Learning Submodular Sequencing from Samples

Jing Yuan[1], Qi Cai[1], Xin Gao[1], and Shaojie Tang[2]✉

[1] Department of Computer Science and Engineering, University of North Texas
[2] Center for AI Business Innovation, Department of Management Science and Systems, University at Buffalo shaojiet@buffalo.edu

Abstract. This paper addresses the problem of sequential submodular maximization: selecting and ranking items in a sequence to optimize some composite submodular function. In contrast to most of the previous works, which assume complete knowledge of the utility function, we assume that we are given only a set of samples. Each sample includes a random sequence of items and its associated utility. We present an algorithm that, given polynomially many samples drawn from a two-stage uniform distribution, achieves an approximation ratio dependent on the curvature of individual submodular functions. Our results apply to a wide variety of real-world scenarios, such as ranking products in online retail platforms, where complete knowledge of the utility function is often impossible to obtain. Our algorithm gives an empirically useful solution in such contexts, thus proving that limited data can be of great use in sequencing tasks. From a technical perspective, our results extend prior work on "optimization from samples" by generalizing from optimizing a set function to a sequence-dependent function.

Keywords: Optimization from samples · Submodular sequencing · Approximation algorithms.

1 Introduction

Submodular optimization is one of the most important problems in machine learning, with applications in sparse reconstruction [10], data summarization [14], active learning [13, 19], and viral marketing [17]. Most of the existing work deals with the problem of selecting a subset of items that maximizes some submodular function. However, many real applications require not only the selection of items, but also their ranking in a certain order [3, 21, 18]. This paper focuses on one such problem, termed *sequential submodular maximization* [2, 22, 20]. The problem's input consists of a ground set Ω and k submodular functions, denoted as $f_1, \cdots, f_k : 2^{\Omega} \rightarrow \mathbb{R}^+$. Our objective is to select a sequence of k items, denoted as $\pi = \{\pi_1, \cdots, \pi_k\}$, from Ω, in order to maximize $F(\pi) \stackrel{\text{def}}{=} \sum_{t \in [k]} f_t(\pi_{[t]})$. Here, $\pi_{[t]} \stackrel{\text{def}}{=} \{\pi_1, \cdots, \pi_t\}$ represents the first t items of π. Notably, each function f_t takes the first t items from the ranking sequence π as its input.

This problem captures the position bias in item selection, finding applications in sequential active learning and recommendation systems [22]. One illustrative

example would be product ranking in any of the online retail platforms, like Amazon [2]. Consider Amazon's daily task of selecting and sequencing a list of products, possibly in vertical order, for display to its customers. Customers browse through this list, reaching a certain position, and may proceed to make purchases from the products they view. Then one of the primary objectives of most platforms is to optimize selection and ranking of products to maximize the chance of a purchase. It turns out that this application can be framed as a sequential submodular maximization problem. In this context, parameters of $F(\pi)$ can be interpreted as follows: Let Ω be the set of products and let k be the window size of displayed products. Given a sequence of products π of length k, for each $t \in \{1, 2, \cdots, k\}$, $f_t(\pi_{[t]})$ is the probability of purchase by customers with patience level t, where a customer with a patience level of t would consider viewing the first t products, $\pi_{[t]}$. Typically, f_t is modeled as a submodular function. In this case, $F(\pi)$ captures the expected purchase probability given that a customer is shown the sequence of products π.

While sequential submodular maximization has been extensively explored in the literature [2, 22, 20], existing studies typically assume complete knowledge of $f_1, \cdots, f_k : 2^{\Omega} \to \mathbb{R}^+$ and consequently F. However, this assumption is often unrealistic. For instance, in the aforementioned context of recommendation systems, accurately estimating the purchase probability for every product set is often extremely challenging, if not impossible. Instead, a more realistic scenario involves the platform gathering a potentially extensive dataset comprising browsing histories. Each record (a.k.a. sample) within this dataset includes the sequence of displayed products along with customer feedback. For instance, a record could look like this: {*Sequence*: Product A, Product B; *Feedback*: B was purchased}. Consequently, the platform aims to identify the best sequence of products based on the samples drawn from some distribution. This problem is highly non-trivial since the platform does not have direct access to the original utility function F, making the existing result on submodular sequencing inapplicable. It has been demonstrated that optimizing a set function from samples is generally impossible, even if the set function is a coverage function [7]. Our challenge is compounded by the fact that our function F is defined over a sequence, rather than a set, of items.

Fortunately, in practice, we often encounter submodular functions that may demonstrate more favorable behavior. To this end, we introduce a notation called *curvature* [6]. Intuitively, curvature measures the deviation of a given function from a modular function. Specifically, we say a submodular function f has curvature $c \in [0, 1]$ if $f(i \mid S) \geq (1 - c)f(\{i\})$ for any $S \subseteq \Omega$ and $i \notin S$. Here $f(i \mid S) \stackrel{\text{def}}{=} f(S \cup \{i\}) - f(S)$ denotes the marginal utility of an item $i \in \Omega$ on top of a set of items $S \subseteq \Omega$. Hence, if f is a modular function, it has a curvature of 0. In general, the complexity of optimizing a submodular function often hinges on the curvature of the focal function. That is, the instances of submodular optimization become challenging typically only when the curvature is unbounded, i.e., c close to 1. In this paper, we study how to optimize a function $F(\pi) = \sum_{t \in [k]} f_t(\pi_{[t]})$ from samples when the curvature of each individual

function f_t is bounded. Our contribution is the development of an approximation algorithm that draws polynomially-many samples from a natural two-stage uniform distribution over feasible sequences and achieves an approximation ratio dependent on the curvature.

2 Related Work

While submodular maximization has been extensively studied in the literature [15], most existing studies assume that the submodular function to be optimized is known. Recently, there has been a line of research focused on learning a submodular function from samples [5, 11, 4, 12], aiming to construct a function that approximates those from which the samples were collected. It has been shown that monotone submodular functions can be approximately learned from samples drawn from a specific distribution [5]. However, it has also been demonstrated that even if an objective function is learnable from samples, optimization for such a function might still be impossible [7]. Despite these negative results, there exists a series of studies [6, 8, 9] that develop effective algorithms to optimize submodular functions from samples.

In our paper, we focus on an important variant of submodular optimization known as sequential submodular maximization. The objective of sequential submodular maximization is more general than simply selecting a subset of items: it involves jointly selecting and sequencing items. [2] studied this problem with monotone and submodular functions. [20] extended this study to the non-monotone setting. However, all these studies assume a known utility function. Our research builds on and extends these studies by expanding the "learning-from-samples" framework [6] from set functions to sequence functions. Moreover, we identify a gap in the analysis presented in existing studies; more details are provided in Section 5.1.

