Efficient Learning in Large-Scale Combinatorial Semi-Bandits

Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Branislav Kveton
Technicolor Labs
Los Altos, CA
{zheng.wen, azin.ashkan, hoda.eydgahi, branislav.kveton}@technicolor.com

Abstract

In this paper, we consider efficient learning in large-scale combinatorial semi-bandits with linear generalization, and as a solution, propose a novel learning algorithm called Randomized Combinatorial Maximization (RCM). RCM is motivated by Thompson sampling, and is computationally efficient as long as the offline version of the combinatorial problem can be solved efficiently. We establish that RCM is provably statistically efficient in the coherent Gaussian case, by developing a Bayes regret bound that is independent of the problem scale (number of items) and sublinear in time. We also evaluate RCM on a variety of real-world problems with thousands of items. Our experimental results demonstrate that RCM learns two orders of magnitude faster than the best baseline.

1 Introduction

Combinatorial optimization is a mature field [Papadimitriou and Steiglitz, 1998], which has countless practical applications. One of the most studied problems in combinatorial optimization is maximization of a modular function subject to combinatorial constraints. Many important problems; such as minimum spanning tree, shortest path, and maximum-weight bipartite matching; can be viewed as instances of this problem.

In practice, the optimized modular function is often unknown and needs to be learned while repeatedly solving the problem. This class of learning problems was recently formulated as a combinatorial bandit / semi-bandit, depending on the feedback model [Audibert et al., 2014]. Since then, many combinatorial bandit algorithms have been proposed: for the stochastic setting [Gai et al., 2012, Chen et al., 2013, Russo and Van Roy, 2014]; for the adversarial setting [Cesa-Bianchi and Lugosi, 2012, Audibert et al., 2014, Neu and Bartk, 2013]; and for subclasses of combinatorial problems, matroid and polymatroid bandits [Kveton et al., 2014a,b], submodular maximization [Wen et al., 2013, Gabillon et al., 2013, 2014], which can be solved extremely efficiently. Many regret bounds have been established for the combinatorial semi-bandit algorithms. To achieve an $O(\sqrt{n})$ dependence on time $n$, all of the regret bounds are $\Omega(\sqrt{L})$, where $L$ is the number of items. Therefore, the algorithms are impractical when the number of items $L$ is huge. The dependence on $L$ is intrinsic because the algorithms estimate the weight of each item separately, and matching lower bounds have been established (Section 3.2).

In many problems, items have features and their weights are similar when the features are similar. In movie recommendation, for instance, the expected ratings of the movies that are close in the latent space are similar. In this work, we show how to leverage this structure to learn the model of our problem more efficiently. More specifically, we assume a linear generalization across the items: conditioned on the features of the item,
the expected weight of the item can be estimated using a linear model. Our goal is to develop more efficient learning algorithms for combinatorial semi-bandits by exploiting linear generalization.

We make three major contributions. First, we propose a learning algorithm for combinatorial semi-bandits with linear generalization. Our proposed algorithm is motivated by Thompson sampling and we refer to it as Randomized Combinatorial Maximization (RCM). RCM is computationally efficient, as long as the offline version of the combinatorial problem can be solved efficiently. Second, we establish two bounds on the Bayes cumulative regret of RCM when it is applied to the coherent Gaussian case. The first bound is $O(\sqrt{\ln L})$ and sublinear in time, while the second bound is an $L$-independent sublinear bound. Finally, we evaluate RCM on a variety of real-world problems with thousands of items. Our experimental results demonstrate that RCM learns two orders of magnitude faster than the best baseline.

2 Combinatorial Optimization

We focus on a class of combinatorial optimization problems that aim to find a maximum-weight set from a given family of sets. Specifically, one such combinatorial optimization problem can be represented as a triple $(E, A, w)$, where (1) $E = \{1, \ldots, L\}$ is a set of $L$ items, called the ground set, (2) $A \subseteq \{A \subseteq E : |A| \leq K\}$ is a family of subsets of $E$ with up to $K$ items, where $K \leq L$, and (3) $w : E \rightarrow \mathbb{R}$ is a weight function that assigns each item $e$ in the ground set $E$ a real number. The total weight of all items in a set $A \subseteq E$ is defined as:

$$f(A, w) = \sum_{e \in A} w(e),$$

which is a linear functional of $w$ and a modular function in $A$. A set $A^{\text{opt}}$ is a maximum-weight set in $A$ if

$$A^{\text{opt}} \in \arg\max_{A \in A} f(A, w) = \arg\max_{A \in A} \sum_{e \in A} w(e).$$

