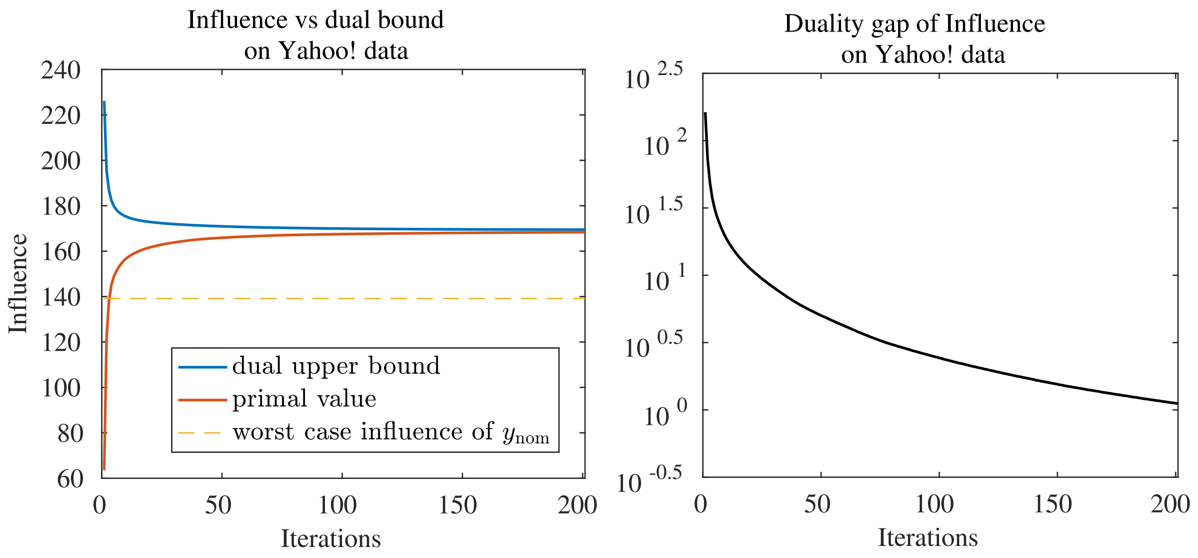


Figure 3: Demonstration of convergence properties of our algorithm on real data. Note that it takes very few iterations for the robustly chosen budget y_{robust} to have much better worst-case properties than the obliviously chosen budget y_{nom}.

	y_{nom}	y_{expect}	y_{robust}
$\mathcal{I}(\hat{x}, y)$	**246.4**	227.9	217.6
$\mathbb{E}[\mathcal{I}(X, y)]$	1071.3	**1076.4**	1074.9
$\min_{x\in\mathcal{X}} \mathcal{I}(x, y)$	139.1	161.5	**168.4**

Figure 4: Table comparing the budgets across all three objective values discussed in this paper.

and adversarial forms of the Budget Allocation problem in each setting. In particular, all budgets perform similarly in terms of expected influence. The oblivious budget y_{nom} performs clearly better in the nominal case and clearly worse in the adversarial case than the robust budget y_{robust}, with y_{expect} performing in the middle. In this sense, y_{robust} is more conservative than y_{expect}, which is in turn more conservative than y_{nom}.

5 Conclusion

References

[1] Yahoo! Webscope dataset ydata-ysm-advertiser-bids-v1_0. URL `http://research.yahoo.com/Academic_Relations`.

[2] Marek Adamczyk, Maxim Sviridenko, and Justin Ward. Submodular Stochastic Probing on Matroids. *Mathematics of Operations Research*, 41(3):1022–1038, April 2016. ISSN 0364-765X. doi: 10.1287/moor.2015.0766.

[3] Noga Alon, Iftah Gamzu, and Moshe Tennenholtz. Optimizing Budget Allocation Among Channels and Influencers. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 381–388, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1229-5. doi: 10.1145/2187836.2187888.

[4] Alper Atamtürk and Vishnu Narayanan. Polymatroids and mean-risk minimization in discrete optimization. *Operations Research Letters*, 36(5):618–622, September 2008. ISSN 0167-6377. doi: 10.1016/j.orl.2008.04.006.

[5] Francis Bach. Submodular Functions: From Discrete to Continous Domains. *arXiv:1511.00394 [cs, math]*, November 2015.

[6] Eric Balkanski, Aviad Rubinstein, and Yaron Singer. The limitations of optimization from samples. *arXiv preprint arXiv:1512.06238*, 2015.

[7] Eric Balkanski, Aviad Rubinstein, and Yaron Singer. The power of optimization from samples. In *Advances In Neural Information Processing Systems*, pages 4017–4025, 2016.

[8] Stephen R Becker, Emmanuel J Candès, and Michael C Grant. Templates for convex cone problems with applications to sparse signal recovery. *Mathematical programming computation*, 3(3):165–218, 2011.

