ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230534

Improved Analysis of Greedy Algorithm on k-Submodular Knapsack

Zhongzheng Tang[a] **and Chenhao Wang**[b,c;*]

[a]School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China
[b]Beijing Normal University, Zhuhai 519087, Guangdong, China
[c]BNU-HKBU United International College, Zhuhai 519087, Guangdong, China

Abstract. A k-submodular function is a generalization of submodular functions that takes k disjoint subsets as input and outputs a real value. It captures many problems in combinatorial optimization and machine leaning such as influence maximization, sensor placement, feature selection, etc. In this paper, we consider the monotone k-submodular maximization problem under a knapsack constraint, and explore the performance guarantee of a greedy-based algorithm: enumerating all size-2 solutions and extending every singleton solution greedily; the best outcome is returned. We provide a novel analysis framework and prove that this algorithm achieves an approximation ratio of at least 0.328. This is the best-known result of combinatorial algorithms on k-submodular knapsack maximization.

In addition, within the framework, we can further improve the approximation ratio to a value approaching $\frac{1}{3}$ with any desirable accuracy, by enumerating sufficiently large base solutions. The results can even be extended to non-monotone k-submodular functions.

1 Introduction

Let V be a set of n items and 2^V be the family of all subsets in V. A submodular function is a set function $f : 2^V \to \mathbb{R}$ such that for all pairs $X, Y \in 2^V$,

$$f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y).$$

The submodularity is characterized by the *diminishing return property*, $f(X \cup \{a\}) - f(X) \geq f(Y \cup \{a\}) - f(Y)$ for any $X \subseteq Y$ and $a \in V \setminus Y$. A k-submodular function generalizes a submodular function in a natural way that captures interactions among k dimensions. While a submodular function takes a single subset of V as input, a k-submodular function considers k disjoint subsets of V, and exhibits the diminishing return property.

Let $(k+1)^V := \{(X_1, \ldots, X_k) \mid X_i \subseteq V \; \forall i \in [k], X_i \cap X_j = \varnothing \; \forall i \neq j\}$ be the family of k disjoint sets, where $[k] := \{1, \ldots, k\}$. A function $f : (k+1)^V \to \mathbb{R}$ is *k-submodular* if and only if for every k-tuples $\mathbf{x} = (X_1, \ldots, X_k)$ and $\mathbf{y} = (Y_1, \ldots, Y_k)$ in $(k+1)^V$,

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \sqcup \mathbf{y}) + f(\mathbf{x} \sqcap \mathbf{y}), \quad \text{where}$$

$$\mathbf{x} \sqcup \mathbf{y} := \Big(X_1 \cup Y_1 \setminus (\bigcup_{i \neq 1} X_i \cup Y_i), \ldots, X_k \cup Y_k \setminus (\bigcup_{i \neq k} X_i \cup Y_i)\Big),$$

$$\mathbf{x} \sqcap \mathbf{y} := (X_1 \cap Y_1, \ldots, X_k \cap Y_k)\,.$$

For a k-tuple $\mathbf{x} = (X_1, \ldots, X_k) \in (k+1)^V$, we define its size by $|\mathbf{x}| = |\cup_{i \in [k]} X_i|$. We say that $f : (k+1)^V \to \mathbb{R}$ is *monotone*, if $f(\mathbf{x}) \leq f(\mathbf{y})$ holds for any $\mathbf{x}$ and $\mathbf{y}$ with $X_i \subseteq Y_i$ for all $i \in [k]$.

Since Huber and Kolmogorov [5] proposed the notion of k-submodularity one decade ago, there have been increased theoretical and algorithmic interests in the study of k-submodular functions, as various problems in combinatorial optimization and machine learning can be formulated as k-submodular function maximization. The application scenarios include influence maximization [12, 18], sensor placement [9, 25], document summarization [7] and feature selection [25], etc.

- **Influence maximization.** Given k topics in a social network, we want to select several influential people for each topic to start its spread, in order to maximize the population influenced by at least one topic. This objective function is k-submodular. It models viral marketing or product recommendation in which a company wants to spread an advertisement campaign about k products to users via social networks or the Internet.
- **Feature selection.** Given k regression targets on a set of features, feature selection partitions the features into $k+1$ subsets so that one feature can be used in at most one regression target or none of them, which has important implications in data privacy and data rights. The feature quality measure can be modeled as a k-submodular function.
- **Sensor placement.** Sensor networks, enabled by IoT, provide real-time monitoring and control to systems such as smart cities and smart homes, and often call for multiple types of sensors. The goal of sensor placement with k types of sensors is to maximize the information obtained from the sensors, which is usually modeled by a k-submodular function [9, 25].

The problem of maximizing a k-submodular function is known to be NP-hard, since it is a generalization of the NP-hard submodular maximization problem. Extensive research has been dedicated to developing efficient algorithms with desirable approximation ratios for the problem under different constraints, for example, cardinality constraints [2, 9, 18], matroid constraints [13, 15], knapsack constraints [17, 12], and the unconstrained setting [19, 6, 14].

In this paper, we study the k-submodular maximization problem under a knapsack constraint, called *k-submodular knapsack maximization* (kSKM). Each item $a \in V$ has a cost $c(a)$, and the total

* Corresponding Author. Email: chenhwang@bnu.edu.cn.

cost of the items selected in the solution cannot exceed a given budget $B \in \mathbb{R}_+$. Given the NP-hardness of the kSKM, we are interested in approximation algorithms. An algorithm has an approximation ratio of $\alpha \leq 1$, if for any problem instance, the function value returned by the algorithm is no less than α times the optimal value.

Our contributions.

In this paper, we present a 0.328-approximation algorithm for the monotone kSKM.

For the kSKM, there are two natural heuristics, $\mathbf{Enu}_2$ and $\mathbf{Greedy}_1$. The former heuristic enumerates all size-2 solutions and returns the best one $\arg\max_{\mathbf{x}:|\mathbf{x}|=2} f(\mathbf{x})$. The latter heuristic first enumerates all singleton (size-1) solution, and then extends each of them *greedily*, that is, maintaining a feasible solution, in each iteration it adds to the current solution a feasible item a in dimension d that maximizes the *marginal density* (the marginal value divided by cost $c(a)$). Both heuristics do not guarantee any constant approximation ratio.

We consider the combination of the two heuristics, namely, $\mathbf{Enu}_2 + \mathbf{Greedy}_1$, which returns the better solution between them. We provide a novel analysis framework, and prove that when the k-submodular function is monotone, this algorithm has an approximation ratio 0.328 for the kSKM with $O(n^3k^2)$ query complexity. This improves the $\frac{1}{2}(1-e^{-1}) \approx 0.316$ approximation by Tang *et al.* [17], and is the best-known result of combinatorial algorithms on the monotone kSKM. Compared with extension-based algorithms (which can achieve an approximation ratio $\frac{1}{2}-\epsilon$ [18], using continuous optimization techniques), combinatorial algorithms have natural advantages such as higher running efficiency, good interpretability and usually more intuitive, and thus receive more attention in the community.

