# Robust Submodular Maximization: Offline and Online Algorithms

Nima Anari* Nika Haghtalab† Joseph (Seffi) Naor‡
Sebastian Pokutta§ Mohit Singh¶ Alfredo Torrico‖

**Abstract**

Submodular function maximization has found numerous applications in constrained subset selection problems, for example picking a subset of candidate sensor locations that are most informative [22, 19, 16]. In many of these applications, the goal is to obtain a solution that optimizes multiple objectives at the same time. Constrained Robust Submodular maximization problems are used as a natural and effective model for such scenarios [15]. In this paper, we consider the robust submodular maximization problem subject to a matroid constraint in the offline as well as online setting.

In the offline version of the problem, we are given a collection of $k$ monotone submodular functions and a matroid on a ground set of size $n$. The goal is to select one independent set that maximizes the minimum of the submodular functions. This problem is known to be NP-hard to approximate to any polynomial factor. We design (nearly) optimal bi-criteria approximation algorithms that returns a set $S$ that is the union of $\ln(\frac{k}{\epsilon}) + O(1)$ independent sets such that each function evaluated on $S$ is at least $(1-\epsilon)$ fraction of the optimal value. These results improve on previous results known for uniform matroids or the general matroid case when $k$ is a constant. We also note that no bi-criteria approximation algorithms are possible for non-monotone submodular functions in contrast to the setting of a single submodular function.

In the online version of the problem, we receive a new collection of functions at each time step and aim to pick an independent set in every stage. We measure the performance of the algorithm in the regret setting where the goal is to give a solution that compares well to picking a single set for all stages. Again, we give a bi-criteria approximation algorithm which gives a (nearly) optimal approximation as well as regret bounds. Our results rely crucially on modifying the *Follow the Perturbed Leader* algorithm of Kalai and Vempala [12] to incorporate the non-convexity introduced in the problem due to submodularity as well as the robustness criteria.

## 1 Introduction

Constrained submodular function maximization has seen significant theoretical progress [4, 9, 1, 27] and found numerous applications especially in constrained subset selection problems [22,

*Stanford University, Stanford. Email: anari@berkeley.edu
†Carnegie Mellon University, Email: nhaghtal@cs.cmu.edu
‡Technion, Haifa, Israel. Email: naor@cs.technion.ac.il
§Georgia Institute of Technology, Atlanta. Email: sebastian.pokutta@isye.gatech.edu
¶Georgia Institute of Technology, Atlanta. Email: mohitsinghr@gmail.com
‖Georgia Institute of Technology, Atlanta. Email: atorrico3@gatech.edu

19, 14, 16, 15, 17]. For instance, it is used to model the problem of picking a subset of candidate sensor locations for spatial monitoring for phenomena such as temperature, ph values, humidity, etc [15]. Here the goal is typically to find sensor locations that achieve the most coverage or give most information about the observed phenomena. Moreover, there are additional combinatorial constraints on the chosen locations, for example size, knapsack or more general constraints.

Submodular function optimization has been a natural tool that has been applied to these problems. Submodularity naturally captures the decreasing marginal gain in the coverage or the information acquired about relevant phenomena by using more sensors. In particular, the reduction in variance of observed parameters follows a diminishing returns property under natural models of parameter distribution underlying the phenomena [7]. While submodular optimization offers a natural approach to model these problems, there are two key shortcomings that are not well captured: (1) The sensors are typically used to measure different parameters at the same time. Observations for each of these parameters can be modelled via a different submodular function. Thus, we aim to select a subset of sensor locations that are good with respect to each of these submodular functions simultaneously. (2) Many of the phenomena being observed are non-stationary and highly variable in certain locations. To obtain a good solution, an approach is to use different submodular functions to model different spatial regions. Thus, again, we aim to obtain a solution that performs well under multiple criteria at the same time.

Robust submodular optimization naturally addresses these deficiencies by optimizing against several functions *simultaneously.* We refer the reader to [15] for other applications of robust submodular maximization in experimental design, variable selection, outbreak detection, and feature deletion.

In this paper we consider offline and online algorithms for *robust submodular maximization under matroid constraints.* In the offline version of the problem, we are given a collection of monotone submodular functions $f_i : 2^V \to \mathbb{R}_+$ on the same ground set $V = [n]$ with $i \in [k] := \{1, \dots, k\}$ and also a matroid $\mathcal{M} = (V, \mathcal{I})$. The goal is to select an independent set $S \in \mathcal{I}$ that maximizes $\min_{i\in[k]} f_i(S)$, i.e., we want to solve

$$\max_{S\in\mathcal{I}} \min_{i\in[k]} f_i(S).$$

This problem is NP-hard even when $k = 1$ and there has been significant progress over the last decade for this case [4]. Special cases of the problem, e.g., when $k$ is a constant [5], or when the matroid is uniform [15], have been studied extensively (see related work for details).

In the online version of the problem, we are given a matroid $\mathcal{M} = (V, \mathcal{I})$. At each time instant $t$, with $1 \le t \le T$, we choose a subset $S_t$, then we receive a new collection of non-negative monotone submodular functions $f_i^t$ with $i \in [k]$ and our reward is $\min_{i\in[k]} f_i^t(S_t)$. We assume these functions to be bounded, namely $0 \le f_i^t(S) \le 1$ for all $S \subseteq V$. Our goal is to maximize the total payoff $\sum_{t\in[T]} \min_{i\in[k]} f_i^t(S_t)$. We compare our performance with respect to the best static decision in hindsight, i.e., $\max_{S\in\mathcal{I}} \sum_{t\in[T]} \min_{i\in[k]} f_i^t(S)$. Since this offline version of the problem is NP-hard to approximate (see Section 2.1), we can only aim for a bi-criteria algorithm even in the regret setting. Therefore, we formally analyze our randomized algorithm by measuring the $(1-\epsilon)$ expected regret, defined as

$$\mathbf{Regret}_{1-\epsilon}(T) = (1-\epsilon) \cdot \max_{S\in\mathcal{I}} \sum_{t\in[T]} \min_{i\in[k]} f_i^t(S) - \sum_{t\in[T]} \min_{i\in[k]} \mathbb{E}\left[f_i^t(S_t)\right],$$

and we will design an algorithm with sub-linear $(1-\epsilon)$-regret.

### 1.1 Our Results and Contributions

Both offline and online problems are known to be NP-hard to approximate to any polynomial factor [15]. In this work we aim to design (nearly) optimal bi-criteria approximation algorithms that output a set of nearly optimal objective value, while ensuring the set is the union of few independent sets. In both settings we assume that the matroid is accessible via an independence oracle and the submodular functions are accessible via a value oracle.

For the offline setting of the problem we obtain the following result:

**Theorem 1.** *Let $(V, \mathcal{I})$ be a matroid and let $f_i : 2^V \to \mathbb{R}_+$ be a monotone submodular function for $i \in [k]$. Then, there is a randomized polynomial time algorithm that with constant probability returns a set $S^{\mathrm{ALG}}$, such that for all $i \in [k]$, for a given $0 < \epsilon < 1$,*

$$f_i(S^{\mathrm{ALG}}) \geq (1-\epsilon) \cdot \max_{S \in \mathcal{I}} \min_{j \in [k]} f_j(S),$$

*and $S^{\mathrm{ALG}} = S_1 \cup \cdots \cup S_m$ for $m = \ln(\frac{k}{\epsilon}) + O(1)$, and $S_1, \ldots, S_m \in \mathcal{I}$.*

This result relies on extending the continuous greedy algorithm [4] by adding new ingredients to it so that it can work in the robust setting. The continuous greedy algorithm uses a continuous, multilinear extension, of a discrete submodular function. Starting with the empty solution, at each (infinitesimal) step it picks a maximum weight independent set, where the weights are given by the gradient of the multilinear extension. We follow the same approach but face multiple issues. The first issue is which of the $k$ weight functions to use. Surprisingly, this issue can be resolved by observing that, at each time step, there is always a single independent set that achieves a good objective with respect to *all $k$ functions simultaneously.* While we cannot compute this set efficiently, we can compute a fractional set that achieves a performance at least as good by solving a linear program (see also [5]). This allows us to obtain a fractional independent set that simultaneously approximates all of the functions within $(1-\frac{1}{e})$ of the optimum providing us with a $(1-\frac{1}{e})$-approximation to our objective. Unfortunately, this fractional solution cannot be rounded to an integral solution via pipage rounding, as is the case for a single submodular function; recall that pipage rounding uses the function explicitly. To remedy this, we go back to the continuous greedy algorithm and run it longer, until time $\tau = \ln(\frac{k}{\epsilon}) + O(1)$. While the fractional solution obtained in the end will no longer be a fractional independent set, we show it is in the independent set polytope for $\mathcal{M}_\tau$ (the $\tau$-fold union of matroid $\mathcal{M}$) using the matroid union theorem. To round the fractional solution, we now use randomized swap rounding [5] over the matroid $\mathcal{M}_\tau$. The rounding gives us the desired set $S^{\mathrm{ALG}}$. While the value of each function is large only in expectation, running the algorithm up to time $t$ with a careful truncation of the submodular functions allows us to use Markov's inequality to prove the desired result. We present the main results in Section 2.