[9] Aharon Ben-Tal and Arkadi Nemirovski. Robust solutions of Linear Programming problems contaminated with uncertain data. *Mathematical Programming*, 88(3):411–424, September 2000. ISSN 0025-5610, 1436-4646. doi: 10.1007/PL00011380.

[10] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

[11] D. Bertsimas, D. Brown, and C. Caramanis. Theory and Applications of Robust Optimization. *SIAM Review*, 53(3):464–501, January 2011. ISSN 0036-1445. doi: 10.1137/080734510.

[12] Michael J. Best and Nilotpal Chakravarti. Active set algorithms for isotonic regression; A unifying framework. *Mathematical Programming*, 47(1-3):425–439, 1990. ISSN 0025-5610, 1436-4646. doi: 10.1007/BF01580873.

[13] Yatao Bian, Baharan Mirzasoleiman, Joachim M. Buhmann, and Andreas Krause. Guaranteed Non-convex Optimization: Submodular Maximization over Continuous Domains. *arXiv:1606.05615 [cs]*, June 2016.

[14] Garrett Birkhoff. Rings of sets. *Duke Mathematical Journal*, 3(3):443–454, 1937.

[15] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. Maximizing Social Influence in Nearly Optimal Time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 946–957, Philadelphia, PA, USA, 2014. Society for Industrial and Applied Mathematics. ISBN 978-1-61197-338-9.

[16] Stephen Boyd and Almir Mutapcic. Subgradient methods. *Lecture notes of EE364b, Stanford University, Winter Quarter*, 2007, 2006.

[17] Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe, and Arash Hassibi. A tutorial on geometric programming. *Optimization and engineering*, 8(1):67–127, 2007.

[18] V. Chandrasekaran and P. Shah. Relative Entropy Relaxations for Signomial Optimization. *SIAM Journal on Optimization*, 26(2):1147–1173, January 2016. ISSN 1052-6234. doi: 10.1137/140988978.

[19] Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 199–208. ACM, 2009.

[20] Wei Chen, Chi Wang, and Yajun Wang. Scalable Influence Maximization for Prevalent Viral Marketing in Large-scale Social Networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 1029–1038, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0055-1. doi: 10.1145/1835804.1835934.

[21] Wei Chen, Tian Lin, Zihan Tan, Mingfei Zhao, and Xuren Zhou. Robust Influence Maximization. *arXiv:1601.06551 [cs]*, January 2016.

[22] Mung Chiang. Geometric Programming for Communication Systems. *Commun. Inf. Theory*, 2(1/2):1–154, July 2005. ISSN 1567-2190. doi: 10.1516/0100000005.

[23] Amol Deshpande, Lisa Hellerstein, and Devorah Kletenik. Approximation Algorithms for Stochastic Submodular Set Cover with Applications to Boolean Function Evaluation and Min-Knapsack. *ACM Trans. Algorithms*, 12(3):42:1–42:28, April 2016. ISSN 1549-6325. doi: 10.1145/2876506.

[24] Pedro Domingos and Matt Richardson. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66. ACM, 2001.

[25] Nan Du, Le Song, Manuel Gomez Rodriguez, and Hongyuan Zha. Scalable influence estimation in continuous-time diffusion networks. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3147–3155. Curran Associates, Inc., 2013. URL `http://papers.nips.cc/paper/4857-scalable-influence-estimation-in-continuous-time-diffusion-networks.pdf`.

[26] Nan Du, Yingyu Liang, Maria-Florina Balcan, and Le Song. Influence function learning in information diffusion networks. In *ICML*, pages 2016–2024, 2014.

[27] Joseph C Dunn and S Harshbarger. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.

[28] J. Ecker. Geometric Programming: Methods, Computations and Applications. *SIAM Review*, 22(3):338–362, July 1980. ISSN 0036-1445. doi: 10.1137/1022058.

[29] Alina Ene and Huy L. Nguyen. A Reduction for Optimizing Lattice Submodular Functions with Diminishing Returns. *arXiv:1606.08362 [cs]*, June 2016.

[30] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, March 1956. ISSN 1931-9193. doi: 10.1002/nav.3800030109.

[31] Satoru Fujishige. *Submodular Functions and Optimization*, volume 58. Elsevier, 2005.

[32] Satoru Fujishige, Takumi Hayashi, and Kiyohito Nagano. Minimizing Continuous Extensions of Discrete Convex Functions with Linear Inequality Constraints. *SIAM J. on Optimization*, 20(2):856–867, June 2009. ISSN 1052-6234. doi: 10.1137/080717675.

[33] Gagan Goel, Chinmay Karande, Pushkar Tripathi, and Lei Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 755–764. IEEE, 2009.

[34] Michel Goemans and Jan Vondrák. Stochastic Covering and Adaptivity. In *LATIN 2006: Theoretical Informatics*, pages 532–543. Springer Berlin Heidelberg, March 2006. doi: 10.1007/11682462_50.

