Robust Submodular Minimization with Applications to Cooperative Modeling

Rishabh Iyer [1]

Abstract. Robust Optimization is becoming increasingly important in machine learning applications. This paper studies the problem of robust submodular minimization subject to combinatorial constraints. Constrained Submodular Minimization arises in several applications such as co-operative cuts in image segmentation, co-operative matchings in image correspondence etc. Many of these models are defined over clusterings of data points (for example pixels in images), and it is important for these models to be robust to perturbations and uncertainty in the data. While several existing papers have studied robust submodular maximization, ours is the first work to study the minimization version under a broad range of combinatorial constraints including cardinality, knapsack, matroid as well as graph based constraints such as cuts, paths, matchings and trees. In each case, we provide scalable approximation algorithms and also study hardness bounds. Finally, we empirically demonstrate the utility of our algorithms on synthetic and real world datasets.

1 Introduction

Submodular functions provide a rich class of expressible models for a variety of machine learning problems. They occur naturally in two flavors. In minimization problems, they model notions of cooperation, attractive potentials, and economies of scale, while in maximization problems, they model aspects of coverage, diversity, and information. A set function $f : 2^V \rightarrow \mathbb{R}$ over a finite set $V = \{1, 2, \ldots, n\}$ is *submodular* [13] if for all subsets $S, T \subseteq V$, it holds that $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$. Given a set $S \subseteq V$, we define the *gain* of an element $j \notin S$ in the context S as $f(j|S) = f(S \cup j) - f(S)$. A perhaps more intuitive characterization of submodularity is as follows: a function f is submodular if it satisfies *diminishing marginal returns*, namely $f(j|S) \geq f(j|T)$ for all $S \subseteq T, j \notin T$, and is *monotone* if $f(j|S) \geq 0$ for all $j \notin S, S \subseteq V$.

Two central optimization problems involving submodular functions are submodular minimization and submodular maximization. Moreover, it is often natural to want to optimize these functions subject to combinatorial constraints [44, 24, 30, 15].

In this paper, we shall study the problem of robust submodular optimization. Often times in applications we want to optimize several objectives (or criteria) together. There are two natural formulations of this. One is the average case, where we can optimize the (weighted) sum of the submodular functions. Examples of this have been studied in data summarization applications [40, 48, 18]. The other is robust or worst case, where we want to minimize (or maximize) the maximum (equivalently minimum) among the functions. Examples of this have been proposed for sensor placement and observation selection [37]. Robust or worst case optimization is becoming increasingly important since solutions achieved by minimization and maximization can be unstable to perturbations in data. Often times submodular functions in applications are instantiated from various properties of the data (features, similarity functions, clusterings etc.) and obtaining

results which are robust to perturbations and variations in this data is critical.

Given monotone submodular functions $f_1, f_2, \cdots, f_l$ to minimize, this paper studies the following optimization problem

$$\text{ROBUST-SUBMIN:} \quad \min_{X \in \mathcal{C}} \max_{i=1:l} f_i(X) \qquad (1)$$

where $\mathcal{C}$ stands for combinatorial constraints, which include cardinality, matroid, spanning trees, cuts, s-t paths etc. We shall call this problems ROBUST-SUBMIN. A closely related problem is the maximization version of this problem called ROBUST-SUBMAX as:

$$\text{ROBUST-SUBMAX:} \quad \max_{X \in \mathcal{C}} \min_{i=1:l} f_i(X) \qquad (2)$$

Note that when $l = 1$, ROBUST-SUBMIN becomes constrained submodular minimization and ROBUST-SUBMAX becomes constrained submodular maximization.

1.1 Motivating Applications

This section provides an overview of two specific applications which motivate ROBUST-SUBMIN.

Figure 1. An illustration of co-operative cuts [30] and cooperative matchings [27].

Robust Co-operative Cuts: Markov random fields with pairwise attractive potentials occur naturally in modeling image segmentation and related applications [5]. While models are tractably solved using graph-cuts, they suffer from the shrinking bias problem, and images with elongated edges are not segmented properly. When modeled via a submodular function, however, the cost of a cut is not just the sum of the edge weights, but a richer function that allows cooperation between edges, and yields superior results on many challenging tasks (see, for example, the results of the image segmentations in [31]). This was achieved in [31] by partitioning the set of edges $\mathcal{E}$ of the grid graph into groups of similar edges (or types) $\mathcal{E}_1, \cdots, \mathcal{E}_k$, and defining a function $f(S) = \sum_{i=1}^{k} \psi_i(w(S \cap \mathcal{E}_i)), S \subseteq \mathcal{E}$, where ψ_is are concave functions and w encodes the edge potentials. This ensures that we offer a *discount* to edges of the same type. The left image in Figure 1.1 illustrates co-operative cuts [30]. However, instead of taking a single clustering (i.e. a single group of edges $\mathcal{E}_1, \cdots, \mathcal{E}_k$), one can instantiate several clusterings $\{(\mathcal{E}_1^1, \cdots, \mathcal{E}_k^1), \cdots, (\mathcal{E}_1^l, \cdots, \mathcal{E}_k^l)\}$ and define a robust objective:

[1] University of Texas at Dallas, USA, email: rishabh.iyer@utdallas.edu

$f_{robust}(S) = \max_{i=1:l} \sum_{j=1}^{k} \psi_j(w(S \cap \mathcal{E}_j^i)), S \subseteq \mathcal{E}$. Minimizing f_{robust} over the family of s-t cuts can achieve segmentations that are robust to many such groupings of edges thereby achieving *robust co-operative segmentations*. We call this *Robust Co-operative Cuts* and this becomes instance of ROBUST-SUBMIN with f_{robust} defined above and the constraint $\mathcal{C}$ being the family of s-t cuts.

Robust Co-operative Matchings: The simplest model for matching key-points in pairs of images (which is also called the correspondence problem) can be posed as a bipartite matching, also called the assignment problem. These models, however, do not capture interaction between the pixels. One kind of desirable interaction is that similar or neighboring pixels be matched together. We can achieve this as follows. First we cluster the key-points in the two images into k groups and the induced clustering of edges can be given a discount via a submodular function. This model has demonstrated improved results for image correspondence problems [27]. The right image in Figure 1.1 illustrates co-operative matchings [27]. Similar to co-operative cuts, it makes sense to be robust among the different clusterings of pixels and minimize the worst case clustering. We call this *Robust Co-operative Matchings*. For more details of this model, please see the experimental section of this manuscript.

1.2 Related Work

Submodular Minimization and Maximization: Most Constrained forms of submodular optimization (minimization and maximization) are NP hard even when f is monotone [44, 51, 15, 30]. The greedy algorithm achieves a $1 - 1/e$ approximation for cardinality constrained maximization and a $1/2$ approximation under matroid constraints [12]. [8] achieved a tight $1 - 1/e$ approximation for matroid constraints using the continuous greedy. [38] later provided a similar $1 - 1/e$ approximation under multiple knapsack constraints. Constrained submodular minimization is much harder – even with simple constraints such as a cardinality lower bound constraints, the problems are not approximable better than a polynomial factor of $\Omega(\sqrt{n})$ [47]. Similarly the problems of minimizing a submodular function under covering constraints [19], spanning trees, perfect matchings, paths [14] and cuts [30] have similar polynomial hardness factors. In all of these cases, matching upper bounds (i.e approximation algorithms) have been provided [47, 19, 14, 30]. In [24, 23], the authors provide a scalable semi-gradient based framework and curvature based bounds which improve upon the worst case polynomial factors for functions with bounded curvature κ_f (which several submodular functions occurring in real world applications have).

