Robust Budget Allocation Via Continuous Submodular Functions

Citation: Staib, Matthew and Stefanie Jegelka. "Robust Budget Allocation Via Continuous Submodular Functions." Applied Mathematics & Optimization 82, 3 (March 2019): 1049–1079 © 2019 Springer Science Business Media, LLC, part of Springer Nature

As Published: https://doi.org/10.1007/s00245-019-09567-0

Publisher: Springer Science and Business Media LLC

Persistent URL: https://hdl.handle.net/1721.1/130073

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Robust Budget Allocation Via Continuous Submodular Functions

Cite this article as: Matthew Staib and Stefanie Jegelka, Robust Budget Allocation Via Continuous Submodular Functions, Applied Mathematics & Optimization https://doi.org/10.1007/s00245-019-09567-0

Noname manuscript No.
(will be inserted by the editor)

Robust Budget Allocation via Continuous Submodular Functions

Matthew Staib · **Stefanie Jegelka**

Received: date / Accepted: date

Abstract The optimal allocation of resources for maximizing influence, spread of information or coverage, has gained attention in the past years, in particular in machine learning and data mining. But in applications, the parameters of the problem are rarely known exactly, and using wrong parameters can lead to undesirable outcomes. 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 (or game). We show that this nonconvex problem can be solved exactly by leveraging connections to continuous submodular functions, and by solving a constrained submodular minimization problem. Although constrained submodular minimization is hard in general, here, we establish conditions under which such a problem can be solved to arbitrary precision ε.

1 Introduction

The optimal allocation of resources for maximizing influence, spread of information or coverage, has gained attention in the past few years, in particular in machine learning and data mining [25,44,22,37,16].

Formally, 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 $\mathcal{I}(y)$. Each edge $(s,t) \in E$ between channel s and person t is weighted with a probability p_{st} that, e.g., an advertisement on radio station s will influence person t to buy some product. The budget $y(s)$ controls how many independent attempts are

M. Staib
E-mail: mstaib@mit.edu

S. Jegelka
E-mail: stefje@csail.mit.edu

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

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

and hence the expected number of influenced people is $\mathcal{I}(y) = \sum_{t\in T} I_t(y)$. We 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 [14].

Since its introduction in [3], several works have extended the formulation of Budget Allocation and provided algorithms [14,40,54,67,68]. Budget Allocation may also be viewed as influence maximization on a bipartite graph, where information spreads as in the Independent Cascade model. For integer y, Budget Allocation and Influence Maximization are NP-hard. Yet, constant-factor approximations are possible, and build on the fact that the influence function is submodular in the binary case, and *DR-submodular* in the integer case [67,40]. If y is continuous, the problem is a concave maximization problem.

The formulation of Budget Allocation assumes that the transmission probabilities are known exactly. But this is rarely true in practice. Typically, the probabilities p_{st}, and possibly the graph itself, must be inferred from observations [36,27,59,26,60]. In Section 6 we will see that a misspecification or point estimate of parameters p_{st} can lead to much reduced outcomes. A more realistic assumption is to know *confidence intervals* for the p_{st}. Realizing this severe 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 [41,20, 52]. The resulting optimization problem is hard but admits bicriteria approximations.

In this work, we revisit Budget Allocation under uncertainty from the perspective of robust optimization [11,9]. 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 pitfalls of the approximation ratio formulation (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:

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

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

The new, key insight we exploit in this work is that $\mathcal{I}(y;p)$ has the property of *continuous submodularity* in p – in contrast to previously exploited submodular maximization in y – and can hence be minimized by generalizing techniques from discrete submodular optimization [5]. The techniques in [5], however, are restricted to box constraints, and do not directly apply to our confidence sets. In fact, general constrained submodular minimization is hard [69,33,42]. We make the following contributions:

1. We provide the first results for continuous submodular minimization with box constraints and one more "nice" constraint, and checkable conditions under which

the algorithm is guaranteed to return a global optimum. In other words, we have a provable algorithm for a new class of constrained nonconvex minimization problems that should be of interest more broadly.
2. Leveraging the above result, we present an algorithm with optimality bounds for Robust Budget Allocation in the nonconvex adversarial scenario (2).

1.1 Background and Related Work

We begin with some background material and, along the way, discuss 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 $F(S)+F(T) \geq F(S \cap T)+F(S \cup T)$. If the reverse inequality holds, then the function is *supermodular*. 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

$$f(x)+f(y) \geq f(x \vee y)+f(x \wedge y). \tag{3}$$

For the integer lattice and vectors x,y, $x \vee y$ denotes the coordinate-wise maximum and $x \wedge y$ the coordinate-wise minimum. Submodularity has also been considered on continuous domains $\mathcal{X} \subset \mathbb{R}^d$, where, if f is also twice-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 $i \neq j$ [70, Theorem 3.2]. These functions may be convex, concave, or neither.

Submodular functions on lattices can be minimized by a reduction to set functions, more precisely, ring families [15]. Combinatorial algorithms for submodular optimization on lattices are discussed in [45]. More recently, [5] extended results based on the convex Lovász extension, by building on connections to optimal transport. The subclass of $L^\natural$-convex functions admits strongly polynomial time minimization [56,47,57], but does not apply in our setting.

Similarly, results for submodular maximization extend to integer lattices, e.g. [38]. 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 $f(x+e_i)-f(x) \geq f(y+e_i)-f(y)$. For such *DR-submodular* functions, many approximation results for the set function case extend [14,68,67]. In particular, [30] show a generic reduction to set function optimization that they apply to maximization. In fact, it also applies to minimization:

Proposition 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 for evaluating f. Here, $[k_i] = \{0,1,\ldots,k_i-1\}$.*

Proof The function f can be reduced to a submodular set function $g : 2^V \to \mathbb{R}$ via [30], where $|V| = O(n \log k)$. The function g can be evaluated via mapping from 2^V to the domain of f, and then evaluating f, in time $O(n \log k \cdot EO)$. We can directly substitute these complexities into the runtime bound from [51].

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

1.1.2 Related Problems

A sister problem of Budget Allocation is *Influence Maximization* on general graphs, where a set of seed nodes is selected to start a propagation process. The influence function is still monotone submodular and amenable to the greedy algorithm [44], but it cannot be evaluated explicitly and requires approximation [21]. *Stochastic Coverage* [34] is a version of Set Cover where the covering sets $S_i \subset V$ are random. A variant of Budget Allocation can be written as stochastic coverage with multiplicity. Stochastic Coverage has mainly been studied in the online or adaptive setting, where logarithmic approximation factors can be achieved [35, 24, 2].

Our objective function (2) is a *signomial* in p, i.e., a linear combination of monomials of the form $\prod_i x_i^{c_i}$. General signomial optimization is NP-hard [23], but certain subclasses are tractable: *posynomials* with all nonnegative coefficients can be minimized via Geometric Programming [17], and signomials with a single negative coefficient admit sum of squares-like relaxations [19]. Our problem, a constrained posynomial maximization, is not in general a geometric program. Some work addresses this setting via monomial approximation [63, 29], but, to our knowledge, our algorithm is the first that solves this problem to arbitrary accuracy.

1.1.3 Robust Optimization

Two prominent strategies of addressing uncertainty in parameters of optimization problems are stochastic and robust optimization. If the distribution of the parameters is known (stochastic optimization), formulations such as value-at-risk (VaR) and conditional value-at-risk (CVaR) [65, 66] apply. In contrast, robust optimization [9, 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:

$$\min_y \sup_{u,A,b \in \mathcal{U}} \{g(y;u) \text{ s.t. } Ay \leq b\}. \tag{4}$$

Here, we will only have uncertainty in the cost function.

In this paper we are principally concerned with robust maximization of the continuous influence function $\mathcal{I}(y)$, but mention some results for the discrete case. While there exist results for robust and CVaR optimization of modular (linear) functions [61, 12], submodular objectives do not in general admit such optimization [53], but variants and relaxations admit approximations [74, 72]. The brittleness of submodular optimization under noise has been studied in [6, 7, 39].

Approximations for robust submodular and influence optimization have been studied in [48,41,20,52], where an adversary can pick among a *finite* set of objective functions or remove selected elements [62].

2 Robust and Stochastic Budget Allocation

The unknown parameters in Budget Allocation are the transmission probabilities p_{st} or edge weights in a graph. If these are estimated from data, we may have posterior distributions or, a weaker assumption, confidence sets for the parameters. For ease of notation, we will work with the failure probabilities $x_{st} = 1 - p_{st}$ instead of the p_{st} directly, and write $\mathcal{I}(y;x)$ instead of $\mathcal{I}(y;p)$.

2.1 Stochastic Optimization

If a (posterior) distribution of the parameters is known, a simple strategy is to use expectations. For example, we can place a uniform prior on x_{st}, and observe n_{st} independent observations drawn from $\text{Ber}(x_{st})$. If we observe α_{st} failures and β_{st} successes, the resulting posterior distribution on the variable X_{st} is $\text{Beta}(1+\alpha_{st}, 1+\beta_{st})$. Given such a posterior, we may optimize

$$\max_{y\in\mathcal{Y}} \mathcal{I}(y;\mathbb{E}[X]) \qquad (5) \qquad \text{or} \qquad \max_{y\in\mathcal{Y}} \mathbb{E}[\mathcal{I}(y;X)]. \qquad (6)$$

Proposition 2 *Problems* (5) *and* (6) *are concave maximization problems over the (convex) set* $\mathcal{Y}$ *and can be solved exactly.*

Concavity of (6) follows since it is an expectation over concave functions, and it can be solved by stochastic gradient ascent or by explicitly computing gradients.

Merely maximizing expectation does not explicitly account for volatility and hence risk. One option is to penalize variance [10,11,4]:

$$\min_{y\in\mathcal{Y}} -\mathbb{E}[\mathcal{I}(y;X)] + \varepsilon\sqrt{\text{Var}(\mathcal{I}(y;X))}, \qquad (7)$$

but in our case this CVaR formulation seems difficult.