[35] Daniel Golovin and Andreas Krause. Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization. *Journal of Artificial Intelligence*, 42:427–486, 2011.

[36] M Gomez Rodriguez, B Schölkopf, Langford J Pineau, et al. Influence maximization in continuous time diffusion networks. In *29th International Conference on Machine Learning (ICML 2012)*, pages 1–8. International Machine Learning Society, 2012.

[37] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 1019–1028, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0055-1. doi: 10.1145/1835804.1835933. URL `http://doi.acm.org/10.1145/1835804.1835933`.

[38] Corinna Gottschalk and Britta Peis. Submodular function maximization on the bounded integer lattice. In *Approximation and Online Algorithms: 13th International Workshop (WAOA)*, 2015.

[39] Avinatan Hassidim and Yaron Singer. Submodular optimization under noise. *arXiv preprint arXiv:1601.03095*, 2016.

[40] Daisuke Hatano, Takuro Fukunaga, Takanori Maehara, and Ken-ichi Kawarabayashi. Lagrangian Decomposition Algorithm for Allocating Marketing Channels. In *AAAI*, pages 1144–1150, 2015.

[41] Xinran He and David Kempe. Robust Influence Maximization. *arXiv:1602.05240 [physics]*, February 2016.

[42] Satoru Iwata and Kiyohito Nagano. Submodular function minimization under covering constraints. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 671–680. IEEE, 2009.

[43] Martin Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. pages 427–435, 2013.

[44] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the Spread of Influence Through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 137–146, New York, NY, USA, 2003. ACM. ISBN 978-1-58113-737-8. doi: 10.1145/956750.956769.

[45] Vladimir Kolmogorov and Akiyoshi Shioura. New algorithms for convex cost tension problem with application to computer vision. *Discrete Optimization*, 6:378–393, 2009.

[46] Andreas Krause, H Brendan McMahan, Carlos Guestrin, and Anupam Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9(Dec):2761–2801, 2008.

[47] Simon Lacoste-Julien. Convergence Rate of Frank-Wolfe for Non-Convex Objectives. *arXiv:1607.00345 [cs, math, stat]*, July 2016.

[48] Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of frank-wolfe optimization variants. In *Advances in Neural Information Processing Systems*, pages 496–504, 2015.

[49] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1049–1065. IEEE, 2015.

[50] Meghna Lowalekar, Pradeep Varakantham, and Akshat Kumar. Robust Influence Maximization: (Extended Abstract). In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, AAMAS '16, pages 1395–1396, Richland, SC, 2016. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-1-4503-4239-1.

[51] Takanori Maehara. Risk averse submodular utility maximization. *Operations Research Letters*, 43(5):526–529, September 2015. ISSN 0167-6377. doi: 10.1016/j.orl.2015.08.001.

[52] Takanori Maehara, Akihiro Yabe, and Ken-ichi Kawarabayashi. Budget Allocation Problem with Multiple Advertisers: A Game Theoretic View. pages 428–437, 2015.

[53] Kazuo Murota. *Discrete convex analysis*. SIAM, 2003.

[54] Kazuo Murota and Akiyoshi Shioura. Exact bounds for steepest descent algorithms of l-convex function minimization. *Operations Research Letters*, 42:361–366, 2014.

[55] Harikrishna Narasimhan, David C Parkes, and Yaron Singer. Learnability of influence in networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3186–3194. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5989-learnability-of-influence-in-networks.pdf`.

[56] Praneeth Netrapalli and Sujay Sanghavi. Learning the graph of epidemic cascades. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '12, pages 211–222, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1097-0. doi: 10.1145/2254756.2254783. URL `http://doi.acm.org/10.1145/2254756.2254783`.

[57] Evdokia Nikolova. Approximation algorithms for offline risk-averse combinatorial optimization. 2010.

[58] James B. Orlin, Andreas Schulz, and Rajan Udwani. Robust monotone submodular function maximization. In *Conference on Integer Programming and Combinatorial Optimization (IPCO)*, 2016.

[59] Luis D. Pascual and Adi Ben-Israel. Constrained maximization of posynomials by geometric programming. *Journal of Optimization Theory and Applications*, 5(2):73–80, March 1970. ISSN 0022-3239, 1573-2878. doi: 10.1007/BF00928296.

[60] Boris T. Polyak. *Introduction to Optimization*. Number 04; QA402. 5, P6. 1987.

[61] R Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

[62] R Tyrrell Rockafellar and Stanislav Uryasev. Conditional value-at-risk for general loss distributions. *Journal of banking & finance*, 26(7):1443–1471, 2002.

[63] Tasuku Soma and Yuichi Yoshida. A Generalization of Submodular Cover via the Diminishing Return Property on the Integer Lattice. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 847–855. Curran Associates, Inc., 2015.