Robust Submodular Maximization: Unlike ROBUST-SUBMIN, ROBUST-SUBMAX has been extensively studied in literature. One of the first papers to study robust submodular maximization was [37], where the authors study ROBUST-SUBMAX with cardinality constraints. The authors reduce this problem to a submodular set cover problem using the *saturate* trick to provide a bi-criteria approximation guarantee. [3] extend this work and study ROBUST-SUBMAX subject to matroid constraint. They provide bi-criteria algorithms by creating a union of $O(\log l/\epsilon)$ independent sets, with the union set having a guarantee of $1 - \epsilon$. They also discuss extensions to knapsack and multiple matroid constraints and provide bicriteria approximation of $(1 - \epsilon, O(\log l/\epsilon))$. [46] also study the same problem. However, they take a different approach by presenting a bi-criteria algorithm that outputs a feasible set that is good only for a fraction of the k submodular functions g_i. [9, 50] study a slightly general problem of robust non-convex optimization (of which robust submodular optimization is a special case), but they provide weaker guarantees compared to [36, 3].

Robust Min-Max Combinatorial Optimization: From a minimization perspective, several researchers have studied robust min-max combinatorial optimization (a special case of ROBUST-SUBMIN with modular functions) under different combinatorial constraints (see [1, 34] for a survey). Unfortunately these problems

are NP hard even for constraints such as knapsack, s-t cuts, s-t paths, assignments and spanning trees where the standard linear cost problems are poly-time solvable [1, 34]. Moreover, the lower bounds on hardness of approximation is $\Omega(log^{1-\epsilon}l)$ (l is the number of functions) for s-t cuts, paths and assignments [32] and $\Omega(log^{1-\epsilon}n)$ for spanning trees [33] for any $\epsilon > 0$. For the case when l is bounded (a constant), fully polynomial time approximation schemes have been proposed for a large class of constraints including s-t paths, knapsack, assignments and spanning trees [2, 1, 34]. From an approximation algorithm perspective, the best known general result is an approximation factor of l for constraints where the linear function can be exactly optimized in poly-time. For special cases such as spanning trees and shortest paths, one can achieve improved approximation factors of $O(\log n)$ [33, 35] and $\bar{O}(\sqrt{n})^2$ [35] respectively.

1.3 Our Contributions

While past work has mainly focused on ROBUST-SUBMAX, This paper is the first work to provide approximation bounds for ROBUST-SUBMIN. We start by first providing the hardness results. These results easily follow from the hardness bounds of constrained submodular minimization and robust min-max combinatorial optimization. Next, we provide four families of algorithms for ROBUST-SUBMIN. The first and simplest approach approximates the max among the functions $f_1, \cdots, f_l$ in ROBUST-SUBMIN with the average. This in turn converts ROBUST-SUBMIN into a constrained submodular minimization problem. We show that solving the constrained submodular minimization with the average of the functions still yields an approximation factor for ROBUST-SUBMIN though it is worse compared to the hardness by a factor of l. While this approach is conceptually very simple, we do not expect this perform well in practice. We next study the Majorization-Minimization family of algorithms, where we sequentially approximate the functions $f_1, \cdots, f_l$ with their modular upper bounds. Each resulting sub-problem involves solving a min-max combinatorial optimization problem and we use specialized solvers for the different constraints. The third algorithm is the Ellipsoidal Approximation (EA) where we replace the functions f_i's with their ellipsoidal approximations. We show that this achieves the tightest approximation factor for several constraints, but is slower to the majorization-minimization approach in practice. The fourth technique is a continuous relaxation approach where we relax the discrete problem into a continuous one via the Lovász extension. We show that the resulting robust problem becomes a convex optimization problem with an appropriate recast of the constraints. We then provide a rounding scheme thereby resulting in approximation factors for ROBUST-SUBMIN. Tables 1 and 2 show the hardness and the resulting approximation factors for several important combinatorial constraints. Along the way, we also provide a family of approximation algorithms and heuristics for ROBUST-MIN (i.e. with modular functions f_i under several combinatorial constraints. We finally compare the four algorithms for different constraints and in each case, discuss heuristics which will make the resulting algorithms practical and scalable for real world problems. Finally, we empirically show the performance of our algorithms on synthetic and real world datasets. We demonstrate the utility of our models in robust co-operative matching and show that our robust model outperforms simple co-operative model from [27] for this task.

2 Main Ideas and Techniques

In this section, we will review some of the constructs and techniques used in this paper to provide approximation algorithms for ROBUST-SUBMIN.

---

[2] Ignores log factors

Constraint	Hardness	MMin-AA	EA-AA	MMin	EA	CR
Cardinality (k)	$K(\sqrt{n}, \kappa)$	$lK(k,\kappa)$	$lK(\sqrt{n}\log n, \kappa)$	$O(\log lK(k, \kappa_{wc})/\log\log l)$	$\tilde{O}(\sqrt{m}\log l/\log\log l)$	$n-k+1$
Trees	$M(K(n,\kappa), \log n)$	$lK(n,\kappa)$	$lK(\sqrt{m}\log m, \kappa)$	$O(\min(\log n, l)K(n, \kappa_{wc}))$	$O(\min(\sqrt{\log n}, \sqrt{l})\sqrt{m}\log m)$	$m-n+1$
Matching	$M(K(n,\kappa), \log l)$	$lK(n,\kappa)$	$lK(\sqrt{m}\log m, \kappa)$	$lK(n,\kappa)$	$\tilde{O}(\sqrt{lm})$	n
s-t Cuts	$M(K(\sqrt{m},\kappa), \log l)$	$lK(n,\kappa)$	$lK(\sqrt{m}\log m, \kappa)$	$lK(n,\kappa)$	$\tilde{O}(\sqrt{lm})$	n
s-t Paths	$M(K(\sqrt{m},\kappa), \log l)$	$lK(n,\kappa)$	$lK(\sqrt{m}\log m, \kappa)$	$O(\min(\sqrt{n}, l)K(n, \kappa_{wc}))$	$O(\min(n^{0.25}, \sqrt{l})\sqrt{m}\log m)$	m
Edge Cov.	$K(n,\kappa)$	$lK(n,\kappa)$	$lK(\sqrt{m}\log m, \kappa)$	$lK(n,\kappa)$	$\tilde{O}(\sqrt{lm})$	n
Vertex Cov.	2	$lK(n,\kappa)$	$lK(\sqrt{n}\log n, \kappa)$	$lK(n,\kappa)$	$\tilde{O}(\sqrt{ln})$	2

Table 1. Approximation bounds and Hardness for ROBUST-SUBMIN. $M(.)$ stands for $\max(.)$

Constraint	Hardness	MMin	EA
Knapsack	$K(\sqrt{n},\kappa)$	$K(n,\kappa)$	$O(K(\sqrt{n}\log n, \kappa))$
Trees	$K(n,\kappa)$	$K(n,\kappa)$	$O(K(\sqrt{m}\log m, \kappa))$
Matchings	$K(n,\kappa)$	$K(n,\kappa)$	$O(K(\sqrt{m}\log m, \kappa))$
s-t Paths	$K(n,\kappa)$	$K(n,\kappa)$	$O(K(\sqrt{m}\log m, \kappa))$

Table 2. Approximation bounds and Hardness of in ROBUST-SUBMIN with l constant

Curvature: Given a submodular function f, the curvature [49, 10, 23]:

$$\kappa_f = 1 - \min_{j \in V} \frac{f(j|V\setminus j)}{f(j)}$$

Thanks to submodularity, it is easy to see that $0 \leq \kappa_f \leq 1$. We define a quantity $K(f,\kappa) = f/(1 + (1 - \kappa)(f - 1))$, where κ is the curvature. Note that $K(f,\kappa)$ interpolates between $K(f,1) = f$ and $K(f,0) = 1$. This quantity shall repeatedly come up in the approximation and hardness bounds in this paper.