Fact 1 For y in the nonnegative orthant, the term $\sqrt{\text{Var}(\mathcal{I}(y;X))}$ need not be convex or concave, and need not be submodular or supermodular.

This observation does not rule out a solution, but the apparent difficulties further motivate a robust formulation that, as we will see, is amenable to optimization.

2.2 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 from assumed bounds. Formally, we solve

$$\max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \mathcal{I}(y;x), \tag{8}$$

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 use uncertainty sets $\mathcal{X} = \{x \in \text{Box}(l,u) : R(x) \leq B\}$, 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, if $[l_{st}, u_{st}] = [0,1]$, they enforce 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 4 applies to both.

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

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

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

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

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

D-norm uncertainty. The D-norm uncertainty set is similar to an ℓ_1-ball around $\hat{x}$, and is defined as

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

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

$$\mathcal{X}^D(\gamma) = \Big\{x \in \text{Box}(\hat{x}, u) : \sum_{(s,t)\in E} R_{st}(x_{st}) \leq \gamma\Big\},$$

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 when increasing x_{st}.

The min-max formulation $\max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{I}(y;x)$ has several benefits: the model is not tied to a specific learning algorithm for the probabilities x as long as we can choose a suitable confidence set. Moreover, this formulation allows to fully hedge against a worst-case scenario.

Algorithm 1 Subgradient Ascent

Input: suboptimality tolerance $\varepsilon > 0$, initial feasible budget $y^{(0)} \in \mathcal{Y}$
Output: ε-optimal budget y for Problem (8)
repeat
  $x^{(k)} \leftarrow \arg\min_{x\in\mathcal{X}} \mathcal{I}(y^{(k)};x)$ ▷ Find worst-case x for $y^{(k)}$
  $g^{(k)} \leftarrow \nabla_y \mathcal{I}(y^{(k)};x^{(k)})$ ▷ Gradient with respect to y of $\min_{x\in\mathcal{X}} \mathcal{I}(y;x)$ at $y = y^{(k)}$
  $L^{(k)} \leftarrow \mathcal{I}(y^{(k)};x^{(k)})$ ▷ Lower bound on optimal value
  $U^{(k)} \leftarrow \max_{y\in\mathcal{Y}} \mathcal{I}(y;x^{(k)})$ ▷ Upper bound on optimal value
  $\gamma^{(k)} \leftarrow (U^{(k)} - L^{(k)})/\|g^{(k)}\|_2^2$ ▷ Polyak's stepsize rule
  $y^{(k+1)} \leftarrow \mathrm{proj}_{\mathcal{Y}}(y^{(k)} + \gamma^{(k)} g^{(k)})$
  $k \leftarrow k+1$
until $U^{(k)} - L^{(k)} \leq \varepsilon$

3 Robust Budget Allocation: Main Ideas

Next, we address in two main steps how to solve Problem (8), first the outer and then the inner optimization problem. As noted above, the function $\mathcal{I}(y;x)$ is concave as a function of y for fixed x. As a pointwise minimum of concave functions, $F(y) := \min_{x\in\mathcal{X}} \mathcal{I}(y;x)$ is concave. Hence, if we 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 y is given by the gradient of $\mathcal{I}(y;x^*)$ for the minimizing $x^* \in \arg\min_{x\in\mathcal{X}} \mathcal{I}(y;x)$, i.e., $g_y = \nabla_y \mathcal{I}(y;x^*)$. Hence, we must be able to compute x^* for any y. We also obtain a duality gap: for any x', y' we have

$$\min_{x\in\mathcal{X}} \mathcal{I}(y';x) \leq \max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{I}(y;x) \leq \max_{y\in\mathcal{Y}} \mathcal{I}(y;x'). \tag{9}$$

This means we can estimate the optimal value $\mathcal{I}^*$ and use it in Polyak's stepsize rule for the subgradient method [64].

What remains to be addressed is how to compute x^*. $\mathcal{I}(y;x)$ is not convex in x, and not even quasiconvex. For example, standard methods [71, Chapter 12] imply that $f(x_1,x_2,x_3) = 1 - x_1x_2 - \sqrt{x_3}$ is not quasiconvex on $\mathbb{R}^3_+$. Moreover, the above-mentioned signomial optimization techniques do not apply for an exact solution either. So, it is not immediately clear that we can solve the inner optimization problem.

The key insight we will be using is that $\mathcal{I}(y;x)$ has a different beneficial property: while not convex, $\mathcal{I}(y;x)$ as a function of x is *continuous submodular*.

Lemma 1 *Suppose we have $n \geq 1$ differentiable functions $f_i : \mathbb{R} \to \mathbb{R}_+$, for $i = 1,\ldots,n$, 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 For $n = 1$, the resulting function is modular and therefore supermodular. In the case $n \geq 2$, we simply need to compute derivatives. The mixed derivatives are

$$\frac{\partial f}{\partial x_i \partial x_j} = f_i'(x_i) f_j'(x_j) \cdot \prod_{k\neq i,j} f_k(x_k). \tag{10}$$

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 by Theorem 3.2 of [70].

Corollary 1 *The influence function $\mathcal{I}(y;x)$ defined in Section 2 is continuous submodular in x over the nonnegative orthant, for each $y \geq 0$.*

Proof Since submodularity is preserved under summation, it suffices to show that each function $I_t(y)$ is continuous submodular. By Lemma 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 hence continuous submodular.

We further conjecture that the functions $\mathcal{I}(y;x)$ enjoy another beneficial property, beyond submodularity:

Conjecture 1 Strong duality holds, i.e.,

$$\max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{I}(y;x) = \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{I}(y;x). \tag{11}$$

If strong duality holds, then the duality gap $\max_{y\in\mathcal{Y}} \mathcal{I}(y;x^*) - \min_{x\in\mathcal{X}} \mathcal{I}(y^*;x)$ in Equation (9) is zero at optimality. If $\mathcal{I}(y;x)$ were quasiconvex in x, strong duality would hold by Sion's min-max theorem, but this is not the case. In practice, we observe that the duality gap always converges to zero.

We have seen the functions $\mathcal{I}(y;x)$ enjoy nice structural properties, but it is still not clear how to solve the inner problem. 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 any previous 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.

4 Constrained Continuous Submodular Function Minimization

The previous section shows that, to solve Robust Budget Allocation, we need an algorithm for minimizing a monotone continuous submodular function $H(x)$ subject to box constraints $x \in \text{Box}(l,u)$ and a constraint $R(x) \leq B$:

$$\begin{aligned} &\text{minimize } H(x) \\ &\text{s.t.} \quad\ R(x) \leq B \\ &\qquad\quad x \in \text{Box}(l,u). \end{aligned} \tag{12}$$

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

$$\begin{aligned} &\text{minimize } H(x) + \lambda^* R(x) \\ &\text{s.t.} \quad\ x \in \text{Box}(l,u). \end{aligned} \tag{13}$$

Although H and R are not necessarily convex here, it turns out that a similar approach indeed applies. The property of submodularity then enables a special relaxation that

allows one to obtain a solution for all possible values of λ via a single convex optimization problem. The main idea of our approach bears similarity with [58] for the set function case, but our setting with continuous functions and various uncertainty sets is more general, and requires more argumentation. We present our theoretical results here, and defer implementation details to the appendix.

4.1 Forming an Equivalent Convex Problem

Following [5], we discretize the problem; for a sufficiently fine discretization, we will achieve arbitrary accuracy. This discretization will in turn lead to a convex relaxation. Let A be an *interpolation mapping* that maps the discrete set $\prod_{i=1}^{n}[k_i]$ into $\mathrm{Box}(l,u) = \prod_{i=1}^{n}[l_i,u_i]$ 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 $x_i \in \{0,1,\ldots,k_i-2\}$. We will further say the full interpolation function A is δ-fine if each A_i is δ-fine.

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 submodular on the integer lattice. This construction reduces Problem (13) to a submodular minimization problem over the integer lattice:

$$\begin{array}{ll} \text{minimize } H^\delta(x) + \lambda R^\delta(x) \\ \text{s.t.} \qquad x \in \prod_{i=1}^{n}[k_i]. \end{array} \tag{14}$$

Motivated by convex optimization, one may hope that there exists a λ whose associated minimizer $x(\lambda)$ yields a nearly optimal solution for the corresponding constrained Problem (12) in the lattice case, where H^δ and R^δ replace H and R. . Theorem 2 below states that, under a condition, this is indeed the case. Moreover, a second benefit of submodularity is that we can find the entire solution path for Problem (14) by solving a single optimization problem.

Lemma 2 *Suppose H is continuous submodular, and suppose the regularizer R is strictly increasing and separable: $R(x) = \sum_{i=1}^{n} R_i(x_i)$. Then we can recover a minimizer $x(\lambda)$ for the induced discrete Problem* (14) *for any $\lambda \in \mathbb{R}$ by solving a single convex optimization problem.*

To formally prove Lemma 2, we need to go into more detail. The convex optimization problem arises from a relaxation $h_\downarrow$ that is an analogue of the *Lovász extension* of set functions to continuous submodular functions [5]. The basic idea for the extension $h_\downarrow$ is: instead of fixing a value for each coordinate of x, we give a distribution over values, and $h_\downarrow$ is the expected function value under that distribution. As a corollary, $h_\downarrow$ coincides with H^δ on lattice points.

Instead of specifying a full joint distribution over all coordinates, we will only need to give coordinatewise marginals μ_i. It is also convenient to represent the distributions μ_i via their (reversed) cumulative distributions functions ρ_i. The best joint distribution follows directly from these marginals: it is the solution to a multi-marginal optimal transport problem between the marginals, where the transport cost is the original submodular function H or H^δ. Formally $h_\downarrow$ can be defined as:

Definition 1 ([5]) Write $\mathcal{X} = \prod_{i=1}^n \mathcal{X}_i$, and let $H : \mathcal{X} \to \mathbb{R}$ be a submodular function (discrete or continuous). We define the *generalized Lovász extension* of H by:

$$h_\downarrow(\rho_1, \ldots, \rho_n) = h_\downarrow(\mu_1, \ldots, \mu_n) := \inf_{\gamma \in \mathcal{P}(\mathcal{X}, \{\mu_i\})} \int_{\mathcal{X}} H(x)\, d\gamma(x) \tag{15}$$

where $\mathcal{P}(\mathcal{X}, \{\mu_i\})$ is the set of measures γ whose marginals match the μ_i, for all coordinates i.

