Linear Submodular Maximization with Bandit Feedback

Wenjing Chen
Texas A&M University

Victoria G. Crawford
Texas A&M University

Abstract

Leveraging the intrinsic structure of submodular functions to design more sample-efficient algorithms for submodular maximization (SM) has gained significant attention in recent studies. In a number of real-world applications such as diversified recommender systems and data summarization, the submodular function exhibits additional linear structure. In this paper, we consider the problem of linear submodular maximization under the bandit feedback in the pure-exploration setting, where the submodular objective function is defined as $f : 2^U \to \mathbb{R}_{\geq 0}$, where $f = \sum_{i=1}^{d} w_i F_i$. It is assumed that we have value oracle access to the functions F_i, but the coefficients w_i are unknown, and f can only be accessed via noisy queries. To harness the linear structure, we develop algorithms inspired by adaptive allocation algorithms in the best-arm identification for the linear bandit, with approximation guarantees arbitrarily close to the setting where we have value oracle access to f. Our approach efficiently leverages information from prior samples, offering a significant improvement in sample efficiency. Experimental results on both synthetic datasets and real-world datasets demonstrate the superior performance of our method compared to baseline algorithms, particularly in terms of sample efficiency.

1 INTRODUCTION

A set function $f : 2^U \to \mathbb{R}$ is submodular if for every $X \subseteq Y \subseteq U$ and for every $x \in U \setminus Y$, we have $f(X \cup \{x\}) - f(X) \geq f(Y \cup \{x\}) - f(Y)$. In addition, f is often monotone, i.e., $f(Y) \geq f(X)$ for every $X \subset Y$. The standard assumption in proposed algorithms for submodular maximization is black-box value oracle access to the objective function f. In particular, algorithms can query any subset $X \subseteq U$ and the value of $f(X)$ is returned. However, in many important applications of submodular optimization, a more realistic model is that we can only query f approximately or subject to some noise (Singla et al., 2016; Horel and Singer, 2016; Hassidim and Singer, 2017a; Peng, 2021; Nie et al., 2022; Foster and Rakhlin, 2021). In this paper, we address the problem where the noisy query to the submodular objective function is i.i.d sub-Gaussian. This problem is also referred to as submodular bandit.

Submodular bandit problems naturally arise in a wide range of applications, including data summarization and diversified recommendation system tasks in the form of feature-based submodular utility models (Yu et al., 2016; Singla et al., 2016). In these applications, d monotone and submodular functions $F_1, ..., F_d : 2^U \to \mathbb{R}_{\geq 0}$ may capture the diminishing returns of adding items to a summary, or presenting recommended items to a group of users, over d different categories. But the subjective trade-off between these functions in terms of overall value of a set is unknown. Instead, access to the function is only indirect and noisy via methods such as crowdsourced user feedback (Singla et al., 2016) or in the form of clicks (Yu et al., 2016; Hiranandani et al., 2020). In particular, the monotone and submodular[1] objective representing the overall utility is of the form $f(S) = \sum_{i=1}^{d} w_i F_i(S)$, and $w_i \geq 0$ are the subjective and unknown preferences over the features. For example, U may be a set of articles, each F_i corresponds to an interest such as "sports" or "cooking" and measures the degree to which a set of articles satisfies that interest, and the w_i are the unknown preferences of a group of users over the different interests. While we cannot compute f, we can receive noisy feedback in the form of clicks on articles. Formally, the optimization problem we study in this paper is defined below.

Definition 1 (Linear Submodular Maximization with a Cardinality Constraint (LSM))**.** *Let U be a universe of elements of size n. Suppose we have d monotone*

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

[1] f is monotone and submodular since it is the linear sum of monotone submodular functions.

and submodular functions denoted as $F_i : 2^U \to \mathbb{R}$ for $i = 1, 2, ..., d$. The objective function $f : 2^U \to \mathbb{R}$ is a weighted sum of the d submodular functions, i.e. $f(S) = \sum_{i=1}^{d} w_i F_i(S) = \boldsymbol{F}^T(S)\boldsymbol{w}$, where $\boldsymbol{F}(S) = (F_1(S), F_2(S), ..., F_d(S))^T$, $\boldsymbol{w} = (w_1, w_2, ..., w_d)^T$, and $w_i \geq 0$ for any $i \in [d]$. LSM is then defined to be, given budget κ, find $\arg\max_{|S|\leq\kappa} f(S)$.

While the problem of LSM has been widely studied in many existing works, the main focus in these works is the regret-minimization setting, where the objective is to minimize total regret, and the solution set selected at each time step should achieve a high submodular function value. In contrast, we study the problem of LSM with bandit feedback in pure exploration setting (Soare et al., 2014; Xu et al., 2018), which is a parallel setting as the regret-minimization setting. In this setting, the goal is to take as few noisy samples as possible to find a good solution that achieves a certain quality. More specifically, our goal is to provide a solution set with a PAC-style theoretical guarantee (Even-Dar et al., 2002; Kalyanakrishnan et al., 2012), where the bandit algorithm identifies an ϵ-optimal arm with probability at least $1-\delta$, in as few samples as possible. Compared with the regret minimization setting, this setting is suited to applications where immediate user experience is secondary, and the focus is on gaining a thorough understanding of user preferences. For instance, user feedback can be gathered through mechanisms like surveys conducted on platforms such as Amazon's Mechanical Turk, and subsequently a solution can be deployed to the general audience that maximizes the user experience.

Submodular maximization under the bandit feedback in the pure exploration setting has been previously studied by (Singla et al., 2016; Chen et al., 2023; Jawanpuria et al.) in the general case where f does not have a linear structure. Under the PAC-learning setting, the primary goal is to propose algorithms that output a subset S that satisfies $f(S) \geq (1-1/e)f(OPT)-\epsilon$ in as few samples as possible. Consequently, a key metric for measuring the performance of our paper is the sample efficiency. As we would demonstrate in our algorithm analysis, the linear structure of f allows us to exploit the information collected in the previous noisy samples in a more consistent and efficient way, which results in algorithms that make significantly fewer noisy samples (as will be demonstrated in Section 5). In particular, the contributions of our paper are as follows:

(i) We propose and analyze the algorithm `Linear Greedy` (`LG`), which leverages the structure of the standard greedy algorithm for cardinality-constrained submodular maximization (SM) with a value oracle, but to deal with the noisy setting, `LG` proceeds as κ rounds of best-arm-identification for linear bandit. We prove that the exact function value of the output solution set of `LG` is arbitrarily close to the optimal approximation guarantee of $1-1/e$ (Nemhauser et al., 1978). In addition, `LG` chooses an optimal sample allocation strategy, and we prove a bound on the total number of noisy samples required by `LG`.

(ii) We next propose and analyze the algorithm `Linear Threshold Greedy` (`LinTG`) which is based on the more scalable threshold greedy algorithm of Badanidiyuru and Vondrák (2014) for SM with a value oracle, but we propose a new linear bandit strategy in a novel setting that is distinct from best-arm-identification. `LinTG` is more efficient compared to `LG` from an algorithmic perspective, but additionally the linear bandit strategy itself is more efficient in several aspects. Similar to `LG`, `LinTG` is proven to return a solution set S that is arbitrarily close to $1-1/e$.

(iii) Finally, we compare our algorithms to several alternative approaches on instances of movie recommendation. We find that our algorithms make significant improvements in terms of sample complexity.

1.1 Related Work

The problem of submodular maximization has found applications in a wide range of prominent machine learning problems, such as image summarization Singla et al. (2016); Chen et al. (2024), reinforcement learning Prajapat et al. (2023); De Santi et al. (2024), and recommender systems Hiranandani et al. (2020); Yu et al. (2016). While the standard assumption is value oracle access to the monotone submodular objective function, maximization of a general monotone submodular objective with approximate or noisy access to f has been considered in a variety of different settings (Horel and Singer, 2016; Hassidim and Singer, 2017b; Crawford et al., 2019). Among these works, the most related setting assumes the noisy feedback is i.i.d R-sub-Gaussian, which is also referred to as bandit feedback (Singla et al., 2016; Karimi et al., 2017; Chen et al., 2023). The problem is also referred to as submodular bandit, which is a specific type of combinatorial multi-armed bandit problem with the reward function assumed to be submodular (Singla et al., 2016; Takemori et al., 2020; Hiranandani et al., 2020; Yue and Guestrin, 2011; Nie et al., 2022, 2023).

However, the majority of existing works in submodular bandits focus on regret minimization rather than the pure-exploration setting considered in this paper Nie et al. (2022); Gabillon et al. (2013); Yue and Guestrin (2011); Takemori et al. (2020). In regret minimization, the goal is to minimize cumulative regret over time, ensuring that the solution set selected at each step achieves a high submodular function value. In contrast, our objective is to identify a high-quality solution set using as few samples as possible. In particular, our problem setting focuses on best-arm identification under the PAC learning framework Even-Dar et al. (2002); Singla et al. (2016); Chen et al. (2023), where the goal is to find a subset S such that $f(S) \geq (1-1/e)f(OPT)-\epsilon$ in as few queries as possible. As the approximation guarantee is fixed, sample efficiency is the most relevant metric in our setting. While these two settings are not directly comparable due to their differing objectives, we include experimental comparisons with the regret minimization algorithm proposed by Yue and Guestrin (2011), in Section 5 to provide additional insights.

A closely related problem is the linear submodular bandit, where the objective function f can be expressed by the linear sum of known submodular functions with unknown weights (Yue and Guestrin, 2011; Takemori et al., 2020; Yu et al., 2016; Hiranandani et al., 2020). The problem was first introduced by Yue and Guestrin (2011), who studied the problem with the goal of regret minimization. While their work shares the same objective function as ours, their focus on regret minimization differs from our pure-exploration setting. Under this setting, the strategy is to explore new choices of the subsets while also exploiting previous samples to select good ones. Among the works that also study submodular bandits, Singla et al. (2016); Chen et al. (2023); Jawanpuria et al. address the same pure-exploration setting as us. However, although the goal of our paper is the same as these previous works, none of these works explore the linear structure of the submodular objective function. Consequently, they cannot leverage samples collected at past greedy rounds, resulting in algorithms that are less efficient in utilizing past information.

Another related line of work is the problem of best-arm identification in linear bandit, which was first studied in Soare et al. (2014). The authors proposed static allocation strategies and adaptive strategies based on successive elimination. Later on, there is a large line of works that proposed more adaptive algorithms where the sampling sequence is not fixed in advance (Xu et al., 2018; Degenne et al., 2020). Xu et al. (2018) proposed an algorithm named LinGap which is claimed to be fully adaptive and performed better than those in Soare et al. (2014). Degenne et al. (2020) proposed the first algorithm that is asymptotically optimal.

2 PRELIMINARIES

In this section, we present the basic setup for our problem, including motivating examples, description of linear bandit setting and related technical background that will be used for our algorithms. First, we provide some additional information about our problem formulation. It is assumed that we have value oracle access to each of the basis functions F_i, and we can query them on any subset $S \subseteq U$, but the weight vector $\mathbf{w} = (w_1, w_2, ..., w_d)^T$ is unknown. We assume that the l_2 norm of $\mathbf{w}$ is bounded by Q, and the l_2 norm of $\mathbf{F}(x)$ is bounded by L for any $x \in U$. Further discussion and justification for these assumptions is described below in Section 2.1.

Let us denote the marginal gain of adding element $x \in U$ to a set $S \subseteq U$ as $\Delta f(S, x)$, i.e., $\Delta f(S, x) := f(S \cup \{x\}) - f(S)$. Therefore, by definition, we have that $\Delta f(S, x) = \sum_{i=1}^{d} w_i \Delta F_i(S, x) = \Delta \mathbf{F}(S, x)^T \mathbf{w}$. Here $\Delta \mathbf{F}(S, x) = (\Delta F_1(S, x), ..., \Delta F_d(S, x))^T$. For any $x \in U$ and $S \subseteq U$, we assume the sampling result of the marginal gain of adding x to the set S is i.i.d R-sub-Gaussian with expectation $\Delta f(S, s)$. In particular, samples from this distribution are of the form $\Delta \mathbf{F}(S, x)^T \mathbf{w} + \xi_t$, where ξ_t are independent and identically distributed samples with zero-mean noise and are R-sub-Gaussian.

We list the necessary notation: Throughout this paper, we define the matrix norm as $||\mathbf{x}||_{\mathbf{A}} = \sqrt{\mathbf{x}^T \mathbf{A} \mathbf{x}}$. We use $[n]$ to denote the set $\{1, 2, ..., n\}$. In particular, we use U and $[n]$ interchangeably.

2.1 Motivating Application

We now illustrate our problem setup with an example of the diversified article recommendation system. Suppose we have a universe U of n articles, and our goal is to identify a subset of at most k of these articles to display on the website reflecting interests that users come to the website searching for an article about. For instance, a user might seek sports-related articles. The goal is to display the selected articles so that users coming to the website for a specific interest are able to quickly find what they are looking for.

We model user behavior probabilistically. Each user i arrives with an unknown probability u_i. Given arrival, user i selects a topic j from a set of d topics with probability $v_{i,j}$, representing their hidden preferences. A click occurs if the displayed articles X contain at least one article covering the selected topic j.

Let $F_j(X)$ be the probability that at least one article in X covers topic j, modeled using a probabilistic coverage model (Yue and Guestrin, 2011; Hiranandani et al., 2020), which is defined in Appendix C. From the definition of the probabilistic coverage model, we can see that F_j is both monotone and submodular for any $j \in [d]$. Notice that that the F_j does not depend on any unknown user preferences and therefore the algorithm can be assumed to have value oracle access to them.

Then the expected value of a click is

$$
\begin{aligned}
f(X) &= \sum_{i=1}^{N} u_i \left(\sum_{j=1}^{d} v_{i,j} F_j(X) \right) \\
&= \sum_{j=1}^{d} \left(\sum_{i=1}^{N} u_i v_{i,j} \right) F_j(X).
\end{aligned}
$$

In this application, $w_j = \sum_{i=1}^{N} u_i v_{i,j}$ is an unknown weight representing the overall popularity of topic j. By definition, we have that $\sum_{j=1}^{d} w_j = 1$. Therefore, the l_2 norm of $\mathbf{w}$ is upper bounded by 1 and thus $Q = 1$. Since the noisy feedback is binary, the sub-Gaussian parameter satisfies $R = 1$ in this case. For any $j \in [d]$, by the probabilistic coverage model, F_j is bounded by 1, L is thus $\sqrt{d}$.

2.2 The Linear Bandit Setting

We describe the linear bandit problem and how it can be related to our noisy submodular optimization setting. In our setting, the arms correspond to queries to different marginal gains of f. In particular, suppose we have some set of arms $\mathcal{X}$ where each $\mathbf{x} \in \mathcal{X}$ is $\langle \Delta F_1(S, x), ..., \Delta F_d(S, x) \rangle$ for some set $S \subseteq U$ and element $x \in U$. We denote the set of features as $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the feature of element $i \in [n]$.

At each step t, the algorithm selects an element $a_t \in [n]$, and observe the reward $r_t = \mathbf{x}_{a_t}^T \mathbf{w} + \xi_t$, where $\mathbf{x}_{a_t}$ is the feature of element a_t. Here $\mathbf{w}$ is unknown and we assume $||\mathbf{w}||_2 \leq Q$. We also define the arm selection sequence of the first t steps as $\mathbf{X}_t = (\mathbf{x}_{a_1}, \mathbf{x}_{a_2}, ..., \mathbf{x}_{a_t})^T$, and the vector of rewards of the first t steps as $R_t = (r_1, r_2, ..., r_t)^T$. A central component of best-arm identification in the linear bandit setting is having an estimate of the unknown vector $\mathbf{w}$. The regularized least-square estimate of the vector $\mathbf{w}$ with some regularizer λ is thus

$$
\begin{aligned}
\hat{\mathbf{w}}_t^\lambda &= (\sum_{i=1}^{t} \mathbf{x}_{a_i} \mathbf{x}_{a_i}^T + \lambda I)^{-1} (\sum_{i=1}^{t} \mathbf{x}_{a_i} r_i) \\
&= (\mathbf{X}_t^T \mathbf{X}_t + \lambda I)^{-1} (\mathbf{X}_t^T R_t).
\end{aligned}
$$

The matrix $\mathbf{X}_t^T \mathbf{X}_t + \lambda I$ is defined as $\mathbf{A}_t$ and the vector $\mathbf{X}_t^T R_t$ is defined as $\mathbf{b}_t$. Consequently, $\hat{\mathbf{w}}_t = \mathbf{A}_t^{-1} \mathbf{b}_t$. Notice that here the inverse of $\mathbf{A}_t$ can be updated quickly by using the Woodbury formula in Lemma 11. Here we explain how the linear bandit model can be used in our algorithms. Suppose we choose to sample the marginal gain of adding element s_t to subset S_t at time step t, then we can set $\mathbf{x}_{a_t} = \Delta \mathbf{F}(S_t, s_t)$, the reward r_t is thus the noisy marginal gain feedback, i.e., $r_t = \Delta f_t(S_t, s_t) = \Delta \mathbf{F}(S_t, s_t)^T \mathbf{w} + \xi_t$.