Many classical combinatorial optimization problems, such as finding a maximum-weight basis in a matroid (e.g. finding a minimum spanning tree (MST)), finding a maximum-weight set in the intersection of two matroids (e.g. bipartite matching), the shortest path problem and the traveling salesman problem (TSP), have form (2). Though some of these combinatorial optimization problems can be solved efficiently (e.g. finding a MST and bipartite matching), other problems (e.g. TSP) are known to be NP-hard. However, for many such NP-hard combinatorial problems, there exist computationally efficient approximation algorithms and/or randomized algorithms that achieve near-optimal solutions with high probability. Similarly as [Chen et al., 2013], in this paper, we allow the agent to use any possibly approximation/randomized algorithm ORACLE to solve (2), and denote its solution as $A^{*} \leftarrow \text{ORACLE}(E, A, w)$. To distinguish from a learning algorithm, we refer to a combinatorial optimization algorithm as an oracle in this paper.

3 Combinatorial Semi-Bandits with Linear Generalization

Many real-world problems are combinatorial in nature. In recommender systems, for instance, the user is typically recommended $K$ items out of $L$. The value of the item, such as the expected rating of a movie with respect to all users, is never known perfectly and has to refined while solving the problem, recommending repeatedly to the pool of the users. Recommender problems are known to be highly structured. In particular, it is well known that the user-item matrix is typically low-rank [Koren et al., 2009] and that the
value of the items can be written as a linear combination of their position in the latent space. In this work, we propose a learning algorithm for combinatorial optimization that leverages this structure. In particular, we assume that the weight of each item is a linear function of its features and then we learn the parameters of this model, jointly for all items.

3.1 Combinatorial Semi-Bandits

We formalize our learning problem as a combinatorial semi-bandit. A combinatorial semi-bandit is a triple \((E, A, P)\), where \(E\) and \(A\) are defined in Section 2 and \(P\) is a probability distribution over the weights \(w \in \mathbb{R}^L\) of the items in the ground set \(E\). We assume that the weights \(w\) are drawn i.i.d. from \(P\). The mean weight is denoted by \(\bar{w} = \mathbb{E}[w]\).

Each item \(e\) is associated with an arm and we assume that multiple arms can be pulled. A subset of arms \(A \subseteq E\) can be pulled if and only if \(A \in A\). The return of pulling arms \(A\) is \(f(A, w)\) (Equation (1)), the sum of the weights of all items in \(A\). After the arms \(A\) are pulled, we observe the individual return of each arm, \(\{w(e) : e \in A\}\). This feedback model is known as semi-bandit [Audibert et al., 2004].

We assume that the combinatorial structure \((E, A)\) is known and the distribution \(P\) is unknown. We would like to stress that we do not make any structural assumptions on \(P\). The optimal solution to our problem is a maximum-weight set in expectation:

\[
A_{\text{opt}} \in \arg \max_{A \in A} \mathbb{E}[f(A, w)] = \arg \max_{A \in A} \sum_{e \in A} \bar{w}(e). \tag{3}
\]

This objective is equivalent to the one in Equation (2).

Our learning problem is episodic. In episode \(t\), we choose a set \(A_t\) according to some possibly episode-dependent policy, and gain \(f(A_t, w_t)\), where \(w_t\) is the realization of the stochastic weight in episode \(t\). Our goal is to design a policy that maximizes the expected cumulative return in \(n\)-episodes \(\mathbb{E}[\sum_{t=1}^{n} f(A_t, w_t)]\), where the expectation is over (1) the random weights \(w_t\)'s, (2) possible randomization in the designed policy, and (3) \(\bar{w}\) if it is randomly generated. Notice that the choice of \(A_t\) impacts both the return and observations in episode \(t\). So we need to trade off exploration and exploitation, similarly to other bandit problems.