3 Preliminaries and Problem Formulation

Throughout the remainder of this paper, let $[m] = \{0, 1, 2, \ldots, m\}$ for any positive integer m. Given a function f, let $f(i \mid S) \stackrel{\text{def}}{=} f(S \cup \{i\}) - f(S)$ denote the marginal utility of an item $i \in \Omega$ on top of a set of items $S \subseteq \Omega$. We say a function f is submodular if and only if for any two sets X and Y such that $X \subseteq Y$ and any item $i \notin Y$, $f(i \mid X) \geq f(i \mid Y)$. Moreover, we say a submodular function f has curvature $c \in [0, 1]$ if $f(i \mid S) \geq (1 - c)f(\{i\})$ for any $S \subseteq \Omega$ and $i \notin S$.

3.1 Utility Function

Now we are ready to introduce our research problem. Given k submodular functions $f_1, \cdots, f_k : 2^{\Omega} \to \mathbb{R}^+$, the sequential submodular maximization problem

aims to find a sequence $\pi = \{\pi_1, \cdots, \pi_k\}$ from a ground set Ω that maximizes the value of $F(\pi)$. Here,

$$F(\pi) \stackrel{\text{def}}{=} \sum_{t \in [k]} f_t(\pi_{[t]}), \tag{1}$$

where $\pi_{[t]} \stackrel{\text{def}}{=} \{\pi_1, \cdots, \pi_t\}$ represents the first t items of π. That is, each function f_t takes the first t items from π as its input. Throughout this paper, we use the notation π to denote both a sequence of items and the set of items in that sequence.

Existing studies on sequential submodular maximization all assume that $f_1, \cdots, f_k$ are known in advance, however, in our setting, we do not have direct access to those functions. Instead, we rely on a dataset comprising observations $(\pi, \phi(\pi))$, where in each sample $(\pi, \phi(\pi))$, π denotes a feasible sequence and $\phi(\pi)$ denotes the observed utility of π. It is important to note that the observed utility of a sequence π may be subject to randomness, rendering $\phi(\pi)$ a realization of this stochastic variable. Take, for instance, the product sequencing example outlined in the introduction: $F(\pi)$ denotes the likelihood of purchase from a product sequence π. Here, the observed utility $\phi(\pi)$ of π is a binary variable, with $\phi(\pi) = 1$ denoting a purchase and $\phi(\pi) = 0$ denoting a non-purchase. In this example, randomness stems from two sources: the user's type, characterized by their patience level (i.e., a random function f_t is sampled from $\{f_1, \cdots, f_k\}$), and the probabilistic decision-making process of whether the user will purchase a product from π (note that f_t represents only the *aggregated* likelihood of purchase).

3.2 Problem Formulation

Our objective is to compute a sequence $\pi = \{\pi_1, \cdots, \pi_k\}$ that maximizes the value of $F(\pi)$ based on the samples drawn from a distribution $\mathcal{D}$. We say this problem is γ-optimizable with respect to a distribution $\mathcal{D}$, if there exists an algorithm which, given polynomially many samples drawn from $\mathcal{D}$, returns with high probability a sequence π of size at most k such that $F(\pi) \geq \gamma F(\pi^*)$ where π^* denotes the optimal solution of this problem.

As with the standard `PMAC`-learning framework, we fix a distribution called *two-stage uniform sampling* and assume that samples are drawn i.i.d. from this distribution. In particular, two-stage uniform sampling works in two stages: In the first stage, a length t is randomly selected from the set $\{1, \cdots, k\}$ with uniform probability. Subsequently, a sequence of length t is randomly chosen, and its realized utility is observed. We would like to clarify that in the first stage, we randomly select the sequence length displayed to the user. However, we do not assume that the attention span of different users (e.g., the actual number of items they browse) follows a uniformly random distribution. In the following, we present an approximation algorithm with respect to this distribution.

Algorithm 1 Sequencing-from-Samples (SeqSamp)

Solve **P.1** to obtain π^s
if $(1-c)^2 \geq \alpha \cdot \frac{1-c}{1+c-c^2}$ **then**
  $\pi^\diamond \leftarrow \pi^s$
else if $(1-c)\sum_{t\in\{1,\cdots,k\}} \widetilde{\Delta}(\pi^s_t, t-1) \geq \text{avg}(\Phi_k)$ **then**
  $\pi^\diamond \leftarrow \pi^s$
else
  $\pi^\diamond \leftarrow$ a random sequence of length k
return $\pi^\diamond$;

4 Algorithm Design

Our algorithm first estimates the expected marginal contribution $\Delta(i,t)$ of each item $i \in \Omega$ to a uniformly random sequence of size t, that does not contain i, for every item $i \in \Omega$ and every size $t \in [k-1]$. A formal definition of $\Delta(i,t)$ is given by:

$$\Delta(i,t) = \mathbb{E}_{\Pi_{t+1,i}}\big[F(\Pi_{t+1,i})\big] - \mathbb{E}_{\Pi_{t,-i}}\big[F(\Pi_{t,-i})\big] \tag{2}$$

where $\Pi_{t+1,i}$ denotes a random sequence of length $t+1$ with i being placed at the last slot and $\Pi_{t,-i}$ denotes a random sequence of length t that does not contain i. Unfortunately, one can not access the value of either $\mathbb{E}_{\Pi_{t+1,i}}\big[F(\Pi_{t+1,i})\big]$ or $\mathbb{E}_{\Pi_{t,-i}}\big[F(\Pi_{t,-i})\big]$ directly. To estimate these values, we draw inspiration from a technique proposed in [6], estimating the value of $\mathbb{E}_{\Pi_{t+1,i}}\big[F(\Pi_{t+1,i})\big]$ and $\mathbb{E}_{\Pi_{t,-i}}\big[F(\Pi_{t,-i})\big]$ using $\texttt{avg}(\Phi_{t+1,i})$ and $\texttt{avg}(\Phi_{t,-i})$ respectively. Here, $\texttt{avg}(\Phi_{t+1,i})$ represents the average (observed) utility of all samples where the length is $t+1$ and i is placed at the last slot, while $\texttt{avg}(\Phi_{t,-i})$ denotes the average (observed) utility of all samples with length t that do not contain i. Then we use

$$\widetilde{\Delta}(i,t) = \texttt{avg}(\Phi_{t+1,i}) - \texttt{avg}(\Phi_{t,-i}) \tag{3}$$

as an estimation of $\Delta(i,t)$ for all $i \in \Omega$ and $t \in [k-1]$.