The Submodular Polyhedron and Lovász extension For a submodular function f, the submodular polyhedron $\mathcal{P}_f$ and the corresponding base polytope $\mathcal{B}_f$ are respectively defined as $\mathcal{P}_f = \{x : x(S) \leq f(S), \forall S \subseteq V\}$ $\mathcal{B}_f = \mathcal{P}_f \cap \{x : x(V) = f(V)\}$. For a vector $x \in \mathbb{R}^V$ and a set $X \subseteq V$, we write $x(X) = \sum_{j \in X} x(j)$. Though $\mathcal{P}_f$ is defined via 2^n inequalities, its extreme point can be easily characterized [13, 11]. Given any permutation σ of the ground set $\{1, 2, \cdots, n\}$, and an associated chain $\emptyset = S_0^\sigma \subseteq S_1^\sigma \subseteq \cdots \subseteq S_n^\sigma = V$ with $S_i^\sigma = \{\sigma(1), \sigma(2), \ldots, \sigma(i)\}$, a vector h_σ^f satisfying, $h_\sigma^f(\sigma(i) = f(S_i^\sigma) - f(S_{i-1}^\sigma) = f(\sigma(i)|S_{i-1}^\sigma), \forall i = 1, \cdots, n$ forms an extreme point of $\mathcal{P}_f$. Moreover, a natural convex extension of a submodular function, called the Lovász extension [42, 11] is closely related to the submodular polyhedron, and is defined as $\hat{f}(x) = \max_{h \in \mathcal{P}_f} \langle h, x \rangle$. Thanks to the properties of the polyhedron, $\hat{f}(x)$ can be efficiently computed: Denote σ_x as an ordering induced by x, such that $x(\sigma_x(1)) \geq x(\sigma_x(2)) \geq \cdots x(\sigma_x(n))$. Then the Lovász extension is $\hat{f}(x) = \langle h_{\sigma_x}^f, x \rangle$ [42, 11]. The gradient of the Lovász extension $\nabla \hat{f}(x) = h_{\sigma_x}^f$.

Modular lower bounds (Sub-gradients): Akin to convex functions, submodular functions have tight modular lower bounds. These bounds are related to the sub-differential $\partial_f(Y)$ of the submodular set function f at a set $Y \subseteq V$, which is defined [13] as: $\partial_f(Y) = \{y \in \mathbb{R}^n : f(X) - y(X) \geq f(Y) - y(Y), \text{ for all } X \subseteq V\}$. Denote a sub-gradient at Y by $h_Y \in \partial_f(Y)$. Define $h_Y = h_{\sigma_Y}^f$ (see the definition of h_σ^f from the previous paragraph) forms a lower bound of f, tight at Y — i.e., $h_Y(X) = \sum_{j \in X} h_Y(j) \leq f(X), \forall X \subseteq V$ and $h_Y(Y) = f(Y)$. Notice that the extreme points of a sub-differential are a subset of the extreme points of the submodular polyhedron.

Modular upper bounds (Super-gradients): We can also define super-differentials $\partial^f(Y)$ of a submodular function [31, 20] at Y: $\partial^f(Y) = \{y \in \mathbb{R}^n : f(X) - y(X) \leq f(Y) - y(Y); \text{ for all } X \subseteq V\}$. It is possible, moreover, to provide specific super-gradients [20,

24, 26, 22, 29] that define the following two modular upper bounds:

$$m_{X,1}^f(Y) \triangleq f(X) - \sum_{j \in X\setminus Y} f(j|X\setminus j) + \sum_{j \in Y\setminus X} f(j|\emptyset), \quad (3)$$

$$m_{X,2}^f(Y) \triangleq f(X) - \sum_{j \in X\setminus Y} f(j|V\setminus j) + \sum_{j \in Y\setminus X} f(j|X). \quad (4)$$

Then $m_{X,1}^f(Y) \geq f(Y)$ and $m_{X,2}^f(Y) \geq f(Y), \forall Y \subseteq V$ and $m_{X,1}^f(X) = m_{X,2}^f(X) = f(X)$. Also note that $m_{\emptyset,1}^f(Y) = m_{\emptyset,2}^f(Y) = \sum_{j \in Y} f(j|\emptyset)$. For simplicity denote this as $m_\emptyset^f(X)$. Then the following result holds:

Lemma 1 *[24, 23] Given a submodular function f with curvature κ_f, $f(X) \leq m_\emptyset^f(X) \leq K(|X|, \kappa_f)f(X)$*

Ellipsoidal Approximation: Another generic approximation of a submodular function, introduced by Goemans et. al [16], is based on approximating the submodular polyhedron by an ellipsoid. The main result states that any polymatroid (monotone submodular) function f, can be approximated by a function of the form $\sqrt{w^f(X)}$ for a certain modular weight vector $w^f \in \mathbb{R}^V$, such that $\sqrt{w^f(X)} \leq f(X) \leq O(\sqrt{n}\log n)\sqrt{w^f(X)}, \forall X \subseteq V$. A simple trick then provides a curvature-dependent approximation [23] — we define the κ_f-*curve-normalized* version of f as follows: $f^\kappa(X) \triangleq [f(X) - (1 - \kappa_f)\sum_{j \in X} f(j)]/\kappa_f$. Then, the submodular function $f^{ea}(X) = \kappa_f\sqrt{w^{f^\kappa}(X)} + (1 - \kappa_f)\sum_{j \in X} f(j)$ satisfies [23]:

$$f^{ea}(X) \leq f(X) \leq O\left(K(\sqrt{n}\log n, \kappa_f)\right)f^{ea}(X), \forall X \subseteq V \quad (5)$$

3 Algorithms and Hardness Results

In this section, we shall go over the hardness and approximation algorithms for ROBUST-SUBMIN. We shall consider two cases, one where l is bounded (i.e. its a constant), and the other where l is unbounded. We start with some notation. We denote the graph as $G = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n, |\mathcal{E}| = m$. Depending on the problem at hand, the ground set V can either be the set of edges ($V = \mathcal{E}$) or the set of vertices ($V = \mathcal{V}$). The groundset is the set of edges in the case of trees, matchings, cuts, paths and edge covers, while in the case of vertex covers, they are defined on the vertices.

3.1 Hardness of ROBUST-SUBMIN

Since ROBUST-SUBMIN generalizes robust min-max combinatorial optimization (when the functions are modular), we have the hardness bounds from [32, 33]. For the modular case, the lower bounds are $\Omega(log^{1-\epsilon}l)$ (l is the number of functions) for s-t cuts, paths and assignments [32] and $\Omega(log^{1-\epsilon}n)$ for spanning trees [33] for any $\epsilon > 0$. These hold unless $NP \subseteq DTIME(n^{poly \log n})$[32, 33]. Moreover, since ROBUST-SUBMIN also generalizes constrained submodular minimization, we get the curvature based hardness bounds from [23, 15, 30]. The hardness results are in the first column of Table 1. The curvature κ corresponds to the worst curvature among the functions f_i (i.e. $\kappa = \max_i \kappa_i$).

3.2 Algorithms with Modular functions f_i's

In this section, we shall study approximation algorithms for ROBUST-SUBMIN when the functions f_i's are modular, i.e. $f_i(X) = \sum_{j \in X} f_i(j)$. We call this problem ROBUST-MIN. Most previous work [2, 1, 34] has focused on fully polynomial approximation schemes when l is small. These algorithms are exponential in l and do not scale beyond $l = 3$ or 4. Instead, we shall study approximation algorithms for this problem. Define two simple approximations of the function $F(X) = \max_i f_i(X)$ when f_i's are modular. The first is $\hat{F}(X) = \sum_{i \in X} \max_{j=1:l} f_j(i)$ and the second is $\tilde{F}(X) = 1/l \sum_{i=1}^{l} \sum_{j \in X} f_i(j) = 1/l \sum_{i=1}^{l} f_i(X)$.

Lemma 2 *Given $f_i(j) \geq 0$, it holds that $\hat{F}(X) \geq F(X) \geq \frac{1}{l}\hat{F}(X)$. Furthermore, $\tilde{F}(X) \leq F(X) \leq l\tilde{F}(X)$. Given a β-approximate algorithm for optimizing linear cost functions over the constraint $\mathcal{C}$, we can achieve an $l\beta$-approximation algorithm for ROBUST-MIN.*

Proof: We first prove the second part. Its easy to see that $F(X) = \max_i f_i(X) \leq \sum_i f_i(X) = l\tilde{F}(X)$ which is the second inequality. The first inequality follows from the fact that $\sum_i f_i(X) \leq l \max_i f_i(X)$. To prove the first result, we start with proving $F(X) \leq \hat{F}(X)$. For a given set X, let i_X be the index which maximizes F so $F(X) = \sum_{j \in X} f_{i_X}(j)$. Then $f_{i_X}(j) \leq \max_i f_i(j)$ from which we get the result. Next, observe that $\hat{F}(X) \leq \sum_{i=1}^{l} f_i(X) = l\tilde{F}(X) \leq lF(X)$ which follows from the inequality corresponding to $\tilde{F}$.