3.2 Linear Generalization

As we have discussed in Section 1, many provably efficient algorithms have been developed for various combinatorial semi-bandits of form (3) (see Chen et al. [2013], Gai et al. [2012], Kveton et al. [2014a], Russo and Van Roy [2014]). However, since there are \(L\) parameters to learn and these algorithms do not consider generalization across items, the derived upper bounds on the expected cumulative regret and/or the Bayes cumulative regret of these algorithms are at least \(O(\sqrt{L})\). Furthermore, Audibert et al. [2004] has derived an \(\Omega(\sqrt{LKn})\) lower bound on adversarial combinatorial semi-bandits, while Kveton et al. [2014a] has derived an asymptotic \(\Omega(L \log(n)/\Delta)\) gap-dependent lower bound on stochastic combinatorial semi-bandits, where \(\Delta\) is the “gap”.

However, in many modern combinatorial semi-bandit problems, \(L\) tends to be enormous. Thus, an \(O(\sqrt{L})\) regret is unacceptably large in these problems. On the other hand, in many practical problems, there exists a generalization model based on which the weight of one item can be (approximately) inferred based on the weights of other items. By exploiting such generalization models, an \(o(\sqrt{L})\) or even an \(L\)-independent cumulative regret might be achieved.

In this paper, we assume that there is a (possibly imperfect) linear generalization model across the items. Specifically, we assume that the agent knows a generalization
Algorithm 1 Randomized Combinatorial Maximization (RCM)

Input: Combinatorial structure \((E,A)\), generalization matrix \(\Phi \in \mathbb{R}^{L \times d}\), tuning parameters \(\lambda, \sigma > 0\), oracle \textsc{Oracle}

// Initialization
Set \(\Sigma_1 \leftarrow \lambda^2 I \in \mathbb{R}^{d \times d}\) and \(\bar{\theta}_1 = 0 \in \mathbb{R}^d\)

for all \(t = 1,2,\ldots,n\) do

Sample \(\theta_t \sim N(\bar{\theta}_t, \Sigma_t)\)

Compute \(A_t \leftarrow \textsc{Oracle}(E,A,\Phi \theta_t)\)

Choose set \(A_t\), and observe \(w_t(e), \forall e \in A_t\)

// Learning
for \(k = 1,\ldots,|A_t|\) do

Update

\[
\bar{\theta}_{t+1} \leftarrow I - \frac{\Sigma_t \phi_{a^t_k} \phi_{a^t_k}^T}{\phi_{a^t_k}^T \Sigma_t \phi_{a^t_k} + \sigma^2} \bar{\theta}_t + \frac{\Sigma_t \phi_{a^t_k}}{\phi_{a^t_k}^T \Sigma_t \phi_{a^t_k} + \sigma^2} w_t(a^t_k)
\]

\[
\Sigma_{t+1} \leftarrow \Sigma_t - \frac{\Sigma_t \phi_{a^t_k} \phi_{a^t_k}^T \Sigma_t}{\phi_{a^t_k}^T \Sigma_t \phi_{a^t_k} + \sigma^2}
\]

where \(a^t_k\) is the \(k\)th element in \(A_t\)

end for

end for

matrix \(\Phi \in \mathbb{R}^{L \times d}\) s.t. \(\bar{\theta}^*\) is “close” to the subspace span \([\Phi]\). We use \(\phi_e\) to denote the transpose the \(e\)-th row of \(\Phi\), and refer to it as the feature vector of item \(e\). Without loss of generality, we assume that \(\|\phi_e\|_2 \leq 1, \forall e \in E\).

4 Randomized Combinatorial Maximization

4.1 Algorithm

We propose a learning algorithm for combinatorial semi-bandits based on randomized reward exploration. More specifically, the algorithm chooses a maximum-weight set based on a randomly sampled coefficient vector \(\theta_t\) and a pre-specified oracle. Because of this, we refer to our algorithm as Randomized Combinatorial Maximization (RCM).

The pseudocode of our algorithm is given in Algorithm [1] where \((E,A)\) is the combinatorial structure, \(\Phi\) is the generalization matrix, \textsc{Oracle} is a combinatorial optimization algorithm, and \(\lambda\) and \(\sigma\) are two tuning parameters. Specifically, \(\lambda\) is a regularization parameter and smaller \(\lambda\) tends to make the sampled coefficient vector \(\theta_t\) closer to 0. On the other hand, \(\sigma\) controls the learning rate. Generally speaking, a large \(\sigma\) will lead to slow learning; however, if \(\lambda\) is mis-specified, a small \(\sigma\) will make the algorithm quickly converge to some sub-optimal coefficient vector.