In the following, we treat $\widetilde{\Delta}(i,t)$ as the weight of placing i at position $t+1$. As a subroutine of our algorithm, we aim to find a feasible sequence that maximizes the total weight. This objective can be reframed as a *maximum weight matching problem*. Specifically, we introduce a set of item-position pairs $\Psi = \{(i,t) \mid i \in \Omega, t \in \{1,2,\cdots,k\}\}$, where selecting a pair (i,t) indicates assigning item i to position t. Consequently, the task of identifying a feasible sequence maximizing the total weight is transformed into the following maximum weight matching problem.

P.1 $\max_{\psi\subseteq\Psi:|\psi|\leq k} \sum_{(i,t)\in\psi} \widetilde{\Delta}(i,t-1)$
subject to $|\psi \cap \Psi_i| \leq 1$ for all $i \in \Omega$; $|\psi \cap \Psi_t| = 1$ for all $t \in [k-1]$.

Here $\Psi_i = \{(i,t) \mid t \in \{1,2,\cdots,k\}\}$ denote the set of all item-position pairs involving item i, and $\Psi_t = \{(i,t) \mid i \in \Omega\}$ denote the set of all item-position pairs

involving position t. The condition "$|\psi \cap \Psi_i| \leq 1$ for all $i \in \Omega$" ensures that each item appears at most once in a sequence, while "$|\psi \cap \Psi_t| = 1$ for all $t \in [k-1]$" ensures that each position contains exactly one item. It is straightforward to confirm the existence of a one-to-one correspondence between feasible sequences and feasible solutions of **P.1**. That is, given a feasible solution ψ of **P.1**, one can construct a feasible sequence such that for each $i \in \Omega$ and $t \in \{1, 2, \cdots, k\}$, item i is placed in position t if and only if $(i, t) \in \psi$.

Because **P.1** is a classic maximum weighted matching problem, it can be solved efficiently in polynomial time [16]. Now we are ready to present our final algorithm SeqSamp (as listed in Algorithm 1). Assume all individual functions $f_1, \cdots, f_k$ have curvature c, that is, for all $t \in [k]$, we have $f_t(i \mid S) \geq (1-c)f_t(\{i\})$ for any $S \subseteq \Omega$ and $i \notin S$. First, we solve **P.1** optimally, and let π^s denote the sequence corresponding to this solution. Then, we compute the final sequence as follows: If $(1-c)^2 \geq \alpha \cdot \frac{1-c}{1+c-c^2}$, where $\alpha = \frac{n-k}{n} \cdot \frac{n-k-1}{n-1} \cdot \ldots \cdot \frac{n-2k+1}{n-k+1}$, then our algorithm returns π^s as the final solution. Otherwise, if $(1-c)^2 < \alpha \cdot \frac{1-c}{1+c-c^2}$ and $(1-c)\sum_{t\in\{1,\cdots,k\}} \widetilde{\Delta}(\pi^s_t, t-1) \geq \texttt{avg}(\Phi_k)$, then our algorithm still returns π^s as the final solution. Here, $\texttt{avg}(\Phi_k)$ denotes the average utility of all samples with a sequence length of k. Otherwise, our algorithm returns a random sequence of length k as the final solution. We note that the overall design-randomly selecting between a carefully crafted solution and a randomly generated one-is inspired by the framework developed in [6].

4.1 Remark

While our algorithm assumes the curvature of all individual functions is known, this assumption can be relaxed. Specifically, if its exact value is unknown, any upper bound on this value can be used as a surrogate for the curvature without affecting the algorithm's analysis. In the extreme case where the curvature is entirely unknown, one can simply adopt π^s, which yields an approximation ratio of $(1-c)^2$ (an immediate corollary of Lemma 1).

5 Performance Analysis

Let $\pi^\diamond$ be the sequence returned from Algorithm 1, we next analyze the approximation ratio of $\pi^\diamond$, assuming f_t is a monotone submodular function with curvature c for all $t \in \{1, 2, \cdots, k\}$. We first present two technical lemmas. The first lemma derives an approximation ratio for the case when $(1-c)^2 \geq \alpha \cdot \frac{1-c}{1+c-c^2}$, while the second lemma derives an approximation ratio for the remaining cases. The final approximation ratio is the better of these values.

Lemma 1. *Assume f_t is a monotone submodular function with curvature c for all $t \in \{1, 2, \cdots, k\}$, for the case when $(1-c)^2 \geq \alpha \cdot \frac{1-c}{1+c-c^2}$, we have that, with a sufficiently large polynomial number of samples, $F(\pi^\diamond) \geq \big((1-c)^2 - o(1)\big)F(\pi^*)$ where $\alpha = \frac{n-k}{n} \cdot \frac{n-k-1}{n-1} \cdot \ldots \cdot \frac{n-2k+1}{n-k+1}$.*

Proof: According to Line 2 in Algorithm 1, when $(1-c)^2 \geq \alpha \cdot \frac{1-c}{1+c-c^2}$, it returns π^s as $\pi^\diamond$. Here π^s denotes the sequence corresponding to the optimal solution of **P.1**. To prove this lemma, it suffices to show that $F(\pi^s) \geq \big((1-c)^2 - o(1)\big)F(\pi^*)$.

Let $\pi^s = \{e_1, e_2, \cdots, e_k\}$ and $\pi^s_{[t]} = \{e_1, e_2, \cdots, e_t\}$, it follows that

$$
\begin{aligned}
F(\pi^s) &= \sum_{t\in[k-1]} F(\pi^s_{[t+1]}) - F(\pi^s_{[t]}) = \sum_{t\in[k-1]} \sum_{j\in\{t+1,\cdots,k\}} f_j(e_{t+1} \mid \pi^s_{[t]}) \\
&\geq (1-c) \sum_{t\in[k-1]} \sum_{j\in\{t+1,\cdots,k\}} f_j(e_{t+1}) \\
&\geq (1-c) \sum_{t\in[k-1]} \sum_{j\in\{t+1,\cdots,k\}} \mathbb{E}_{R_{t,-e_{t+1}}}\left[f_j(e_{t+1} \mid R_{t,-e_{t+1}})\right] \\
&= (1-c) \sum_{t\in[k-1]} \left(\mathbb{E}_{\Pi_{t+1,e_{t+1}}}\left[F(\Pi_{t+1,e_{t+1}})\right] - \mathbb{E}_{\Pi_{t,-e_{t+1}}}\left[F(\Pi_{t,-e_{t+1}})\right]\right) \\
&= (1-c) \sum_{t\in[k-1]} \Delta(e_{t+1}, t)
\end{aligned}
$$

where $R_{t,-e_{t+1}}$ denotes a random set of size t that excludes item e_{t+1}. The first inequality is by the curvature of f_t and fact that $e_{t+1} \notin \pi^s_{[t]}$ for all $t \in [t-1]$, and the second inequality is by the assumption that f_t is submodular for all $t \in \{1, 2, \cdots, k\}$.

