Robust Submodular Maximization: A Non-Uniform Partitioning Approach

Ilija Bogunovic [1] **Slobodan Mitrović** [2] **Jonathan Scarlett** [1] **Volkan Cevher** [1]

Abstract

We study the problem of maximizing a monotone submodular function subject to a cardinality constraint k, with the added twist that a number of items τ from the returned set may be removed. We focus on the worst-case setting considered in (Orlin et al., 2016), in which a constant-factor approximation guarantee was given for $\tau = o(\sqrt{k})$. In this paper, we solve a key open problem raised therein, presenting a new Partitioned Robust (PRO) submodular maximization algorithm that achieves the same guarantee for more general $\tau = o(k)$. Our algorithm constructs partitions consisting of buckets with exponentially increasing sizes, and applies standard submodular optimization subroutines on the buckets in order to construct the robust solution. We numerically demonstrate the performance of PRO in data summarization and influence maximization, demonstrating gains over both the greedy algorithm and the algorithm of (Orlin et al., 2016).

1. Introduction

Discrete optimization problems arise frequently in machine learning, and are often NP-hard even to approximate. In the case of a set function exhibiting *submodularity*, one can efficiently perform maximization subject to cardinality constraints with a $\left(1-\frac{1}{e}\right)$-factor approximation guarantee. Applications include influence maximization (Kempe et al., 2003), document summarization (Lin & Bilmes, 2011), sensor placement (Krause & Guestrin, 2007), and active learning (Krause & Golovin, 2012), just to name a few.

[1]LIONS, EPFL, Switzerland [2]LTHC, EPFL, Switzerland. Correspondence to: Ilija Bogunovic <ilija.bogunovic@epfl.ch>, Slobodan Mitrović <slobodan.mitrovic@epfl.ch>, Jonathan Scarlett <jonathan.scarlett@epfl.ch>, Volkan Cevher <volkan.cevher@epfl.ch>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

In many applications of interest, one requires *robustness* in the solution set returned by the algorithm, in the sense that the objective value degrades as little as possible when some elements of the set are removed. For instance, (i) in influence maximization problems, a subset of the chosen users may decide not to spread the word about a product; (ii) in summarization problems, a user may choose to remove some items from the summary due to their personal preferences; (iii) in the problem of sensor placement for outbreak detection, some of the sensors might fail.

In situations where one does not have a reasonable prior distribution on the elements removed, or where one requires robustness guarantees with a high level of certainty, protecting against worst-case removals becomes important. This setting results in the *robust submodular function maximization* problem, in which we seek to return a set of cardinality k that is robust with respect to the worst-case removal of τ elements.

The robust problem formulation was first introduced in (Krause et al., 2008), and was further studied in (Orlin et al., 2016). In fact, (Krause et al., 2008) considers a more general formulation where a constant-factor approximation guarantee is impossible in general, but shows that one can match the optimal (robust) objective value for a given set size at the cost of returning a set whose size is larger by a logarithmic factor. In contrast, (Orlin et al., 2016) designs an algorithm that obtains the first constant-factor approximation guarantee to the above problem when $\tau = o(\sqrt{k})$. A key difference between the two frameworks is that the algorithm complexity is exponential in τ in (Krause et al., 2008), whereas the algorithm of (Orlin et al., 2016) runs in polynomial time.

Contributions. In this paper, we solve a key open problem posed in (Orlin et al., 2016), namely, whether a constant-factor approximation guarantee is possible for general $\tau = o(k)$, as opposed to only $\tau = o(\sqrt{k})$. We answer this question in the affirmative, providing a new Partitioned Robust (PRO) submodular maximization algorithm that attains a constant-factor approximation guarantee; see Table 1 for comparison of different algorithms for robust monotone submodular optimization with a cardinality constraint.

Algorithm	Max. Robustness	Cardinality	Oracle Evals.	Approx.
SATURATE (KRAUSE ET AL., 2008)	Arbitrary	$k(1+\Theta(\log(\tau k \log n)))$	exponential in τ	1.0
OSU (ORLIN ET AL., 2016)	$o(\sqrt{k})$	k	$\mathcal{O}(nk)$	0.387
PRO-GREEDY (OURS)	$o(k)$	k	$\mathcal{O}(nk)$	0.387

Table 1. Algorithms for robust monotone submodular optimization with a cardinality constraint. The proposed algorithm is efficient and allows for greater robustness.

Achieving this result requires novelty both in the algorithm and its mathematical analysis: While our algorithm bears some similarity to that of (Orlin et al., 2016), it uses a novel structure in which the constructed set is arranged into partitions consisting of buckets whose sizes increase exponentially with the partition index. A key step in our analysis provides a recursive relationship between the objective values attained by buckets appearing in adjacent partitions.

In addition to the above contributions, we provide the first empirical study beyond what is demonstrated for $\tau = 1$ in (Krause et al., 2008). We demonstrate several scenarios in which our algorithm outperforms both the greedy algorithm and the algorithm of (Orlin et al., 2016).

S	$f(S)$	$\min_{s\in S} f(S \setminus s)$
$\emptyset$	0	0
$\{s_1\}$	n	0
$\{s_2\}$	ϵ	0
$\{s_3\}$	$n-1$	0
$\{s_1, s_2\}$	$n+\epsilon$	ϵ
$\{s_1, s_3\}$	n	$\mathbf{n-1}$
$\{s_2, s_3\}$	n	ϵ

Table 2. Function f used to demonstrate that GREEDY can perform arbitrarily badly.