Importantly, $h_\downarrow$ is convex if and only if H is submodular [5, Theorem 1]. This makes optimizing $h_\downarrow$ tractable. To prove Lemma 2 we will use a specific correspondence between a discrete submodular function H^δ and its extension $h_\downarrow$:

Theorem 1 (Theorem 4 from [5]) *Let $H^\delta : \prod_{i=1}^n [k_i] \to \mathbb{R}$ be a submodular function with generalized Lovász extension $h_\downarrow$. Also let a_{iy_i} be strictly convex functions for all $i = 1, \ldots, n$ and each $y_i \in [k_i]$. The set $\mathbb{R}^k_\downarrow$ refers to the set of ordered vectors $z \in \mathbb{R}^k$ that satisfy $z_1 \geq z_2 \geq \cdots \geq z_k$, and the notation $\rho_i(x_i)$ denotes the x_i-th coordinate of the vector ρ_i. The vector ρ_i should still be understood as a discrete reverse cumulative distribution function, as stated earlier. For convenience we also write $\rho = \rho_1, \ldots, \rho_n$. Then the two problems*

$$\begin{aligned} &\text{minimize } H^\delta(x) + \textstyle\sum_{i=1}^n \sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda) \\ &\text{s.t.} \qquad\quad x \in \textstyle\prod_{i=1}^n [k_i]. \end{aligned} \tag{16}$$

and

$$\begin{aligned} &\text{minimize } h_\downarrow(\rho) + \textstyle\sum_{i=1}^n \sum_{x_i=1}^{k_i - 1} a_{ix_i}[\rho_i(x_i)] \\ &\text{s.t.} \qquad\quad \rho \in \textstyle\prod_{i=1}^n \mathbb{R}^{k_i - 1}_\downarrow \end{aligned} \tag{17}$$

are equivalent. Specifically, one recovers a solution to Problem (16) *for any λ: find ρ^* which solves Problem* (17) *and, for each component i, choose x_i to be the maximal value for which $\rho_i^*(x_i) \geq \lambda$.*

With Theorem 1 in hand, we are finally ready to prove Lemma 2. Our high-level strategy is to convert Problem (14) into the form of Problem (16). Per Theorem 1, we can solve Problem (16) and hence Problem (14) simultaneously for all λ, simply by solving the single convex Problem (17).

Proof (Lemma 2) 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 $i = 1, \ldots, n$ 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,

$$\lambda R_i^\delta(x_i) = \lambda \cdot \left[R_i^\delta(0) + \sum_{y_i=1}^{x_i} \left(R_i^\delta(y_i) - R_i^\delta(y_i - 1) \right) \right] \tag{18}$$

$$= \lambda R_i^\delta(0) + \sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda), \tag{19}$$

so that the discretized version of the minimization problem (14) can be written as

$$\begin{array}{ll}\text{minimize} & H^\delta(x)+\lambda R^\delta(0)+\sum_{i=1}^n\sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda)\\ \text{s.t.} & x\in\prod_{i=1}^n[k_i].\end{array} \tag{20}$$

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

$$\begin{array}{ll}\text{minimize} & H^\delta(x)+\sum_{i=1}^n\sum_{y_i=1}^{x_i} a'_{iy_i}(\lambda)\\ \text{s.t.} & x\in\prod_{i=1}^n[k_i].\end{array} \tag{21}$$

This problem is in the precise form where we can apply Theorem 1 to show equivalence between Problems (16) and (17), so we are done. □

Problem (17) can be solved by Frank-Wolfe methods [31,28,49,43]. This is because the greedy algorithm for computing subgradients of the Lovász extension can be generalized, and yields a linear optimization oracle for the dual of Problem (17). We detail the relationship between Problems (14) and (17), as well as how to implement the Frank-Wolfe methods, in Appendix C.1.

4.2 Bounding Solution Quality for the Constrained Problem

We now have a tractable convex formulation, Equation (17), of the *regularized* problem. But it is not yet clear if we can also recover a good solution to the original *constrained* problem.

Let ρ^* be the optimal solution for Problem (17). For any λ, we obtain a rounded solution $x(\lambda)$ for Problem (14) by thresholding: we set $x(\lambda)_i = \max\{j \mid 1 \le j \le k_i-1,\ \rho_i^*(j)\ge\lambda\}$, or zero if $\rho_i^*(j)<\lambda$ for all j. Each $x(\lambda')$ is the optimal solution for Problem (14) with $\lambda=\lambda'$. We use the largest parameterized solution $x(\lambda)$ that is still feasible, i.e. the solution $x(\lambda^*)$ where λ^* solves

$$\begin{array}{ll}\min & H^\delta(x(\lambda))\\ \text{s.t.} & \lambda\ge 0\\ & R^\delta(x(\lambda))\le B.\end{array} \tag{22}$$

This λ^* can be found efficiently via binary search or a linear scan.

Theorem 2 *Let H be continuous submodular and monotone decreasing, with ℓ_∞-Lipschitz constant G, and let R be strictly increasing and separable. Assume all entries $\rho_i^*(j)$ of the optimal solution ρ^* of Problem* (17) *are distinct. Let $x'=A(x(\lambda^*))$ be the thresholding corresponding to the optimal solution λ^* of Problem* (22)*, mapped back into the original continuous domain $\mathcal{X}$. Then x' is feasible for the continuous Problem* (12)*, and is a $2G\delta$-approximate solution:*

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

Theorem 2 implies an algorithm for solving Problem (12) to ε-optimality: (1) set $\delta=\varepsilon/G$, (2) compute ρ^* that solves Problem (17), (3) find the optimal thresholding of ρ^* by determining the smallest λ^* for which $R^\delta(x(\lambda^*))\le B$, and (4) map $x(\lambda^*)$ back into continuous space via the interpolation mapping A.

Proof (Theorem 2) The general idea of this proof is to first show that the best integer-valued point x_d^* that solves

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

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.

We begin with a Lemma bounding the optimal discrete solution by the optimal continuous solution:

Lemma 3 *With x_d^* defined as above,*

$$H^\delta(x_d^*) \le G\delta + \min_{x \in \mathcal{X} : R(x) \le B} H(x). \tag{23}$$

Proof Consider $x^* \in \arg\min_{x \in \mathcal{X} : R(x) \le B} H(x)$. If x^* corresponds to an integral point in the discretized domain, then $H(x^*) = H^\delta(x_d^*)$ and we are done. Else, since our discretization is δ-fine, we can find a discrete point x_{floor} with $x^* - \delta \le A(x_{\text{floor}}) \le x^*$ elementwise. Algorithmically, x_{floor} is a kind of elementwise floor of x^* with respect to the discretization. There are two implications of the bound between $A(x_{\text{floor}})$ and x^*: first, by monotonicity, $R^\delta(x_{\text{floor}}) \le B$, i.e. $A(x_{\text{floor}})$ is feasible for the original continuous problem; second, we must have $\|x^* - A(x_{\text{floor}})\|_\infty \le \delta$. Applying the Lipschitz property of H and then the optimality of x_d^*, we have

$$G\delta \ge H(A(x_{\text{floor}})) - H(x^*) = H^\delta(x_{\text{floor}}) - H(x^*) \ge H^\delta(x_d^*) - H(x^*),$$

from which (23) follows. □

The next step in proving our suboptimality bound is to bound the suboptimality of our thresholded solutions relative to the true discrete solution:

Lemma 4 *Define λ_- and λ_+ by*

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

Then, we can bound the discrete optimal value $H^\delta(x_d^*)$ on both sides by

$$H^\delta(x(\lambda_+)) \le H^\delta(x_d^*) \le H^\delta(x(\lambda_-)). \tag{24}$$

Proof Note that

$$\min_{x \in \prod_{i=1}^n [k_i] : R^\delta(x) \le B} H^\delta(x) = \min_{x \in \prod_{i=1}^n [k_i]} \max_{\lambda \ge 0} \left\{ H^\delta(x) + \lambda (R^\delta(x) - B) \right\}, \tag{25}$$

since either the term $R^\delta(x)-B$ does not contribute, or it blows up when x is infeasible. Continuing, we can bound:

$$\min_{x\in\prod_{i=1}^n[k_i]:R^\delta(x)\le B} H^\delta(x) = \min_{x\in\prod_{i=1}^n[k_i]} \max_{\lambda\ge 0}\Big\{H^\delta(x)+\lambda(R^\delta(x)-B)\Big\} \tag{26}$$

$$\overset{(a)}{\ge} \max_{\lambda\ge 0}\min_{x\in\prod_{i=1}^n[k_i]}\Big\{H^\delta(x)+\lambda(R^\delta(x)-B)\Big\} \tag{27}$$

$$\overset{(b)}{=} \max_{\lambda\ge 0}\Big\{H^\delta(x(\lambda))+\lambda(R^\delta(x(\lambda))-B)\Big\} \tag{28}$$

$$\overset{(c)}{\ge} \max_{\lambda\ge 0:R^\delta(x(\lambda))\ge B}\Big\{H^\delta(x(\lambda))+\lambda(R^\delta(x(\lambda))-B)\Big\} \tag{29}$$

$$\overset{(d)}{\ge} \max_{\lambda\ge 0:R^\delta(x(\lambda))\ge B} H^\delta(x(\lambda)) \tag{30}$$

$$\overset{(e)}{=} H^\delta(x(\lambda_+)), \tag{31}$$

where (a) uses weak duality, (b) plugs in the definition of $x(\lambda)$, (c) shrinks the set of candidate λ, (d) bounds the regularizing term by zero, and (e) is the definition of $x(\lambda_+)$. We can also bound the optimal value of $H^\delta(x_d^*)$ from the other side:

$$H^\delta(x_d^*) = \min_{x\in\prod_{i=1}^n[k_i]:R^\delta(x)\le B} H^\delta(x) \le \min_{\lambda\ge 0:R^\delta(x(\lambda))\le B} H^\delta(x(\lambda)) = H^\delta(x(\lambda_-)) \tag{32}$$