2.3 Concentration Properties of Estimation of Weight Vector

Our estimate of $\mathbf{w}$ is used alongside a concentration inequality that tells us how accurate our estimate is in the form of a confidence interval. In contrast to concentration inequalities such as Hoeffding's which is for a real-valued random variable, the confidence region in the concentration inequality we use can be geometrically interpreted as a ellipsoid. Pulling different arms results in the shrinking of the confidence region along different directions. The following is the concentration inequality that we will use for our least-squares estimator. The result is obtained from Thm. 2 in Abbasi-Yadkori et al. (2011). The proof can be found in the appendix.

Proposition 1. *Let $\hat{\boldsymbol{w}}_t^\lambda$ be the solution to the regularized least-squares problem with regularizer λ and let $\boldsymbol{A}_t = \boldsymbol{X}_t^T \boldsymbol{X}_t + \lambda \boldsymbol{I}$. Then for any $N \geq 0$ and every adaptive sequence $\boldsymbol{X}_t$ such that at any step t, $\boldsymbol{x}_{a_t}$ only depends on past history, we have that with probability at least $1 - \delta$, it holds that $|\boldsymbol{y}^T (\hat{\boldsymbol{w}}_t^\lambda - \boldsymbol{w})| \leq ||\boldsymbol{y}||_{(\boldsymbol{A}_t)^{-1}} C_t$ for all $t \geq N$ and all $\boldsymbol{y} \in \mathbb{R}^d$ that only depends on past history up to time N, where C_t is defined as*

$$
C_t = R \sqrt{2 \log \frac{\det(\boldsymbol{A}_t)^{\frac{1}{2}} \det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}} Q.
$$

3 THE STANDARD GREEDY ALGORITHM

In this section, we present and analyze our algorithm `Linear Greedy` (`LG`). The standard greedy algorithm for SM proceeds in a series of rounds, and during each round the element with the highest marginal gain $u = \arg\max_{x \in U} \Delta f(S, x)$ is identified and added to the solution set S. `LG` is based on the standard greedy algorithm, except at each round the highest marginal gain element must be identified by taking noisy samples instead of querying f directly. Our main challenge is upon determining the sampling strategy where we can construct our solution in as few noisy queries as possible.

Each round of the greedy algorithm can be viewed as a best-arm-identification problem in linear bandit where the marginal gains $\Delta f(S, u)$ for each $u \in U$ correspond to the set of arms. In particular, we are in the PAC-learning setting where the goal is to find an arm $\hat{a}$ that satisfies $\mathbf{x}_{\hat{a}}^T \mathbf{w} \geq \max_{u \in [n]} \mathbf{x}_u^T \mathbf{w} - \epsilon'$ where $\epsilon' = \epsilon / \kappa$. Therefore, existing algorithms for best-arm-identification in linear bandit in the PAC-learning setting can be used for each round of the greedy algorithm. In particular, we develop an algorithm using the static allocation strategy proposed in (Soare et al., 2014) as the subroutine as presented in Section A.1. However, the static allocation strategy imposes a key limitation: it fixes the sampling sequence in advance, making it impossible to reuse samples collected during earlier greedy rounds.

In fact, the method of simply incorporating any existing algorithms for best-arm-identification doesn't take advantage of the fact that the weight vector estimated in different rounds of the algorithm is the same. In contrast, in LG, we leverage the noisy samples obtained from past greedy rounds to get a better estimate of $\mathbf{w}$. In particular, we view each round in LG as a linear bandit problem under the offline-to-online setting (Wagenmaker and Pacchiano, 2023) where we view the samples gained from past rounds of LG as the offline dataset. In addition, unlike in the traditional setting where the set of arms we want to estimate to obtain the best arm (evaluation set) and the set of arms where we can sample from (sampling set) are the same, in our setting during the ℓ-th round the evaluation set is the set of current marginal gains $\{\Delta\mathbf{F}(S_l, a)\}_{a\in[n]}$ while the sampling set includes all previous marginal gains in addition $\{\Delta\mathbf{F}(S_{l'}, a)\}_{l'\in[l],a\in[n]}$. This problem is also referred to as transductive bandit (Fiez et al., 2019). In order to reuse the samples from the past greedy round, we cannot use any static allocation strategy where the sampling sequence is fixed in advance. Our sampling strategy is inspired by the algorithm of Xu et al. (2018), but extends beyond merely integrating best-arm identification into the greedy algorithm. The novelty of our approach lies in its integration of best-arm identification within the greedy framework, specifically addressing the offline data and the discrepancy between the evaluation and sampling sets.

Algorithm Overview. Pseudocode for LG is given in Algorithm 1. LG takes as input the error probability parameter δ, the approximation error parameter ϵ, and the budget κ. It operates in κ rounds, adding one element of approximately maximum marginal gain (with high probability) to the solution set S in each round. We denote the solution set before adding the l-th element as S_l. LG keeps track of the following throughout its duration: (i) A variable t indicating the total number of noisy samples that have been taken since the beginning of LG, (ii) the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ for the least-squares estimate of $\mathbf{w}$ as described in Section 2.2, and (iii) the total number of noisy samples of the marginal gain $\Delta f(S_{l'}, j)$ for each $l' \in \{1, 2, 3..., l\}$ and $j \in [n]$ denoted as $T_{l',u}$ where l is the index of the current round. Here $\beta_t(i,j) = C_t||\Delta\mathbf{F}(S,j) - \Delta\mathbf{F}(S,i)||_{\mathbf{A}_t^{-1}}$ where C_t is defined to be $C_t = R\sqrt{2\log\frac{\kappa\det(\mathbf{A}_t)^{\frac{1}{2}}\det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}}Q$.

We now describe a round of LG. The round begins by taking a single noisy sample of the marginal gain of adding each element to the current solution S (Lines 7 to 10). Subsequently, the algorithm enters a loop where each iteration corresponds to taking a single noisy sample of a specific marginal gain. During each iteration of the loop, we update the estimate of $\mathbf{w}$ based on the least-square estimate procedure described in Section 2.2.

Next, LG determines the best marginal gain vector to sample from the set of current and past set of marginal gains, i.e. the set $\{\Delta\mathbf{F}(S_{l'}, a)\}_{l'\in[l],a\in[n]}$ (Lines 15 to 18). This is done by first selecting two elements i_t and j_t from the universe U (Line 15), and then select the marginal gain vector that makes the confidence interval $||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||_{\mathbf{A}_t^{-1}}$ to shrink as fast as possible. In particular, we choose the asymptotic-optimal ratio for decreasing the confidence interval $||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||_{\mathbf{A}_t^{-1}}$ when the number of samples goes to infinity. We denote the asymptotic-optimal allocation ratio of the marginal gain vector $\Delta\mathbf{F}(S_{l'}, a)$ at the round l for estimating the direction of $\Delta\mathbf{F}(S,i) - \Delta\mathbf{F}(S,j)$ as $p^*(l', a|l, i, j)$, which can be calculated through equation (4) and through solving the linear programming problem in (5) (see the discussion in Appendix A.2 for more details). We pull the arm so that the number of samples of arms is close to the ratio p. More specifically,

$$(l_t, a_t) = \arg\min_{l'\in[l],a\in[n]:p^*_{l',a}(i_t,j_t)>0} \frac{T_{l',a}}{p^*(l', a|l, i_t, j_t)}.$$

Then LG samples the direction of $\Delta\mathbf{F}(S_{l_t}, a_t)$ where S_{l_t} is the solution set S before the l_t-th greedy round.

Theoretical Guarantees. Finally, we present the theoretical guarantees of LG. Here we denote the total number of past noisy samples of $\Delta\mathbf{F}(S_{l'}, a)$ before round l as $N^{(l)}_{l',a}$ where $S_{l'}$ is the solution set before the l'-th greedy round ($l' \le l$), and the total number of past noisy samples before round l as $N^{(l)} = \sum_{l'\in[l],a\in[n]} N^{(l)}_{l',a}$. Moreover, we define $\Delta_{l,i} = \max_{a\in[n]} \Delta\mathbf{F}(S_l, a)^T\mathbf{w} - \Delta\mathbf{F}(S_l, i)^T\mathbf{w}$. Then we have the following results. The proof is deferred to Section A.2 in the appendix.

Theorem 2. *With probability at least $1 - \delta$, the following statements hold:*

1. *The exact function value of the output solution set S satisfies that $f(S) \ge (1 - e^{-1})f(OPT) - \epsilon$. Here OPT is an optimal solution to the LSM problem;*

Algorithm 1 `Linear Greedy` (`LG`)

Input: δ, ϵ
Output: solution set $S \subseteq U$
$S \leftarrow \emptyset$
$\mathbf{A} \leftarrow \lambda\mathbf{I}$, $\mathbf{b} \leftarrow \mathbf{0}$, $t \leftarrow 1$
for $l = 1, ..., \kappa$ **do**
  $S_l \leftarrow S$
  for $a \in [n]$ **do**
    sample $r_t = \Delta\mathbf{F}^T(S, a)\mathbf{w} + \xi_t$ and obtain reward r_t
    update $\mathbf{A} \leftarrow \mathbf{A} + \Delta\mathbf{F}(S, a)\Delta\mathbf{F}^T(S, a)$ and $\mathbf{b} \leftarrow \mathbf{b} + r_t\Delta\mathbf{F}(S, a)$
    $t \leftarrow t + 1$, $T_{l,a} \leftarrow 1$
  for i, j **in** $[n]$ **do**
    $p^*(\cdot, \cdot|l, i, j) \leftarrow$ calculated from solving equation (4) and LP problem in (5)
  while true do
    $\hat{\mathbf{w}}_t = \mathbf{A}^{-1}\mathbf{b}$
    $i_t \leftarrow \arg\max_{i \in [n]} \Delta\mathbf{F}(S, i)^T\hat{\mathbf{w}}_t$
    $j_t \leftarrow \arg\max_{j \in [n]} \Delta\mathbf{F}(S, j)^T\hat{\mathbf{w}}_t - \Delta\mathbf{F}(S, i_t)^T\hat{\mathbf{w}}_t + \beta_t(j, i_t)$
    $B(t) \leftarrow \Delta\mathbf{F}(S, j_t)^T\hat{\mathbf{w}}_t - \Delta\mathbf{F}(S, i_t)^T\hat{\mathbf{w}}_t + \beta_t(j_t, i_t)$
    $(l_t, a_t) = \arg\min_{l' \in [l], a \in [n] p^*(l', a|l, i_t, j_t) > 0} \frac{T_{l',a}}{p^*(l', a|l, i_t, j_t)}$
    if $B(t) \leq \epsilon/\kappa$ **then**
      $S \leftarrow S \cup \{i_t\}$; **break**
    sample $r_t = \Delta\mathbf{F}^T(S_{l_t}, a_t)\mathbf{w} + \xi_t$
    pdate $\mathbf{A} \leftarrow \mathbf{A} + \Delta\mathbf{F}(S_{l_t}, a_t)\Delta\mathbf{F}^T(S_{l_t}, a_t)$ and $\mathbf{b} \leftarrow \mathbf{b} + r_t\Delta\mathbf{F}(S_{l_t}, a_t)$
    $T_{l_t,a_t} \leftarrow T_{l_t,a_t} + 1$, $t \leftarrow t + 1$
return S

2. During each round in `LG`*, as the number of samples increases, the confidence region would shrink, and the stopping condition $B(t) \leq \epsilon/\kappa$ would be met ultimately. Assuming $\lambda \leq \frac{2R^2}{Q^2} \log \frac{1}{\delta}$, then* `LG` *at round l takes at most*

$$\sum_{l' \in [l], a \in [n]} \max\{4H_{l',a}^{(l)} R^2 (2 \log \frac{1}{\delta} + d \log M^{(l)}) + 1 - N_{l',a}^{(l)}, 0\}$$

number of samples, where $M^{(l)} = \frac{16H_\epsilon^{(l)} R^2 L^2}{\lambda} \log(\frac{1}{\delta}) + \frac{32(H_\epsilon^{(l)})^2 R^4 L^4}{\lambda^2} + \frac{2(N^{(l)} + nl)L^2}{\lambda d} + 2$. Here $H_{l',a}^{(l)} = \max_{i,j \in [n]} \frac{p^*(l', a|l, i, j)\rho_{i,j}^l}{\max\{\epsilon/\kappa, \frac{\Delta_{l,i} \vee \Delta_{l,j} + \epsilon/\kappa}{3}\}^2}$ and $H_\epsilon^{(l)} = \sum_{l' \in [l], a \in [n]} H_{l',a}^{(l)}$, where $\rho_{i,j}^l$ is the optimal value of (5).

Discussion We include a discussion below about the implications of Theorem 2, but additional details are provided in Appendix A.2. First of all, if we set $N_{l',a}^{(l)} = 0$, then the result in Theorem 2 is reduced to the case where we don't utilize the samples of the marginal gains from the past rounds. Compared with this case, our sample complexity of using past data can be improved by at most $N_{l',a}^{(l)}$ from the past rounds for each l', a while only sacrificing the factor of $O(\log(N^{(l)}))$.

Second, we compare our results with the result of the algorithm where the element-selecting procedure is replaced by LinGapE in Xu et al. (2018), which does not leverage samples of noisy marginal gains from previous rounds of the greedy algorithm. The sample complexity of LinGapE is $4H_\epsilon^{(l)} R^2 (2 \log \frac{\kappa n^2}{\delta} + d \log M^{(l)}) + n$, which is worse than the sample complexity of `LG` in the case where we don't utilize samples from the past by a factor of $O(\log n^2)$.

Third, if we set the marginal gain to be $\{\Delta\mathbf{F}(S_l, a)\}_{l \in [\kappa], a \in [n]} = (\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_{\kappa n})$, then our problem is reduced to the general bandit case. In this case, the allocation strategy is the same as that used in the CLUCB algorithm in Chen et al. (2014). The sample complexity becomes $8H_\epsilon^{(l)} R^2 \log \frac{1}{\delta} + n + O(H_\epsilon^{(l)} dR^2 \log \frac{H_\epsilon^{(l)} RL}{\lambda})$. The last term $O(H_\epsilon^{(l)} dR^2 \log \frac{H_\epsilon^{(l)} RL}{\lambda})$ can be seen as resulting from the linear structure while the first term matches the sample complexity result under the general bandit setting in Chen et al. (2014) up to a log-factor.

Lastly, while solving linear programming problems can be computationally intensive in practice, we can avoid this by replacing the asymptotic optimal allocation ratio $p^*(l', a|l, i, j)$ with any other allocation policy $p(l', a|l, i, j)$. Then the first item in Theorem 2 still holds and the guarantee on the sample complexity would be worse. In particular, we can set the allocation ratio so that `LG` follows the same strategy as the CLUCB algorithm in Chen et al. (2014). Further discussion on the comparison of our approach and CLUCB algorithm can be found in Section 5.

4 THRESHOLD GREEDY WITH ADAPTIVE ALLOCATION STRATEGY

In the previous section, we introduced the `LG` algorithm, which builds on the standard greedy approach by iteratively selecting elements with approximately the highest marginal gains. However, `LG` faces significant scalability challenges, particularly due to the need for best-arm identification in each round of the greedy process. In contrast, threshold-based greedy algorithms, which simply check whether the marginal gain exceeds a certain threshold, tend to be more efficient both in terms of sample complexity and computational requirements. Specifically, LG suffers from several limitations: (i) The allocation strategy in LG necessitates solving $O(\kappa n^2)$ linear programming problems, (ii) LG requires updating the inverse of matrix $\mathbf{A}$ and the confidence interval $\beta_t(i_t, j_t) = C_t||\Delta\mathbf{F}(S, j_t) - \Delta\mathbf{F}(S, i_t)||_{\mathbf{A}_t^{-1}}$ during each sampling of a noisy query, (iii) LG may need many samples to distinguish between marginal gains that are very similar to each other, and finally (iv) LG requires $n\kappa$ marginal gain contributions, which may be too costly in the case κ is large.

