TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 01/20 pp1633–1641
DOI: 10.26599/TST.2023.9010122
Volume 29, Number 6, December 2024

Approximation Algorithms for Maximization of *k*-Submodular Function Under a Matroid Constraint

Yuezhu Liu, Yunjing Sun, and Min Li*

Abstract: In this paper, we design a deterministic 1/3-approximation algorithm for the problem of maximizing non-monotone k-submodular function under a matroid constraint. In order to reduce the complexity of this algorithm, we also present a randomized 1/3-approximation algorithm with the probability of $1-\varepsilon$, where ε is the probability of algorithm failure. Moreover, we design a streaming algorithm for both monotone and non-monotone objective k-submodular functions.

Key words: *k*-submodular; matroid constraint; deterministic algorithm; randomized algorithm; streaming algorithm

1 Introduction

k-submodular function is a generalization of submodular function[2] and has been applied in machine learning and data mining, including influence maximization with k kinds of topics or sensor placement with k kinds of sensors. For the problem of maximizing monotone k-submodular functions, Ward and Živný[3] gave the deterministic 1/2-approximation algorithm. Later, Iwata et al.[4] presented a randomized approximation algorithm with approximation ratio $k/(2k-1)$. For the maximization of non-monotone k-submodular function without constraints, the approximation ratio of $\max\{1/3, 1/(1+a)\}$ is given, where $a=\max\left\{1, \sqrt{(k-1)/4}\right\}$[3]. Later, Iwata et al.[4] improved the approximation guarantee to 1/2 based on randomized algorithm. Based on their algorithm, Oshima[5] improved it again and obtained a $(k^2+1)/(2k^2+1)$-approximation for $k\geqslant 3$. Meanwhile, he also gave a randomized $(\sqrt{17}-3)/2$-approximation algorithm for $k=3$. Besides, there are also some results on the maximization of monotone k-submodular functions with constraints. Under the size constraint, Ohsaka and Yoshida[6] gave constant-factor approximation algorithms with approximation ratios of 1/2 for the total size constraint and 1/3 for the individual size constraint. Later, Nguyen and Thai[7] proposed new streaming algorithms for the maximization of k-submodular functions with a total size constraint. Under the knapsack constraint, Tang et al.[8] presented a deterministic $(1/2-1/2\mathrm{e})$-approximation algorithm. With a matroid constraint, Calinescu et al.[9] designed a greedy algorithm that outputs a 1/2-approximation solution. Furthermore, there is a deterministic 1/3-approximation algorithm for the maximization of non-monotone k-submodular functions with a total size constraint[7]. For more research on k-submodular maximization, we can see the Refs. [10–14].

In this paper, we plan to study the the problem of maximizing the k-submodular function under a matroid constraint and give the following contributions:

- We design a deterministic algorithm with an approximation ratio of 1/3 and complexity of $O(|V|r(a+kb))$ for the non-monotone k-submodular function, where $|V|$ represents the number of elements in V, r represents the rank of the given matroid, a is

• Yuezhu Liu, Yunjing Sun, and Min Li are with School of Mathematics and Statistics, Shandong Normal University, Jinan 250014, China. E-mail: lyz01301111@163.com; syj985211@163.com; liminEmily@sdnu.edu.cn.
• A preliminary version of this paper appears in Proceedings of the 17th International Conference on heory and Applications of Models of Computation (TAMC)[1].
* To whom correspondence should be addressed.
Manuscript received: 2023-01-16; revised: 2023-05-31; accepted: 2023-09-11

the number of times to calculate whether a set is an independent set in this matroid, and b is the number of times to calculate a value of the k-submodular function.

- We design a randomized 1/3-approximation algorithm with the probability of $1-\varepsilon$ for the non-monotone k-submodular function, which reduces the complexity of the deterministic algorithm to $O\left(|V|\left(a\log\frac{r}{\varepsilon_1}+kb\log\frac{r}{\varepsilon_2}\right)\log r\right)$, where $\varepsilon=\max\{\varepsilon_1, \varepsilon_2\}$, and ε_1 and ε_2 denote the probabilities of algorithm failure.
- We also try to give a streaming algorithm for both monotone and non-monotone k-submodular functions.

The rest of this paper is organized as follows. In Section 2, we briefly introduce the relevant knowledge of k-submodular functions and matroid. In Section 3, two algorithms are included for the non-monotone k-submodular functions. Furthermore, a streaming algorithm is introduced for the monotone and non-monotone objective functions in Section 4. We give the final conclusions in Section 5.

2 Preliminary

At beginning, we give the definition of the submodular functions. Given a finite set V, a real-valued set function $f: 2^V \to \mathbf{R}$ is called a "submodular function" if

$$f(S)+f(T) \geqslant f(S\cup T)+f(S\cap T),\ \forall S, T\subseteq V.$$

The submodular functions are also known for the property of diminishing marginal gain. For any subset $S\subseteq V$ and an element $e\in V\setminus S$, we use

$$f_S(e)=f(S\cup\{e\})-f(S)$$

to denote the the marginal gain of e added to S. Then f is a submodular function if and only if

$$f_S(e)\geqslant f_T(e),\ \forall S\subseteq T,\ \forall e\in V\setminus T.$$

The k-submodular function is a generalization of the submodular function. Suppose that there is a position for each element in ground set V, the submodular function just decides whether the element is chosen or not. That is, there are only two choices for each element. But there will be k positions for each item of V for k-submodular functions. Now, we introduce the definition of k-submodular functions. We still use V to denote the finite ground set, assume k is a positive integer, and denote $[k]:=\{1, 2, \ldots, k\}$. Let $(k+1)^V := \{(S_1, S_2, \ldots, S_k) \mid S_i\subseteq V, \forall i\in[k], S_i\cap S_j=\varnothing, \forall i, j\in[k], i\neq j\}$, then the k-submodular functions can be defined as follows.