To our knowledge, for the online setting only the case $k = 1$ has been studied before, see [10]. Our approach is somewhat related, but we present a novel perspective to the robust problem by using the soft-min function in an online manner. We obtain the following main result:

**Theorem 2.** *For the online robust submodular optimization problem with parameters $\epsilon, \eta > 0$, there is a randomized algorithm that returns a set $S_t$ for each $1 \leq t \leq T$, such that it is the union of at most $O\left(\ln \frac{1}{\epsilon}\right)$ independent sets and*

$$\sum_{t\in[T]} \min_{i\in[k]} \mathbb{E}\left[f_i^t(S_t)\right] \geq (1-\epsilon)\cdot \max_{S\in\mathcal{I}} \sum_{t\in[T]} \min_{i\in[k]} f_i^t(S) - O\left(n^{\frac{5}{4}}\sqrt{T}\ln\frac{1}{\epsilon}\right).$$

We remark that since the bound in the above algorithm is on the minimum, taken over the expected function values at each iteration, the output set is a union of only $O\left(\ln \frac{1}{\epsilon}\right)$ independent sets, as compared to the offline setting where we needed the union of $O\left(\ln \frac{k}{\epsilon}\right)$ independent sets. Our results rely crucially on modifying the *Follow-the-Perturbed-Leader (FPL)* algorithm [12] to incorporate non-convexity arising from submodularity, as well as the robustness criteria. The main challenge is that the robustness criteria, $\min_{i\in[k]} f_i$, is a non-smooth function. Moreover, in the offline approach, we are able to optimize multiple objectives simultaneously, which is not feasible in an online setting. Our approach to deal with these issues is to use the *soft-min* function $\frac{1}{\alpha}\ln\sum_{i\in[k]} e^{-\alpha f_i}$, defined for some parameter $\alpha > 0$. While the choice of the specific soft-min function is seemingly arbitrary, a feature of the chosen soft-min which is crucial for us is that its gradient is a convex combination of the gradients of the $f_i$'s. However, choosing a small $\alpha$ leads to a large error in the approximation, compared to $\min_{i\in[k]} f_i$, and choosing a large $\alpha$ makes the soft-min function non-smooth, leading to large regret errors. We remedy this by optimizing both the $\alpha$ parameter and a discretization parameter $\delta$, effectively trading off the discretization error and the error arising from $\alpha$. The discretization parameter corresponds to a discretization of the continuous algorithm (see also [26]) into the discrete setting. The algorithm then runs a different instance of the FPL algorithm for each discretization. Putting all the FPL instances together, we arrive at the discretized version of the same recurrence as in the offline setting, giving us the result. We believe that the algorithm might be of independent interest to perform online learning over a minimum of many functions. The online result appears in Section 3.

### 1.2 Related Work

Building on the classical work of Nemhauser et al. [20], constrained submodular maximization problems have seen much progress recently (see for example [4, 5, 2, 3]). Robust submodular maximization generalizes submodular function maximization under a matroid constraint for which a $(1-\frac{1}{e})$-approximation is known [4] and is optimal. The problem has been studied for constant $k$ in [5] who give a $(1-\frac{1}{e}-\epsilon)$-approximation algorithm with running time $O\left(n^{\frac{k}{\epsilon}}\right)$. Closely related to our problem is the submodular cover problem where we are given a submodular function $f$, a target $b \in \mathbb{R}_+$, and the goal is to find a set $S$ of minimum cardinality such that $f(S) \geq b$. A simple reduction shows that robust submodular maximization under a cardinality constraint reduces to the submodular cover problem [15]. Wolsey [29] showed that the greedy algorithm gives an $O(\ln \frac{n}{\epsilon})$-approximation, where the output set $S$ satisfies $f(S) \geq (1-\epsilon)b$. Robust submodular cover under various constraints has been studied from an algorithmic viewpoint as well as a modeling tool for problems appearing in practice [23, 15]. Orlin et al. [21] study robust submodular optimization under a different measure. Influence maximization [13] in a network has been a successful application of submodular maximization and recently, He and Kempe [11] and Chen et al. [6] study the robust influence maximization problem.

There has been some prior work on online submodular function maximization that we briefly review here. In [26] the authors study the *budgeted maximum submodular coverage* problem and consider several feedback cases (let $B \in \mathbb{Z}_+$ be a bound for the budget): in the full information case, a $(1-1/e)$-expected regret of $O(\sqrt{BT\ln n})$ is achieved, however, there are $B$ experts considered which may be deemed large. Secondly, the opaque feedback model, which is reminiscent of the bandit model, a $(1-1/e)$-expected regret of $O(B(n\ln n)^{1/3}T^{2/3})$ is achieved. In a follow-up work [10], online submodular function maximization under both partition matroid constraints and general matroid constraints are considered. In the former case, an online greedy algorithm, which uses a discretization of "colors", gives an expected regret for full information $O(C\sum_{k=1}^{K}\sqrt{T\ln|P_k|})$, where $C$ is the number of colors, and the ground set is a disjoint union of $K$ sets $P_1,\ldots,P_K$. Optimizing over $C$, yields a $(1-1/e)$-regret of $\tilde{\Theta}(K^{3/2}T^{1/4}\sqrt{\mathrm{OPT}})$, where $\mathrm{OPT} = \max_{S\in\mathcal{I}}\sum_{t\in[T]} f_t(S)$. For bandit feedback, an extra term of $O((TnCK)^{2/3}(\ln n)^{1/3})$ is incurred. Finally, Golovin et al. [10] improve this previous result by presenting an online version of the continuous greedy algorithm, which relies on the Follow-the-Perturbed-Leader algorithm [12] and get a $(1-1/e)$-expected regret of $O(\sqrt{T})$. Similar to the previous approaches, our bi-criteria online algorithm will also use the Follow-the-Perturbed-Leader algorithm from [12] as a subroutine.

# 2 The Offline Case

## 2.1 Preliminaries

Consider a non-negative set function $f: 2^V \to \mathbb{R}_+$. Let us denote the marginal value for any subset $A \subseteq V$ and $e \in V$ by $f_A(e) := f(A+e) - f(A)$, where $A + e := A \cup \{e\}$. Recall that $f$ is *submodular* if and only if it satisfies the *diminishing returns property*. Namely, for any $e \in V$ and $A \subseteq B \subseteq V\backslash\{e\}$, $f_A(e) \geq f_B(e)$. Also, we say that $f$ is *monotone* if for any $A \subseteq B \subseteq V$, we have $f(A) \leq f(B)$.

For a set function $f$, its *multilinear extension* $F: [0,1]^V \to \mathbb{R}_+$ is defined for any $y \in [0,1]^V$ as the expected value of $f(S_y)$, where $S_y$ is the random set generated by drawing independently each element $e \in V$ with probability $y_e$. Formally,

$$F(y) = \mathbb{E}_{S\sim y}[f(S)] = \sum_{S\subseteq V} f(S)\prod_{e\in S} y_e \prod_{e\notin S}(1-y_e).$$

Observe, this is in fact an extension of $f$, since for any subset $S \subseteq V$, we have $f(S) = F(\mathbf{1}_S)$, where $\mathbf{1}_S(e) = 1$ if $e \in S$ and zero otherwise. The multilinear extension plays a crucial role in designing approximation algorithms for various constrained submodular optimization problems (see for example [4]). We will now present some general properties; for a proof we refer to [4]. For any vectors $x, y \in \mathbb{R}^V$, we denote by $x \vee y$ the vector obtained by taking coordinate-wise maximum.

