Minimum Robust Multi-Submodular Cover for Fairness

Lan N. Nguyen  My T. Thai

Department of Computer and Information Science and Engineering
University of Florida, Gainesville, Florida 32611
lan.nguyen@ufl.edu mythai@cise.ufl.edu

Abstract

In this paper, we study a novel problem, Minimum Robust Multi-Submodular Cover for Fairness (MıNRF), as follows: given a ground set \( V \); \( m \) monotone submodular functions \( f_1, ..., f_m \); \( m \) thresholds \( T_1, ..., T_m \) and a non-negative integer \( r \), MıNRF asks for the smallest set \( S \) such that for all \( i \in [m] \), \( \min_{|X| \leq r} f_i(S \setminus X) \geq T_i \). We prove that MıNRF is inapproximable within \((1-\epsilon)\ln m\) and no algorithm, taking fewer than exponential number of queries in term of \( r \), is able to output a feasible set to MıNRF with high certainty. Three bicriteria approximation algorithms with performance guarantees are proposed: one for \( r = 0 \), one for \( r = 1 \), and one for general \( r \). We further investigate our algorithms’ performance in two applications of MıNRF, Information Propagation for Multiple Groups and Movie Recommendation for Multiple Users. Our algorithms have shown to outperform baseline heuristics in both solution quality and the number of queries in most cases.

Introduction

In a minimum submodular cover, given a ground set \( V \), a monotone submodular set function \( f: 2^V \rightarrow \mathbb{R} \) and a number \( T \), the problem asks for a set \( S \subseteq V \) of minimum size such that \( f(S) \geq T \). This problem was studied extensively in the literature because of its wide-range applications, e.g. data summarization (Mirzasoleiman et al. 2015), active set selection (Norouzi-Fard et al. 2016), recommendation systems (Gulley and Bilmes 2011), information propagation in social networks (Kuhnle et al. 2017), and network resilience assessment (Nguyen and Thai 2019, Dinh and Thai 2014).

However, a single objective function \( f \) may not well model several practical applications where achieving multiple goals is required, especially when group fairness is considered. Let us consider the following two representative applications.

Information Propagation in Social Network for Multiple Groups. Social networks are cost-effective tools for information spreading by selecting a set of highly influential people (called seed set) that, through the word-of-mouth effects, the information will be reached to a large number of population (Kuhnle et al. 2017, Nguyen, Zhou, and Thai 2019, Zhang et al. 2014, Nguyen, Thai, and Dinh 2016). For many applications (e.g. broadening participants in STEM), it is important to ensure the diversity and fairness among different ethnicities and genders. Therefore, those applications aim to find a minimum seed set such that the information can reach to each group in a fair manner.

Items Recommendation for Multiple Users. Recommendation systems aim to make a good recommendation, e.g. a set of items, which can match users’ preferences. In many situations, an item can be served for multiple users, e.g. a family. In this problem, a user’s utility level to a set of items is modelled under a monotone submodular function. The objective, therefore, is to find the smallest set of items, from which we can design a recommendation for all users in a way that all reach a certain utility level.

Additionally, these problems require robustness in the solution set, in the sense that the solution satisfies all the constraints even if some elements were removed. Those removal can be from various reasons. For instance, in information propagation, a subset of users may decide not to spread the information (Bogunovic et al. 2017). Or in recommendation systems, due to the uncertainty of underlying data, information of some items may not be accurate (Orlin, Schulz, and Udwani 2018). Achieving a reasonable prior distribution on the removed elements may not be practical in many situations. Or even when the distribution is known, it is critical to obtain a robust solution with a high level of certainty, in a way that all goals are still achieved under the worst-case removal. Motivated by that observation, in this work, we study a novel problem, Minimum Robust Multi-Submodular Cover for Fairness (MıNRF), defined as follows.

Definition 1. (MıNRF) Given a finite set \( V \); \( m \) monotone submodular functions \( f_1, ..., f_m \) where \( f_i: 2^V \rightarrow \mathbb{R}_+ \); \( m \) non-negative numbers \( T_1, ..., T_m \), and a non-negative integer \( r \), find a set \( S \subseteq V \) of minimum size such that \( \forall i \in [m], \min_{|X| \leq r} f_i(S \setminus X) \geq T_i \).

MıNRF’s objective can also be understood as finding \( r \) of minimum size such that for all \( X \subseteq V \) that \( |X| \leq r \) and \( i \in [m] \), \( f_i(S \setminus X) \geq T_i \). Beside two applications as stated early, MıNRF can also be applied in many other applications, such as Sensor Placement (Orlin, Schulz, and Udwani 2018), Ohsaka and Yoshida 2015, which guarantees each measurement (e.g. temperature, humidity) reaches a certain information gain while being robust against sensors’ failure;
or Feature Selection [Qian et al. 2017] [Orlin, Schulz, and Udwan 2018], which aims for a smallest set of features that can retain information at a certain level while guaranteeing the set is not dependent on a few features.

To our knowledge, this work provides the first solutions to RF for a general r. In this paper, we pay attention to recent studies on minimum multi-submodular cover (MRF) when r = 0, and robust submodular optimization.

With minimum submodular cover (m = 1, r = 0), Goyal et al. [2013] showed that the classical greedy algorithm is able to obtain a bi-criteria ratio of $O(\ln \alpha^{-1})$. If we run m instances of greedy, each with a constraint $f_i(\cdot) \geq T_i$, get m output $S_i$ and returns $\bigcup_{i \in [m]} S_i$, we can get the ratio of $O(m \ln \alpha^{-1})$ for MRF when r = 0. In this paper, we aim for algorithms with better ratios.

Krause et al. [2008] was the first one proposing a problem of minimum multi-submodular cover; and the problem was then further studied by Mirzasoleiman, Zadimoghaddam, and Karbasi [2016]; Iyer and Bilmes [2013]. In general, their solution made a reduction from multiple submodular objectives to a single instance of a submodular cover problem by defining $F(\cdot) = \sum_{i \in [m]} \min(f_i(\cdot), T)$ (all thresholds are the same); and find S of minimum size such that $F(S) = mT$. Two algorithms were proposed, GREEDY (Krause et al. 2008), Iyer and Bilmes [2013] and THRESGR (Mirzasoleiman, Zadimoghaddam, and Karbasi [2016]). Their performance analysis requires $\{f_i\}_{i \in [m]}$ to receive values in $\mathbb{Z}$ to obtain ratio of $O(\ln \max_{i \in [m]} F(\{\{\})$).

However, requiring $\{f_i\}_{i \in [m]}$ to receive values in $\mathbb{Z}$ is not practical in many applications. In our work, we re-investigate Greedy and THRESGR’s performance without such requirement. Also, our RANDGR algorithm differs from such methods in which RANDGR does not unite objectives into a single function. Furthermore, RANDGR adds randomness to reduce the query complexity while still obtaining an asymptotically equal performance guarantee to that of GREEDY.

With robust submodular optimization, the concept of finding set that is robust to the removal of r elements was first proposed by Orlin, Schulz, and Udwan [2018]. However, their problem is a maximization, namely Robust Submodular Maximization (RSM), defined as follows: Given a ground set V, a monotone submodular function $f_1, \ldots, f_r$ and a non-negative integers $k$, find $S \subseteq \mathbb{Z}$ such that $F(S \setminus V) \geq \alpha T$. This problem was later studied further by Bogunovic et al. [2017], Mitrovic et al. [2017], Srba, Wilder, and Jegelka [2019], and Anari et al. [2019]. RSM and MRF both focus on the worst-case scenario, where the removal of r elements has the greatest impact on the returned solution. Other than that, the two problems are basically different and we are unable to adopt existing algorithms for RSM to solve MRF with performance guarantees. The key bottleneck preventing us to adapt those algorithms is how to guarantee that a returned solution is robust and satisfies submodular constraints.