[64] Tasuku Soma, Naonori Kakimura, Kazuhiro Inaba, and Ken-ichi Kawarabayashi. Optimal Budget Allocation: Theoretical Guarantee and Efficient Algorithm. pages 351–359, 2014.

[65] Zoya Svitkina and Lisa Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. *SIAM Journal on Computing*, 40(6):1715–1737, 2011.

[66] Kevin Wainwright and Alpha Chiang. *Fundamental Methods of Mathematical Economics.* McGraw-Hill Education, 2004. ISBN 0070109109.

[67] Peng Zhang, Wei Chen, Xiaoming Sun, Yajun Wang, and Jialin Zhang. Minimizing seed set selection with probabilistic coverage guarantee in a social network. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1306–1315, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623684. URL `http://doi.acm.org/10.1145/2623330.2623684`.

A Worst-Case Approximation Ratio versus True Worst-Case

Consider the function $f(x;\theta)$ defined on $\{0,1\} \times \{0,1\}$, with values given by:

$$f(x;0) = \begin{cases} 1 & x = 0 \\ 0.6 & x = 1, \end{cases} \qquad f(x;1) = \begin{cases} 1 & x = 0 \\ 2 & x = 1. \end{cases} \tag{22}$$

We wish to choose x to maximize $f(x;\theta)$ robustly with respect to adversarial choices of θ. If θ were fixed, we could directly choose x^*_θ to maximize $f(x;\theta)$. In particular, $x^*_0 = 0$ and $x^*_1 = 1$. Of course, we want to deal with worst-case θ. One option is to maximize the worst-case approximation ratio:

$$\max_x \min_\theta \frac{f(x;\theta)}{f(x^*_\theta;\theta)}. \tag{23}$$

One can verify that the best x according to this criterion is $x = 1$, with worst-case approximation ratio 0.6 and worst-case function value 0.6. In this paper, we optimize the worst-case of the actual function value:

$$\max_x \min_\theta f(x;\theta). \tag{24}$$

This criterion will select $x = 0$, which has a worse worst-case approximation ratio of 0.5, but actually guarantees a function value of 1, significantly better than the 0.6 achieved by the other formulation of robustness.

B Constrained Continuous Submodular Function Minimization

Define $\mathbb{R}^n_\downarrow$ to be the set of vectors ρ in $\mathbb{R}^n$ which are monotone nonincreasing, i.e. $\rho(1) \geq \rho(2) \geq \cdots \geq \rho(n)$. As in the main text, define $[k] = \{0, 1, \ldots, k-1\}$. One of the key results from [5] is that an arbitrary submodular function $H(x)$ defined on $\prod_{i=1}^n [k_i]$ can be extended to a particular convex function $h_\downarrow(\rho)$ so that

$$\begin{array}{ll} \text{minimize} & H(x) \\ \text{s.t.} & x \in \prod_{i=1}^n [k_i] \end{array} \Leftrightarrow \begin{array}{ll} \text{minimize} & h_\downarrow(\rho) \\ \text{s.t.} & \rho \in \prod_{i=1}^n \mathbb{R}^{k_i - 1}_\downarrow. \end{array} \tag{25}$$

Moreover, Theorem 4 from [5] states that, if a_{iy_i} are strictly convex functions for all $i = 1, \dots, n$ and each $y_i \in [k_i]$, then the two problems

$$\begin{array}{ll} \text{minimize} & H(x) + \sum_{i=1}^n \sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda) \\ \text{s.t.} & x \in \prod_{i=1}^n [k_i]. \end{array} \tag{26}$$

and

$$\begin{array}{ll} \text{minimize} & h_\downarrow(\rho) + \sum_{i=1}^n \sum_{x_i=1}^{k_i-1} a_{ix_i}[\rho_i(x_i)] \\ \text{s.t.} & \rho \in \prod_{i=1}^n \mathbb{R}_\downarrow^{k_i-1} \end{array} \tag{27}$$

are equivalent. In particular, one recovers a solution to Problem (26) for any λ just as alluded to in Lemma 3.2: find ρ^* which solves Problem (27) and, for each component i, choose x_i to be the maximal value for which $\rho_i^*(x_i) \geq \lambda$.