2. Problem Statement

Let V be a ground set with cardinality $|V| = n$, and let $f : 2^V \to \mathbb{R}_{\geq 0}$ be a set function defined on V. The function f is said to be *submodular* if for any sets $X \subseteq Y \subseteq V$ and any element $e \in V \setminus Y$, it holds that

$$f(X \cup \{e\}) - f(X) \geq f(Y \cup \{e\}) - f(Y).$$

We use the following notation to denote the marginal gain in the function value due to adding the elements of a set Y to the set X:

$$f(Y|X) := f(X \cup Y) - f(X).$$

In the case that Y is a singleton of the form $\{e\}$, we adopt the shorthand $f(e|X)$. We say that f is *monotone* if for any sets $X \subseteq Y \subseteq V$ we have $f(X) \leq f(Y)$, and *normalized* if $f(\emptyset) = 0$.

The problem of maximizing a normalized monotone submodular function subject to a cardinality constraint, i.e.,

$$\max_{S\subseteq V, |S|\leq k} f(S), \tag{1}$$

has been studied extensively. A celebrated result of (Nemhauser et al., 1978) shows that a simple greedy algorithm that starts with an empty set and then iteratively adds elements with highest marginal gain provides a $(1-1/e)$-approximation.

In this paper, we consider the following *robust* version of (1), introduced in (Krause et al., 2008):

$$\max_{S\subseteq V, |S|\leq k} \min_{Z\subseteq S, |Z|\leq \tau} f(S \setminus Z) \tag{2}$$

We refer to τ as the robustness parameter, representing the size of the subset Z that is removed from the selected set S. Our goal is to find a set S such that it is robust upon the worst possible removal of τ elements, i.e., after the removal, the objective value should remain as large as possible. For $\tau = 0$, our problem reduces to Problem (1).

The greedy algorithm, which is near-optimal for Problem (1) can perform arbitrarily badly for Problem (2). As an elementary example, let us fix $\epsilon \in [0, n-1)$ and $n \geq 0$, and consider the non-negative monotone submodular function given in Table 2. For $k = 2$, the greedy algorithm selects $\{s_1, s_2\}$. The set that maximizes $\min_{s\in S} f(S\setminus s)$ (i.e., $\tau = 1$) is $\{s_1, s_3\}$. For this set, $\min_{s\in\{s_1,s_2\}} f(\{s_1, s_2\} \setminus s) = n-1$, while for the greedy set the robust objective value is ϵ. As a result, the greedy algorithm can perform arbitrarily worse.

In our experiments on real-world data sets (see Section 5), we further explore the empirical behavior of the greedy solution in the robust setting. Among other things, we observe that the greedy solution tends to be less robust when the objective value largely depends on the first few elements selected by the greedy rule.

Related work. (Krause et al., 2008) introduces the following generalization of (2):

$$\max_{S\subseteq V,|S|\leq k} \min_{i\in\{1,\cdots,n\}} f_i(S), \tag{3}$$

where f_i are normalized monotone submodular functions. The authors show that this problem is inapproximable in general, but propose an algorithm SATURATE which, when applied to (2), returns a set of size $k(1+\Theta(\log(\tau k \log n)))$ whose robust objective is at least as good as the optimal size-k set. SATURATE requires a number of function evaluations that is exponential in τ, making it very expensive to run even for small values. The work of (Powers et al., 2016) considers the same problem for different types of submodular constraints.

Recently, robust versions of submodular maximization have been applied to influence maximization. In (He & Kempe, 2016), the formulation (3) is used to optimize a worst-case approximation ratio. The confidence interval setting is considered in (Chen et al., 2016), where two runs of the GREEDY algorithm (one pessimistic and one optimistic) are used to optimize the same ratio. By leveraging connections to continuous submodular optimization, (Staib & Jegelka, 2017) studies a related continuous robust budget allocation problem.

(Orlin et al., 2016) considers the formulation in (2), and provides the first constant 0.387-factor approximation result, valid for $\tau = o(\sqrt{k})$. The algorithm proposed therein, which we refer to via the authors surnames as OSU, uses the greedy algorithm (henceforth referred to as GREEDY) as a sub-routine $\tau+1$ times. On each iteration, GREEDY is applied on the elements that are not yet selected on previous iterations, with these previously-selected elements ignored in the objective function. In the first τ runs, each solution is of size $\tau \log k$, while in the last run, the solution is of size $k-\tau^2 \log k$. The union of all the obtained disjoint solutions leads to the final solution set.

3. Applications

In this section, we provide several examples of applications where the robustness of the solution is favorable. The objective functions in these applications are non-negative, monotone and submodular, and are used in our numerical experiments in Section 5.

Robust influence maximization. The goal in the influence maximization problem is to find a set of k nodes (i.e., a targeted set) in a network that maximizes some measure of influence. For example, this problem appears in viral marketing, where companies wish to spread the word of a new product by targeting the most influential individuals in a social network. Due to poor incentives or dissatisfaction with the product, for instance, some of the users from the targeted set might make the decision not to spread the word about the product.

For many of the existing diffusion models used in the literature (e.g., see (Kempe et al., 2003)), given the targeted set S, the expected number of influenced nodes at the end of the diffusion process is a monotone and submodular function of S (He & Kempe, 2016). For simplicity, we consider a basic model in which all of the neighbors of the users in S become influenced, as well as those in S itself.