Definition 1 A set function $f:(k+1)^V\to\mathbf{R}$ is called a "k-submodular function" if for any $\boldsymbol{S}=(S_1, S_2, \ldots, S_k)$ and $\boldsymbol{T}=(T_1, T_2, \ldots, T_k)$ in $(k+1)^V$,

$$f(\boldsymbol{S})+f(\boldsymbol{T})\geqslant f(\boldsymbol{S}\sqcap\boldsymbol{T})+f(\boldsymbol{S}\sqcup\boldsymbol{T}),$$

where

$$\boldsymbol{S}\sqcap\boldsymbol{T}=(S_1\cap T_1, S_2\cap T_2, \ldots, S_k\cap T_k),$$
$$\boldsymbol{S}\sqcup\boldsymbol{T}=\Big((S_1\cup T_1)\setminus\Big(\bigcup_{i\neq 1}(S_i\cup T_i)\Big), (S_2\cup T_2)\setminus\Big(\bigcup_{i\neq 2}(S_i\cup T_i)\Big), \ldots, (S_k\cup T_k)\setminus\Big(\bigcup_{i\neq k}(S_i\cup T_i)\Big)\Big).$$

The domain of the k-submodular functions can also be expressed in the form of vectors. Given a k-tuple $\boldsymbol{S}=(S_1, S_2, \ldots, S_k)\in(k+1)^V$, we specify that if $e\in S_i$, then $\boldsymbol{S}(e)=i$, otherwise $e\notin\bigcup_{i\in[k]}S_i$, then $\boldsymbol{S}(e)=0$. At the same time, we define the support set of $\boldsymbol{S}$ as $S(\boldsymbol{S})=\bigcup_{i\in[k]}S_i$, which is composed of all elements appearing in $S_1, S_2, \ldots, S_k$, regardless of location. The cardinality of $S(\boldsymbol{S})$ will be used to represent the size of $\boldsymbol{S}$. Then we begin to characterize the relationship of the k-submodular definition field with partial order. The partial ordering relation is defined as follows. Taking $\boldsymbol{S}=(S_1, S_2, \ldots, S_k)$ and $\boldsymbol{T}=(T_1, T_2, \ldots, T_k)$ in $(k+1)^V$, if $S_i\subseteq T_i$ for all $i\in[k]$, we define $\boldsymbol{S}\preceq\boldsymbol{T}$. Then, we say that a k-submodular function is "monotone" if for any $\boldsymbol{S}\preceq\boldsymbol{T}$, we have $f(\boldsymbol{S})\leqslant f(\boldsymbol{T})$.

For the convenience of subsequent expression, we can also use the following symbols to represent the k-tuples in $(k+1)^V$. Given $\boldsymbol{S}=(S_1, S_2, \ldots, S_k)\in(k+1)^V$, we can also denote it by $\{(e, \boldsymbol{S}(e))\}$ with $e\in V$. Note that we donnot distinguish the elements not included by the support set $S(\boldsymbol{S})$ of $\boldsymbol{S}$. For example, assume that $V=\{e_1, e_2, e_3\}$, $k=2$, and take $\boldsymbol{S}=(\{e_2\}, \{e_1\})$, then $\boldsymbol{S}=\{(e_1, 2), (e_2, 1), (e_3, 0)\}$, and it can be also denoted by $\{(e_1, 2), (e_2, 1)\}$. If there is only one element in the support of $\boldsymbol{S}$, i.e., $\boldsymbol{S}=(S_1, S_2, \ldots, S_k)$ with $S_i=\{e\}$ and $S_j=\varnothing$ for all $i\neq j$, then we use (e, i) to denote $\boldsymbol{S}$ for short. So for a given $\boldsymbol{T}=(T_1, T_2, \ldots, T_k)\in(k+1)^V$ and $e\in V\setminus S(\boldsymbol{T})$, that adding e into T_i can be expressed as $\boldsymbol{T}+(e, i)$, and the corresponding marginal gain of adding e to T_i of $\boldsymbol{T}$ can be denoted by

$$f_{\boldsymbol{T}}(e, i) = f(\boldsymbol{T}+(e, i)) - f(\boldsymbol{T}) = f(T_1, T_2, \ldots, T_{i-1}, T_i \cup \{e\}, T_{i+1}, \ldots, T_k) - f(\boldsymbol{T}).$$

Using this definition, we can also obtain that the k-submodular function is monotone if and only if $f_{\boldsymbol{T}}(e, i) \geqslant 0$ for any $\boldsymbol{T} = (T_1, T_2, \ldots, T_k) \in (k+1)^V$, $e \in V \setminus S(\boldsymbol{T})$ and $i \in [k]$.

There are two important properties of k-submodular functions. One is orthant submodularity, the other is pairwise monotonicity. A k-submodular function f is "orthant submodular" if

$$f_{\boldsymbol{S}}(e, i) \geqslant f_{\boldsymbol{T}}(e, i),$$

for all $\boldsymbol{S}$ and $\boldsymbol{T}$ in $(k+1)^V$ and $\forall i \in [k]$, where $\boldsymbol{S} \leq \boldsymbol{T}$, and $e \in V \setminus S(\boldsymbol{T})$. The k-submodular function f is "pairwise monotone" if

$$f_{\boldsymbol{S}}(e, i) + f_{\boldsymbol{S}}(e, j) \geqslant 0,$$

for any $\boldsymbol{S} \in (k+1)^V$, $e \in V \setminus S(\boldsymbol{S})$ and $i, j \in [k]$ with $i \neq j$.

There has been an important result about how to characterize the k-submodular functions by orthant submodularity and pairwise monotonicity.

Theorem 1 (Ward and Živný[3]) A function $f: (k+1)^V \to \mathbf{R}$ is a k-submodular function if and only if f is orthant submodular and pairwise monotone.

Now we present some definitions about matroid.

Definition 2 (Korte and Vygen[15]) Assuming V is a finite set and $\mathcal{F} \subseteq 2^V$ (the power set of V), the system $(V, \mathcal{F})$ is named a matroid if it satisfies the following conditions:

(1) $\varnothing \in \mathcal{F}$,

(2) If $X \subseteq Y \in \mathcal{F}$, then $X \in \mathcal{F}$,

(3) If $X, Y \in \mathcal{F}$ and $|X| < |Y|$, then there is an element $e \in Y \setminus X$, such that $X \cup \{e\} \in \mathcal{F}$.