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

Corollary 2 *In the same setting as Lemma 4, it holds that*

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

Proof Via Lemma 4 we can bound 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 $x(\lambda)$:

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

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, in steps of one. 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 $\left|H^\delta(x_d^*)-H^\delta(x(\lambda_-))\right| \le G\delta$. □

With the above technical results in place, we can easily prove Theorem 2:

Proof (Theorem 2) We now combine Lemma 3 and Corollary 4.2. We have that

$$H(x') \overset{(a)}{=} H^\delta(x(\lambda_-)) \tag{34}$$
$$\overset{(b)}{\leq} G\delta + H^\delta(x_d^*) \tag{35}$$
$$\overset{(c)}{\leq} G\delta + \left(G\delta + \min_{x \in \text{Box}(l,u),\, R(x) \leq B} H(x) \right) \tag{36}$$
$$= 2G\delta + \min_{x \in \text{Box}(l,u),\, R(x) \leq B} H(x), \tag{37}$$

where (a) is the definition of x', (b) follows from Lemma 3 and (c) follows from Corollary 4.2. □

4.2.1 Computable Optimality Bounds

Beyond the theoretical guarantee of Theorem 2, for any problem instance and candidate solution x', we can compute bounds on the gap between $H(x')$ and $H^\delta(x_d^*)$:

1. The discrete point $x(\lambda_+)$ yields the bound

$$H(x') \leq [H(x') - H^\delta(x(\lambda_+))] + H^\delta(x_d^*). \tag{38}$$

2. The Lagrangian yields the bound

$$H(x') \leq \lambda^*(B - R(x')) + H^\delta(x_d^*). \tag{39}$$

The first bound is a simple consequence of Lemma 4:

$$H^\delta(x(\lambda_+)) \leq H^\delta(x_d^*) \tag{40}$$
$$\implies 0 \leq -H^\delta(x(\lambda_+)) + H^\delta(x_d^*) \tag{41}$$
$$\implies H(x') \leq H(x') - H^\delta(x(\lambda_+)) + H^\delta(x_d^*). \tag{42}$$

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

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

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

$$H(x') = H^\delta(x(\lambda^*)) \tag{44}$$
$$\leq H^\delta(x_d^*) + \lambda^*(R^\delta(x_d^*) - R^\delta(x(\lambda^*))) \tag{45}$$
$$\leq H^\delta(x_d^*) + \lambda^*(B - R(x')). \tag{46}$$

One can also combine either of these bounds with the result from the proof of Theorem 2 that $H^\delta(x_d^*) \leq G\delta + H(x^*)$ yielding e.g.

$$H(x') \leq G\delta + \lambda^*(B - R(x')) + H^\delta(x^*). \tag{47}$$

4.2.2 Improvements

The requirement in Theorem 2 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 6.1.1 that in practice, ρ^* almost always has distinct elements in the regime we care about, and the bounds of Remark 4.2.1 are very good.

If H is DR-submodular and R is affine in each coordinate, then Problem (14) can be represented more compactly via the reduction of [30], and hence problem (12) can be solved more efficiently. In particular, the influence function $\mathcal{I}(y;x)$ is DR-submodular in x when for each s, $y(s) = 0$ or $y(s) \geq 1$.

4.2.3 Application to Robust Budget Allocation

The above algorithm directly applies to Robust Allocation with the uncertainty sets in Section 2.2. 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 2 applies.

For D-norm sets, we have $R_{st}(x_{st}) = (x_{st} - \hat{x}_{st})/(u_{st} - \hat{x}_{st})$. Since each R_{st} is monotone, Theorem 2 applies.

4.2.4 Runtime and Alternatives

The core part of the algorithm uses Frank-Wolfe to optimize the regularized convex extension, Problem (17). This convex problem can be solved to ε-suboptimality in time $O(\varepsilon^{-1} n^2 \delta^{-3} \alpha^{-1} |T|^2 \log n\delta^{-1})$, where α is the minimum derivative of the functions R_i (see Appendix C.2 for details). Suppose that the optimal solution ρ^* to Problem (17) has distinct elements separated by η; then choosing $\varepsilon = \eta^2 \alpha \delta / 8$ results in an exact solution to the discrete regularized Problem (14) in total time $O(\eta^{-2} n^2 \delta^{-4} \alpha^{-2} |T|^2 \log n\delta^{-1})$.

Noting that $H^\delta + \lambda R^\delta$ is submodular for all λ, one could instead perform binary search over λ, each time converting the objective into a submodular *set* function via Birkhoff's theorem and solving submodular minimization e.g. via a fast, recent method [18,51]. However, we are not aware of a practical implementation of the algorithm in [51]. The algorithm in [18] yields a solution only in expectation. This approach also requires care in the precision of the search over λ, whereas our approach solves for all λ simultaneously, and picks directly from the $O(n\delta^{-1})$ elements of ρ^*.

A host of alternate approaches are also possible, e.g. a generalization of the minimum norm point algorithm [73,32] which is also suggested by [5]. However such development is out of scope for this paper: our focus is on developing a convex formulation, rather than optimizing the algorithm.

5 Simple examples where our approach is optimal

Next, we take a view beyond Budget Allocation, and theoretically and empirically evaluate the optimality of our constrained submodular minimization algorithm on two classes of nonconvex problem where the optimal solution can be computed. For one class, the algorithm provably yields the global optimal solution. For the other class, the algorithm empirically yields solutions that are very close to globally optimal solutions from a specialized SDP relaxation.

5.1 Separable problems

First, we assume that the objective function $H(x)$ and constraint function $R(x)$ are continuous submodular and *separable*. Some such problems admit simple analytic solutions despite nonconvexity. Our approach will recover these solutions. To understand how our method behaves when the objective and constraints are separable, the following structural result about the convex extension will be useful.

Lemma 5 (also appears informally in [5]) *Suppose the objective is separable:* $H(x) = \sum_{i=1}^n H_i(x_i)$. *Then the convex extension* $h_\downarrow(\mu_1, \dots, \mu_n)$ *is also separable:*

$$h_\downarrow(\mu_1, \dots, \mu_n) = \sum_{i=1}^n h_{i\downarrow}(\mu_i), \tag{48}$$

where $h_{i\downarrow}$ *is the extension for* H_i.

Note that this separability also holds for any block-separable structure.

Proof First we write out the definition of the convex extension $h_\downarrow(\mu)$:

$$\begin{aligned} h_\downarrow(\mu_1, \dots, \mu_n) &= \inf_{\gamma \in \mathcal{P}(\mathcal{X}, \{\mu_i\})} \int_{\mathcal{X}} H(x)\, d\gamma(x) && (49)\\ &= \inf_{\gamma \in \mathcal{P}(\mathcal{X}, \{\mu_i\})} \int_{\mathcal{X}} \sum_{i=1}^n H_i(x_i)\, d\gamma(x) && (50)\\ &= \inf_{\gamma \in \mathcal{P}(\mathcal{X}, \{\mu_i\})} \sum_{i=1}^n \int_{\mathcal{X}} H_i(x_i)\, d\gamma(x). && (51) \end{aligned}$$

Each integral of $H_i(x_i)$ only depends on the marginal of γ, which by definition is μ_i. Since this is the only dependence on γ, the infimum is now unnecessary:

$$h_\downarrow(\mu_1, \dots, \mu_n) = \sum_{i=1}^n \int_{\mathcal{X}_i} H_i(x_i)\, d\mu_i(x_i) = \sum_{i=1}^n h_{i\downarrow}(\mu_i). \tag{52}$$

□

Write $\Delta H_i(x_i) = H_i(x_i) - H_i(x_i - 1)$ and similarly for $R_i(x_i)$. Using Lemma 5 we can prove the following structural result:

Proposition 3 *Suppose $H(x)$ and $R(x)$ are both separable as above. Consider the regularized problem:*

$$\begin{array}{ll}\text{minimize} & h_{\downarrow}(\rho)+\sum_{i=1}^{n}\sum_{x_i=1}^{k_i-1}a_{ix_i}(\rho_i(x_i)) \\ \text{s.t.} & \rho\in\prod_{i=1}^{n}\mathbb{R}_{\downarrow}^{k_i-1}.\end{array} \tag{53}$$

If $Q_i(x_i) := \frac{\Delta H_i(x_i)}{\Delta R_i(x_i)}$ is nondecreasing in x_i for all i, then the optimal solution for Problem (53) *is given by $\rho_i^*(x_i) = -Q_i(x_i)$.*

