Non-monotone Submodular Maximization under Matroid and Knapsack Constraints

Jon Lee
IBM T. J. Watson Research
Yorktown Heights, NY, USA
jonlee@us.ibm.com

Vahab S. Mirrokni
Google Research
New York, NY, USA
mirrokni@gmail.com

Viswanath Nagarajan*
Tepper School of Business
CMU, Pittsburgh, PA, USA
viswa@cmu.edu

Maxim Sviridenko
IBM T. J. Watson Research
Yorktown Heights, NY, USA
sviri@us.ibm.com

ABSTRACT

Submodular function maximization is a central problem in combinatorial optimization, generalizing many important problems including Max Cut in directed/undirected graphs and in hypergraphs, certain constraint satisfaction problems, maximum entropy sampling, and maximum facility location problems. Unlike submodular minimization, submodular maximization is NP-hard. In this paper, we give the first constant-factor approximation algorithm for maximizing any non-negative submodular function subject to multiple matroid or knapsack constraints. We emphasize that our results are for *non-monotone* submodular functions. In particular, for any constant k, we present a $\left(\frac{1}{k+2+\frac{1}{k}+\epsilon}\right)$-approximation for the submodular maximization problem under k matroid constraints, and a $\left(\frac{1}{5}-\epsilon\right)$-approximation algorithm for this problem subject to k knapsack constraints ($\epsilon > 0$ is any constant). We improve the approximation guarantee of our algorithm to $\frac{1}{k+1+\frac{1}{k-1}+\epsilon}$ for $k \geq 2$ partition matroid constraints. This idea also gives a $\left(\frac{1}{k+\epsilon}\right)$-approximation for maximizing a *monotone* submodular function subject to $k \geq 2$ partition matroids, which improves over the previously best known guarantee of $\frac{1}{k+1}$.

Categories and Subject Descriptors

F.2.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Algorithms

*Supported by an IBM graduate fellowship and NSF award CCF-0728841. Work done while visiting IBM T.J. Watson Research Center.

1. INTRODUCTION

In this paper, we consider the problem of maximizing a *nonnegative submodular function f, defined on a ground set V, subject to matroid constraints or knapsack constraints.* A function $f : 2^V \to \mathbb{R}$ is *submodular* if for all $S, T \subseteq V$, $f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$. Throughout, we assume that our submodular function f is given by a *value oracle*; i.e., for a given set $S \subseteq V$, an algorithm can query an oracle to find its value $f(S)$. Furthermore, all submodular functions we deal with are assumed to be non-negative. We also denote the ground set $V = [n] = \{1, 2, \cdots, n\}$.

We emphasize that our focus is on submodular functions that are *not required to be monotone* (i.e., we do *not* require that $f(X) \leq f(Y)$ for $X \subseteq Y \subseteq V$). Non-monotone submodular functions appear in several places including cut functions in weighted directed or undirected graphs or even hypergraphs, maximum facility location, maximum entropy sampling, and certain constraint satisfaction problems.

Given a weight vector w for the ground set V, and a knapsack of capacity C, the associated *knapsack constraint* is that the sum of weights of elements in the solution S should not exceed the capacity C, i.e, $\sum_{j \in S} w_j \leq C$. In our usage, we consider k knapsack constraints defined by weight vectors w^i and capacities C_i, for $i = 1, \ldots, k$.

We assume some familiarity with matroids [41] and associated algorithmics [46]. Briefly, for a matroid $\mathcal{M}$, we denote the ground set of $\mathcal{M}$ by $\mathcal{E}(\mathcal{M})$, its set of independent sets by $\mathcal{I}(\mathcal{M})$, and its set of bases by $\mathcal{B}(\mathcal{M})$. For a given matroid $\mathcal{M}$, the associated *matroid constraint* is $S \in \mathcal{I}(\mathcal{M})$ and the associated *matroid base constraint* is $S \in \mathcal{B}(\mathcal{M})$. As is standard, $\mathcal{M}$ is a *uniform matroid* of rank r if $\mathcal{I}(\mathcal{M}) := \{X \subseteq \mathcal{E}(\mathcal{M}) : |X| \leq r\}$. A *partition matroid* is the direct sum of uniform matroids. Note that uniform matroid constraints are equivalent to cardinality constraints, i.e, $|S| \leq k$. In our usage, we deal with k matroids $\mathcal{M}_1, \ldots, \mathcal{M}_k$ on the common ground set $V := \mathcal{E}(\mathcal{M}_1) = \cdots = \mathcal{E}(\mathcal{M}_k)$ (which is also the ground set of our submodular function f), and we let $\mathcal{I}_i := \mathcal{I}(\mathcal{M}_i)$ for $i = 1, \ldots, k$.

Background. Optimizing submodular functions is a central subject in operations research and combinatorial optimization [36]. This problem appears in many important optimization problems including cuts in graphs [19, 42, 26], rank function of matroids [12, 16], set covering problems [13], plant location problems [9, 10, 11, 2], certain satisfiability problems [25, 14], and maximum entropy sampling [32, 33]. Other than many heuristics that have been developed for optimizing these functions [20, 21, 27, 44, 31], many exact and constant-factor approximation algorithms are also known for this problem [39, 40, 45, 26, 15, 50, 18]. In some settings such as set covering or matroid optimization, the relevant submodular functions are monotone. Here, we are more interested in the general case where $f(S)$ is not necessarily monotone.

Unlike submodular minimization [45, 26], submodular function maximization is NP-hard as it generalizes many NP-hard problems, like Max-Cut [19, 14] and maximum facility location [9, 10, 2]. Other than generalizing combinatorial optimization problems like Max Cut [19], Max Directed Cut [4, 22], hypergraph cut problems, maximum facility location [2, 9, 10], and certain restricted satisfiability problems [25, 14], maximizing non-monotone submodular functions have applications in a variety of problems, e.g, computing the core value of supermodular games [47], and optimal marketing for revenue maximization over social networks [23]. As an example, we describe one important application in the statistical design of experiments. Let A be the n-by-n covariance matrix of a set of Gaussian random variables indexed by $[n]$. For $S \subseteq [n]$, let $A[S]$ denote the principal submatrix of A indexed by S. It is well known that the entropy[1] of the random variables indexed by S is

$$f(S) = \frac{1+\ln(2\pi)}{2}|S| + \frac{1}{2}\ln\det A[S] \ .$$

Certainly $|S|$ is non-negative, monotone and (sub)modular on $[n]$. It is also well known that $\ln\det A[S]$ is submodular on $[n]$, but $\ln\det A[S]$ is not even approximately monotone (see [30, Section 8.2]): For example, for

$$A = \begin{pmatrix} \delta & \sqrt{\delta-1} \\ \sqrt{\delta-1} & 1 \end{pmatrix},$$

with $\delta > 1$, it is clear that $\ln\det A[\{1,2\}] = 0$, while $\ln\det A[\{1\}] = \ln(\delta)$ can be made arbitrarily large, by taking δ large. So the entropy $f(S)$ is submodular but not generally monotone. The *maximum entropy sampling problem*, introduced in [48], is to maximize $f(S)$ over subsets $S \subseteq [n]$ having cardinality s fixed. So the maximum entropy sampling problem is precisely one of maximizing a non-monotone submodular function subject to a cardinality constraint. Of course a cardinality constraint is just a matroid base constraint for a uniform matroid. The maximum entropy sampling problem has mostly been studied from a computational point of view (often in the context of locating environmental monitoring stations), focusing on calculating optimal solutions for moderate-sized instances (say $n < 200$) using mathematical programming methodologies (e.g, see [32, 33, 34, 29, 6, 5]), and our results provide the first set of algorithms with provable constant-factor approximation guarantee (for cases in which the entropy is non-negative).

Recently, a $\frac{2}{5}$-approximation was developed for maximizing non-negative non-monotone submodular functions without any side constraints [15]. This algorithm also provides a tight $\frac{1}{2}$-approximation algorithm for maximizing a symmetric[2] submodular function [15]. However, the algorithms developed in [15] for non-monotone submodular maximization do not handle any extra constraints.