Motivated by these weaknesses of LG, we introduce an alternative algorithm Linear Threshold Greedy (LinTG) which builds upon the threshold greedy algorithm of Badanidiyuru and Vondrák (2014), but incorporates a novel allocation strategy. We now describe LinTG, pseudocode for which is given in Algorithm 2. LinTG proceeds in a series of rounds (Lines 10 to 26), each round corresponding to a threshold value w. During each round of LinTG, we iterate over the ground set U and make a decision about whether to add each element $a \in U$ to our solution S or not depending on whether the marginal gain $\Delta f(S, a)$ is above w. At the end of each round, the threshold w decreases by a factor of $1 - \alpha$. Here α determines the rate at which the threshold w decreases across different iterations. It is this part of the algorithm that follows the same framework as the threshold greedy algorithm of Badanidiyuru and Vondrák (2014).

Throughout LinTG, we keep track of the following parameters: the covariance matrix $\mathbf{A}$, vector $\mathbf{b}$, and the set of marginal gain vectors to sample from, which is denoted as $\mathcal{X}$. Here we denote the size of $\mathcal{X}$ as m. $\mathcal{X}$ includes all the past marginal gains that have been evaluated. Let us denote $\mathcal{X} = \{\Delta\mathbf{F}(S_i, a_i)\}_{i\in[m]}$ where S_i and a_i are the i-th evaluated solution set and element. In addition, we keep track of the total number of noisy queries denoted as t, and the total number of noisy samples of the marginal gain $\Delta f(S_i, a_i)$ denoted as T_{S_i,a_i}.

We now describe the adaptive allocation strategy we have developed for LinTG. In contrast to the standard greedy algorithm and LG, LinTG is not identifying a best arm and therefore the allocation strategy used in LG is not effective. LinTG decides whether to add each $a \in U$ to S or not on Lines 13 to 26. In order to identify whether the marginal gain $\Delta f(S, a)$ is above or below the threshold, LinTG iteratively identifies the best marginal gain vector to sample from the set $\mathcal{X}$ during the loop on Lines 18 to 26 to decrease $||\Delta\mathbf{F}(S, a)||_{\mathbf{A}^{-1}}$ as fast as possible (recall in LG we tried to shrink $||\Delta\mathbf{F}(S, j_t) - \Delta\mathbf{F}(S, i_t)||_{\mathbf{A}_t^{-1}}$ as fast as possible). Here β_t denotes the confidence interval which is defined as $\beta_t = \left(R\sqrt{2\log\frac{2\det(\mathbf{A}_t)^{\frac{1}{2}}\det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}}Q\right)||\Delta\mathbf{F}(S, a)||_{\mathbf{A}_t^{-1}}$. Let p_i^* denote the fraction of samples allocated to element $\Delta\mathbf{F}(S_i, a_i)$, with $\sum_{i\in[m]} p_i^* = 1$, which is asymptotically optimal for decreasing $||\Delta\mathbf{F}(S, a)||_{\mathbf{A}^{-1}}$. p_i^* can be calculated by solving a linear programming problem (see Appendix B for more details about how to calculate p_i^*). Then the strategy is to sample the marginal gain of $\Delta\mathbf{F}(S_{i_t}, a_{i_t})$ where $i_t = \arg\min_{i\in[|\mathcal{X}|]:p_i^*>0} \frac{T_{S_i,a_i}}{p_i^*}$. Each iterative sample is followed by an update of $\mathbf{A}, \mathbf{b}$, and then the estimate $\hat{\mathbf{w}}$ are all updated according to the procedure described in Section 2.2 on Lines 25.

LinTG is more scalable compared to LG in the following ways: (i) LinTG requires solving only $O\left(\frac{n}{\alpha}\log\frac{\kappa}{\alpha}\right)$ linear programming problems, which is substantially fewer than LG. (ii) Unlike LG, LinTG does not rely on pairwise comparisons of marginal gains between elements, making it less sensitive to small differences in marginal gains. (iii) LinTG makes a total of only $O(\frac{n}{\alpha}\log\frac{\kappa}{\alpha})$ marginal gain evaluations.

Additionally, LinTG allows for a more efficient batch-based implementation. Since we only consider one allocation ratio for each evaluated marginal gain, the greedy selection strategy in Line 19 can be changed to run in batches. By setting the batch size to B, the algorithm updates the confidence intervals and checks the stopping condition after every B queries. This reduces the number of times the inverse of $\mathbf{A}$ and the confidence intervals need to be updated by a multiplicative factor of $1/B$ while increasing the total number of queries by only an additive factor of B.

The theoretical guarantee of LinTG is presented in Theorem 3, and the proof can be found in Section B of the appendix. Let $N^{(m)}$ be the the total number of samples before the m-th evaluation of marginal gains, and $N_i^{(m)}$ be the number of samples to the marginal gain $\Delta f(S_i, a_i)$ before the m-th evaluation of marginal gains.

Theorem 3. *LinTG makes $n\log(\kappa/\alpha)/\alpha$ evaluations of the marginal gains. In addition, with probability at least $1 - \delta$, the following statements hold:*

1. *The function value of the output solution set S satisfies that $f(S) \geq (1 - e^{-1} - \alpha)f(OPT) - 2\epsilon$. Here OPT is an optimal solution to the LSM problem;*

2. *Assuming $\lambda \leq \frac{2R^2}{Q^2}\log\frac{2}{\delta}$, the m-th evaluation of the marginal gain of adding an element a to S in LinTG takes at most*

$$\sum_{i\in[m]} \max\{4H_i^{(m)}R^2(2\log\frac{2}{\delta} + d\log M^{(m)}) + 1 - N_i^{(m)}, 0\}$$

samples, where $M^{(m)} = \frac{16H_\epsilon^{(m)}R^2L^2}{\lambda}\log\frac{2}{\delta} +$

Algorithm 2 `Linear Threshold Greedy` (`LinTG`)

1: **Input:** ϵ, δ, α
2: $\mathcal{X} \leftarrow \emptyset$, $N_0 \leftarrow \frac{2R^2\kappa^2}{\epsilon^2}\log(6n/\delta)$,
3: $\mathbf{A} \leftarrow \lambda\mathbf{I}$, $\mathbf{b} \leftarrow \mathbf{0}$
4: **for all** $a \in [n]$ **do**
5: $\quad\hat{f}(a) \leftarrow$ sample average over N_0 samples with features $\mathbf{F}(a)$
6: $\quad\mathbf{A} \leftarrow \mathbf{A} + N_0\mathbf{F}(a)\mathbf{F}^T(a)$, $\mathbf{b} \leftarrow \mathbf{b} + N_0\hat{f}(a)\mathbf{F}(a)$
7: $\quad T_{\emptyset,a} = N_0$, $\mathcal{X} \leftarrow \mathcal{X} \cup \{\mathbf{F}(a)\}$
8: $g := \max_{a\in[n]} \hat{f}(a)$,
9: $S \leftarrow \emptyset$, $t \leftarrow nN_0$, $m = n$
10: **for** $w = g$; $w = (1-\alpha)w$; $w > \alpha g/\kappa$ **do**
11: $\quad$**for all** $a \in [n]$ **do**
12: $\quad\quad p^* \leftarrow$ calculated from solving equation (13) and LP problem in (14)
13: $\quad\quad$**if** $|S| < \kappa$ **then**
14: $\quad\quad\quad$sample $r_t = \Delta\mathbf{F}(S,a)^T\mathbf{w} + \xi_t$ and obtain the reward r_t
15: $\quad\quad\quad$update $\mathbf{A}$ and $\mathbf{b}$, $\hat{\mathbf{w}} = \mathbf{A}^{-1}\mathbf{b}$
16: $\quad\quad\quad\mathcal{X} \leftarrow \mathcal{X} \cup \{\Delta\mathbf{F}(S,a)\}$, $m = m+1$
17: $\quad\quad\quad S_m \leftarrow S$, $a_m \leftarrow a$, $T_{S_m,a_m} \leftarrow 1$, $t \leftarrow t+1$
18: $\quad\quad\quad$**while true do**
19: $\quad\quad\quad\quad i_t = \arg\min_{i\in[|\mathcal{X}|]:p_i^*>0} \frac{T_{S_i,a_i}}{p_i^*}$
20: $\quad\quad\quad\quad$**if** $\Delta\mathbf{F}(S,a)^T\hat{\mathbf{w}} - \beta_t \geq w - \epsilon/\kappa$ **then**
21: $\quad\quad\quad\quad\quad S \leftarrow S \cup \{a\}$; **break**
22: $\quad\quad\quad\quad$**else if** $\Delta\mathbf{F}(S,a)^T\hat{\mathbf{w}} + \beta_t \leq w + \epsilon/\kappa$ **then**
23: $\quad\quad\quad\quad\quad$**break**
24: $\quad\quad\quad\quad$sample $r_t = \Delta\mathbf{F}(S_{i_t}, a_{i_t})^T\mathbf{w} + \xi_t$
25: $\quad\quad\quad\quad$update $\mathbf{A}$ and $\mathbf{b}$, $\hat{\mathbf{w}} = \mathbf{A}^{-1}\mathbf{b}$
26: $\quad\quad\quad\quad T_{S_{i_t},a_{i_t}} \leftarrow T_{S_{i_t},a_{i_t}} + 1$, $t \leftarrow t+1$
27: **return** S

$\frac{32(H_\epsilon^{(m)})^2R^4L^4}{\lambda^2} + \frac{2(N^{(m)}+m)L^2}{\lambda d} + 2$. *Here* $H_i^{(m)} = \frac{p_i^*\rho^{(m)}}{\max\{\frac{\epsilon/\kappa+|w-\Delta F(S,a)^T w|}{2}, \epsilon/\kappa\}^2}$ *and* $H_\epsilon^{(m)} = \sum_{i\in[m]} H_i^{(m)}$. $\rho^{(m)}$ *is the optimal vale of (14).*

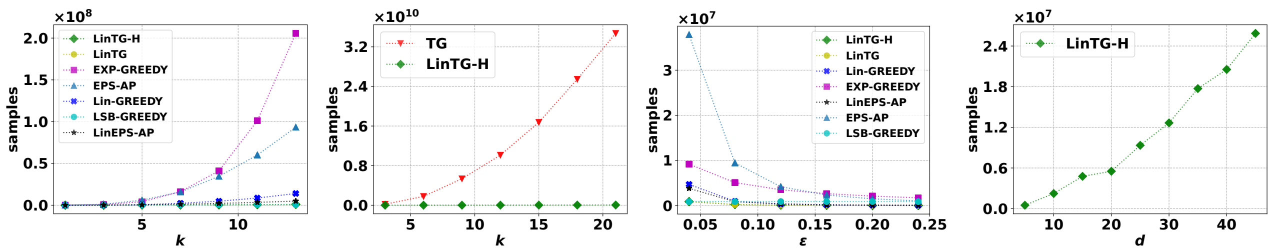
Finding the optimal allocation policy p^* involves solving a LP, which can be costly. If we instead follow a heuristic approach to pulling arms, e.g. repeatedly pulling the arm corresponding to the marginal gain we are currently estimating, then the approximation guarantee of Theorem 3 would still hold. Further discussion about the result in the theorem can be found in Appendix B.

5 EXPERIMENTS

In this section, we conduct experiments on the instance of movie recommendation. The dataset used in the experiments are subsets extracted from the MovieLens 25M dataset Harper and Konstan (2015), which comprises 162,541 ratings for 13,816 movies, each associated with 1,128 topics. We compare the sample complexity of different algorithms with different values of size constraint κ, error parameter ϵ, and different datasets with different values of d. When κ is varied, ϵ is fixed at 0.1. The dataset movie60 contains 60 elements with $d = 5$, $|V| = 500$. The dataset movie5000 contains $n = 5000$ elements with $d = 30$, $|V| = 1000$. When d increases, as has been discussed in the previous section, the computational load for LinTG, Lin-GREEDY would increase heavily and are therefore hard to implement on instances of large d. In contrast, we evaluate the LinTG-H algorithm on instances with different values of d on different datasets to demonstrate the scalability of our algorithm. Here n and $|V|$ are fixed at 500. Additional details about the applications, setup, and results can be found in Section C in the supplementary material.

In the application of movie recommendation, the submodular basis function F_i denotes how well a subset of movies covers the topic i, which is defined by the probabilistic coverage model as presented in the appendix. The noisy marginal gain is sampled in the following way: first we uniformly sample a user a_t from the set of users V, then the noisy marginal gain of adding a new movie x into a subset S is $\Delta f_t(S,x) = \sum_{i\in[d]} w(a_t,i)\Delta F_i(S,x) = \mathbf{w}(a_t,\cdot)^T\Delta\mathbf{F}(S,x)$. Consequently, the exact value of our submodular objective is the average of marginal gain for all elements, i.e., $\Delta f(S,x) = \frac{1}{|V|}\sum_{a\in V}\mathbf{w}(a,\cdot)^T\Delta\mathbf{F}(S,x)$. Denote the average weight vector $\bar{\mathbf{w}}$ as $\bar{\mathbf{w}} = \frac{1}{|V|}\sum_{a\in V}\mathbf{w}(a,\cdot)$, then it follows that $\Delta f(S,x) = \bar{\mathbf{w}}^T\Delta\mathbf{F}(S,x)$.

We compare the solutions returned by the following eight algorithms: (i). The algorithm `LG` ("Lin-GREEDY"). Due to the high computational cost, here we don't solve the linear programming problems and replace the asymptotic optimal allocation ratio p^* with the allocation p that satisfies $p(l',a|l,i,j) = 0.5$ if a is i or j and $l' = l$. Otherwise $p(l',a|l,i,j) = 0$; (ii). The linear greedy algorithm `LinTG`. Here we consider two choices of the allocation parameters. The first one satisfies that $p_m = 1$ and $p_a = 0$ for $a \neq m$, i.e., the algorithm only samples the current marginal gain ("LinTG-H"). The second one is the asymptotic-optimal allocation strategy with ratio p^* as discussed in the algorithm overview and in the appendix ("LinTG"); (iii). The ExpGreedy algorithm from Singla et al. (2016) with the parameter k' set to be 1 ("Exp-GREEDY"); (iv). The CTG algorithm from Chen et al. (2023) ("TG"); (v). The baseline fixed-ϵ approximation algorithm from Chen et al. (2023) ("EPS-AP"); (vi). A fixed-ϵ approximation version of LinTG-H ("LinEPS-AP"). LinEPS-AP exists the while loop from Line 18 to Line 26 in Algorithm 2 in our paper after the



(a) movie60 κ (b) movie5000 κ (c) movie60 ϵ (d) movie500 d

Figure 1: The experimental results of running the algorithms on instances of movie recommendation on the subsets of MovieLens 25M dataset with $n = 60$, $d = 5$ ("movie60") and $n = 5000$, $d = 30$ ("movie5000"), and different datasets with different values of d.

confidence interval β_t shrinks below ϵ; (vii). The regret-minimization algorithm proposed by Yue and Guestrin (2011) (LSB-GREEDY). Notice that this algorithm is developed for the regret-minimization setting where the sample complexity is determined by the input parameter T. To ensure a fair comparison, we matched the sample complexity of LinTG-H and LSB-GREEDY, comparing the function values of the output solution sets. More details about the algorithms can be found in the appendix.

We present the results in terms of the number of evaluations in Figure 1. Due to the high sample complexity and heavy computational burden resulting from solving the linear programming problem, here we only run the algorithms EXP-GREEDY, Lin-GREEDY, LSB-GREEDY, EPS-AP, LinEPS-AP, and LinTG on the relatively small dataset movie60. From the results on the dataset movie60, we can see that the sample complexity of the algorithms that don't consider the linear structure (EXP-GREEDY and TG) is much higher than the algorithms proposed in this paper, which demonstrates the advantages of our proposed methods. In particular, the algorithms LinTG and LinTG-H have the lowest sample complexity compared with other methods. Besides, we notice that the algorithm when ϵ is very small, the algorithm TG has an increase in terms of the sample complexity. From Figure 1(d), we can see the sample complexity of the algorithm LinTG-H increases as d increases, which is consistent with our theoretical results on sample complexity. The regret-minimization algorithm LSB-GREEDY achieves function values comparable to other algorithms on the movie60 dataset when but underperforms on the synthetic dataset, as is shown in Figure 3(b) in Appendix C. This aligns with the differences in objectives between regret-minimization and pure exploration approaches. Both EPS-AP and LinEPS-AP exhibit higher sample complexity compared to LinTG and LinTG-H. This highlights the efficiency of our proposed methods in reducing sample complexity while maintaining strong performance.