Recall that $\Delta(i,t) = \mathbb{E}_{\Pi_{t+1,i}}\big[F(\Pi_{t+1,i})\big] - \mathbb{E}_{\Pi_{t,-i}}\big[F(\Pi_{t,-i})\big]$ and $\widetilde{\Delta}(i,t) = \texttt{avg}(\Phi_{t+1,i}) - \texttt{avg}(\Phi_{t,-i})$ is an estimation of $\Delta(i,t)$. In the appendix (Lemma 3), we show that with a sufficiently large polynomial number of samples, the estimation $\widetilde{\Delta}(i,t)$ is n^2-close to $\Delta(i,t)$ for all $i \in \Omega$ and $t \in [k-1]$, with high probability, i.e.,

$$
\Delta(i,t) + \frac{\delta}{n^2} \geq \widetilde{\Delta}(i,t) \geq \Delta(i,t) - \frac{\delta}{n^2}. \tag{4}
$$

where $\delta = \max_{\pi:|\pi|\leq k} \phi(\pi)$ denotes the maximum realized value of any sequence with a length of at most k. Recall that in the example of product sequencing, $\phi(\pi) = 1$ indicates a purchase, while $\phi(\pi) = 0$ indicates a non-purchase. Therefore, in this example, $\delta = 1$.

This, together with the previous inequality, implies that

$$
F(\pi^s) \geq (1-c) \sum_{t\in[k-1]} \Delta(e_{t+1}, t) \geq (1-c) \sum_{t\in[k-1]} \widetilde{\Delta}(e_{t+1}, t) - \frac{\delta}{n}. \tag{5}
$$

Recall that $\pi^s = \{e_1, e_2, \cdots, e_k\}$ is the sequence corresponding to the optimal solution of **P.1**, we have

$$
\sum_{t\in[k-1]} \widetilde{\Delta}(e_{t+1}, t) \geq \sum_{t\in[k-1]} \widetilde{\Delta}(e^*_{t+1}, t) \geq \sum_{t\in[k-1]} \Delta(e^*_{t+1}, t) - \frac{\delta}{n}. \tag{6}
$$

Here the second inequality is derived using inequality (4).

In addition, observe that

$$\begin{aligned}
&\sum_{t\in[k-1]} \Delta(e^*_{t+1}, t)\\
&= \sum_{t\in[k-1]} \left(\mathbb{E}_{\Pi_{t+1,e^*_{t+1}}}\left[F(\Pi_{t+1,e^*_{t+1}})\right] - \mathbb{E}_{\Pi_{t,-e^*_{t+1}}}\left[F(\Pi_{t,-e^*_{t+1}})\right]\right)\\
&= \sum_{t\in[k-1]} \sum_{j\in\{t+1,\cdots,k\}} \mathbb{E}_{R_{t,-e^*_{t+1}}}\left[f_j(e^*_{t+1} \mid R_{t,-e^*_{t+1}})\right]\\
&\geq \sum_{t\in[k-1]} \sum_{j\in\{t+1,\cdots,k\}} \mathbb{E}_{R_{t,-e^*_{t+1}}}\left[(1-c)f_j(e^*_{t+1})\right]\\
&= (1-c)\sum_{t\in[k-1]} \sum_{j\in\{t+1,\cdots,k\}} f_j(e^*_{t+1}) \geq (1-c)F(\pi^*)
\end{aligned}$$

where the first inequality is by the curvature of f_t and fact that $e^*_{t+1} \notin R_{t,-e^*_{t+1}}$ for all $t \in [k-1]$, and the second inequality is by the assumption that f_t is submodular for all $t \in \{1, 2, \cdots, k\}$.

This, together with inequality (6), implies that

$$\sum_{t\in[k-1]} \widetilde{\Delta}(e_{t+1}, t) \;\geq \sum_{t\in[k-1]} \Delta(e^*_{t+1}, t) - \frac{\delta}{n} \geq (1-c)F(\pi^*) - \frac{\delta}{n}. \tag{7}$$

Inequalities (5) and (7) imply that

$$F(\pi^s) \geq \big((1-c)^2 - o(1)\big)F(\pi^*). \tag{8}$$

□

We proceed to providing the second technical lemma.

Lemma 2. *Assume f_t is a monotone submodular function with curvature c for all $t \in \{1, 2, \cdots, k\}$, for the case when $(1-c)^2 < \alpha \cdot \frac{1-c}{1+c-c^2}$, we have that, with a sufficiently large polynomial number of samples,*

$$F(\pi^\diamond) \geq \alpha \cdot (\frac{1-c}{1+c-c^2} - o(1))F(\pi^*) \tag{9}$$

where $\alpha = \frac{n-k}{n} \cdot \frac{n-k-1}{n-1} \cdot \ldots \cdot \frac{n-2k+1}{n-k+1}$.

Proof: Let us define a function $F(\pi \uplus \pi^*)$ for a sequence π of length k and an optimal solution π^* as follows:

$$F(\pi \uplus \pi^*) = \sum_{t\in[k]} f_t(\pi_{[t]} \cup \pi^*_{[t]}) \tag{10}$$

Here, $\pi_{[t]}$ (and $\pi^*_{[t]}$) represent all items from π (and π^*) respectively, that are placed up to position t. That is, $\pi \uplus \pi^*$ can be envisioned as a virtual sequence where both π_t and π^*_t are placed at position t for all $t \in \{1, 2, \cdots, k\}$.

Let Π' denote a random sequence of length k that is sampled over items from $\Omega \setminus \pi^*$, and $\Pi'_{[t]}$ denotes the first t items from Π', observe that,

$$\begin{aligned}
&\mathbb{E}_{\Pi'}\big[F(\Pi' \uplus \pi^*) - F(\pi^*)\big] = \mathbb{E}_{\Pi'}\big[\sum_{t\in\{1,2,\cdots,k\}} f_t(\Pi'_{[t]} \cup \pi^*_{[t]}) - \sum_{t\in\{1,2,\cdots,k\}} f_t(\pi^*_{[t]})\big] \\
&= \mathbb{E}_{\Pi'}\big[\sum_{t\in\{1,2,\cdots,k\}} (f_t(\Pi'_{[t]} \cup \pi^*_{[t]}) - f_t(\pi^*_{[t]}))\big] = \mathbb{E}_{\Pi'}\big[\sum_{t\in\{1,2,\cdots,k\}} f_t(\Pi'_{[t]} \mid \pi^*_{[t]})\big] \\
&= \sum_{t\in\{1,2,\cdots,k\}} \mathbb{E}_{\Pi'}\big[f_t(\Pi'_{[t]} \mid \pi^*_{[t]})\big] \geq \sum_{t\in\{1,2,\cdots,k\}} (1-c)\mathbb{E}_{\Pi'}[f_t(\Pi'_{[t]})] \\
&\geq (1-c)\mathbb{E}_{\Pi'}[\sum_{t\in\{1,2,\cdots,k\}} f_t(\Pi'_{[t]})] = (1-c)\mathbb{E}_{\Pi'}[F(\Pi')].
\end{aligned}$$

To establish the first inequality, we utilize the fact that $\Pi'_{[t]}$ is a random set of size t and $\Pi'_{[t]} \subseteq \Omega \setminus \pi^*_{[t]}$. Consequently, this inequality can be derived by substituting $R = \Pi'_{[t]}$ and $S = \pi^*_{[t]}$ into Lemma 4 which is presented in the appendix.