In each episode \(t\), Algorithm [1] consists of three steps. First, it randomly samples a coefficient vector \(\theta_t\) from a Gaussian distribution. Second, it computes \(A_t\) based on \(\theta_t\) and the pre-specified oracle. Finally, it updates the mean vector \(\bar{\theta}\) and the covariance matrix \(\Sigma\) of the Gaussian distribution based on Kalman filtering.
4.2 RCM and Thompson Sampling

It is worth pointing out that RCM algorithm is motivated by Thompson sampling (Russo and Van Roy [2013, 2014], Thompson [1933]) and Gaussian approximation. Specifically, if (1) \( \bar{w} = \Phi \theta^* \), (2) the prior on \( \theta^* \) is \( N(0, \lambda^2 I) \), and (3) \( \forall (t, e) \), the noise \( \eta_t(e) = w_t(e) - \bar{w}(e) \) is independently sampled from \( N(0, \sigma^2) \), then the RCM algorithm with parameter \( (\Phi, \lambda, \sigma) \) reduces to the Thompson sampling algorithm. We henceforth refer to a case satisfying condition 1-3 as a coherent Gaussian case.

Obviously, the RCM algorithm can be applied to more general cases, even to cases with no prior and/or \( \bar{w} \notin \text{span}[\Phi] \). Though it is hard to derive theoretical results in such cases, however, as we will see in Section 5, the performance of RCM is very good in several real-world examples.

It is worth pointing out that though in theory Thompson sampling can be applied to any combinatorial semi-bandit with linear generalization, however, in practice, it suffers from two problems. First, in many practical problems, it is not clear how to choose the “right” prior, especially when \( \bar{w} \notin \text{span}[\Phi] \). Second, computing the posterior distributions can be intractable unless we choose a conjugate prior. On the other hand, RCM is computationally efficient as long as the specified oracle is computationally efficient, and only two parameters need to be tuned in this algorithm. Notice that sampling from Gaussian distributions is reasonable based on central limit theorem.

4.3 Performance Metrics

Let \( A^* \leftarrow \text{ORACLE}(E, A, \bar{w}) \). In this paper, we measure the performance loss with respect to \( A^* \). Specifically, we define \( R_t = f(A^*, \bar{w}_t) - f(A^t, \bar{w}_t) \) as the realized regret in episode \( t \). If the expected weight \( \bar{w} \) is fixed but unknown, we define the expected cumulative regret in \( n \) episodes as

\[
R(n) = \sum_{t=1}^{n} \mathbb{E}[R_t|\bar{w}],
\]

where the expectation is over random weights and possible randomization in the designed policy. If necessary, we denote \( R(n) \) as \( R(n; \bar{w}) \) to emphasize the dependence on \( \bar{w} \). On the other hand, if \( \bar{w} \) is randomly generated or the agent has a prior belief in \( \bar{w} \), then from [Russo and Van Roy, 2013], the Bayes cumulative regret in \( n \) episodes is defined as

\[
BR(n) = \mathbb{E}_{\bar{w}}[R(n; \bar{w})] = \sum_{t=1}^{n} \mathbb{E}[R_t],
\]

where the expectation is also over \( \bar{w} \). That is, \( BR(n) \) is a weighted average of \( R(n; \bar{w}) \) under the prior on \( \bar{w} \).

4.4 Upper Bound

Based on some recent work on Thompson sampling [Russo and Van Roy, 2013, 2014] and linear bandits [Dani et al., 2008], we can derive the following problem-independent upper bound on \( BR(n) \) when RCM is applied to the coherent Gaussian case with the right parameter. Recall that in this case, RCM reduces to a Thompson sampling algorithm.

**Theorem 1.** If \( \bar{w} = \Phi \theta^* \), the prior on \( \theta^* \) is \( N(0, \lambda^2 I) \), and the noises are i.i.d. sampled
from $N(0, \sigma^2)$, then under RCM algorithm with parameter $(\Phi, \lambda, \sigma)$, then we have

$$BR(n) \leq 1 + K\lambda \min \left\{ L \ln \left( \frac{\lambda L}{\sqrt{2\pi}} \right), \sqrt{d} \ln \left( \frac{2dKn\lambda}{\sqrt{2\pi}} \right) \right\} \sqrt{2dn \ln \left( 1 + \frac{nK\lambda^2}{d} \right)}. \quad (7)$$