**Fact 1.** *[Multilinear Extensions of Monotone Submodular Functions] Let $f$ be a monotone submodular function and $F$ its multilinear extension.*

1. *By monotonicity of $f$, we have $\frac{\partial F}{\partial y_e} \geq 0$ for any $e \in V$. This implies that for any $x \leq y$ coordinate-wise, $F(x) \leq F(y)$. On the other hand, by submodularity of $f$, $F$ is concave in any positive direction, i.e., for any $e, f \in V$ we have $\frac{\partial^2 F}{\partial y_e \partial y_f} \leq 0$.*

2. *Throughout the paper we will denote by $\nabla_e F(y) := \frac{\partial F(y)}{\partial y_e}$, and $\Delta_e F(y) := \mathbb{E}_{S \sim y}[f_S(e)]$. It is easy to see that $\Delta_e F(y) = (1 - y_e)\nabla_e F(y)$. Now, consider two points $x, y \in [0,1]^V$ and two sets sampled independently from these vectors: $S \sim x$ and $U \sim y$. Then, by submodularity*

$$f(S \cup U) \leq f(S) + \sum_{e \in V} \mathbf{1}_U(e) f_S(e). \tag{1}$$

3. *By taking expectation over $x$ and $y$ in (1), we obtain*

$$F(x \vee y) \leq F(x) + \sum_{e \in V} y_e \Delta_e F(x) \leq F(x) + \sum_{e \in V} y_e \nabla_e F(x).$$

*Therefore, we get the following important property*

$$F(x \vee y) \leq F(x) + y \cdot \nabla F(x). \tag{2}$$

Let $\mathcal{M} = (V, \mathcal{I})$ be a matroid with ground set $V = [n]$ and a family of independent sets $\mathcal{I}$. We denote the matroid polytope by $\mathcal{P}(\mathcal{M}) = \text{conv}\{\mathbf{1}_I \mid I \in \mathcal{I}\}$ and for any $\tau$, let $\tau\mathcal{P}(\mathcal{M}) = \text{conv}\{\tau \cdot \mathbf{1}_I \mid I \in \mathcal{I}\}$ be the scaling of the matroid polytope. Consider a fixed integer $k \geq 1$ and for each $i \in [k]$ let $f_i : 2^V \to \mathbb{R}_+$ be a monotone submodular function. We are interested in studying a robust version of the well-known problem of maximizing a submodular function under matroid constraints. Formally, we are interested in

$$\max_{S \in \mathcal{I}} \min_{i \in [k]} f_i(S). \tag{3}$$

### 2.2 Offline Algorithm and Analysis

In this section, we present a procedure that achieves a tight bi-criteria approximation for the robust submodular optimization problem (3) and prove Theorem 1. Our overall approach is to first find a fractional solution with a desirable approximation guarantee and then round it to an integral solution. We use a relaxation of a matroid to its convex hull to accommodate the search for a fractional solution.

For this algorithm, we need an estimate of the value of the optimal solution which we denote by OPT. We prove the following lemma which solves an approximate decision version of our optimization problem. The proof of Theorem 1 follows from the lemma and a search over an approximate value for OPT.

**Lemma 1.** *There is a randomized polynomial time algorithm that given $\gamma \leq \text{OPT}$ and $0 < \epsilon < 1$ returns with constant probability a set $S^{\text{ALG}}$ such that for all $i \in [k]$,*

$$f_i(S^{\text{ALG}}) \geq (1 - \epsilon) \cdot \gamma,$$

*where $S^{\text{ALG}} = \bigcup_{j \in [m]} S_j$ with $m = \ln(\frac{k}{\epsilon}) + O(1)$ and $S_j \in \mathcal{I}$ for each $j \in [m]$.*

We first finish the proof of Theorem 1 assuming Lemma 1.

**Theorem 1.** *Let* $(V, \mathcal{I})$ *be a matroid and let* $f_i : 2^V \to \mathbb{R}_+$ *be a monotone submodular function for* $i \in [k]$*. Then, there is a randomized polynomial time algorithm that with constant probability returns a set* $S^{\text{ALG}}$*, such that for all* $i \in [k]$*, for a given* $0 < \epsilon < 1$*,*

$$f_i(S^{\text{ALG}}) \geq (1-\epsilon) \cdot \max_{S \in \mathcal{I}} \min_{j \in [k]} f_j(S),$$

*and* $S^{\text{ALG}} = S_1 \cup \cdots \cup S_m$ *for* $m = \ln(\frac{k}{\epsilon}) + O(1)$*, and* $S_1, \ldots, S_m \in \mathcal{I}$*.*

*Proof of Theorem 1.* We apply the algorithm from Lemma 1 with approximation loss $\epsilon/2$ and with different values of $\gamma$, some of which may be larger than OPT, but at least one of them is guaranteed to satisfy $(1-\epsilon/2)\,\text{OPT} \leq \gamma \leq \text{OPT}$. At the end we return the set $S^{\text{ALG}}$ from our runs with the highest value of $\min_{i \in [k]} f_i(S^{\text{ALG}})$.

Before describing the set of candidate values of $\gamma$ that we try, note that if the algorithm succeeds for the particular value of $\gamma$ satisfying $(1-\epsilon/2)\,\text{OPT} \leq \gamma \leq \text{OPT}$, then we get

$$\min_{i \in [k]} f_i(S^{\text{ALG}}) \geq (1-\epsilon/2) \cdot \gamma \geq (1-\epsilon)\,\text{OPT},$$

and since we return the set with the highest $\min_{i \in [k]} f_i(S^{\text{ALG}})$, the algorithm's output will have the desired approximation guarantee.

It remains to show that a set of polynomial size of values for $\gamma$ exists such that one of them satisfies $(1-\epsilon)\,\text{OPT} \leq \gamma \leq \text{OPT}$. To this end we simply try $\gamma = nf_i(e)(1-\epsilon/2)^j$ for all $i \in [k]$, $e \in V$, and $j = 0, \ldots, \lceil \ln_{1-\epsilon/2}(1/n) \rceil$. Note that there exists an index $i \in [k]$ and a set $S \in \mathcal{I}$ such that $\text{OPT} = f_i(S)$. Now let $e = \text{argmax}_{e \in S} f_i(e)$. Because of submodularity and monotonicity we have $\frac{1}{|S|} f_i(S) \leq f_i(e) \leq f_i(S)$. So, we can conclude that $1 \geq \text{OPT}/nf_i(e) \geq 1/n$, which implies that $j = \lceil \ln_{1-\epsilon/2}(\text{OPT}/nf_i(e)) \rceil$ is in the correct interval, obtaining

$$(1-\epsilon/2)\,\text{OPT} \leq nf_i(e)(1-\epsilon/2)^j \leq \text{OPT}.$$

This finishes the proof. □

We remark that the dependency of the running time on $\epsilon$ can be made logarithmic by running a binary search on $j$ as opposed to trying all $j = 0, \ldots, \lceil \ln_{1-\epsilon/2}(1/n) \rceil$. We just need to run the algorithm from Lemma 1 for each $\gamma$ polynomially many times to make the failure probability exponentially small whenever $\gamma \leq \text{OPT}$.

The rest of this section is devoted to the proof of Lemma 1. To achieve a strong concentration bound when rounding the fractional solution, we truncate $f_i$ to $\min\{\gamma, f_i\}$. Hereafter, and with a slight abuse of notation, we use $f_i$ to refer to $\min\{\gamma, f_i\}$. Note that submodularity is preserved under this truncation. Also, we denote by $F_i$ the corresponding multilinear extension of $f_i$.

We describe the continuous process counterpart of the algorithm in this section and discuss the discretization details in the Appendix (see Section 5.2).

**Continuous Greedy.** We start a continuous gradient step process where $y(\tau)$ represents the point at time $\tau$ we are at. We start at $y(0) = 0$ and take continuous gradient steps in direction $\frac{dy}{d\tau} = v_{\text{all}}(y)$, such that $v_{\text{all}}(y)$ satisfies the following conditions:

(a) $v_{\text{all}}(y) \cdot \nabla F_i(y) \geq \gamma - F_i(y)$ for all $i \in [k]$,

(b) $v_{\text{all}}(y) \in \mathcal{P}(\mathcal{M})$, and

(c) $v_{\text{all}}(y) + y \in [0,1]^V$.

First, we show that such $v_{\text{all}}$ always exists. Take $x^*$ to be the indicator vector corresponding to the optimal solution. For any $y$, $v^* = (x^* - y) \vee 0$ is a positive direction satisfying Equation (2), and for all $i \in [k]$:

$$v^* \cdot \nabla F_i(y) \geq F_i(y + v^*) - F_i(y) = \gamma - F_i(y), \tag{4}$$

where the last equality holds since the truncated values of $f_i$ satisfy $F_i(y) \leq \gamma$ for all $y$. It is easy to check that $v^*$ satisfies the rest of the constraints (a)-(c), implying that there exists a feasible solution to the above system of linear inequalities. Therefore, we can solve a linear program defined by these inequalities to obtain a solution $v_{\text{all}}(y)$.