In addition, observe that

$$\begin{aligned}
&F(\pi^*) + \mathbb{E}_{\Pi'}[F(\Pi' \uplus \pi^*)] - F(\pi^*) \\
&= \mathbb{E}_{\Pi'}[F(\Pi')] + \mathbb{E}_{\Pi'}[F(\Pi' \uplus \pi^*)] - \mathbb{E}_{\Pi'}[F(\Pi')]
\end{aligned}$$

and $\sum_{t\in[k-1]} \Delta(e^*_{t+1}, t) \geq \alpha \cdot \mathbb{E}_{\Pi'}\big[F(\Pi' \uplus \pi^*) - F(\Pi')\big]$ where $\alpha = \frac{n-k}{n} \cdot \frac{n-k-1}{n-1} \cdot \ldots \cdot \frac{n-2k+1}{n-k+1}$ (by Lemma 5 in the appendix). We have

$$F(\pi^*) + \mathbb{E}_{\Pi'}[F(\Pi' \uplus \pi^*)] - F(\pi^*) \leq \mathbb{E}_{\Pi'}[F(\Pi')] + \frac{1}{\alpha} \sum_{t\in[k-1]} \Delta(e^*_{t+1}, t).$$

This, together with the previous observation that $\mathbb{E}_{\Pi'}[F(\Pi' \uplus \pi^*) - F(\pi^*)] \geq (1-c)\mathbb{E}_{\Pi'}[F(\Pi')]$, implies that $F(\pi^*) + (1-c)\mathbb{E}_{\Pi'}[F(\Pi')] \leq \mathbb{E}_{\Pi'}[F(\Pi')] + \frac{1}{\alpha}\sum_{t\in[k-1]} \Delta(e^*_{t+1}, t)$. It follows that

$$\sum_{t\in[k-1]} \Delta(e^*_{t+1}, t) \geq \alpha\left(1 - c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)}\right)F(\pi^*). \tag{11}$$

This, together with inequality (7), implies that

$$\begin{aligned}
&(1-c)\sum_{t\in[k-1]} \widetilde{\Delta}(e_{t+1}, t) \geq (1-o(1))(1-c)\sum_{t\in[k-1]} \Delta(e^*_{t+1}, t) \\
&\geq (1-o(1))(1-c)\alpha\left(1 - c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)}\right)F(\pi^*).
\end{aligned} \tag{12}$$

According to Line 4 of Algorithm 1 and inequality (5), when $(1-c)^2 < \alpha \cdot \frac{1-c}{1+c-c^2}$, $\pi^\diamond$ achieves an utility of at least $\max\{(1-o(1))\mathbb{E}_{\Pi}[F(\Pi)], (1-o(1))(1-c)\sum_{t\in[k-1]} \widetilde{\Delta}(e_{t+1}, t)\}$ where Π denotes a random sequence of length k that is sampled over items from Ω. Hence, the approximation ratio of our algorithm

is at least $\max\{(1-o(1))\frac{\mathbb{E}_{\Pi}[F(\Pi)]}{F(\pi^*)}, (1-o(1))\frac{(1-c)\sum_{t\in[k-1]}\widetilde{\Delta}(e_{t+1},t)}{F(\pi^*)}\}$. According to inequality (12), $\frac{(1-c)\sum_{t\in[k-1]}\widetilde{\Delta}(e_{t+1},t)}{F(\pi^*)} \geq (1-o(1))\alpha(1-c)(1-c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)})$. It follows that the approximation ratio of our algorithm is at least $\max\{(1-o(1))\frac{\mathbb{E}_{\Pi}[F(\Pi)]}{F(\pi^*)}, (1-o(1))\alpha(1-c)(1-c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)})\} = (1-o(1))\max\{\frac{\mathbb{E}_{\Pi}[F(\Pi)]}{F(\pi^*)}, \alpha(1-c)(1-c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)})\} \geq (1-o(1))\max\{\frac{\alpha\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)}, \alpha(1-c)(1-c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)})\} = (1-o(1))\alpha\max\{\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)}, (1-c)(1-c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)})\}$ where the inequality is by the observation that the probability that Π is sampled from $\Omega\backslash\pi^*$ is at least $\alpha = \frac{n-k}{n}\cdot\frac{n-k-1}{n-1}\cdot\ldots\cdot\frac{n-2k+1}{n-k+1}$. Observe that $\max\{\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)}, (1-c)(1-c\frac{\mathbb{E}_{\Pi'}[F(\Pi')]}{F(\pi^*)})\}$ is at least $\frac{1-c}{1+c-c^2}$, hence, the approximation of $\pi^\diamond$ is at least $\alpha\cdot\frac{1-c}{1+c-c^2} - o(1)$. □

Combining Lemma 1 and Lemma 2, we have the following theorem.

Theorem 1. *Let $\pi^\diamond$ be the sequence returned from Algorithm 1, assuming f_t is a monotone submodular function with curvature c for all $t \in \{1, 2, \cdots, k\}$, we have that, with a sufficiently large polynomial number of samples,*

$$F(\pi^\diamond) \geq \max\{(1-c)^2 - o(1), \alpha\cdot\frac{1-c}{1+c-c^2} - o(1)\}F(\pi^*) \tag{13}$$

where $\alpha = \frac{n-k}{n}\cdot\frac{n-k-1}{n-1}\cdot\ldots\cdot\frac{n-2k+1}{n-k+1}$.

5.1 Remark

While our study builds on the work of [6] by extending the "learning-from-samples" approach from set functions to sequence functions, there is a potential gap in their original analysis. Specifically, their proof of Lemma 1 relies on the assumption that $f(R \mid S^\star) \geq (1-c)f(R)$, where $S^\star$ is an optimal set solution, R is a uniformly random set of size $k-1$ (with k being the size constraint of the final solution) and c is the curvature of function f. This assumption is, unfortunately, not generally valid; according to the definition of the curvature c, this assumption holds only if $R \cap S^\star = \emptyset$. Our study addresses this issue by introducing the notion of α and further extends their research to a more complex sequence function.