Now given a β-approximation algorithm for optimizing linear cost functions over the constraint $\mathcal{C}$, denote $\tilde{X}$ by optimizing $\tilde{F}(X)$ over $\mathcal{C}$. Also denote X^* as the optimal solution by optimizing F over $\mathcal{C}$ and $\tilde{X}^*$ be the optimal solution for optimizing $\tilde{F}$ over $\mathcal{C}$. Since $\tilde{X}$ is a β approximation for $\tilde{F}$ over $\mathcal{C}$, $\tilde{F}(\tilde{X}) \leq \beta\tilde{F}(\tilde{X}^*) \leq \beta\tilde{F}(X^*)$. The last inequality holds since X^* is feasible (i.e. it belongs to $\mathcal{C}$) so it must hold that $\tilde{F}(X^*) \geq \tilde{F}(\tilde{X}^*)$. A symmetric argument applies to $\hat{F}$. In particular, $F(\hat{X}) \leq \hat{F}(\hat{X}) \leq \beta\hat{F}(X^*) \leq \beta lF(X^*)$. The inequality $\hat{F}(\hat{X}) \leq \beta\hat{F}(X^*)$ again holds due to an argument similar to the $\tilde{F}$ case. $\square$

Note that for most constraints $\mathcal{C}$, β is 1. In practice, we can take the better of the two solutions obtained from the two approximations above.

Next, we shall look at higher order approximations of F. Define the function $F_a(X) = (\sum_{i=1}^{l}[f_i(X)]^a)^{\frac{1}{a}}$. Its easy to see that $F_a(X)$ comes close to F as a becomes large.

Lemma 3 *Define $F_a(X) = (\sum_{i=1}^{l}[f_i(X)]^a)^{\frac{1}{a}}$. Then it holds that $F(X) \leq F_a(X) \leq l^{\frac{1}{a}}F(X)$. Moreover, given a β-approximation algorithm for optimizing $F_a(X)$ over $\mathcal{C}$, we can obtain a $\beta l^{\frac{1}{a}}$-approximation for ROBUST-MIN.*

Proof: Its easy to see that $F(X) \leq F_a(X)$ since $[F(X)]^a \leq \sum_{i=1}^{l}[f_i(X)]^a$. Next, note that $\forall i, f_i(X) \leq F(X)$ and hence $\sum_{i=1}^{l}[f_i(X)]^a \leq l[F(X)]^a$ which implies that $F_a(X) \leq k^{\frac{1}{a}}F(X)$ which proves the result. The second part follows from arguments similar to the Proof of Lemma 2. $\square$

Note that optimizing $F_a(X)$ is equivalent to optimizing $\sum_{i=1}^{l}[f_i(X)]^a$ which is a higher order polynomial function. For example, when $a = 2$ we get the quadratic combinatorial program [6] which can possibly result in a a $\sqrt{l}$ approximation given a exact or approximate quadratic optimizer. Unfortunately, optimizing this problem is NP hard for most constraints when $a \geq 2$. However, several heuristics and efficient solvers [4, 39, 6, 41] exist for solving this in the quadratic setting. Moreover, for special cases such as the assignment (matching) problem, specialized algorithms such as the Graduated assignment algorithm [17] exist for solving this. While these are not guaranteed to theoretically achieve the optimal solution, they work quite in practice, thus yielding a family of heuristics for ROBUST-MIN.

3.3 Average Approximation based Algorithms

In this section, we shall look at a simple approximation of $\max_i f_i(X)$, which in turn shall lead to an approximation algorithm for ROBUST-SUBMIN. We first show that $f_{avg}(X) = 1/l \sum_{i=1}^{l} f_i(X)$ is an l-approximation of $\max_{i=1:l} f_i(X)$, which implies that minimizing $f_{avg}(X)$ implies an approximation for ROBUST-SUBMIN.

Theorem 4 *Given a non-negative set function f, define $f_{avg}(X) = \frac{1}{l}\sum_{i=1}^{l} f_i(X)$. Then $f_{avg}(X) \leq \max_{i=1:l} f_i(X) \leq lf_{avg}(X)$. Denote $\hat{X}$ as β-approximate optimizer of f_{avg}. Then $\max_{i=1:l} f_i(\hat{X}) \leq l\beta \max_{i=1:l} f_i(X^*)$ where X^* is the exact minimizer of ROBUST-SUBMIN.*

Proof: To prove the first part, notice that $f_i(X) \leq \max_{i=1:l} f_i(X)$, and hence $1/l\sum_i f_i(X) \leq \max_{i=1:l} f_i(X)$. The other inequality also directly follows since the f_i's are non-negative and hence $\max_{i=1:l} f_i(X) \leq \sum_i f_i(X) = lf_{avg}(X)$. To prove the second part, observe that $\max_i f_i(\hat{X}) \leq lf_{avg}(\hat{X}) \leq l\beta f_{avg}(X^*) \leq l\beta \max_i f_i(X^*)$. The first inequality holds from the first part of this result, the second inequality holds since $\hat{X}$ is a β-approximate optimizer of f_{avg} and the third part of the theorem holds again from the first part of this result. $\square$

Since f_{avg} is a submodular function, we can use the majorization-minimization (which we call MMin-AA) and ellipsoidal approximation (EA-AA) for constrained submodular minimization [24, 23, 15, 30]. The following corollary provides the approximation guarantee of MMin-AA and EA-AA for ROBUST-SUBMIN.

Corollary 5 *Using the majorization minimization (MMin) scheme with the average approximation achieves an approximation guarantee of $lK(|X^*|, \kappa_{avg})$ where X^* is the optimal solution of ROBUST-SUBMIN and κ_{avg} is the curvature of f_{avg}. Using the curvature-normalized ellipsoidal approximation algorithm from [23, 16] achieves a guarantee of $O(lK(|\mathcal{V}|\log|\mathcal{V}|, \kappa_{avg}))$*

This corollary directly follows by combining the approximation guarantee of MMin and EA [23, 24] with Theorem 1. Substituting the values of $|\mathcal{V}|$ and $|X^*|$ for various constraints, we get the results in Table 1. While the average case approximation method provides a bounded approximation guarantee for ROBUST-SUBMIN, it defeats the purpose of the robust formulation. Moreover, the approximation factor is worse by a factor of l. Below, we shall study some techniques which directly try to optimize the robust formulation.

3.4 Majorization-Minimization Algorithm

The Majorization-Minimization algorithm is a sequential procedure which uses upper bounds of the submodular functions defined via supergradients. Starting with $X^0 = \emptyset$, the algorithm proceeds as follows. At iteration t, it constructs modular upper bounds for each function f_i, $m_{X^t}^{f_i}$ which is tight at X^t. We can use either one of the two modular upper bounds defined in Section 2. The set $X^{t+1} = \operatorname{argmin}_{X \in \mathcal{C}} \max_i m_{X^t}^{f_i}(X)$. This is a min-max robust optimization problem. The following theorem provides the approximation guarantee for MMin.

Theorem 6 *If l is a constant, MMin achieves an approximation guarantee of $(1 + \epsilon)K(|X^*|, \kappa_{wc})$ for the knapsack, spanning trees, matching and s-t path problems. The complexity of this algorithm is exponential in l. When l is unbounded, MMin achieves an approximation guarantee of $lK(|X^*|, \kappa_{wc})$. For spanning trees and shortest path constraints, MMin achieves a $O(\min(\log n, l)K(n, \kappa_{wc}))$ and a $O(\min(\sqrt{n}, l)K(n, \kappa_{wc}))$ approximation. Under cardinality and partition matroid constraints, MMin achieves a $O(\log lK(n, \kappa_{wc})/\log \log l)$ approximation.*

Substituting the appropriate bounds on $|X^*|$ for the various constraints, we get the results in Tables 1 and 2. κ_{wc} corresponds to the worst case curvature $\max_i \kappa_{f_i}$.