Definitions & Complexity

In this part, we present definitions and theories that would be used frequently in our analysis; and analyze complexity of solving MRF. Due to page limit, detailed proofs of lemmas and theorems of this part are provided in Appendix.

\textbf{Definition 2.} Given an instance of MRF, including $V, \{f_i\}_{i \in [m]}, \{T_i\}_{i \in [m]}$, a set $A \subseteq V$ is $(t, \alpha)$-robust iff for all $i \in [m]$, $\min_{X \subseteq A \setminus X} f_i(A \setminus X) \geq (1 - \alpha) T_i$.

Speaking in another way, MRF asks us to find a minimum $(r, 0)$-robust set.

Without loss of generality, in our algorithm, we change $f_i(\cdot) := \min(f_i(\cdot) / T_i, 1)$. It is trivial that $f_i$ is still monotone submodular; and MRF’s objective now is to find $S$ that $\min_{X \subseteq S \subseteq V} f_i(S \setminus X) \geq 1 + \alpha m$.

If there exists a $(r, 0)$-robust set, denote $S^*$ as an optimal solution; and $OPT(U, t)$ as a size of the minimum $(t, 0)$-
robust set that is a subset of $U$ if there is any. So $|S^*| = OPT(V, r)$. We have the following key lemma:

**Lemma 1.** For all $X_1, X_2 \subseteq V$ that $|X_1| = r_1, |X_2| = r_2$ and $r_1 + r_2 \leq r$

$$OPT(V, r) \geq OPT(V \setminus X_1, r - r_1) \geq OPT(V \setminus (X_1 \cup X_2), r - r_1 - r_2) \geq OPT(V, 0)$$

Lemma 1 is very critical and will be used frequently to obtain performance guarantees of our algorithms.

Given a MinRF instance and $\alpha \in [0, 1]$, we aims to devise algorithms that guarantee:

- If there exists $(r, \alpha)$-robust sets in the MinRF instance, the returned solution is $(r, \alpha)$-robust with high probability to $OPT(V, r)$.
- Otherwise, the algorithms notify no $(r, 0)$-robust set exists.

We first study the hardness of devising such an algorithm to solve MinRF. First, we show that: even the sub-task of outputting a $(r, 0)$-robust set if there is any, is already very expensive. That is stated in the following theorem.

**Theorem 1.** There exists no algorithm, taking fewer than exponential number of queries in term of $r$, is able to verify existence of a $(r, 0)$-robust set to MinRF.

Theorem 1 is proven by taking one instance of MinRF, in which the removal of any subset $X \subseteq V$ of a same size shows a similar behavior on the submodular objectives except for only one unique subset $R$ of size $r$. The thresholds $\{T_i\}_i$ are set so that: if there exists a $(r, 0)$-robust set then $V$ is the only $(r, 0)$-robust set and $R$ is the only set that would make $V \setminus R$ violate the constraints. Thus any algorithm, taking fewer than $O((V_i)^1)$ queries is unable to verify whether $V$ is $(r, 0)$-robust. The full description of the MinRF instance is provided in Appendix.

Furthermore, even there exists $(r, 0)$-robust sets, devising approximation algorithms for MinRF is NP-hard. We have the following theorem.

**Theorem 2.** There exists no polynomial algorithm that can approximate MinRF, even with $r = 0$, within a factor of $(1 - \epsilon) \ln m$ given $\epsilon > 0$ unless $P = NP$.

### Algorithms when $r = 0$

We first study MinRF with $r = 0$ since complexity and solution quality of algorithms for MinRF with $r = 0$ play critical roles on the performance of Alg 1 and AlgR. Although MinRF with $r = 0$ has been studied in the literature, these results cannot applied directly. The key barrier is that the initial solution set may not be empty.

In this part, we propose RandGR, a randomized algorithm with bicriteria approximation ratio of $O(\ln \frac{m}{\alpha})$. Also, we re-investigate performance guarantees of Greedy and ThresGR, extending from their performance when $f_i$ receives values in $\mathbb{Z}$.

With RandGR, checking if there exists feasible solutions with $r = 0$ is quite trivial. RandGR simply verifies whether $f_i(V) > 1 - \alpha$ for all $i \in [m]$. If no, the algorithm notifies no feasible set exists and terminates.

If there exists feasible solutions, RandGR works in rounds in order to find a $(0, \alpha)$-robust solution. For each round, a new random process is introduced as follows: the algorithm randomly selects half of functions $f_i$s, each of which is still less than $1 - \alpha$; and greedily chooses an element that maximizes the sum of marginal gains of the selected functions. This random process helps RandGR (1) reduce the number of queries to $f_i$s by half at each round; and (2) establish a recursive relationship of obtained solutions at different rounds, which is critical for RandGR to obtain its performance guarantee with high probability.

RandGR’s pseudocode is presented by Alg. 1. In Alg. 1, $S_t$ represents an obtained solution at round $t$ and $F_t$ is a set of $f_i$s that $f_j(S_t) \geq 1 - \alpha \forall f_j \not\in F_t$. Note that RandGR starts with $S_0$ as an input; and as can be seen later, RandGR is used as a subroutine function in case $r > 0$, in which $S_0$ may not be empty. Therefore, analyzing performance of RandGR with $S_0 \neq \emptyset$ is necessary and challenging.

To obtain RandGR’s performance guarantee, we have the following lemma.

**Lemma 2.** At round $t$:

$$E \left[ \sum_{f_i \in F_{t+1}} (1 - f_i(S_{t+1})) \right] \leq \left( \frac{1}{1 - OPT(V, 0)} \right) \sum_{f_i \in F_t} (1 - f_i(S_t))$$

Lemma 2 establishes a recursive relationship between $F_t$ and $S_t$ at different rounds. This is a key to obtain RandGR’s approximation ratio. Assuming RandGR stops after $L$ rounds, $L = |S_L \setminus S_0|$. By using Markov inequality, we can bound $L$ w.h.p to obtain RandGR’s performance guarantee as Theorem 3. Full proofs of Lemma 2 and Theorem 3 are presented in Appendix.

**Theorem 3.** Given an instance of MinRF with input $V, \{f_i\}_{i \in [m]}, S_0$ such that $\sum_{i \in [m]} f_i(S_0) \geq (1 - \eta)m$, and $r = 0$. If $S$ is an output of RandGR then w.h.p $|S \setminus S_0| \leq OPT(V, 0)O(\ln \frac{m}{\alpha})$ and each $f_i$ is queried at most $O(VOPT(V, 0) \ln \frac{m}{\alpha})$ times.

We now investigate the performance of Greedy and ThresGR. Their performance guarantees are stated by Theorem 2 (Greedy) and 5 (ThresGR). Due to page limit, their detailed description and proofs are presented in Appendix.

---

**Algorithm 1 RandGR**

**Input** $V, S_0, \{f_i\}_{i \in [m]}$

1. if There exists $i \in [m]$ s.t. $f_i(V) < 1 - \alpha$ then
2. **Return** no feasible solution
3. $t = 0; F_0 = \{f_i\}_{i \in [m]}$
4. while $F_t \neq \emptyset$ do
5. $F = \text{randomly select } \lfloor |F_t|/2 \rfloor \text{ constraints from } F_t$
6. $e_t = \text{argmax}_{e_i \in V \setminus S_t} \sum_{f_i \in F} \Delta_e f_i(S_t)$
7. $S_{t+1} = S_t \cup \{e_t\}; F_{t+1} = F_t; t = t + 1$
8. Remove all $f_i \in F_t$ that $f_i(S_t) \geq 1 - \alpha$ out of $F_t$

**Return** $S_t$
Theorem 4. Given an instance of MINRF with input $V; \{f_i\}_{i \in [m]}$, $S_0$ such that $\sum_{i \in [m]} f_i(S_0) \geq (1 - \eta)m$, and $r = 0$. If GREEDY terminates with a $(0, \alpha)$-robust solution $S$, then $|S \setminus S_0| \leq OPT(V, 0)O(\frac{\ln \frac{m}{\alpha}}{\alpha})$ and each $f_i$ is queried at most $O(V OPT(V, 0) \ln \frac{m}{\alpha})$ times.