Proof By Lemma 5 we have $h_{\downarrow}(\rho) = \sum_{i=1}^{n} h_{i\downarrow}(\rho_i)$. In the single dimensional case, $h_{i\downarrow}$, the extension is very easy to compute, as it is given by $h_{i\downarrow}(\mu_i) = \int_{\mathcal{X}_i} H_i(x_i)\, d\mu_i(x_i)$. We instead use the alternative characterization of $h_{i\downarrow}$ in terms of the reversed cumulative distribution function ρ_i. In the discrete case, we write $\rho_i(x_i) = \mu_i(x_i) + \mu_i(x_i+1) + \cdots + \mu_i(k_i-1)$, and so $h_{i\downarrow}(\rho_i)$ is given by:

$$h_{i\downarrow}(\rho_i) = H_i(0) + \sum_{x_i=1}^{k_i-1} \rho_i(x_i)(H_i(x_i) - H_i(x_i-1)) \tag{54}$$

$$= H_i(0) + \sum_{x_i=1}^{k_i-1} \rho_i(x_i)\Delta H_i(x_i). \tag{55}$$

We assumed the constraint functions $R(x) = \sum_{i=1}^{n} R_i(x_i)$ are separable over i. As in the proof of Lemma 2, we convert each R_i into strongly convex functions $a_{ix_i}(t) = \frac{1}{2}t^2\Delta R_i(x_i)$. Since the convex extension $h_{\downarrow}$ is now also separable, as are the monotonicity constraints, we may separately consider n problems of the form:

$$\begin{array}{ll}\text{minimize} & h_{i\downarrow}(\rho_i)+\sum_{x_i=1}^{k_i-1}a_{ix_i}(\rho_i(x_i)) \\ \text{s.t.} & \rho_i\in\mathbb{R}_{\downarrow}^{k_i-1}\end{array} \tag{56}$$

The first term $h_{i\downarrow}(\rho_i)$ is also a sum over x_i, so we may rewrite the objective as:

$$\sum_{x_i=1}^{k_i-1} \rho_i(x_i)\Delta H_i(x_i) + \frac{1}{2}\sum_{x_i=1}^{k_i-1}[\rho_i(x_i)]^2\Delta R_i(x_i) \tag{57}$$

$$= \sum_{x_i=1}^{k_i-1}\left\{\rho_i(x_i)\Delta H_i(x_i) + \frac{1}{2}[\rho_i(x_i)]^2\Delta R_i(x_i)\right\}. \tag{58}$$

Completing the square, we may write

$$\rho_i(x_i)\Delta H_i(x_i) + \frac{1}{2}[\rho_i(x_i)]^2\Delta R_i(x_i) \tag{59}$$

$$= \frac{\Delta R_i(x_i)}{2}\left(\rho_i(x_i) + \frac{\Delta H_i(x_i)}{\Delta R_i(x_i)}\right)^2 - \frac{\Delta R_i}{2}\cdot\left(\frac{\Delta H_i(x_i)}{\Delta R_i(x_i)}\right)^2. \tag{60}$$

The last term is a constant that does not depend on ρ_i, so we ignore it. Using, in addition, the identity $Q_i(x_i) = \frac{\Delta H_i(x_i)}{\Delta R_i(x_i)}$, the problem we wish to solve is:

$$\begin{array}{ll}\text{minimize} & \sum_{x_i=1}^{k_i-1}\Delta R_i(x_i)\left(\rho_i(x_i)+Q_i(x_i)\right)^2 \\ \text{s.t.} & \rho_i\in\mathbb{R}_{\downarrow}^{k_i-1}\end{array}. \tag{61}$$

This is a weighted isotonic regression problem. We try to fit $\rho_i(x_i)$ to the value $-Q_i(x_i)$, with associated weight $\Delta R_i(x_i)$. We assumed $Q_i(x_i)$ is nondecreasing, so $-Q_i(x_i)$ is nonincreasing. Therefore setting $\rho_i^*(x_i) = -Q_i(x_i)$ is feasible and obtains optimal objective value (zero). □

There are many situations in which $Q_i(x_i)$ is nondecreasing, and therefore $\rho_i^*(x_i) = -Q_i(x_i)$. We focus on a particularly simple one: let $H_i(x_i) = a_i x_i^p$ and $R_i(x_i) = r_i x_i^p$, so that the overall optimization problem is

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n a_i x_i^p \\ \text{s.t.} & \sum_{i=1}^n r_i x_i^p \le B \\ & \mathbf{0} \le x \le \mathbf{1}. \end{array} \tag{62}$$

The problem might be nonconvex in its current form, but the transformation $y_i = r_i x_i^p$ gives a convex problem:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n \frac{a_i}{r_i} \cdot y_i \\ \text{s.t.} & \mathbf{1}^T y \le B \\ & \mathbf{0} \le y \le r. \end{array} \tag{63}$$

The constraint here is a scaled version of the simplex. An optimal solution can be found by sorting the indices so $\frac{a_1}{r_1} \le \cdots \le \frac{a_n}{r_n}$, and saturating $y_1, y_2, \ldots$ in order until the total budget B is achieved.

This procedure is precisely equivalent to what our algorithm will do, even without the convex reparameterization. In our case,

$$Q_i(x_i) = \frac{a_i}{r_i} \cdot \frac{x_i^p - (x_i - 1)^p}{x_i^p - (x_i - 1)^p} = \frac{a_i}{r_i} \tag{64}$$

is constant, hence nondecreasing. Therefore $\rho_i^*(x_i) = -\frac{a_i}{r_i}$ solves Problem (53). Consider now the thresholding step of our algorithm, where we search over λ and take $\rho_i^*(x_i)$ only if $\rho_i^*(x_i) \ge \lambda$. Since $\rho_i^*(x_i) = \rho_i^*$ is constant, we will take entire coordinates at a time: we will first find the coordinate i with $-Q_i = -\frac{a_i}{r_i}$ maximized, i.e. $\frac{a_i}{r_i}$ minimized. We set $x_i = k_i - 1$ (which maps back to $x_i = 1$ when we un-discretize). Then we move onto the next best coordinate, and so on.

Proposition 3 confirms at least for this special case that our algorithm finds the optimal solution. Moreover, we can find the optimal solution without using a more specialized approach that depends on e.g. quadratic problem structure.

5.2 Non-separable quadratics and SDP relaxations

Reasoning about guaranteed performance gets more difficult as we move away from separable problems. Even empirical evaluation becomes problematic in nonconvex settings where all we can know about the globally optimal solution is the suboptimality bound returned by our algorithm. Before addressing these harder regimes, we

study a harder class of nonconvex problems that still admit global optimality guarantees. Specifically we look at a certain class of nonconvex quadratically-constrained quadratic problems:

$$\begin{array}{ll} \min\limits_{x} & \frac{1}{2}x^T Ax + c^T x \\ \text{s.t.} & x \in \text{Box}(0,1) \\ & \frac{1}{2}x^T \operatorname{diag}(r)x \le B \end{array} \quad \Leftrightarrow \quad \begin{array}{ll} \min\limits_{x} & \frac{1}{2}x^T Ax + c^T x \\ \text{s.t.} & x_i^2 \le x_i \; \forall i \\ & \frac{1}{2}x^T \operatorname{diag}(r)x \le B, \end{array}$$

where A has nonpositive off-diagonal entries. These problems are a useful benchmark because they can be solved globally via an SDP relaxation [46]. Concretely:

Theorem 3 (Theorem 4 from [46]) *Let A have nonpositive off-diagonal entries. Let (X^*, x^*) be a solution to the SDP*

$$\begin{array}{ll} \min\limits_{X,x} & \frac{1}{2}\operatorname{tr}(AX) + c^T x \\ \text{s.t.} & \operatorname{diag}(X) \le x \\ & \frac{1}{2}\operatorname{tr}(\operatorname{diag}(r)X) \le B \\ & \begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0. \end{array}$$

Taking z^* to be the elementwise square root of $\operatorname{diag}(X^*)$ yields an optimal solution to the original nonconvex problem.

We emphasize that this is a very special subclass of constrained submodular problems: the SDP approach only applies when both the objective and constraint are quadratic, while our algorithm applies more broadly.

We compare our constrained submodular optimization algorithm to the SDP relaxation on random problems. For our algorithm, we discretize each coordinate into $k = 1000$ pieces, and run only 300 iterations of Frank-Wolfe on the convex relaxation (17). The matrix A is set to $M + M^T$, where each entry of M is sampled uniformly in $[-1, 0]$. The linear term c is set to zero, the constraint vector r is set to the all ones vector, and we vary B. Figure 1 shows histograms of the gap in performance between the two algorithms. For most instances our approach does nearly as well as the globally optimal SDP solution.

In fact, if A is restricted to be a diagonal matrix (we take only its diagonal part), our algorithm always achieved a relative suboptimality gap below 10^{-4}. As predicted by Proposition 3, for separable problems our algorithm is essentially optimal.

5.3 Evaluation of suboptimality bounds

In Section 4.2.1 we give solution-dependent suboptimality bounds for the algorithm. These require only the discrete solution, (possibly) the optimal Lagrange multiplier, the granularity δ (chosen to be 0.001 in these experiments), and the ℓ_∞ Lipschitz

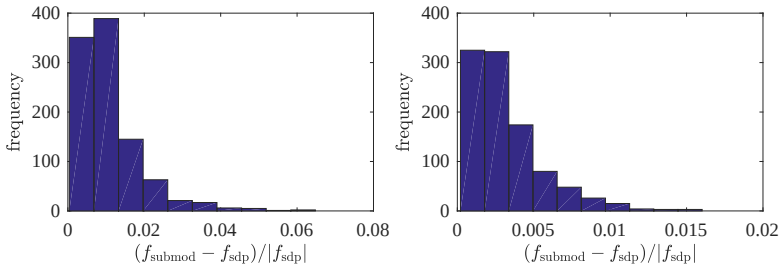

Fig. 1 Relative suboptimality of the submodular optimization solution (objective value f_{submod}) vs the globally optimal SDP solution (objective value f_{sdp}). Quadratic constraints $\|x\|^2 \leq B$ with $B = 0.1$ (left) and $B = 1$ (right).

constant of $H(x) = \frac{1}{2}x^T Ax$. Since $x \leq 1$ elementwise, we can equivalently bound the ℓ_∞ norm of the linear map $x \mapsto \frac{1}{2}\mathbf{1}^T Ax$, which is just $\frac{1}{2}\|\mathbf{1}^T A\|_1$. Since the bounds do assume access to the optimal ρ^* for Problem (17), it is important to run Frank-Wolfe for enough iterations to get a good approximation to ρ^*.