We now elaborate on the Majorization-Minimization algorithm. At every round of the majorization-minimization algorithm we need to solve

$$X^{t+1} = \operatorname{argmin}_{X \in \mathcal{C}} \max_i m_{X^t}^{f_i}(X). \tag{6}$$

We consider three cases. The first is when l is a constant. In that case, we can use an FPTAS to solve Eq. (6) [1, 2]. We can obtain a $1 + \epsilon$ approximation with complexities of $O(n^{l+1}/\epsilon^{l-1})$ for shortest paths, $O(mn^{l+4}/\epsilon^l \log n/\epsilon)$ for spanning trees, $O(mn^{l+4}/\epsilon^l \log n/\epsilon)$ for matchings and $O(n^{l+1}/\epsilon^l)$ for knapsack [2]. The results for constant l is shown in Table 2 (column corresponding to MMin). The second case is a generic algorithm when l is not constant. Note that we cannot use the FPTAS since they are all exponential in l. In this case, at every iteration of MMin can use the framework of approximations discussed in Section 3.2, and choose the solution with a better objective value. In particular, if we use the two modular bounds of the $\max$ function (i.e. the average the the max bounds), we can obtain l-approximations of $\max_i m_{X^t}^{f_i}$. One can also use the quadratic and cubic bounds which can possibly provide $\sqrt{l}$ or $l^{0.3}$ bounds. While these higher order bounds are still theoretically NP hard, there exist practically efficient solvers for various constraints (for e.g. quadratic assignment model for matchings [41]). Finally, for the special cases of spanning trees, shortest paths, cardinality and partition matroid constraints, there exist LP relaxation based algorithms which achieve approximation factors of $O(\log n)$, $O(\sqrt{n \log l/\log \log l}$, $O(\log l/\log \log l)$ and $O(\log l/\log \log l)$ respectively. The approximation guarantees of MMin for unbounded l is shown in Table 1.

We now prove Theorem 6.

Proof: Assume we have an α approximation algorithm for solving problem (6). We start MMin with $X^0 = \emptyset$. We prove the bound for MMin for the first iteration. Observe that $m_\emptyset^{f_i}(X)$ approximate the submodular functions $f_i(X)$ up to a factor of $K(|X|, \kappa_i)$ [23]. If κ_{wc} is the maximum curvature among the functions f_i, this means that $m_\emptyset^{f_i}(X)$ approximate the submodular functions $f_i(X)$ up to a factor of $K(|X|, \kappa_{wc})$ as well. Hence $\max_i m_\emptyset^{f_i}(X)$ approximates $\max_i f_i(X)$ with a factor of $K(|X|, \kappa_{wc})$. In other words, $\max_i f_i(X) \leq \max_i m_\emptyset^{f_i}(X) \leq K(|X|, \kappa_{wc}) \max_i f_i(X)$. Let $\hat{X}_1$ be the solution obtained by optimizing $m_\emptyset^{f_i}$ (using an α-approximation algorithms for the three cases described

above). It holds that $\max_i m_\emptyset^{f_i}(\hat{X}_1) \leq \alpha \max_i m_\emptyset^{f_i}(X_1^m)$ where X_1^m is the optimal solution of $\max_i m_\emptyset^{f_i}(X)$ over the constraint $\mathcal{C}$. Furthermore, denote X^* as the optimal solution of $\max_i f_i(X)$ over $\mathcal{C}$. Then $\max_i m_\emptyset^{f_i}(X_1^m) \leq \max_i m_\emptyset^{f_i}(X^*) \leq K(|X^*|, \kappa_{wc}) \max_i f_i(X^*)$. Combining both, we see that $\max_i f(\hat{X}_1) \leq \max_i m_\emptyset^{f_i}(\hat{X}_1) \leq \alpha K(|X^*|, \kappa_{wc}) \max_i f_i(X^*)$. We then run MMin for more iterations and only continue if the objective value increases in the next round. Using the values of α for the different cases above, we get the results. $\square$

3.5 Ellipsoidal Approximation Based Algorithm

Next, we use the Ellipsoidal Approximation to approximate the submodular function f_i. To account for the curvature of the individual functions f_i's, we use the curve-normalized Ellipsoidal Approximation [23]. We then obtain the functions $\hat{f}_i(X)$ which are of the form $(1 - \kappa_{f_i})\sqrt{w_{f_i}(X)} + \kappa_{f_i} \sum_{j \in X} f_i(j)$, and the problem is then to optimize $\max_i \hat{f}_i(X)$ subject to the constraints $\mathcal{C}$. This is no longer a min-max optimization problem. The following result shows that we can still achieve approximation guarantees in this case.

Theorem 7 *For the case when l is a constant, EA achieves an approximation guarantee of $O(K(\sqrt{|\mathcal{V}|}\log|\mathcal{V}|, \kappa_{wc}))$ for the knapsack, spanning trees, matching and s-t path problems. The complexity of this algorithm is exponential in l. When l is unbounded, the EA algorithm achieves an approximation guarantee of $O(\sqrt{l}\sqrt{|\mathcal{V}|}\log|\mathcal{V}|)$ for all constraints. In the case of spanning trees, shortest paths, the EA achieves approximation factors of $O(\min(\sqrt{\log n}, \sqrt{l})\sqrt{m}\log m)$, and $O(\min(n^{0.25}, \sqrt{l})\sqrt{m}\log m)$. Under cardinality and partition matroid constraints, EA achieves a $O(\sqrt{\log l/\log \log l}\sqrt{n}\log n)$ approximation.*

For the case when l is bounded, we reduce the optimization problem after the Ellipsoidal Approximation into a multi-objective optimization problem, which provides an FPTAS for knapsack, spanning trees, matching and s-t path problems [45, 43]. When l is unbounded, we further reduce the EA approximation objective into a linear objective which then provides the approximation guarantees similar to MMin above. However, as a result, we loose the curvature based guarantee.

Proof: First we start with the case when l is a constant. Observe that the optimization problem is

$$\min_{X \in \mathcal{C}} \max_i \hat{f}_i(X) = \min_{X \in \mathcal{C}} \max_i (1 - \kappa_{f_i})\sqrt{w_{f_i}(X)} + \\ \kappa_{f_i} \sum_{j \in X} f_i(j)$$

This is of the form $\min_{X \in \mathcal{C}} \max_i \sqrt{w_1^i(X)} + w_2^i(X)$. Define $h(y_1^1, y_2^1, y_2^2, y_2^2, \cdots, y_1^l, y_2^l) = \max_i \sqrt{y_1^i} + y_2^i$. Note that the optimization problem is $\min_{X \in \mathcal{C}} h(w_1^1(X), w_2^1(X), \cdots, w_1^l(X), w_2^l(X))$. Observe that $h(\mathbf{y}) \leq h(\mathbf{y}')$ if $\mathbf{y} \leq \mathbf{y}'$. Furthermore, note that $\mathbf{y} \geq 0$. Then given a $\lambda > 1$, $h(\lambda \mathbf{y}) = \max_i \sqrt{\lambda y_1^i} + \lambda y_2^i \leq \lambda \sqrt{y_1^i} + \lambda y_2^i \leq \lambda h(\mathbf{y})$. As a result, we can use Theorem 3.3 from [43] which provides an FPTAS as long as the following exact problem can be solved on $\mathcal{C}$: Given a constant C and a vector $c \in \mathbf{R}^n$, does there exist a x such that $\langle c, x \rangle = C$? A number of constraints including matchings, knapsacks, s-t paths and spanning trees satisfy this [45]. For these constraints, we can obtain a $1 + \epsilon$ approximation algorithm in complexity exponential in l.

When l is unbounded, we directly use the Ellipsoidal Approximation and the problem then is to optimize $\min_{X \in \mathcal{C}} \max_i \sqrt{w_{f_i}(X)}$.