Each element in $\mathcal{F}$ is called an "independent set" of the matroid. For an independent set X, if any subset containing X is not an independent set in $\mathcal{F}$, then X is called a "maximal independent set", also called a *basis*. In this paper, $\mathcal{B}$ is used to represent the set formed by the bases of $\mathcal{F}$. In fact, for any $X, Y \in \mathcal{B}$, we have $|X| = |Y|$, and this size is called the "rank" of matroid. In this paper, r is used to represent the rank of the given matroid.

Here is a property about the independent sets of a matroid.

Lemma 1 (Korte and Vygen[15]) Given a matroid $(V, \mathcal{F})$, assume that X is an independent set and Y is a basis containing X, then for any element $e \in V$ not in X such that $X \cup \{e\}$ belongs to $\mathcal{F}$, there should be an element e' in $Y \setminus X$ satisfying that $(Y \setminus \{e'\}) \cup \{e\}$ is a basis too.

Given a matroid $(V, \mathcal{F})$ as well as a k-submodular function $f: (k+1)^V \to \mathbf{R}$ with $f(\varnothing) = 0$, the problem of maximizing k-submodular function with a matroid constraint (denoted by MkSfM for short) is expressed as follows:

$$\max_{\boldsymbol{S} \in (k+1)^V} f(\boldsymbol{S}) \text{ subject to } S(\boldsymbol{S}) \in \mathcal{F} \tag{1}$$

If the objective function f is monotone, we use mMkSfM to represent MkSfM. Otherwise, we use nMkSfM to represent it.

In the following part, we introduce a special kind of solutions of MkSfM.

Definition 3 A feasible solution $\bar{\boldsymbol{S}}$ to Formula (1) is called a "maximal solution" if it satisfies: For any feasible solution $\boldsymbol{T}$ to Formula (1) with $f(\bar{\boldsymbol{S}}) = f(\boldsymbol{T})$, we have $|S(\bar{\boldsymbol{S}})| \geqslant |S(\boldsymbol{T})|$. Specially, if $\bar{\boldsymbol{S}}$ is an optimal solution with maximal support set size, then it is also called a "maximal optimal solution".

Calinescu et al.[9] proved that in the monotone case, the size of the maximal optimal solution of the mMkSfM problem is rank r. Here, we will show that a similar result can be obtained in the non-monotone case.

Lemma 2 (Sun et al.[1]) For nMkSfM, the size of any maximal optimal solution is still r, that is, $|S(\bar{\boldsymbol{S}})| = r$.

Proof If not, there is a maximal optimal solution $\boldsymbol{O}$ with $|S(\boldsymbol{O})| < r$. Then the element e satisfied the following two conditions should exist:

(1) $e \notin S(\boldsymbol{O})$,

(2) $\{e\} \cup S(\boldsymbol{O}) \in \mathcal{F}$.

Since f is pairwise monotone, if we add the above e to two different positions i and j of $\boldsymbol{O}$, the sum of marginal gains should be nonnegative. That is, $f_{\boldsymbol{O}}(e, i) + f_{\boldsymbol{O}}(e, j) \geqslant 0$. Thus, it is concluded that at least one of $f_{\boldsymbol{O}}(e, i) \geqslant 0$ and $f_{\boldsymbol{O}}(e, j) \geqslant 0$ is true. If $f_{\boldsymbol{O}}(e, i) \geqslant 0$ is correct, we find that it does not reduce $f(\boldsymbol{O})$ by adding e to $\boldsymbol{O}$. Then there is still a feasible solution $\boldsymbol{O} + (e, i)$ making $f(\boldsymbol{O} + (e, i)) \geqslant f(\boldsymbol{O})$. This is in contradiction to that $\boldsymbol{O}$ is a maximal optimal solution. ■

3 Main Result for nMkSfM

In this part, we mainly analyze the nMkSfM problem, and design a deterministic algorithm and a randomized algorithm to obtain an approximation ratio of $1/3$. The difference between the two algorithms is that in the randomized algorithm, the approximation ratio is obtained under the failure probability of ε, but the complexity of this algorithm is reduced.

3.1 Deterministic algorithm for nMkSfM

Calinescu et al.[9] designed a greedy algorithm for the mMkSfM problem. Based on this algorithm, we give the deterministic Algorithm 1 for the nMkSfM problem. According to Lemma 2, we can know that the algorithm requires a total of r iterations, and the number of elements in the final output solution $\boldsymbol{S}$ of Algorithm 1 is r and $S(\boldsymbol{S})\in\mathcal{B}$. In each iteration j, we choose the element from a constructing set $V(\boldsymbol{S}^j)$ rather than the total set V. It can make sure that any element e added to the support set of $\boldsymbol{S}^j$ is still an independent set. Then we put the best position for these elements and add the one with maximal gain.

Theorem 2 For nMkSfM problem, we can get the approximation ratio of $1/3$ through Algorithm 1, and the complexity of the algorithm is $O(|V|\,r\,(a+kb))$.

Proof In fact, for the deterministic Algorithm 1, its complexity is easy to obtain, so we will focus on the derivation of the approximation ratio. Our analysis is based on the idea of exchanging elements with their positions between algorithm output solution and optimal solution. Before showing the following proof, we define several symbols.

In each step $j\in[r]$, let e^j and i^j represent the best element and position selected greedily in this step of Algorithm 1, respectively, and let $n^j\in[k]\setminus\{i^j\}$ be any other position except the position i^j. According to Algorithm 1, $\boldsymbol{S}^0=\boldsymbol{0}$, $\boldsymbol{S}=\boldsymbol{S}^r$, where $\boldsymbol{S}^j$ denotes the solution output by Algorithm 1 when it reaches the j-th step. Obviously, there is a relationship of $\boldsymbol{S}^j=\boldsymbol{S}^{j-1}+(e^j,i^j)$. Next, we construct a sequence $\boldsymbol{O}^j$ for $j=0,1,\ldots,r$, such that $\boldsymbol{O}^0=\boldsymbol{O}$, $\boldsymbol{O}^r=\boldsymbol{S}$. Then make $S(\boldsymbol{S}^j)$ and $S(\boldsymbol{O}^j)$ represent the support sets of $\boldsymbol{S}^j$ and $\boldsymbol{O}^j$, respectively, and let $L^{j+1}=S(\boldsymbol{O}^j)\setminus S(\boldsymbol{S}^j)$.