The above continuous process goes on until time $m = \ln(\frac{k}{\epsilon}) + O(1)$. We intentionally set $m > 1$ to obtain a (fractional) solution with a higher budget, which is useful for achieving a bi-criteria approximation. Next we show the following claim.

**Claim 1.** *For any $\tau \geq 0$, $y(\tau) \in \tau\mathcal{P}(\mathcal{M}) \cap [0,1]^V$ and for all $i \in [k]$,*

$$F_i(y(\tau)) \geq (1 - e^{-\tau})\gamma.$$

*Proof.* For any $\tau \geq 0$, we have

$$y(\tau) = \int_0^\tau v_{\text{all}}(y(s))\ ds = \int_0^1 \tau \cdot v_{\text{all}}(y(\tau s))\ ds.$$

So, $y(\tau)$ is a convex combination of vectors in $\tau\mathcal{P}(\mathcal{M})$. Moreover, $(v_{\text{all}}(y))_j = 0$ when $y_j = 1$, thus $y(\tau) \in [0,1]^V$ proving the first part of the claim.

For the second part, observe that we have for all $i \in [k]$,

$$\frac{dF_i(y(\tau))}{d\tau} = \frac{dy(\tau)}{d\tau} \cdot \nabla F_i(y(\tau)) = v_{\text{all}}(y(\tau)) \cdot \nabla F_i(y(\tau)) \geq \gamma - F_i(y(\tau)).$$

Moreover, $F_i(0) = 0$. Now we solve the above differential equation to obtain

$$F_i(y(\tau)) \geq (1 - e^{-\tau})\gamma.$$

Therefore $F_i(y(\tau)) \geq (1 - e^{-\tau})\gamma$ for each $i \in [k]$ as claimed. □

Thus, by setting $m = \ln(\frac{k}{\epsilon}) + O(1)$, we obtain that for all $i \in [k]$, $F_i(y(m)) \geq \left(1 - \frac{\epsilon}{k} \cdot c\right) \cdot \gamma$ for a desired constant $c < 1$. We next show how to obtain an integral solution.

**Rounding.** The next lemma summarizes our rounding. We first show that the fractional solution at time $m$ is contained in the matroid polytope of the $t$-fold union of matroid $\mathcal{M}$. We then do randomized swap rounding [5] in this matroid polytope. The truncation of the submodular functions, as well as properties of randomized swap rounding, play a crucial role in the proof.

**Lemma 2.** *Let $m = \ln(\frac{k}{\epsilon}) + O(1)$ be an integer and $y(m)$ be the output of the continuous greedy algorithm at time $m$ such that $F_i(y(m)) \geq \left(1 - \frac{\epsilon}{k} \cdot c\right) \cdot \gamma$ for each $i \in [k]$ and some constant $c < 1$. Then, there exists a polynomial time randomized algorithm that outputs a set $S$ such that with probability at least $\Omega(1)$ we have for each $i \in [k]$:*

$$f_i(S) \geq (1 - \epsilon) \cdot \gamma.$$

*Moreover, $S$ is a union of at most $m$ independent sets in $\mathcal{M}$.*

*Proof.* Let $\mathcal{M}_m = \bigvee_m \mathcal{M}$ be the $m$-fold union of matroid $\mathcal{M}$, i.e., $I$ is an independent set in $\mathcal{M}_m$ if and only if $I$ is a union of $m$ independent sets of $\mathcal{M}$. We denote by $\mathcal{I}_m$ the set of independent sets of $\mathcal{M}_m$. The rank function of $\mathcal{M}_m$ is given by $r_{\mathcal{M}_m}(S) = \min_{A \subseteq S} |S \setminus A| + m \cdot r_{\mathcal{M}}(A)$ (see [25]). We first show that $y = y(m)$ is in the convex hull of independent sets of matroid $\mathcal{M}_m$, i.e., $\mathcal{P}(\mathcal{M}_m)$. This polytope is given by $\mathcal{P}(\mathcal{M}_m) = \{x \in \mathbb{R}_+^V \mid \ x(S) \leq r_{\mathcal{M}_m}(S) \quad \forall\ S \subseteq V\}$, where $x(S) = \sum_{e \in S} x_e$. We now prove that $y \in \mathcal{P}(\mathcal{M}_m)$. For any $S \subseteq V$ and $A \subseteq S$, we have $y(S) = \sum_{e \in S \setminus A} y_e + y(A) \leq |S \setminus A| + m \cdot r_{\mathcal{M}}(A)$, where the last inequality is due to the fact that $y_e \leq 1$ for all $e$, and $y(A) \leq m \cdot r_{\mathcal{M}}(A)$ because $y \in m\mathcal{P}(\mathcal{M})$ by Claim 1. Therefore, $y \in \mathcal{P}(\mathcal{M}_m)$.

Next, we apply a randomized swap rounding [5] in matroid $\mathcal{M}_m$ to round the solution. A feature of the randomized swap rounding is that it is oblivious to the specific function $f_i$ used, and it is only a randomized function of the matroid space and the fractional solution.

**Theorem 1** (Theorem II.1 of [5])**.** *Let $f$ be a monotone submodular function and $F$ be its multilinear extension. Let $x \in \mathcal{P}(\mathcal{M})$ be a point in a matroid polytope and $R$ a random independent set obtained from it by randomized swap rounding. Then, $\mathbb{E}[f(R)] \geq F(x)$.*

Applying Theorem 1 to fractional solution $y$ and matroid $\mathcal{M}_m$, we obtain a random set $S \in \mathcal{I}^m$ such that

$$\mathbb{E}[f_i(S)] \geq F_i(y) \geq \left(1 - \frac{\epsilon}{k} \cdot c\right) \cdot \gamma$$

for all $i \in [k]$.

Due to the initial truncation, we have that $f_i(S) \leq \gamma$ with probability one. Thus, using Markov's inequality for each $i \in [k]$, we obtain that with probability at least $1 - \frac{c}{k}$, we have $f_i(S) \geq (1 - \epsilon)\gamma$. Therefore, taking a union bound over $k$ functions, we obtain that with probability at least $1 - c$, for all $i \in [k]$, we have $f_i(S) \geq (1 - \epsilon)\gamma$, thus producing an integral solution $S$ with max-min value at least $(1 - \epsilon)\gamma$ as claimed. □

### 2.3 Hardness of Approximation

We now present a hardness result for a general matroid that motivates the need for a bi-criteria approximation. We show that to achieve any polynomial approximation in polynomial time for the robust submodular maximization problem, one needs to look for a solution in the extended matroid, i.e., one needs to be allowed to return a set $S \in c\mathcal{I}$ where $c\mathcal{I} = \{S \mid \exists S_1, \ldots, S_c \in \mathcal{I}, S = \bigcup_i S_i\}$.

**Hardness result.** We provide a simple reduction from SET COVER. Consider a set cover instance over a universal set $U$ of $k$ elements and a family of subsets $\mathcal{A} \subseteq 2^U$ with $|\mathcal{A}| = n \leq 2^k$ admitting a set cover of size $\ell$. Construct a robust submodular maximization instance as follows. Let $\mathcal{M} = (V, \mathcal{I})$ be a uniform matroid over $\mathcal{A}$ of rank $\ell$, where $V = \mathcal{A}$ and $\mathcal{I} = \{\mathcal{B} \subseteq \mathcal{A} : |\mathcal{B}| \leq \ell\}$, i.e., subfamilies with at most $\ell$ sets in $\mathcal{A}$. For any $j \in U$, define $f_j : 2^{\mathcal{A}} \to \mathbb{R}_+$ as follows: for any $\mathcal{B} \subseteq \mathcal{A}$, $f_j(\mathcal{B}) = 1$ if there exists some $S \in \mathcal{B}$ such that $j \in S$ and zero otherwise. Clearly, $f_j$ is monotone and submodular for all $j \in [k]$. Moreover, any $\ell$-cover $C \subseteq \mathcal{A}$ for $U$ corresponds to a solution of value 1 for all $f_j$. Conversely, any solution of value 1 for all $f_j$ corresponds to a cover. By the hardness of approximation of SET COVER, a polynomial time algorithm cannot return a solution $\mathcal{B} \in ((1-\epsilon)\ln n)\mathcal{I}$ for any $\epsilon > 0$ that achieves an objective function value greater than zero, unless $P = NP$ [8]. Thus, we have the following lemma.