6 Acknowledgements

Victoria Crawford is supported in part by the Seed Program for AI, Computing, and Data Science created by the Texas A&M Institute for Data Science, as well as by the Texas A&M University Targeted Proposal Teams (TPT) funding program.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24, 2011.

Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1497–1514. SIAM, 2014.

Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. Combinatorial pure exploration of multi-armed bandits. *Advances in neural information processing systems*, 27, 2014.

Wenjing Chen, Shuo Xing, and Victoria G Crawford. A threshold greedy algorithm for noisy submodular maximization. *arXiv preprint arXiv:2312.00155*, 2023.

Yixin Chen, Ankur Nath, Chunli Peng, and Alan Kuhnle. Discretely beyond $1/e$: Guided combinatorial algorithms for submodular maximization. *arXiv preprint arXiv:2405.05202*, 2024.

Victoria Crawford, Alan Kuhnle, and My Thai. Submodular cost submodular cover with an approximate oracle. In *International Conference on Machine Learning*, pages 1426–1435. PMLR, 2019.

Riccardo De Santi, Manish Prajapat, and Andreas Krause. Global reinforcement learning: Beyond linear and convex rewards via submodular semi-gradient methods. *arXiv preprint arXiv:2407.09905*, 2024.

Rémy Degenne, Pierre Ménard, Xuedong Shang, and Michal Valko. Gamification of pure exploration for linear bandits. In *International Conference on Machine Learning*, pages 2432–2442. PMLR, 2020.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Pac bounds for multi-armed bandit and markov decision processes. In *Computational Learning Theory: 15th Annual Conference on Computational Learning Theory, COLT 2002 Sydney, Australia, July 8–10, 2002 Proceedings 15*, pages 255–270. Springer, 2002.

Tanner Fiez, Lalit Jain, Kevin G Jamieson, and Lillian Ratliff. Sequential experimental design for transductive linear bandits. *Advances in neural information processing systems*, 32, 2019.

Dean P Foster and Alexander Rakhlin. On submodular contextual bandits. *arXiv preprint arXiv:2112.02165*, 2021.

Victor Gabillon, Branislav Kveton, Zheng Wen, Brian Eriksson, and Shanmugavelayutham Muthukrishnan. Adaptive submodular maximization in bandit setting. *Advances in Neural Information Processing Systems*, 26, 2013.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Avinatan Hassidim and Yaron Singer. Robust guarantees of stochastic greedy algorithms. In *International Conference on Machine Learning*, pages 1424–1432. PMLR, 2017a.

Avinatan Hassidim and Yaron Singer. Submodular optimization under noise. In *Conference on Learning Theory*, pages 1069–1122. PMLR, 2017b.

Gaurush Hiranandani, Harvineet Singh, Prakhar Gupta, Iftikhar Ahamath Burhanuddin, Zheng Wen, and Branislav Kveton. Cascading linear submodular bandits: Accounting for position bias and diversity in online learning to rank. In *Uncertainty in Artificial Intelligence*, pages 722–732. PMLR, 2020.

Thibaut Horel and Yaron Singer. Maximization of approximately submodular functions. *Advances in neural information processing systems*, 29, 2016.

Pratik Jawanpuria, Bamdev Mishra, and Karthik S Gurumoorthy. Revisiting stochastic submodular maximization with cardinality constraint: A bandit perspective. *Transactions on Machine Learning Research*.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. Pac subset selection in stochastic multi-armed bandits. In *ICML*, volume 12, pages 655–662, 2012.

Mohammad Karimi, Mario Lucic, Hamed Hassani, and Andreas Krause. Stochastic submodular maximization: The case of coverage functions. *Advances in Neural Information Processing Systems*, 30, 2017.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14:265–294, 1978.

Guanyu Nie, Mridul Agarwal, Abhishek Kumar Umrawal, Vaneet Aggarwal, and Christopher John Quinn. An explore-then-commit algorithm for submodular maximization under full-bandit feedback. In *Uncertainty in Artificial Intelligence*, pages 1541–1551. PMLR, 2022.

Guanyu Nie, Yididiya Y Nadew, Yanhui Zhu, Vaneet Aggarwal, and Christopher John Quinn. A framework for adapting offline algorithms to solve combinatorial multi-armed bandit problems with bandit feedback. In *International Conference on Machine Learning*, pages 26166–26198. PMLR, 2023.

Binghui Peng. Dynamic influence maximization. *Advances in Neural Information Processing Systems*, 34:10718–10731, 2021.

Manish Prajapat, Mojmír Mutnỳ, Melanie N Zeilinger, and Andreas Krause. Submodular reinforcement learning. *arXiv preprint arXiv:2307.13372*, 2023.

Adish Singla, Sebastian Tschiatschek, and Andreas Krause. Noisy submodular maximization via adaptive sampling with applications to crowdsourced image collection summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Marta Soare, Alessandro Lazaric, and Rémi Munos. Best-arm identification in linear bandits. *Advances in Neural Information Processing Systems*, 27, 2014.

Sho Takemori, Masahiro Sato, Takashi Sonoda, Janmajay Singh, and Tomoko Ohkuma. Submodular bandit problem under multiple constraints. In *Conference on Uncertainty in Artificial Intelligence*, pages 191–200. PMLR, 2020.

Andrew Wagenmaker and Aldo Pacchiano. Leveraging offline data in online reinforcement learning. In *International Conference on Machine Learning*, pages 35300–35338. PMLR, 2023.

Liyuan Xu, Junya Honda, and Masashi Sugiyama. A fully adaptive algorithm for pure exploration in linear bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 843–851. PMLR, 2018.

Baosheng Yu, Meng Fang, and Dacheng Tao. Linear submodular bandits with a knapsack constraint. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Yisong Yue and Carlos Guestrin. Linear submodular bandits and their application to diversified retrieval. *Advances in Neural Information Processing Systems*, 24, 2011.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [**Yes**/No/Not Applicable] Clarification: Please see description in Section 2.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [**Yes**/No/Not Applicable] Clarification: Please see the description in Section 3 and 4.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes/No/Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [**Yes**/No/Not Applicable]
   (b) Complete proofs of all theoretical results. [**Yes**/No/Not Applicable]
   (c) Clear explanations of any assumptions. [**Yes**/No/Not Applicable]

   Clarification: Please see the description in Section 3 and 4.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [**Yes**/No/Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes/No/**Not Applicable**]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes/No/**Not Applicable**]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes/No/**Not Applicable**]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [**Yes**/No/Not Applicable]
   (b) The license information of the assets, if applicable. [**Yes**/No/Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes/No/**Not Applicable**]
   (d) Information about consent from data providers/curators. [Yes/No/**Not Applicable**]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Yes/No/**Not Applicable**]

   Clarification: We evaluate our algorithms on the public dataset MovieLens 25M.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Yes/No/**Not Applicable**]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Yes/No/**Not Applicable**]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Yes/No/**Not Applicable**]

   Clarification: We don't use crowdsourcing or conduct research with human subjects.

Supplementary Material for Linear Submodular Maximization with Bandit Feedback

Wenjing Chen
Texas A&M University

Victoria G. Crawford
Texas A&M University

A APPENDIX FOR SECTION 3

In this portion of the appendix, we present missing details and proofs from Section 3 in the main paper. We first present Lemma 1, which guarantees that for any standard greedy-based algorithm which satisfies that the element added at each round has the highest marginal gain within an ϵ/κ-additive error, then the ultimate solution set satisfies that $f(S) \geq (1-e^{-1})f(OPT) - \epsilon$. We then introduce Algorithm `LBSS`, which leverages a static allocation strategy for linear bandit best-arm identification as a subroutine within a standard greedy framework. Finally, we present the missing proofs and details about the theoretical guarantee of `LG` from Section 3 and present the discussion on the result in Theorem 3 in Section A.2. Lemma 1 is given as follows.

Lemma 1. *For any standard greedy-based algorithm proposed for our problem, if with probability at least $1-\delta$, the element a added to the solution set S at each round satisfies that $\Delta f(S,a) \geq \max_{s\in U} \Delta f(S,s) - \epsilon/\kappa$, then it follows that the output solution set S satisfies that with probability at least $1-\delta$*

$$f(S) \geq (1-e^{-1})f(OPT) - \epsilon.$$

Proof. The proof of this lemma is similar to the standard greedy algorithm and can be derived from the proof of ExpGreedy in Singla et al. (2016), so we omit the proof here. □

As we have illustrated in Section 3, we can use any best-arm identification algorithms that satisfy the PAC bound as a subroutine for the standard greedy-based algorithm. Therefore we propose and analyze the static allocation strategy `LBSS` in Section A.1. The pseudocode of `LBSS` is presented in Algorithm 3 in Section A.1.

A.1 Warm-up: static allocation strategy

In this section, we present the warm-up algorithm `LBSS`, which combines the standard greedy algorithm with a subroutine inspired by the static allocation proposed in best-arm identification problem in linear bandit Soare et al. (2014). The strategy is static in that the sequence of sampled feature vectors $\mathbf{x}_{a_t}$ is independent from the sampling results. However, unlike the static allocation algorithm proposed in Soare et al. (2014), which finds the exact optimal solution, here in `LBSS`, the algorithm finds the element with the highest marginal gain within an ϵ/κ additive error. The theoretical guarantee is stated below. In particular, we first give a description of the static strategy. Next, we provide the detailed statement and proofs of Theorem 4.

The algorithm proceeds in rounds. In each round, the algorithm selects the element with the highest marginal gain by taking noisy samples of the marginal gains of adding new elements to the current solution set. For notation simplicity, in the following part, we use $\mathbf{F}_x$ to denote the marginal gain of adding element x to the solution set in the current round S. i.e., $\mathbf{F}_x = \Delta\mathbf{F}(S,x)$. The sampled element is chosen in a greedy manner to make $\max_{x,x'\in U} ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}}$ to decrease as fast as possible. It is worthy to note that in `LBSS`, the update of parameters $\hat{\mathbf{w}}_t$ is calculated based on the least-square estimator. Therefore, $\mathbf{A}_0 = \mathbf{0}_{d\times d}$, $\mathbf{A}_{t+1} = \mathbf{A}_t + \mathbf{x}_{a_t}\mathbf{x}_{a_t}^T$ and $\mathbf{b}_{t+1} = \mathbf{b}_t + r_t\mathbf{x}_{a_t}$, and that $\hat{\mathbf{w}}_{t+1} = \mathbf{A}_{t+1}^{-1}\mathbf{b}_{t+1}$. The parameter D_t is defined as $D_t = R\sqrt{2\log(\frac{\pi^2 t^2 \kappa n^2}{3\delta})}$.

Algorithm 3 `Linear Bandit with Static Strategy` (`LBSS`)

Input: δ, ϵ
Output: solution set $S \subseteq U$
$S \leftarrow \emptyset$
$\mathbf{A} \leftarrow \mathbf{0}_{d\times d}$, $\mathbf{b} \leftarrow \mathbf{0}_d$
for $l = 1, ..., \kappa$ **do**
  $t \leftarrow 1$
  Define $\mathbf{F}_x = \Delta\mathbf{F}(S, x)$
  while true do
    $x_t = \arg\min_{x\in U} \max_{x',x''\in U} (\mathbf{F}_{x'} - \mathbf{F}_{x''})^T(\mathbf{A}_t + \mathbf{F}_x\mathbf{F}_x^T)^{-1}(\mathbf{F}_{x'} - \mathbf{F}_{x''})$
    Obtain the reward r_t of sampling $\mathbf{F}_{x_t}$
    Update $\mathbf{A}_{t+1} = \mathbf{A}_t + \mathbf{F}_{x_t}\mathbf{F}_{x_t}^T$, $\mathbf{b}_{t+1} = \mathbf{b}_t + r_t\mathbf{F}_{x_t}$
    Update $\hat{\mathbf{w}}_{t+1} = \mathbf{A}_{t+1}^{-1}\mathbf{b}_{t+1}$
    if $\exists x \in U$, $\forall x' \in U$, $D_t||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} \leq (\mathbf{F}_x - \mathbf{F}_{x'})^T\hat{\mathbf{w}}_t + \epsilon$ **then**
      return x
    $t \leftarrow t + 1$

A.1.1 proof of Theorem 4

We now present the missing theoretical proofs for Theorem 4, where we provide bounds on the expectation of the reward of the output element and the sample complexity of `LBSS` in Algorithm 3.

Theorem 4. *With probability at least $1 - \delta$, the following statements hold:*

1. *The exact function value of the output solution set S satisfies that $f(S) \geq (1 - e^{-1})f(OPT) - \epsilon$. Here OPT is an optimal solution to the SM problem;*
2. *During each round in `LBSS`, the number of samples required is upper bounded by*

$$t \leq \max\{\frac{256R^2d}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}\log(\frac{256R^2d\pi n}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}\sqrt{\frac{\kappa}{3\delta}}), \frac{d + d^2 + 2}{2}\}$$

where $\Delta_{\min} = \min_{\{x'\in[n], x'\neq x^*\}} \boldsymbol{F}_{x^*}^T\boldsymbol{w} - \boldsymbol{F}_{x'}^T\boldsymbol{w}$ and $x^* = \arg\max_{x\in[n]} \boldsymbol{F}_x^T\boldsymbol{w}$.

From Lemma 1, we can see that the theorem can be proved immediately by Lemma 2, which is presented below.

Lemma 2. *With probability at least $1 - \delta$, in each round of the Algorithm 3, the selected element x satisfies*

$$\boldsymbol{F}_x^T\boldsymbol{w} \geq \boldsymbol{F}_{x^*}^T\boldsymbol{w} - \epsilon/\kappa. \tag{1}$$

Besides, the number of samples required by each round is upper bounded by

$$t \leq \max\{\frac{256R^2d}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}\log(\frac{256R^2d\pi n}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}\sqrt{\frac{\kappa}{3\delta}}), \frac{d + d^2 + 2}{2}\}$$

where $\Delta_{\min} = \min_{\{x'\in[n], x'\neq x^*\}} \boldsymbol{F}_{x^*}^T\boldsymbol{w} - \boldsymbol{F}_{x'}^T\boldsymbol{w}$, and $x^* = \arg\max_{x'\in[n]} \boldsymbol{F}_{x'}^T\boldsymbol{w}$.

Next, we present the proof of Lemma 2.