Our analysis relies on an observation for the unconstrained maximization problem (Lemma 3). It bounds the optimal value in terms of the value of partial greedy solution (the incomplete solution in iterations of greedy procedure) and the marginal gains, and implies that a partial greedy solution with *large enough cost* is a good approximation (Lemma 4). Then our analysis works as follows: if there is a partial greedy solution with large enough cost before it discards any item in an optimal solution OPT, then it is good enough. Otherwise, we can identify one or two large-cost items in OPT. And if these large-cost items are good enough, the enumeration procedure will find them; otherwise, the rest of the items in OPT are good enough but have a small cost, so that the greedy procedure can easily approximate them.

An immediate question is whether the approximation ratio can be further improved by enumerating more items? We generalize the $\mathbf{Enu}_2$+$\mathbf{Greedy}_1$ algorithm to $\mathbf{Enu}_p$+$\mathbf{Greedy}_{p-1}$, which enumerates all size-p solutions and greedily extends all size-$(p-1)$ solutions. We show that it can achieve an approximation ratio sufficiently close to $\frac{1}{3}$ when p increases. Further, our analysis framework even applies to non-monotone cases. For the non-monotone kSKM, $\mathbf{Enu}_2$+$\mathbf{Greedy}_1$ is 0.249-approximation, and $\mathbf{Enu}_p + \mathbf{Greedy}_{p-1}$ has an approximation approaching $\frac{1}{4}$.

Organizations. In Section 2 the model and preliminaries are presented. In Section 3 we give a useful lemma for the unconstrained maximization problem. In Section 4 we analyze the approximation of $\mathbf{Enu}_2$+$\mathbf{Greedy}_1$ for monotone kSKM. In Section 5 we extend the results to more general algorithms and non-monotone functions. In Section 6 we discuss the comparison with the work [17] and future directions.

Related work.

One decade ago, k-submodular functions were introduced by Huber and Kolmogorov [5] to express submodularity on choosing k disjoint sets of elements instead of a single set. Subsequently, k-submodular functions have become a popular research direction [3, 4, 8, 14, 21], especially the problem of maximizing k-submodular functions.

For the k-submodular maximization under a knapsack constraint (i.e., kSKM), Tang *et al.* [17] were the first to consider it in the community. When the function is monotone, they claimed that the algorithm that greedily extends all feasible size-2 solution is $\frac{1}{2}(1-\frac{1}{e})$-approximation, with a flaw in their proof (as discussed in Section 6). Chen *et al.* [1] considered the **Greedy+Singleton** algorithm, which returns the better one between the greedy solution and the best singleton solution. This algorithm has been well studied in knapsack problems with linear and submodular objective functions due to its simplicity and efficiency, and Chen *et al.* proved an approximation ratio $\frac{1}{4}(1-\frac{1}{e})$ for the kSKM. Pham *et al.* [12] proposed streaming algorithms with approximation ratios $\frac{1}{4}-\epsilon$ and $\frac{1}{5}-\epsilon$ for the monotone and non-monotone cases, respectively, which requires $O(\frac{n}{\epsilon}\log n)$ queries of the k-submodular function. Other works related to kSKM include [11, 16, 22, 23, 24]. While all the above mentioned are combinatorial algorithms, the best known result for kSKM comes from extension-based methods. Wang and Zhou [18] presented an algorithm based on multilinear extension that first extends a k-submodular function to a continuous space and then rounds the fractional solution. It achieves an asymptotically optimal ratio of $\frac{1}{2}-\epsilon$.

In addition to the knapsack constraint, the problem of maximizing k-submodular functions subject to non-constraint or other types of constraints has also received attention from researchers. For unconstrained *monotone* k-submodular maximization, Ward and Živnỳ [19] proved that a greedy algorithm is $\frac{1}{2}$-approximation, and later, Iwata *et al.* [6] proposed a randomized $\frac{k}{2k-1}$-approximation algorithm, and showed that the ratio is asymptotically tight. Oshima [10] provided a $\frac{k^2+1}{2k^2+1}$-approximation for unconstrained *non-monotone* maximization. For monotone k-submodular maximization under a total size constraint (i.e., at most a given number of items can be selected), Ohsaka and Yoshida [9] proposed a $\frac{1}{2}$-approximation algorithm, and for the problem under individual size constraints (i.e., in each dimension at most a given number of items can be selected), they gave a $\frac{1}{3}$-approximation algorithm. Under a matroid constraint, Sakaue [13] shown that fully greedy algorithm is $\frac{1}{2}$-approximation for the monotone case, and Sun *et al.* [15] gave a $\frac{1}{3}$-approximation algorithm for the non-monotone case.

2 Preliminaries

We introduce more characteristics and notations of k-submodular functions. Recall that V is the ground set and f is a function mapping every k-tuple in $(k+1)^V$ to a real value.

Every k-tuple $\mathbf{x} = (X_1, \ldots, X_k) \in (k+1)^V$ uniquely corresponds to a set $S = \{(a,d) \mid a \in X_d,\ d \in [k]\}$ that consists of *item-dimension pairs*. That is, an item-dimension pair (a,d) belongs to S (called a *solution*) if and only if $a \in X_d$ in $\mathbf{x}$. From now on, for notational convenience, we write $\mathbf{x}$ and its corresponding solution S interchangeably. For any solution $S \in (k+1)^V$, we define $U(S) := \{a \in V \mid \exists d \in [k]\ \ (a,d) \in S\}$ to be the set of items included, and the *size* is $|S| = |U(S)|$. For two solutions $S, S' \in (k+1)^V$, $S \subseteq S'$ means that all items in $U(S)$ also belong to $U(S')$ and the dimensions are consistent. In this paper, let f be a non-negative k-submodular function, and we further assume w.l.o.g.

that $f(\varnothing) = 0$.

The marginal gain of adding an item-dimension pair (a, d) to S is

$$\Delta_{a,d}(S) = f(S \cup \{(a, d)\}) - f(S),$$

and the marginal density is $\frac{\Delta_{a,d}(S)}{c(a)}$. A k-submodular function f clearly satisfies the *orthant submodularity*

$$\Delta_{a,d}(S) \geq \Delta_{a,d}(S'),$$

$$\forall S, S' \in (k+1)^V \text{ with } S \subseteq S', a \notin U(S'), d \in [k],$$

and the *pairwise monotonicity*

$$\Delta_{a,d_1}(S) + \Delta_{a,d_2}(S) \geq 0,$$

$$\forall S \in (k+1)^V \text{ with } a \notin U(S), d_1, d_2 \in [k], d_1 \neq d_2.$$

Ward and Živnỳ [19] prove this observation and further show that the converse is also true.