6 Performance Evaluation

We evaluate the performance of the proposed sequential submodular maximization algorithm in the context of assortment optimization, where a sequence of items are selected for display to the users with the goal of maximizing the purchase probability. Given the dynamic nature of user purchase behavior, there is always a challenge to accurately capture the probability of a user purchasing one product from the sequence that is displayed to her. In this case, it is essential to relax the assumption that there is a known function that captures the purchase behavior of each individual user type. To obtain a high quality sequence of items for display, our algorithms merely require a small collection of samples each

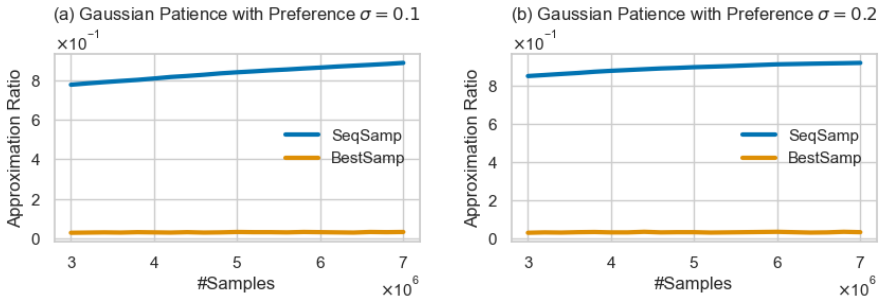

Fig. 1. SeqSamp achieves superior approximation ratio under normal distributed user preferences.

comprises a random sequence of displayed items and its corresponding realized outcome (purchase or no purchase). Our results demonstrate the superiority of our algorithm that lies in its comparable performance to the upper bound. The performance is evaluated in terms of the expected utility with respect to changes in number of samples, user patience level, and item preference distribution.

Experimental Setup. We exploit the widely used Multinomial Logit (MNL) model [1] to capture the underlying user behavior. Our individual function is defined as $f_t(\pi_{[t]}) = \lambda_t * (\sum_{i \in [t]} v_{u,i})/(1 + \sum_{i \in [t]} v_{u,i})$, where $v_{u,i}$ denotes the user u's preference towards item i. Given a sequence π, $f_t(\pi_{[t]})$ captures the purchase probability of a user with a specific patience level t who is willing to browse the first t items $\pi_{[t]}$. Note that our algorithms do not have direct access to this model, which is only used as an underlying model for generating samples and calculating the utility of a solution.

Algorithms. We compare our proposed Sequencing-from-Samples algorithm (labeled as SeqSamp) against two benchmarks, namely, GKF and BestSamp, under various parameter settings. GKF *has access to the underlying choice model* and operates greedily, iteratively selecting the item with the largest marginal gain with respect to $F(\pi)$ in each iteration. BestSamp is a sampling-based algorithm that selects the single best sample with the largest utility. Like SeqSamp, BestSamp does not have access to the choice model, when multiple samples with the same largest value exist, it randomly picks one to break the tie. We use the solution returned by GKF as an upper bound. We obtain the approximation ratio of the expected utility of the sampling-based algorithms to the upper bound.

Parameter Settings. Note that choice models can vary among individuals. We test different user preference distributions to capture this phenomena. Specifically, we follow a normal distribution $\mathcal{N}(\mu, \sigma^2)$ to sample the values for λ_t. We explore the impact of user preference distribution on the performance of the algorithms by varying the value of μ and σ. We also evaluate the impact of

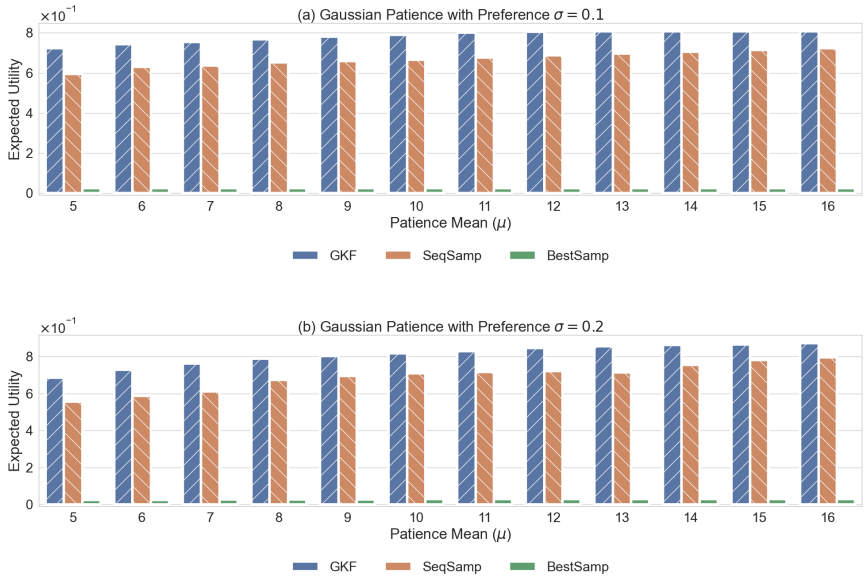

Fig. 2. SeqSamp achieves superior expected utility with respect to varying user patience levels.

user type (patience level t) distribution. Our experiments are run on AMD Ryzen Threadripper PRO Processor with 128GB RAM. We run each set of experiments 100 rounds and the average results are reported below. Complete source code is provided in supplementary materials.

Experimental Results. In the first set of experiments, we measure the performance of the algorithms in terms of their approximation ratio to the upper bound with respect to various sizes of samples. For each item i, we set $v_{u,i}$, users' preference towards i, to follow a normal distribution with μ ranging from 0.01 to 0.5 with $\sigma = 0.1$ and 0.2, respectively. We vary the value of σ to test the reliability of our algorithm when different users' have diverse preferences towards the same item. A larger σ indicates more diverse opinions. We set the number of items to be $n = 10^4$ and number of user types to be $k = 20$. The average user's patience is set to be 15. As shown in Figure 1(a), for the case of $\sigma = 0.1$, we observe that with 3.5×10^6 samples, SeqSamp already achieves a 80% approximation ratio to the upper bound. With 7×10^6 samples, SeqSamp achieves a 88.6% approximation ratio. As shown in Figure 1(b), when users' preferences are more diverse, SeqSamp achieves a 91.95% approximation ratio with 7×10^6 samples. All of our experiments have the solutions returned within one minute. Note that for our main theoretical result to hold (i.e., Lemma 3), a sample size on the order of n^8 (i.e., 10^{32}) is required. This highlights that,

in practice, our approach requires significantly fewer samples than our analysis suggests.