Proof. Our argument to prove Lemma 2 follows a similar approach as Soare et al. (2014). First, we provide proof of (1) on the expectation of the reward of the returned element. Notice that by Proposition 5 in Appendix D.2, and by applying a union bound over all κ rounds, with probability at least $1 - \delta$, we have that for each round l, and for any $x, x' \in U$,

$$|(\mathbf{F}_{x'} - \mathbf{F}_x)^T(\mathbf{w} - \hat{\mathbf{w}}_t)| \leq ||\mathbf{F}_{x'} - \mathbf{F}_x||_{\mathbf{A}_t^{-1}}D_t. \tag{2}$$

From the stopping condition of the algorithm, we know that when the algorithm stops, the algorithm outputs the element $\mathbf{x}$ such that $\forall \mathbf{x}' \in U$

$$D_t||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} \leq (\mathbf{F}_x - \mathbf{F}_{x'})^T\hat{\mathbf{w}}_t + \epsilon/\kappa.$$

Thus

$$\begin{aligned}(\mathbf{F}_x - \mathbf{F}_{x'})^T \mathbf{w} &= (\mathbf{F}_x - \mathbf{F}_{x'})^T (\mathbf{w} - \hat{\mathbf{w}}_t) + (\mathbf{F}_x - \mathbf{F}_{x'})^T \hat{\mathbf{w}}_t \\ &\geq -D_t ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} + D_t ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} - \epsilon/\kappa \geq -\epsilon/\kappa.\end{aligned}$$

Since the above inequality holds for any element $\mathbf{x'} \in U$, we can conclude the proof. In the following part, we prove the sample complexity for this algorithm. By the stopping condition in `LBSS` in Algorithm 3, we have that during the execution of the algorithm `LBSS`, for any element x, there exists at least one element x' such that

$$D_t ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} \geq (\mathbf{F}_x - \mathbf{F}_{x'})^T \hat{\mathbf{w}}_t + \epsilon/\kappa.$$

Therefore, combining the above inequality with (2), it follows that

$$\begin{aligned}(\mathbf{F}_x - \mathbf{F}_{x'})^T \mathbf{w} &= (\mathbf{F}_x - \mathbf{F}_{x'})^T (\mathbf{w} - \hat{\mathbf{w}}_t) + (\mathbf{F}_x - \mathbf{F}_{x'})^T \hat{\mathbf{w}}_t \\ &\leq 2D_t ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} - \epsilon/\kappa.\end{aligned}$$

Since the above result holds for any $\mathbf{x}$, we can set $\mathbf{x}$ to be the optimal solution $\mathbf{x}^*$, then we would have that

$$(\mathbf{F}_{x^*} - \mathbf{F}_{x'})^T \mathbf{w} \leq 2D_t ||\mathbf{F}_{x^*} - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} - \epsilon/\kappa$$

for some x'. Therefore,

$$\Delta_{\min} \leq 2D_t ||\mathbf{F}_{x^*} - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} - \epsilon/\kappa \leq 2D_t \max_{x,x' \in U} ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} - \epsilon/\kappa \tag{3}$$

where $\Delta_{\min} = \min_{\{x' \in [n], x' \neq a^*\}} \mathbf{F}_{x^*}^T \mathbf{w} - \mathbf{F}_{x'}^T \mathbf{w}$. Notice that by Lemma 7 and the efficient rounding procedure in Soare et al. (2014), it follows that

$$\max_{x,x' \in U} ||\mathbf{F}_x - \mathbf{F}_{x'}||_{\mathbf{A}_t^{-1}} \leq 2\sqrt{\frac{(1 + \frac{d+d^2+2}{2t})d}{t}}.$$

Combine this inequality with (3), we would get

$$4D_t \sqrt{\frac{(1 + \frac{d+d^2+2}{2t})d}{t}} \geq \Delta_{\min} + \epsilon/\kappa.$$

It follows that

$$t \leq \frac{16(1 + \frac{d+d^2+2}{2t})dD_t^2}{(\Delta_{\min} + \epsilon/\kappa)^2} \leq \frac{16(1 + \frac{d+d^2+2}{2t})dD_t^2}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}$$

We then have that

$$t \leq \max\{\frac{32dD_t^2}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}, \frac{d + d^2 + 2}{2}\}.$$

From the definition of D_t, we can get that

$$t \leq \max\{\frac{64R^2 d \log(\frac{n^2 t^2 \pi^2}{3\delta})}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}}, \frac{d + d^2 + 2}{2}\}.$$

From Lemma 14, we conclude that

$$t \leq \max\{\frac{256R^2 d}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}} \log(\frac{256R^2 d\pi n}{\max\{\Delta_{\min}^2, \epsilon^2/\kappa^2\}} \sqrt{\frac{\kappa}{3\delta}}), \frac{d + d^2 + 2}{2}\}.$$

$\square$

A.2 Additional Content to Section 3

In this appendix, we present the missing content in Section 3. First, we provide a detailed discussion on the sampling allocation ratio p^* used in LG in Section A.2.1. Next, we delve into the proof of Theorem 2, which provides the theoretical guarantee concerning the algorithm LG in Section A.2.2. Finally, we present the missing detailed discussion on the result in Section A.2.3.

A.2.1 Discussion on the sample allocation ratio

In our algorithm LG, we use the allocation ratio $p^*(l', a|l, i, j)$. $p^*(l', a|l, i, j)$ is the asymptotic-optimal ratio of sampling $\Delta\mathbf{F}(S_{l'}, a)$ for decreasing the confidence interval $||\Delta\mathbf{F}(S, i) - \Delta\mathbf{F}(S, j)||_{\mathbf{A}_t^{-1}}$ when the number of samples goes to infinity at the greedy round l in LG. Here $\Delta\mathbf{F}(S_{l'}, a)$ is the marginal gain vector of adding element a to set $S_{l'}$. By definition, finding $p^*(l', a|l, i, j)$ means to shrink $||\Delta\mathbf{F}(S, i) - \Delta\mathbf{F}(S, j)||_{\mathbf{A}_t^{-1}}$ as fast as possible. Therefore, finding $p^*(l', a|l, i, j)$ is equivalent to solving the problem below in the case where n goes to infinity.

$$\arg\min_{\{p(l',a|l,i,j)\}} (\Delta\mathbf{F}(S,i) - \Delta\mathbf{F}(S,j))^T(\sum_{l'\in[l],a\in[n]} p(l', a|l, i, j)\mathbf{F}(S_{l'}, a)\mathbf{F}(S_{l'}, a)^T + \frac{\lambda}{n}I)^{-1}(\Delta\mathbf{F}(S,i) - \Delta\mathbf{F}(S,j))$$
$$s.t. \sum_{l'\in[l],a\in[n]} p(l', a|l, i, j) = 1.$$

By Lemma 13, the optimal solution $p^*(l', a|l, i, j)$ of the optimal problem above satisfies that

$$p^*(l', a|l, i, j) = \frac{|w^*_{l',a}(i, j)|}{\sum_{l''=1}^{l}\sum_{a'=1}^{n} |w^*_{l'',a'}(i, j)|}, \tag{4}$$

where $\{w^*_{l',a}(i, j)\}_{a\in[n]}$ is the optimal solution of the linear program in (5) below

$$\arg\min_{\{w_{l',a}(i,j)\}} \sum_{l'=1}^{l}\sum_{a=1}^{n} |w_{l',a}(i, j)|$$
$$s.t.\ \Delta\mathbf{F}(S, i) - \Delta\mathbf{F}(S, j) = \sum_{l'=1}^{l}\sum_{a=1}^{n} w_{l',a}(i, j)\Delta\mathbf{F}(S_{l'}, a). \tag{5}$$

A.2.2 Proof of Theorem 2

In this section, we first prove Theorem 2, which guarantees the sample complexity and approximation ratio of LG. From Lemma 1, we can see that proving Theorem 2 is equivalent to proving the following result.

Lemma 3. *With probability at least $1 - \delta$, in each round of the LG in Algorithm 1, the selected element a satisfies*

$$\Delta\boldsymbol{F}^T(S, a)\boldsymbol{w} \geq \max_{x\in[n]} \Delta\boldsymbol{F}^T(S, x))\boldsymbol{w} - \epsilon/\kappa$$

Besides, assume $\lambda \leq \frac{2R^2}{Q^2}\log\frac{1}{\delta}$, then during the round l, the total number of samples τ satisfies

$$\tau \leq \sum_{l'\in[l],a\in[n]} \max\{4H^{(l)}_{l',a}R^2(2\log\frac{1}{\delta} + d\log M^{(l)}) + 1 - N^{(l)}_{l',a}, 0\},$$

where $M^{(l)} = \frac{16H^{(l)}_\epsilon R^2L^2}{\lambda}\log(\frac{1}{\delta}) + \frac{32(H^{(l)}_\epsilon)^2R^4L^4}{\lambda^2} + \frac{2(N^{(l)}+nl)L^2}{\lambda d} + 2$. Here $H^{(l)}_{l',a} = \max_{i,j\in[n]} \frac{p^*(l',a|l,i,j)\rho^l_{i,j}}{\max\{\epsilon/\kappa, \frac{\Delta_{l,i}\vee\Delta_{l,j}+\epsilon/\kappa}{3}\}^2}$ and $H^{(l)}_\epsilon = \sum_{l'\in[l],a\in[n]} H^{(l)}_{l',a}$, where $\rho^l_{i,j}$ is the optimal value of (5).

Proof. Throughout the proof, we assume the clean event $\mathcal{E}$ occurs. The event $\mathcal{E}$ is defined as

$$\mathcal{E} = \{\forall t > 0, \forall i, j \in [n], |(\Delta\mathbf{F}(S, i) - \Delta\mathbf{F}(S, j))^T(\hat{\mathbf{w}} - \mathbf{w})| \leq \beta_t(i, j)\},$$

where the confidence interval $\beta_t(i, j) = C_t||\Delta\mathbf{F}(S, i) - \Delta\mathbf{F}(S, j)||^{-1}_{\mathbf{A}_t}$ and that $C_t = R\sqrt{2\log\frac{\det(\mathbf{A}_t)^{\frac{1}{2}}\det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}}Q$. Then by Proposition 1, and the fact that for all i, $\Delta\mathbf{F}(S, i)$ only depends on the past samples of marginal gains, we have the following lemma.

Lemma 4. *With probability at least $1-\delta$, event $\mathcal{E}$ holds during each greedy round of* `LG`.

First of all, we prove the result on the correctness of the output element i_t. Let us denote that $a^* = \arg\max_{a\in[n]} \Delta\mathbf{F}(S,a)^T\mathbf{w}$, and that $\epsilon' = \epsilon/\kappa$. By the stopping condition in `LG`, we have that

$$B(t) = (\Delta\mathbf{F}(S,j_t) - \Delta\mathbf{F}(S,i_t))^T\hat{\mathbf{w}} + \beta_t(i_t,j_t) \leq \epsilon'.$$

By the selection strategy of j_t, we have that $B(t) \geq (\Delta\mathbf{F}(S,a^*) - \Delta\mathbf{F}(S,i_t))^T\hat{\mathbf{w}} + \beta_t(a^*,i_t)^T$. Therefore, by Lemma 4, we have that

$$(\Delta\mathbf{F}(S,a^*) - \Delta\mathbf{F}(S,i_t))^T\mathbf{w} \leq (\Delta\mathbf{F}(S,a^*) - \Delta\mathbf{F}(S,i_t))^T\hat{\mathbf{w}} + \beta_t(a^*,i_t) \leq \epsilon'.$$

Next, we prove the theoretical guarantee of the sample complexity. From Lemma 4 in the appendix of Xu et al. (2018), we have

Lemma 5. *Under event $\mathcal{E}$, throughout the execution of* `LG`*, $B(t)$ can be bounded as*

$$B(t) \leq \min(0, -(\Delta_{i_t} \vee \Delta_{j_t}) + 2\beta_t(i_t,j_t)) + \beta_t(i_t,j_t).$$

The proof is the same as in Xu et al. (2018), so we omit the proof here. First of all, we derive the bound on the total number of samples for each element a from the sampling set $\mathcal{X}$. Then by Lemma 5 and that $B(t) \geq \epsilon'$ we have that

$$\beta_t(i_t,j_t) \geq \max\{\epsilon', \frac{\Delta_{i_t} \vee \Delta_{j_t} + \epsilon'}{3}\}.$$

By the definition of the confidence interval $\beta_t(i_t,j_t)$, we have that

$$||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||_{\mathbf{A}_t^{-1}} \geq \frac{\max\{\epsilon', \frac{\Delta_{i_t}\vee\Delta_{j_t}+\epsilon'}{3}\}}{C_t}.$$

Since we have

$$\begin{aligned}
&||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||_{\mathbf{A}_t^{-1}} \qquad (6)\\
=&\sqrt{(\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t))^T\mathbf{A}_t^{-1}(\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t))}\\
=&\sqrt{(\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t))^T(\sum_{l''\in[l]}\sum_{a'\in[n]} T_{l'',a'}(t)\Delta\mathbf{F}(S_{l''},a')\Delta\mathbf{F}(S_{l''},a')^T))^{-1}(\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t))}. \qquad (7)
\end{aligned}$$

Suppose the marginal gain of adding an element a to subset $S_{l'}$ is sampled at time t, from the element selection strategy, we have that for any other past marginal gain vectors, it follows that for any $l'' \in [l]$ and $a' \in [n]$,

$$\frac{T_{l',a}(t)}{p^*(l',a|l,i_t,j_t)} \leq \frac{T_{l'',a'}(t)}{p^*(l'',a'|l,i_t,j_t)}.$$

Combine (6) with the above inequality, it follows that

$$||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||_{\mathbf{A}_t^{-1}} \leq \sqrt{\frac{p^*(l',a|l,i_t,j_t)}{T_{l',a}(t)}}||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||_{\Lambda_{i_t,j_t}^{-1}}$$

where we define

$$\Lambda_{i,j} = \sum_{l''\in[l]}\sum_{a'\in[n]} p^*(l'',a'|l,i,j)\Delta\mathbf{F}(S_{l''},a')\Delta\mathbf{F}^T(S_{l''},a'). \qquad (8)$$

Therefore, we have

$$\begin{aligned}
T_{l',a}(t) &\leq \frac{p^*(l',a|l,i_t,j_t)||\Delta\mathbf{F}(S,i_t) - \Delta\mathbf{F}(S,j_t)||^2_{\Lambda_{i_t,j_t}^{-1}}C_t^2}{\max\{\epsilon', \frac{\Delta_{i_t}\vee\Delta_{j_t}+\epsilon'}{3}\}^2}\\
&\leq \max_{i,j\in[n]} \frac{p^*(l',a|l,i,j)||\Delta\mathbf{F}(S,i) - \Delta\mathbf{F}(S,j)||^2_{\Lambda_{i,j}^{-1}}C_t^2}{\max\{\epsilon', \frac{\Delta_i\vee\Delta_j+\epsilon'}{3}\}^2}
\end{aligned}$$

Let τ' denote the total number of samples in round l, encompassing both the offline dataset and the samples acquired during the execution of LG. Consequently, upon termination of the algorithm, the cumulative number of samples associated with element a is bounded by:

$$T_{l',a}(\tau') \leq \max_{i,j\in[n]} \frac{p^*(l',a|l,i,j)||\Delta\mathbf{F}(S,i)-\Delta\mathbf{F}(S,j)||^2_{\Lambda^{-1}_{i,j}} C^2_{\tau'}}{\max\{\epsilon', \frac{\Delta_i \vee \Delta_j + \epsilon'}{3}\}^2} + 1. \tag{9}$$

Notice that we can reuse $N^{(l)}_{l',a}$ samples from the past rounds, therefore, the total number of samples of element a in round l in LG is bounded by

$$\max\{T_{l',a}(\tau') - N^{(l)}_{l',a}, 0\}.$$

The total number of samples for all the elements in round l during the execution of LG is bounded by

$$\tau \leq \sum_{l'\in[l]} \sum_{a\in[n]} \max\{T_{l',a}(\tau') - N^{(l)}_{l',a}, 0\}.$$

Combine this inequality with (9), we get that

$$\begin{aligned}\tau &\leq \sum_{l'\in[l]} \sum_{a\in[n]} \max\{\max_{i,j\in[n]} \frac{p^*(l',a|l,i,j)||\Delta\mathbf{F}(S,i)-\Delta\mathbf{F}(S,j)||^2_{\Lambda^{-1}_{i,j}} C^2_{\tau'}}{\max\{\epsilon', \frac{\Delta_i \vee \Delta_j + \epsilon'}{3}\}^2} + 1 - N^{(l)}_{l',a}, 0\} \\ &\leq \sum_{l'\in[l]} \sum_{a\in[n]} \max\{\max_{i,j\in[n]} \frac{p^*(l',a|l,i,j)||\Delta\mathbf{F}(S,i)-\Delta\mathbf{F}(S,j)||^2_{\Lambda^{-1}_{i,j}} C^2_{\tau+N^{(l)}}}{\max\{\epsilon', \frac{\Delta_i \vee \Delta_j + \epsilon'}{3}\}^2} + 1 - N^{(l)}_{l',a}, 0\}\end{aligned}$$

where $N^{(l)} = \sum_{l'\in[l]} \sum_{a\in[n]} N^{(l)}_{l',a}$ is the total number of samples in the offline dataset.

From Lemma 13 and the discussion given in Appendix B in Xu et al. (2018), we have that the allocation ratio p^* is the optimal solution of the optimization problem as defined in (4) and (5), then it follows that

$$||\Delta\mathbf{F}(S,i)-\Delta\mathbf{F}(S,j)||^2_{\Lambda^{-1}_{i,j}} \leq \rho^l_{i,j}.$$

where $\rho^l_{i,j}$ is the optimal value of (5) in the round l.