Algorithm 1 Deterministic algorithm for nMkSfM

Require: Non-monotone k-submodular function $f:(k+1)^V\rightarrow\mathbf{R}_+$, matroid $(V,\mathcal{F})$ with bases $\mathcal{B}$ and rank r

Ensure: Vector $\boldsymbol{S}$ whose support set belongs to $\mathcal{B}$

```
S^0 ← ∅ and j ← 1;
while j ⩽ r do
    Construct a set
        V(S^j) := {e ∈ V\S(S^{j−1}) | S(S^{j−1}) ∪ {e} ∈ F};
    (e, i) ← argmax_{e∈V(S^j), i∈[k]} f(S^{j−1} + (e, i));
    S^j ← S^{j−1} + (e, i);
    j ← j + 1;
end while
return S
```

Now, we explain how to construct a series of vectors $\boldsymbol{O}^j$ in $\mathcal{B}$ for $j=0,1,\ldots,r$, satisfying

$$\boldsymbol{O}^j\begin{cases}\succ \boldsymbol{S}^j, & \text{if } j=0,1,\ldots,r-1;\\ =\boldsymbol{S}^j, & \text{if } j=r.\end{cases}$$

We use the following ways to exchange elements for constructing the sequence:

$$o^j=\begin{cases}e^j, & \text{if } e^j\in L^j;\\ \text{any element in } L^j, & \text{otherwise.}\end{cases}$$

Then we define

$$\boldsymbol{O}^{j-\frac{1}{2}}=\boldsymbol{O}^{j-1}-(o^j,\boldsymbol{O}^{j-1}(o^j)),$$
$$\boldsymbol{O}^j=\boldsymbol{O}^{j-\frac{1}{2}}+(e^j,i^j).$$

With the sequence constructed above, we begin to prove the approximation ratio.

It can be known from Algorithm 1,

$$f(\boldsymbol{S}^j)-f(\boldsymbol{S}^{j-1})=f_{\boldsymbol{S}^{j-1}}(e^j,i^j),$$

Since e^j and i^j are greedily selected with the best element and location in step j of Algorithm 1, then we have

$$f_{\boldsymbol{S}^{j-1}}(e^j,i^j)\geqslant f_{\boldsymbol{S}^{j-1}}(o^j,\boldsymbol{O}^{j-1}(o^j)).$$

Due to $\boldsymbol{S}^{j-1}\preceq\boldsymbol{O}^{j-\frac{1}{2}}$, then

$$f_{\boldsymbol{S}^{j-1}}(o^j,\boldsymbol{O}^{j-1}(o^j))\geqslant f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j)) \tag{2}$$

By using the property of the pairwise monotonicity of f and $n^j\neq i^j$, we have

$$f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,i^j)+f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,n^j)\geqslant 0.$$

Therefore, applying this relationship to the right hand of inequality (2), we get

$$f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j))\geqslant f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j))-f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,i^j)-f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,n^j).$$

And using $\boldsymbol{S}^{j-1}\preceq\boldsymbol{O}^{j-\frac{1}{2}}$ again, we can obtain

$$f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j))\geqslant f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j))-f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,i^j)-f_{\boldsymbol{S}^{j-1}}(e^j,n^j).$$

Applying the condition that e^j and i^j are the best once more, we get

$$f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j))\geqslant f_{\boldsymbol{O}^{j-\frac{1}{2}}}(o^j,\boldsymbol{O}^{j-1}(o^j))-f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,i^j)-f_{\boldsymbol{S}^{j-1}}(e^j,i^j).$$

That is, the right side of the above inequality is equivalent to

$$f(\boldsymbol{O}^{j-1}) - f(\boldsymbol{O}^{j-\frac{1}{2}}) - f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j, i^j) - f_{\boldsymbol{S}^{j-1}}(e^j, i^j).$$

After sorting out all the above inequalities, we can get

$$\begin{aligned} &2f_{\boldsymbol{S}^{j-1}}(e^j, i^j) \geqslant \\ &f(\boldsymbol{O}^{j-1}) - f(\boldsymbol{O}^{j-\frac{1}{2}}) - f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j, i^j) = \\ &f(\boldsymbol{O}^{j-1}) - f(\boldsymbol{O}^{j-\frac{1}{2}}) - f(\boldsymbol{O}^j) + f(\boldsymbol{O}^{j-\frac{1}{2}}) = \\ &f(\boldsymbol{O}^{j-1}) - f(\boldsymbol{O}^j). \end{aligned}$$

Thus,

$$2(f(\boldsymbol{S}^j) - f(\boldsymbol{S}^{j-1})) \geqslant f(\boldsymbol{O}^{j-1}) - f(\boldsymbol{O}^j).$$

By summing the two sides of the above inequalion, we get the following results:

$$\begin{aligned} f(\boldsymbol{O}) - f(\boldsymbol{S}) = \sum_{j=1}^{r}(f(\boldsymbol{O}^{j-1}) - f(\boldsymbol{O}^j)) \leqslant \\ 2\sum_{j=1}^{r}(f(\boldsymbol{S}^j) - f(\boldsymbol{S}^{j-1})) = 2f(\boldsymbol{S}). \end{aligned}$$

Finally, we draw the following conclusion:

$$f(\boldsymbol{S}) \geqslant \frac{1}{3} f(\boldsymbol{O}).$$

■

In the next subsection, we will design a randomized algorithm in Algorithm 2. The complexity of this randomized algorithm is $O(|V|(a\log\frac{r}{\varepsilon_1} + kb\log\frac{r}{\varepsilon_2})\log r)$, which is lower than that of the deterministic Algorithm 1 in this section, but it is possible to achieve the same approximation ratio with probability at least $1-\varepsilon$, where $\varepsilon = \max\{\varepsilon_1, \varepsilon_2\}$.