Lemma 1 ([19]). *A function $f : (k+1)^V \to \mathbb{R}$ is k-submodular if and only if f is orthant submodular and pairwise monotone.*

In the kSKM problem, each item $a \in V$ has a non-negative cost $c(a)$, and the cost of solution S is $c(S) = \sum_{a \in U(S)} c(a)$. The goal is to find a solution that maximizes the function value within the given budget $B \in \mathbb{R}_+$, i.e., $\max_{S \in (k+1)^V : c(S) \leq B} f(S)$. We point out the following lemma that will repeatedly and implicitly used in our analysis.

Lemma 2 ([17]). *For any solutions S, S' with $S \subseteq S'$, we have*

$$f(S') - f(S) \leq \sum_{(a,d) \in S' \setminus S} \Delta_{a,d}(S).$$

3 A useful lemma

Before looking at the kSKM problem, let us first consider unconstrained maximization problem. Precisely, given a set of items $V' = \{e_1, e_2, \ldots, e_m\}$, we want to find the maximum function value $\max_{S \in (k+1)^{V'}} f(S)$, without any constraint. Due to the monotonicity of the function, an optimal solution for the unconstrained maximization will select all items. (This statement is even true for non-monotone functions, because a k-submodular function must be pairwise monotone, indicating that there always exists a dimension so that the marginal gain of selecting an item is non-negative.) Thus the only thing to do is to assign the items in V' to the k dimensions. A natural heuristic is the **Greedy** (Algorithm 1), which considers items one by one in an arbitrary order, and assigns each item to the best dimension each time, that is, the dimension that brings the maximum marginal gain.

Algorithm 1 Greedy

Input: Set V', k-submodular function f
Output: A solution $S \in (k+1)^V$

```
S ← ∅
for each item a ∈ V' do
    d_a ← arg max_{d∈[k]} Δ_{a,d}(S)
    S ← S ∪ {(a, d_a)}
end for
return S
```

Let $T = \{(e_1, d_1^*), \ldots, (e_m, d_m^*)\}$ be an optimal solution that maximizes the function value over V'. Assume without loss of generality that the items are considered by **Greedy** in the order of $e_1, e_2, \ldots, e_m$, and denote the returned greedy solution by $S = \{(e_1, d_1), \ldots, (e_m, d_m)\}$.

For $j = 0, 1, \ldots, m$, define

$$S_j = \{(e_1, d_1), \ldots, (e_j, d_j)\} \text{ and} \tag{3.1}$$

$$T_j = \big(T \setminus \{(e_1, d_1^*), \ldots, (e_j, d_j^*)\}\big) \cup S_j. \tag{3.2}$$

That is, S_j is the first j item-dimension pairs in the greedy solution S (called a partial greedy solution), and T_j is obtained from the optimal solution T by replacing the first j item-dimension pairs with S_j. Clearly, $S_0 = \varnothing, S_m = S, T_0 = T$ and $T_m = S$.

The following useful lemma says that the optimal value $f(T)$ is no more than twice the value of any partial greedy solution S_t, plus the total marginal gain of other item-dimension pairs in the optimal solution. This conclusion is firstly noticed by Ward and Živnỳ (implicitly in Theorem 5.1 [19]) and formalized by Xiao *et al.* [22]. For completeness, we write down the proof in our notations.

Lemma 3. *If f is monotone, we have $f(T) \leq 2f(S_t) + \sum_{(a,d) \in T_t \setminus S_t} \Delta_{a,d}(S_t)$, for $t = 0, 1, \ldots, m$.*

Proof. Define an intermediate solution $P_j := T_j \setminus (e_{j+1}, d_{j+1}^*) = T_{j+1} \setminus (e_{j+1}, d_{j+1})$, for $j = 0, \ldots, t-1$. That is, P_j consists of $m-1$ items (excluding e_{j+1}), where the dimension of items $e_1, \ldots, e_j$ align with S, and the dimensions of other items align with T. It is easy to see

$$f(T_j) = f(P_j) + \Delta_{e_{j+1}, d_{j+1}^*}(P_j),$$

$$f(T_{j+1}) = f(P_j) + \Delta_{e_{j+1}, d_{j+1}}(P_j).$$

Then the difference between $f(T_j)$ and $f(T_{j+1})$ is

$$\begin{aligned} f(T_j) - f(T_{j+1}) &= \Delta_{e_{j+1}, d_{j+1}^*}(P_j) - \Delta_{e_{j+1}, d_{j+1}}(P_j) \\ &\leq \Delta_{e_{j+1}, d_{j+1}^*}(P_j) && (3.3) \\ &\leq \Delta_{e_{j+1}, d_{j+1}^*}(S_j) && (3.4) \\ &\leq \Delta_{e_{j+1}, d_{j+1}}(S_j) && (3.5) \\ &= f(S_{j+1}) - f(S_j). \end{aligned}$$

Eq. (3.3) comes from the monotonicity of f, and Eq. (3.4) comes from the fact of $S_j \subseteq P_j$. Eq. (3.5) follows from the fact that **Greedy** always assign the item considered to the dimension with maximum marginal gain, and (e_{j+1}, d_{j+1}) is the $(j+1)$-th pair added. Summing this inequality from $j = 0$ to $t-1$, we obtain

$$f(T_0) - f(T_t) \leq f(S_t) - f(S_0) = f(S_t).$$

Since $S_t \subseteq T_t$ and Lemma 2, we have

$$f(T) \leq f(S_t) + f(T_t) \leq 2f(S_t) + \sum_{(a,d) \in T_t \setminus S_t} \Delta_{a,d}(S_t).$$

□

This lemma bounds the optimal value in terms of the partial greedy solution's value $f(S_t)$ and the marginal gains. Our analysis for kSKM in the next section will heavily rely on this bound.

4 Algorithm and Approximation Analysis

In this section we consider the kSKM problem under a budget constraint B, and let $OPT \in (k+1)^V$ be the optimal solution. For convenience of analysis, we assume that B is an integer, and all items have integer cost. This assumption can be easily relaxed.

We investigate Algorithm 2. It compares all size-2 solutions and all greedy solutions extended from a singleton, and returns the winner as outcome. In Line 1, the algorithm finds the best size-2 solution, with $O(n^2k^2)$ queries of the function. In Line 2-14, it exhaustively enumerates (iterates) over all singleton $Y \subseteq (k+1)^V$ and extend each singleton Y greedily. There are $O(nk)$ singletons, and each singleton Y has n iterations, each of which takes $O(nk)$ queries. Therefore, the query complexity of Algorithm 2 is $O(n^2k^2 + n^3k^2) = O(n^3k^2)$. We state our main theorem below.