Theorem 5. Given an instance of MINRF with input $V; \{f_i\}_{i \in [m]}$, $S_0$ such that $\sum_{i \in [m]} f_i(S_0) \geq (1 - \eta)m$, and $r = 0$. If THRESGR terminates with a $(0, \alpha)$-robust solution $S$, then $|S \setminus S_0| \leq OPT(V, 0)O(\frac{1}{1 - \gamma} \ln \frac{m}{\alpha})$ where $\gamma \in (0, 1)$ is the algorithm’s parameter; and each $f_i$ is queried at most $O(\frac{\ln \frac{m}{\alpha}}{\alpha})$ times.

With $S_0 = \emptyset$, RANDGR, GREEDY and THRESGR ($\gamma$ is close to 0) can obtain a ratio of $O(\ln \frac{m}{\alpha})$, which is tight to the inapproximability of MINRF when $r = 0$ (Theorem 2).

Algorithms when $r \geq 1$

In this section, we propose two algorithms to solve MINRF when $r > 0$: ALG1 for a special case of $r = 1$ and ALG2 for general $r$. Both algorithms frequently call an algorithm to MINRF when $r = 0$ as a subroutine, which could be either RANDGR, GREEDY or THRESGR as discussed earlier. In short, we use ALG0 to refer to any of these three.

For simplicity, we ignore the step of notifying if there exists no $(r, \alpha)$ robust set in ALG1 and ALG2’s description since it can trivially inferred from the outputs of ALG0. Without loss of generality, in our analysis, we assume there exists $(r, 0)$-robust sets.

Algorithm when $r = 1$ (ALG1)

In general, ALG1 is an iterative algorithm, which iteratively checks if there exists an element whose removal causes an obtained solution $S$ to violate at least one constraint. If such an element (let’s call it $e$) exists, ALG1 gathers all violated constraints to form a new MINRF instance with $r = 0$, $V \setminus \{e\}$ as an input ground set and $S \setminus \{e\}$ as a new initial set. This is a key of ALG1 because by solving that new MINRF instance using ALG0, ALG1 guarantees the obtained solution is robust against $e$’s removal; and the algorithm can significantly tighten an upper bound on the number of newly-added elements in order to obtain a tight approximation ratio.

ALG1’s pseudocode is presented by Alg. 1. Alg. 1: $S_1$ is a $(0, \alpha)$-robust set, found by using ALG0 with the original MINRF’s input (line 1). $S$, returned by ALG1, is $(1, \alpha)$-robust because:

- For any $e \in S_1$ that violates the condition of while loop (line 2), ALG1 guarantees $f_i(S \setminus \{e\}) \geq 1 - \alpha$ (output of ALG0, line 3).
- For any $e \notin S_1$, as $S_1 \subseteq S \setminus \{e\}$, we have $f_i(S \setminus \{e\}) \geq f_i(S_1) \geq 1 - \alpha$ (output of ALG0, line 4).

Denote $E$ as a set of $e \in S_1$ that violate the condition of while loop (line 2). For each $e \in E$, denote $S^c$ as $S$ right before $e$ is considered by the while loop of line 3. Let $\rho_e = \sum_{i \in E} \Delta_e f_i(S^c \setminus \{e\}) / m$. To obtain ALG1’s performance guarantee, we have the following lemma.

Lemma 3. $\sum_{e \in E} \rho_e \leq 1$

Algorithm 2: Algorithm when $r = 1$ (ALG1)

Input $V; \{f_i\}_{i \in [m]}$
1: $S = S_1 = ALG0(V, \emptyset, \{f_i\}_{i \in [m]})$
2: while $\exists e \in S_1$ that $\sum_{i \in [m]} f_i(S \setminus \{e\}) < 1 - \alpha - \rho_e$
3: $F' = \text{set of all } f_i \text{ that } f_i(S \setminus \{e\}) < 1 - \alpha$
4: $S' = ALG0(V \setminus \{e\}, S \setminus \{e\}, F')$
5: $S = S \cup S'$
Return $S$

Proof. Let’s sort elements in $S_1 = \{u_1, u_2, \ldots\}$ in the order of being added into $S_1$ by ALG1 (line 1). Let $S'_1 = \{u_1, \ldots, u_{e-1}\}$. Due to submodularity, $\sum_i \Delta_e f_i(S'_1) \geq \sum_i \Delta_e f_i(S \setminus \{e\}) = \rho_e m$. Then:

$$\sum_{e \in E} \rho_e m \leq \sum_{e \in E} \sum_{i} \Delta_e f_i(S'_1) \leq \sum_{e \in S_1} \sum_{i} \Delta_e f_i(S'_1) \leq m$$

which means $\sum_{e \in E} \rho_e \leq 1$ and the proof is completed.

We then obtain ALG1’s performance guarantee as stated in Theorem 6.

Theorem 6. Given an instance of MINRF with input $V; \{f_i\}_{i \in [m]}$ and $r = 1$. If $S$ is an output of ALG1 and $S_1$ is a $(0, \alpha)$-robust set outputted by ALG0($V, \emptyset, \{f_i\}_{i \in [m]}$) then $|S| \leq OPT(V, 1)O(|S_1| \ln m + 1/\alpha)$.

Proof. From a ratio of ALG0 and lemma 1, we have $|S_1| \leq O(\frac{\ln \frac{m}{\alpha}}{\alpha})OPT(V, 0) \leq O(\ln \frac{m}{\alpha})OPT(V, 1)$.

For each $e \in E$ and $S^c$ as defined before, we have:

$$\sum_i f_i(S^c \setminus \{e\}) = \sum_i f_i(S^c) - \rho_e m \geq m(1 - \alpha - \rho_e)$$

The last inequality comes from the fact that $S_1 \subseteq S^c$ and $S_1$ is $(0, \alpha)$-robust. Then, with $S' = ALG0(V \setminus \{e\}, S^c \setminus \{e\}, F')$ in line 4, denote $\delta S' = S' \setminus S^c$. From the ratio of ALG0 and lemma 1, we have:

$$|\delta S'| \leq O(\ln \frac{\ln \frac{m}{\alpha}}{\alpha})OPT(V \setminus \{e\}, 0)$$

$$\leq O(\ln \frac{\ln \frac{m}{\alpha}}{\alpha})OPT(V, 1)$$

Therefore, with $S$ is the returned solution, we have:

$$|S| = |S_1| + \sum_{e \in E} |\delta S'|$$

$$\leq O\left(\ln \frac{\ln \frac{m}{\alpha}}{\alpha} + \sum_{e \in E} \ln \frac{\ln \frac{m}{\alpha}}{\alpha}\right)OPT(V, 1)$$

$$= O\left(\frac{\ln \frac{m}{\alpha}}{\alpha} + \ln \prod_{e \in E} \frac{\ln \frac{m}{\alpha}}{\alpha}\right)OPT(V, 1)$$

$$\leq O\left(\ln \frac{m}{\alpha} + \ln \left(\sum_{e \in E} \frac{\ln \frac{m}{\alpha}}{\alpha}\right)\right)OPT(V, 1)$$

$$\leq O\left(\frac{\ln \frac{m}{\alpha}}{\alpha} + \ln \left(\frac{m}{\alpha}\right)^{\frac{1}{\alpha}}\right)OPT(V, 1)$$

$$\leq O(|E| \ln m + \frac{1}{\alpha})OPT(V, 1)$$
which completes the proof. □

Problem 3. Algorithm with general (ALGR)