For the problem of maximizing a monotone submodular function subject to a matroid or multiple knapsack constraints, tight $\left(1-\frac{1}{e}\right)$-approximations are known [39, 7, 51, 49, 28]. Maximizing monotone submodular functions over k matroid constraints was considered in [40], where a $\left(\frac{1}{k+1}\right)$-approximation was obtained. This bound is currently the best known ratio, even in the special case of partition matroid constraints. However, none of these results generalize to non-monotone submodular functions.

Better results are known either for specific submodular functions or for special classes of matroids. A $\frac{1}{k}$- approximation algorithm using local search was designed in [43] for the problem of maximizing a linear function subject to k matroid constraints. Constant factor approximation algorithms are known for the problem of maximizing directed cut [1] or hypergraph cut [3] subject to a uniform matroid (i.e. cardinality) constraint.

Hardness of approximation results are known even for the special case of maximizing a linear function subject to k partition matroid constraints. The best known lower bound is an $\Omega(\frac{k}{\log k})$ hardness of approximation [24]. Moreover, for the unconstrained maximization of non-monotone submodular functions, it has been shown that achieving a factor better than $\frac{1}{2}$ cannot be done using a subexponential number of value queries [15].

Our Results. In this paper, we give the first constant-factor approximation algorithms for maximizing a non monotone submodular function subject to multiple matroid constraints, or multiple knapsack constraints. More specifically, we give the following new results (below $\epsilon > 0$ is any constant).

(1) For every constant $k \geq 1$, we present a $\left(\frac{1}{k+2+\frac{1}{k}+\epsilon}\right)$-approximation algorithm for maximizing any non-negative submodular function subject to k matroid constraints. This implies a $\left(\frac{1}{4+\epsilon}\right)$-approximation algorithm for maximizing non-monotone submodular functions subject to a single matroid constraint. Moreover, this algorithm is a $\left(\frac{1}{k+2+\epsilon}\right)$-approximation in the case of *symmetric* submodular functions. This algorithm involves a natural local search procedure, that is iteratively executed $k+1$ times. Asymptotically, this result is nearly best possible because there is an $\Omega(\frac{k}{\log k})$ hardness of approximation, even in the monotone case [24].

(2) For every constant $k \geq 1$, we present a $\left(\frac{1}{5}-\epsilon\right)$- approximation algorithm for maximizing any nonnegative submodular function subject to a k-dimensional knapsack constraint. To achieve the approximation guarantee, we first give a $\left(\frac{1}{4}-\epsilon\right)$-approximation algorithm for a fractional relaxation (similar to the one used in [51]). This is again based

[1]sometimes also referred to as *differential entropy* or *continuous entropy*

[2]The function $f : 2^V \to \mathbb{R}$ is symmetric if for all $S \subseteq V$, $f(S) = f(V \setminus S)$. For example, cut functions in undirected graphs are well-known examples of symmetric (non-monotone) submodular functions

on a local search procedure, that is iterated twice. We then use a simple randomized rounding technique to convert a fractional solution to an integral one. A similar approach was recently used in [28] for maximizing a monotone submodular function over multiple knapsack constraints. However their algorithm for the fractional relaxation uses the 'continuous greedy' algorithm of Vondrák [51] that requires a monotone function; moreover, even their rounding method is not directly applicable to non-monotone submodular functions.

(3) For submodular maximization under $k \geq 2$ *partition matroid* constraints, we obtain improved approximation guarantees. We give a $\left(\frac{1}{k+1+\frac{1}{k-1}+\epsilon}\right)$-approximation algorithm for maximizing non-monotone submodular functions subject to k partition matroids. Moreover, our idea gives a $\left(\frac{1}{k+\epsilon}\right)$-approximation algorithm for maximizing a monotone submodular function subject to $k \geq 2$ partition matroid constraints. This is an improvement over the previously best known bound of $\frac{1}{k+1}$ from [40].

(4) Finally, we study submodular maximization subject to a matroid *base* constraint. We give a $\left(\frac{1}{3}-\epsilon\right)$-approximation in the case of symmetric submodular functions. Our result for general submodular functions only holds for special matroids: we obtain a $(\frac{1}{6}-\epsilon)$-approximation when the matroid contains two disjoint bases. In particular, this implies a $\left(\frac{1}{6}-\epsilon\right)$-approximation for the problem of maximizing any non-negative submodular function subject to an exact cardinality constraint. Previously, only special cases of directed cut [1] or hypergraph cut [3] subject to an exact cardinality constraint were considered.

Due to lack of space, in this paper we only present (1) general matroid constraints in Section 2, and (2) knapsack constraints in Section 3. Details of the other two results can be found in the full version [35].

All our algorithms run in time $n^{O(k)}$, where k is the number of matroid or knapsack constraints.

Our main technique for the above results is local search. Our local search algorithms are different from the previously used variant of local search for unconstrained maximization of a non-negative submodular function [15], or the local search algorithms used for Max Directed Cut [4, 22]. In the design of our algorithms, we also use structural properties of matroids, a fractional relaxation of submodular functions, and a randomized rounding technique.

2. MATROID CONSTRAINTS

In this section, we give an approximation algorithm for submodular maximization subject to k matroid constraints. The problem is as follows: Let f be a *non-negative* submodular function defined on ground set V. Let $\mathcal{M}_1, \cdots, \mathcal{M}_k$ be k arbitrary matroids on the common ground set V. For each matroid $\mathcal{M}_j$ (with $j \in [k]$) we denote the set of its independent sets by $\mathcal{I}_j$. We consider the following problem:

$$\max\left\{f(S) \ : \ S \in \cap_{j=1}^{k}\mathcal{I}_j\right\}. \tag{1}$$

We give an approximation algorithm for this problem using value queries to f that runs in time $n^{O(k)}$. The starting point is the following local search algorithm. Starting with $S = \emptyset$, repeatedly perform one of the following local improvements:

- **Delete operation.** If $e \in S$ such that $f(S \setminus \{e\}) > f(S)$, then $S \leftarrow S \setminus \{e\}$.
- **Exchange operation.** If $d \in V \setminus S$ and $e_i \in S \cup \{\emptyset\}$ (for $1 \leq i \leq k$) are such that $(S \setminus \{e_i\}) \cup \{d\} \in \mathcal{I}_i$ for all $i \in [k]$ and $f((S \setminus \{e_1, \cdots, e_k\}) \cup \{d\}) > f(S)$, then $S \leftarrow (S \setminus \{e_1, \cdots, e_k\}) \cup \{d\}$.

When dealing with a single matroid constraint ($k = 1$), the local operations correspond to: *delete* an element, *add* an element (i.e. an exchange when no element is dropped), *swap* a pair of elements (i.e. an element from outside the current set is exchanged with an element from the set). With $k \geq 2$ matroid constraints, we permit more general exchange operations, involving adding one element and dropping *up to* k elements.

Note that the size of any local neighborhood is at most n^{k+1}, which implies that each local step can be performed in polynomial time for a constant k. Let S denote a locally optimal solution. Next we prove a key lemma for this local search algorithm, which is used in analyzing our algorithm. Before presenting the lemma, we state a useful exchange property of matroids (see [46]). Intuitively, this property states that for any two independent sets I and J, we can add any element of J to the set I, and remove at most one element from I while keeping the set independent. Moreover, each element of I is allowed to be removed by at most one element of J.