More formally, we are given a graph $G = (V, E)$, where V stands for nodes and E are the edges. For a set S, let $\mathcal{N}(S)$ denote all of its neighboring nodes. The goal is to solve the robust *dominating set problem*, i.e., to find a set of nodes S of size k that maximizes

$$\min_{|R_S|\leq\tau, R_S\subseteq S} |(S\setminus R_S)\cup\mathcal{N}(S\setminus R_S)|, \tag{4}$$

where $R_S \subseteq S$ represents the users that decide not to spread the word. The non-robust version of this objective function has previously been considered in several different works, such as (Mirzasoleiman et al., 2015b) and (Norouzi-Fard et al., 2016).

Robust personalized image summarization. In the personalized image summarization problem, a user has a collection of images, and the goal is to find k images that are representative of the collection.

After being presented with a solution, the user might decide to remove a certain number of images from the representative set due to various reasons (e.g., bad lighting, motion blur, etc.). Hence, our goal is to find a set of images that remain good representatives of the collection even after the removal of some number of them.

One popular way of finding a representative set in a massive dataset is via exemplar based clustering, i.e., by minimizing the sum of pairwise dissimilarities between the exemplars S and the elements of the data set V. This problem can be posed as a submodular maximization problem subject to a cardinality constraint; *cf.*, (Lucic et al., 2016).

Here, we are interested in solving the robust summarization problem, i.e., we want to find a set of images S of size k that maximizes

$$\min_{|R_S|\leq\tau, R_S\subseteq S} f(\{e_0\}) - f((S\setminus R_S)\cup\{e_0\}), \tag{5}$$

where e_0 is a reference element and $f(S) = \frac{1}{|V|}\sum_{v\in V}\min_{s\in S} d(s,v)$ is the *k-medoid* loss function, and where $d(s,v)$ measures the dissimilarity between images s and v.

Further potential applications not covered here include robust sensor placement (Krause et al., 2008), robust protection of networks (Bogunovic & Krause, 2012), and robust feature selection (Globerson & Roweis, 2006).

4. Algorithm and its Guarantees

4.1. The algorithm

Our algorithm, which we call the Partitioned Robust (PRO) submodular maximization algorithm, is presented in Algorithm 1. As the input, we require a non-negative monotone submodular function $f : 2^V \to \mathbb{R}_{\geq 0}$, the ground set of elements V, and an optimization subroutine $\mathcal{A}$. The subroutine $\mathcal{A}(k', V')$ takes a cardinality constraint k' and a ground set of elements V'. Below, we describe the properties of $\mathcal{A}$ that are used to obtain approximation guarantees.

The output of the algorithm is a set $S \subseteq V$ of size k that is robust against the worst-case removal of τ elements. The returned set consists of two sets S_0 and S_1, illustrated in Figure 1. S_1 is obtained by running the subroutine $\mathcal{A}$ on $V \setminus S_0$ (i.e., ignoring the elements already placed into S_0), and is of size $k - |S_0|$.

We refer to the set S_0 as the robust part of the solution set S. It consists of $\lceil \log \tau \rceil + 1$ partitions, where every partition $i \in \{0, \cdots, \lceil \log \tau \rceil\}$ consists of $\lceil \tau / 2^i \rceil$ buckets B_j, $j \in \{1, \cdots, \lceil \tau / 2^i \rceil\}$. In partition i, every bucket contains $2^i \eta$ elements, where $\eta \in \mathbb{N}_+$ is a parameter that is arbitrary for now; we use $\eta = \log^2 k$ in our asymptotic theory, but our numerical studies indicate that even $\eta = 1$ works well in practice. Each bucket B_j is created afresh by using the subroutine $\mathcal{A}$ on $V \setminus S_{0,\text{prev}}$, where $S_{0,\text{prev}}$ contains all elements belonging to the previous buckets.

The following proposition bounds the cardinality of S_0, and is proved in the supplementary material.

Proposition 4.1 *Fix $k \geq \tau$ and $\eta \in \mathbb{N}_+$. The size of the robust part S_0 constructed in Algorithm 1 is*

$$|S_0| \;=\; \sum_{i=0}^{\lceil \log \tau \rceil} \lceil \tau / 2^i \rceil 2^i \eta \;\leq\; 3\eta\tau(\log k + 2).$$

This proposition reveals that the feasible values of τ (i.e., those with $|S_0| \leq k$) can be as high as $O\big(\frac{k}{\eta \tau}\big)$. We will later set $\eta = O(\log^2 k)$, thus permitting all $\tau = o(k)$ up to a few logarithmic factors. In contrast, we recall that the algorithm OSU proposed in (Orlin et al., 2016) adopts a simpler approach where a robust set is used consisting of τ buckets of equal size $\tau \log k$, thereby only permitting the scaling $\tau = o(\sqrt{k})$.

We provide the following intuition as to why PRO succeeds despite having a smaller size for S_0 compared to the algorithm given in (Orlin et al., 2016). First, by the design of the partitions, there always exists a bucket in partition i that at most 2^i items are removed from. The bulk of our analysis is devoted to showing that the union of these buckets yields a sufficiently high objective value. While the earlier buckets have a smaller size, they also have a higher objective value per item due to diminishing returns, and our analysis quantifies and balances this trade-off. Similarly, our analysis quantifies the trade-off between how much the adversary can remove from the (typically large) set S_1 and the robust part S_0.