Algorithm 2 Enu$_2$+Greedy$_1$

Input: Set V, monotone k-submodular function f, costs $c(a)$ for $a \in V$, budget B
Output: A solution in $(k+1)^V$

1: Let $S^* \in \arg\max_{S:\,|S|=2, c(S)\le B} f(S)$ be a size-2 solution giving the largest value.
2: **for** every $Y \in (k+1)^V$ with $|Y| \le 1$ **do**
3: $G_0 \leftarrow Y, V_0 \leftarrow V \backslash U(Y)$
4: **for** t from 1 to n **do**
5: Let $(a_t, d_t) = \arg\max_{a \in V_{t-1}, d \in [k]} \frac{\Delta_{a,d}(G_{t-1})}{c(a)}$ maximize the marginal density, and denote $\theta_t = \frac{\Delta_{a_t,d_t}(G_{t-1})}{c(a_t)}$
6: **if** $c(G_{t-1}) + c(a_t) \le B$ **then**
7: $G_t \leftarrow G_{t-1} \cup \{(a_t, d_t)\}$
8: **else**
9: $G_t \leftarrow G_{t-1}$
10: **end if**
11: $V_t \leftarrow V_{t-1} \backslash \{a_t\}$
12: **end for**
13: $S^* \leftarrow G_n$ if $f(G_n) > f(S^*)$
14: **end for**
15: **return** S^*

Theorem 1. *Algorithm 2 has an approximation ratio at least* 0.328 *for the kSKM, with query complexity* $O(n^3k^2)$.

In the remainder of this section, we prove Theorem 1. For each greedy procedure (in Line 3-12) extended from any Y we are interested in, let $l+1$ be the first time when it does not add an item in $U(OPT)$ to the current solution because its addition would violate the budget (i.e., $a_{l+1} \in U(OPT)$ and $c(a_{l+1}) + c(G_l) > B$). We can further assume that $l+1$ is the first time t for which $G_t = G_{t-1}$. This assumption is without loss of generality, because if it happens earlier for some $t' < l+1$, then $a_{t'}$ does not belong to the optimal solution, nor the greedy solution we are interested in; thus, we can remove $a_{t'}$ from the ground set V, without affecting the analysis, the optimal solution, and the greedy solution. Thus, we can safely assume $G_t = G_{t-1} \cup \{(a_t, d_t)\}$ for $t = 1, \ldots, l$.

The proof of Theorem 1 consists of the following steps. Set $\beta \in (0,1)$ to be a tunable parameter in the analysis.

- **Step 1.** Section 4.1 considers the greedy procedure extended from $Y = \varnothing$. If G_l has a cost at least βB, then it has a desired approximation. Otherwise, the $(l+1)$-th item (denoted by a^*) has a cost greater than $(1-\beta)B$, and then go to Step 2.
- **Step 2.** Section 4.2 considers the greedy procedure extended from a singleton Y containing item a^*. If $G_l \setminus Y$ has a cost larger than $\frac{B-c(Y)}{2}$, then G_l has a desired approximation. Otherwise, the $(l+1)$-th item (denoted by a^{**}) has a cost greater than $\frac{B-c(Y)}{2}$, and then go to Step 3.
- **Step 3.** Section 4.3 considers the two large-cost items a^*, a^{**} in OPT. If they can achieve a large function value, it is attained in Line 1 by enumerating all size-2 solutions. Otherwise, the remaining items $U(OPT) \setminus \{a^*, a^{**}\}$ has a small cost but a large function value. We show that the greedy procedure extended from $Y = \varnothing$ has a good approximation on this value.

4.1 Extending $Y = \varnothing$ greedily

In this subsection, we consider the fully greedy procedure of extending an empty set $Y = \varnothing$ greedily (Line 3-12 in Algorithm 2).

We will show that if the partial greedy solution G_l has a large enough cost (above half the budget), it achieves a good approximation ratio, even though discarding the remaining part of the final outcome of the greedy procedure. This claim heavily relies on the following lemma.

Lemma 4. *For an arbitrary set* $R \in (k+1)^V$ *and any* $t = 1, \ldots, l$,

$$f(G_t) \ge \frac{1}{2}\left(1 - e^{-\frac{2c(G_t)}{c(R)}}\right) \cdot f(R).$$

Proof. Fixing t, for an arbitrary $j = 0, \ldots, t$, we consider the unconstrained maximization over the items in $V' := U(G_j) \cup U(R) = \{e_1 \ldots, e_m\}$. Assume w.l.o.g. that $e_1 = a_1, e_2 = a_2, \ldots, e_j = a_j$. Let Algorithm 1 consider the items in the order of $e_1, \ldots, e_m$. Denote by T the optimal solution of the unconstrained maximization over $U(G_j) \cup U(R)$. Recall the notations in Eq. (3.1) and (3.2), and note that $G_j = S_j$. Then we can apply Lemma 3 to bound $f(T)$:

$$\begin{aligned}
f(R) &\le f(T) \\
&\le 2f(G_j) + \sum_{(a,d) \in T_j \setminus G_j} \Delta_{a,d}(G_j) \\
&= 2f(G_j) + \sum_{(a,d) \in T_j \setminus G_j} c(a) \cdot \frac{\Delta_{a,d}(G_j)}{c(a)} \\
&\le 2f(G_j) + \sum_{(a,d) \in T_j \setminus G_j} c(a) \cdot \frac{\Delta_{a_{j+1},d_{j+1}}(G_j)}{c(a_{j+1})} \qquad (4.1) \\
&\le 2f(G_j) + c(R) \cdot \frac{\Delta_{a_{j+1},d_{j+1}}(G_j)}{c(a_{j+1})}, \qquad (4.2) \\
&= 2\left(f(G_j) + \frac{c(R)}{2} \cdot \theta_{j+1}\right). \qquad (4.3)
\end{aligned}$$

where Eq. (4.1) is because (a_{j+1}, d_{j+1}) is the pair of maximum marginal density by the greedy algorithm, and Eq. (4.2) is because the items in $T_j \setminus G_j$ must belong to R and their total cost is at most $c(R)$.

Let $B_j = c(G_j) = \sum_{\tau=1}^{j} c(a_\tau)$ for $j = 0, \ldots, l$, and let $B_{l+1} = c(G_l) + c(a_{l+1})$ (note that $B_{l+1} > B$ by definition). For $i = 1, \ldots, B_{l+1}$, we define

$$\rho_i = \theta_j \text{ if } i = B_{j-1} + 1, \ldots, B_j.$$

That is,

$$\rho_1 = \cdots = \rho_{B_1} = \theta_1,$$

$$\rho_{B_1+1} = \cdots = \rho_{B_2} = \theta_2,$$
$$\vdots$$
$$\rho_{B_l+1} = \cdots = \rho_{B_{l+1}} = \theta_{l+1}.$$

Using this definition, we obtain $f(G_j) = \sum_{\tau=1}^{j} c(a_\tau)\theta_\tau = \sum_{i=1}^{B_j} \rho_i$. Moreover,

$$\begin{aligned} &\min_{s=1,\ldots,B_t} \{\sum_{i=1}^{s-1} \rho_i + \frac{c(R)}{2} \cdot \rho_s\} \\ =& \min_{j=0,\ldots,t-1} \{\sum_{i=1}^{B_j} \rho_i + \frac{c(R)}{2} \cdot \rho_{B_j+1}\} \\ =& \min_{j=0,\ldots,t-1} \{f(G_j) + \frac{c(R)}{2} \cdot \theta_{j+1}\}. \end{aligned} \tag{4.4}$$

With these notations, we can use the following proposition.