THEOREM 1. *Let $\mathcal{M}$ be a matroid and $I, J \in \mathcal{I}(\mathcal{M})$ be two independent sets. Then there is a mapping $\pi : J \setminus I \rightarrow (I \setminus J) \cup \{\emptyset\}$ such that:*

1. *$(I \setminus \pi(b)) \cup \{b\} \in \mathcal{I}(\mathcal{M})$ for all $b \in J \setminus I$.*
2. *$|\pi^{-1}(e)| \leq 1$ for all $e \in I \setminus J$.*

PROOF. We outline the proof for completeness. We proceed by induction on $t = |J \setminus I|$. If $t = 0$, there is nothing to prove; so assume $t \geq 1$. Suppose there is an element $b \in J \setminus I$ with $I \cup \{b\} \in \mathcal{I}(\mathcal{M})$. In this case we apply induction on I and $J' = J \setminus \{b\}$ (where $|J' \setminus I| = t - 1 < t$). Since $I \setminus J' = I \setminus J$, we obtain a map $\pi' : J' \setminus I \rightarrow (I \setminus J) \cup \{\emptyset\}$ satisfying the two conditions. The desired map π for $\langle I, J\rangle$ is then $\pi(b) = \emptyset$ and $\pi(b') = \pi'(b')$ for all $b' \in J \setminus I \setminus \{b\} = J' \setminus I$.

Now we may assume that I is a maximal independent set in $I \cup J$. Let $\mathcal{M}' \subseteq \mathcal{M}$ denote the matroid $\mathcal{M}$ restricted to $I \cup J$; so I is a base in $\mathcal{M}'$. We augment J to some base $\tilde{J} \supseteq J$ in $\mathcal{M}'$ (since any maximal independent set in $\mathcal{M}'$ is a base). Thus we have two bases I and $\tilde{J}$ in $\mathcal{M}'$. Theorem 39.12 from [46] implies the existence of elements $b \in \tilde{J} \setminus I$ and $e \in I \setminus \tilde{J}$ such that both $(\tilde{J} \setminus b) \cup \{e\}$ and $(I \setminus e) \cup \{b\}$ are bases in $\mathcal{M}'$. Note that $J' := (J \setminus \{b\}) \cup \{e\} \subseteq (\tilde{J} \setminus \{b\}) \cup \{e\} \in \mathcal{I}(\mathcal{M})$; also $I \setminus J' = (I \setminus J) \setminus \{e\}$ and $J' \setminus I = (J \setminus I) \setminus \{b\}$. By induction on I and J' (since $|J' \setminus I| = t - 1 < t$) we obtain map $\pi' : J' \setminus I \rightarrow I \setminus J'$ satisfying the two conditions. The map π for $\langle I, J\rangle$ is then $\pi(b) = e$ and $\pi(b') = \pi'(b')$ for all $b' \in (J \setminus I) \setminus \{b\} = J' \setminus I$. The first condition on π is satisfied by induction (for elements $(J \setminus I) \setminus \{b\}$) and since $(I \setminus e) \cup \{b\} \in \mathcal{I}(\mathcal{M})$ (see above). The second condition on π is satisfied by induction and the fact that $e \notin I \setminus J'$. $\square$

LEMMA 1. *For a local optimal solution S and any $C \in \cap_{j=1}^{k}\mathcal{I}_j$, $(k+1)\cdot f(S) \geq f(S\cup C)+k\cdot f(S\cap C)$. Additionally for $k=1$, if $S \in \mathcal{I}_1$ is any locally optimal solution under only the swap operation, and $C \in \mathcal{I}_1$ with $|S|=|C|$, then $2\cdot f(S) \geq f(S\cup C)+f(S\cap C)$.*

PROOF. The following proof is due to Jan Vondrák [52]. Our original proof [35] was more complicated– we thank Jan for letting us present this simplified proof.

For each matroid $\mathcal{M}_j$ ($j \in [k]$), because both $C, S \in \mathcal{I}_j$ are independent sets, Theorem 1 implies a mapping $\pi_j : C\setminus S \to (S\setminus C)\cup\{\emptyset\}$ such that:

1. $(S\setminus \pi_j(b))\cup\{b\} \in \mathcal{I}_j$ for all $b \in C\setminus S$.
2. $|\pi_j^{-1}(e)| \leq 1$ for all $e \in S\setminus C$.

When $k=1$ and $|S|=|C|$, Corollary 39.12a from [46] implies the stronger condition that $\pi_1 : C\setminus S \to S\setminus C$ is in fact a *bijection*.

For each $b \in C\setminus S$, let $A_b = \{\pi_1(b),\cdots,\pi_k(b)\}$. Note that $(S\setminus A_b)\cup\{b\} \in \cap_{j=1}^{k}\mathcal{I}_j$ for all $b \in C\setminus S$. Hence $(S\setminus A_b)\cup\{b\}$ is in the local neighborhood of S, and by local optimality under exchanges:

$$f(S) \geq f\left((S\setminus A_b)\cup\{b\}\right), \qquad \forall b \in C\setminus S. \quad (2)$$

In the case $k=1$ with $|S|=|C|$, these are only *swap* operations (because π_1 is a bijection here).

By the property of mappings $\{\pi_j\}_{j=1}^{k}$, each element $i \in S\setminus C$ is contained in $n_i \leq k$ of the sets $\{A_b \mid b \in C\setminus S\}$; and elements of $S\cap C$ are contained in none of these sets. So the following inequalities are implied by local optimality of S under deletions.

$$(k-n_i)\cdot f(S) \geq (k-n_i)\cdot f(S\setminus\{i\}), \qquad \forall i \in S\setminus C. \quad (3)$$

Note that these inequalities are not required when $k=1$ and $|S|=|C|$ (then $n_i=k$ for all $i \in S\setminus C$).

For any $b \in C\setminus S$, we have (below, the first inequality is submodularity and the second is from (2)):

$$\begin{aligned} f(S\cup\{b\})-f(S) &\leq f\left((S\setminus A_b)\cup\{b\}\right)-f\left(S\setminus A_b\right) \\ &\leq f(S)-f\left(S\setminus A_b\right) \end{aligned}$$

Adding this inequality over all $b \in C\setminus S$ and using submodularity,

$$\begin{aligned} f(S\cup C)-f(S) &\leq \sum_{b\in C\setminus S}[f(S\cup\{b\})-f(S)] \\ &\leq \sum_{b\in C\setminus S}[f(S)-f\left(S\setminus A_b\right)] \end{aligned}$$

Adding to this, the inequalities (3), i.e. $0 \leq (k-n_i)\cdot[f(S)-f(S\setminus\{i\})]$ for all $i \in S\setminus C$,

$$\begin{aligned} f(S\cup C)-f(S) &\leq \sum_{b\in C\setminus S}[f(S)-f\left(S\setminus A_b\right)] \\ &\quad + \sum_{i\in S\setminus C}(k-n_i)\cdot[f(S)-f(S\setminus\{i\})] \\ &= \sum_{l=1}^{\lambda}[f(S)-f\left(S\setminus T_l\right)] \qquad (4) \end{aligned}$$

where $\lambda = |C\setminus S| + \sum_{i\in S\setminus C}(k-n_i)$ and $\{T_l\}_{l=1}^{\lambda}$ is some collection of subsets of $S\setminus C$ such that each $i \in S\setminus C$ appears in *exactly* k of these subsets. We simplify the expression (4) using the following claim.

CLAIM 1. *Let $f : 2^V \to \mathbb{R}_+$ be any submodular function and $S' \subseteq S \subseteq V$. Let $\{T_l\}_{l=1}^{\lambda}$ be a collection of subsets of $S\setminus S'$ such that each element of $S\setminus S'$ appears in exactly k of these subsets. Then,*

$$\sum_{l=1}^{\lambda}[f(S)-f\left(S\setminus T_l\right)] \leq k\cdot\left(f(S)-f(S')\right)$$

PROOF. Let $s=|S|$ and $|S'|=c$; number the elements of S as $\{1,2,\cdots,s\}=[s]$ such that $S'=\{1,2,\cdots,c\}=[c]$. Then for any $T\subseteq S\setminus S'$, by submodularity: $f(S)-f(S\setminus T) \leq \sum_{p\in T}[f([p])-f([p-1])]$. Using this we obtain:

$$\begin{aligned} \sum_{l=1}^{\lambda}[f(S)-f\left(S\setminus T_l\right)] &\leq \sum_{l=1}^{\lambda}\sum_{p\in T_l}[f([p])-f([p-1])] \\ &= k\sum_{i=c+1}^{s}[f([i])-f([i-1])] \\ &= k\cdot\left(f(S)-f(S')\right) \end{aligned}$$

The second equality follows from $S\setminus C=\{c+1,\cdots,s\}$ and the fact that each element of $S\setminus C$ appears in exactly k of the sets $\{T_l\}_{l=1}^{\lambda}$. The last equality is due to a telescoping summation. ☐

Setting $S'=S\cap C$ in Claim 1 to simplify expression (4), we obtain $(k+1)\cdot f(S) \geq f(S\cup C)+k\cdot f(S\cap C)$.