Algorithm 1 Partitioned Robust Submodular optimization algorithm (PRO)

Require: Set V, k, τ, $\eta \in \mathbb{N}_+$, algorithm $\mathcal{A}$
Ensure: Set $S \subseteq V$ such that $|S| \leq k$
$S_0, S_1 \leftarrow \emptyset$
for $i \leftarrow 0$ **to** $\lceil \log \tau \rceil$ **do**
  for $j \leftarrow 1$ **to** $\lceil \tau / 2^i \rceil$ **do**
    $B_j \leftarrow \mathcal{A}\left(2^i \eta,\ (V \setminus S_0)\right)$
    $S_0 \leftarrow S_0 \cup B_j$
$S_1 \leftarrow \mathcal{A}\left(k - |S_0|,\ (V \setminus S_0)\right)$
$S \leftarrow S_0 \cup S_1$
return S

4.2. Subroutine and assumptions

PRO accepts a subroutine $\mathcal{A}$ as the input. We consider a class of algorithms that satisfy the *β-iterative property*, defined below. We assume that the algorithm outputs the final set in some specific order $(v_1, \ldots, v_k)$, and we refer to v_i as the *i-th output element*.

Definition 4.2 *Consider a normalized monotone submodular set function f on a ground set V, and an algorithm $\mathcal{A}$. Given any set $T \subseteq V$ and size k, suppose that $\mathcal{A}$ outputs an ordered set $(v_1, \ldots, v_k)$ when applied to T, and define $\mathcal{A}_i(T) = \{v_1, \ldots, v_i\}$ for $i \leq k$. We say that $\mathcal{A}$ satisfies the* β-iterative property *if*

$$f(\mathcal{A}_{i+1}(T)) - f(\mathcal{A}_i(T)) \geq \frac{1}{\beta} \max_{v \in T} f(v | \mathcal{A}_i(T)). \quad (6)$$

Intuitively, (6) states that in every iteration, the algorithm adds an element whose marginal gain is at least a $1/\beta$ fraction of the maximum marginal. This necessarily requires that $\beta \geq 1$.

Examples. Besides the classic greedy algorithm, which satisfies (6) with $\beta = 1$, a good candidate for our subroutine is THRESHOLDING-GREEDY (Badanidiyuru & Vondrák, 2014), which satisfies the β-iterative property with $\beta = 1/(1-\epsilon)$. This decreases the number of function evaluations to $\mathcal{O}(n/\epsilon \log n/\epsilon)$.

STOCHASTIC-GREEDY (Mirzasoleiman et al., 2015a) is another potential subroutine candidate. While it is unclear whether this algorithm satisfies the β-iterative property, it requires an even smaller number of function eval-

uations, namely, $\mathcal{O}(n \log 1/\epsilon)$. We will see in Section 5 that PRO performs well empirically when used with this subroutine. We henceforth refer to PRO used along with its appropriate subroutine as PRO-GREEDY, PRO-THRESHOLDING-GREEDY, and so on.

Properties. The following lemma generalizes a classical property of the greedy algorithm (Nemhauser et al., 1978; Krause & Golovin, 2012) to the class of algorithms satisfying the β-iterative property. Here and throughout the paper, we use $\mathrm{OPT}(k, V)$ to denote the following optimal set for *non-robust* maximization:

$$\mathrm{OPT}(k, V) \in \operatorname*{argmax}_{S \subseteq V, |S|=k} f(S),$$

Lemma 4.3 *Consider a normalized monotone submodular function $f : 2^V \to \mathbb{R}_{\geq 0}$ and an algorithm $\mathcal{A}(T)$, $T \subseteq V$, that satisfies the β-iterative property in (6). Let $\mathcal{A}_l(T)$ denote the set returned by the algorithm $\mathcal{A}(T)$ after l iterations. Then for all $k, l \in \mathbb{N}_+$*

$$f(\mathcal{A}_l(T)) \geq \left(1 - e^{-\frac{l}{\beta k}}\right) f(\mathrm{OPT}(k, T)). \tag{7}$$

We will also make use of the following property, which is implied by the β-iterative property.

Proposition 4.4 *Consider a submodular set function $f : 2^V \to \mathbb{R}_{\geq 0}$ and an algorithm $\mathcal{A}$ that satisfies the β-iterative property for some $\beta \geq 1$. Then, for any $T \subseteq V$ and element $e \in V \setminus \mathcal{A}(T)$, we have*

$$f(e|\mathcal{A}(T)) \leq \beta \frac{f(\mathcal{A}(T))}{k}. \tag{8}$$

Intuitively, (8) states that the marginal gain of any non-selected element cannot be more than β times the average objective value of the selected elements. This is one of the rules used to define the β-nice class of algorithms in (Mirrokni & Zadimoghaddam, 2015); however, we note that in general, neither the β-nice nor β-iterative classes are a subset of one another.

4.3. Main result: Approximation guarantee

For the robust maximization problem, we let $\mathrm{OPT}(k, V, \tau)$ denote the optimal set:

$$\mathrm{OPT}(k, V, \tau) \in \operatorname*{argmax}_{S \subseteq V, |S|=k} \min_{E \subseteq S, |E| \leq \tau} f(S \setminus E).$$

Moreover, for a set S, we let E_S^* denote the minimizer

$$E_S^* \in \operatorname*{argmin}_{E \subseteq S, |E| \leq \tau} f(S \setminus E). \tag{9}$$

With these definitions, the main theoretical result of this paper is as follows.