For each quadratic experiment from the previous section, we compute the best available suboptimality bound. Here, even 300 iterations were easily sufficient to approximate ρ^* for this purpose. For the quadratic problems, the bounds from Section 4.2.1 were typically ≈ 0.1 for $B = 0.1$ and ≈ 0.01 for $B = 1$. The computed bounds were always an upper bound on the true suboptimality in our experiments.

6 Robust Budget Allocation Experiments

After testing the core submodular minimization subroutine, we return to the motivating application of Robust Budget Allocation. We evaluate our algorithm on both synthetic test data and a real-world bidding dataset from Yahoo! Webscope [1] to demonstrate that our method yields real improvements. For all experiments, we used Algorithm 1 as the outer loop. For the inner submodular minimization step, we implemented the pairwise Frank-Wolfe algorithm of [50]. 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. Our code is available at `git.io/vHXk0`.

6.1 Synthetic

On the synthetic data, we probe two questions: (1) how often does the distinctness condition of Theorem 2 hold, so that we are guaranteed an optimal solution; and (2) what is the gain of using a robust versus non-robust solution in an adversarial setting? For both settings, we set $|S| = 6$ and $|T| = 2$ and discretize with $\delta = 0.001$. We generated true probabilties p_{st}, created Beta posteriors, and built both Ellipsoidal uncertainty sets $\mathcal{X}^Q(\gamma)$ and D-norm sets $\mathcal{X}^D(\gamma)$.

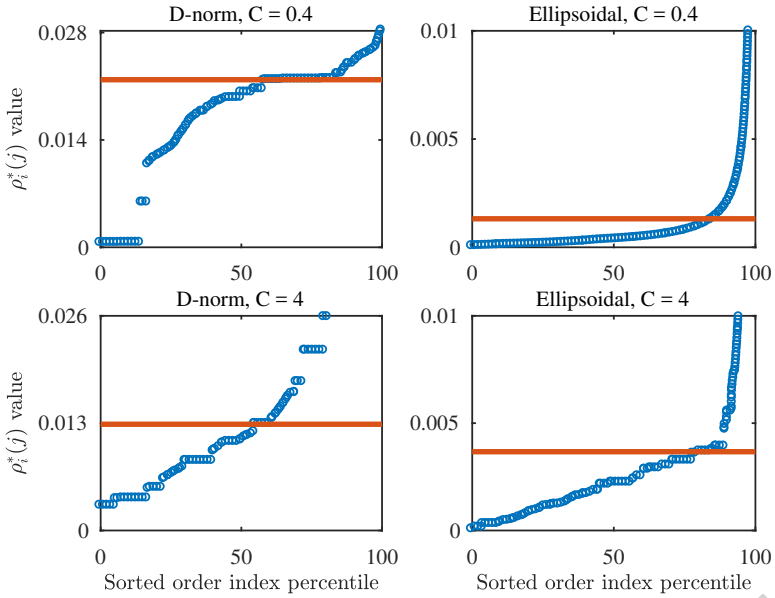

Fig. 2 Visualization of the sorted values of $\rho_i^*(j)$ (blue dots) with comparison to the particular Lagrange multiplier λ^* (orange line). In most regimes there are no duplicate values, so that Theorem 2 applies. The theorem only needs distinctness at λ^*.

6.1.1 Optimality

Theorem 2 and Remark 4.2.2 demand that the values $\rho_i^*(j)$ be distinct at our chosen Lagrange multiplier λ^* and, under this condition, guarantee optimality. We illustrate this in four examples: for Ellipsoidal or a D-norm uncertainty set, and a total influence budget $C \in \{0.4, 4\}$. Figure 2 shows all elements of ρ^* in sorted order, as well as a horizontal line indicating our Lagrange multiplier λ^* which serves as a threshold. Despite some plateaus, the entries $\rho_i^*(j)$ are distinct in most regimes, in particular around λ^*, the regime that is needed for our results. Moreover, in practice (on the Yahoo data) we observe later in Figure 4 that both solution-dependent bounds from Remark 4.2.1 are very good, and all solutions are optimal within a very small gap.

6.1.2 Robustness and Quality

Next, we probe the effect of a robust versus non-robust solution for different uncertainty sets and budgets γ of the adversary. We compare our robust solution with using a point estimate for x, i.e., $y_{\text{nom}} \in \operatorname{argmax}_{y \in \mathcal{Y}} \mathcal{I}(y; \hat{x})$, treating estimates as ground truth, and the stochastic solution $y_{\text{expect}} \in \operatorname{argmax}_{y \in \mathcal{Y}} \mathbb{E}[\mathcal{I}(y; X)]$ as per Section 2.1. These two optimization problems were solved via standard first-order methods using TFOCS [8].

Figure 3 demonstrates that indeed, the alternative budgets are sensitive to the adversary and the robustly-chosen budget y_{robust} performs better, even in cases where

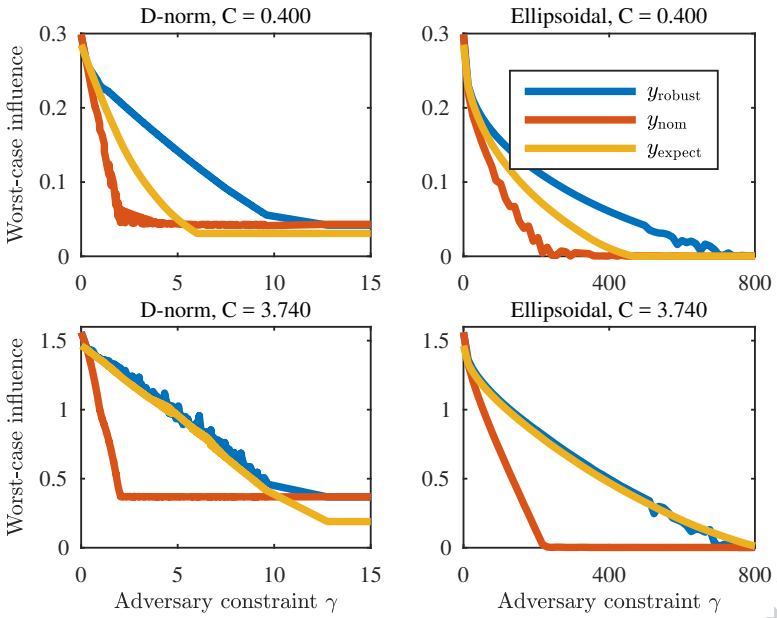

Fig. 3 Comparison of worst-case expected influences for D-norm uncertainty sets $\mathcal{X}^D(\gamma)$ (left) and ellipsoidal uncertainty sets $\mathcal{X}^Q(\gamma)$ (right), for different total budget bounds C. For any particular adversary budget γ, we compare $\min_{x\in\mathcal{X}(\gamma)} \mathcal{I}(y;x)$ for each candidate allocation y.

the other budgets achieve zero influence. When the total budget C is large, y_{expect} performs nearly as well as y_{robust}, but when resources are scarce (C is small) and the actual choice seems to matter more, y_{robust} performs far better.

6.2 Yahoo! data

To evaluate our method on real-world data, we formulate a Budget Allocation instance on 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 not to be 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 between s and t in the dataset.

This transformation yields a bipartite graph with $|S| = 1000$, $|T| = 10475$, and more than 50,000 edges that we use for Budget Allocation. In our experiments, the typical gap between the naive y_{nom} and robust y_{robust} was 100-500 expected influenced people. We plot convergence of the outer loop in Figure 4, where we observe fast convergence of both primal influence value and the dual bound.

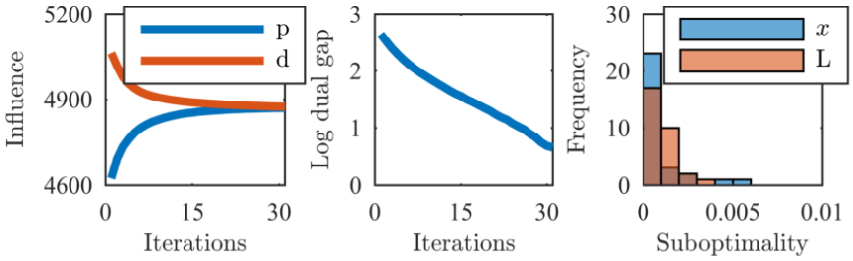

Fig. 4 Convergence properties of our algorithm on real data. In the first plot, 'p' and 'd' refer to primal and dual values, with dual gap shown on the second plot. The third plot demonstrates that the problem-dependent suboptimality bounds of Remark 4.2.1 (x for $x(\lambda_+)$ and L for Lagrangian) are very small (good) for all inner iterations of this run.

6.3 Comparison to first-order methods

Given the success of first-order methods on nonconvex problems in practice, it is natural to compare these to our method for finding the worst-case vector x. On one of our Yahoo problem instances with D-norm uncertainty set, we compared our submodular minimization scheme to Frank-Wolfe with fixed stepsize as in [49], implementing the linear oracle using MOSEK [55]. Interestingly, from various initializations, Frank-Wolfe finds an optimal solution, as verified by comparing to the guaranteed solution of our algorithm. Note that, due to nonconvexity, there are no formal guarantees for Frank-Wolfe to be optimal here, motivating the question of global convergence properties of Frank-Wolfe in the presence of submodularity.

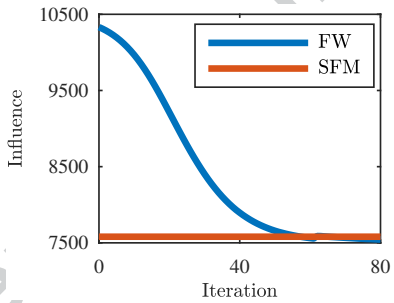

Fig. 5 Convergence properties of Frank-Wolfe (FW), versus the optimal value attained with our scheme (SFM).