**Lemma 3.** *Unless $P = NP$, for any $\epsilon > 0$, there is no polynomial time algorithm for the robust submodular maximization problem under matroid constraints that returns a solution $S$ whose objective is within any positive factor of the optimum and $S$ is a union of $(1-\epsilon)\ln n$ independent sets.*

**Necessity of monotonicity.** In light of the approximation algorithms for non-monotone submodular function maximization under matroid constraints (see, e.g., [18]), one might hope that an analogous bi-criteria approximation algorithm could exist for robust non-monotone submodular function maximization. However, we show that even without any matroid constraints, getting any approximation in the non-monotone case is $NP$-hard.

**Lemma 4.** *Unless $P = NP$, no polynomial time algorithm can output a set $\tilde{S} \subseteq V$ given general submodular functions $f_1, \dots, f_k$ such that $\min_{i \in [k]} f_i(\tilde{S})$ is within a positive factor of $\max_{S \subseteq V} \min_{i \in [k]} f_i(S)$.*

*Proof.* We use a reduction from SAT. Suppose that we have a SAT instance with variables $x_1, \dots, x_n$. We let $V = \{1, \dots, n\}$. For every clause in the SAT instance we introduce a nonnegative linear (and by extension submodular) function. For a clause $\bigvee_{i \in A} x_i \vee \bigvee_{i \in B} \overline{x_i}$ define

$$f(S) := |S \cap A| + |B \setminus S|.$$

It is easy to see that $f$ is linear and nonnegative. If we let $S$ be the set of true variables in a truth assignment, then it is easy to see that $f(S) > 0$ if and only if the corresponding clause is satisfied. Therefore, finding a set $S$ such that all functions $f$ corresponding to different clauses are positive is as hard as finding a satisfying assignment for the SAT instance. □

# 3 The Online Case

## 3.1 Preliminaries

Consider a set of $k$ twice differentiable, real-valued functions $g_1, \dots, g_k$. Let $g_{min}$ be the minimum among these functions, i.e., for each point $x$ in the domain, define $g_{min}(x) := \min_{i \in [k]} g_i(x)$. This function can be approximated by using the so-called *soft-min* function $H$

defined as follows

$$H(x) = -\frac{1}{\alpha} \ln \sum_{i \in [k]} e^{-\alpha g_i(x)}.$$

where $\alpha > 0$ is a fixed parameter. Some key properties of this function are stated in the following lemma.

**Lemma 5.** *For any set of $k$ twice differentiable, real-valued functions $g_1, \ldots, g_k$, the soft-min function $H$ satisfies the following properties:*

1. *Bounds:*

$$g_{min}(x) - \frac{\ln(k)}{\alpha} \leq H(x) \leq g_{min}(x). \tag{5}$$

2. *Gradient:*

$$\nabla H(x) = \sum_{i \in [k]} p_i(x) \nabla g_i(x), \tag{6}$$

*where $p_i(x) := e^{-\alpha g_i(x)} / \sum_{j \in [k]} e^{-\alpha g_j(x)}$. Clearly, if $\nabla g_i \geq 0$ for all $i \in [k]$, then $\nabla H \geq 0$.*

3. *Hessian:*

$$\begin{aligned} \frac{\partial^2 H(x)}{\partial x_f \partial x_e} &= \sum_{i \in [k]} p_i(x) \left( -\alpha \frac{\partial g_i(x)}{\partial x_f} \frac{\partial g_i(x)}{\partial x_e} + \frac{\partial^2 g_i(x)}{\partial x_f \partial x_e} \right) \\ &\quad + \alpha \left( \sum_{i \in [k]} p_i(x) \frac{\partial g_i(x)}{\partial x_e} \right) \left( \sum_{i \in [k]} p_i(x) \frac{\partial g_i(x)}{\partial x_f} \right) \end{aligned} \tag{7}$$

*Moreover, if for all $i \in [k]$ we have $\left|\frac{\partial g_i}{\partial x_e}\right| \leq L_1$, and $\left|\frac{\partial^2 g_i}{\partial x_e \partial x_f}\right| \leq L_2$, then $\left|\frac{\partial^2 H}{\partial x_e \partial x_f}\right| \leq 2\alpha L_1^2 + L_2$.*

4. *Comparing the average of the $g_i$ functions with $H$: given $T > 0$ we have*

$$H(x) \leq \sum_{i \in [k]} p_i(x) g_i(x) \leq H(x) + \frac{n + \ln T}{\alpha} + \frac{\ln(k)}{\alpha} + \frac{k e^{-n}}{T}. \tag{8}$$

*So, for $\alpha > 0$ sufficiently large $\sum_{i \in [k]} p_i(x) g_i(x)$ is a good approximation of $H(x)$.*

For the proof we refer the interested reader to the Appendix. We will need the following lemma to prove Theorem 2.

**Lemma 6.** *Fix parameter $\delta > 0$. Consider $T$ sets of $k$ twice-differentiable functions, namely $\{g_i^1\}_{i \in [k]}, \ldots, \{g_i^T\}_{i \in [k]}$. Assume for all $t \in [T]$ and $i \in [k]$, we have $0 \leq g_i^t(x) \leq 1$ for any $x$ in the domain. Define the corresponding sequence of soft-min functions $H^1, \ldots, H^T$, with a common parameter $\alpha > 0$. Then, any sets of points $\{x^t\}_{t \in [T]}, \{y^t\}_{t \in [T]} \subseteq [0,1]^V$ with $|x^t - y^t| \leq \delta$ satisfy*

$$\sum_{t \in [T]} H^t(y^t) - \sum_{t \in [T]} H^t(x^t) \geq \sum_{e \in V} \sum_{t \in [T]} \nabla_e H^t(x^t)(y_e^t - x_e^t) - O(T n^3 \delta^2 \alpha).$$

### 3.2 Online Algorithm and Analysis

By relying on the FPL algorithm 2 (see Appendix), our goal is to design a bi-criteria online algorithm that in every time step chooses a subset $S_t$ such that it is the union of at most $O\left(\ln \frac{1}{\epsilon}\right)$ independent sets and the $(1-\epsilon)$-regret is sub-linear. In each time step $t$, we will use the soft-min function $H^t(y) = -\frac{1}{\alpha} \ln \sum_{i\in[k]} e^{-\alpha F_i^t(y)}$ defined by the corresponding $k$ multilinear extensions $F_i^t$ to generate a new decision set $S_t$. Similarly to the FPL algorithm (see Section 5.1), we need to assume some conditions regarding $H^t$ for any $t \in [T]$ and $\mathcal{P}(\mathcal{M})$:

1. bounded diameter of $\mathcal{P}(\mathcal{M})$, i.e., for all $y, y' \in \mathcal{P}(\mathcal{M})$, $\|y - y'\|_1 \leq D$;
2. for all $x, y \in \mathcal{P}(\mathcal{M})$, we require $\left|y \cdot \Delta H^t(x)\right| \leq L$;
3. for all $y \in \mathcal{P}(\mathcal{M})$, we require $\|\Delta H^t(y)\|_1 \leq A$,

where $\Delta_e H^t(y) = (1 - y_e)\nabla_e H^t(y) = \sum_{i\in[k]} p_i^t(y)\Delta_e F_i^t(y)$ for every $e \in V$, $t \in [T]$, and $\Delta_e F_i^t(y) = \mathbb{E}_{S\sim y}[f_i^t(S+e) - f_i^t(S)]$. Recall we assume that $0 \leq f_i^t \leq 1$ in the online case, so $0 \leq F_i^t(y) \leq 1$ for every $i \in [k]$ as well.

**Algorithm 1** OnlineSoftMin

**Input:** $\eta, \epsilon > 0$, $\alpha = n^2T^2$, $\delta = n^{-6}T^{-3}$.
**Output:** Sequence of sets $S_1, \ldots, S_T$.
1: **Sample** $p \sim [0, 1/\eta]^V$
2: **for** $t = 1$ to $T$ **do**
3: $\quad y_0^t = 0$
4: $\quad$ **for** $\tau \in \{\delta, 2\delta, \ldots, \ln \frac{1}{\epsilon}\}$ **do**
5: $\quad\quad z_\tau^t = \text{argmax}_{z\in\mathcal{P}(\mathcal{M})} \left[\sum_{s=1}^{t-1} \Delta H^s(y_{\tau-\delta}^s) + p\right]^\top z$
6: $\quad\quad$ **Update** $y_{\tau,e}^t = y_{\tau-\delta,e}^t + \delta(1 - y_{\tau-\delta,e}^t)z_{\tau,e}^t$ for each $e \in V$.
7: $\quad$ **Play** $S_t$ by doing randomized swap rounding on $y_{\ln\frac{1}{\epsilon}}^t$, receive functions $f_i^t$.