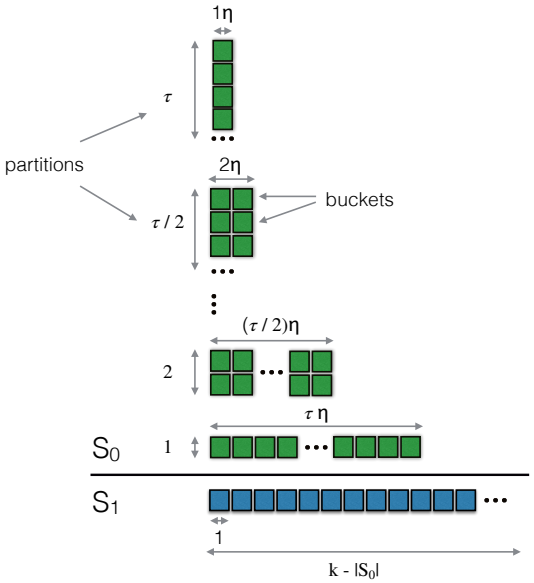

Figure 1. Illustration of the set $S = S_0 \cup S_1$ returned by PRO. The size of $|S_1|$ is $k - |S_0|$, and the size of $|S_0|$ is given in Proposition 4.1. Every partition in S_0 contains the same number of elements (up to rounding).

Theorem 4.5 *Let f be a normalized monotone submodular function, and let $\mathcal{A}$ be a subroutine satisfying the β-iterative property. For a given budget k and parameters $2 \leq \tau \leq \frac{k}{3\eta(\log k + 2)}$ and $\eta \geq 4(\log k + 1)$,* PRO *returns a set S of size k such that*

$$f(S \setminus E_S^*) \geq \frac{\frac{\eta}{5\beta^3 \lceil \log \tau \rceil + \eta} \left(1 - e^{-\frac{k - |S_0|}{\beta(k-\tau)}}\right)}{1 + \frac{\eta}{5\beta^3 \lceil \log \tau \rceil + \eta} \left(1 - e^{-\frac{k - |S_0|}{\beta(k-\tau)}}\right)} \times f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}), \tag{10}$$

where E_S^* and $E^*_{\mathrm{OPT}(k,V,\tau)}$ are defined as in (9).

In addition, if $\tau = o\left(\frac{k}{\eta \log k}\right)$ and $\eta \geq \log^2 k$, then we have the following as $k \to \infty$:

$$f(S \setminus E_S^*) \geq \left(\frac{1 - e^{-1/\beta}}{2 - e^{-1/\beta}} + o(1)\right) \times f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}). \tag{11}$$

In particular, PRO-GREEDY *achieves an asymptotic approximation factor of at least* 0.387*, and* PRO-THRESHOLDING-GREEDY *with parameter ϵ achieves an asymptotic approximation factor of at least* $0.387(1 - \epsilon)$.

This result solves an open problem raised in (Orlin et al., 2016), namely, whether a constant-factor approximation guarantee can be obtained for $\tau = o(k)$ as opposed to

only $\tau = o(\sqrt{k})$. In the asymptotic limit, our constant factor of 0.387 for the greedy subroutine matches that of (Orlin et al., 2016), but our algorithm permits significantly "higher robustness" in the sense of allowing larger τ values. To achieve this, we require novel proof techniques, which we now outline.

4.4. High-level overview of the analysis

The proof of Theorem 4.5 is provided in the supplementary material. Here we provide a high-level overview of the main challenges.

Let E denote a cardinality-τ subset of the returned set S that is removed. By the construction of the partitions, it is easy to verify that each partition i contains a bucket from which at most 2^i items are removed. We denote these by $B_0, \dots, B_{\lceil \log \tau \rceil}$, and write $E_{B_i} := E \cap B_i$. Moreover, we define $E_0 := E \cap S_0$ and $E_1 := E \cap S_1$.

We establish the following lower bound on the final objective function value:

$$f(S \setminus E) \geq \max \Bigg\{ f(S_0 \setminus E_0), f(S_1) - f(E_1 | (S \setminus E)), f\Bigg(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i}) \Bigg) \Bigg\}. \quad (12)$$

The arguments to the first and third terms are trivially seen to be subsets of $S \setminus E$, and the second term represents the utility of the set S_1 subsided by the utility of the elements removed from S_1.

The first two terms above are easily lower bounded by convenient expressions via submodular and the β-iterative property. The bulk of the proof is dedicated to bounding the third term. To do this, we establish the following recursive relations with suitably-defined "small" values of α_j:

$$f\left(\bigcup_{i=0}^{j} (B_i \setminus E_{B_i}) \right) \geq \left(1 - \frac{1}{1 + \frac{1}{\alpha_j}} \right) f(B_j)$$

$$f\left(E_{B_j} \;\middle|\; \bigcup_{i=0}^{j-1} (B_i \setminus E_{B_i}) \right) \leq \alpha_j f\left(\bigcup_{i=0}^{j-1} (B_i \setminus E_{B_i}) \right).$$

Intuitively, the first equation shows that the objective value from buckets $i = 0, \dots, j$ *with removals* cannot be too much smaller than the objective value in bucket j *without removals*, and the second equation shows that the loss in bucket j due to the removals is at most a small fraction of the objective value from buckets $0, \dots, j-1$. The proofs of both the base case of the induction and the inductive step make use of submodularity properties and the β-iterative property (*cf.*, Definition 4.2).

Once the suitable lower bounds are obtained for the terms in (12), the analysis proceeds similarly to (Orlin et al., 2016). Specifically, we can show that as the second term increases, the third term decreases, and accordingly lower bound their maximum by the value obtained when the two are equal. A similar balancing argument is then applied to the resulting term and the first term in (12).