3.2 Randomized algorithm for nMkSfM

In order to reduce the complexity of the deterministic algorithm for the non-monotone case, we adopt the uniform random sampling, which is shown as Algorithm 2. In this algorithm, we construct two sets R_1^j and R_2^j by random sampling, and their sizes are r_1^j and r_2^j respectively, where

$$r_1^j = |R_1^j| = \min\left\{\frac{|V| - j + 1}{r - j + 1}\log\left(\frac{r}{\varepsilon_1}\right), |V|\right\},$$

$$r_2^j = |R_2^j| = \min\left\{\frac{r_1^j - j + 1}{r - j + 1}\log\left(\frac{r}{\varepsilon_2}\right), |V(\boldsymbol{S}^j)|\right\}.$$

There are two main differences between deterministic Algorithm 1 and randomized Algorithm 2. Firstly, the randomized algorithm randomly selects some elements from V to form R_1^j, and reduces the elements available for selection when constructing independent sets by adding R_1^j as the set. Then when constructing independent set $V(\boldsymbol{S}^j)$, the elements in it are selected from $R_1^j \setminus S(\boldsymbol{S}^{j-1})$ instead of $V \setminus S(\boldsymbol{S}^{j-1})$. That is, $V(\boldsymbol{S}^j) = \{e \in R_1^j \setminus S(\boldsymbol{S}^{j-1}) \mid S(\boldsymbol{S}^{j-1}) \cup \{e\} \in \mathcal{F}\}$. Secondly, when $V(\boldsymbol{S}^j)$ is not empty, some elements are randomly selected from $V(\boldsymbol{S}^j)$ to form R_2^j. The appearance of R_2^j greatly reduces the number of elements available for greedy selection and improves the quality of selected elements. Based on the above analysis, we can draw the following conclusions.

Algorithm 2 Randomized algorithm for nMkSfM

Require: Non-monotone k-submodular function $f:(k+1)^V \to \mathbf{R}_+$, matroid $(V, \mathcal{F})$ with bases $\mathcal{B}$, rank r, and two failed probabilities ε_1 and ε_2

Ensure: Vector $\boldsymbol{S}$ with $S(\boldsymbol{S}) \in \mathcal{B}$

1: $\boldsymbol{S}^0 \leftarrow \varnothing$;

2: **for** $j = 1$ to r **do**

3: $R_1^j \leftarrow$ a random subset uniformly chosen from V with size $\min\left\{\frac{|V| - j + 1}{r - j + 1}\log\left(\frac{r}{\varepsilon_1}\right), |V|\right\}$;

4: Construct $V(\boldsymbol{S}^j) := \{e \in R_1^j \setminus S(\boldsymbol{S}^{j-1}) \mid S(\boldsymbol{S}^{j-1}) \cup \{e\} \in \mathcal{F}\}$ by using the independence oracle;

5: $R_2^j \leftarrow$ a subset picked uniformly from $V(\boldsymbol{S}^j)$, whose size is $\min\left\{\frac{r_1^j - j + 1}{r - j + 1}\log\left(\frac{r}{\varepsilon_2}\right), |V(\boldsymbol{S}^j)|\right\}$;

6: $(e, i) \leftarrow \arg\max_{e \in R_2^j, i \in [k]} f(\boldsymbol{S}^{j-1} + (e, i))$;

7: $\boldsymbol{S}^j \leftarrow \boldsymbol{S}^{j-1} + (e, i)$;

8: **end for**

9: **return** $\boldsymbol{S}$

Lemma 3 We can get $\Pr(V(\boldsymbol{S}^j) \neq \varnothing) \geqslant 1 - \varepsilon_1$, likewise, we can also obtain $\Pr(R_2^j \cap L^j \neq \varnothing) \geqslant 1 - \varepsilon_2$ for every $0 \leqslant j \leqslant r$.

Proof First, for $0 \leqslant j \leqslant r$, if $r_1^j = |R_1^j| = |V|$, there must be an element e in $V \setminus S(\boldsymbol{S}^{j-1})$ to make $S(\boldsymbol{S}^{j-1}) \cup \{e\} \in \mathcal{F}$. Then $\Pr(V(\boldsymbol{S}^j) \neq \varnothing) = 1$. Then we can assume that $|R_1^j| < |V|$ for some j. In fact, in this case, we have

$$\begin{aligned} &\Pr(V(\boldsymbol{S}^j) = \varnothing) = \\ &[\Pr(\{e\} \cup S(\boldsymbol{S}^{j-1}) \notin \mathcal{F})]^{r_1^j} = \\ &[1 - \Pr(\{e\} \cup S(\boldsymbol{S}^{j-1}) \in \mathcal{F})]^{r_1^j}. \end{aligned}$$

When we construct $V(\boldsymbol{S}^j)$, there are $r_1^j - (j-1)$ elements e that can be selected, and there are at least $r - (j-1)$ elements that meet the condition of

$\{e\}\cup S(\boldsymbol{S}^{j-1})\in\mathcal{F}$, so

$$\Pr(\{e\}\cup S(\boldsymbol{S}^{j-1})\in\mathcal{F})\geqslant\frac{r-(j-1)}{r_1^j-(j-1)}\geqslant\frac{r-j+1}{|V|-j+1}.$$

Thus, we have

$$\Pr(V(\boldsymbol{S}^j)=\varnothing)\leqslant\left(1-\frac{r-j+1}{|V|-j+1}\right)^{r_1^j}\leqslant$$
$$\exp\left\{-\frac{r-j+1}{|V|-j+1}\frac{|V|-j+1}{r-j+1}\log\frac{r}{\varepsilon_1}\right\}\leqslant\varepsilon_1.$$

Therefore, the probability of $V(\boldsymbol{S}^j)\neq\varnothing$ is at least $1-\varepsilon_1$.

The following part uses the similar way to prove that the probability of $R_2^j\cap L^j\neq\varnothing$ is at least $1-\varepsilon_2$. For $0\leqslant j\leqslant r$, if R_2^j and $V(\boldsymbol{S}^j)$ have the same size, and $V(\boldsymbol{S}^j)$ is not an empty set, obviously there is $R_2^j\cap L^j\neq\varnothing$, then $\Pr[R_2^j\cap L^j\neq\varnothing]=1$. Otherwise $|R_2^j|<|V(\boldsymbol{S}^j)|$ for each element in R_2^j. If the probability of its occurrence in L^j is $p=\frac{|L^j|}{|V(\boldsymbol{S}^j)|}\geqslant\frac{r-j+1}{r_1^j-j+1}$, then we have

$$\Pr[R_2^j\cap L^j=\varnothing]=(1-p)^{r_2^j}\leqslant\left(1-\frac{r-j+1}{r_1^j-j+1}\right)^{r_2^j}\leqslant$$
$$\exp\left\{-\frac{r-j+1}{r_1^j-j+1}\frac{r_1^j-j+1}{r-j+1}\log\frac{r}{\varepsilon_2}\right\}\leqslant\varepsilon_2.$$

■

In the following part, we will give the proof of Theorem 3 based on Lemma 2, which shows that Algorithm 2 does not fail with high probability.