We state the guarantee of the online algorithm in the following theorem which directly implies Theorem 2 since we have $L \leq n$, $A \leq n$ and $D \leq \sqrt{n}$.

**Theorem 3.** *For the online robust submodular optimization problem with parameters $\epsilon, \eta > 0$, there is a randomized algorithm that returns a set $S_t$ for each $1 \leq t \leq T$, such that it is the union of at most $O\left(\ln \frac{1}{\epsilon}\right)$ independent sets and*

$$\sum_{t\in[T]} \min_{i\in[k]} \mathbb{E}\left[f_i^t(S_t)\right] \geq (1-\epsilon) \cdot \max_{S\in\mathcal{I}} \sum_{t\in[T]} \min_{i\in[k]} f_i^t(S) - O\left(R_\eta \sqrt{T} \ln \frac{1}{\epsilon}\right).$$

*where $R_\eta = \eta LAT + \frac{D}{\eta}$ . Moreover, for $\eta = \sqrt{D/LAT}$, we get* $\mathbf{Regret}_{1-\epsilon}(T) = O(\sqrt{T} \ln \frac{1}{\epsilon})$.

*Proof.* Consider the sequence of multilinear extensions $\{F_i^1\}_{i\in[k]}, \ldots, \{F_i^T\}_{i\in[k]}$ derived from the monotone submodular functions $f_i^t$ obtained during the dynamic process. Since $f_i^t$ is monotone for all $i \in [k]$, we can assume that $0 \leq F_i^t(y) \leq 1$ for any $y \in [0,1]^n$ and $i \in [k]$. Recall that for $\alpha = n^2T^2$ we denote by $H^t(y) = -\frac{1}{\alpha} \ln \sum_{i\in[k]} e^{-\alpha F_i^t(y)}$ the soft-min function defined by $\{F_i^t\}_{i\in[k]}$. Fix $\tau \in \{\delta, 2\delta, \ldots, \ln \frac{1}{\epsilon}\}$ with $\delta = n^{-6}T^{-3}$. According to the update in Algorithm 1, $\{y_\tau^t\}_{t\in[T]}$ and $\{y_{\tau-\delta}^t\}_{t\in[T]}$ satisfy conditions in Lemma 6. Thus, we obtain

$$\sum_{t\in[T]} H^t(y_\tau^t) - H^t(y_{\tau-\delta}^t) \geq \sum_{t\in[T]} \nabla H^t(y_{\tau-\delta}^t)\cdot(y_\tau^t - y_{\tau-\delta}^t) - O(Tn^3\delta^2\alpha).$$

Then, since $y_{\tau,e}^t = y_{\tau-\delta,e}^t + \delta(1-y_{\tau-\delta,e}^t)z_{\tau,e}^t$, we get

$$\begin{aligned}\sum_{t\in[T]} H^t(y_\tau^t) - H^t(y_{\tau-\delta}^t) &\geq \delta\sum_{t\in[T]}\sum_{e\in V}\nabla_e H^t(y_{\tau-\delta}^t)(1-y_{\tau-\delta,e}^t)z_{\tau,e}^t - O(Tn^3\delta^2\alpha)\\ &= \delta\sum_{t\in[T]}\Delta H^t(y_{\tau-\delta}^t)\cdot z_\tau^t - O(Tn^3\delta^2\alpha). \end{aligned} \tag{9}$$

Observe that an FPL algorithm is implemented for each $\tau$, so we can state a regret bound for each $\tau$ by using Theorem 2 (see Appendix). Specifically,

$$\max_{z\in\mathcal{P}(\mathcal{M})}\mathbb{E}\left[\sum_{t\in[T]}\Delta H^t(y_{\tau-\delta}^t)\cdot z\right] - \mathbb{E}\left[\sum_{t\in[T]}\Delta H^t(y_{\tau-\delta}^t)\cdot z_\tau^t\right] \leq R_\eta,$$

where $R_\eta$ is the regret guarantee from Theorem 2 for a given $\eta > 0$. By taking expectation in (9) and using the regret bound, we obtain

$$\begin{aligned}\mathbb{E}\left[\sum_{t\in[T]} H^t(y_\tau^t) - H^t(y_{\tau-\delta}^t)\right] &\geq \delta\left(\max_{z\in\mathcal{P}(\mathcal{M})}\mathbb{E}\left[\sum_{t\in[T]}\Delta H^t(y_{\tau-\delta}^t)\cdot z\right]\right) - \delta R_\eta - O(Tn^3\delta^2\alpha)\\ &\geq \delta\mathbb{E}\left(\sum_{t\in[T]}\left[H^t(x^*) - \sum_{i\in[k]} p_i^t(y_{\tau-\delta}^t)F_i^t(y_{\tau-\delta}^t)\right]\right) - \delta R_\eta - O(Tn^3\delta^2\alpha), \end{aligned} \tag{10}$$

where $x^*$ is the true optimum for $\max_{x\in\mathcal{P}(\mathcal{M})}\sum_{t\in[T]}\min_{i\in[k]} F_i^t(x)$. Observe that (10) follows from monotonicity and submodularity of each $f_i^t$:

$$\begin{aligned}\Delta H^t(y)^\top z &= \sum_{i\in[k]} p_i^t(y)\Delta F_i^t(y)\cdot z\\ &\geq \sum_{i\in[k]} p_i^t(y)F_i^t(x^*) - \sum_{i\in[k]} p_i^t(y)F_i^t(y)\\ &\geq F_{min}^t(x^*) - \sum_{i\in[k]} p_i^t(y)F_i^t(y)\\ &\geq H^t(x^*) - \sum_{i\in[k]} p_i^t(y)F_i^t(y). \end{aligned}$$

By applying (8) in expression (10) we get

$$\begin{aligned}\mathbb{E}\left[\sum_{t\in[T]} H^t(y_\tau^t) - H^t(y_{\tau-\delta}^t)\right] &\geq \delta\mathbb{E}\left(\sum_{t\in[T]} H^t(x^*) - H^t(y_{\tau-\delta}^t)\right) - \delta R_\eta - O(Tn^3\delta^2\alpha)\\ &\quad -\delta T\left(\frac{n+\ln T}{\alpha} - \frac{\ln(k)}{\alpha} - \frac{ke^{-n}}{T}\right), \end{aligned} \tag{11}$$

Given the choice of $\alpha$ and $\delta$, the last two terms in the right-hand side of (11) is small, so we can state the following

$$\sum_{t\in[T]} H^t(x^*) - \mathbb{E}\left[\sum_{t\in[T]} H^t(y^t_\tau)\right] \leq (1-\delta)\left(\sum_{t\in[T]} H^t(x^*) - \mathbb{E}\left[\sum_{t\in[T]} H^t(y^t_{\tau-\delta})\right]\right) + 2\delta R_\eta$$

By iterating $\frac{1}{\delta}\ln\frac{1}{\epsilon}$ times in $\tau$, we get

$$\begin{aligned}\sum_{t\in[T]} H^t(x^*) - \mathbb{E}\left[\sum_{t\in[T]} H^t(y^t_{\ln\frac{1}{\epsilon}})\right] &\leq (1-\delta)^{\frac{1}{\delta}\ln\frac{1}{\epsilon}}\left(\sum_{t\in[T]} H^t(x^*) - \sum_{t\in[T]} H^t(y^t_0)\right) + 2R_\eta \ln\frac{1}{\epsilon} \\ &\leq \epsilon\left[\sum_{t\in[T]} H^t(x^*) + \frac{\ln(k)}{n^2 T}\right] + 2R_\eta \ln\frac{1}{\epsilon},\end{aligned}$$

where in the second inequality we used $(1-\delta) \leq e^{-\delta}$. Given that the term $\epsilon\frac{\ln(k)}{n^2T}$ is small (for $T$ and $n$ sufficiently large) we can bound it by $R_\eta \ln\frac{1}{\epsilon}$. Since $\alpha$ is sufficiently large, we can apply (5) to obtain the following regret bound

$$(1-\epsilon)\cdot\sum_{t\in[T]}\min_{i\in[k]} F_i^t(x^*) - \mathbb{E}\left[\sum_{t\in[T]}\min_{i\in[k]} F_i^t\left(y^t_{\ln\frac{1}{\epsilon}}\right)\right] \leq 3R_\eta \ln\frac{1}{\epsilon}.$$