The condition $\tau \leq \frac{k}{3\eta(\log k + 2)}$ follows directly from Proposition 4.1; namely, it is a sufficient condition for $|S_0| \leq k$, as is required by PRO.

5. Experiments

In this section, we numerically validate the performance of PRO and the claims given in the preceding sections. In particular, we compare our algorithm against the OSU algorithm proposed in (Orlin et al., 2016) on different datasets and corresponding objective functions (see Table 3). We demonstrate matching or improved performance in a broad range of settings, as well as observing that PRO can be implemented with larger values of τ, corresponding to a greater robustness. Moreover, we show that for certain real-world data sets, the classic GREEDY algorithm can perform badly for the robust problem. We do not compare against SATURATE (Krause et al., 2008), due to its high computational cost for even a small τ.

Setup. Given a solution set S of size k, we measure the performance in terms of the minimum objective value upon the worst-case removal of τ elements, i.e. $\min_{Z \subseteq S, |Z| \leq \tau} f(S \setminus Z)$. Unfortunately, for a given solution set S, finding such a set Z is an instance of the submodular minimization problem with a cardinality constraint,[1] which is known to be NP-hard with polynomial approximation factors (Svitkina & Fleischer, 2011). Hence, in our experiments, we only implement the optimal "adversary" (i.e., removal of items) for small to moderate values of τ and k, for which we use a fast C++ implementation of branch-and-bound.

Despite the difficulty in implementing the optimal adversary, we observed in our experiments that the *greedy adversary*, which iteratively removes elements to reduce the objective value as much as possible, has a similar impact on the objective compared to the optimal adversary for the data sets considered. Hence, we also provide a larger-scale experiment in the presence of a greedy adversary. Throughout, we write OA and GA to abbreviate the optimal adversary and greedy adversary, respectively.

In our experiments, the size of the robust part of the solution set (i.e., $|S_0|$) is set to τ^2 and $\tau \log \tau$ for OSU and PRO, respectively. That is, we set $\eta = 1$ in PRO, and similarly ignore constant and logarithmic factors in OSU, since both appear to be unnecessary in practice. We show

[1] This can be seen by noting that for submodular f and any $Z \subseteq X \subseteq V$, $f'(Z) = f(X \setminus Z)$ remains submodular.

both the "raw" objective values of the solutions, as well as the objective values after the removal of τ elements. In all experiments, we implement GREEDY using the LAZY-GREEDY implementation given in (Minoux, 1978).

The objective functions shown in Table 3 are given in Section 3. For the exemplar objective function, we use $d(s, v) = \|s - v\|^2$, and let the reference element e_0 be the zero vector. Instead of using the whole set V, we approximate the objective by considering a smaller random subset of V for improved computational efficiency. Since the objective is additively decomposable and bounded, standard concentration bounds (e.g., the Chernoff bound) ensure that the empirical mean over a random subsample can be made arbitrarily accurate.

Data sets. We consider the following datasets, along with the objective functions given in Section 3:

- EGO-FACEBOOK: This network data consists of social circles (or friends lists) from Facebook forming an undirected graph with 4039 nodes and 88234 edges.
- EGO-TWITTER: This dataset consists of 973 social circles from Twitter, forming a directed graph with 81306 nodes and 1768149 edges. Both EGO-FACEBOOK and EGO-TWITTER were used previously in (Mcauley & Leskovec, 2014).
- TINY10K and TINY50K: We used two Tiny Images data sets of size $10k$ and $50k$ consisting of images each represented as a 3072-dimensional vector (Torralba et al., 2008). Besides the number of images, these two datasets also differ in the number of classes that the images are grouped into. We shift each vectors to have zero mean.
- CM-MOLECULES: This dataset consists of 7211 small organic molecules, each represented as a 276 dimensional vector. Each vector is obtained by processing the molecule's *Coulomb* matrix representation (Rupp, 2015). We shift and normalize each vector to zero mean and unit norm.

Dataset	n	dimension	f
Tiny-10k	10 000	3074	Exemplar
Tiny-50k	50 000	3074	Exemplar
CM-Molecules	7211	276	Exemplar
Network	**# nodes**	**# edges**	f
ego-Facebook	4039	88 234	DomSet
ego-Twitter	81 306	1 768 149	DomSet

Table 3. Datasets and corresponding objective functions.

Results. In the first set of experiments, we compare PRO-GREEDY (written using the shorthand PRO-GR in the legend) against GREEDY and OSU on the EGO-FACEBOOK and EGO-TWITTER datasets. In this experiment, the dominating set selection objective in (4) is considered. Figure 2 (a) and (c) show the results before and after the worst-case removal of $\tau = 7$ elements for different values of k. In Figure 2 (b) and (d), we show the objective value for fixed $k = 50$ and $k = 100$, respectively, while the robustness parameter τ is varied.

GREEDY achieves the highest raw objective value, followed by PRO-GREEDY and OSU. However, after the worst-case removal, PRO-GREEDY-OA outperforms both OSU-OA and GREEDY-OA. In Figure 2 (a) and (b), GREEDY-OA performs poorly due to a high concentration of the objective value on the first few elements selected by GREEDY. While OSU requires $k \geq \tau^2$, PRO only requires $k \geq \tau \log \tau$, and hence it can be run for larger values of τ (e.g., see Figure 2 (b) and (c)). Moreover, in Figure 2 (a) and (b), we can observe that although PRO uses a smaller number of elements to build the robust part of the solution set, it has better robustness in comparison with OSU.