Observe that when $k=1$ and $|S|=|C|$, we only used the inequalities (2) from the local search, which are only swap operations. Hence in this case, the statement also holds for any solution S that is locally optimal under only swap operations. In the general case, we use both inequalities (2) (from exchange operations) and inequalities (3) (from deletion operations). ☐

A simple consequence of Lemma 1 implies bounds analogous to known approximation factors [40, 43] in the cases when the submodular function f has additional structure.

COROLLARY 1. *For a locally optimal solution S and any $C \in \cap_{j=1}^{k}\mathcal{I}_j$ the following inequalities hold:*

1. *$f(S) \geq f(C)/(k+1)$ if function f is monotone,*
2. *$f(S) \geq f(C)/k$ if function f is linear.*

The local search algorithm defined above could run for an exponential amount of time until it reaches a locally optimal solution. To ensure polynomial runtime, we follow the standard approach of an approximate local search under a suitable (small) parameter $\epsilon > 0$, as described in Figure 1. The following Lemma 2 is a simple extension of Lemma 1 for approximate local optimum.

LEMMA 2. *For an approximately locally optimal solution S (in procedure B) and any $C \in \cap_{j=1}^{k}\mathcal{I}_j$, $(1+\epsilon)(k+1)\cdot f(S) \geq f(S\cup C)+k\cdot f(S\cap C)$ where $\epsilon > 0$ the parameter used in the procedure B (Figure 1). Additionally for $k=1$, if $S \in \mathcal{I}_1$ is any locally optimal solution under only the swap operation, and $C \in \mathcal{I}_1$ with $|S|=|C|$, then $2(1+\epsilon)\cdot f(S) \geq f(S\cup C)+f(S\cap C)$.*

PROOF. The proof of this lemma is almost identical to the proof of the Lemma 1 the only difference is that left-hand sides of inequalities (2) and inequalities (3) are multiplied

Approximate Local Search Procedure B:

Input: Ground set X of elements and value oracle access to submodular function f.

1. Set $v \leftarrow \arg\max\{f(u) \mid u \in X\}$ and $S \leftarrow \{v\}$.
2. While one of the following local operations applies, update S accordingly.
   - **Delete operation on S.** If $e \in S$ such that $f(S \setminus \{e\}) \geq (1 + \frac{\epsilon}{n^4})f(S)$, then $S \leftarrow S \setminus \{e\}$.
   - **Exchange operation on S.** If $d \in X \setminus S$ and $e_i \in S \cup \{\emptyset\}$ (for $1 \leq i \leq k$) are such that $(S \setminus \{e_i\}) \cup \{d\} \in \mathcal{I}_i$ for all $i \in [k]$ and $f((S \setminus \{e_1, \cdots, e_k\}) \cup \{d\}) \geq (1 + \frac{\epsilon}{n^4})f(S)$, then $S \leftarrow (S \setminus \{e_1, \cdots, e_k\}) \cup \{d\}$.

Figure 1: The approximate local search procedure.

Algorithm A:

1. Set $V_1 = V$.
2. For $i = 1, \cdots, k+1$, do:
   (a) Apply the approximate local search procedure B on the ground set V_i to obtain a solution $S_i \subseteq V_i$ corresponding to the problem:
   $$\max\{f(S) \ : \ S \in \cap_{j=1}^{k} \mathcal{I}_j, \ S \subseteq V_i\}. \tag{5}$$
   (b) Set $V_{i+1} = V_i \setminus S_i$.
3. Return the solution corresponding to $\max\{f(S_1), \cdots, f(S_{k+1})\}$.

Figure 2: Approximation algorithm for submodular maximization under k matroid constraints.

by $1 + \frac{\epsilon}{n^4}$. Therefore, after following the steps in Lemma 1, we obtain the inequality:

$$(k + 1 + \frac{\epsilon}{n^4}\lambda) \cdot f(S) \geq f(S \cup C) + k \cdot f(S \cap C).$$

Since $\lambda \leq (k+1)n$ (see Lemma 1) and we may assume that $n^4 >> (k+1)n$, we obtain the lemma. □

We now present the main algorithm (Figure 2) for submodular maximization over matroid constraints. This performs the approximate local search procedure B iteratively $k+1$ times, and outputs the best solution found.

THEOREM 2. *Algorithm A in Figure 2 is a $\left(\frac{1}{(1+\epsilon)(k+2+\frac{1}{k})}\right)$-approximation algorithm for maximizing a non-negative submodular function subject to any k matroid constraints, running in time $n^{O(k)}$.*

PROOF. Bounding the running time of Algorithm A is easy. The parameter $\epsilon > 0$ in Procedure B is any value such that $\frac{1}{\epsilon}$ is at most a polynomial in n. Note that using approximate local operations in the local search procedure B (in Figure 1) makes the running time of the algorithm polynomial. The reason is as follows: one can easily show that for any ground set X of elements, the value of the initial set $S = \{v\}$ is at least $\mathsf{Opt}(X)/n$, where $\mathsf{Opt}(X)$ is the optimal value of problem (1) restricted to X. Each local operation in procedure B increases the value of the function by a factor $1+\frac{\epsilon}{n^4}$. Therefore, the number of local operations for procedure B is at most $\log_{1+\frac{\epsilon}{n^4}} \frac{\mathsf{Opt}(X)}{\frac{\mathsf{Opt}(X)}{n}} = O(\frac{1}{\epsilon}n^4 \log n)$, and thus the running time of the whole procedure is $\frac{1}{\epsilon} \cdot n^{O(k)}$. Moreover, the number of procedure calls of Algorithm A for procedure B is $k+1$, and thus the running time of Algorithm A is also polynomial.

Next, we prove the performance guarantee of Algorithm A. Let C denote the optimal solution to the original problem $\max\{f(S) \ : \ S \in \cap_{j=1}^{k}\mathcal{I}_j, \ S \subseteq V\}$. Let $C_i = C \cap V_i$ for each $i \in [k+1]$; so $C_1 = C$. Observe that C_i is a feasible solution to the problem (5) solved in the ith iteration. Applying Lemma 2 to problem (5) using the local optimum S_i and solution C_i, we obtain for all $1 \leq i \leq k+1$:

$$(1+\epsilon)(k+1) \cdot f(S_i) \geq f(S_i \cup C_i) + k \cdot f(S_i \cap C_i), \tag{6}$$

Using $f(S) \geq \max_{i=1}^{k+1} f(S_i)$, we add these $k+1$ inequalities and simplify inductively as given in the following claim.