Since we are doing randomized swap rounding on each $y^t_{\ln\frac{1}{\epsilon}}$, Theorem 1 shows that there is a random set $S_t$ that is independent in $\mathcal{M}_{\ln\frac{1}{\epsilon}}$ (i.e., $S_t$ is the union of at most $O(\ln(1/\epsilon))$ independent sets in $\mathcal{M}$) such that $\mathbb{E}\left[f_i^t(S_t)\right] \geq F_i^t\left(y^t_{\ln\frac{1}{\epsilon}}\right)$. Thus we obtain

$$(1-\epsilon)\cdot\max_{S\in\mathcal{I}}\sum_{t\in[T]}\min_{i\in[k]} f_i^t(S) - \sum_{t\in[T]}\min_{i\in[k]}\mathbb{E}\left[f_i^t(S_t)\right] \leq 3R_\eta \ln\frac{1}{\epsilon}.$$

□

**Observation 1.** *Theorem 2 can be easily extended to adaptive adversaries by sampling in each stage $t \in [T]$ a different vector $p \sim [0, 1/\eta]^V$ as shown in [12].*

# 4 Acknowledgements

This research was partially supported by NSF Award CCF-1717947, NSF CAREER award CMMI-1452463, an IBM Ph.D. fellowship, and a Microsoft Research Ph.D. fellowship.

# References

[1] N. Buchbinder and M. Feldman. Deterministic algorithms for submodular maximization problems. In *Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 392–403, 2016.

[2] N. Buchbinder, M. Feldman, J. S. Naor, and R. Schwartz. Submodular maximization with cardinality constraints. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1433–1452. Society for Industrial and Applied Mathematics, 2014.

[3] N. Buchbinder, M. Feldman, and R. Schwartz. Comparing apples and oranges: Query trade-off in submodular maximization. *Mathematics of Operations Research*, 42(2):308–329, 2016.

[4] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[5] C. Chekuri, J. Vondrak, and R. Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 575–584. IEEE, 2010.

[6] W. Chen, T. Lin, Z. Tan, M. Zhao, and X. Zhou. Robust influence maximization. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 795–804, New York, NY, USA, 2016. ACM.

[7] A. Das and D. Kempe. Algorithms for subset selection in linear regression. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 45–54. ACM, 2008.

[8] I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 624–633. ACM, 2014.

[9] A. Ene and H. L. Nguyen. Constrained submodular maximization: Beyond 1/e. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 248–257, 2016.

[10] D. Golovin, A. Krause, and M. Streeter. Online submodular maximization under a matroid constraint with application to learning assignments. Technical Report, arXiv, 2014.

[11] X. He and D. Kempe. Robust influence maximization. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 885–894, New York, NY, USA, 2016. ACM.

[12] A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291 – 307, 2005.

[13] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.

[14] A. Krause and C. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *21st Conference on Uncertainty in Artificial Intelligence (UAI)*, page 5, 2005.

[15] A. Krause, H. B. McMahan, C. Guestrin, and A. Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9(Dec):2761–2801, 2008.

[16] A. Krause, R. Rajagopal, A. Gupta, and C. Guestrin. Simultaneous placement and scheduling of sensors. In *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*, pages 181–192, 2009.

[17] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.*, 9:235–284, June 2008.

[18] J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 323–332. ACM, 2009.

[19] H. Lin and J. A. Bilmes. How to select a good training-data subset for transcription: submodular active selection for sequences. In *INTERSPEECH*, 2009.

[20] 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.

[21] J. B. Orlin, A. S. Schulz, and R. Udwani. Robust monotone submodular function maximization. In *Proceedings of the 18th International Conference on Integer Programming and Combinatorial Optimization - Volume 9682*, IPCO 2016, pages 312–324, New York, NY, USA, 2016. Springer-Verlag New York, Inc.

[22] T. Powers, J. Bilmes, D. W. Krout, and L. Atlas. Constrained robust submodular sensor selection with applications to multistatic sonar arrays. In *2016 19th International Conference on Information Fusion (FUSION)*, pages 2179–2185, July 2016.

[23] T. Powers, J. Bilmes, S. Wisdom, D. W. Krout, and L. Atlas. Constrained robust submodular optimization. In *NIPS OPT2016 workshop*, 2016.

[24] A. Rakhlin and A. Tewari. Lecture notes on online learning. *Draft, April*, 2009.

[25] A. Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

[26] M. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, pages 1577–1584, 2008.

[27] M. Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41 – 43, 2004.

[28] J. Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 67–74. ACM, 2008.

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

## 5 Appendix

*Proof Lemma 5.* We will just prove properties 1 and 4, since the rest is an straightforward calculation.

1. First, for all $i \in [k]$ we have $e^{-\alpha g_i(x)} \le e^{-\alpha g_{min}(x)}$. Thus,

$$H(x) = -\frac{1}{\alpha} \ln \sum_{i\in[k]} e^{-\alpha g_i(x)} \ge -\frac{1}{\alpha} \ln \left(k e^{-\alpha g_{min}(x)}\right) = g_{min}(x) - \frac{\ln(k)}{\alpha}$$

On the other hand, $\sum_{i\in[k]} e^{-\alpha g_i(x)} \ge e^{-\alpha g_{min}(x)}$. Hence,

$$H(x) \le -\frac{1}{\alpha} \ln \left(e^{-\alpha g_{min}(x)}\right) = g_{min}(x).$$

4. Let us consider sets $A_1 = \{i \in [k] : \ g_i(x) \le g_{min}(x) + (n + \ln T)/\alpha\}$ and $A_2 = \{i \in [k] : \ g_i(x) > g_{min}(x) + (n + \ln T)/\alpha\}$. Our intuitive argument is the following: when $\alpha$ is sufficiently large, those $p_i(x)$'s with $i \in A_2$ are exponentially small, and $p_i(x)$'s with $i \in A_1$ go to a uniform distribution over elements in $A_1$. First, observe that for each $i \in A_2$ we have

$$p_i(x) = \frac{e^{-\alpha g_i(x)}}{\sum_{i\in[k]} e^{-\alpha g_i(x)}} < \frac{e^{-\alpha[g_{min}(x)+(n+\ln T)/\alpha]}}{e^{-\alpha g_{min}(x)}} = \frac{e^{-n}}{T},$$

so $\sum_{i\in A_2} p_i(x) g_i(x) \le \frac{ke^{-n}}{T}$. On the other hand, for any $i \in A_1$ we have

$$\sum_{i\in A_1} p_i(x) g_i(x) \le \left(g_{min}(x) + \frac{n + \ln T}{\alpha}\right) \sum_{i\in A_1} p_i(x) \le H(x) + \frac{n+\ln T}{\alpha} + \frac{\ln(k)}{\alpha}$$

where in the last inequality we used (5). Therefore,

$$\sum_{i\in[k]} p_i(x) g_i(x) \le H(x) + \frac{n+\ln T}{\alpha} + \frac{\ln(k)}{\alpha} + \frac{ke^{-n}}{T}.$$

Finally, the other inequality is clear since $\sum_{i\in[k]} p_i(x) g_i(x) \ge g_{min}(x) \ge H(x)$.