Constraints	$\hat{\mathcal{P}}_{\mathcal{C}}$
Matroids (includes Spanning Trees, Cardinality)	$\{x \in [0,1]^n, x(S) \geq r_{\mathcal{M}}(V) - r_{\mathcal{M}}(V \backslash S), \forall S \subseteq V\}$
Set Covers (includes Vertex Covers and Edge Covers)	$\{x \in [0,1]^{|\mathcal{S}|} \mid \sum_{i:u \in S_i} x(i) \geq c_u, \forall u \in \mathcal{U}\}$
s-t Paths	$\{x \in [0,1]^{|\mathcal{E}|} \mid \sum_{e \in C} x(e) \geq 1, \text{ for every s-t cut } C \subseteq \mathcal{E}\}$
s-t Cuts	$\{x \in [0,1]^{|\mathcal{E}|} \mid \sum_{e \in P} x(e) \geq 1, \text{ for every s-t path } P \subseteq \mathcal{E}\}$

Table 3. The Extended Polytope $\hat{\mathcal{P}}_{\mathcal{C}}$ for many of the combinatorial constraints discussed in this paper. See [25] for more details.

We then transform this to the following optimization problem: $\min_{X \in \mathcal{C}} \max_i w_{f_i}(X)$. Assume we can obtain an α approximation to the problem $\min_{X \in \mathcal{C}} \max_i w_{f_i}(X)$. This means we can achieve a solution $\hat{X}$ such that $\max_i w_{f_i}(\hat{X}) \leq \alpha \max_i w_{f_i}(X^{ea})$ where X^{ea} is the optimal solution for the problem $\min_{X \in \mathcal{C}} \max_i w_{f_i}(X)$. Then observe that $\max_i \sqrt{w_{f_i}(X^{ea})} \leq \max_i \sqrt{w_{f_i}(X^*)} \leq \max_i f_i(X^*)$. Combining all the inequalities and also using the bound of the Ellipsoidal Approximation, we have $\max_i f_i(\hat{X}) \leq \beta \max_i \sqrt{w_{f_i}(\hat{X})} \leq \beta\sqrt{\alpha} \max_i \sqrt{w_{f_i}(X^{ea})} \leq \beta\sqrt{\alpha} \max_i \sqrt{w_{f_i}(X^*)} \leq \beta\sqrt{\alpha} \max_i f_i(X^*)$ where β is the approximation of the Ellipsoidal Approximation.

We now use this result to prove the theorem. Consider two cases. First, we optimize the *avg* and *max* versions of $\max_i w_{f_i}(X)$ which provide $\alpha = l$ approximation. Secondly, for the special cases of spanning trees, shortest paths, cardinality and partition matroid constraints, there exist LP relaxation based algorithms which achieve approximation factors α being $O(\log n)$, $O(\sqrt{n} \log l / \log \log l)$, $O(\log l / \log \log l)$ and $O(\log l / \log \log l)$ respectively. Substitute these values of α and using the fact that $\beta = O(\sqrt{|V|} \log |V|)$, we get the approximation bound. $\square$

3.6 Continuous Relaxation Algorithm

Here, we use the continuous relaxation of a submodular function. In particular, we use the relaxation $\max_i \hat{f}_i(x), x \in [0,1]^{|\mathcal{V}|}$ as the continuous relaxation of the original function $\max_i f_i(X)$ (here $\hat{f}$ is the Lovász extension). Its easy to see that this is a continuous relaxation. Since the Lovász extension is convex, the function $\max_i \hat{f}_i(x)$ is also a convex function. This means that we can exactly optimize the continuous relaxation over a convex polytope. The remaining question is about the rounding and the resulting approximation guarantee due to the rounding. Given a constraint $\mathcal{C}$, define the up-monotone polytope similar to [28] $\hat{\mathcal{P}}_{\mathcal{C}} = \mathcal{P}_{\mathcal{C}} + [0,1]^{|\mathcal{V}|}$. We then use the observation from [28] that all the constraints considered in Table 1 can be expressed as:

$$\hat{\mathcal{P}}_{\mathcal{C}} = \{x \in [0,1]^n \mid \sum_{i \in W} x_i \geq b_W, \text{ for all } W \in \mathcal{W}\}. \quad (7)$$

We then round the solution using the following rounding scheme. Given a continuous vector $\hat{x}$ (which is the optimizer of $\max_i \hat{f}_i(x), x \in \hat{\mathcal{P}}_{\mathcal{C}}$, order the elements based on $\sigma_{\hat{x}}$. Denote $X_i = [\sigma_{\hat{x}}[1], \cdots, \sigma_{\hat{x}}[i]]$ so we obtain a chain of sets $\emptyset \subseteq X_1 \subseteq X_2 \subseteq \cdots \subset X_n$. Our rounding scheme picks the smallest k such that $X_k \in \hat{\mathcal{C}}$. Another way of checking this is if there exists a set $X \subset X_k$ such that $X \in \mathcal{C}$. Since $\hat{\mathcal{C}}$ is up-monotone, such a set must exist. The following result shows the approximation guarantee.

Theorem 8 *Given submodular functions f_i and constraints $\mathcal{C}$ which can be expressed as $\{x \in [0,1]^n \mid \sum_{i \in W} x_i \geq b_W$ for all $W \in \mathcal{W}\}$ for a family of sets $\mathcal{W} = \{W_1, \cdots\}$, the continuous relaxation scheme achieves an approximation guarantee of $\max_{W \in \mathcal{W}} |W| -$*

$b_W + 1$. *If we assume the sets in $\mathcal{W}$ are disjoint, the integrality bounds matches the approximation bounds.*

Proof: The proof of this theorem is closely in line with Lemma 2 and Theorem 1 from [28]. We first show the following result. Given monotone submodular functions $f_i, i \in 1, \cdots, l$, and an optimizer $\hat{x}$ of $\max_i \hat{f}(x)$, define $\hat{X}_\theta = \{i : \hat{x}_i \geq \theta\}$. We choose θ such that $\hat{X}_\theta \in \mathcal{C}$. Then $\max_i f_i(\hat{X}_\theta) \leq 1/\theta \max_i f_i(X^*)$ where X^* is the optimizer of $\min_{X \in \mathcal{C}} \max_i f_i(X)$. To prove this, observe that, by definition $\theta 1_{\hat{X}_\theta} \leq \hat{x}^3$. As a result, $\forall i, \hat{f}_i(\theta 1_{\hat{X}_\theta}) = \theta f_i(\hat{X}_\theta) \leq \hat{f}_i(\hat{x})$ (this follows because of the positive homogeneity of the Lovász extension. This implies that $\theta \max_i f_i(\hat{X}_\theta) \leq \hat{f}_i(\hat{x}) \leq \min_{x \in \mathcal{P}_{\mathcal{C}}} \hat{f}_i(\hat{x}) \leq \min_{X \in \mathcal{C}} \max_i f_i(X)$. The last inequality holds from the fact that the discrete solution is greater than the continuous one since the continuous one is a relaxation. This proves this part of the theorem.

Next, we show that the approximation guarantee holds for the class of constraints defined as $\{x \in [0,1]^n \mid \sum_{i \in W} x_i \geq b_W\}$.

From these constraints, note that for every $W \in \mathcal{W}$, at least $b_W \leq |W|$ elements need to "covered". Consequently, to round a vector $x \in \hat{\mathcal{P}}_{\mathcal{C}}$, it is sufficient to choose $\theta = \min_{W \in \mathcal{W}} x_{[b_W, W]}$ as the rounding threshold, where $x_{[k,A]}$ denotes the kth largest entry of x in a set A. The worst case scenario is that the $b_W - 1$ entries of x with indices in the set W are all 1, and the remaining mass of 1 is equally distributed over the remaining elements in W. In this case, the value of $x_{[b_W, W]}$ is $1/(|W| - b_W + 1)$. Since the constraint requires $\sum_{i \in W} x_i \geq b_W$, it must hold that $x_{[b_W, W]} \geq 1/(|W| - b_W + 1)$. Combining this with the first part of the result proves this theorem. $\square$

We can then obtain the approximation guarantees for different constraints including cardinality, spanning trees, matroids, set covers, edge covers and vertex covers, matchings, cuts and paths by appropriately defining the polytopes $\mathcal{P}_{\mathcal{C}}$ and appropriately setting the values of $\mathcal{W}$ and $\max_{W \in \mathcal{W}} |W| - b_W + 1$. Combining this with the appropriate definitions (shown in Table 3), we get the approximation bounds in Table 1.