In the second set of experiments, we perform the same type of comparisons on the TINY10 and CM-MOLECULES datasets. The exemplar based clustering in (5) is used as the objective function. In Figure 2 (e) and (h), the robustness parameter is fixed to $\tau = 7$ and $\tau = 6$, respectively, while the cardinality k is varied. In Figure 2 (f) and (h), the cardinality is fixed to $k = 100$ and $k = 50$, respectively, while the robustness parameter τ is varied.

Again, GREEDY achieves the highest objective value. On the TINY10 dataset, GREEDY-OA (Figure 2 (e) and (f)) has a large gap between the raw and final objective, but it still slightly outperforms PRO-GREEDY-OA. This demonstrates that GREEDY can work well in some cases, despite failing in others. We observed that it succeeds here because the objective value is relatively more uniformly spread across the selected elements. On the same dataset, PRO-GREEDY-OA outperforms OSU-OA. On our second dataset CM-MOLECULES (Figure 2 (g) and (h)), PRO-GREEDY-OA achieves the highest robust objective value, followed by OSU-OA and GREEDY-OA.

In our final experiment (see Figure 2 (i)), we compare the performance of PRO-GREEDY against two instances of PRO-STOCHASTIC-GREEDY with $\epsilon = 0.01$ and $\epsilon = 0.08$ (shortened to PRO-ST in the legend), seeking to understand to what extent using the more efficient stochastic subroutine impacts the performance. We also show the performance of OSU. In this experiment, we fix $k = 100$ and vary τ. We use the greedy adversary instead of the optimal one, since the latter becomes computationally challenging for larger values of τ.

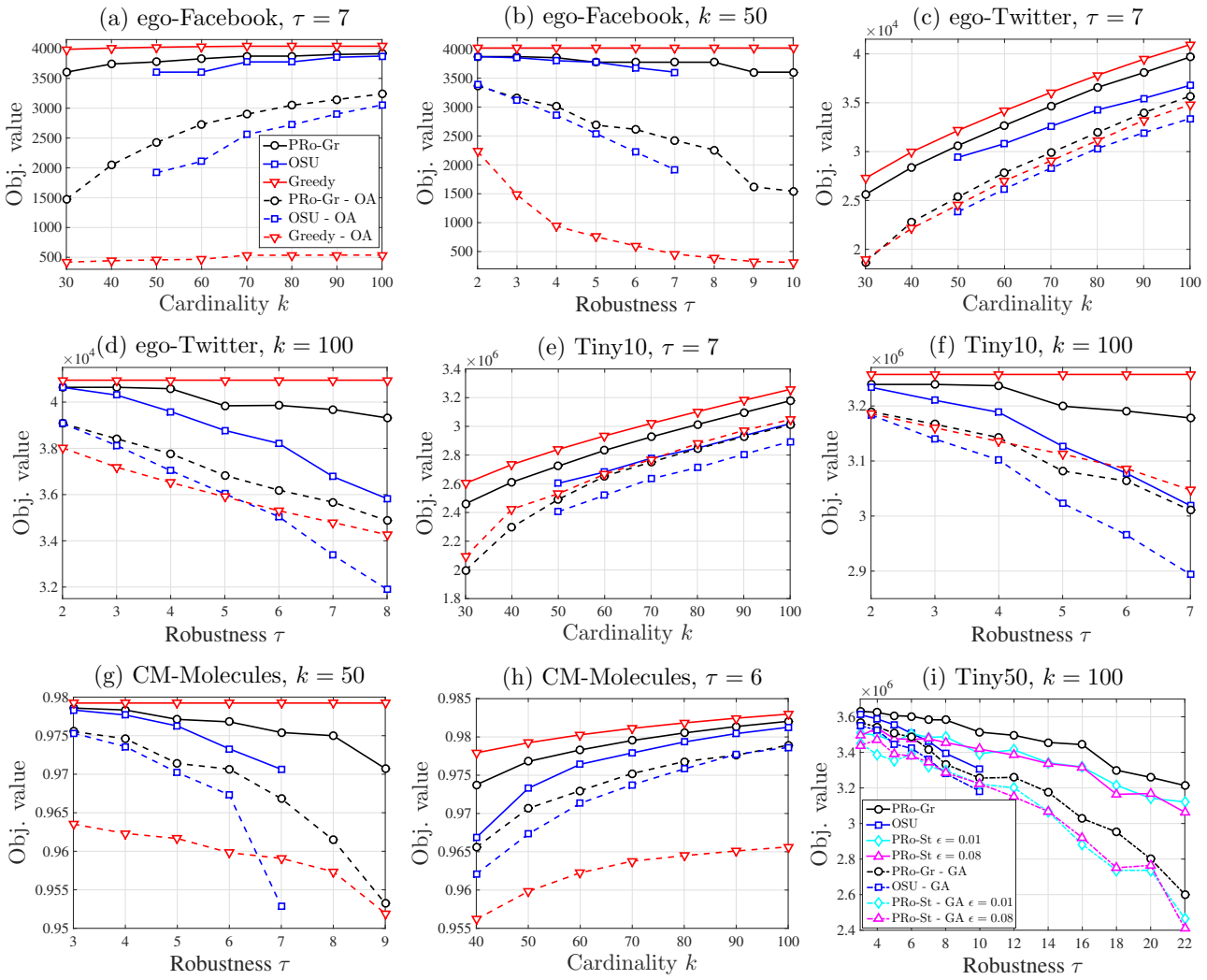

Figure 2. Numerical comparisons of the algorithms PRO-GREEDY, GREEDY and OSU, and their objective values PRO-OA, OSU-OA and GREEDY-OA once τ elements are removed. Figure (i) shows the performance on the larger scale experiment where both GREEDY and STOCHASTIC-GREEDY are used as subroutines in PRO.