CLAIM 2. *For any $1 \leq l \leq k+1$, we have:*

$$\begin{aligned} & (1+\epsilon)(k+1)^2 \cdot f(S) \\ \geq \quad & (l-1) \cdot f(C) + f(\cup_{p=1}^{l} S_p \cup C_1) + \sum_{i=l+1}^{k+1} f(S_i \cup C_i) \\ & + \sum_{p=1}^{l-1} (k-l+p) \cdot f(S_p \cap C_p) + k \cdot \sum_{i=l}^{k+1} f(S_i \cap C_i). \end{aligned}$$

PROOF. We argue by induction on l. The base case $l = 1$ is trivial, by just considering the sum of the $k+1$ inequalities in statement (6) above. Assuming the statement for some value $1 \leq l < k+1$, we prove the corresponding statement for $l+1$, using the simplification in Figure 3.

$$\begin{aligned}
& (1+\epsilon)(k+1)^2 \cdot f(S) \\
\geq\ & (l-1)\cdot f(C) + f(\cup_{p=1}^{l} S_p \cup C_1) + \sum_{i=l+1}^{k+1} f(S_i \cup C_i) + \sum_{p=1}^{l-1}(k-l+p) f(S_p \cap C_p) + k \cdot \sum_{i=l}^{k+1} f(S_i \cap C_i) \qquad (7) \\
=\ & (l-1)\cdot f(C) + f(\cup_{p=1}^{l} S_p \cup C_1) + f(S_{l+1} \cup C_{l+1}) + \sum_{i=l+2}^{k+1} f(S_i \cup C_i) + \sum_{p=1}^{l-1}(k-l+p) f(S_p \cap C_p) + k \cdot \sum_{i=l}^{k+1} f(S_i \cap C_i) \\
\geq\ & (l-1)\cdot f(C) + f(\cup_{p=1}^{l+1} S_p \cup C_1) + f(C_{l+1}) + \sum_{i=l+2}^{k+1} f(S_i \cup C_i) + \sum_{p=1}^{l-1}(k-l+p) f(S_p \cap C_p) + k \cdot \sum_{i=l}^{k+1} f(S_i \cap C_i) \qquad (8) \\
=\ & (l-1)\cdot f(C) + f(\cup_{p=1}^{l+1} S_p \cup C_1) + f(C_{l+1}) + \sum_{p=1}^{l} f(S_p \cap C_p) + \sum_{i=l+2}^{k+1} f(S_i \cup C_i) \\
& + \sum_{p=1}^{l}(k-l-1+p) f(S_p \cap C_p) + k \cdot \sum_{i=l+1}^{k+1} f(S_i \cap C_i) \\
\geq\ & l \cdot f(C) + f(\cup_{p=1}^{l+1} S_p \cup C_1) + \sum_{i=l+2}^{k+1} f(S_i \cup C_i) + \sum_{p=1}^{l}(k-l-1+p) f(S_p \cap C_p) + k \cdot \sum_{i=l+1}^{k+1} f(S_i \cap C_i). \qquad (9)
\end{aligned}$$

Figure 3: Inequalities used in the inductive step for Claim 2.

Inequality (7) is the induction hypothesis, inequality (8) follows from submodularity using:

$$\left(\cup_{p=1}^{l} S_p \cup C_1\right) \cap (S_{l+1} \cup C_{l+1}) = C_{l+1},$$

and inequality (9) is by submodularity since $\left(\cup_{p=1}^{l} S_p \cap C_p\right) \cup C_{l+1} = C$. □

Using the statement of Claim 2 when $l = k+1$, we obtain $(1+\epsilon)(k+1)^2 \cdot f(S) \geq k \cdot f(C)$. □

Finally, we give an improved approximation algorithm for symmetric submodular functions f, that satisfy $f(S) = f(\overline{S})$ for all $S \subset V$. Symmetric submodular functions have been considered widely in the literature [17, 42], and it appears that symmetry allows for better approximation results and thus deserves separate attention.

Theorem 3. *There is a* $\left(\frac{1}{(1+\epsilon)(k+2)}\right)$*-approximation algorithm for maximizing a non-negative symmetric submodular functions subject to* k *matroid constraints.*

Proof. The algorithm for symmetric submodular functions is much simpler. In this case, we only need to perform *one* iteration of the approximate local search procedure B (as opposed to $k+1$ in Theorem 2). Let C denote the optimal solution, and S_1 the result of the local search (on V). Then Lemma 1 implies:

$$\begin{aligned}
(1+\epsilon)(k+1)\cdot f(S_1) &\geq f(S_1 \cup C) + k \cdot f(S_1 \cap C) \\
&\geq f(S_1 \cup C) + f(S_1 \cap C).
\end{aligned}$$

Because f is symmetric, we also have $f(S_1) = f(\overline{S_1})$. Adding these two inequalities,

$$\begin{aligned}
(1+\epsilon)(k+2)\cdot f(S_1) &\geq f(\overline{S_1}) + f(S_1 \cup C) + f(S_1 \cap C) \\
&\geq f(C \setminus S_1) + f(S_1 \cap C) \geq f(C)\ .
\end{aligned}$$

Thus we have the desired approximation guarantee. □

3. KNAPSACK CONSTRAINTS

In this section, we give an approximation algorithm for submodular maximization subject to multiple knapsack constraints. Let $f : 2^V \to \mathbb{R}_+$ be a submodular function, and $w^1, \cdots, w^k$ be k weight-vectors corresponding to knapsacks having capacities $C_1, \cdots, C_k$ respectively. The problem we consider in this section is:

$$\max\{f(S) \ : \ \sum_{j\in S} w_j^i \leq C_i,\ \forall 1 \leq i \leq k,\ S \subseteq V\}. \qquad (10)$$

By scaling each knapsack, we assume that $C_i = 1$ for all $i \in [k]$; we also assume that all weights are rational. We denote $f_{max} = \max\{f(v) \ : \ v \in V\}$. We assume without loss of generality that for every $i \in V$, the singleton solution $\{i\}$ is feasible for all the knapsacks (otherwise such elements can be dropped from consideration). To solve the above problem, we first define a fractional relaxation of the submodular function, and give an approximation algorithm for this fractional relaxation (Section 3.2). Then, we show how to design an approximation algorithm for the original integral problem using the solution for the fractional relaxation (Section 3.3). Let $F : [0,1]^n \to \mathbb{R}_+$, the *fractional relaxation of* f, be the 'extension-by-expectation' [7],

$$F(x) = \sum_{S \subseteq V} f(S) \cdot \Pi_{i\in S} x_i \cdot \Pi_{j \notin S}(1 - x_j).$$

Note that F is a multi-linear polynomial in variables $x_1, \cdots, x_n$, and has continuous derivatives of all orders. Furthermore, as shown in Vondrák [51], for all $i, j \in V$, $\frac{\partial^2}{\partial x_j \partial x_i} F \leq 0$ everywhere on $[0,1]^n$; we refer to this condition as *continuous submodularity.*

3.1 Extending function f on scaled ground sets

Let $s_i \in \mathbb{Z}_+$ be arbitrary values for each $i \in V$. Define a new ground-set U that contains s_i 'copies' of each element $i \in V$; so the total number of elements in U is $\sum_{i\in V} s_i$. We will denote any subset T of U as $T = \cup_{i\in V} T_i$ where each T_i

Input: Knapsack weights $\{w^s\}_{s=1}^k$, variable upper bounds $\{u_i \in [0,1]\}_{i=1}^n$, discretization $\mathcal{G}$, parameter ϵ, and value oracle access to submodular function f.

1. Set $a \leftarrow \arg\max\{u_a \cdot f(\{a\}) \mid a \in X\}$.
2. If $u_a \cdot f(\{a\}) \le f(\emptyset)$, set $y(i) \leftarrow 0$ for all $i \in V$; else set
$$y(i) = \begin{cases} u_a & i = a \\ 0 & i \in V \setminus \{a\} \end{cases}$$
3. While the following local operation applies, update y accordingly.
   - Let $A, D \subseteq [n]$ with $|A|, |D| \le k$. Decrease the variables $y(D)$ to any values in $\mathcal{G}$ and increase variables $y(A)$ to any values in $\mathcal{G}$ such that the resulting solution y' still satisfies all knapsacks and $y' \in \mathcal{U}$. If $F(y') > (1+\epsilon) \cdot F(y)$ then set $y \leftarrow y'$.
4. Output y as the local optimum.

Figure 4: The approximate local search procedure for Problem (11)**.**

consists of all copies of element $i \in V$ from T. Now define function $g : 2^U \to \mathbb{R}_+$ as $g(\cup_{i\in V} T_i) = F(\cdots, \frac{|T_i|}{s_i}, \cdots)$. The following lemma is Lemma 2.3 from [37].