B.1 Proof of Lemma 3.2

Proof. The discretized form of the regularizer R^δ is also separable and can be written $R^\delta(x) = \sum_{i=1}^n R_i^\delta(x)$. For each $i = 1, \dots, n$ and each $y_i \in [k_i]$ with $y_i \geq 1$, define $a_{iy_i}(t) = \frac{1}{2}t^2 \cdot [R_i^\delta(y_i) - R_i^\delta(y_i - 1)]$, so that $a'_{iy_i}(t) = t \cdot [R_i^\delta(y_i) - R_i^\delta(y_i - 1)]$. Since we assumed $R(x)$ is strictly increasing, the coefficient of t^2 in each $a_{iy_i}(t)$ is strictly positive, so that each $a_{iy_i}(t)$ is strictly convex. Then,

$$\begin{aligned} \lambda R_i^\delta(x_i) &= \lambda \cdot \left[R_i^\delta(0) + \sum_{y_i=1}^{x_i} \Big(R_i^\delta(y_i) - R_i^\delta(y_i - 1) \Big) \right] && (28) \\ &= \lambda R_i^\delta(0) + \sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda), && (29) \end{aligned}$$

so that the discretized version of the minimization problem can be written as

$$\begin{array}{ll} \text{minimize} & H^\delta(x) + \lambda R^\delta(0) + \sum_{i=1}^n \sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda) \\ \text{s.t.} & x \in \prod_{i=1}^n [k_i]. \end{array} \tag{30}$$

Since the term $R^\delta(0)$ does not depend on the variable x, this minimization is equivalent to

$$\begin{array}{ll} \text{minimize} & H^\delta(x) + \sum_{i=1}^n \sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda) \\ \text{s.t.} & x \in \prod_{i=1}^n [k_i]. \end{array} \tag{31}$$

This problem is in the precise form where we can apply the preceding equivalence result between Problems (26) and (27), so we are done. □

B.2 Proof of Theorem 3.1

Proof. The general idea of this proof is to first show that the integer-valued point x_d^* which solves

$$x_d^* \in \underset{x \in \prod_{i=1}^n [k_i] : R^\delta(x) \leq B}{\operatorname{argmin}} H^\delta(x)$$

is also nearly a minimizer of the continuous version of the problem, due to the fineness of the discretization. Then, we show that the solutions traced out by $x(\lambda)$ get very close to x_d^*. These two results are simply combined via the triangle inequality.

B.2.1 Continuous and Discrete Problems

We begin by proving that

$$H^\delta(x_d^*) \leq G\delta + \min_{x\in\mathcal{X}:R(x)\leq B} H(x). \tag{32}$$

Consider $x^* \in \arg\min_{x\in\mathcal{X}:R(x)\leq B} H(x)$. If x^* corresponds to an integral point in the discretized domain, then $H(x^*) = H^\delta(x_d^*)$ and we are done. Else, x^* has at least one non-integral coordinate. By rounding coordinatewise, we can construct a set $X = \{x_1, \ldots, x_m\} \subseteq \prod_{i=1}^n [k_i]$ so that $x^* \in \mathrm{conv}(\{A(x_1), \ldots, A(x_m)\}$. By monotonicity, there must be some $x_i \in X$ with $R^\delta(x_i) \leq B$, i.e. $A(x_i)$ is feasible for the original continuous problem. By construction, since the discretization given by A is δ-fine, we must have $\|x^* - A(x_i)\|_\infty \leq \delta$. Applying the Lipschitz property of H and the optimality of x^*, we have

$$G\delta \geq H(A(x_i)) - H(x^*) = H^\delta(x_i) - H(x^*) \geq H^\delta(x_d^*) - H(x^*),$$

from which (32) follows.

B.2.2 Discrete and Parameterized Discrete Problems

Define λ_- and λ_+ by

$$\lambda_- \in \operatorname*{argmin}_{\lambda\geq 0:R^\delta(x(\lambda))\leq B} H^\delta(x(\lambda)) \quad \text{and}$$
$$\lambda_+ \in \operatorname*{argmax}_{\lambda\geq 0:R^\delta(x(\lambda))\geq B} H^\delta(x(\lambda)).$$

The next step in proving our suboptimality bound is to prove that

$$H^\delta(x(\lambda_+)) \leq H^\delta(x_d^*) \leq H^\delta(x(\lambda_-)), \tag{33}$$

from which it will follow that

$$H^\delta(x(\lambda_-)) \leq G\delta + H^\delta(x_d^*).$$

We begin by stating the min-max inequality, i.e. weak duality:

$$\begin{aligned}
\min_{x\in\prod_{i=1}^n[k_i]:R^\delta(x)\leq B} H^\delta(x) &= \min_{x\in\prod_{i=1}^n[k_i]} \max_{\lambda\geq 0} \left\{H^\delta(x) + \lambda(R^\delta(x) - B)\right\} && (34)\\
&\geq \max_{\lambda\geq 0} \min_{x\in\prod_{i=1}^n[k_i]} \left\{H^\delta(x) + \lambda(R^\delta(x) - B)\right\} && (35)\\
&= \max_{\lambda\geq 0} \left\{H^\delta(x(\lambda)) + \lambda(R^\delta(x(\lambda)) - B)\right\} && (36)\\
&\geq \max_{\lambda\geq 0:R^\delta(x(\lambda))\geq B} \left\{H^\delta(x(\lambda)) + \lambda(R^\delta(x(\lambda)) - B)\right\} && (37)\\
&\geq \max_{\lambda\geq 0:R^\delta(x(\lambda))\geq B} H^\delta(x(\lambda)) && (38)\\
&= H^\delta(x(\lambda_+)). && (39)
\end{aligned}$$