Let us denote $\Delta_{l,i}$ to be the gap between the marginal gain of adding the element i and the element with the highest marginal gain at round l, i.e., $\Delta_{l,i} = \max_{x\in[n]} \Delta\mathbf{F}(S_l,x)^T\mathbf{w} - \Delta\mathbf{F}(S_l,i)^T\mathbf{w}$. Let us denote $H^{(l)}_{l',a} = \max_{i,j\in[n]} \frac{p^*(l',a|l,i,j)\rho^l_{i,j}}{\max\{\epsilon/\kappa, \frac{\Delta_{l,i}\vee\Delta_{l,j}+\epsilon/\kappa}{3}\}^2}$ and $H^{(l)}_\epsilon = \sum_{l'\in[l],a\in[n]} H^{(l)}_{l',a}$. Then the inequality above is

$$\tau \leq \sum_{l'\in[l]} \sum_{a\in[n]} \max\{H^{(l)}_{l',a} C^2_{\tau+N} + 1 - N^{(l)}_{l',a}, 0\}. \tag{10}$$

It then follows that

$$\tau \leq H^{(l)}_\epsilon C^2_{\tau+N} + nl. \tag{11}$$

By definition, we have that

$$C_t = R\sqrt{2\log\frac{\det(\mathbf{A}_t)^{\frac{1}{2}}\det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}}Q.$$

By Lemma 12, we can get that

$$C_t \leq R\sqrt{2\log\frac{1}{\delta} + d\log(1+\frac{tL^2}{\lambda d})} + \lambda^{\frac{1}{2}}Q.$$

Since $\lambda \leq \frac{2R^2}{Q^2}\log\frac{1}{\delta}$, it follows that

$$C_t \leq 2R\sqrt{2\log\frac{1}{\delta} + d\log(1+\frac{tL^2}{\lambda d})}. \tag{12}$$

Combine (11) with (12), we can get

$$\tau \leq 4H_\epsilon^{(l)}R^2\big(2\log\frac{1}{\delta} + d\log(1+\frac{(\tau+N^{(l)})L^2}{\lambda d})\big) + nl$$

It then follows that

$$\frac{(\tau+N^{(l)})L^2}{\lambda d} + 1 \leq \frac{4H_\epsilon^{(l)}R^2L^2}{\lambda d}\big(2\log\frac{1}{\delta} + d\log(1+\frac{(\tau+N^{(l)})L^2}{\lambda d})\big) + \frac{(N^{(l)}+nl)L^2}{\lambda d} + 1$$

By Lemma 15, we have that

$$\begin{aligned}\frac{(\tau+N^{(l)})L^2}{\lambda d} + 1 &\leq \frac{16H_\epsilon^{(l)}R^2L^2}{\lambda d}\log\frac{1}{\delta} + \frac{8H_\epsilon^{(l)}R^2L^2}{\lambda}\log(\frac{4H_\epsilon R^2L^2}{\lambda}) + \frac{2(N^{(l)}+nl)L^2}{\lambda d} + 2\\ &\leq \frac{16H_\epsilon^{(l)}R^2L^2}{\lambda}\log\frac{1}{\delta} + \frac{32(H_\epsilon^{(l)})^2R^4L^4}{\lambda^2} + \frac{2(N^{(l)}+nl)L^2}{\lambda d} + 2.\end{aligned}$$

Plugging the above inequality into (10), it follows that

$$\tau \leq \sum_{l'\in[l]}\sum_{a\in[n]}\max\{4H_{l',a}^{(l)}R^2(2\log\frac{1}{\delta} + d\log M) + 1 - N_{l',a}^{(l)}, 0\}.$$

where $M = \frac{16H_\epsilon^{(l)}R^2L^2}{\lambda}\log\frac{1}{\delta} + \frac{32(H_\epsilon^{(l)})^2R^4L^4}{\lambda^2} + \frac{2(N^{(l)}+nl)L^2}{\lambda d} + 2.$ □

A.2.3 Discussion on the result in Theorem 2

In this part of the appendix, we present omitted discussion about the implications of Theorem 2, and the comparison of the sample complexity in Theorem 2 to the result in Xu et al. (2018).

We first analyze the case where past samples are not leveraged. As is discussed in the algorithm overview of LG, selecting an element to add to the solution set corresponds to a best-arm identification problem in linear bandits. Thus, we can compare our results with the element-selecting procedure being replaced by LinGapE in Xu et al. (2018), which does not leverage samples of noisy marginal gains from previous rounds of the greedy algorithm. The sample complexity of LinGapE is $\tau_2 = 4H_\epsilon^{(l)}R^2(2\log\frac{\kappa n^2}{\delta} + d\log M^{(l)}) + n$. Here $H_\epsilon^{(l)} = \sum_{a\in[n]}\max_{i,j}\frac{p^*(l,a|l,i,j)\rho_{i,j}^l}{\max\{\epsilon/\kappa, \frac{\Delta_{l,i}\vee\Delta_{l,j}+\epsilon/\kappa}{3}\}^2}$. In contrast, if we don't leverage samples from the past ($N_{l',a}^{(l)} = 0$) and only sample marginal gains of the current greedy round, LG achieves a sample complexity of $4H_\epsilon^{(l)}R^2(2\log(\frac{1}{\delta}) + d\log M^{(l)}) + n$. Therefore, the result in Xu et al. (2018) is worse than the sample complexity of LG in the case where we don't utilize samples from the past by a factor of $O(\log n^2)$.

Next, we consider our result in the case of the general bandit by setting the marginal gain to be $\{\Delta\mathbf{F}(S_l,a)\}_{l\in[\kappa],a\in[n]} = (\mathbf{e}_1,\mathbf{e}_2,...,\mathbf{e}_{\kappa n})$. This leads to $p^*(l',a|l,i,j) = 0.5$ for $l' = l$ and $a = i$ or j, and $p^*(l',a|l,i,j) = 0$ otherwise from the result in Equations (19) and (4). In fact, the allocation strategy in this case is the same as that used in the CLUCB algorithm in Chen et al. (2014). Both these two algorithms sample the element with a smaller confidence interval between i_t and j_t. In this case, $H_\epsilon^{(l)} = \sum_{a\in[n]}\frac{2}{\max\{\epsilon/\kappa, \frac{\Delta_{l,a}+\epsilon/\kappa}{3}\}^2}$, and the sample complexity becomes $8H_\epsilon^{(l)}R^2\log\frac{1}{\delta} + n + O(H_\epsilon^{(l)}dR^2\log\frac{H_\epsilon^{(l)}RL}{\lambda})$. The second term $O(H_\epsilon^{(l)}dR^2\log\frac{H_\epsilon^{(l)}RL}{\lambda})$ reflects the exploitation of linear structure while the first term matches the sample complexity result under the general bandit setting in Chen et al. (2014) up to a log-factor, which is $O(H_\epsilon^{(l)}R^2\log\frac{H_\epsilon^{(l)}R^2\kappa}{\delta})$.

Finally, we have to notice that to find the asymptotic sample selection ratio $p^*(l',a|l,i,j)$, the algorithm involves solving a linear programming problem for each i,j at each round l to find $\{p^*(l',a|l,i,j)\}_{l'\in[l],a\in[n]}$, which results in solving a total of $O(\kappa n^2)$ number of the optimization problems. In particular, as is illustrated in the proof of Theorem 2 in Section A.2.2, we can replace the asymptotic allocation ratio $p^*(l', a|l, i, j)$ with any allocation policy $p(l', a|l, i, j)$. The result on the upper bound of the sample complexity in the theorem would still hold, with the value of $\rho^l_{i,j}$ being replaced by $||\Delta\mathbf{F}(S, i) - \Delta\mathbf{F}(S, j)||^2_{\Lambda^{-1}_{i,j}}$ where $\Lambda^{-1}_{i,j}$ is defined in (8).

In particular, by setting the allocation ratio to $p(l', a|l, i, j) = 0.5$ for $l' = l$ and $a = i$ or j, and $p(l', a|l, i, j) = 0$ otherwise, the sampling strategy follows the same idea as the CLUCB algorithm. Although this allocation ratio is not asymptotically optimal and follows a similar principle to the CLUCB algorithm, there are notable differences: (i) Our algorithm `LG` leverages the linear structure, and the concentration inequality used is derived from the linear bandit literature. (ii). The linear submodular bandit algorithm allows us to exploit sampling results from past greedy rounds while in the CLUCB algorithm developed for the general bandit setting, we are not allowed to do that.

B APPENDIX FOR SECTION 4

In this section, we present the proofs and discussions of the theoretical results from Section 4. First, in Section B.1, we present and prove the Lemma 6 concerning the theoretical guarantee on the value of the evaluated marginal gain and sample complexity of `LinTG`. Next, we apply the theoretical results of Lemma 6 and prove the final theoretical guarantee for the linear threshold greedy algorithm `LinTG` in Algorithm 2. Finally, in Section B.3, we present the discussion and analysis of the result in Theorem 3.

B.1 Proof of Lemma 6

We prove the theoretical guarantee about evaluation of each marginal gain in `LinTG`. First of all, we present the detailed statements of Lemma 6. First, we notice that by Lemma 13, we have that p^* at the m-th evaluation of marginal gain can be calculated by

$$p_i^* = \frac{|w_i^*|}{\sum_{i\in[m]}|w_i^*|} \tag{13}$$

where w_i^* is the optimal solution to the linear programming problem below.

$$\arg\min_{\{w_i\}} \sum_{i\in[m]} |w_i| \qquad s.t. \qquad \Delta\mathbf{F}(S_m,a_m) = \sum_{i=1}^m w_i \Delta\mathbf{F}(S_i,a_i). \tag{14}$$

In addition, we denote the optimal value of the above optimization problem as $\rho^{(m)}$. In our experiments, we also consider the allocation ratio of $p_i = 0$ for $i \neq m$ and $p_m = 1$, i.e., the algorithm always samples the marginal gain $\Delta f(S_m,a_m)$ to determine whether $\Delta f(S_m,a_m)$ is approximately above w or not.

Lemma 6. *During the m-th evaluation on the marginal gain of adding element a_m to S_m in `LinTG`, we have that with probability at least $1-\delta/2$, the following statement holds.*

1. *Assuming $\lambda \le \frac{2R^2}{Q^2}\log\frac{2}{\delta}$, then it takes at most*

   $$\sum_{i\in[m]} \max\{4H_i^{(m)} R^2 (2\log\frac{2}{\delta} + d\log M^{(m)}) + 1 - N_i^{(m)}, 0\}$$

   samples to evaluate the m-th marginal gain, where $M^{(m)} = \frac{16H_\epsilon^{(m)}R^2L^2}{\lambda}\log\frac{2}{\delta} + \frac{32(H_\epsilon^{(m)})^2R^4L^4}{\lambda^2} + \frac{2(N^{(m)}+m)L^2}{\lambda d} + 2$. Here $H_i^{(m)} = \frac{p_i^*\rho^{(m)}}{\max\{\frac{\epsilon/\kappa + |w - \Delta F(S,a)^T w|}{2}, \epsilon/\kappa\}^2}$ and $H_\epsilon^{(m)} = \sum_{i\in[m]} H_i^{(m)}$. p_i^* is the fraction of samples allocated to sample $\Delta\boldsymbol{F}(S_i,a_i)$ and therefore $\sum_{i\in[m]} p_i^* = 1$. $\rho^{(m)}$ is the optimal value of

   $$\arg\min_{\{w_i\}} \sum_{i\in[m]} |w_i| \qquad s.t. \qquad \Delta\boldsymbol{F}(S_m,a_m) = \sum_{i=1}^m w_i \Delta\boldsymbol{F}(S_i,a_i). \tag{15}$$

2. *If the element is added to the solution set, then $\Delta\boldsymbol{F}(S_m,a_m)^T\boldsymbol{w} \ge w - \epsilon/\kappa$. Otherwise, $\Delta\boldsymbol{F}(S_m,a_m)^T\boldsymbol{w} \le w + \epsilon/\kappa$.*

For notation simplicity, throughout the proof, we define $\epsilon' = \epsilon/\kappa$. To prove this lemma, We first present the following Lemma 7 and Lemma 8 about our algorithm `LinTG`. Once these lemmas are proven, we next prove the main result of Lemma 6 using the lemmas. We start by introducing the following "clean event".

$$\mathcal{E} = \{|\Delta\mathbf{F}(S_m,a_m)^T(\hat{\mathbf{w}} - \mathbf{w})| \le \beta_t, \forall t\}$$

Here β_t is the confidence interval and is defined as $\beta_t = C_t||\Delta\mathbf{F}(S_m,a_m)||_{\mathbf{A}_t^{-1}} = (R\sqrt{2\log\frac{2\det(\mathbf{A}_t)^{\frac{1}{2}}\det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}}Q)||\Delta\mathbf{F}(S_m,a_m)||_{\mathbf{A}_t^{-1}}$. From the fact that $\Delta\mathbf{F}(S_m,a_m)$ only depends on the offline dataset, i.e., the history up to time t, we can apply the result in Proposition 1. It follows that $P(\mathcal{E}) \ge 1 - \frac{\delta}{2}$. Therefore, we have the following lemma.

Lemma 7. *With probability at least $1-\delta/2$, it holds that*

$$P(\mathcal{E}) \geq 1-\delta.$$

Next, we prove the bound on the confidence interval β_t.

Lemma 8. *Conditioned on the event $\mathcal{E}$, the confidence interval β_t satisfies that*

$$\beta_t \geq \max\{\frac{\epsilon' + |w - \Delta\boldsymbol{F}(S_m, a_m)^T\boldsymbol{w}|}{2}, \epsilon'\}$$

before the evaluation of the marginal gain ends.

Proof. It is equivalent to prove that when $\beta_t < \frac{\epsilon'+|w-\Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}$, the evaluation of the marginal gain ends. If $\beta_t < \frac{\epsilon'+w-\Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}}{2}$, then we have $\Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \leq w + \epsilon' - 2\beta_t$. From Lemma 4, we have that with probability at least $1-\delta$, it holds that $\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} - \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \leq \beta_t$. Therefore,

$$\begin{aligned}
&\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} + \beta_t \\
&\leq (\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} - \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w}) + \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} + \beta_t \\
&\leq w + \epsilon'.
\end{aligned}$$

Thus the algorithm ends.

Similarly, we consider the case where $\beta_t < \frac{\epsilon'-w+\Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}}{2}$. In this case, we have that $\Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \geq 2\beta_t + w - \epsilon'$. Notice that conditioned on the clean event defined in Lemma 4, we have that $\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} - \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \geq -\beta_t$. Then

$$\begin{aligned}
\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} - \beta_t &\geq \Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} - \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \\
&\quad + \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} - \beta_t \\
&\geq -\beta_t + 2\beta_t \\
&\quad + w - \epsilon' - \beta_t \\
&= w - \epsilon'.
\end{aligned}$$

Therefore, the algorithm ends. In the case where $\beta_t \leq \epsilon'$, then $w + \epsilon' - \beta_t \geq w - \epsilon' + \beta_t$. Therefore, either we have $\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} \geq w + \beta_t - \epsilon'$ or $\Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} \leq w - \beta_t + \epsilon'$. Thus the the evaluation of the m-th marginal gain ends. □

Now we can prove Lemma 6.