Proposition 1 ([20]). *Let P and D be arbitrary positive integers, and $(\rho_i)_{i=1}^{P}$ be arbitrary nonnegative real values with $\rho_1 > 0$. Then*

$$\frac{\sum_{i=1}^{P} \rho_i}{\min_{s=1,\ldots,P}(\sum_{i=1}^{s-1} \rho_i + D\rho_t)} \geq 1 - (1 - \frac{1}{D})^P \geq 1 - e^{-P/D}.$$

Finally, by (4.3), (4.4) and Proposition 1, we obtain

$$\begin{aligned} \frac{f(G_t)}{f(R)} &= \frac{\sum_{i=1}^{B_t} \rho_i}{f(R)} \\ &\geq \frac{\sum_{i=1}^{B_t} \rho_i}{2 \cdot \min_{j=0,\ldots,t-1}\{f(G_j) + \frac{c(R)}{2} \cdot \theta_{j+1}\}} \\ &\geq \frac{\sum_{i=1}^{B_t} \rho_i}{2 \cdot \min_{s=1,\ldots,B_t}\{\sum_{i=1}^{s-1} \rho_i + \frac{c(R)}{2} \cdot \rho_s\}} \\ &\geq \frac{1}{2}(1 - e^{-\frac{B_t}{c(R)/2}}), \end{aligned} \tag{4.5}$$

establishing the proof. □

If $c(G_l) \geq \beta B$, then applying Lemma 4 to $R = OPT$ and $t = l$ immediately gives

$$\begin{aligned} f(G_l) &\geq \frac{1}{2}(1 - e^{-\frac{2 \cdot c(G_l)}{c(OPT)}}) \cdot f(OPT) \\ &\geq \frac{1}{2}(1 - e^{-\frac{2 \cdot \beta B}{B}}) \cdot f(OPT) \\ &= \frac{1}{2}(1 - e^{-2\beta}) \cdot f(OPT) \end{aligned}$$

which implies that G_l already gives a $\frac{1}{2}(1 - e^{-2\beta})$-approximation.

Corollary 1. *When $c(G_l) \geq \beta B$, Algorithm 2 has an approximation ratio at least $\frac{1}{2}(1 - e^{-2\beta})$.*

In the next section, we will consider the case when $c(G_l) < \beta B$. By the definition of the index $l+1$, item a_{l+1} belongs to the optimal solution OPT, but fails to add into the greedy solution due to the budget constraint. Thus it has a large cost $c(a_{l+1}) > B - c(G_l) > (1-\beta)B$. Specifically, denote this item by $a^* \in U(OPT)$, and assume that its dimension aligned with OPT is d^*, that is, $(a^*, d^*) \in OPT$.

4.2 Extending $Y = \{(a^*, d^*)\}$ greedily

According to the analysis in Section 4.1, it can be assumed that there is an item a^* in OPT with a cost larger than half the budget, i.e., $c(a^*) > (1-\beta)B$. In this subsection, we consider the procedure of extending the singleton $Y = \{(a^*, d^*)\} \subseteq OPT$ greedily (Line 3-12 in Algorithm 2).

Again, with a slight abuse of notations, let $l+1$ be the first time when the greedy procedure does not add an item in $U(OPT)$ to the current solution because its addition would violate the budget. Further assume that $l+1$ is the first time t for which $G_t = G_{t-1}$.

The following lemma gives a lower bound on the function value of G_l.

Lemma 5. *The greedy procedure extended from $Y = \{(a^*, d^*)\} \subseteq OPT$ guarantees*

$$f(G_l) \geq \frac{1}{2}(1 - e^{\frac{-2c(G_l \setminus Y)}{B - c(Y)}})f(OPT) + \frac{1}{2}(1 + e^{\frac{-2c(G_l \setminus Y)}{B - c(Y)}})f(Y).$$

Proof. Define a function

$$h(S) := f(S \cup Y) - f(Y), \quad \forall S \in (k+1)^{V \setminus U(Y)}.$$

This function is defined on ground set $V \setminus U(Y)$, and is clearly k-submodular.

Note that the greedy procedure of extending Y with respect to f is equivalent to that of extending $\varnothing$ with respect to h. Thus, Lemma 4 also works for h. Applying Lemma 4 to function h and set $R = OPT \setminus Y$ gives

$$\begin{aligned} f(G_l) - f(Y) = h(G_l \setminus Y) &\geq \frac{1}{2}(1 - e^{-\frac{2c(G_l \setminus Y)}{c(OPT \setminus Y)}}) \cdot h(OPT \setminus Y) \\ &\geq \frac{1}{2}(1 - e^{-\frac{2c(G_l \setminus Y)}{B - c(Y)}}) \cdot h(OPT \setminus Y) \\ &= \frac{1}{2}(1 - e^{-\frac{2c(G_l \setminus Y)}{B - c(Y)}}) \cdot (f(OPT) - f(Y)). \end{aligned}$$

Rearranging the terms proves the lemma. □

The lower bound on $f(G_l)$ in the above lemma relies on the budget consumed by G_l. The following says that if G_l consumes a large proportion of budget, then it has a good approximation.

Corollary 2. *When $c(G_l \setminus Y) \geq \beta(B - c(Y))$, we have $f(G_l) \geq \frac{1}{2}(1 - e^{-2\beta}) \cdot f(OPT)$.*

In the next section, we focus on the case when $c(G_l \setminus Y) < \beta(B - c(Y))$. By the definition of the index $l+1$, item a_{l+1} belongs to the optimal solution OPT, but fails to add into the greedy solution due to the budget constraint. That is, $c(G_l \setminus Y) + c(a_{l+1}) > B - c(Y)$. Then we derive

$$c(a_{l+1}) > B - c(Y) - \beta(B - c(Y)) = (1-\beta)(B - c(Y)). \tag{4.6}$$

Specifically, denote this large-cost item a_{l+1} by a^{**}, and assume that its dimension aligned with OPT is d^{**}, that is, $(a^{**}, d^{**}) \in OPT$.