In the second set of experiments, we measure the performance of the algorithms in terms of their expected utility with respect to various user patience levels, ranging from 5 to 16. We follow the parameter settings as above. Figure 2 shows the results for SepSamp and BestSamp based on 7×10^6 samples, compared to the upper bound (GKF). We observe that as users are willing to browse more items on the average, the expected utility tends to increase with a diminishing marginal gain. SeqSamp achieves a comparable performance to the upper bound and significantly outperforms BestSamp, both for $\sigma = 0.1$ (Figure 2(a)) and $\sigma = 0.2$ (Figure 2(b)). This again demonstrates the superiority of our sequencing-from-sampling approach that it yields an outstanding utility with a small collection of samples and is reliable under various user purchase behaviors.

7 Conclusion

In this paper, we build on the framework of "optimization from samples" by extending the focus from optimizing set functions to sequence-dependent functions. Our objective is to determine an optimal ordering of items that maximizes an unknown utility function, given only a set of i.i.d. samples drawn from a specific distribution. We propose an approximation algorithm, with its performance guarantee depending on the curvature of the underlying functions. In future work, we aim to apply our findings to other real-world applications and further extend our results to encompass broader classes of sample distributions.

8 Acknowledgement

This work was supported in part by CAHSI-Google institutional Research Program.

9 Appendix

Lemma 3. *With a sufficiently large polynomial number of samples, the estimation $\widetilde{\Delta}(i,t)$ is n^2-close to $\Delta(i,t)$ for all $i \in \Omega$ and $t \in [k-1]$, with high probability, i.e., $\Delta(i,t) + \frac{\delta}{n^2} \geq \widetilde{\Delta}(i,t) \geq \Delta(i,t) - \frac{\delta}{n^2}$ where $\delta = \max_{\pi:|\pi|\leq k} \phi(\pi)$ denotes the maximum realized value of any sequence with a length of at most k.*

Proof: Our proof is inspired by the one presented in [6] (Appendix A) and extends their analysis from set functions to sequence functions. Consider an arbitrary pair of $i \in \Omega$ and $t \in [k-1]$.

Observation 1: The probability of sampling a sequence of length t is no less than $1/k$, whose value is at least $1/n$. Note that the case when $t = 0$ is trivial because the value of an empty sequence is known to be zero. Furthermore, given that the sampled sequence has a length of t, the probability of it not containing

item i is at least $1 - t/n \geq 1/n$. Hence, the probability of sampling a sequence of length t without i is at least $1/n^2$.

Observation 2: The probability of sampling a sequence of length $t+1$ is no less than $1/k$, where $1/k$ is at least $1/n$. Additionally, given that the sampled sequence has a length of $t+1$, the likelihood of the last item being i is at least $1/n$. Consequently, the probability of sampling a sequence of length $t+1$ with i at position $t+1$ is at least $1/n^2$.

The above two observations, together with Chernoff bounds, imply that gathering a minimum of n^5 samples of length t that do not contain i, and at least n^5 samples of length $t+1$ wherein i resides at position $t+1$, can be accomplished with high probability by obtaining n^8 samples.

By Hoeffding's inequality and the fact that δ is the largest possible value observed from any sequence of size at most k, we have

$$\Pr[|\texttt{avg}(\pi_{t,-i}) - \mathbb{E}_{\Pi_{t,-i}}[F(\Pi_{t,-i})]| \geq \frac{\delta}{2n^2}] \leq 2e^{-2n^5(\delta/2n^2)^2/\delta^2} \leq 2e^{-n/2},$$

and

$$\Pr[|\texttt{avg}(\pi_{t+1,i}) - \mathbb{E}_{\Pi_{t+1,i}}[F(\Pi_{t+1,i})]| \geq \frac{\delta}{2n^2}] \leq 2e^{-n/2}.$$

Given that $\Delta(i,t) = \mathbb{E}_{\Pi_{t+1,i}}[F(\Pi_{t+1,i})] - \mathbb{E}_{\Pi_{t,-i}}[F(\Pi_{t,-i})]$ and $\widetilde{\Delta}(i,t) = \texttt{avg}(\pi_{t+1,i}) - \texttt{avg}(\pi_{t,-i})$, we can deduce that, with a sample size of n^8, the following inequalities hold for all $i \in \Omega$ and $t \in [k-1]$, with high probability: $\Delta(i,t) + \frac{\delta}{n^2} \geq \widetilde{\Delta}(i,t) \geq \Delta(i,t) - \frac{\delta}{n^2}$. □

Lemma 4. *Let $f : 2^\Omega \to \mathbb{R}_{\geq 0}$ be a monotone and submodular function, given any subset of items $S \subseteq \Omega$ such that $|S| \leq k$, let R be a set of size t that is randomly sampled from $\Omega \setminus S$, for any $t \leq \min\{k, |\Omega \setminus S|\}$, $\mathbb{E}_R[f(R \mid S)] \geq (1-c)\mathbb{E}_R[f(R)]$.*

Proof: Assuming R is obtained by sequentially sampling t items without replacement, let $R = \{r_1, \cdots, r_t\}$, where r_j represents the j-th sampled item. Let $R_{[j]} = \{r_1, \cdots, r_j\}$ denote the first j sampled items,

$$\mathbb{E}_R[f(R \mid S)] = \sum_{j \in [t-1]} \mathbb{E}_R[f(r_{j+1} \mid R_{[j]} \cup S)]. \tag{14}$$

Consider any given sample R, because $r_{j+1} \notin R_{[j]}$ and $r_{j+1} \notin S$ (by the assumption that $R \subseteq \Omega \setminus S$), then by the curvature of f, $f(r_{j+1} \mid R_{[j]} \cup S) \geq (1-c)f(r_{j+1})$. It follows that $\mathbb{E}_R[f(R \mid S)] = \mathbb{E}_R[\sum_{j\in[t-1]} f(r_{j+1} \mid R_{[j]} \cup S)] = \sum_{j\in[t-1]} \mathbb{E}_R[f(r_{j+1} \mid R_{[j]} \cup S)] \geq \sum_{j\in[t-1]}(1-c)\mathbb{E}_R[f(r_{j+1})] = (1-c)\mathbb{E}_R[\sum_{j\in[t-1]} f(r_{j+1})] \geq (1-c)\mathbb{E}_R[f(R)]$ where the first inequality is by the observation that $f(r_{j+1} \mid R_{[j]} \cup S) \geq (1-c)f(r_{j+1})$ for any R and the last inequality is by the assumption that f is a submodular function. □

Lemma 5. *Let Π' denote a random sequence of length k that is sampled over items from $\Omega \setminus \pi^*$ where $\pi^* = \{e_1^*, \cdots, e_k^*\}$ denotes the optimal solution, we have $\sum_{t\in[k-1]} \Delta(e_{t+1}^*, t) \geq \alpha \cdot \mathbb{E}_{\Pi'}[F(\Pi' \uplus \pi^*) - F(\Pi')]$ where $\alpha = \frac{n-k}{n} \cdot \frac{n-k-1}{n-1} \cdot \ldots \cdot \frac{n-2k+1}{n-k+1}$.*