It is important to note that there are many cases where first-order methods are inefficient or do not apply to our setup. These methods require either a projection oracle onto or linear optimization oracle over the feasible set $\mathcal{X}$ defined by ℓ, u and $R(x)$. The D-norm set admits a linear optimization oracle via linear programming, but we are not aware of any efficient linear optimization oracle for Ellipsoidal uncertainty, nor projection oracle for either set, that does not require quadratic programming. Even more, our algorithm applies for nonconvex functions $R(x)$ which induce nonconvex feasible sets $\mathcal{X}$. Such nonconvex sets may not even admit a unique projection, while our algorithm achieves provable solutions.

7 Conclusion

We address the issue of uncertain parameters (or, model mis-specification) in Budget Allocation or Bipartite Influence Maximization [3] from a robust optimization perspective. The resulting *Robust Budget Allocation* is a nonconvex-concave saddle point problem. Although the inner optimization problem is nonconvex, we show how continuous submodularity can be leveraged to solve the problem to arbitrary accuracy ε, as can be verified with the proposed bounds on the duality gap. In particular, our approach extends continuous submodular minimization methods [5] to more general constraint sets, introducing a mechanism to solve a new class of constrained nonconvex optimization problems. Our method provably performs well on a class of separable nonconvex problems, and empirically well on nonconvex quadratics. For Robust Budget Allocation, we confirm on synthetic and real data that our method finds high-quality solutions that are robust to parameters varying arbitrarily in an uncertainty set, and scales up to graphs with over 50,000 edges.

There are many compelling directions for further study. The uncertainty sets we use are standard in the robust optimization literature, but have not been applied to e.g. Robust Influence Maximization; it would be interesting to generalize our ideas to general graphs. Finally, despite the inherent nonconvexity of our problem, first-order methods are often able to find a globally optimal solution. Explaining this phenomenon requires further study of the geometry of constrained monotone submodular minimization.

Acknowledgements We thank the anonymous reviewers for their helpful suggestions. We also thank MIT Supercloud and the Lincoln Laboratory Supercomputing Center for providing computational resources. This research was conducted with Government support under and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a, and also supported by NSF CAREER award 1553284 and The Defense Advanced Research Projects Agency (grant number YFA17 N66001-17-1-4039). The views, opinions, and/or findings contained in this article are those of the author and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

References

1. Yahoo! Webscope dataset ydata-ysm-advertiser-bids-v1_0. URL `http://research.yahoo.com/Academic_Relations`
2. Adamczyk, M., Sviridenko, M., Ward, J.: Submodular Stochastic Probing on Matroids. Mathematics of Operations Research **41**(3), 1022–1038 (2016). DOI 10.1287/moor.2015.0766
3. Alon, N., Gamzu, I., Tennenholtz, M.: Optimizing Budget Allocation Among Channels and Influencers. In: WWW, pp. 381–388. ACM, New York, NY, USA (2012). DOI 10.1145/2187836.2187888
4. Atamtürk, A., Narayanan, V.: Polymatroids and Mean-Risk Minimization in Discrete Optimization. Operations Research Letters **36**(5), 618–622 (2008). DOI 10.1016/j.orl.2008.04.006
5. Bach, F.: Submodular Functions: From Discrete to Continous Domains. Mathematical Programming (2018). DOI 10.1007/s10107-018-1248-6
6. Balkanski, E., Rubinstein, A., Singer, Y.: The Power of Optimization from Samples. In: NIPS, pp. 4017–4025 (2016)
7. Balkanski, E., Rubinstein, A., Singer, Y.: The Limitations of Optimization from Samples. In: STOC (2017)

8. Becker, S.R., Candès, E.J., Grant, M.C.: Templates for Convex Cone Problems with Applications to Sparse Signal Recovery. Mathematical programming computation **3**(3), 165–218 (2011)
9. Ben-Tal, A., El Ghaoui, L., Nemirovski, A.: Robust Optimization. Princeton University Press (2009)
10. Ben-Tal, A., Nemirovski, A.: Robust Solutions of Linear Programming Problems Contaminated with Uncertain Data. Mathematical Programming **88**(3), 411–424 (2000). DOI 10.1007/PL00011380
11. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and Applications of Robust Optimization. SIAM Review **53**(3), 464–501 (2011). DOI 10.1137/080734510
12. Bertsimas, D., Sim, M.: Robust Discrete Optimization and Network Flows. Mathematical programming **98**(1), 49–71 (2003)
13. Best, M.J., Chakravarti, N.: Active Set Algorithms for Isotonic Regression; A Unifying Framework. Mathematical Programming **47**(1-3), 425–439 (1990). DOI 10.1007/BF01580873
14. Bian, A.A., Mirzasoleiman, B., Buhmann, J.M., Krause, A.: Guaranteed Non-convex Optimization: Submodular Maximization over Continuous Domains. In: AISTATS (2017)
15. Birkhoff, G.: Rings of Sets. Duke Mathematical Journal **3**(3), 443–454 (1937)
16. Borgs, C., Brautbar, M., Chayes, J., Lucier, B.: Maximizing Social Influence in Nearly Optimal Time. In: SODA, pp. 946–957. Philadelphia, PA, USA (2014)
17. Boyd, S., Kim, S.J., Vandenberghe, L., Hassibi, A.: A Tutorial on Geometric Programming. Optimization and engineering **8**(1), 67–127 (2007)
18. Chakrabarty, D., Lee, Y.T., Sidford, A., Wong, S.C.w.: Subquadratic Submodular Function Minimization. In: STOC (2017)
19. Chandrasekaran, V., Shah, P.: Relative Entropy Relaxations for Signomial Optimization. SIAM Journal on Optimization **26**(2), 1147–1173 (2016). DOI 10.1137/140988978
20. Chen, W., Lin, T., Tan, Z., Zhao, M., Zhou, X.: Robust Influence Maximization. In: KDD, pp. 795–804. ACM, New York, NY, USA (2016). DOI 10.1145/2939672.2939745
21. Chen, W., Wang, C., Wang, Y.: Scalable Influence Maximization for Prevalent Viral Marketing in Large-scale Social Networks. In: KDD, pp. 1029–1038. New York, NY, USA (2010). DOI 10.1145/1835804.1835934
22. Chen, W., Wang, Y., Yang, S.: Efficient Influence Maximization in Social Networks. In: KDD, pp. 199–208 (2009)
23. Chiang, M.: Geometric Programming for Communication Systems. Commun. Inf. Theory **2**(1/2), 1–154 (2005). DOI 10.1516/0100000005
24. Deshpande, A., Hellerstein, L., Kletenik, D.: 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 (2016). DOI 10.1145/2876506
25. Domingos, P., Richardson, M.: Mining the Network Value of Customers. In: KDD, pp. 57–66 (2001)
26. Du, N., Liang, Y., Balcan, M.F., Song, L.: Influence Function Learning in Information Diffusion Networks. In: ICML, pp. 2016–2024 (2014)
27. Du, N., Song, L., Gomez Rodriguez, M., Zha, H.: Scalable Influence Estimation in Continuous-Time Diffusion Networks. In: NIPS, pp. 3147–3155 (2013)
28. Dunn, J.C., Harshbarger, S.: Conditional Gradient Algorithms with Open Loop Step Size Rules. Journal of Mathematical Analysis and Applications **62**(2), 432–444 (1978)
29. Ecker, J.: Geometric Programming: Methods, Computations and Applications. SIAM Review **22**(3), 338–362 (1980). DOI 10.1137/1022058
30. Ene, A., Nguyen, H.L.: A Reduction for Optimizing Lattice Submodular Functions with Diminishing Returns. arXiv:1606.08362 (2016)
31. Frank, M., Wolfe, P.: An Algorithm for Quadratic Programming. Naval Research Logistics Quarterly **3**(1-2), 95–110 (1956). DOI 10.1002/nav.3800030109
32. Fujishige, S.: Submodular Functions and Optimization, vol. 58. Elsevier (2005)
33. Goel, G., Karande, C., Tripathi, P., Wang, L.: Approximability of Combinatorial Problems with Multi-Agent Submodular Cost Functions. In: FOCS, pp. 755–764 (2009)
34. Goemans, M., Vondrák, J.: Stochastic Covering and Adaptivity. In: LATIN 2006: Theoretical Informatics, pp. 532–543. Springer Berlin Heidelberg (2006). DOI 10.1007/11682462_50
35. Golovin, D., Krause, A.: Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization. Journal of Artificial Intelligence **42**, 427–486 (2011)
36. Gomez Rodriguez, M., Leskovec, J., Krause, A.: Inferring Networks of Diffusion and Influence. In: KDD, pp. 1019–1028. New York, NY, USA (2010). DOI 10.1145/1835804.1835933
37. Gomez Rodriguez, M., Schölkopf, B.: Influence Maximization in Continuous Time Diffusion Networks. In: ICML (2012)