Input $V, \{f_i\}_{i \in [m]}$, $r$

1: $F_0 = \{f_i\}_{i \in [m]}$, $S_0 = \text{ALGO}(V, \emptyset, F_0)$;
2: for $t = 1 \rightarrow r$ do
3: $F_t = \emptyset$
4: for each set $X \subseteq S_{t-1}$ that $|X| = r, X \not\subseteq S_t$ do
5: for each $i \in [m]$ s.t. $f_i(S_t \setminus X) < 1 - \alpha$ do
6: Define $f_i(X) = f_i(S_t \setminus X)$
7: $F_t = F_t \cup \{f_i, X\}$
8: $S_t = \text{ALGO}(V, S_{t-1}, F_t)$

Return $S_r$.

Algorithm for general $r$ (ALGR)

ALGR works in at most $r$ rounds, in which after $t$ rounds, ALGR guarantees an obtained solution is $(t, \alpha)$-robust. Denote $S_t$ as the obtained solution after $t$ rounds. At round $t$, ALGR introduces a new MIRNF instance after a new set of functions $F_t$. Each function in $F_t$ is defined by a function $f_i$ and a set $X \subseteq S_t$ that $f_i(S_t \setminus X) < 1 - \alpha$ and $|X| = r$. This is a key of ALGR because by solving the new MIRNF instance to obtain $S_t+1$, RANDGR guarantees $S_t+1$ is $(t, \alpha)$-robust. Also the algorithm is able to bound the number of newly-added elements in term of $|S^*|$ by observing that $S^*$ is also a feasible solution to the new MIRNF instance.

ALGR’s pseudocode is presented by Alg. 3. Note that ALGR guarantees $S_t$ is $(r, \alpha)$-robust without a need of scanning all the removals of its subsets of size $r$. We prove that by using contradiction as follows:

Assume $S_t$ is not $(r, \alpha)$-robust, then there exists $X \subseteq V$ and $f_i$ such that $|X| = r$ and $f_i(S_t \setminus X) < 1 - \alpha$. Let $X_0 = X \cap S_t$ and $X_t = X \cap (S_t \setminus S_{t-1})$ for $t = 1 \rightarrow r$.

If there exists an empty $X_t$, let $X_r = \bigcup_{t=0}^{r} X_t$. We have $|X_r| \leq r$ and $X_r \subseteq S_{t-1}$. Due to the output of ALGO in line 3, $f_i(S_t \setminus X') \geq 1 - \alpha$. But $S_t \setminus X' \subseteq S_t \setminus X$, so $f_i(S_t \setminus X') \geq 1 - \alpha$, which contradicts to our assumption.

Thus, no $X_t$ should be empty, which is impossible since $|X| = r \geq \bigcup_{t=0}^r X_t$, and $X_0, \ldots, X_r$ are disjoint. Therefore, $S_t$ should be $(r, \alpha)$-robust.

To obtain ALGR’s performance guarantee, we have the following lemma.

Lemma 4. $|S_t \setminus S_{t-1}| \leq O\left(\frac{\ln |F_t|}{\alpha}\right)OPT(V, r)$ for all $t \leq r$

Proof. Considering a new constraint $f_{i, X}$ created in line 6, it is trivial that the function $f_{i, X}$ is monotone submodular.

Also, as $S^*$ is $(r, 0)$-robust, $f_{i, X}(S^*) = f_{i, X}(S^* \setminus X) \geq 1$. That means $S^*$ is feasible for the MIRNF instance in line 6 with $F_t$ as a set of constraint and $r = 0$. The lemma follows from the ratio of ALGO. □

Lemma 4 is critical to obtain ALGR’s ratio, stated in the following theorem.

Theorem 7. Given an instance of MIRNF with input $V, \{f_i\}_{i \in [m]}$, $r$, if $S$ is an output of ALGR, then:

$$|S| \leq OPT(V, r)O(r \ln m/\alpha + r^2 \ln n)$$

Proof. Using lemma 1 and ALGO’s ratio, we have: $|S_0| \leq OPT(V, 0)O(ln m/\alpha) \leq OPT(V, r)O(ln m/\alpha)$. Therefore, from lemma 4 we have:

$$|S_r| = |S_0| + \sum_{t=1}^{r} |S_t \setminus S_{t-1}|$$

$$\leq O(ln m/\alpha + \sum_{t=1}^{r} \ln |F_t|/\alpha)OPT(V, r)$$

Furthermore, $\sum_{t=1}^{r} |F_t| \leq m (\ln S_{t-1})$ because: (1) No subset $X \subseteq S_{t-1}$ of size $r$ is considered more than one round (line 2) as $f_i(S_t \setminus X) \leq 1 - \alpha$ then $f_i(S_t+1 \setminus X) \geq 1 - \alpha$; and (2) each subset $X$ added to $F_t$ at most $m$ new constraints.

Therefore, by using AM-GM inequality, we have:

$$\prod_{t=1}^{r} |F_t| \leq \left(\frac{\sum_{t=1}^{r} |F_t|}{r}\right)^r \leq \left(\frac{\ln S_{t-1}}{r}\right)^r$$

Thus, $|S_r| \leq O(r \ln m/\alpha + r^2 \ln n)OPT(V, r)$. □

Query Complexity. The bottleneck of ALGR is from the task of finding all subsets $X$ in line 4. As there is $(\ln S_{t-1})$ subsets $X$, ALGR takes $(\ln S_{t-1})$ queries for each $f_i$ to only find $X$; and in the worst case, each $f_i$ will generate $(\ln S_{t-1})$ functions $f_{i, X}$ (line 6). Then, if ALGR uses RANDGR or GREEDY as ALGO, the worst case, each $f_i$ is queried at most $O(n(r \ln n/\alpha + r^2 \ln n))OPT(V, r)(\ln S_{t-1})$ times. If THRESGR is used, at round $t$, each $f_i$ is queried at most $O(n|F_i|/\gamma)\ln |F_i|/\alpha$. Overall, ALGR using THRESGR will query each $f_i$ at most $O(\frac{n}{\gamma}(\ln S_{t-1})(r \ln n/\alpha + r^2 \ln n))$ times. ALGR is polynomial with fixed $r$ and favourable if $OPT(V, r) \ll n$.

Experimental Evaluation

In this section, we compare our algorithms with existing methods and intuitive heuristics on two applications of MIRNF, Information Propagation for Multiple Groups (IP) and Movie Recommendation for Multiple Users (MR). The source code is available at https://github.com/lannn2410/minrf.

Information Propagation for Multiple Groups (IP) In this problem, a social network is modeled as a directed graph $G = (V, E)$ where $V$ is a set of social users. Each edge $(u, v)$ is associated with a weight $w_{u, v}$, representing the strength of influence from user $u$ to $v$.

To model the information propagation process, we use Linear Threshold (LT) Model (Kempe, Kleinberg, and Tardos...
In general, the process is as follows: Each user in a set $S$ becomes active. Next, information cascades in discrete steps and in each step, a user $v$ becomes active if $\sum_{u \in S} w_{u,v} \geq \theta_v$. The process stops when no more user can become active.