Proof. First of all, we prove the second result in the Lemma 6. Conditioned on the event $\mathcal{E}$, we have that if the a_m element is added to S_m, then

$$\Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \geq \Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} - \beta_t \geq w - \epsilon'.$$

If the a_m element is not added to S_m, then

$$\Delta\mathbf{F}(S_m, a_m)^T\mathbf{w} \leq \Delta\mathbf{F}(S_m, a_m)^T\hat{\mathbf{w}} + \beta_t \leq w + \epsilon'.$$

Next, we prove our first result, which provides the theoretical guarantee of sample complexity. The proof depends on Lemma 8. By Lemma 8, and the definition of β_t, we have that

$$C_t||\Delta\mathbf{F}(S_m, a_m)||_{\mathbf{A}_t^{-1}} \geq \max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m, a_m)^T\mathbf{w}|}{2}, \epsilon'\}.$$

Similar to the proof in Lemma 5, we have that the sampled element i_t satisfies

$$||\Delta\mathbf{F}(S_m, a_m)||_{\mathbf{A}_t^{-1}} \leq \sqrt{\frac{p^*_{i_t}}{T_{i_t}(t)}}||\Delta\mathbf{F}(S_m, a_m)||_{\Lambda^{-1}},$$

where $\Lambda = \sum_{i\in[m]} p_i^* \Delta\mathbf{F}(S_i, a_i)\Delta\mathbf{F}(S_i, a_i)^T$. Therefore, before the marginal gain of adding a_{i_t} to S_{i_t} is sampled at time t, we have

$$T_{i_t}(t) \leq \frac{p_{i_t}^* ||\Delta\mathbf{F}(S_m, a_m)||_{\Lambda^{-1}}^2 C_t^2}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2}.$$

For any other marginal gain with index i, suppose $\tilde{t}$ is the last time $\Delta\mathbf{F}(S_i, a_i)$ is sampled, and that τ' is the total number of samples, then

$$T_i(\tau') = T_i(\tilde{t}) + 1 \leq \frac{p_i^* ||\Delta\mathbf{F}(S_m, a_m)||_{\Lambda^{-1}}^2 C_{\tau'}^2}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2} + 1.$$

Since $T_i(\tau')$ is the total number of samples to the marginal gain $\Delta\mathbf{F}(S_i, a_i)$ including the past samples, then the total number of samples of $\Delta\mathbf{F}(S_i, a_i)$ during the evaluation of $\Delta\mathbf{F}(S_m, a_m)$ is upper bounded by

$$\max\{\frac{p_i^* ||\Delta\mathbf{F}(S_m, a_m)||_{\Lambda^{-1}}^2 C_{\tau'}^2}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2} + 1 - N_i^{(m)}, 0\}.$$

Summing over all elements, we would get

$$\begin{aligned}
\tau &\leq \sum_{i\in[m]} \max\{\frac{p_i^* ||\Delta\mathbf{F}(S_m, a_m)||_{\Lambda^{-1}}^2 C_{\tau'}^2}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2} + 1 - N_i^{(m)}, 0\} \\
&\leq \sum_{i\in[m]} \max\{\frac{p_i^* ||\Delta\mathbf{F}(S_m, a_m)||_{\Lambda^{-1}}^2 C_{\tau+N^{(m)}}^2}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2} + 1 - N_i^{(m)}, 0\},
\end{aligned}$$

where $N^{(m)}$ is the total number of samples in the offline dataset. By Lemma 13, it follows that

$$\tau \leq \sum_{i\in[m]} \max\{\frac{p_i^* \rho^{(m)} C_{\tau+N^{(m)}}^2}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2} + 1 - N_i^{(m)}, 0\},$$

where $\rho^{(m)}$ is the optimal value of the optimization problem in (14). Let us denote $H_i^{(m)} = \frac{p_i^* \rho^{(m)}}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2}$ and $H_\epsilon^{(m)} = \sum_{i\in[m]} H_i^{(m)} = \frac{\rho^{(m)}}{\max\{\frac{\epsilon' + |w - \Delta\mathbf{F}(S_m,a_m)^T\mathbf{w}|}{2}, \epsilon'\}^2}$. Then the inequality above is

$$\tau \leq \sum_{i\in[m]} \max\{H_i^{(m)} C_{\tau+N^{(m)}}^2 + 1 - N_i^{(m)}, 0\}. \tag{16}$$

It then follows that

$$\tau \leq H_\epsilon^{(m)} C_{\tau+N^{(m)}}^2 + m \tag{17}$$

Similar to the proof of Theorem 3, we have that

$$\tau \leq \sum_{i\in[m]} \max\{4H_i^{(m)} R^2 (2\log\frac{2}{\delta} + d\log M^{(m)}) + 1 - N_i^{(m)}, 0\}.$$

where $M^{(m)} = \frac{16H_\epsilon^{(m)} R^2 L^2}{\lambda} \log\frac{2}{\delta} + \frac{32(H_\epsilon^{(m)})^2 R^4 L^4}{\lambda^2} + \frac{2(N^{(m)} + m)L^2}{\lambda d} + 2.$ □

B.2 Proof of Theorem 3

In this section, we present the proof of Theorem 3.

Theorem 3. *`LinTG` makes $n\log(\kappa/\alpha)/\alpha$ evaluations of the marginal gains. In addition, with probability at least $1-\delta$, the following statements hold:*

1. *The function value of the output solution set S satisfies that $f(S) \geq (1 - e^{-1} - \alpha) f(OPT) - 2\epsilon$. Here OPT is an optimal solution to the SM problem;*

2. *Assuming $\lambda \leq \frac{2R^2}{Q^2} \log \frac{2}{\delta}$, the m-th evaluation of the marginal gain of adding an element a to S in `LinTG` takes at most*

$$\sum_{i \in [m]} \max\{4H_i^{(m)} R^2 (2 \log \frac{2}{\delta} + d \log M^{(m)}) + 1 - N_i^{(m)}, 0\}$$

samples, where $M^{(m)} = \frac{16 H_\epsilon^{(m)} R^2 L^2}{\lambda} \log \frac{2}{\delta} + \frac{32 (H_\epsilon^{(m)})^2 R^4 L^4}{\lambda^2} + \frac{2(N^{(m)} + m) L^2}{\lambda d} + 2$. Here $H_i^{(m)} = \frac{p_i^* \rho^{(m)}}{\max\{\frac{\epsilon/\kappa + |w - \Delta \boldsymbol{F}(S_m, a_m)^T \boldsymbol{w}|}{2}, \epsilon/\kappa\}^2}$ and $H_\epsilon^{(m)} = \sum_{i \in [m]} H_i^{(m)}$. p_i^* is the fraction of samples allocated to sample $\Delta \boldsymbol{F}(S_i, a_i)$. $N^{(m)}$ is the total number of samples before the m-th evaluation of marginal gains. $N_i^{(m)}$ is the number of samples to the marginal gain $\Delta f(S_i, a_i)$ before the m-th evaluation of marginal gains.

Next, we present the proof of Theorem 3. The second result can be obtained by applying the result in Lemma 6. To prove the first result, we first present a series of needed lemmas. In order for the guarantees of Theorem 3 to hold, two random events must occur during `LG`. The first event is that the estimate of the max singleton value of f on Line 5 in `LG` is an ϵ-approximation of its true value. More formally, we have the following lemma.

Lemma 9. *With probability at least $1 - \delta/3$, we have $\max_{s \in U} f(s) - \epsilon' \leq g \leq \max_{s \in U} f(s) + \epsilon'$.*

Proof. For a fix $s \in U$, by Hoeffding's inequality we would have that

$$P(|\hat{f}(s) - f(s)| \geq \epsilon') \leq \frac{\delta}{3n}. \tag{18}$$

Taking a union bound over all elements we would have that

$$P(\exists s \in U, s.t. |\hat{f}(s) - f(s)| \geq \epsilon') \leq \frac{\delta}{3}.$$

Then with probability at least $1 - \frac{\delta}{3}$, $|\hat{f}(s) - f(s)| \leq \epsilon'$ for all $s \in U$. It then follows that $\forall s \in U$, $f(s) - \epsilon' \leq \hat{f}(s) \leq f(s) + \epsilon'$. Therefore

$$\max_{s \in U} (f(s) - \epsilon') \leq \max_{s \in U} \hat{f}(s) \leq \max_{s \in U} (f(s) + \epsilon').$$

Thus we have

$$\max_{s \in U} f(s) - \epsilon' \leq g \leq \max_{s \in U} f(s) + \epsilon'.$$

□

With the above Lemma 9 and Lemma 6, and by taking the union bound, we have that with probability at least $1 - \delta$, the two events in Lemma 9 and Lemma 6 both hold during the `LG`. Our next step is to show that if both of the events occur during `LG`, the approximation guarantees and sample complexity of Theorem 3 hold. To this end, we need the following Lemma 10.

Lemma 10. *Assume the events defined in Lemma 9 and Lemma 6 above hold during `LG`. Then for any element s that is added to the solution set S, the following statement holds.*

$$\Delta f(S, s) \geq \frac{1 - \alpha}{\kappa} (f(OPT) - f(S)) - 2\epsilon'.$$

Proof. At the first round in the for loop from Line 10 to Line 26 in Algorithm 2, if an element s is added to the solution set, it holds by Lemma 9 that $\Delta f(S, s) \geq w - \epsilon'$. Since at the first round $w = g$ and $g \geq \max_{s \in U} f(s) - \epsilon'$. It follows that $\Delta f(S, s) \geq \max_{s \in U} f(s) - 2\epsilon'$. By submodularity we have that $\kappa \max_{s \in U} f(s) \geq f(OPT)$. Therefore, $\Delta f(S, s) \geq \frac{f(OPT) - f(S)}{\kappa} - 2\epsilon'$.

At the round i in the for loop from Line 10 to Line 26 where $i > 1$, if an element $o \in OPT$ is not added to the solution set, then it is not added to the solution at the last iteration, where the threshold is $\frac{w}{1-\alpha}$. By Lemma 4, we have $\Delta f(S, o) \leq \frac{w}{1-\alpha} + \epsilon'$. Since for any element s that is added to the solution at iteration i, by Lemma 4 it holds that $\Delta f(S, s) \geq w - \epsilon'$. Therefore, we have

$$\begin{aligned}\Delta f(S, s) &\geq w - \epsilon' \\ &\geq (1-\alpha)(\Delta f(S, o) - \epsilon') - \epsilon' \\ &\geq (1-\alpha)\Delta f(S, o) - 2\epsilon'.\end{aligned}$$

By submodularity, it holds that $\Delta f(S, s) \geq (1-\alpha)\frac{f(OPT)-f(S)}{\kappa} - 2\epsilon'$. □

We now prove Theorem 3, which relies on the previous Lemma 9, 6 and 10.

Proof. The events defined in Lemma 6, 9 hold with probability at least $1-\delta$ by combining Lemma 6, 9, and taking the union bound. Therefore in order to prove Theorem 3, we assume that both the two events have occurred. The proof of the first result in the theorem depends on the Lemma 10. First, consider the case where the output solution set satisfies $|S| = \kappa$. Denote the solution set S after the i-th element is added as S_i. Then by Lemma 10, we have

$$f(S_{i+1}) \geq \frac{1-\alpha}{\kappa} f(OPT) + (1 - \frac{1-\alpha}{\kappa}) f(S_i) - 2\epsilon'.$$

By induction, we have that

$$\begin{aligned}f(S_\kappa) &\geq (1 - (1 - \frac{1-\alpha}{\kappa})^\kappa)\{f(OPT) - \frac{2\kappa\epsilon'}{1-\alpha}\} \\ &\geq (1 - e^{-1+\alpha})\{f(OPT) - \frac{2\kappa\epsilon'}{1-\alpha}\} \\ &\geq (1 - e^{-1} - \alpha)\{f(OPT) - \frac{2\epsilon}{1-\alpha}\} \\ &\geq (1 - e^{-1} - \alpha) f(OPT) - 2\epsilon.\end{aligned}$$

If the size of the output solution set S is smaller than κ, then any element $o \in OPT$ that is not added to S at the last iteration satisfies that $\Delta f(S, o) \leq w + \epsilon'$. Since the threshold w in the last iteration satisfies that $w \leq \frac{\alpha d}{\kappa}$, we have

$$\Delta f(S, o) \leq \frac{\alpha d}{\kappa} + \epsilon'.$$

It follows that

$$\begin{aligned}\sum_{o \in OPT \setminus S} \Delta f(S, o) &\leq \alpha(\max_{s \in S} f(s) + \epsilon') + \kappa\epsilon' \\ &\leq \alpha f(OPT) + 2\kappa\epsilon'.\end{aligned}$$

By submodularity and monotonicity of f, we have $f(S) \geq (1-\alpha) f(OPT) - 2\kappa\epsilon' = (1-\alpha) f(OPT) - 2\epsilon$.

□

B.3 Discussion on Theorem 3

We now present additional details of the discussion on Theorem 3.

First of all, the guarantee on the approximation ratio is $f(S) \geq (1 - e^{-1} - \alpha) f(OPT) - 2\epsilon$. By setting $\alpha = \epsilon$, the approximation ratio is $f(S) \geq (1 - e^{-1}) f(OPT) - (2 + f(OPT))\epsilon$. For applications like movie recommendations where $f(\mathrm{OPT}) \leq 1$, we obtain $f(S) \geq (1 - e^{-1}) f(\mathrm{OPT}) - 3\epsilon$.

Second, we illustrate the result of sample complexity in Theorem 3. Here the problem complexity term $H_i^{(m)}$ is the analogous term as $H_{l',a}^{(l)}$ in the result in Theorem 2 for Algorithm `LG`. The key difference is that $H_i^{(m)}$ is dependent on the gap of the marginal gain, which is denoted as $\Delta_{l,i}$, between the optimal element (with the highest marginal gain) and the suboptimal element while $H^{(l)}_{l',a}$ in the result of our algorithm depends on the difference between the evaluated marginal gain and the threshold value w ($|w - \mathbf{\Delta F}(S_m, a_m)^T \mathbf{w}|$).

Next, we consider the case where we don't utilize the samples of the marginal gains from the past rounds by setting $N_i^{(m)} = 0$. Here the term $N_i^{(m)}$ reflects the advantage of reusing past samples. If we lower bound this term to be 0, then the sample complexity is upper bounded by $4H_\epsilon^{(m)} R^2(2\log\frac{2}{\delta} + d\log M^{(m)}) + m$, where $H_\epsilon^{(m)} = \sum_{i\in[m]} H_i^{(m)}$. Compared with this case, our sample complexity of using past data can be improved by at most $N_i^{(m)}$ from the past rounds for each l', a while only sacrificing the factor of $O(\log(N^{(m)}))$.

The term $H_\epsilon^{(m)}$ satisfies $H_\epsilon^{(m)} = \sum_{i\in[m]} H_i^{(m)} = \sum_{i\in[m]} \frac{p_i^* \rho^{(m)}}{\max(\frac{\epsilon/\kappa + |w - \mathbf{\Delta F}(S,a)^T\mathbf{w}|}{2}, \epsilon/\kappa)^2}$. Since $\sum_{i\in[m]} p_i^* = 1$, $H_\epsilon^{(m)} = \frac{\rho^{(m)}}{\max(\frac{\epsilon/\kappa + |w - \mathbf{\Delta F}(S,a)^T\mathbf{w}|}{2}, \epsilon/\kappa)^2} \le \frac{\kappa^2\rho^{(m)}}{\epsilon^2}$. Therefore, the sample complexity for evaluating a single marginal gain is upper bounded by $\frac{4\kappa^2\rho^{(m)}R^2}{\epsilon^2}(2\log\frac{2}{\delta} + d\log M^{(m)}) + m$. In the case of the movie recommendation example, $R = 1$, the sample complexity for evaluating a single marginal gain is thus $\frac{4\kappa^2\rho^{(m)}}{\epsilon^2}(2\log\frac{2}{\delta} + d\log M^{(m)}) + m$.

Next, if the set of all the evaluated marginal gain vectors during the execution of `LinTG` is $\{\mathbf{\Delta F}(S_i, a_i)\}_{i\in[nh(\alpha)]} = (\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_{nh(\alpha)})$, then our problem is reduced to the general bandit case. Here $h(\alpha) = \frac{\log(\kappa/\alpha)}{\alpha}$ is the total number of rounds in `LinTG`. In this case, the allocation strategy would be always choosing the current marginal gain to sample and never sample marginal gain vectors evaluated in the past, which follows the same strategy as the CTG algorithm in Chen et al. (2023). Since Chen et al. (2023) also studies the same setting as us, with the difference that they don't exploit the linear structure of the submodular objective, we can compare our result with the theoretical result of CTG algorithm in Chen et al. (2023). In this case, the sample complexity of `LinTG` becomes $8H_\epsilon^{(m)} R^2 \log\frac{2}{\delta} + O(H_\epsilon^{(m)} R^2 d \log\frac{H_\epsilon^{(m)} RLm}{\lambda}\log\frac{1}{\delta})$ where $H_\epsilon^{(m)} = \frac{1}{\max\{\frac{\epsilon/\kappa + |w - \Delta f(S_m, a_m)|}{2}, \epsilon/\kappa\}^2}$. Since

$$\frac{\epsilon/\kappa + |w - \Delta f(S_m, a_m)|}{2} \le \max\{\frac{\epsilon/\kappa + |w - \Delta f(S_m, a_m)|}{2}, \epsilon/\kappa\} \le \epsilon/\kappa + |w - \Delta f(S_m, a_m)|,$$

the term $\phi(S, u)$ in the result of CTG in Theorem 3 in Chen et al. (2023) satisfies that $\frac{1}{\phi(S,u)^2} \le H_\epsilon^{(m)} \le \frac{4}{\phi(S,u)^2}$. Therefore, the first term in the sample complexity, which is $8H_\epsilon^{(m)} R^2 \log\frac{2}{\delta}$, matches the sample complexity result under the general bandit setting in Chen et al. (2023) up to a log-factor. The last term $O(H_\epsilon^{(m)} R^2 d \log\frac{H_\epsilon^{(m)} RLm}{\lambda}\log\frac{1}{\delta})$ can be seen as resulting from the linear structure.

C APPENDIX FOR SECTION 5

In this section, we present supplementary material to Section 5. In particular, we present the missing details about the experimental setup in Section C.1, and the additional experimental results in Section C.2.