Lemma 3 ([37]). *Set function g is a submodular function on ground set U.*

3.2 Solving the fractional relaxation

We now present an algorithm for obtaining a near-optimal fractional feasible solution for maximizing a non-negative submodular function over k knapsack constraints. Let $\{w^s\}_{s=1}^k$ denote the weight-vectors in each of the k knapsacks; recall that all knapsacks have capacity one. For ease of exposition, it is useful to consider a more general problem where each variable has additional upper bounds $\{u_i \in [0,1]\}_{i=1}^n$, i.e.,

$$\max\{F(y) \ : \ w^s \cdot y \le 1 \ \ \forall s \in [k], \ \ 0 \le y_i \le u_i \ \ \forall i \in V\}. \tag{11}$$

We first define a local search procedure for problem (11), and prove some properties of it (Lemmas 4 and 5). Then we present the approximation algorithm for solving the fractional relaxation when all upper-bounds are one (Theorem 4).

3.2.1 Local search for problem (11)

Denote the region $\mathcal{U} := \{y \ : \ 0 \le y_i \le u_i \ \forall i \in V\}$. For the local search, we only consider values for each variable from a discrete set of values in $[0,1]$, namely $\mathcal{G} = \{p \cdot \zeta \ : \ p \in \mathbb{N}, \ 0 \le p \le \frac{1}{\zeta}\}$ where $\zeta = \frac{1}{8n^4}$. Using standard scaling methods, we assume (at the loss of $1 + o(1)$ factor in the optimal value of (11)) that all upper bounds $\{u_i\}_{i\in V} \subseteq \mathcal{G}$. Let $\epsilon > 0$ be a parameter to be fixed later. The local search procedure for Problem (11) is given in Figure 4. Note that the size of each local neighborhood is $n^{O(k)}$. The following simple lemma bounds the runtime of the local search procedure.

Lemma 4. *The local search procedure (Figure 4) terminates in $O(\frac{1}{\epsilon}\log n)$ iterations.*

Proof. Observe that the initial solution y_o chosen in Step 2 satisfies $F(y_o) \ge \max\{u_a \cdot f(\{a\}), f(\emptyset)\}$, where a is the index chosen in Step 1. Submodularity implies that $f(R) \le \sum_{e\in R} f(\{e\})$ for any $\emptyset \subsetneq R \subseteq [n]$. Thus for any $x \in \mathcal{U}$ (using linearity of expectation),

$$\begin{aligned} F(x) &\le \sum_{i=1}^n x_i \cdot f(\{i\}) + f(\emptyset) \le \sum_{i=1}^n u_i \cdot f(\{i\}) + f(\emptyset) \\ &\le (n+1) \cdot F(y_o) \end{aligned}$$

Since the F-value increases by a $1+\epsilon$ factor in each iteration, the number of iterations of this local search is bounded by $O(\frac{1}{\epsilon}\log n)$. ☐

Define $f_{max} := \max\{f(\emptyset), \max\{f(\{v\}) \ : \ v \in V\}\}$; by submodularity, $\max_{S\subseteq[n]} f(S) \le n \cdot f_{max}$. Let $\tilde{y} \in \mathcal{U} \cap \mathcal{G}^n$ denote a local optimal solution obtained upon running the local search in Figure 4. We also need the following simple claim based on the discretization $\mathcal{G}$ (see [35] for proof).

Claim 3. *Suppose $\alpha, \beta \in [0,1]^n$ are such that each has at most k positive coordinates, $y' := \tilde{y} - \alpha + \beta \in \mathcal{U}$, and y' satisfies all knapsacks. Then $F(y') \le (1+\epsilon) \cdot F(\tilde{y}) + \frac{1}{4n^2} f_{max}$.*

For any $x, y \in \mathbb{R}^n$, we define $x \vee y$ (meet operator) and $x \wedge y$ (join operator) by $(x \vee y)_j := \max(x_j, y_j)$ and $(x \wedge y)_j := \min(x_j, y_j)$ for all $j \in [n]$.

Lemma 5. *For local optimal $\tilde{y} \in \mathcal{U} \cap \mathcal{G}^n$ and any $\tilde{x} \in \mathcal{U}$ satisfying the knapsack constraints, we have $(2 + 2n \cdot \epsilon) \cdot F(\tilde{y}) \ge F(\tilde{y} \wedge \tilde{x}) + F(\tilde{y} \vee \tilde{x}) - \frac{1}{2n} \cdot f_{max}$.*

Proof. For the sake of analysis, we add the following k *dummy elements* to the ground-set: for each knapsack $s \in [k]$, element d_s has weight 1 in knapsack s and zero in all other knapsacks, and upper-bound of 1. The function f remains the same: it only depends on the original variables V. Let $W := V \cup \{d_s\}_{s=1}^k$ denote the new ground-set. Using the dummy elements, any fractional feasible solution can be augmented to another of the same F-value, while satisfying all knapsacks at *equality*. We augment $\tilde{y}$ and $\tilde{x}$ using dummy elements to obtain y and x, that both satisfy all knapsacks at equality. Clearly $F(y) = F(\tilde{y})$, $F(y \wedge x) = F(\tilde{y} \wedge \tilde{x})$ and $F(y \vee x) = F(\tilde{y} \vee \tilde{x})$. Let $y' = y - (y \wedge x)$ and $x' = x - (y \wedge x)$. Note that for all $s \in [k]$, $w^s \cdot y' = w^s \cdot x'$ and let $c_s = w^s \cdot x'$. We will decompose y' and x' into an equal number of terms as $y' = \sum_t \alpha_t$ and $x' = \sum_t \beta_t$ such that the αs and βs have small support, and $w^s \cdot \alpha_t = w^s \cdot \beta_t$ for all t and $s \in [k]$.

1. Initialize $t \leftarrow 1$, $\gamma \leftarrow 1$, $x'' \leftarrow x'$, $y'' \leftarrow y'$.

Input: Knapsack weights $\{w^s\}_{s=1}^k$, parameter η, and value oracle to submodular function f.

1. Set $c \leftarrow \frac{16}{\eta}$, $\delta \leftarrow \frac{1}{8c^3k^4}$ and $\epsilon \leftarrow \frac{1}{ck}$.
2. Define an element $e \in V$ as *heavy* if $w^s(e) \geq \delta$ for some knapsack $s \in [k]$. All other elements are called *light*.
3. Enumerate over all feasible (under the knapsacks) sets consisting of up to k/δ heavy elements, to obtain T_1 having maximum f-value.
4. Restricting to only light elements, solve the fractional relaxation (problem (11)) with all upper-bounds one, using the algorithm in Section 3.2.2 (with parameter $\eta/2$). Let x denote the fractional solution found.
5. Obtain random set R as follows: Pick each light element $e \in V$ into R independently with probability $(1-\epsilon)x_e$.
6. If R satisfies all knapsacks, set $T_2 \leftarrow R$; otherwise set $T_2 \leftarrow \emptyset$.
7. **Output** $\arg\max\{f(T_1), f(T_2)\}$.

Figure 5: Approximation algorithm for submodular maximization under k knapsacks.

2. While $\gamma > 0$, do:
   (a) Consider $LP_x := \{z \geq 0 \ : \ z \cdot w^s = c_s, \ \forall s \in [k]\}$ where the variables are restricted to indices $i \in [n]$ with $x''_i > 0$. Similarly $LP_y := \{z \geq 0 \ : \ z \cdot w^s = c_s, \ \forall s \in [k]\}$ where the variables are restricted to indices $i \in [n]$ with $y''_i > 0$. Let $u \in LP_x$ and $v \in LP_y$ be extreme points.
   (b) Set $\delta_1 = \max\{\chi \ : \ \chi \cdot u \leq x''\}$ $\delta_2 = \max\{\chi \ : \ \chi \cdot v \leq y''\}$, and $\delta = \min\{\delta_1, \delta_2\}$.
   (c) Set $\beta_t \leftarrow \delta \cdot u$, $\alpha_t \leftarrow \delta \cdot v$, $\gamma \leftarrow \gamma - \delta$, $x'' \leftarrow x'' - \beta_t$, and $y'' \leftarrow y'' - \alpha_t$.
   (d) Set $t \leftarrow t+1$.