4.3 Two large-cost items in OPT

According to the analysis in Sections 4.1 and 4.2, it remains to consider the case when there is an item $a^* \in U(OPT)$ with cost $c(a^*) > (1-\beta)B$, and an item $a^{**} \in U(OPT)$ with cost $c(a^{**}) > (1 - \beta)(B - c(a^*))$. Denote $S' = \{(a^*, d^*), (a^{**}, d^{**})\} \subseteq$

OPT. If these two large-cost items give a large function value $f(S') \geq \frac{1}{2}(1-e^{-2\beta})f(OPT)$, then Line 1 in Algorithm 2 (selecting the best size-2 solution) already achieves an approximation ratio $\frac{1}{2}(1-e^{-2\beta})$. Hence, this subsection considers the subcase when $f(S') < \frac{1}{2}(1-e^{-2\beta})f(OPT)$.

Let $R := OPT \setminus S'$ be the remaining part in the optimal solution. We have

$$f(R) \geq f(OPT) - f(S') > \frac{1}{2}(1+e^{-2\beta})f(OPT). \quad (4.7)$$

The cost of R is

$$c(R) \leq B - c(S') < B - c(a^*) - (1-\beta)(B-c(a^*)) = \beta(B-c(a^*)).$$

Thus, R has a relatively small cost but a large function value.

Next, we use the greedy procedure extended from empty set $Y = \varnothing$ to give an approximation of $f(R)$. Recall that $l+1$ is the first time when an item in $U(OPT)$ is not added to the current solution due to budget constraint B, and in this procedure the $(l+1)$-th item is $a_{l+1} = a^*$. Thus, the cost of the partial greedy solution G_l plus the cost of a^* exceeds the budget B, that is, $c(G_l) > B - c(a^*)$.

By Lemma 4 and Eq. (4.7), we have

$$\begin{aligned} f(G_l) &\geq \frac{1}{2}(1-e^{\frac{-2c(G_l)}{c(R)}}) \cdot f(R) \\ &\geq \frac{1}{2}(1-e^{-2\frac{B-c(a^*)}{\beta(B-c(a^*))}}) \cdot f(R) \\ &= \frac{1}{2}(1-e^{-\frac{2}{\beta}}) \cdot f(R) \\ &\geq \frac{1}{4}(1-e^{-\frac{2}{\beta}})(1+e^{-2\beta})f(OPT). \end{aligned}$$

Now, we are ready to summarize the above results, and set a best value of the parameter β to optimize the approximation ratio.

- When G_l obtained from greedily extending $\varnothing$ has cost at least βB, it has a ratio $\frac{1}{2}(1-e^{-2\beta})$ (Corollary 1).
- When G_l obtained from greedily extending (a^*, d^*) has cost at least $\beta(B-c(Y))$, the ratio is $\frac{1}{2}(1-e^{-2\beta})$ (Corollary 2).
- When $f(S') \geq \frac{1}{2}(1-e^{-2\beta})f(OPT)$, the ratio is $\frac{1}{2}(1-e^{-2\beta})$; otherwise, G_l obtained from greedily extending $\varnothing$ has a ratio of $\frac{1}{4}(1-e^{-\frac{2}{\beta}})(1+e^{-2\beta})$.

Thus, the approximation ratio is at least

$$\min\left\{\frac{1}{2}(1-e^{-2\beta}),\ \frac{1}{4}(1-e^{-\frac{2}{\beta}})(1+e^{-2\beta})\right\}.$$

Setting $\beta = 0.5337$, we obtain an approximation ratio of

$$\frac{1}{2}(1-e^{-2\beta}) \approx \frac{1}{4}(1-e^{-\frac{2}{\beta}})(1+e^{-2\beta}) = 0.3280\ldots,$$

establishing the proof of Theorem 1.

5 Extensions

In this section, we first extend Algorithm 2 to a general version by enumerating more items, and present better approximation ratios. Then, we show that the results can also be extended to non-monotone k-submodular functions.

Generalization of Algorithm 2. The approximation ratio 0.328 can be improved by enumerating more items. Consider the follow general algorithm $\mathbf{Enu}_p + \mathbf{Greedy}_{p-1}$. Given an enumeration size $p \in \mathbb{Z}$, it first enumerates all solutions that contain p items (Line 1, $\mathbf{Enu}_p$), and then greedily extend every solution that contains $p-1$ items (Line 2-14, $\mathbf{Greedy}_{p-1}$). The outcome is the best among the solutions considered. The query complexity of the algorithm is $O(n^{p+1}k^p)$. It is easy to see that Algorithm 2 is exactly $\mathbf{Enu}_2 + \mathbf{Greedy}_1$.

Algorithm 3 $\mathbf{Enu}_p + \mathbf{Greedy}_{p-1}$

Input: Set V, monotone k-submodular function f, costs $c(a)$ for $a \in V$, budget B, enumeration size $p \in \mathbb{Z}$
Output: A solution in $(k+1)^V$

1: Let $S^* \in \arg\max_{S:\ |S|=p, c(S)\leq B} f(S)$ be a size-p solution giving the largest value.
2: **for** every $Y \in (k+1)^V$ with $|Y| = p-1$ **do**
3: $G_0 \leftarrow Y, V_0 \leftarrow V \backslash U(Y)$
4: **for** t from 1 to n **do**
5: Let $(a_t, d_t) = \arg\max_{a \in V_{t-1}, d \in [k]} \frac{\Delta_{a,d}(G_{t-1})}{c(a)}$ maximize the marginal density, and denote $\theta_t = \frac{\Delta_{a_t,d_t}(G_{t-1})}{c(a_t)}$
6: **if** $c(G_{t-1}) + c(a_t) \leq B$ **then**
7: $G_t \leftarrow G_{t-1} \cup \{(a_t, d_t)\}$
8: **else**
9: $G_t \leftarrow G_{t-1}$
10: **end if**
11: $V_t \leftarrow V_{t-1} \backslash \{a_t\}$
12: **end for**
13: $S^* \leftarrow G_n$ if $f(G_n) > f(S^*)$
14: **end for**
15: **return** S^*

We can prove the approximation ratios of $\mathbf{Enu}_p$+$\mathbf{Greedy}_{p-1}$ under the analysis framework in Section 4. Given $p \geq 3$, we first follow the analysis in Sections 4.1 and 4.2. After that we know that there are two large-cost items in OPT, and then we consider the procedure of extending them greedily: if G_l has a large cost, then $f(G_l)$ has a good approximation $\frac{1}{2}(1-e^{-2\beta})$; otherwise, there is a third large-cost item, and then we extend these three large-cost items greedily. This process is repeated until we have known that there are p large-cost items in OPT (denoted by S'). If S' have a good approximation $\frac{1}{2}(1-e^{-2\beta})$, it is achieved by the enumeration in Line 1; otherwise, $R = OPT \setminus S'$ must have an approximation $1 - \frac{1}{2}(1-e^{-2\beta})$ for the optimum, and the algorithm can achieve an approximation $\frac{1}{2}(1-e^{-\frac{2}{\beta^{p-1}}})$ for $f(R)$.