3.7 Analysis of the Bounds for Various Constraints

Given the bounds in Tables 1 and 2, we discuss the tightness of these bounds viz-a-via the hardness. In the case when l is a constant, MMin achieves tight bounds for Trees, Matchings and Paths while the EA achieves tight bounds up to log factors for knapsack constraints. In the case when l is not a constant, MMin achieves a tight bound up to log factors for spanning tree constraints. The continuous relaxation scheme obtains tight bounds in the case of vertex covers. In the case when the functions f_i have curvature $\kappa = 1$ CR also obtains tight bounds for edge-covers and matchings. We also point out that the bounds of average approximation (AA) depend on the average case curvature as opposed to the worst case curvature. However, in prac-

---
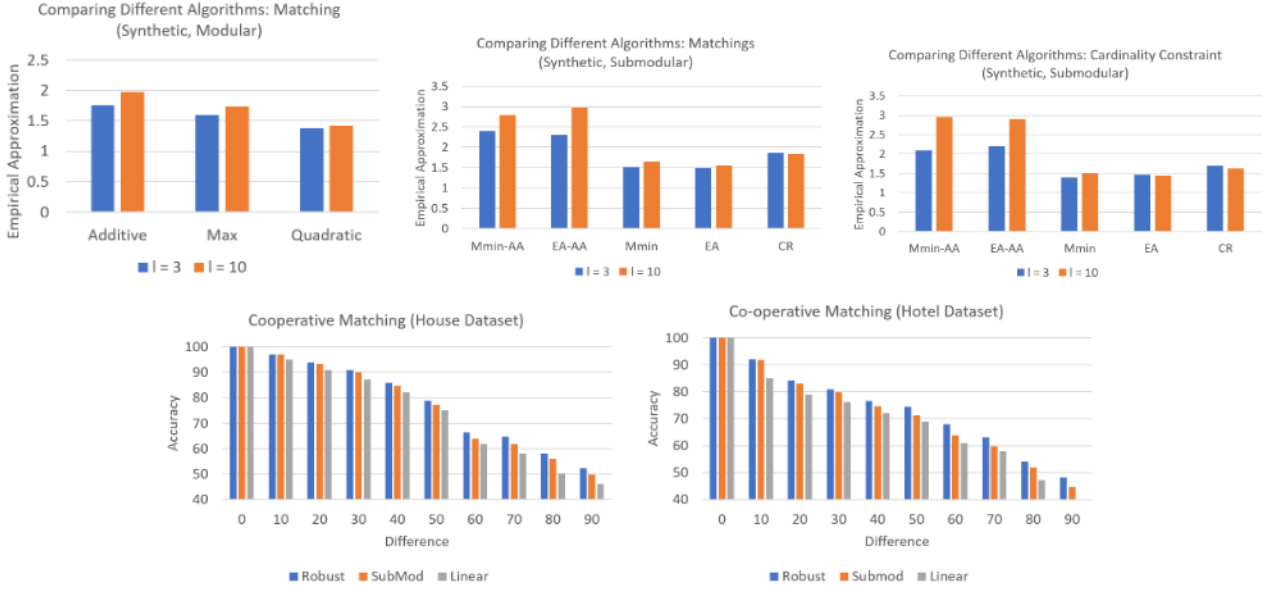[3] 1_A is the indicator vector of set A such that $1_A[i] = 1$ iff $i \in A$.

Figure 2. Top Row: Synthetic experiments with left: matching and modular, center: matching and submodular and right: cardinality and submodular. Bottom Row: Co-operative Matching with left: House and right: Hotel datasets.

tice, the functions f_i often belong to the same class of functions in which case all the functions f_i have the same curvature.

4 Experimental Results

4.1 Synthetic Experiments

The first set of experiments are synthetic experiments. We define $f_j(X) = \sum_{i=1}^{|C_j|} \sqrt{w(X \cap C_{ij})}$ for a clustering $C_j = \{C_1, C_2, \cdots, C_{|C_j|}\}$. We define l different random clusterings (with $l = 3$ and $l = 10$). We choose the vector w at random with $w \in [0, 1]^n$. We compare the different algorithms under cardinality constraints ($|X| \leq 10$) and matching constraints. For cardinality constraints, we set $n = 50$. For matchings, we define a fully connected bi-partite graph with $m = 7$ nodes on each side and correspondingly $n = 49$. The results are over 20 runs of random choices in w and C's and shown in Figure 1 (top row). The first plot compares the different algorithms with a modular function (the basic min-max combinatorial problem). The second and third plot on the top row compare the different algorithms with a submodular objective function defined above. In the submodular setting, we compare the average approximation baselines (MMin-AA, EA-AA), Majorization-Minimization (MMin), Ellipsoidal Approximation (EA) and Continuous Relaxation (CR). For the Modular cases, we compare the simple additive approximation of the worst case function, the Max-approximation and the quadratic approximation. We use the graduated assignment algorithm [17] for the quadratic approximation to solve the quadratic assignment problem. For other constraints, we can use the efficient algorithms from [6]

Figure 2 (left) shows the results in the modular setting. first that as expected that int he modular setting, the simple additive approximation of the max function doesn't perform well and the quadratic approximation approach performs the best. Figure 2 center and right show the results with the submodular function under matching and cardinality constraint. Since the quadratic approximation performs the best, we use this in the MMin and EA algorithms. First we see that the average approximations (MMin-AA and EA-AA) don't perform well since it optimizes the average case instead of the worst case. Directly optimizing the worst case performs much better. Next, we

observe that MMin performs comparably to EA though its a simpler algorithm (a fact which has been noticed in several other scenarios as well [24, 21, 30]

4.2 Co-operative Matchings

In this set of experiments, we compare ROBUST-SUBMIN in co-operative matchings. We follow the experimental setup in [27]. We run this on the House and Hotel Datasets [7]. The results in Figure 2 (bottom row). The baselines are a simple modular (additive) baseline where the image correspondence problem becomes an assignment problem and the co-operative matching model from [27] which uses a single clustering. For the robust model, we use the class of functions [27], except with multiple clusterings instead of one. Note that these clusterings are over the pixels in the two images, which then induce a clustering on the set of edges. In particular, we construct l clusterings of the pixels: $\{(\mathcal{E}_1^1, \cdots, \mathcal{E}_k^1), \cdots, (\mathcal{E}_1^l, \cdots, \mathcal{E}_k^l)\}$ and define a robust objective: $f_{robust}(S) = \max_{i=1:l} \sum_{j=1}^{k} \psi_j(w(S \cap \mathcal{E}_j^i)), S \subseteq \mathcal{E}$. We run our experiments with $l = 10$ different clustering each obtained by different random initializations of k-means. From the results, we see that the robust technique consistently outperforms a single submodular function. In this experiment, we consider all pairs of images with the difference in image numbers from 1 to 90 (The x-axis in Figure 2 (bottom row) – this is similar to the setting in [27]).

5 Conclusions

In this paper, we study Robust Submodular Minimization from a theoretical and empirical perspective. We study approximation algorithms and hardness results. We propose a scalable family of algorithms (including the majorization-minimization algorithm) and theoretically and empirically contrast their performance. In future work, we would like to address the gap between the hardness and approximation bounds, and achieve tight curvature-based bounds in each case. We would also like to study other settings and formulations of robust optimization in future work.

REFERENCES

[1] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten, 'Min–max and min–max regret versions of combinatorial optimization problems: A survey', *European journal of operational research*, **197**(2), 427–438, (2009).

[2] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten, 'General approximation schemes for min–max (regret) versions of some (pseudo-) polynomial problems', *Discrete Optimization*, **7**(3), 136–148, (2010).

[3] Nima Anari, Nika Haghtalab, Sebastian Pokutta, Mohit Singh, Alfredo Torrico, et al., 'Robust submodular maximization: Offline and online algorithms', *In Proc. AISTATS*, (2019).