38. Gottschalk, C., Peis, B.: Submodular Function Maximization on the Bounded Integer Lattice. In: Approximation and Online Algorithms: 13th International Workshop (WAOA) (2015)
39. Hassidim, A., Singer, Y.: Submodular Optimization under Noise. In: S. Kale, O. Shamir (eds.) Proceedings of the 2017 Conference on Learning Theory, *Proceedings of Machine Learning Research*, vol. 65, pp. 1069–1122. PMLR, Amsterdam, Netherlands (2017). URL `http://proceedings.mlr.press/v65/hassidim17a.html`
40. Hatano, D., Fukunaga, T., Maehara, T., Kawarabayashi, K.i.: Lagrangian Decomposition Algorithm for Allocating Marketing Channels. In: AAAI, pp. 1144–1150 (2015)
41. He, X., Kempe, D.: Robust Influence Maximization. In: KDD, pp. 885–894. ACM, New York, NY, USA (2016). DOI 10.1145/2939672.2939760
42. Iwata, S., Nagano, K.: Submodular Function Minimization under Covering Constraints. In: FOCS, pp. 671–680 (2009)
43. Jaggi, M.: Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In: ICML, pp. 427–435 (2013)
44. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the Spread of Influence Through a Social Network. In: KDD, pp. 137–146. New York, NY, USA (2003). DOI 10.1145/956750.956769
45. Khachaturov, V.R., Khachaturov, R.V., Khachaturov, R.V.: Supermodular Programming on Finite Lattices. Computational Mathematics and Mathematical Physics **52**(6), 855–878 (2012). DOI 10.1134/S0965542512060097
46. Kim, S., Kojima, M.: Exact Solutions of Some Nonconvex Quadratic Optimization Problems via SDP and SOCP Relaxations. Computational Optimization and Applications **26**(2), 143–154 (2003). DOI 10.1023/A:1025794313696. URL `https://doi.org/10.1023/A:1025794313696`
47. Kolmogorov, V., Shioura, A.: New Algorithms for Convex Cost Tension Problem with Application to Computer Vision. Discrete Optimization **6**, 378–393 (2009)
48. Krause, A., McMahan, H.B., Guestrin, C., Gupta, A.: Robust Submodular Observation Selection. Journal of Machine Learning Research **9**(Dec), 2761–2801 (2008)
49. Lacoste-Julien, S.: Convergence Rate of Frank-Wolfe for Non-Convex Objectives. arXiv:1607.00345 (2016)
50. Lacoste-Julien, S., Jaggi, M.: On the Global Linear Convergence of Frank-Wolfe Optimization Variants. In: NIPS, pp. 496–504 (2015)
51. Lee, Y.T., Sidford, A., Wong, S.C.w.: A faster Cutting Plane Method and its Implications for Combinatorial and Convex Optimization. In: FOCS, pp. 1049–1065 (2015)
52. Lowalekar, M., Varakantham, P., Kumar, A.: Robust Influence Maximization: (Extended Abstract). In: AAMAS, pp. 1395–1396. Richland, SC (2016)
53. Maehara, T.: Risk Averse Submodular Utility Maximization. Operations Research Letters **43**(5), 526–529 (2015). DOI 10.1016/j.orl.2015.08.001
54. Maehara, T., Yabe, A., Kawarabayashi, K.i.: Budget Allocation Problem with Multiple Advertisers: A Game Theoretic View. In: ICML, pp. 428–437 (2015)
55. MOSEK ApS: MOSEK MATLAB Toolbox 8.0.0.57 (2015). URL `http://docs.mosek.com/8.0/toolbox/index.html`
56. Murota, K.: Discrete Convex Analysis. SIAM (2003)
57. Murota, K., Shioura, A.: Exact Bounds for Steepest Descent Algorithms of *L*-convex Function Minimization. Operations Research Letters **42**, 361–366 (2014)
58. Nagano, K., Kawahara, Y., Aihara, K.: Size-Constrained Submodular Minimization through Minimum Norm Base. In: ICML, pp. 977–984 (2011)
59. Narasimhan, H., Parkes, D.C., Singer, Y.: Learnability of Influence in Networks. In: NIPS, pp. 3186–3194 (2015)
60. Netrapalli, P., Sanghavi, S.: Learning the Graph of Epidemic Cascades. In: SIGMETRICS, pp. 211–222. ACM, New York, NY, USA (2012). DOI 10.1145/2254756.2254783
61. Nikolova, E.: Approximation Algorithms for Reliable Stochastic Combinatorial Optimization. In: APPROX, pp. 338–351. Springer-Verlag, Berlin, Heidelberg (2010)
62. Orlin, J.B., Schulz, A., Udwani, R.: Robust Monotone Submodular Function Maximization. In: IPCO (2016)
63. Pascual, L.D., Ben-Israel, A.: Constrained Maximization of Posynomials by Geometric Programming. Journal of Optimization Theory and Applications **5**(2), 73–80 (1970). DOI 10.1007/BF00928296
64. Polyak, B.T.: Introduction to Optimization. 04; QA402. 5, P6. (1987)
65. Rockafellar, R.T., Uryasev, S.: Optimization of Conditional Value-at-Risk. Journal of risk **2**, 21–42 (2000)

66. Rockafellar, R.T., Uryasev, S.: Conditional Value-at-Risk for General Loss Distributions. Journal of banking & finance **26**(7), 1443–1471 (2002)
67. Soma, T., Kakimura, N., Inaba, K., Kawarabayashi, K.i.: Optimal Budget Allocation: Theoretical Guarantee and Efficient Algorithm. In: ICML, pp. 351–359 (2014)
68. Soma, T., Yoshida, Y.: A Generalization of Submodular Cover via the Diminishing Return Property on the Integer Lattice. In: NIPS, pp. 847–855 (2015)
69. Svitkina, Z., Fleischer, L.: Submodular Approximation: Sampling-Based Algorithms and Lower Bounds. SIAM Journal on Computing **40**(6), 1715–1737 (2011)
70. Topkis, D.M.: Minimizing a Submodular Function on a Lattice. Operations research **26**(2), 305–321 (1978)
71. Wainwright, K., Chiang, A.: Fundamental Methods of Mathematical Economics. McGraw-Hill Education (2004)
72. Wilder, B.: Risk-Sensitive Submodular Optimization. In: AAAI Conference on Artificial Intelligence (2018)
73. Wolfe, P.: Finding the nearest point in a polytope. Mathematical Programming **11**(1), 128–149 (1976). DOI 10.1007/BF01580381. URL `https://doi.org/10.1007/BF01580381`
74. Zhang, P., Chen, W., Sun, X., Wang, Y., Zhang, J.: Minimizing Seed Set Selection with Probabilistic Coverage Guarantee in a Social Network. In: KDD, pp. 1306–1315. New York, NY, USA (2014). DOI 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:

$$f(x;0)=\begin{cases}1 & x=0\\ 0.6 & x=1,\end{cases}\qquad f(x;1)=\begin{cases}1 & x=0\\ 2 & x=1.\end{cases} \tag{65}$$

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:

$$\max_x \min_\theta \frac{f(x;\theta)}{f(x^*_\theta;\theta)}. \tag{66}$$

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:

$$\max_x \min_\theta f(x;\theta). \tag{67}$$

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 DR-submodularity and $L^\natural$-convexity

A function is $L^\natural$-convex if it satisfies a discrete version of midpoint convexity, i.e. for all x,y it holds that

$$f(x)+f(y)\geq f\left(\left\lceil\frac{x+y}{2}\right\rceil\right)+f\left(\left\lfloor\frac{x+y}{2}\right\rfloor\right), \tag{68}$$

where the floor $\lfloor\cdot\rfloor$ and ceiling $\lceil\cdot\rceil$ functions are interpreted elementwise.

Remark 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.

Proof 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.

Intuitively-speaking, $L^\natural$-convex functions look like discretizations of convex functions. The continuous objective function $\mathcal{I}(x,y)$ we consider need not be convex, hence its discretization need not be $L^\natural$-convex, and we cannot use those tools. However, in some regimes (namely if each $y(s) \in \{0\} \cup [1,\infty)$), it happens that $\mathcal{I}(x,y)$ is DR-submodular in x.

C Constrained Continuous Submodular Function Minimization

C.1 Solving the Optimization Problem

Here, we describe how to solve the convex problem (17) to which we reduced the original constrained submodular minimization problem. Bach [5], at the beginning of Section 5.2, states that this surrogate problem can be optimized via the Frank-Wolfe method and its variants. However, [5] only elaborates on the simpler version of Problem (17) without the extra functions a_{ix_i}. 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 practitioners.

Bach [5] notes that by duality, Problem (17) 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 [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 $D = \mathrm{diag}(r)$, the $(k-1) \times (k-1)$ matrix with diagonal entries r_j. Our problem can then be written as

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

Completing the square, the above problem is equivalent to

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

This last expression is precisely the problem which is called weighted isotonic regression: we are fitting ρ to $\mathrm{diag}(r^{-1})w$, with weights r, subject to a monotonicity constraint. Weighted isotonic regression is solved efficiently via the Pool Adjacent Violators algorithm of [13].

C.2 Runtime

Frank-Wolfe returns an ε-suboptimal solution in $O(\varepsilon^{-1} D^2 L)$ iterations, where D is the diameter of the feasible region, and L is the Lipschitz constant for the gradient of the objective [43]. Our optimization problem is $\max_{w \in B(H)} f(w)$ as defined in the previous section. Each $w \in B(H)$ has $O(n\delta^{-1})$ coordinates of the form $H^\delta(x + e_i) - H^\delta(x)$. Since H^δ is an expected influence in the range $[0, T]$, we can bound the magnitude of each coordinate of w by T and hence D^2 by $O(n\delta^{-1}T^2)$. If α is the minimum derivative of the functions R_i, then the smallest coefficient of the functions $a_{ix_i}(t)$ is bounded below by $\alpha\delta$. Hence the objective is the conjugate of an $\alpha\delta$-strongly convex function, and therefore has $\alpha^{-1}\delta^{-1}$-Lipschitz gradient. Combining these, we arrive at the $O(\varepsilon^{-1} n\delta^{-2}\alpha^{-1}T^2)$ iteration bound. The most expensive step in each iteration is computing the subgradient, which requires sorting the $O(n\delta^{-1})$ elements of ρ in time $O(n\delta^{-1}\log n\delta^{-1})$. Hence the total runtime of Frank-Wolfe is $O(\varepsilon^{-1} n^2\delta^{-3}\alpha^{-1}T^2 \log n\delta^{-1})$.

As specified in the main text, relating an approximate solution of (17) to a solution of (14) is nontrivial. Assume ρ^* has distinct elements separated by η, and chose ε to be less than $\eta^2\alpha\delta/8$. If ρ is ε-suboptimal, then by $\alpha\delta$-strong convexity we must have $\|\rho - \rho^*\|_2 < \eta/2$, and therefore $\|\rho - \rho^*\|_\infty < \eta/2$. Since the smallest consecutive gap between elements of ρ^* is η, this implies that ρ and ρ^* have the same ordering, and therefore admit the same solution x after thresholding. Accounting for this choice in ε, we have an exact solution to (14) in total runtime of $O(\eta^{-2} n^2\delta^{-4}\alpha^{-2}T^2 \log n\delta^{-1})$.