We can also bound the optimal value of $H^\delta(x_d^*)$ from the other side:

$$H^\delta(x_d^*) = \min_{x\in\prod_{i=1}^n[k_i]:R^\delta(x)\leq B} H^\delta(x) \leq \min_{\lambda\geq 0:R^\delta(x(\lambda))\leq B} H^\delta(x) = H^\delta(x(\lambda_-)) \tag{40}$$

because the set of $x(\lambda)$ parameterized by λ is a subset of the full set $\{x \in \prod_{i=1}^n [k_i] : R^\delta(x) \le B\}$.

We have now bounded the optimal value of $H^\delta(x_d^*)$ on either side by optimization problems where we seek an optimal $\lambda \ge 0$ for the parameterization $x(\lambda)$:

$$H^\delta(x(\lambda_+)) \le H^\delta(x_d^*) \le H^\delta(x(\lambda_-)). \tag{41}$$

Recall that $x(\lambda)$ comes from thresholding the values of ρ^* by λ, and that we assume that the elements of ρ^* are unique. Hence, as we increase λ, the components of x decrease by 1 each time. Combining this with the strict monotonicity of R, we see that $\|x(\lambda_+) - x(\lambda_-)\|_\infty \le 1$. By the Lipschitz properties of H^δ, it follows that $\left|H^\delta(x(\lambda_+)) - H^\delta(x(\lambda_-))\right| \le G\delta$. Since $H^\delta(x_d^*)$ lies in the interval between $H^\delta(x(\lambda_+))$ and $H^\delta(x(\lambda_-))$, it follows that $\left|H^\delta(x_d^*) - H^\delta(x(\lambda_-))\right| \le G\delta$. $\square$

B.3 Proof of Remark 3.1

Define $\lambda^* = \lambda_-$ as in the previous section, so that $x' = A(x(\lambda^*)$. The $x(\lambda_+)$ bound is a simple consequence from the above result that

$$H^\delta(x(\lambda_+)) \le H^\delta(x_d^*) \le H^\delta(x(\lambda_-)) = H(x').$$

As for the Lagrangian bound, since $x(\lambda^*)$ is a minimizer for the regularized function $H^\delta(x) + \lambda^*(R^\delta(x) - B)$, it follows that

$$H^\delta(x(\lambda^*)) + \lambda^*(R^\delta(x(\lambda^*)) - B) \le H^\delta(x_d^*) + \lambda^*(R^\delta(x_d^*) - B). \tag{42}$$

Rearranging, and observing that $R^\delta(x_d^*) \le B$ because x_d^* is feasible, it holds that

$$H(x') = H^\delta(x(\lambda^*)) \le H^\delta(x_d^*) + \lambda^*(R^\delta(x_d^*) - R^\delta(x(\lambda^*))) \le H^\delta(x_d^*) + \lambda^*(B - R(x')). \tag{43}$$

One can also combine either of these bounds with the result from the proof of Theorem 3.1 that $H^\delta(x_d^*) \le G\delta + H(x^*)$ yielding e.g.

$$H(x') \le G\delta + \lambda^*(B - R(x')) + H^\delta(x_d^*).$$

B.4 Solving the Optimization Problem

Now that we have proven equivalence results between the constrained problem we want to solve and the convex problem (27), we need to actually solve the convex problem. At the beginning of Section 5.2 in [5], it is stated that this surrogate problem can optimized via the Frank-Wolfe method and its variants, but only the the version of Problem (27) without the extra functions a_{ix_i} is elaborated upon. Here we detail how Frank-Wolfe algorithms can be used to solve the more general parametric regularized problem. Our aim is to spell out very clearly the applicability of Frank-Wolfe to this problem, for the ease of future practitioners.

Bach [5] notes that by duality, Problem (27) is equivalent to:

$$\begin{aligned}
\min_{\rho \in \prod_{i=1}^n \mathbb{R}_\downarrow^{k_i - 1}} h_\downarrow(\rho) - H(0) + \sum_{i=1}^n \sum_{x_i=1}^{k_i-1} a_{ix_i}[\rho_i(x_i)] &= \min_{\rho \in \prod_{i=1}^n \mathbb{R}_\downarrow^{k_i - 1}} \max_{w \in B(H)} \langle \rho, w \rangle + \sum_{i=1}^n \sum_{x_i=1}^{k_i-1} a_{ix_i}[\rho_i(x_i)] \\
&= \max_{w \in B(H)} \left\{ \min_{\rho \in \prod_{i=1}^n \mathbb{R}_\downarrow^{k_i - 1}} \langle \rho, w \rangle + \sum_{i=1}^n \sum_{x_i=1}^{k_i-1} a_{ix_i}[\rho_i(x_i)] \right\} \\
&:= \max_{w \in B(H)} f(w).
\end{aligned}$$

Here, the base polytope $B(H)$ happens to be the convex hull of all vectors w which could be output by the greedy algorithm in [5].