Proof: Let Π denote a random sequence of length k that is sampled over items from Ω. Hence, the probability that the first t items $\Pi_{[t]}$ is sampled from $\Omega \setminus \pi^*$ is at least $\alpha = \frac{n-k}{n} \cdot \frac{n-k-1}{n-1} \cdot \ldots \cdot \frac{n-2k+1}{n-k+1}$ for any $t \in \{1, \cdots, k\}$. Recall that Π' denotes a random sequence of length k that is sampled over items from $\Omega \setminus \pi^*$. It follows that $\mathbb{E}_{\Pi}[f_t(i \mid \Pi_{[t]})] \geq \alpha \mathbb{E}_{\Pi'}[f_t(i \mid \Pi'_{[t]})]$ for all $t \in \{1, \cdots, k\}$ and any item $i \in \Omega$.

Observe that $\sum_{t \in [k-1]} \Delta(e^*_{t+1}, t)$

$$
\begin{aligned}
&= \sum_{t \in [k-1]} \mathbb{E}_{\Pi_{t+1, e^*_{t+1}}}[F(\Pi_{t+1, e^*_{t+1}})] - \mathbb{E}_{\Pi_{t, -e^*_{t+1}}}[F(\Pi_{t, -e^*_{t+1}})] \\
&= \sum_{t \in [k-1]} \mathbb{E}_{\Pi_{t, -e^*_{t+1}}}[\sum_{z \in \{t+1, \cdots, k\}} f_z(e^*_{t+1} \mid \Pi_{t, -e^*_{t+1}})] \\
&\geq \sum_{t \in [k-1]} \mathbb{E}_{\Pi}[\sum_{z \in \{t+1, \cdots, k\}} f_z(e^*_{t+1} \mid \Pi_{[t]})] \\
&\geq \sum_{t \in [k-1]} \mathbb{E}_{\Pi}[\sum_{z \in \{t+1, \cdots, k\}} f_z(e^*_{t+1} \mid \Pi_{[z]})] \\
&= \sum_{t \in [k-1]} \sum_{z \in \{t+1, \cdots, k\}} \mathbb{E}_{\Pi}[f_z(e^*_{t+1} \mid \Pi_{[z]})] \\
&\geq \sum_{t \in [k-1]} \sum_{z \in \{t+1, \cdots, k\}} \alpha \mathbb{E}_{\Pi'}[f_z(e^*_{t+1} \mid \Pi'_{[z]})] \\
&= \alpha \mathbb{E}_{\Pi'}[\sum_{t \in [k-1]} \sum_{z \in \{t+1, \cdots, k\}} f_z(e^*_{t+1} \mid \Pi'_{[z]})] \\
&\geq \alpha \mathbb{E}_{\Pi'}[F(\Pi' \uplus \pi^*) - F(\Pi')]
\end{aligned}
$$

where the forth inequality is by the previous observation that $\mathbb{E}_{\Pi}[f_t(i \mid \Pi_{[t]})] \geq \alpha \mathbb{E}_{\Pi'}[f_t(i \mid \Pi'_{[t]})]$ for all $t \in \{1, \cdots, k\}$. □

References

1. Anderson, S.P., De Palma, A., Thisse, J.F.: Discrete choice theory of product differentiation. MIT press (1992)
2. Asadpour, A., Niazadeh, R., Saberi, A., Shameli, A.: Sequential submodular maximization and applications to ranking an assortment of products. Operations Research (2022)
3. Azar, Y., Gamzu, I.: Ranking with submodular valuations. In: Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms. pp. 1070–1079. SIAM (2011)
4. Balcan, M.F., Constantin, F., Iwata, S., Wang, L.: Learning valuation functions. In: Conference on Learning Theory. pp. 4–1. JMLR Workshop and Conference Proceedings (2012)
5. Balcan, M.F., Harvey, N.J.: Learning submodular functions. In: Proceedings of the forty-third annual ACM symposium on Theory of computing. pp. 793–802 (2011)
6. Balkanski, E., Rubinstein, A., Singer, Y.: The power of optimization from samples. Advances in Neural Information Processing Systems **29** (2016)

7. Balkanski, E., Rubinstein, A., Singer, Y.: The limitations of optimization from samples. In: Proceedings of the 49th annual acm sigact symposium on theory of computing. pp. 1016–1027 (2017)
8. Chen, W., Sun, X., Zhang, J., Zhang, Z.: Optimization from structured samples for coverage functions. In: International Conference on Machine Learning. pp. 1715–1724. PMLR (2020)
9. Chen, W., Sun, X., Zhang, J., Zhang, Z.: Network inference and influence maximization from samples. In: International Conference on Machine Learning. pp. 1707–1716. PMLR (2021)
10. Das, A., Kempe, D.: Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In: Proceedings of the 28th International Conference on International Conference on Machine Learning. pp. 1057–1064 (2011)
11. Feldman, V., Kothari, P.: Learning coverage functions and private release of marginals. In: Conference on Learning Theory. pp. 679–702. PMLR (2014)
12. Feldman, V., Kothari, P., Vondrák, J.: Representation, approximation and learning of submodular functions using low-rank decision trees. In: Conference on Learning Theory. pp. 711–740. PMLR (2013)
13. Golovin, D., Krause, A.: Adaptive submodularity: Theory and applications in active learning and stochastic optimization. Journal of Artificial Intelligence Research **42**, 427–486 (2011)
14. Lin, H., Bilmes, J.: A class of submodular functions for document summarization. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 510–520 (2011)
15. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions-i. Mathematical programming **14**(1), 265–294 (1978)
16. Schrijver, A., et al.: Combinatorial optimization: polyhedra and efficiency, vol. 24. Springer (2003)
17. Tang, S., Yuan, J.: Influence maximization with partial feedback. Operations Research Letters **48**(1), 24–28 (2020)
18. Tang, S., Yuan, J.: Cascade submodular maximization: Question selection and sequencing in online personality quiz. Production and Operations Management **30**(7), 2143–2161 (2021)
19. Tang, S., Yuan, J.: Optimal sampling gaps for adaptive submodular maximization. In: AAAI (2022)
20. Tang, S., Yuan, J.: Non-monotone sequential submodular maximization. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 15284–15291 (2024)
21. Tschiatschek, S., Singla, A., Krause, A.: Selecting sequences of items via submodular maximization. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
22. Zhang, G., Tatti, N., Gionis, A.: Ranking with submodular functions on a budget. Data mining and knowledge discovery **36**(3), 1197–1218 (2022)