Given a collection $U$ of subsets of $V$, i.e. $U = \{C_1, ..., C_m\}$ where $C_i \subseteq V$. Each $C_i$ represents a group that we need to influence. Denote $I_i(S)$ as the expected number of active users in $C_i$ by a seed set $S$. Given a number $T \in [0, 1]$, IP aims to find the smallest $S$ such that for all $C_i \in U$, $\min_{|S| \leq T} I_i(S \setminus X) \geq T |C_i|$. We use Facebook dataset from SNAP database (Leskovec and Krevl 2014), an undirected graph with 4,039 nodes and 88,234 edges. Since it is undirected, we treat each edge as two directed edges. The weight $w_{u,v}$ is set to be $1/d_v$, where $d_v$ is in-degree of $v$. $U$ is a collection of groups to which users are classified based on their gender or race. Due to lack of data information, we use a user's race and gender are randomly assigned. $I_i(S)$ is estimated over 100 graph samples.

### Movie Recommendation for Multiple Users (MR)

In this problem, given a set $M$ of movies, a set $U$ of users, each user $u$ has a list $L_u$ of his/her favourite movies. Given $S \subseteq U$, a utility score of $u$ to $S$ is defined as $f_u(S) = \sum_{i,j \in L_u, s_{i,j} \in S} \text{sim}(i, j)$, where $s_{i,j} \in [0, 1]$ which measures the similarity between movie $i$ and $j$. Given a number $T$, the objective is to find the smallest set of movies to recommend to all users in a way such that every user’s utility level is at least $T$ under any $r$ “inaccurate-data” movies removal, i.e. $\min_{|S| \leq T} f_u(S \setminus X) \geq T$ for all $u \in U$.

We use Movie Lens dataset from GroupLens (2015) database, which includes information of 10,381 movies; and their 200,000,264 ratings (ranging in $[0, 5]$) from 138,493 users. We randomly pick 4 users for a set $U$, $L_u$ contains movies that $u$ rated at least 4. Each movie $i$ is associated by a 1,129-dimension vector $v_i$, where each entry (ranging in $[0, 1]$) represents the relevant score between the movie and a keyword. The relevant scores are available in the dataset. We use cosine similarity score $\text{sim}(v_i, v_j)$ to present $s_{i,j}$. For each user $u$, $f_u(S)$ is normalized to be in range $[0, 1]$.

**Compared Algorithms**

With $r = 0$, we compare RANDGR, GREEDY and THRESGR ($\gamma = 0.2$) with SEP algorithm: which considers each constraint separately, runs greedy to find a set $S_i$ that $f_i(S_i) \geq 1 - \alpha$ and return $\cup_{i \in [m]} S_i$. SEP obtains a ratio of $O(m \ln \frac{1}{\alpha})$.

With $r > 0$, we compare ALGR’s performance in combination with each ALG0, including RANDGR, GREEDY, THRESGR, SEP. Each combination of ALGR to a ALG0 algorithm is denoted, in short, ALGR-name of the ALG0 algorithm, e.g. ALGR-RANDGR.

We also compare ALGR with DISJOINT, a heuristic we propose to evaluate. DISJOINT finds $r + 1$ disjoint sets $S_1, ..., S_{r+1}$ such that $f_i(S_j) \geq 1 - \alpha$ for all $i \in [m]$ and $j \in [r+1]$; and returns $S = \cup_{j \in [r+1]} S_j$. If DISJOINT successfully finds all $\{S_j\}_{j \in [r+1]}$, then $S$ is feasible to MINRF without the need for checking all subsets of size $r$. This is because for any set $X$ of size $r$, there should exist $S_j \cap X = \emptyset$. Thus, $S_j \subseteq S \setminus X$, which means $f_i(S \setminus X) \geq f_i(S_j) \geq 1 - \alpha$ for all $i \in [m]$. However, there are two problems with DISJOINT: (1) If DISJOINT cannot find all $\{S_j\}_{j \in [r+1]}$, the algorithm does not guarantee there exists no feasible solution to MINRF; and (2) DISJOINT does not obtain any approximation ratio.

For $r = 1$, we also evaluate ALG1 performance in combination with each ALG0 algorithm, including RANDGR, GREEDY, THRESGR.

**Other.** We set $\alpha = 0.1$. Results are averaged over 10 repetitions.

**Experimental Results**

Fig. 1 shows the performances of different ALG0 algorithms in comparison with SEP. We can see that ALG0 algorithms totally outperformed SEP in solution quality by a huge margin. RANDGR returned solutions approximately close to GREEDY, which is the best one in term of solution quality. However, in term of query efficiency, RANDGR took much fewer queries than GREEDY and; and was the fastest algorithm in the IP problem. This confirms the efficiency of RANDGR by introducing randomness and discarding satisfied constraints after each iteration.

Fig. 2 shows algorithms’ performance on the IP and MR problems when $r = 1$. The two proposed heuristics, ALG1-SEP and DISJOINT, showed the worst performance in solu-
That can be explained by the fact that whenever \( A_r \) subsets of size \( r \) significantly the number of queries of \( A_r \) increased by an exponent rate in term of \( r \) by other algorithms by a huge margin in solution quality; observed that \( A \) larger set of submodular functions than \( A \) one constraint, to form a new M gathers all elements, each element’s removal violates at least other elements’ removal as well. On the other hand, \( A \) is robust to \( \text{IP} \) and \( A \) more queries than any other algorithms. Also, by finding disjoint subsets, \( D \) all constraints, is not a necessary condition to guarantee robustness. Also, by finding disjoint subsets, \( D \)’s undesirable performance came from \( \text{IP} \). On the other hand, \( D \) was the most efficient \( A \) algorithm when \( r = 0 \) (standalone) or \( r = 1 \) (combining with \( A \) or \( A \)). \( A\)’s performances were undesirable with large \( r \). This is because \( \text{REEDY} \) tends to return larger solution than \( \text{RANDGR} \) and \( \text{GREEDY} \). Therefore, \( A\)\text{REEDY} requires more queries to scan over all subset of size \( r \) of \( S_r \) than \( A\)\text{RANDGR} and \( A\)\text{GREEDY}.

**Conclusion**

Motivated by real-world applications, in this work, we studied a problem of minimum robust set subject to multiple submodular constraints, namely \( \text{MINRF} \). We investigate \( \text{MINRF}’s \) hardness using complexity theories; and proposed multiple approximation algorithms to solve \( \text{MINRF} \). Our algorithms are proven to return tight performance guarantees to \( \text{MINRF}’s \) inapproximability and required query complexity. Finally, we empirically demonstrated that our algorithms outperform several intuitive methods in terms of the solution quality and number of queries.
Acknowledgements
This work was supported in part by the National Science Foundation (NSF) grants IIS-1908594, IIS-1939725, and the University of Florida Informatics Institute Fellowship Program. We would like to thank the anonymous reviewers for their helpful feedback.

References
Anari, N.; Haghtalab, N.; Naor, S.; Pokutta, S.; Singh, M.; and Torrico, A. 2019. Structured Robust Submodular Maximization: Offline and Online Algorithms. In The 22nd International Conference on Artificial Intelligence and Statistics, 3128–3137.

Bogunovic, I.; Mitrovic, S.; Scarlett, J.; and Cevher, V. 2017. Robust submodular maximization: A non-uniform partitioning approach. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, 508–516. JMLR.org.

Dinh, T. N.; and Thai, M. T. 2014. Network under joint node and link attacks: Vulnerability assessment methods and analysis. IEEE/ACM Transactions on Networking 23(3): 1001–1011.

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

Goyal, A.; Bonchi, F.; Lakshmanan, L. V.; and Venkatasubramanian, S. 2013. On minimizing budget and time in influence propagation over social networks. Social network analysis and mining 3(2): 179–192.