We first show that this procedure is well-defined. A simple induction shows that at the start of every iteration, $w^s \cdot x'' = w^s \cdot y'' = \gamma \cdot c_s$ for all $s \in [k]$. Thus in step 2a, LP_x (resp. LP_y) is non-empty: x''/γ (resp. y''/γ) is a feasible solution. From the definition of LP_x and LP_y it also follows that $\delta > 0$ in step 2b and at least one coordinate of x'' or y'' is zeroed out in step 2c. This implies that the decomposition procedure terminates in $r \leq 2n$ steps.

At the end of the procedure, we have decompositions $x' = \sum_{t=1}^r \beta_t$ and $y' = \sum_{t=1}^r \alpha_t$. Furthermore, each α_t (resp. β_t) corresponds to an *extreme point* of LP_y (resp. LP_x) in some iteration: hence the number of positive components in any of $\{\alpha_t, \beta_t\}_{t=1}^r$ is at most k, and all these values are rational. Finally note that for all $t \in [r]$, $w^s \cdot \alpha_t = w^s \cdot \beta_t$ for all knapsacks $s \in [k]$. Note that x, y, x', y', αs and βs are vectors over W.

For each $t \in [r]$, define $\tilde{\alpha}_t$ (resp. $\tilde{\beta}_t$) to be α_t (resp. β_t) restricted to the original variables V. From the above decomposition, it is clear that $\tilde{y} = \tilde{y} \wedge \tilde{x} + \sum_{t=1}^r \tilde{\alpha}_t$ and $\tilde{x} = \tilde{y} \wedge \tilde{x} + \sum_{t=1}^r \tilde{\beta}_t$, where the $\tilde{\alpha}$s and $\tilde{\beta}$s are non-negative. Thus for any $t \in [r]$, $\tilde{y} - \tilde{\alpha}_t + \tilde{\beta}_t \in \mathcal{U}$. Furthermore, for any $t \in [r]$, $y - \alpha_t + \beta_t \geq 0$ coordinate-wise and satisfies all knapsacks at equality; hence dropping the dummy variables, we obtain that $\tilde{y} - \tilde{\alpha}_t + \tilde{\beta}_t$ satisfies all knapsacks (perhaps not at equality). Now observe that Claim 3 applies to $\tilde{y}$, $\tilde{\alpha}_t$ and $\tilde{\beta}_t$ (for any $t \in [r]$) because each of $\tilde{\alpha}_t, \tilde{\beta}_t$ has support-size at most k, and (as argued above) $\tilde{y} - \tilde{\alpha}_t + \tilde{\beta}_t \in \mathcal{U}$ and satisfies all knapsacks. Thus:

$$F(\tilde{y} - \tilde{\alpha}_t + \tilde{\beta}_t) \leq (1+\epsilon) \cdot F(\tilde{y}) + \frac{f_{max}}{4n^2} \qquad \forall t \in [r] \ . \quad (12)$$

Let $M \in \mathbb{Z}_+$ be large enough so that $M\tilde{\alpha}_t$ and $M\tilde{\beta}_t$ are integral for all $t \in [r]$. In the rest of the proof, we consider a scaled ground-set U containing M copies of each element in V. We define function $g : 2^U \to \mathbb{R}_+$ as $g(\cup_{i \in V} T_i) = F(\cdots, \frac{|T_i|}{M}, \cdots)$ where each T_i consists of copies of element $i \in V$. Lemma 3 implies that g is submodular. Corresponding to $\tilde{y}$ we have a set $P = \cup_{i \in V} P_i$ consisting of the first $|P_i| = M \cdot \tilde{y}_i$ copies of each element $i \in V$. Similarly, $\tilde{x}$ corresponds to set $Q = \cup_{i \in V} Q_i$ consisting of the first $|Q_i| = M \cdot \tilde{x}_i$ copies of each element $i \in V$. Hence $P \cap Q$ (resp. $P \cup Q$) corresponds to $\tilde{x} \wedge \tilde{y}$ (resp. $\tilde{x} \vee \tilde{y}$) scaled by M. Again, $P \setminus Q$ (resp. $Q \setminus P$) corresponds to scaled version of $\tilde{y} - (\tilde{y} \wedge \tilde{x})$ (resp. $\tilde{x} - (\tilde{y} \wedge \tilde{x})$). The decomposition $\tilde{y} = (\tilde{y} \wedge \tilde{x}) + \sum_{t=1}^r \tilde{\alpha}_t$ from above suggests *disjoint* sets $\{A_t\}_{t=1}^r$ such that $\cup_t A_t = P \setminus Q$; i.e. each A_t corresponds to $\tilde{\alpha}_t$ scaled by M. Similarly there are *disjoint* sets $\{B_t\}_{t=1}^r$ such that $\cup_t B_t = Q \setminus P$. Observe also that $g((P \setminus A_t) \cup B_t) = F(\tilde{y} - \tilde{\alpha}_t + \tilde{\beta}_t)$, so (12) corresponds to:

$$g((P \setminus A_t) \cup B_t) \leq (1+\epsilon) \cdot g(P) + \frac{f_{max}}{4n^2} \qquad \forall t \in [r] \ . \quad (13)$$

Adding all these r inequalities to $g(P) = g(P)$, we obtain $(r + \epsilon \cdot r + 1) g(P) + \frac{r}{4n^2} f_{max} \geq g(P) + \sum_{t=1}^r g((P \setminus A_t) \cup B_t)$. Using submodularity of g and the disjointness of families $\{A_t\}_{t=1}^r$ and $\{B_t\}_{t=1}^r$, this simplifies to $(r + \epsilon \cdot r + 1) \cdot g(P) + \frac{r}{4n^2} f_{max} \geq (r-1) \cdot g(P) + g(P \cup Q) + g(P \cap Q)$. Hence $(2 + \epsilon \cdot r) \cdot g(P) \geq g(P \cup Q) + g(P \cap Q) - \frac{r}{4n^2} f_{max}$. This implies the lemma because $r \leq 2n$. $\square$

3.2.2 *Approximation algorithm for Problem* (11) *with all upper-bounds one*

This algorithm is similar to the way Algorithm A in Section 2 uses the local search Procedure B. The algorithm takes as input a parameter δ, and proceeds as follows.

1. Set T_0 to be one of $\emptyset, \{1\}, \{2\}, \cdots, \{n\}$ having maximum f-value.

2. Choose $\epsilon \leftarrow \delta/8n$ as the parameter for local search (Figure 4).
3. Run the local search (Figure 4) with all upper bounds at 1, to get local optimum y_1.
4. Run the local search in Figure 4 again, with upper-bound $1 - y_1(i)$ for each $i \in [n]$, to obtain local optimum y_2.
5. **Output** $\arg\max\{f(T_0), F(y_1), F(y_2)\}$.

The proof of the following theorem can be found in the full version [35].

THEOREM 4. *For any $\frac{1}{n} \ll \delta < \frac{1}{4}$, the above algorithm is a $(\frac{1}{4} - \delta)$-approximation algorithm for the fractional knapsack problem* (11) *when upper bounds $u_i = 1$, $\forall i \in V$.*

3.3 Rounding the fractional knapsack

Figure 5 describes our algorithm for submodular maximization subject to k knapsack constraints (problem (10)). The following theorem is proved in the full version [35].

THEOREM 5. *For any constant $\eta > 0$, the algorithm in Figure 5 is a $(\frac{1}{5} - \eta)$-approximation algorithm for maximizing non-negative submodular functions over k knapsack constraints.*

Note that the running time of the algorithm Figure 5 is polynomial for any fixed k: the enumeration in Step 3 takes $n^{O(k/\delta)}$ time, and the algorithm for light elements from Section 3.2.2 is also polynomial-time.

Acknowledgment: The proof of Lemma 1 presented in this paper is due to Jan Vondrák. Our original proof [35] was more complicated — we thank Jan for letting us present this simplified proof.