Before outlining the proof of Theorem 1, we first briefly discuss the result. First, notice that if $\lambda \geq \sigma$, which almost always holds, then we have

$$BR(n) \leq 1 + K\lambda \min \left\{ L \ln \left( \frac{\lambda L}{\sqrt{2\pi}} \right), \sqrt{d} \ln \left( \frac{2dKn\lambda}{\sqrt{2\pi}} \right) \right\} \sqrt{2dn \log_2 \left( 1 + \frac{nK\lambda^2}{d} \right)}$$

which is a minimum of two bounds. The first bound is $L$-dependent, but it is only $O(\sqrt{\ln(L)})$; on the other hand, the second bound is $L$-independent, but is $O(d)$ instead of $O(\sqrt{d})$.

We briefly discuss the tightness of our bound. Without loss of generality, we assume that $\lambda = 1$. For the special case when $\Phi = I$ (i.e. no generalization), Russo and Van Roy [2013] provides an $O(\sqrt{LKn\ln(L/K\ln n)}$ upper bound on $BR(n)$ when Thompson sampling is applied, and Audibert et al. [2004] provides an $\Omega(\sqrt{LK\ln n})$ lower bound. Since $L = d$ when $\Phi = I$, the above results indicate that for general $\Phi$, the best upper bound one can hope is $O(\sqrt{Kn\ln n})$. Hence, our bound is at most $O(\sqrt{\ln \min \{\ln(L), d\}})$ larger. As is discussed in the appendix (see Remark 1), the extra $O(\sqrt{K})$ factor is due to linear generalization. It might be intrinsic, but we leave the final word and tightness analysis to future work.

### 4.5 Analysis

We now outline the proof of Theorem 1, which is based on Russo and Van Roy [2013, Dani et al. 2008]. Let $H_t$ denote the “history” (i.e. all the available information) by the start of episode $t$. Note that from the Bayesian perspective, conditioning on $H_t$, $\theta^*$ and $\theta_t$ are i.i.d. drawn from $N(\theta_t, \Sigma_t)$ (see Russo and Van Roy [2013]). This is because that conditioning on $H_t$, the posterior belief in $\theta^*$ is $N(\theta_t, \Sigma_t)$ and based on Algorithm 1 $\theta_t$ is independently sampled from $N(\theta_t, \Sigma_t)$. Since ORACLE is a fixed combinatorial optimization algorithm (even though it can be independently randomized), and $E, A, \Phi$ are all fixed, then conditioning on $H_t$, $A^*$ and $A^t$ are also i.i.d., furthermore, $A^*$ is conditionally independent of $\theta_t$, and $A^t$ is conditionally independent of $\theta^*$.

To simplify the exposition, $\forall \theta \in \mathbb{R}^d$ and $\forall A \subseteq E$, we define

$$g(A, \theta) = \sum_{e \in A} \langle \phi_e, \theta \rangle, \quad (9)$$

then we have $E[f(A^*, w_t) | H_t, \theta_t, \theta^*, A^*, A^t] = g(A^*, \theta^*)$ and $E[f(A^t, w_t) | H_t, \theta^*, \theta_t, A^*, A^t] = g(A^t, \theta^*)$, hence we have $E[R_t | H_t] = E[g(A^*, \theta^*) - g(A^t, \theta^*) | H_t]$. We also define the upper confidence bound (UCB) function $U_t : 2^E \to \mathbb{R}$ as

$$U_t(A) = \sum_{e \in A} \langle \phi_e, \theta_t \rangle + c \sqrt{\phi_t^2 \Sigma_t \phi_e}, \quad (10)$$

where $c > 0$ is a constant to be specified. Notice that conditioning on $H_t$, $U_t$ is a deterministic function and $A^*, A^t$ are i.i.d., then $E[R_t | H_t] = 0$ and

$$E[R_t | H_t] = E[g(A^*, \theta^*) - U_t(A^*) | H_t] + E[U_t(A^t) - g(A^t, \theta^*) | H_t]. \quad (11)$$
One key observation is that
\[
\mathbb{E}_t \left[ U_t(A^t) - g(A^t, \theta^*) \right] \leq c \sum_{e \in A^t} \sqrt{\phi_t^e \Sigma_t \phi_e},
\]
where (b) follows from the fact that $A^t$ and $\theta^*$ are conditionally independent, and (c) follows from $\mathbb{E}_t[\theta^* | H_t] = \theta_t$. Hence
\[
BR(n) = \sum_{t=1}^n \mathbb{E}_t [g(A^*, \theta^*) - U_t(A^*)] + c \sum_{t=1}^n \mathbb{E}_t \left[ \sum_{e \in A^t} \sqrt{\phi_t^e \Sigma_t \phi_e} \right].
\]
records corresponding to the top 800 popular queries in the sample are used so that we can perform matrix factorization in our experiment. There are about 378k unique users and 212k unique document URLs associated with these queries.