GroupLens. 2015. MovieLens 20M Dataset. https://grouplens.org/datasets/movielens/20m/.

Guillory, A.; and Bilmes, J. A. 2011. Simultaneous learning and covering with adversarial noise.

Iyer, R. K.; and Bilmes, J. A. 2013. Submodular optimization with submodular cover and submodular knapsack constraints. In Advances in Neural Information Processing Systems, 2436–2444.

Kempe, D.; Kleinberg, J.; and Tardos, É. 2003. Maximizing the spread of influence through a social network. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 137–146. ACM.

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

Kuhnle, A.; Pan, T.; Alim, M. A.; and Thai, M. T. 2017. Scalable bicriteria algorithms for the threshold activation problem in online social networks. In IEEE INFOCOM 2017-IEEE Conference on Computer Communications, 1–9. IEEE.

Leskovec, J.; and Krevl, A. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

Mirzasoleiman, B.; Karbasi, A.; Badanidiyuru, A.; and Krause, A. 2015. Distributed submodular cover: Succinctly summarizing massive data. In Advances in Neural Information Processing Systems, 2881–2889.

Mirzasoleiman, B.; Zadimoghaddam, M.; and Karbasi, A. 2016. Fast distributed submodular cover: Public-private data summarization. In Advances in Neural Information Processing Systems, 3594–3602.

Mitrovic, S.; Bogunovic, I.; Norouzi-Fard, A.; Tarnawski, J. M.; and Cevher, V. 2017. Streaming robust submodular maximization: A partitioned thresholding approach. In Advances in Neural Information Processing Systems, 4557–4566.

Nguyen, H. T.; Thai, M. T.; and Dinh, T. N. 2016. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In Proceedings of the 2016 International Conference on Management of Data, 695–710. ACM.

Nguyen, L.; and Thai, M. T. 2020. Streaming k-Submodular Maximization under Noise subject to Size Constraint. In International Conference on Machine Learning, 7338–7347. PMLR.

Nguyen, L. N.; and Thai, M. T. 2019. Network Resilience Assessment via QoS Degradation Metrics: An Algorithmic Approach. Proceedings of the ACM on Measurement and Analysis of Computing Systems 3(1): 1–32.

Nguyen, L. N.; Zhou, K.; and Thai, M. T. 2019. Influence maximization at community level: A new challenge with non-submodularity. In 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 327–337. IEEE.

Norouzi-Fard, A.; Bazzi, A.; Bogunovic, I.; El Halabi, M.; Hsieh, Y.-P.; and Cevher, V. 2016. An efficient streaming algorithm for the submodular cover problem. In Advances in Neural Information Processing Systems, 4493–4501.

Ohsaka, N.; and Yoshida, Y. 2015. Monotone k-submodular function maximization with size constraints. In Advances in Neural Information Processing Systems, 694–702.

Orlin, J. B.; Schulz, A. S.; and Udwani, R. 2018. Robust monotone submodular function maximization. Mathematical Programming 172(1-2): 505–537.

Qian, C.; Shi, J.-C.; Yu, Y.; Tang, K.; and Zhou, Z.-H. 2017. Subset selection under noise. In Advances in neural information processing systems, 3560–3570.

Staib, M.; Wilder, B.; and Jegelka, S. 2019. Distributionally Robust Submodular Maximization. In The 22nd International Conference on Artificial Intelligence and Statistics, 506–516.

Zhang, H.; Mishra, S.; Thai, M. T.; Wu, J.; and Wang, Y. 2014. Recent advances in information diffusion and influence maximization in complex social networks. Opportunistic Mobile Social Networks 37(1.1): 37.
Hardness and complexity requirement of M\textsc{inrf}

Proof of Lemma 1

Proof. We first focus on the second inequality since the first inequality can be trivially inferred by the second one. Let \( l = |OPT(V \setminus X_1,r-r_1) \cap X_2| \leq 2 \). Since \( OPT(V \setminus X_1,r-r_1) \) is robust to a removal of \( r-r_1 \) elements, \( OPT(V \setminus X_1,r-r_1) \setminus X_2 \) is robust to a removal of \( r-r_1-r_2 \) elements and contains no elements from \( X_2 \). Therefore, \( OPT(V \setminus X_1,r-r_1) \setminus X_2 \) is a feasible solution given input \( V \setminus X_1 \setminus X_2 = V \setminus (X_1 \cup X_2) \) and \( r-r_1-r_2 \). So \( OPT(V \setminus X_1,r-r_1) \geq OPT(V \setminus (X_1 \cup X_2),r-r_1-r_2) \).

The last inequality comes from observation that \( f_i(OPT(V \setminus (X_1 \cup X_2),r-r_1-r_2)) \geq T_i \) for all \( i \in [m] \), thus it is feasible to input \( V \) and \( r = 0 \).

\( \square \)

Proof of Theorem 2

Proof. The main idea of this proof is to find a submodular function \( f \) and a threshold \( T \) such that the removal of any subset \( X \subseteq S \) of the same size on \( V \) shows a similar behavior on \( f \) except for only one unique subset \( R \) of size \( r \). \( R \) is the only set that \( f(V \setminus R) < T \). Thus any algorithm, taking fewer than \( O((|V|/r)) \) queries is unable (or only with tiny probability) to verify whether \( V \) is \((r,0)\)-robust or not.

The instance is as follows: Given the ground set \( V \) and \( r \), we randomly choose a subset \( R \subseteq V \) that \(|R| = r \). The submodular function \( f \) is defined as follows:

- For any \( Z \subseteq V \) that \(|Z| < |V| - r \), \( f(Z) = 3|Z| \).
- \( f(V \setminus R) = 3|V \setminus R| - 1 \)
- For any \( Z \subseteq V \) that \(|Z| \geq |V| - r \) and \( Z \neq V \setminus R \), \( f(Z) = 3(|V| - r) + (|Z| - |V| + r) \)

It is trivial that \( f \) is monotone. We now prove that \( f \) is submodular. Given \( A \subset B \subseteq V \) and \( e \notin B \), we have:

- If \(|A| < |V| - r \) then:
  - If \( A \cup \{e\} = V \setminus R \), \( \Delta_e f(A) = 2 \) while \( \Delta_e f(B) = 1 \)
  - Otherwise \( \Delta_e f(A) = 3 \) while
    * \( \Delta_e f(B) = 3 \) if \( |B \cup \{e\}| \leq |V| - r \) and \( B \cup \{e\} \neq V \setminus R \)
    * \( \Delta_e f(B) = 2 \) if \( B \cup \{e\} = V \setminus R \) or \( B = V \setminus R \)
    * Otherwise, \( \Delta_e f(B) = 1 \)

- If \(|A| \geq |V| - r \), then:
  - If \( A = V \setminus R \), \( \Delta_e f(A) = 2 \) while \( \Delta_e f(B) = 1 \)
  - Otherwise \( \Delta_e f(A) = \Delta_e f(B) = 1 \)

So, in any cases, \( \Delta_e f(A) \geq \Delta_e f(B) \). Thus, \( f \) is submodular.

Let \( T = 3(|V| - r) \). Then \( R \) is an only set of size \( r \) that satisfies \( f(V \setminus R) < T \). Thus, any algorithm making sub exponentially many queries will be unable (except with tiny probability) to find \( R \).

\( \square \)

Proof of Theorem 3

Proof. We reduce SET COVER to M\textsc{inrf} with \( r = 0 \).