Theorem 3 For nMkSfM, we can obtain a 1/3-approximation solution from Algorithm 2 in complexity of $O\left(|V|\left(a\log\frac{r}{\varepsilon_1}+kb\log\frac{r}{\varepsilon_2}\right)\log r\right)$ with the probability at least $1-\varepsilon$, where $\varepsilon=\max\{\varepsilon_1,\varepsilon_2\}$.

Proof For all $j\in[r]$, we will construct a sequence $\boldsymbol{O}^0=\boldsymbol{O}$, $\boldsymbol{O}^1,\ldots,\boldsymbol{O}^r=\boldsymbol{S}$ based on a maximal optimal solution $\boldsymbol{O}$. Denote the symbols e^j, i^j, L^j, $\boldsymbol{S}^j$, $\boldsymbol{O}^j$, $S(\boldsymbol{S}^j)$, and $S(\boldsymbol{O}^j)$ as the same meanings as those in deterministic case (see the proof of Theorem 2). If $V(\boldsymbol{S}^j)$ or $R_2^j\cap L^j$ is empty, we consider that the algorithm fails. (When $V(\boldsymbol{S}^j)$ is an empty set, no more elements can be selected so that the algorithm cannot continue, so it is considered invalid. Under the random algorithm, we want to adopt the same construction mode as the deterministic algorithm, and when $R_2^j\cap L^j$ is empty, o cannot be constructed, then the algorithm is considered invalid.) Suppose both $V(\boldsymbol{S}^j)$ and $R_2^j\cap L^j$ are non-empty, we have the following definitions. We specify that if $e^j\in R_2^j\cap L^j$, then $o^j=e^j$, otherwise, o^j is any element in $R_2^j\cap L^j$. And we let

$$\boldsymbol{O}^{j-\frac{1}{2}}=\boldsymbol{O}^{j-1}-(o^j,\boldsymbol{O}^{j-1}(o^j)),$$
$$\boldsymbol{O}^j=\boldsymbol{O}^{j-\frac{1}{2}}+(e^j,i^j).$$

Based on the construction of the sequence $\boldsymbol{O}^j$ above and just like the deterministic Algorithm 1, if Algorithm 2 does not fail, we have $\boldsymbol{O}^r=\boldsymbol{S}^r$, and can get

$$2(f(\boldsymbol{S}^j)-f(\boldsymbol{S}^{j-1}))\geqslant f(\boldsymbol{O}^{j-1})-f(\boldsymbol{O}^j).$$

Thus,

$$3f(\boldsymbol{S})\geqslant f(\boldsymbol{O}).$$

In Line 4 of Algorithm 2, we need to judge whether $S(\boldsymbol{S}^{j-1})\cup\{e\}$ is an independent set, and there are at most r_1^j elements, so the number of times required for this step is at most

$$a\sum_{j=1}^{r}r_1^j=a\sum_{j=1}^{r}\frac{|V|-j+1}{r-j+1}\log\frac{r}{\varepsilon_1}=$$
$$a\log\frac{r}{\varepsilon_1}\sum_{j=1}^{r}\frac{|V|-r+j}{j}\leqslant$$
$$a|V|\log\frac{r}{\varepsilon_1}\sum_{j=1}^{r}\frac{1}{j}\leqslant$$
$$a|V|\log\frac{r}{\varepsilon_1}\log r.$$

Line 6 in Algorithm 2 needs to calculate the function value of k-submodular function. There are r_2^j elements in total. Each element needs to be selected from k positions, so the maximum number of times required for this step is $kb\sum_{j=1}^{r}r_2^j$. The same operation as the previous step yields

$$kb\sum_{j=1}^{r}r_2^j\leqslant kb|V|\log\frac{r}{\varepsilon_2}\log r.$$

Therefore, the maximum number of times Algorithm 2 runs is

$$a\sum_{j=1}^{r}r_1^j+kb\sum_{j=1}^{r}r_2^j\leqslant|V|\left(a\log\frac{r}{\varepsilon_1}\log r+kb\log\frac{r}{\varepsilon_2}\log r\right).$$

■

4 Streaming Algorithm for Both mMkfSM and nMkfSM

In some practical applications, the amount of relevant data may be much larger than the memory capacity of the computer. In this case, it will be very difficult to apply the above algorithm and analysis to solve this problem. So we try to design an algorithm with higher efficiency to process these large amounts of data[16]. In the following part, we design a streaming Algorithm 3 to solve the problem of maximizing k-submodular function under a matroid constraint. Under some assumption with constant factor $\alpha\in(0,1)$, we obtain a $\alpha/2$-approximation ratio for mMkfSM and a $\alpha/3$-approximation ratio for nMkfSM.

Assumption 1 For MkfSM and a given constant $\alpha\in(0,1)$, there is a maximal optimization solution $\boldsymbol{O}$ satisfies

$$f(\tilde{\boldsymbol{O}})\leqslant(1-\alpha)f(\boldsymbol{O}) \tag{3}$$

where $\tilde{\boldsymbol{O}}\prec\boldsymbol{O}$.

This assumption makes sure that there is a certain amount of descend when any part of $\boldsymbol{O}$ is removing.

Then we design the streaming algorithm in Algorithm 3, which is different from Algorithm 1. Let V be the ground set with a pretend stream ordering and these elements arrive one by one. Since only the information of the current and before elements are known, so the set $V(\boldsymbol{S}^j)$ cannot be constructed. When an element e arrives, in Line 3, we judge whether $S(\boldsymbol{S}^{j-1})\cup\{e\}$ is an independent set. If yes, we find the best position i for this element based on $\boldsymbol{S}^{j-1}$ in Line 4, then adding (e,i) into $\boldsymbol{S}^{j-1}$ to form $\boldsymbol{S}^j$. If not, we will discard this element, then $\boldsymbol{S}^j=\boldsymbol{S}^{j-1}$.