C.1 Experimental setup

First of all, we provide more details about the movie recommendation dataset MovieLens 25M dataset Harper and Konstan (2015) used to evaluate the algorithms proposed in the main paper. The dataset contains ratings on a 5-star scale of $X = 13,816$ movies by the users. Each movie x is associated with $1,128$ topics and a relevance score is assigned to each movie and each topic. Here we use $G(x,i)$ to denote the relevance score of movie x and the topic i, which is defined in the dataset. In addition, we use $R(a,x)$ to denote the rating of user a for the movie x. Since the number of topics is too large for our experiments, we select a small subset of topics by the following procedure. To extract the most relevant topics, we conduct correlation analysis to select a subset of topics. For each pair of features i and j, if $Cor(i,j) \geq 0.4$ then either i or j is removed from the tag list. Subsequently, we remove any features that are negatively or weakly correlated to the ratings, i.e., $\forall i$ if $Cor(i,s) \leq 0.2$ then i is removed from the list. In this way, we select 79 tags. Then we randomly select different number of topics to use as the features.

In the experiments, the submodular basis functions $\{\mathbf{F}_i\}_{i\in[d]}$ is defined by the probabilistic coverage model. Next, we present the definition of the probabilistic coverage model, which captures how well a topic is covered by a subset of items Hiranandani et al. (2020); Yu et al. (2016). The problem definition is as follows.

Definition 2. *(probabilistic coverage model) Suppose there are a total of n elements denoted as U. For each element x and each topic i, let $G(x,i)$ be the relevance score that quantifies how relevant the movie x is to the topic i. For any tag i and subset of elements S, function $F_i(S)$ is defined to be*

$$F_i(S) = 1 - \prod_{x\in S}(1 - G(x,i)).$$

Therefore, the marginal gain of adding a new movie x to a subset S is defined as

$$\Delta F_i(S,x) = G(x,i)\prod_{x'\in S}(1 - G(x',i)), \qquad \forall x \notin S.$$

For each user, we calculate the preference of the user for different topics as follows.

$$w(a,i) = \frac{\sum_{x\in X} R(a,x)G(x,i)}{\sum_{x\in X, i'\in[d]} R(a,x)G(x,i')},$$

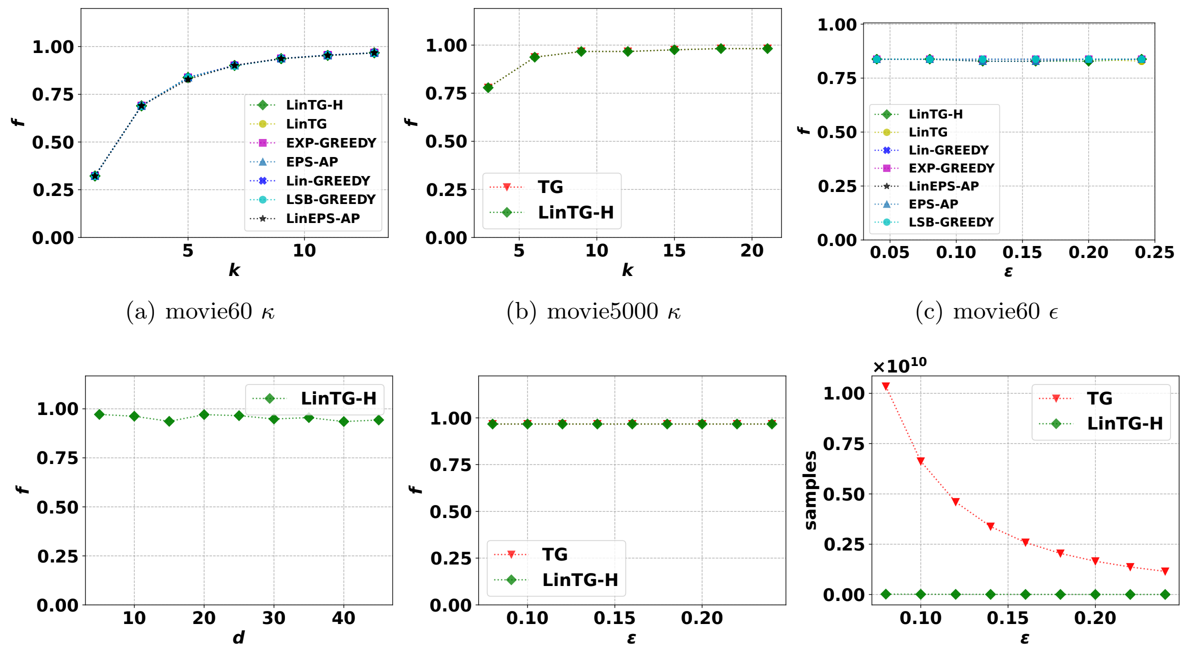
where X is the set of movies.

In addition to the MovieLens 25M dataset, we also generate a synthetic dataset comprising 60 elements and $d = 5$. This dataset was generated with a structure similar to the movie60 dataset. Specifically, the relevance scores between an item x and a topic i, denoted as $G(x,i)$, were sampled from a Gaussian distribution with a mean of 0.08 and a variance of 0.01, truncated to the range $[0,1]$.

Next, we describe more details about the algorithms that we compare to in the experiments. The algorithm EXP-GREEDY is proposed in Singla et al. (2016) and is based on the standard greedy algorithm with the element selecting procedure in each greedy round replaced by the CLUCB algorithm for best arm identification for general bandit.

The TG algorithm that we compare to is proposed in Chen et al. (2023) and is based on the threshold greedy algorithm. TG determines whether to add an element to the solution set or not by repeatedly sampling the evaluated marginal gain to identify whether the marginal gain is above the threshold or not. While TG achieves the same approximation ratio as our algorithms, it doesn't utilize the linear structure either. As a result, both EXP-GREEDY and TG can't be used to reuse previous samples.

The LSB-GREEDY algorithm is proposed in Yue and Guestrin (2011), which is designed for the regret-minimization algorithms that leverage linear structure. Since regret-minimization algorithms are designed for a different purpose, we adapt their method to align with our objective as follows:



(a) movie60 κ (b) movie5000 κ (c) movie60 ϵ

(d) movie5000 d (e) movie5000 ϵ (f) movie5000 ϵ

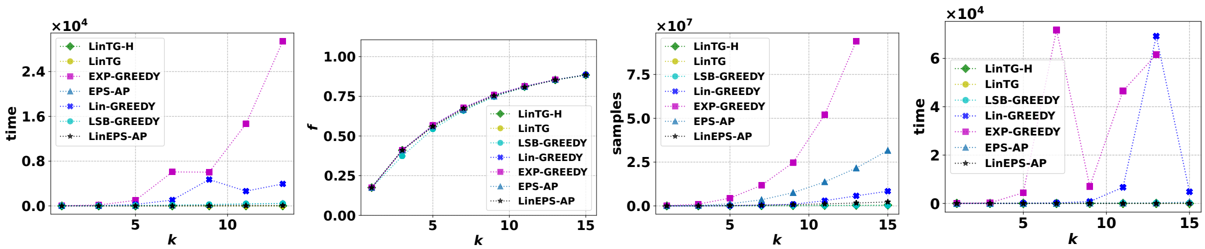
Figure 2: The experimental results of running the algorithms on instances of movie recommendation on the subsets of MovieLens 25M dataset with $n = 60$, $d = 5$ (”movie60”) and $n = 5000$, $d = 30$ (”movie5000”), and different datasets with different value of d.

1. **Final Solution Set:** The regret-minimization algorithm does not directly output a final solution set. To address this, we use the subset sampled at the last step of the algorithm as our final solution set.

2. **Fixed Horizon Setting:** The regret minimization algorithm operate in a fixed horizon setting, where the total number of steps is determined by a given input parameter T, and the number of total noisy samples would be $T\kappa$ (For each time step, there are κ noisy marginal gains samples collected.) To make a fair comparison, we adjust the time horizon of the regret-minimization setting to match the total sample complexity of our linear threshold greedy algorithm divided by κ. This ensures that the sample complexity is the same for both algorithms, allowing for a direct comparison of the output solution quality.

C.2 Exprimental results

The additional results of the experiments of the MovieLens 25M dataset are presented in Figure 2 and 3(a). From the results, we can see that the exact function values of different algorithms are almost the same. On the ”movie5000” dataset, the sample complexity of our proposed method LinTG-H is much smaller than TG, which demonstrates the sample efficiency of our algorithm. Additional results on the synthetic dataset in terms of f, sample queries and runtime are in Figure 3. From the results, we can see that the query complexity of our proposed LinTG and LinTG-H outperforms the other algorithms in terms of sample complexity, and achieves a better function value compared with the regret-minimization algorithm LSB-GREEDY.



(a) movie60 runtime (b) synthetic f (c) synthetic queries (d) synthetic time

Figure 3: The experimental results of running the algorithms on instances of movie recommendation on the subsets of MovieLens 25M dataset with $n = 60$, $d = 5$ ("movie60") and the synthetic dataset with $n = 60$, $d = 5$ ("synthetic") in terms of function value f, the number of noisy queries and runtime.

D TECHNICAL LEMMAS AND DETAILS

D.1 Matrix Computations and Runtime

In this section, we present the missing discussions on the implementation and runtime of the matrix inverse and multiplications in the main paper.

Inverse of matrix. Notice that all of the subroutine algorithms presented in the main paper keep track of $\mathbf{A}_t^{-1}$ throughout the algorithm. From Lemma 11, we can see that the inverse of $(\mathbf{A}_t + \mathbf{x}\mathbf{x}^T)$ can be computed as

$$(\mathbf{A}_t + \mathbf{x}\mathbf{x}^T)^{-1} = \mathbf{A}_t^{-1} - \frac{(\mathbf{A}_t^{-1}\mathbf{x})(\mathbf{A}_t^{-1}\mathbf{x})^T}{1 + \mathbf{x}^T\mathbf{A}_t^{-1}\mathbf{x}},$$

for any $\mathbf{x} \in U$. This step takes $O(d^2)$ in time complexity.

D.2 Technical Lemmas

Proposition 5. *(Proposition 1 in Soare et al. (2014)). Suppose the feature vector set is denoted as $\mathcal{X}$ with $|\mathcal{X}| = n$. Let $\hat{\boldsymbol{w}}_t$ be the solution to the least-squares problem and let $\boldsymbol{A}_t = \boldsymbol{X}_t^T \boldsymbol{X}_t$, and assume the noise variable ξ_t is R-sub-Gaussian. Assume $\mathcal{Y}$ is a subset of $\mathbb{R}^d$ with that $|\mathcal{Y}| = Y$. Then for all $\boldsymbol{y} \in \mathcal{Y}$ and any fixed sequence $\boldsymbol{x}_t$, we have that with probability at least $1-\delta$, it holds that*

$$|\boldsymbol{y}^T(\hat{\boldsymbol{w}}_t - \boldsymbol{w})| \leq R||\boldsymbol{y}||_{(\boldsymbol{A}_t)^{-1}}\sqrt{2\log(\frac{\pi^2 t^2 Y}{3\delta})}$$

for all $t > 0$.

Proposition 1. *Let $\hat{\boldsymbol{w}}_t^\lambda$ be the solution to the regularized least-squares problem with regularizer λ and let $\boldsymbol{A}_t^\lambda = \boldsymbol{X}_t^T \boldsymbol{X}_t + \lambda \boldsymbol{I}$. Then for every adaptive sequence $\boldsymbol{X}_t$ such that at any step t, $\boldsymbol{x}_{a_t}$ only depends on past history, we have that with probability at least $1-\delta$, it holds that for all $t \geq 0$ and all $\boldsymbol{y}_t \in \mathbb{R}^d$ that only depends on past history up to time t,*

$$|\boldsymbol{y}_t^T(\hat{\boldsymbol{w}}_t^\lambda - \boldsymbol{w})| \leq ||\boldsymbol{y}_t||_{(\boldsymbol{A}_t^\lambda)^{-1}} C_t,$$

where C_t is defined as

$$C_t = R\sqrt{2\log\frac{\det(\boldsymbol{A}_t^\lambda)^{\frac{1}{2}}\det(\lambda I)^{-\frac{1}{2}}}{\delta}} + \lambda^{\frac{1}{2}} Q.$$

Moreover, if $||\boldsymbol{x}_{a_t}|| \leq L$ holds for all $t > 0$, then

$$C_t \leq R\sqrt{d\log(\frac{1 + tL^2/\lambda}{\delta})} + \lambda^{\frac{1}{2}} Q.$$

Proof. Let us denote the history up to time t as $\mathcal{H}_t$, then by Cauchy–Schwarz inequality, we have that for each fixed time t

$$P(|\mathbf{y}_t^T(\hat{\mathbf{w}}_t^\lambda - \mathbf{w})| \leq ||\mathbf{y}||_{(\mathbf{A}_t^\lambda)^{-1}} C_t, \forall \mathbf{y}_t \text{ depend on } \mathcal{H}_t | \mathcal{H}_t) \geq P(||\hat{\mathbf{w}}_t^\lambda - \mathbf{w}||_{(\mathbf{A}_t^\lambda)^{-1}} \leq C_t | \mathcal{H}_t)$$

It then follows that

$$P(|\mathbf{y}_t^T(\hat{\mathbf{w}}_t^\lambda - \mathbf{w})| \leq ||\mathbf{y}||_{(\mathbf{A}_t^\lambda)^{-1}} C_t, \forall \mathbf{y}_t \text{ depend on } \mathcal{H}_t) \geq P(||\hat{\mathbf{w}}_t^\lambda - \mathbf{w}||_{(\mathbf{A}_t^\lambda)^{-1}} \leq C_t).$$

From Theorem 2 in Abbasi-Yadkori et al. (2011), we can see that with probability at least $1-\delta$, at any step t, $||\hat{\mathbf{w}}_t^\lambda - \mathbf{w}||_{(\mathbf{A}_t^\lambda)^{-1}} \leq C_t$. Therefore, by applying this result and taking a union bound over all time step t, we can prove the result in the proposition. □

Lemma 11. *(Woodbury formula.) The inverse of matrix $A + UCV$ can be computed as*

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}.$$

Lemma 12. *(Abbasi-Yadkori et al. (2011), Lemma 10) Let the maximum l_2-norm of the feature vectors be denoted as L. The determinant of matrix $\boldsymbol{A}_t$ is bounded as*

$$det(\boldsymbol{A}_t) \leq (\lambda + tL^2/d)^d$$

Lemma 13. *For any vector $\boldsymbol{x}_1, \boldsymbol{x}_2 ... \boldsymbol{x}_m \in \mathbb{R}^d$ and $\boldsymbol{y} \in \mathbb{R}^d$ and , if we set p_a^* to be*

$$p_a^* = \frac{|w_a^*|}{\sum_{a'=1}^m |w_{a'}^*|},$$

where $w_{a'}^*$ is the optimal solution of the linear program in (5) below

$$\arg\min_{\{w_a\}} \sum_{a=1}^m |w_a| \qquad s.t. \qquad \boldsymbol{y} = \sum_{a=1}^m w_a \boldsymbol{x}_a. \tag{19}$$

Then p_a^* is also the asymptotic optimal solution of the optimization problem below when $n \to \infty$.

$$\begin{aligned} &\arg\min_{\{p_a\}} \ \boldsymbol{y}^T (\sum_{a\in[m]} p_a \boldsymbol{x}_a \boldsymbol{x}_a^T + \frac{\lambda}{n} I)^{-1} \boldsymbol{y} \\ &s.t. \ \sum_{a\in[m]} p_a = 1. \end{aligned}$$

Moreover, we have that

$$\sqrt{\boldsymbol{y}^T (\sum_{a\in[m]} p_a^* \boldsymbol{x}_a \boldsymbol{x}_a^T)^{-1} \boldsymbol{y}} \leq \rho,$$

where ρ is the optimal value of the optimization problem in (19).

Proof. The proof can be found in Appendix B in Xu et al. (2018), so we omit the proof here. □

Lemma 14. *Suppose $x \in \mathbb{R}$ and $x \geq 2$, if we have $x \geq \frac{2}{a} \log \frac{2}{a}$, then it holds that*

$$\frac{\log x}{x} \leq a$$

Proof. Since $y = \frac{\log x}{x}$ is decreasing when $x \geq 2$, if $x > \frac{2}{a} \log \frac{2}{a}$, then we have

$$\frac{\log x}{x} < \frac{a}{2} \cdot \frac{\log(\frac{2}{a} \log \frac{2}{a})}{\log \frac{2}{a}} \leq a.$$

□

Lemma 15. *Suppose $x, a, b \in \mathbb{R}_+$, if we have $x \leq a + b \log x$, then it holds that*

$$x \leq 2a + 2b \log b.$$