Indeed, the approximation ratio is at least

$$\min\left\{\frac{1}{2}(1-e^{-2\beta}),\ \frac{1}{4}(1-e^{-\frac{2}{\beta^{p-1}}})(1+e^{-2\beta})\right\}. \quad (5.1)$$

By optimizing the parameter β, we can find the specific values of the approximation ratios, as shown in Figure 1 (a).

Note that the enumeration size p only influences the second term. When p is large enough, Eq. (5.1) becomes

$$\min\left\{\frac{1}{2}(1-e^{-2\beta}),\ \frac{1}{4}(1+e^{-2\beta})\right\}, \quad (5.2)$$

which is at most $\frac{1}{3}$, attained by $e^{-2\beta} = \frac{1}{3}$. The value of $\frac{1}{3}$ is natural, because our analysis uses a ratio $\frac{1}{2}(1-e^{-2\beta})$ to bound $f(S')$ (which

is achievable by the algorithm), and then use the ratio $1-\frac{1}{2}(1-e^{-2\beta})$ to bound $f(OPT \setminus S')$, for which the best possible approximation is $\frac{1}{2}$. In this way we cannot expect an approximation ratio beating $\frac{1}{3}$.

Proposition 2. *For the kSKM, Algorithm* $\textbf{Enu}_p + \textbf{Greedy}_{p-1}$ *has an approximation ratio given in Eq.* (5.1)*. It approaches* $\frac{1}{3}$ *as* p *increases.*

Non-monotone functions. We look at the performance guarantee of Algorithm 2 when f is a non-monotone k-submodular function. The proposed analysis framework still works. A k-submodular function must be pairwise monotone, which implies that for any item a and any $S \in (k+1)^V$, there always exists a dimension d so that the marginal gain $\Delta_{a,d}(S)$ is non-negative. Thus, the greedy algorithm will have a non-negative marginal gain in each iteration, i.e., the function value is non-decreasing. This good property partially makes the analysis for non-monotone functions quite similar.

The main difference is that when we consider the unconstrained k-submodular maximization, the statement in Lemma 3 would be replaced by $f(T) \leq 3f(S_t) + \sum_{(a,d)\in T_t \setminus S_t} \Delta_{a,d}(S_t)$. Subsequently, the statement in Lemma 4 becomes $f(G_t) \geq \frac{1}{3}(1-e^{-\frac{3c(G_t)}{c(R)}}) \cdot f(R)$. Finally, the approximation ratio is at least

$$\min\left\{\frac{1}{3}(1-e^{-3\beta}),\ \frac{1}{9}(1-e^{-\frac{3}{\beta}})(2+e^{-3\beta})\right\}.$$

Setting $\beta = 0.46$, we obtain an approximation ratio of

$$\frac{1}{3}(1-e^{-3\beta}) \approx \frac{1}{9}(1-e^{-\frac{3}{\beta}})(2+e^{-3\beta}) = 0.249\ldots$$

Proposition 3. *For the non-monotone kSKM, Algorithm 2 has an approximation ratio 0.249.*

Of course, in the non-monotone case, Algorithm 2 can also be generalized to $\textbf{Enu}_p$+$\textbf{Greedy}_{p-1}$. The approximation ratio is at least

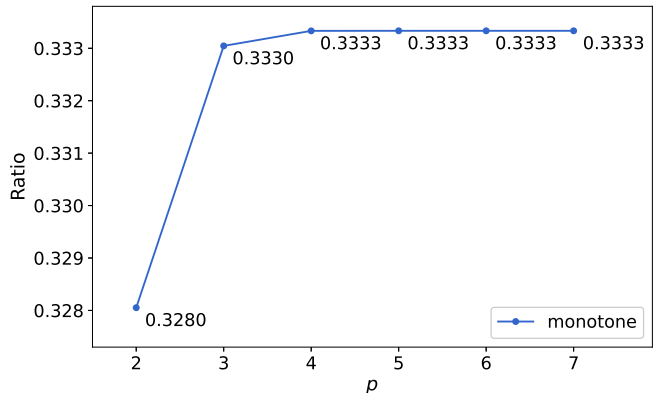
$$\min\left\{\frac{1}{3}(1-e^{-3\beta}),\ \frac{1}{9}(1-e^{-\frac{3}{\beta^{p-1}}})(2+e^{-3\beta})\right\}.$$

By optimizing β we can find the specific values, as shown in Figure 1 (b). When p increases, the ratio approaches $\frac{1}{4}$ very quickly.

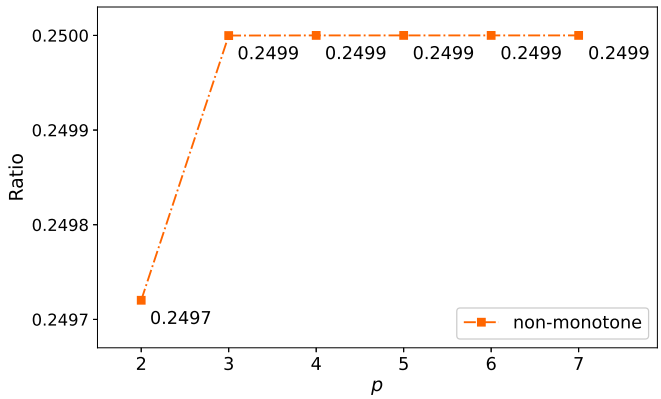
6 Discussions

For the monotone kSKM, Tang *et al.* [17] considered the algorithm of enumerating all size-2 solutions and then extending each of them greedily, that is, $\textbf{Greedy}_2$ in our notations. They proved that this algorithm has an approximation ratio at least $\frac{1}{2}(1-e^{-1}) \approx 0.316$, however, their proof is flawed. In the analysis, setting $Y \subseteq OPT$ consisting of the two items in OPT with largest marginal gain (which would be enumerated by the algorithm), consider the greedy procedure of extending Y. They assumed (in Eq. (6) [17]) that the marginal gain of the item a_{l+1} is $\Delta_{a_{l+1},d_{l+1}}(G_l) \leq \frac{f(Y)}{2}$, where $l+1$ is the first time when the greedy procedure does not add an item in OPT to the current solution due to budget, and d_{l+1} is the dimension greedily assigned to a_{l+1} in that iteration. This statement may be not true, because the item a_{l+1} may have a different dimension in OPT with d_{l+1} so that the marginal gain $\Delta_{a_{l+1},d_{l+1}}(G_l)$ could be much larger than $\frac{f(Y)}{2}$.