In Figure 2 (i), we observe a slight decrease in the objective value of PRO-STOCHASTIC-GREEDY due to the stochastic optimization. On the other hand, the gaps between the robust and non-robust solutions remain similar, or even shrink. Overall, we observe that at least in this example, the stochastic subroutine does not compromise the quality of the solution too significantly, despite having a lower computational complexity.

6. Conclusion

We have provided a new Partitioned Robust (PRO) submodular maximization algorithm attaining a constant-factor approximation guarantee for general $\tau = o(k)$, thus resolving an open problem posed in (Orlin et al., 2016). Our algorithm uses a novel partitioning structure with partitions consisting of buckets with exponentially decreasing size, thus providing a "robust part" of size $O(\tau \text{poly} \log \tau)$. We have presented a variety of numerical experiments where PRO outperforms both GREEDY and OSU. A potentially interesting direction for further research is to understand the *linear regime*, in which $\tau = \alpha k$ for some constant $\alpha \in (0, 1)$, and in particular, to seek a constant-factor guarantee for this regime.

Acknowledgment. This work was supported in part by the European Commission under Grant ERC Future Proof, SNF 200021-146750 and SNF CRSII2-147633, and 'EPFL Fellows' (Horizon2020 665667).

References

Badanidiyuru, Ashwinkumar and Vondrák, Jan. Fast algorithms for maximizing submodular functions. In *ACM-SIAM Symp. Disc. Alg. (SODA)*, pp. 1497–1514, 2014.

Bogunovic, Ilija and Krause, Andreas. Robust protection of networks against cascading phenomena. *Tech. Report ETH Zürich*, 2012.

Chen, Wei, Lin, Tian, Tan, Zihan, Zhao, Mingfei, and Zhou, Xuren. Robust influence maximization. *arXiv preprint arXiv:1601.06551*, 2016.

Globerson, Amir and Roweis, Sam. Nightmare at test time: robust learning by feature deletion. In *Int. Conf. Mach. Learn. (ICML)*, 2006.

He, Xinran and Kempe, David. Robust influence maximization. In *Int. Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 885–894, 2016.

Kempe, David, Kleinberg, Jon, and Tardos, Éva. Maximizing the spread of influence through a social network. In *Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, 2003.

Krause, Andreas and Golovin, Daniel. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3(19):8, 2012.

Krause, Andreas and Guestrin, Carlos. Near-optimal observation selection using submodular functions. In *Conf. Art. Intell. (AAAI)*, 2007.

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

Lin, Hui and Bilmes, Jeff. A class of submodular functions for document summarization. In *Assoc. for Comp. Ling.: Human Language Technologies-Volume 1*, 2011.

Lucic, Mario, Bachem, Olivier, Zadimoghaddam, Morteza, and Krause, Andreas. Horizontally scalable submodular maximization. In *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016.

Mcauley, Julian and Leskovec, Jure. Discovering social circles in ego networks. *ACM Trans. Knowl. Discov. Data*, 2014.

Minoux, Michel. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pp. 234–243. Springer, 1978.

Mirrokni, Vahab and Zadimoghaddam, Morteza. Randomized composable core-sets for distributed submodular maximization. In *ACM Symposium on Theory of Computing (STOC)*, 2015.

Mirzasoleiman, Baharan, Badanidiyuru, Ashwinkumar, Karbasi, Amin, Vondrák, Jan, and Krause, Andreas. Lazier than lazy greedy. In *Proc. Conf. Art. Intell. (AAAI)*, 2015a.

Mirzasoleiman, Baharan, Karbasi, Amin, Badanidiyuru, Ashwinkumar, and Krause, Andreas. Distributed submodular cover: Succinctly summarizing massive data. In *Adv. Neur. Inf. Proc. Sys. (NIPS)*, pp. 2881–2889, 2015b.

Nemhauser, George L, Wolsey, Laurence A, and Fisher, Marshall L. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294, 1978.

Norouzi-Fard, Ashkan, Bazzi, Abbas, Bogunovic, Ilija, El Halabi, Marwa, Hsieh, Ya-Ping, and Cevher, Volkan. An efficient streaming algorithm for the submodular cover problem. In *Adv. Neur. Inf. Proc. Sys. (NIPS)*, 2016.

Orlin, James B, Schulz, Andreas S, and Udwani, Rajan. Robust monotone submodular function maximization. In *Int. Conf. on Integer Programming and Combinatorial Opt. (IPCO)*. Springer, 2016.

Powers, Thomas, Bilmes, Jeff, Wisdom, Scott, Krout, David W, and Atlas, Les. Constrained robust submodular optimization. NIPS OPT2016 workshop, 2016.

Rupp, Matthias. Machine learning for quantum mechanics in a nutshell. *Int. Journal of Quantum Chemistry*, 115 (16):1058–1073, 2015.

Staib, Matthew and Jegelka, Stefanie. Robust budget allocation via continuous submodular functions. `http://people.csail.mit.edu/stefje/papers/robust_budget.pdf`, 2017.

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

Torralba, Antonio, Fergus, Rob, and Freeman, William T. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Patt. Ana. Mach. Intel.*, 30(11):1958–1970, 2008.