[4] Steven J Benson, Yinyu Yeb, and Xiong Zhang, 'Mixed linear and semidefinite programming for combinatorial and quadratic optimization', *Optimization Methods and Software*, **11**(1-4), 515–544, (1999).

[5] Y. Boykov and V. Kolmogorov, 'An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision', *TPAMI*, **26**(9), 1124–1137, (2004).

[6] Christoph Buchheim and Emiliano Traversi, 'Quadratic combinatorial optimization using separable underestimators', *INFORMS Journal on Computing*, **30**(3), 424–437, (2018).

[7] Tibério S Caetano, Julian John McAuley, Li Cheng, Quoc V Le, and Alexander J Smola, 'Learning graph matching', *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **31**(6), 1048–1058, (2009).

[8] C. Chekuri, J. Vondrák, and R. Zenklusen, 'Submodular function maximization via the multilinear relaxation and contention resolution schemes', *STOC*, (2011).

[9] Robert S Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis, 'Robust optimization for non-convex objectives', in *Advances in Neural Information Processing Systems*, pp. 4705–4714, (2017).

[10] M. Conforti and G. Cornuejols, 'Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the Rado-Edmonds theorem', *Discrete Applied Mathematics*, **7**(3), 251–274, (1984).

[11] J. Edmonds, 'Submodular functions, matroids and certain polyhedra', *Combinatorial structures and their Applications*, (1970).

[12] M.L. Fisher, G.L. Nemhauser, and L.A. Wolsey, 'An analysis of approximations for maximizing submodular set functions—ii', *Polyhedral combinatorics*, 73–87, (1978).

[13] S. Fujishige, *Submodular functions and optimization*, volume 58, Elsevier Science, 2005.

[14] G. Goel, C. Karande, P. Tripathi, and L. Wang, 'Approximability of combinatorial problems with multi-agent submodular cost functions', in *FOCS*, (2009).

[15] G. Goel, P. Tripathi, and L. Wang, 'Combinatorial problems with discounted price functions in multi-agent systems', in *FSTTCS*, (2010).

[16] M.X. Goemans, N.J.A. Harvey, S. Iwata, and V. Mirrokni, 'Approximating submodular functions everywhere', in *SODA*, pp. 535–544, (2009).

[17] Steven Gold and Anand Rangarajan, 'A graduated assignment algorithm for graph matching', *IEEE Transactions on pattern analysis and machine intelligence*, **18**(4), 377–388, (1996).

[18] Michael Gygli, Helmut Grabner, and Luc Van Gool, 'Video summarization by learning submodular mixtures of objectives', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3090–3098, (2015).

[19] S. Iwata and K. Nagano, 'Submodular function minimization under covering constraints', in *In FOCS*, pp. 671–680. IEEE, (2009).

[20] R. Iyer and J. Bilmes, 'The submodular Bregman and Lovász-Bregman divergences with applications', in *NIPS*, (2012).

[21] R. Iyer and J. Bilmes, 'Submodular Optimization with Submodular Cover and Submodular Knapsack Constraints', in *NIPS*, (2013).

[22] R. Iyer, S. Jegelka, and J. Bilmes, 'Mirror descent like algorithms for submodular optimization', *NIPS Workshop on Discrete Optimization in Machine Learning (DISCML)*, (2012).

[23] R. Iyer, S. Jegelka, and J. Bilmes, 'Curvature and Optimal Algorithms for Learning and Minimizing Submodular Functions ', in *Neural Information Processing Society (NIPS)*, (2013).

[24] R. Iyer, S. Jegelka, and J. Bilmes, 'Fast Semidifferential based Submodular function optimization', in *ICML*, (2013).

[25] R. Iyer, S. Jegelka, and J. Bilmes, 'Fast Algorithms for Submodular Optimization based on Continuous Relaxations and Rounding', in *UAI*, (2014).

[26] Rishabh Iyer and Jeff Bilmes, 'Polyhedral aspects of submodularity, convexity and concavity', *arXiv preprint arXiv:1506.07329*, (2015).

[27] Rishabh Iyer and Jeff Bilmes, 'Near optimal algorithms for hard submodular programs with discounted cooperative costs', *In Proc. AISTATS*, (2019).

[28] Rishabh Iyer, Stefanie Jegelka, and Jeff Bilmes, 'Monotone closure of relaxed constraints in submodular optimization: Connections between minimization and maximization: Extended version', in *UAI*, (2014).

[29] Rishabh Krishnan Iyer, *Submodular optimization and machine learning: Theoretical results, unifying and scalable algorithms, and applications*, Ph.D. dissertation, 2015.

[30] S. Jegelka and J. A. Bilmes, 'Approximation bounds for inference using cooperative cuts', in *ICML*, (2011).

[31] S. Jegelka and J. A. Bilmes, 'Submodularity beyond submodular energies: coupling edges in graph cuts', in *CVPR*, (2011).

[32] Adam Kasperski and Paweł Zieliński, 'On the approximability of min-max (regret) network optimization problems', *Information Processing Letters*, **109**(5), 262–266, (2009).

[33] Adam Kasperski and Paweł Zieliński, 'On the approximability of robust spanning tree problems', *Theoretical Computer Science*, **412**(4-5), 365–374, (2011).

[34] Adam Kasperski and Paweł Zieliński, 'Robust discrete optimization under discrete and interval uncertainty: A survey', in *Robustness analysis in decision aiding, optimization, and analytics*, 113–143, Springer, (2016).

[35] Adam Kasperski and Pawel Zielinski, 'Approximating some network problems with scenarios', *arXiv preprint arXiv:1806.08936*, (2018).

[36] Andreas Krause, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos, 'Efficient sensor placement optimization for securing large water distribution networks', *Journal of Water Resources Planning and Management*, **134**(6), 516–526, (2008).

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

[38] A. Kulik, H. Shachnai, and T. Tamir, 'Maximizing submodular set functions subject to multiple linear constraints', in *SODA*, (2009).

[39] Eugene L Lawler, 'The quadratic assignment problem', *Management science*, **9**(4), 586–599, (1963).

[40] H. Lin and J. Bilmes, 'Learning mixtures of submodular shells with application to document summarization', in *UAI*, (2012).

[41] Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido, 'A survey for the quadratic assignment problem', *European journal of operational research*, **176**(2), 657–690, (2007).

[42] L. Lovász, 'Submodular functions and convexity', *Mathematical Programming*, (1983).

[43] Shashi Mittal and Andreas S Schulz, 'A general framework for designing approximation schemes for combinatorial optimization problems with many objectives combined into one', *Operations Research*, **61**(2), 386–397, (2013).

[44] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher, 'An analysis of approximations for maximizing submodular set functions—i', *Mathematical Programming*, **14**(1), 265–294, (1978).

[45] Christos H Papadimitriou and Mihalis Yannakakis, 'On the approximability of trade-offs and optimal access of web sources', in *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pp. 86–92. IEEE, (2000).

[46] Thomas Powers, David W Krout, Jeff Bilmes, and Les Atlas, 'Constrained robust submodular sensor selection with application to multistatic sonar arrays', *IET Radar, Sonar & Navigation*, **11**(12), 1776–1781, (2017).

[47] Z. Svitkina and L. Fleischer, 'Submodular approximation: Sampling-based algorithms and lower bounds', in *FOCS*, pp. 697–706, (2008).

[48] S. Tschiatschek, R. Iyer, H. Wei, and J. Bilmes, 'Learning mixtures of submodular functions for image collection summarization', in *Neural Information Processing Society (NIPS)*, Montreal, CA, (December 2014).

[49] J. Vondrák, 'Submodularity and curvature: the optimal algorithm', *RIMS Kokyuroku Bessatsu*, **23**, (2010).

[50] Bryan Wilder, 'Equilibrium computation for zero sum games with submodular structure', *arXiv preprint arXiv:1710.00996*, (2017).

[51] Laurence A. Wolsey, 'An analysis of the greedy algorithm for the submodular set covering problem', *Combinatorica*, **2**(4), 385–393, (1982).