Fortunately, $\textbf{Greedy}_2$ does have an approximation ratio at least 0.316, although the proof in [17] is flawed. Note that $\textbf{Greedy}_2$ is stronger than our Algorithm 2 ($\textbf{Enu}_2 + \textbf{Greedy}_1$), because it first enumerates all size-2 solutions and then extend each of them greedily, while Algorithm 2 enumerates all size-2 solutions and merely extend each singleton. Hence, $\textbf{Greedy}_2$ must also have an approximation ratio 0.328 for the monotone kSKM, even better than the claimed 0.316.



(a) Monotone



(b) Non-monotone

Figure 1: The approximation ratios of $\textbf{Enu}_p + \textbf{Greedy}_{p-1}$ are presented with 4 significant digits.

For future directions, we notice that our analysis indeed discards the second term about $f(Y)$ in the RHS of Lemma 5. It may derive a better approximation if one can utilize this term in a more carefully approach.

Acknowledgements. Our work was supported in part by the Guangdong Provincial Key Laboratory of Interdisciplinary Research and Application for Data Science, UIC, project code 2022B1212010006, by Guangdong Higher Education Upgrading Plan (2021-2025) of "Rushing to the Top, Making Up Shortcomings and Strengthening Special Features" with UIC research grant R0400001-22, and by Artificial Intelligence and Data Science Research Hub, UIC, No. 2020KSYS007 and No. UICR0400025-21. Zhongzheng Tang is supported by National Natural Science Foundation of China under Grant No. 12101069. Chenhao Wang is supported by NSFC under Grant No. 12201049, and is also supported by UIC grants of UICR0400014-22, UICR0200008-23 and UICR0700036-22.

References

[1] Jingwen Chen, Zhongzheng Tang, and Chenhao Wang, 'Monotone k-submodular knapsack maximization: An analysis of the greedy+ singleton algorithm', in *Proceedings of the 16th International Conference on Algorithmic Aspects in Information and Management (AAIM)*, pp. 144–155. Springer, (2022).

[2] Alina Ene and Huy Nguyen, 'Streaming algorithm for monotone k-submodular maximization with cardinality constraints', in *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pp. 5944–5967. PMLR, (2022).

[3] Igor Gridchyn and Vladimir Kolmogorov, 'Potts model, parametric maxflow and k-submodular functions', in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2320–2327, (2013).

[4] Hiroshi Hirai and Yuni Iwamasa, 'On k-submodular relaxation', *SIAM Journal on Discrete Mathematics*, **30**(3), 1726–1736, (2016).

[5] Anna Huber and Vladimir Kolmogorov, 'Towards minimizing k-submodular functions', in *Proceedings of the 2nd International Symposium on Combinatorial Optimization (ISCO)*, pp. 451–462. Springer, (2012).

[6] Satoru Iwata, Shin-ichi Tanigawa, and Yuichi Yoshida, 'Improved approximation algorithms for k-submodular function maximization', in *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 404–413, (2016).

[7] Hui Lin and Jeff Bilmes, 'Multi-document summarization via budgeted maximization of submodular functions', in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 912–920, (2010).

[8] Lan Nguyen and My T Thai, 'Streaming k-submodular maximization under noise subject to size constraint', in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 7338–7347. PMLR, (2020).

[9] Naoto Ohsaka and Yuichi Yoshida, 'Monotone k-submodular function maximization with size constraints', in *Proceedings of the 28th International Conference on Neural Information Processing Systems (NeurIPS)*, volume 1, pp. 694–702, (2015).

[10] Hiroki Oshima, 'Improved randomized algorithm for k-submodular function maximization', *SIAM Journal on Discrete Mathematics*, **35**(1), 1–22, (2021).

[11] Canh V Pham, Dung KT Ha, Huan X Hoang, and Tan D Tran, 'Fast streaming algorithms for k-submodular maximization under a knapsack constraint', in *Proceedings of the IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10. IEEE, (2022).

[12] Canh V Pham, Quang C Vu, Dung KT Ha, Tai T Nguyen, and Nguyen D Le, 'Maximizing k-submodular functions under budget constraint: applications and streaming algorithms', *Journal of Combinatorial Optimization*, **44**(1), 723–751, (2022).

[13] Shinsaku Sakaue, 'On maximizing a monotone k-submodular function subject to a matroid constraint', *Discrete Optimization*, **23**, 105–113, (2017).

[14] Tasuku Soma, 'No-regret algorithms for online k-submodular maximization', in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1205–1214. PMLR, (2019).

[15] Yunjing Sun, Yuezhu Liu, and Min Li, 'Maximization of k-submodular function with a matroid constraint', in *Proceedings of the 17th Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pp. 1–10. Springer, (2022).

[16] Zhongzheng Tang, Chenhao Wang, and Hau Chan, 'Monotone k-submodular secretary problems: cardinality and knapsack constraints', *Theoretical Computer Science*, **921**, 86–99, (2022).

[17] Zhongzheng Tang, Chenhao Wang, and Hau Chan, 'On maximizing a monotone k-submodular function under a knapsack constraint', *Operations Research Letters*, **50**(1), 28–31, (2022).

[18] Baoxiang Wang and Huanjian Zhou, 'Multilinear extension of k-submodular functions', *arXiv preprint arXiv:2107.07103*, (2021).

[19] Justin Ward and Stanislav Živnỳ, 'Maximizing k-submodular functions and beyond', *ACM Transactions on Algorithms*, **12**(4), 1–26, (2016).

[20] Laurence A Wolsey, 'Maximising real-valued submodular functions: Primal and dual heuristics for location problems', *Mathematics of Operations Research*, **7**(3), 410–425, (1982).

[21] Hao Xiao, Qian Liu, Yang Zhou, and Min Li, 'Non-monotone k-submodular function maximization with individual size constraints', in *International Conference on Computational Data and Social Networks*, pp. 268–279. Springer, (2022).

[22] Hao Xiao, Qian Liu, Yang Zhou, and Min Li, 'Approximation algorithms for k-submodular maximization subject to a knapsack constraint', *arXiv preprint arXiv:2306.14520*, (2023).

[23] Kemin Yu, Min Li, Yang Zhou, and Qian Liu, 'Guarantees for maximization of k-submodular functions with a knapsack and a matroid constraint', in *Proceedings of the 16th International Conference on Algorithmic Aspects in Information and Management (AAIM)*, pp. 156–167. Springer, (2022).

[24] Kemin Yu, Min Li, Yang Zhou, and Qian Liu, 'On maximizing monotone or non-monotone k-submodular functions with the intersection of knapsack and matroid constraints', *Journal of Combinatorial Optimization*, **45**(3), 1–21, (2023).

[25] Qimeng Yu and Simge Küçükyavuz, 'An exact cutting plane method for k-submodular function maximization', *Discrete Optimization*, **42**, 100670, (2021).