For each query associated to a user’s session, we extract click or no click events between the user and all documents listed on the result page. All such events of a query are put together and divided into training and test sets as follows: We first remove those events containing documents that appear only once. For each of the remaining documents, we place $\frac{2}{3}$ of events containing this document into the training set and the remaining $\frac{1}{3}$ of events into the test set. Next we explain how we perform the experiment for each query using the corresponding training and test data.

We define a matrix $M_{\text{train}}$ from the training data, where $M_{\text{train}}(i,j) = 1$ if user $i$ is shown document $j$ and clicks on it, $M_{\text{train}}(i,j) = -1$ if user $i$ is shown document $j$ and does not click on it, and $M_{\text{train}}(i,j) = 0$ if the entry is unobserved. We factor $M_{\text{train}}$ as $\hat{M}_{\text{train}} = U_{\text{train}}V_{\text{train}}^\top$ using matrix factorization [Keshavan et al., 2010], a standard approach to collaborative filtering. We repeat the same process in order to create and factor the test matrix $M_{\text{test}}$ as $\hat{M}_{\text{test}} = U_{\text{test}}V_{\text{test}}^\top$ based on the test data. In the rest of the experiment, the document matrix $V_{\text{train}}$ from the training data represents the features vector of documents, while the completed matrix $\hat{M}_{\text{test}}$ from the test data is used to build our model of user behavior over documents.

Our goal is to learn how to rank $K$ most relevant documents for a given user’s query. The ground set $E$ are all documents that appeared for this query and their features $\Phi$ are the item matrix $V_{\text{train}}$. Given query $q$ and document $j$, $\tilde{w}(j)$ is the relevance of document $j$ to this query. We estimate it as $\tilde{w}(j) = \frac{1}{n_q} \sum_{i=1}^{n_q} w_i(j)$, where $n_q$ is the number of users who issued $q$ and $w_i(j) = \hat{M}_{\text{test}}(i,j)$ is the relevance of document $j$ to user $i$, which is estimated by matrix factorization on the test data.

Our experiment is episodic. In each episode, a user is chosen at random from $\hat{M}_{\text{test}}$. Top $K = 10$ documents are ranked for this user, and their weight scores are updated based on the observation of corresponding $K$ entries from $\hat{M}_{\text{test}}$. The expected per-step-return in each episode is computed as the average of the expected per-step-return of the episode across all queries. Our learning results are shown in Figure 1 where we observe two major trends. First, the expected return of RCM approaches that of the optimal policy $A^*$ as the number of episodes increases. Second, RCM outperforms both UCB1 and Thompson sampling.

Since this experiment deals with learning to rank Web documents, we evaluate the results with respect to a well-known ranking metric in information retrieval, called nor-
malized discounted cumulative gain (nDCG) [Järvelin and Kekäläinen 2002]. Given a result list of K documents ranked for a query, document cumulative gain (DCG) measures the accumulated gain of documents from the top of the list to the bottom with the gain of each document \( d_r \) discounted at position \( r \) in the list:

\[
DCG = \sum_{r=1}^{K} \frac{w(d_r)}{\log(r+1)}
\]

The normalized \( DCG \) is a value in \([0, 1]\) computed as: \( nDCG = \frac{DCG}{IDCG} \) where \( IDCG \) is the ideal gain obtained from the optimal policy. We use the average of \( nDCG \) across all queries to compare the performance of each policy in terms of the ranked results they produce at the end of learning process for these queries. Figure 1-b reports \( nDCG \) for top 1, 5, and 10 results. As it can be seen, \( RCM \) outperforms the other two policies in terms of \( nDCG \) at different values of \( K \).