Theorem 4 For mMkfSM and based on Assumption 1, Algorithm 3 returns a $\alpha/2$-approximation solution with the query complexity of $O(|V|a+krb)$ and takes $O(kr)$ memory.

Proof In the next proof process, we still use the idea of changing points to prove the approximation ratio. Before proof, let us make the following explanation. Assume that e^j and i^j represent the j-th element entering the algorithm and the position selected in Algorithm 3, and $\boldsymbol{S}^j$ represents the renewed solution when e^j arrives. If e^j does not meet the condition for forming an independent set in the algorithm, then $\boldsymbol{S}^j=\boldsymbol{S}^{j-1}$. According to Algorithm 3, the support set $S(\boldsymbol{S})$ must be a basis, i.e., it is a maximal solution. So when $|S(\boldsymbol{S})|=r$ (the rank of the matroid), the elements not arriving cannot be added to $\boldsymbol{S}$. We might as well set l as the number of steps used when $S(\boldsymbol{S})$ reaches a basis, and finally the output solution $\boldsymbol{S}=\boldsymbol{S}^l$. Let $\boldsymbol{O}$ be an optimal solution satisfying Assumption 1. Next, we will construct a sequence $\boldsymbol{O}^0=\boldsymbol{O},\boldsymbol{O}^1,\ldots,\boldsymbol{O}^l$ such that $\boldsymbol{S}^l\preceq\boldsymbol{O}^l$. Our construction method is divided into two cases:

(1) When e^j satisfies the independent set condition, we construct it in the following way. If $e^j\in S(\boldsymbol{O}^{j-1})$, then

$$\boldsymbol{O}^{j-\frac{1}{2}}=\begin{cases}\boldsymbol{O}^{j-1}\sqcup\boldsymbol{S}^j, & \text{if } \boldsymbol{O}^{j-1}\left(e^j\right)\neq i^j;\\ \boldsymbol{O}^{j-1}-\left(e^j,\boldsymbol{O}^{j-1}\left(e^j\right)\right), & \text{otherwise.}\end{cases}$$

Otherwise, i.e., $e^j\notin S(\boldsymbol{O}^{j-1})$, let

$$\boldsymbol{O}^{j-\frac{1}{2}}=\boldsymbol{O}^{j-1},$$

$$\boldsymbol{O}^j:=\boldsymbol{O}^{j-\frac{1}{2}}+(e^j,i^j).$$

(2) When e^j does not satisfy the independent set condition, we make

$$\boldsymbol{O}^{j-\frac{1}{2}}=\boldsymbol{O}^j=\boldsymbol{O}^{j-1}.$$

Next we give proof according to the different cases. In Case (1), we can draw the conclusion of $\boldsymbol{S}^{j-1}\preceq\boldsymbol{O}^{j-\frac{1}{2}}$ and $\boldsymbol{S}^l\preceq\boldsymbol{O}^l$ from the construction of the above sequence. Applying these two conclusions and k-submodularity, we have

$$\begin{aligned}&f(\boldsymbol{O}^{j-1})-f(\boldsymbol{O}^j)=\\&f(\boldsymbol{O}^{j-1})-f(\boldsymbol{O}^{j-\frac{1}{2}})-(f(\boldsymbol{O}^j)-f(\boldsymbol{O}^{j-\frac{1}{2}}))\leqslant\\&f(\boldsymbol{O}^{j-1})-f(\boldsymbol{O}^{j-\frac{1}{2}})=\\&f_{\boldsymbol{O}^{j-\frac{1}{2}}}(e^j,\boldsymbol{O}^{j-1}(e^j))\leqslant\\&f_{\boldsymbol{S}^{j-\frac{1}{2}}}(e^j,\boldsymbol{O}^{j-1}(e^j))\leqslant\\&f_{\boldsymbol{S}^{j-\frac{1}{2}}}(e^j,i^j)=\\&f(\boldsymbol{S}^j)-f(\boldsymbol{S}^{j-1}).\end{aligned}$$

The first inequality above is based on the monotonicity condition of the k-submodular function, the second inequality above uses the partial order

---

Algorithm 3 Streaming algorithm for MkfSM

Require: V, k and k-submodular function f, matroid $(V,\mathcal{F})$ with the bases $\mathcal{B}$

Ensure: vector $\boldsymbol{S}$ with $S(\boldsymbol{S})\in\mathcal{B}$

1: $\boldsymbol{S}^0\leftarrow\varnothing, j\leftarrow 1$;
2: **for** each $e\in V$ **do**
3:     **if** $S(\boldsymbol{S}^{j-1})\cup\{e\}\in\mathcal{F}$ **then**
4:         $i\leftarrow\arg\max_{i\in[k]}f(\boldsymbol{S}^{j-1}+(e,i))$;
5:         $\boldsymbol{S}^j\leftarrow\boldsymbol{S}^{j-1}+(e,i)$;
6:     **end if**
7:     $j\leftarrow j+1$;
8: **end for**
9: **return** $\boldsymbol{S}$

relationship between S^{j-1} and $O^{j-\frac{1}{2}}$, as well as the orthant submodularity of k-submodular function, then the third inequality above is obtained by the greedy algorithm to choose the best position.