It is the dual problem, where we maximize over w, which is amenable to Frank-Wolfe. For Frank-Wolfe methods, we need two oracles: an oracle which, given w, returns $\nabla f(w)$; and an oracle which, given $\nabla f(w)$, produces a point s which solves the linear optimization problem $\max_{s\in B(H)}\langle s, \nabla f(w)\rangle$.

Per Bach [5], an optimizer of the linear problem can be computed directly from the greedy algorithm. For the gradient oracle, recall that we can find a subgradient of $g(x) = \min_y h(x, y)$ at the point x_0 by finding $y(x_0)$ which is optimal for the inner problem, and then computing $\nabla_x h(x, y(x_0))$. Moreover, if such $y(x_0)$ is the unique optimizer, then the resulting vector is indeed the *gradient* of $g(x)$ at x_0. Hence, in our case, it suffices to first find $\rho(w)$ which solves the inner problem, and then $\nabla f(w)$ is simply $\rho(w)$ because the inner function is linear in w. Since each function a_{ix_i} is strictly convex, the minimizer $\rho(w)$ is unique, confirming that we indeed get a gradient of f, and that f is differentiable.

Of course, we still need to compute the minimizer $\rho(w)$. For a given w, we want to solve

$$\min_{\rho\in\prod_{i=1}^n \mathbb{R}_{\downarrow}^{k_i-1}} \langle\rho, w\rangle + \sum_{i=1}^{n}\sum_{x_i=1}^{k_i-1} a_{ix_i}[\rho_i(x_i)]$$

There are no constraints coupling the vectors ρ_i, and the objective is similarly separable, so we can independently solve n problems of the form

$$\min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \langle\rho, w\rangle + \sum_{j=1}^{k-1} a_j(\rho_j).$$

Recall that each function $a_{iy_i}(t)$ takes the form $\frac{1}{2}t^2 r_{iy_i}$ for some $r_{iy_i} > 0$. Let $D = \text{diag}(r)$, the $(k-1)\times(k-1)$ matrix with diagonal entries r_j. Our problem can then be written as

$$\begin{aligned}\min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \langle\rho, w\rangle + \frac{1}{2}\sum_{j=1}^{k-1} r_j\rho_j^2 &= \min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \langle\rho, w\rangle + \frac{1}{2}\langle D\rho,\ \rho\rangle \\ &= \min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \langle D^{1/2}\rho,\ D^{-1/2}w\rangle + \frac{1}{2}\langle D^{1/2}\rho,\ D^{1/2}\rho\rangle.\end{aligned}$$

Completing the square, the above problem is equivalent to

$$\begin{aligned}\min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \|D^{1/2}\rho + D^{-1/2}w\|_2^2 &= \min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \sum_{j=1}^{k-1}(r_j^{1/2}\rho_j + r_j^{-1/2}w_j)^2 \\ &= \min_{\rho\in\mathbb{R}_{\downarrow}^{k-1}} \sum_{j=1}^{k-1} r_j(\rho_j + r_j^{-1}w_j)^2.\end{aligned}$$

This last expression is precisely the problem which is called weighted isotonic regression: we are fitting ρ to $\text{diag}(r^{-1})w$, with weights r, subject to a monotonicity constraint. Weighted isotonic regression is solved efficiently via the Pool Adjacent Violators algorithm of [12].

C Expectation and Variance of the Influence Function

We wish to study the influence $\mathcal{I}(X, y)$, its expectation and its variance as a function of y. By definition, the influence function is given by

$$\mathcal{I}(X, y) = \sum_{t \in T} \left(1 - \prod_{(s,t) \in E} X_{st}^{y(s)} \right). \tag{44}$$

Before we prove the stated results, we will simplify the functions involved.

Maximizing $\mathcal{I}(X, y)$ is equivalent to minimizing the function

$$\sum_{t \in T} \prod_{(s,t) \in E} X_{st}^{y(s)} \tag{45}$$

and vice-versa. The particular properties we are interested in, namely convexity and submodularity, are preserved under sums. Moreover, expectation is linear and variances add, so for our purposes we can focus on only one term of the above sum. After reindexing in terms of $i = 1, \ldots, n$ instead of $(s, t) \in E$, we are left studying functions of the form

$$f(y) = \prod_{i=1}^{n} X_i^{y_i}. \tag{46}$$

If $f(y)$ is always convex (or supermodular), then $\mathcal{I}(X, y)$ is always concave (submodular) in y, and similarly for their expectations and variances.