4. REFERENCES

[1] A. Ageev, R. Hassin and M. Sviridenko, An 0.5-approximation algorithm for MAX DICUT with given sizes of parts. SIAM J. Discrete Math. 14 (2001), no. 2, 246–255 (electronic).

[2] A. Ageev and M. Sviridenko. An 0.828 Approximation algorithm for the uncapacitated facility location problem, *Discrete Applied Mathematics* 93(2-3): 149–156 (1999).

[3] A. Ageev and M. Sviridenko, Pipage rounding: a new method of constructing algorithms with proven performance guarantee. J. Comb. Optim. 8 (2004), no. 3, 307–328.

[4] P. Alimonti. Non-oblivious local search for MAX 2-CCSP with application to MAX DICUT, In *Proceedings of the 23rd International Workshop on Graph-theoretic Concepts in Computer Science, 1997.*

[5] K. M. Anstreicher, M. Fampa, J. Lee and J. Williams. Using continuous nonlinear relaxations to solve constrained maximum-entropy sampling problems. *Mathematical Programming, Series A*, 85:221-240, 1999.

[6] S. Burer and J. Lee. Solving maximum-entropy sampling problems using factored masks. *Mathematical Programming*, Volume 109, Numbers 2-3, 263-281, 2007

[7] G. Calinescu, C. Chekuri, M. Pál and J. Vondrák. Maximizing a monotone submodular function under a matroid constraint, *IPCO 2007.*

[8] V. Cherenin. Solving some combinatorial problems of optimal planning by the method of successive calculations, *Proc. of the Conference of Experiences and Perspectives of the Applications of Mathematical Methods and Electronic Computers in Planning* (in Russian), Mimeograph, Novosibirsk (1962).

[9] G. Cornuéjols, M. Fischer and G. Nemhauser. Location of bank accounts to optimize oat: An analytic study of exact and approximation algorithms, *Management Science*, 23 (1977), 789–810.

[10] G. Cornuéjols, M. Fischer and G. Nemhauser. On the uncapacitated location problem, *Annals of Discrete Math* 1 (1977), 163–178.

[11] G. P. Cornuéjols, G. L. Nemhauser and L. A. Wolsey. The uncapacitated facility location problem. In *Discrete Location Theory* (1990), 119–171.

[12] J. Edmonds. Matroids, submodular functions, and certain polyhedra, *Combinatorial Structures and Their Applications* (1970), 69–87.

[13] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of ACM* 45 (1998), 634–652.

[14] U. Feige and M. X. Goemans. Approximating the value of two-prover systems, with applications to MAX-2SAT and MAX-DICUT. *Proc. of the 3rd Israel Symposium on Theory and Computing Systems*, Tel Aviv (1995), 182–189.

[15] U. Feige, V. Mirrokni and J. Vondrák. Maximizing non-monotone submodular functions, *FOCS 2007.*

[16] A. Frank. Matroids and submodular functions, *Annotated Biblographies in Combinatorial Optimization* (1997), 65–80.

[17] S. Fujishige. Canonical decompositions of symmetric submodular systems, *Discrete Applied Mathematics* 5 (1983), 175–190.

[18] M. Goemans, N. Harvey, S. Iwata, V. Mirrokni. Approximating submodular functions everywhere. In *SODA 2009.*

[19] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *Journal of ACM* 42 (1995), 1115–1145.

[20] B. Goldengorin, G. Sierksma, G. Tijsssen and M. Tso. The data correcting algorithm for the minimization of supermodular functions, *Management Science*, 45:11 (1999), 1539–1551.

[21] B. Goldengorin, G. Tijsssen and M. Tso. The maximization of submodular Functions: Old and new proofs for the correctness of the dichotomy algorithm, *SOM Report*, University of Groningen (1999).

[22] E. Halperin and U. Zwick. Combinatorial approximation algorithms for the maximum directed cut problem. *Proc. of 12th SODA* (2001), 1–7.

[23] J. Hartline, V. Mirrokni and M. Sundararajan. Optimal marketing strategies over social networks, World Wide Web Conference (WWW), 2008, 189–198.

[24] E. Hazan, S. Safra and O. Schwartz. On the complexity of approximating k-set packing. Computational Complexity, 15(1), 20–39, 2006.

[25] J. Håstad. Some optimal inapproximability results. *Journal of ACM* 48 (2001): 798–869.

[26] S. Iwata, L. Fleischer and S. Fujishige. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions, *Journal of ACM* 48:4 (2001), 761–777.

[27] V. R. Khachaturov, Mathematical Methods of Regional Programming, Nauka, Moscow (in Russian), 1989.

[28] A. Kulik, H. Shachnai and T. Tamir. Maximizing submodular functions subject to multiple linear constraints. *Proc. of SODA, 2009*.

[29] C.-W. Ko, J. Lee and M. Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research* 43(4):684-691, 1995.

[30] A. Krause, A. Singh and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies, *Journal of Machine Learning Research* 9 (2008) 235–284.

[31] H. Lee, G. Nemhauser and Y. Wang. Maximizing a submodular function by integer programming: Polyhedral results for the quadratic case, *European Journal of Operational Research* 94, 154–166.

[32] J. Lee. Maximum entropy sampling. In: A.H. El-Shaarawi and W.W. Piegorsch, editors, "Encyclopedia of Environmetrics". Wiley, 2001.

[33] J. Lee. Semidefinite programming in experimental design. In: H. Wolkowicz, R. Saigal and L. Vandenberghe, editors, "Handbook of Semidefinite Programming", International Series in Operations Research and Management Science, Vol. 27, Kluwer, 2000.

[34] J. Lee. Constrained maximum-entropy sampling. *Operations Research*, 46:655-664, 1998.

[35] J. Lee, V. Mirrokni, V. Nagarajan and M. Sviridenko. Maximizing Non-Monotone Submodular Functions under Matroid and Knapsack Constraints. IBM Research Report RC24679, 2008.

[36] L. Lovász. Submodular functions and convexity. In: A. Bachem et. al., eds, "Mathematical Programmming: The State of the Art, " 235–257.

[37] V. Mirrokni, J. Vondrák and M. Schapira. Tight Information-Theoretic Lower Bounds for Welfare Maximization in Combinatorial Auctions. *Proc. of EC, 2008*, 70-77.

[38] R. Motwani and P. Raghavan. Randomized Algorithms, Cambridge University Press, 1995.

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

[40] G. L. Nemhauser, L. A. Wolsey and M. L. Fisher. An analysis of approximations for maximizing submodular set functions II. *Mathematical Programming Study* 8 (1978), 73–87.

[41] J. G. Oxley, "Matroid theory," Oxford Science Publications. The Clarendon Press, Oxford University Press, New York, 1992.

[42] M. Queyranne. A combinatorial algorithm for minimizing symmetric submodular functions, ACM-SIAM *Symposium on Discrete Algorithms* (1995), 98–101.

[43] J. Reichel and M. Skutella, Evolutionary algorithms and matroid optimization problems, in Proceedings of the 9th Genetic and Evolutionary Computation Conference (GECCO'07), 947–954, 2007.

[44] T. Robertazzi and S. Schwartz, An accelated sequential algorithm for producing D-optimal designs. *SIAM Journal on Scientific and Statistical Computing* 10, 341–359.

[45] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time, *Journal of Combinatorial Theory, Series B* 80 (2000), 346–355.

[46] A. Schrijver. "Combinatorial Optimization," Volumes A-C. Springer-Verlag, Berlin, 2003.

[47] A. Schulz, N. Uhan, Encouraging Cooperation in Sharing Supermodular Costs, APPROX-RANDOM 2007: 271–285

[48] M. C. Shewry, H. P. Wynn, Maximum entropy sampling. J. Appl. Stat. 14 (1987), 165–170.

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

[50] Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *FOCS 2008*.

[51] J. Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC, 2008*.

[52] J. Vondrák, Personal communication, 2008.