The SET COVER problem is: Given a finite set \( V' \) and a collection \( C \) of subset \( S_1,...,S_l \) \((S_i \subseteq V')\), find \( S \subseteq C \) of minimum size such that \( \cup_{S_i \in S} S_i = V \). SET COVER can be formulated by the following Integer Programming.

\[ \text{Minimize } \sum_{i=1}^{l} x_i \quad \text{s.t. } \sum_{i \in S} x_i \geq 1 \quad \forall e \in V'; \quad x_i \in \{0,1\} \quad \forall i = 1,...,l \quad (1) \]

To reduce it to an instance of M\textsc{inrf}, we define \( V = C \). For each \( e \in V' \), define \( f_e(S) \) as number of sets in \( S \) that contains \( e \). \( f_e \) is not only submodular, but modular. \( T_e = 1 \) for all \( e \in V' \). Then solving the above Integer Programming is equivalent to finding minimum \( S \) that \( f_e(S) \geq T_e \) for all \( e \in V' \). In this M\textsc{inrf} instance, \( m = |V'| \).

If there exists a \((1 - \epsilon) \ln m \) approximation algorithm \( A \) for M\textsc{inrf}, which means we can use \( A \) to approximate SET COVER within \((1 - \epsilon) \ln |V'| \) ratio. That contradicts with Dinur and Steurer (2014) that SET COVER is inapproximable within ratio \((1 - \epsilon) \ln |V'| \) unless \( P = NP \).
Omitted proofs of **RANDGR**

**Proof of Lemma 2**

*Proof.* Considering at round $t$ and $F$ is a set of randomly selected constraints, we have:

\[
\sum_{f_i \in F} (1 - f_i(S_t)) \leq \sum_{f_i \in F} (f_i(S^* \cup S_t) - f_i(S_t)) \leq \sum_{f_i \in F} \sum_{e \in S^* \setminus S_t} \Delta_e f_i(S_t)
\]

\[
\leq \sum_{e \in S^* \setminus S_t} \sum_{f_i \in F} \Delta_e f_i(S_t) \leq \text{OPT}(V, 0) \sum_{f_i \in F} \Delta_e f_i(S_t)
\]

Let $\mathcal{H}$ is a set of all combination of size $|\mathcal{F}_t|/2$ of $\mathcal{F}_t$. For each $F \in \mathcal{H}$, denote $e_{t,F}$ as $e_i$ if $F$ is selected. Then,

\[
E_{F \sim \mathcal{H}} \left[ \sum_{f_i \in F} (1 - f_i(S_t)) \right] = \frac{1}{|\mathcal{H}|} \sum_{F \in \mathcal{H}} \sum_{f_i \in F} (1 - f_i(S_t))
\]

\[
= \frac{1}{|\mathcal{H}|} \sum_{f_i \in \mathcal{F}_t} \sum_{F \in \mathcal{H}, f_i \in F} (1 - f_i(S_t))
\]

\[
= \frac{1}{2} \sum_{f_i \in \mathcal{F}_t} (1 - f_i(S_t))
\]

On the other hand,

\[
E_{F \sim \mathcal{H}} \left[ \text{OPT}(V, 0) \sum_{f_i \in F} \Delta_e f_i(S_t) \right] = \frac{1}{|\mathcal{H}|} \text{OPT}(V, 0) \sum_{f_i \in F} \Delta_e f_i(S_t)
\]

\[
= \text{OPT}(V, 0) \sum_{f_i \in \mathcal{F}_t} \frac{1}{|\mathcal{H}|} \sum_{F \in \mathcal{H}, f_i \in F} \Delta_e f_i(S_t)
\]

\[
\leq \text{OPT}(V, 0) \sum_{f_i \in \mathcal{F}_t} \frac{1}{|\mathcal{H}|} \sum_{F \in \mathcal{H}} \Delta_e f_i(S_t)
\]

\[
= \text{OPT}(V, 0) \sum_{f_i \in \mathcal{F}_t} \text{E}_t \left[ \Delta_e f_i(S_t) \right]
\]

Combining (3), (6), (10) and the fact that $\mathcal{F}_{t+1} \subseteq \mathcal{F}_t$ and $f_i(S) \leq 1$, we have:

\[
\text{E}_t \left[ \sum_{f_i \in \mathcal{F}_{t+1}} (1 - f_i(S_{t+1})) \right] \leq \sum_{f_i \in \mathcal{F}_t} (1 - \text{E}_t \left[ f_i(S_{t+1}) \right]) \leq \left(1 - \frac{1}{2 \text{OPT}(V, 0)}\right) \sum_{f_i \in \mathcal{F}_t} (1 - f_i(S_t))
\]

which completes the proof. \qed

**Proof of Theorem 3**

*Proof.* From Lemma 2 after adding $L$ elements, RANDGR guarantees:

\[
\text{E} \left[ \sum_{f_i \in \mathcal{F}_L} (1 - f_i(S_L)) \right] \leq \left(1 - \frac{1}{2 \text{OPT}(V, 0)}\right)^L \sum_{f_i \in \mathcal{F}_0} (1 - f_i(S_0)) \leq e^{-\frac{\text{OPT}(V, 0)\alpha}{m}}
\]

Let’s consider the probability the algorithm cannot terminate after adding $L$ elements. That probability is equal to the probability that there exists $f_i \in \mathcal{F}_L$ that $f_i(S_L) < 1 - \alpha$. We have:

\[
\text{Pr}[\exists f_i \in \mathcal{F}_L \text{ that } f_i(S_L) < 1 - \alpha] \leq \text{Pr} \left[ \sum_{f_i \in \mathcal{F}_L} (1 - f_i(S_L)) > \alpha \right]
\]

\[
\leq e^{-\frac{\alpha}{m}} \frac{\text{E} \left[ \sum_{f_i \in \mathcal{F}_L} (1 - f_i(S_L)) \right]}{\alpha}
\]

\[
\leq e^{-\frac{\text{OPT}(V, 0)\alpha}{m}}
\]

where the inequality (*) is from Markov inequality.

Therefore, with high probability $1 - o(1)$, the algorithm terminates after adding $L = \text{OPT}(V, 0) \Theta(\ln \frac{mn}{\alpha})$ elements. Which also means: With $S_0 = \emptyset$, RANDGR obtains ratio of $O(\ln \frac{m}{\alpha})$ w.h.p and each $f_i$ is queries by at most $O(|\mathcal{V}| \text{OPT}(V, 0) \ln \frac{m}{\alpha})$. \qed
GREEDY and THRESGr

In general, GREEDY and THRESGr contain the following steps:

1. Set $F(\cdot) = \sum_{i \in [m]} f_i(\cdot)$; $t = 0$
2. While there exists $f_i$ that $f_i(S_t) < 1 - \alpha$
   (a) Find $e_t \in V \setminus S_t$ that $\Delta_{e_t} F(S_t) \geq \delta \times \max_{e \in V \setminus S_t} \Delta_e F(S_t)$
   (b) $S_{t+1} = S_t \cup \{e_t\}$; $t = t + 1$
3. Return $S_t$

It is trivial that $F(\cdot)$ is monotone submodular. The two algorithms are basically different on the value of $\delta$ on step 2(a). To obtain their ratios, we observe that: at round $t$

$$m - F(S_t) \leq F(S^* \cup S_t) - F(S_t) \leq \sum_{e \in S^* \setminus S_t} \Delta_e F(S_t) \leq \frac{1}{\delta} OPT(V, 0) \Delta_{e_t} F(S_t)$$

(11)

Then, their ratio is presented by Theorem 8.