For Case (2), since e^j does not meet the condition of independent set, so we have

$$f(O^{j-1}) - f(O^j) = f(S^j) - f(S^{j-1}) = 0.$$

Therefore, in both cases, we have

$$f(O^{j-1}) - f(O^j) \leqslant f(S^j) - f(S^{j-1}).$$

Based on the above analysis, sum both sides of the inequalities above, we obtain

$$\sum_{j=1}^{l}(f(O^{j-1}) - f(O^j)) \leqslant \sum_{j=1}^{l}(f(S^j) - f(S^{j-1})).$$

Thus, we finally get

$$f(O) - f(O^l) \leqslant f(S^l) \tag{4}$$

In this construction mode, we cannot get $O^l = S^l$, so Formula (4) cannot directly show the relationship between $f(O)$ and $f(S)$. We need to further analyze it through the following analysis based on Assumption 1. Let us use $\widetilde{O} \preceq O$ to denote the set of elements not selected into S^l. And then $O^l = \widetilde{O} \sqcup S^l$. Moreover, we have $\widetilde{O} \neq O$ by the construction of the above sequences. Thus by Assumption 1, we have

$$f(\widetilde{O}) \leqslant (1-\alpha) f(O) \tag{5}$$

Moreover, from the definition of k-submodular function, we have

$$f(O^l) = f(\widetilde{O} \sqcup S^l) \leqslant f(\widetilde{O}) + f(S^l).$$

Inequality (5) is combined with above ineqatily to obtain

$$f(O^l) \leqslant (1-\alpha) f(O) + f(S^l).$$

At this point, an upper bound of $f(O^l)$ is found. Combining inequality (4), we have

$$f(S) \geqslant \frac{\alpha}{2} f(O).$$

■

By replacing the monotonicity by pairwise monotonicity, we can get the following corollary.

Corollary 1 For nMkfSM and based on Assumption 1, Algorithm 3 returns a $\alpha/3$-approximation solution with the query complexity of $O(|V|a + krb)$ and takes $O(kr)$ memory.

Remark The approximation ratios of the streaming algorithm designed in this paper are presented with a constant factor α, which represents the decrease when some elements in the optimal solution are removed. As the decrease increases, the approximation ratios will be better. Moreover, the memory complexity is nice. However, the performance guarantee is close to $1/2$ as the best case for mMkfSM and $1/3$ for nMkfSM.

5 Conclusion

In this paper, we have solved the problem of maximizing the k-submodular function under a matroid constraint. At beginning, we design a deterministic algorithm with approximation ratio of $1/3$ for the non-monotone k-submodular function. Later, we also design a randomized algorithm to reduce the complexity of this algorithm. Finally, we give a streaming algorithm for both monotone and non-monotone k-submodular functions. In addition, we plan to improve the streaming algorithm depending on the property of basis of a matroid.

Acknowledgment

This work was supported by the Natural Science Foundation of Shandong Province of China (No. ZR2020MA029).

References

[1] Y. Sun, Y. Liu, and M. Li, Maximization of k-submodular function with a matroid constraint, in *Proc. 17th Annual Conference of Theory and Applications of Models of Computation*, Tianjin, China, 2022, pp. 1−10.

[2] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, Maximizing a monotone submodular function subject to a matroid constraint, *SIAM J. Comput.*, vol. 40, no. 6, pp. 1740–1766, 2011.

[3] J. Ward and S. Živný, Maximizing k-submodular functions and beyond, *ACM Trans. Algorithms*, vol. 12, no. 4, pp. 1–26, 2016.

[4] S. Iwata, S. Tanigawa, and Y. Yoshida, Improved approximation algorithms for k-submodular function maximization, in *Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, Arlington, VA, USA, 2016, pp. 404–413.

[5] H. Oshima, Improved randomized algorithm for k-submodular function maximization, *SIAM J. Discrete Math.*, vol. 35, no. 1, pp. 1–22, 2021.

[6] N. Ohsaka and Y. Yoshida, Monotone k-submodular function maximization with size constraints, in *Proc. 28th International Conference on Neural Information Processing Systems*, Montreal, Canada, 2015, pp. 694–702.

[7] L. N. Nguyen and M. T. Thai, Streaming k-submodular maximization under noise subject to size constraint, in *Proc. 37th International Conference on Machine Learning*, Virtual Event, 2020, pp. 7338–7347.

[8] Z. Tang, C. Wang, and H. Chan, On maximizing a

monotone k-submodular function under a knapsack constraint, *Oper. Res. Lett.*, vol. 50, no. 1, pp. 28–31, 2022.

[9] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, Maximizing a monotone submodular function subject to a matroid constraint, *SIAM J. Comput.*, vol. 40, no. 6, pp. 1740–1766, 2011.

[10] S. Fujishige and S. Iwata, Bisubmodular function minimization, *SIAM J. Discrete Math.*, vol. 19, no. 4, pp. 1065–1073, 2005.

[11] A. Huber and V. Kolmogorov, Towards minimizing *k*-submodular functions, in *Proc. of the Second international conference on Combinatorial Optimization*, Athens, Greece, 2012, pp. 451–462.

[12] C. Pham, Q. Vu, D. Ha, and T. Nguyen, Streaming algorithms for budgeted *k*-submodular maximization problem, in *Proc. 10th International Conference on Computational Data and Social Networks*, Virtual Event, 2021, pp. 27–38.

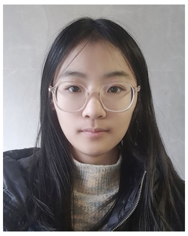
[13] A. Rafiey and Y. Yoshida, Fast and private submodular and *k*-submodular functions maximization with matroid constraints, in *Proc. 37th International Conference on Machine Learning*, Virtual Event, 2020, pp. 7887–7897.

[14] Z. Tang, C. Wang, and H. Chan, Monotone *k*-submodular secretary problems: Cardinality and knapsack constraints, *Theor. Comput. Sci.*, vol. 921, pp. 86–99, 2022.

[15] B. Korte and J. Vygen, *Combinatorial Optimization*: *Theory and Algorithms*. Berlin, Germany: Springer, 2012.

[16] C. Chekuri, S. Gupta, and K. Quanrud, Streaming algorithms for submodular function maximization, in *Proc. 42nd International Colloquium*, *ICALP 2015*, Kyoto, Japan, 2015, pp. 318–330.

Yuezhu Liu is currently an undergraduate student at School of Mathematics and Statistics, Shandong Normal University, China. Her main research interest is the algorithms for numerical optimization.



Yunjing Sun is currently an undergraduate student at School of Mathematics and Statistics, Shandong Normal University, China. Her main research interest is the algorithms for numerical optimization.

Min Li received the PhD degree in applied mathematics from Coimbra University, Portugal in 2013. She is currently an associate professor at School of Mathematics and Statistics, Shandong Normal University, China. Her research interests include combinatorial optimization, robust optimization, and mixed-integer programming.