□

*Proof Lemma 6.* For every $t \in [T]$ define a matroid $\mathcal{M}_t = (V \times \{t\}, \mathcal{I} \times \{t\}) = (V_t, \mathcal{I}_t)$. Given this, the union matroid is given by a ground set $V^{[T]} = \bigcup_{t=1}^T V_t$, and independent set family $\mathcal{I}^{[T]} = \{S \subseteq V^{1:T} : \ S \cap V_t \in \mathcal{I}_t\}$. Define $\mathbb{H}(X) := \sum_{t\in[T]} H^t(x^t)$ for any matrix $X \in \mathcal{P}(\mathcal{M})^T$, where $x^t$ denotes the $t$-th column of $X$. Clearly, $\nabla_{(e,t)} \mathbb{H}(X) = \nabla_e H^t(x^t)$. Moreover, the Hessian corresponds to

$$\nabla^2_{(e,t),(f,s)} \mathbb{H}(X) = \begin{cases} 0 & \text{if } t \ne s \\ \nabla^2_{e,f} \mathbb{H}^t(x^t) & \text{if } t = s \end{cases}$$

Consider any $X, Y \in \mathcal{P}(\mathcal{M})^T$ with $|y_e^t - x_e^t| \leq \delta$. Therefore, a Taylor's expansion of $\mathbb{H}$ gives

$$\mathbb{H}(Y) = \mathbb{H}(X) + \nabla\mathbb{H}(X) \cdot (Y - X) + \frac{1}{2}(Y - X)^\top \nabla^2\mathbb{H}(\xi) \cdot (Y - X)$$

where $\xi$ is on the line between $X$ and $Y$. If we expand the previous expression we obtain

$$\mathbb{H}(Y) - \mathbb{H}(X) = \sum_{e \in V} \sum_{t \in [T]} \nabla_e H^t(x^t)(y_e^t - x_e^t) + \frac{1}{2} \sum_{e,f \in V} \sum_{t \in [T]} (y_e^t - x_e^t) \nabla_{e,f}^2 H^t(\xi)(y_f^t - x_f^t)$$

Finally, by using property 3 in Lemma 5

$$\mathbb{H}(Y) - \mathbb{H}(X) \geq \sum_{e \in V} \sum_{t=1}^{T} \nabla_e H^t(x^t)(y_e^t - x_e^t) - O(Tn^3\delta^2\alpha),$$

which is equivalent to

$$\sum_{t \in [T]} H^t(y^t) - \sum_{t \in [T]} H^t(x^t) \geq \sum_{e \in V} \sum_{t \in [T]} \nabla_e H^t(x^t)(y_e^t - x_e^t) - O(Tn^3\delta^2\alpha).$$

□

### 5.1 Follow-the-Perturbed-Leader algorithm

In this section, we briefly recall the well-known Follow-the-Perturbed-Leader (FPL) algorithm introduced in [12] and used in many online optimization problems (see e.g., [24]). The classical online learning framework is as follows: Consider a dynamic process over $T$ time steps. In each time step $t \in [T]$, a decision-maker has to choose a point $d_t \in \mathcal{D}$ from a fixed (possibly infinite) set of actions $\mathcal{D} \subseteq \mathbb{R}^n$, then an adversary chooses a vector $s_t$ from a set $\mathcal{S}$. Finally, the player observes vector $s_t$ and receives reward $s_t \cdot d_t$, and the process continues. The goal of the player is to maximize the total reward $\sum_{t=1}^T d_t \cdot s_t$, and we compare her performance with respect to best single action picked in hindsight, i.e., $\max_{d \in \mathcal{D}} \sum_{t=1}^T s_t \cdot d$. This performance with respect to the best single action in hindsight is called (expected) *regret*, formally:

$$\mathbf{Regret}(T) = \max_{d \in \mathcal{D}} \sum_{t \in [T]} s_t \cdot d - \mathbb{E}\left[\sum_{t \in [T]} d_t \cdot s_t\right].$$

Kalai and Vempala [12] showed that even if one has only access to a linear programming oracle for $\mathcal{D}$, i.e., we can solve $\max_{d \in \mathcal{D}} s \cdot d$ for any $s \in \mathcal{S}$, then the FPL algorithm 2 achieves sub-linear regret, specifically $O(\sqrt{T})$.

In order to state the main result in [12], we need the following. We assume that the decision set $\mathcal{D}$ has diameter at most $D$, i.e., for all $d, d' \in \mathcal{D}$, $\|d - d'\|_1 \leq D$. Further, for all $d \in \mathcal{D}$ and $s \in \mathcal{S}$ we assume that the absolute loss is bounded by $L$, i.e., $|d \cdot s| \leq L$ and that the $\ell_1$-norm of the loss vectors is bounded by $A$, i.e., for all $s \in \mathcal{S}$, $\|s\|_1 \leq A$.

**Theorem 2** ([12]). *Let $s_1, \ldots, s_T \in \mathcal{S}$ be a sequence of states. Running the FPL algorithm 2 with parameter $\epsilon \leq 1$ ensures regret*

$$\mathbf{Regret}(T) \leq \eta LAT + \frac{D}{\eta}.$$

*Moreover, if we choose $\eta = \sqrt{D/LAT}$, then $\mathbf{Regret}(T) \leq 2\sqrt{DLAT} = O(\sqrt{T})$.*

**Algorithm 2** Follow-the-Perturbed-Leader (FPL) [12]

**Input:** Parameter $\eta > 0$

**Output:** Sequence of decisions $d_1, \ldots, d_T$

1: **Sample** $p \sim [0, 1/\eta]^n$.

2: **for** $t = 1$ to $T$ **do**

3: **Play** $d_t = \text{argmax}_{d \in \mathcal{D}} \left( \sum_{j=1}^{t-1} s_j + p \right)^\top d$.

### 5.2 Discretized Algorithm

In this section we describe how the continuous greedy process described in Section 2.2 can be discretized and implemented. Most of this section is directly adapted from [28]. The two main ingredients needed to implement the continuous greedy process are discretization of time and estimation of the values $\nabla F_i(y)$ and $F_i(y)$ using random sampling. We assume time is discretized in steps of length $\delta$. In order to estimate $F_i(y)$ we sample a random set $S_y$ where each element $e$ is picked independently with probability $y_e$. We average the observed values of $f_i(S_y)$ for $M$ such samples, where $M$ is large enough. We use a similar strategy for estimating $\nabla F_i(y)$. An outline of the algorithm follows.

1. Start with $\tau = 0$ and $y(\tau) = 0$.

2. Use random sampling to get estimates $\nabla \tilde{F_i}(y)$ and $\tilde{F_i(y)}$ for $\nabla F_i(y)$ and $F_i(y)$ respectively.

3. Solve the following linear program to find the direction of movement $v_{\text{all}}$.

$$\begin{array}{ll} \max_{u, v_{\text{all}}} & u \\ \text{subject to} & v_{\text{all}} \in \mathcal{P}(\mathcal{M}), \\ & y(\tau) + v_{\text{all}} \in [0,1]^V, \\ & v_{\text{all}} \cdot \nabla \tilde{F_i}(y) \geq \gamma - \tilde{F_i(y)} + u \qquad \forall i \in [k]. \end{array}$$

4. Let $y(\tau + \delta) = y(\tau) + \delta v_{\text{all}}$. Set $\tau$ to $\tau + \delta$ and repeat until the desired time $\tau = \ln(\frac{k}{\epsilon}) + O(1)$ is reached.

Note that we still have $y(\tau) \in \tau \mathcal{P}(\mathcal{M}) \cap [0,1]^V$, i.e., the first part of Claim 1 is still correct. We prove an approximate version of the second part next.

Since $f_i(S)$ lies between 0 and $\gamma$ we can take $M$ to be large enough so that the error of estimation in $\nabla \tilde{F_i}(y)$ and $\tilde{F_i(y)}$ is inverse-polynomially small relative to $\gamma$ with very high probability; see the proof of Lemma 4.2 in [28] for details of the Chernoff bound. This ensures that the $v_{\text{all}}$ that we find satisfies for all $i \in [k]$,

$$v_{\text{all}} \cdot \nabla F_i(y) \geq \gamma - F_i(y) - \gamma / \operatorname{poly}(n, k).$$

When $\delta$ is small enough the second order effects of the changes in $F_i(y)$ can be ignored when going from $\tau$ to $\tau + \delta$. This is formalized by the following inequality whose proof is identical to the proof of Lemma 4.2 in [28].

$$F_i(y + \delta v_{\text{all}}) - F_i(y) \geq \delta(1 - n\delta) v_{\text{all}} \cdot \nabla F_i(y).$$

It now follows that

$$F_i(y + \delta v_{\text{all}}) - F_i(y) \geq \delta(1 - n\delta)(\gamma - F_i(y) - \gamma/\operatorname{poly}(n,k)).$$

Let $\tilde{\gamma} = (1 - n\delta)(1 - 1/\operatorname{poly}(n,k))\gamma$. Then it follows that

$$\tilde{\gamma} - F_i(y + \delta v_{\text{all}}) \leq (1 - \delta)(\tilde{\gamma} - F_i(y)),$$

and by induction

$$\tilde{\gamma} - F_i(y(\tau)) \leq \tilde{\gamma}(1 - \delta)^{\tau/\delta},$$

or equivalently

$$F_i(y(\tau)) \geq (1 - (1 - \delta)^{\tau/\delta})\tilde{\gamma} \geq (1 - e^{-\tau})\tilde{\gamma}.$$

By setting $\delta$ inverse-polynomially small enough and $M$ polynomially large enough we can ensure that the relative error between $\tilde{\gamma}$ and $\gamma$ is inverse-polynomially small which gives us an approximate version of Claim 1.