**Theorem 8.** Given $S_0$ that $F(S_0) \geq (1 - \eta) m$; if $S$ is the returned solution, then:

$$|S \setminus S_0| \leq OPT(V, 0) \frac{1}{\delta} \ln \frac{mn}{\alpha} + 1$$

**Proof.** Considering after adding $e_t$, by a simple math transformation from Equ. [11] we have:

$$m - F(S_{t+1}) \leq \left(1 - \frac{\delta}{OPT(V, 0)}\right) (m - F(S_t))$$

Assume the algorithm terminates at $t = L$. Then at $t = L - 1$, there should exist a constraint $f_i$ that $f_i(S_{L-1}) < 1 - \alpha$, which means $F(S_{L-1}) < m - \alpha$. Furthermore:

$$m - F(S_{L-1}) \leq \left(1 - \frac{\delta}{OPT(V, 0)}\right) (m - F(S_{L-2})) \leq \cdots$$

$$\leq \left(1 - \frac{\delta}{OPT(V, 0)}\right)^{L-1} (m - F(S_0))$$

$$\leq e^{-\frac{L(1-\gamma)}{\delta} \ln \frac{mn}{\alpha}}$$

Thus, $L \leq OPT(V, 0) \frac{1}{\delta} \ln \frac{mn}{\alpha} + 1$. With $S_0 = \emptyset$, the algorithm obtains the ratio of $\frac{1}{\delta} \ln \frac{mn}{\alpha} + 1$. \qed

We now go over each algorithm’s value of $\delta$ and their query complexity.

With GREEDY, follow the framework, at step 2(a) GREEDY simply chooses $e_t = \arg \max_{e \in S_t} \sum_{f \in F_e} \Delta_e f_i(S_t)$. Then the $\delta$’s value of GREEDY is 1. From Theorem 8 with $S_0 = \emptyset$, GREEDY obtains ratio of $O(\ln \frac{m}{\alpha})$. Furthermore, the algorithm scans over $V$ by at most $O(OPT(V, 0) \ln \frac{m}{\alpha})$ times. Then, each $f_i$ is queried at most $O(|V|OPT \ln \frac{m}{\alpha})$ times.

THRESGr setups a threshold $\pi$ and adds $e \in V$ to $S$ if $\sum_{i \in [m]} \Delta_{e_i} f_i(S) \geq \pi$. If no more element can be added, the algorithm reduces $\pi$ by a factor of $1 - \gamma$ and scans over $V$ again. The algorithm stops when $S$ satisfies $f_i(S) \geq 1 - \alpha$ for all $i \in [m]$. The pseudocode of THRESGr is presented by Alg. 4.

THRESGr always guarantees to terminate since $V$ is feasible and as long as there exists $e$ that $\sum_{i \in [m]} \Delta_{e_i} f_i(S) > 0$, $\pi$ would decrease until $e$ can be added to $S$.

In THRESGr, $\delta = 1 - \gamma$. To show $\Delta_{e_t} F(S_t) \geq (1 - \gamma) \max_{e \in V \setminus S_t} \Delta_e F(S_t)$ for each $t$, considering at the moment $e_t$ is added into $S_t$, assume $\pi$’s value is $\pi_t$, then there exists no element $e \in V \setminus S_t$ that $\Delta_e F(S_t) \geq \frac{\pi_t}{1 - \gamma}$. If there exists such element, then $e$ should be added to $S_t$ when $\pi \geq \frac{\pi_t}{1 - \gamma}$. The inequality follows since $\Delta_{e_t} F(S_t) \geq \pi_t$.

In term of query complexity, we need to bound on how many times the algorithm has to scan over $V$. We have the following observation:

**Lemma 5.** If $V$ is a feasible set, given a non-feasible set $X$, there exists $e \in V$ that $\Delta_e F(X) \geq \frac{\alpha}{n}$

**Proof.** We use contradiction: assume there exists no such $e$. Then:

$$F(V) \leq F(X) + \sum_{e \in V \setminus X} \Delta_e F(X) < m - \alpha + \frac{\alpha}{n} = m$$

which contradicts to the assumption that $V$ is feasible. \qed

Therefore, THRESGr should terminates when $\pi \geq \frac{\alpha(1 - \gamma)}{n}$. While $\pi \leq m$, the number of times the algorithm has to scan through $V$ is $\frac{1}{\gamma} \ln \frac{mn}{\alpha}$. So each $f_i$ is queried at most $O\left(\frac{\alpha}{\gamma} \ln \frac{mn}{\alpha}\right)$
**Algorithm 4 THRESGr**

**Input** $V, \{f_i\}_{i \in [m]}, S^0, \gamma$

**Output** $S$ that $f_i(S) \geq 1 - \alpha \forall i \in [m]$

1. $t = 0$; $\pi = \max_{e \in V} \Delta_e F(S_0)$
2. $O = \text{ordered set of elements of } V$
3. $e = \text{first elements of } O$;
4. **while** There exists $f_i$ that $f_i(S_t) < 1 - \alpha$ **do**
5.  **if** $\Delta_e F(S_t) \geq \pi$ **then**
6.  $e_t = e$
7.  $S_{t+1} = S_t \cup \{e_t\}$
8.  $t = t + 1$
9.  **if** $e$ is last element in $O$ **then**
10.  $\pi = (1 - \gamma)\pi$
11.  **else**
12.  $e = \text{first element in } O$
13. **end if**
14. **end if**
15. **end while**
16. **return** $S_t$

**Tight example of ALG1**

We consider a special example of MINRF, called Robust Set Cover, defined as follows: Given a ground set $U$ and a family $S$ of subsets of $U$, find a robust set cover $C \subseteq S$ of minimum size such that for all set $A \in S$, $\cup S \in C \setminus A = U$.

Considering the following instance of Robust Set Cover: The ground set $U$ containing $n = 2^k$ elements $\{e_1, \ldots, e_n\}$ and the collection $S$ contains:

- $S_a = S_{a'} = \{e_1, e_3, \ldots, e_{n-1}\}$
- $S_b = S_{b'} = \{e_2, e_4, \ldots, e_n\}$
- $S_i$ stores next $\frac{n}{2^i}$ elements to $S_{i-1}$. For example, $S_1$ contains $\{e_1, \ldots, e_{n/2}\}$, $S_2$ contains $\{e_{n/2+1}, \ldots, e_{3n/4}\}$ and so on.
- $\{S_{j,i}\}$; are subsets of $S_j$ and $S_{j,i}$ store next $\frac{|S_j|}{2^i}$ elements to $S_{j,i-1}$.

Figure 4 shows an example of this special instance with $n = 16$.

![Figure 4](image_url)

**Figure 4:** Example of a tight MINRF instance with $n = 16$

With this instance, the optimal solution is $C^{opt} = \{S_a, S_{a'}, S_b, S_{b'}\}$. 
We make it an instance of MINRF by defining an input set of submodular functions \( \{ f_e \}_{e \in \mathcal{U}} \), where \( f_e(\mathcal{C}) \) is the number of sets in \( \mathcal{C} \) containing \( e \). The threshold \( T_e = 1 \) for all \( e \in \mathcal{U} \). With this MINRF instance, to find a feasible set cover, we simply set \( \alpha \) large and close to 1.

So, if applying ALG1 to this MINRF instance, \( S_1 \) returned from ALG0 in line 1 of Alg. 2 may contain all \( S_i \) sets. It is trivial that \( |S_1| \leq O(\ln n)|S^\text{opt}| \). Also, any removal of a set \( S_i \) from \( S_1 \) makes it violate at least one constraint since \( S_i \)s are disjoint.

Then, again, applying ALG0 to \( \mathcal{S} \setminus S_1 \) for each \( S_i \) may result to all \( S_{i,j} \) set are added. Then the final solution’s size would be \( O(\sum_{S_i \in S_1} \ln |S_i|) \leq O(|S_1| \ln n) \), which is tight to our analysis.