Expectation By independence,

$$\mathbb{E}[f(y)] = \prod_{i=1}^{n} \mathbb{E}[X_i^{y_i}]. \tag{47}$$

Suppose that each $X_i \sim \mathrm{Beta}(\alpha_i, \beta_i)$, so that

$$\begin{aligned} \mathbb{E}[X_i^{y_i}] &= \frac{\Gamma(\alpha_i + \beta_i)\Gamma(\alpha_i + y_i)}{\Gamma(\alpha_i + \beta_i + y_i)} && (48) \\ &= \frac{\Gamma(\alpha_i + \beta_i)\Gamma(\alpha_i + y_i)\Gamma(\beta_i)}{\Gamma(\alpha_i + \beta_i + y_i)\Gamma(\beta_i)} && (49) \\ &= \frac{B(\alpha_i + y_i, \beta_i)}{B(\alpha_i, \beta_i)}. && (50) \end{aligned}$$

Then,

$$\mathbb{E}[f(y)] = \prod_{i=1}^{n} \frac{B(\alpha_i + y_i, \beta_i)}{B(\alpha_i, \beta_i)} \propto \prod_{i=1}^{n} B(\alpha_i + y_i, \beta_i), \tag{51}$$

where by $\propto$ we mean that the product of the denominators is a positive constant, dependent on the problem data but independent of y.

Variance The variance of $f(y)$ can be written as

$$\operatorname{Var}\left[\prod_{i=1}^{n} X_i^{y_i}\right] = \mathbb{E}\left[\prod_{i=1}^{n} X_i^{2y_i}\right] - \mathbb{E}\left[\prod_{i=1}^{n} X_i^{y_i}\right]^2 \tag{52}$$

$$= \prod_{i=1}^{n} \mathbb{E}\left[X_i^{2y_i}\right] - \prod_{i=1}^{n} \mathbb{E}\left[X_i^{y_i}\right]^2 \tag{53}$$

$$= \prod_{i=1}^{n} \frac{B(\alpha_i + 2y_i, \beta_i)}{B(\alpha_i, \beta_i)} - \prod_{i=1}^{n} \left(\frac{B(\alpha_i + y_i, \beta_i)}{B(\alpha_i, \beta_i)}\right)^2. \tag{54}$$

C.1 Gradient of Expected Influence

Recall the identity

$$\frac{\partial}{\partial a} B(a,b) = B(a,b)(\psi(a) - \psi(a+b)), \tag{55}$$

where ψ is the digamma function. We can then compute each component of the gradient of $\mathbb{E}[f(y)]$:

$$\frac{\partial}{\partial y_i}\left(\mathbb{E}[f(y)]\right) = \prod_{i=1}^{n} \frac{1}{B(\alpha_i, \beta_i)} \cdot \prod_{j \neq i} B(\alpha_j + y_j, \beta_j) \cdot \frac{\partial}{\partial y_i}\left(B(\alpha_i + y_i, \beta_i)\right) \tag{56}$$

$$= \prod_{i=1}^{n} \frac{1}{B(\alpha_i, \beta_i)} \cdot \prod_{j \neq i} B(\alpha_j + y_j, \beta_j) \cdot B(\alpha_i + y_i, \beta_i) \cdot (\psi(\alpha_i + y_i) - \psi(\alpha_i + y_i + \beta_i)) \tag{57}$$

$$= \mathbb{E}[f(y)] \cdot (\psi(\alpha_i + y_i) - \psi(\alpha_i + y_i + \beta_i)). \tag{58}$$

C.2 Counterexample for Fact 2.1

We give a specific choice of parameters n, α_i, β_i and y_i for which the resulting function $\sqrt{\operatorname{Var}(f(y))}$ is non-convex, non-concave, non-submodular and non-supermodular for various points $y \in \mathbb{R}_+^n$. For the case $T = 1$, the function $1 - f(y)$ is a valid influence function, so we have a valid counterexample for $\sqrt{\operatorname{Var}(\mathcal{I}(X, y))}$.

Consider the case $n = 2$, with $\alpha_1 = \alpha_2 = 1$ and $\beta_1 = \beta_2 = 1$. This corresponds to the Budget Allocation problem where we have two sources each with an edge to one customer, and we have only our prior (i.e. no data) on either of the edge probabilities. Using equation (54), we can directly compute the Hessian of $\sqrt{\operatorname{Var}(f(y))}$ at any point y, e.g. using Mathematica. In particular, for $y_1 = y_2 = 1$, the Hessian has a positive and a negative eigenvalue, so $\sqrt{\operatorname{Var}(f(y))}$ is neither convex nor concave at this point. Also for $y_1 = y_2 = 1$, the off-diagonal element is negative, so $\sqrt{\operatorname{Var}(f(y))}$ is not supermodular over all of $\mathbb{R}_+^2$. However, for $y_1 = y_2 = 3$, the off-diagonal element is positive, so our function is also not submodular.