5.2 Learning Recommendation of Diverse Movies
In the second experiment, we evaluate \( RCM \) on the problem of recommending \( K \) diverse movies that are most likely to be chosen in expectation. We experiment with MovieLens dataset [Lam and Herlocker 2013], a dataset of 6k people and 4k movies with 1 million ratings. We cast our problem as learning a maximum-weight basis of a linear matroid in expectation [Kveton et al. 2014a]. This optimization problem can be solved greedily. A set of movies is considered independent if no genre vector of a movie in the set can be written as a linear combination of the genre vectors of the remaining movies. The total number of genres in our dataset is 18. Therefore, the rank of our matroid is \( K = 18 \) and so is the cardinality of any set of recommended movies. The weight \( \bar{w}(j) \) is the probability that movie \( j \) is chosen. We estimate it as \( \bar{w}(j) = \frac{1}{n_p} \sum_{i=1}^{n_p} w_i(j) \), where \( n_p \) is the number of people in our dataset and \( w_i(j) \) is an indicator that person \( i \) chooses movie \( j \). We sample \( w_i(j) \) from the Bernoulli distribution with mean \( 1/(1 + \exp(-(r_{i,j} - 3.5)/0.5)) \), where \( r_{i,j} \) is the rating that user \( i \) gives to movie \( j \).

Similarly to the previous experiment, we divide our dataset into training and test sets, and complete the rating matrix for each set separately. In each episode, we choose a random person from the test matrix, a binary vector \( w_i \). The features of the movies are the item matrix in the training set \( V_{\text{train}} \).

Our learning results are reported in Figure 2-a. We observe two major trends. First, the expected return of \( RCM \) approaches that of the optimal policy \( A^* \) as the number of episodes increases. Second, \( RCM \) outperforms both UCB1 and Thompson sampling. More specifically, \( RCM \) learns more than two orders of magnitude faster than Thompson sampling. For instance, the expected return of Thompson sampling after 10k episodes is 12.5. \( RCM \) reaches this point in mere 30 episodes. Twelve most popular movies from the optimal solution \( A^* \) are shown in Figure 2-b. These movies range from Comedy to War and Sci-fi, and are quite diverse. This validates our assumption that linear independence can be used to represent diversity.

6 Conclusions
This is the first work that proposes an algorithm for large-scale combinatorial semi-bandits. The key structural assumption is linear generalization: the expected weight of each item can be estimated using a linear model based on its features. Our goal is to learn to solve the large-scale combinatorial semi-bandits efficiently. Our learning algorithm, \( RCM \), is practical and we prove two upper bounds on its Bayes regret in the coherent Gaussian case. The algorithm is evaluated on two real-world problems and we demonstrate that it learns two orders of magnitude faster than the best baseline.

In future work, we would like to advance the analysis of \( RCM \) as well as its practical applications. In particular, we want to analyze \( RCM \) from the frequentist perspective and...
bound its regret $R(n)$, a stronger notion of the regret than the Bayes regret $BR(n)$ in our paper. Furthermore, we would like to analyze RCM under the assumption that the linear generalization model is imperfect. Finally, one limitation of our experiments is that they are partially simulated, we build the model of the users. We would like to apply RCM to a problem where RCM interacts directly with people and learns. Our empirical results indicate that RCM can learn high-quality policies fast. Therefore, the method is extremely practical.

References

Jean-Yves Audibert, Universit Paris Est, and Sebastien Bubeck. Minimax policies for adversarial and stochastic bandits, in. In Proceedings of the 22nd Annual Conference on Learning Theory, Omnipress, pages 773–818, 2004.

Jean-Yves Audibert, Sebastien Bubeck, and Gabor Lugosi. Regret in online combinatorial optimization. Mathematics of Operations Research, 39(1):31–45, 2014.

Nicolò Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. Journal of Computer and System Sciences, 78(5):1404–1422, 2012.

Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In Proceedings of the 30th International Conference on Machine Learning, pages 151–159, 2013.
Varsha Dani, Thomas Hayes, and Sham Kakade. Stochastic linear optimization under bandit feedback. In Proceedings of the 21st Annual Conference on Learning Theory, pages 355–366, 2008.

Victor Gabillon, Branislav Kveton, Zheng Wen, Brian Eriksson, and S. Muthukrishnan. Adaptive submodular maximization in bandit setting. In Advances in Neural Information Processing Systems 26, pages 2697–2705, 2013.

Victor Gabillon, Branislav Kveton, Zheng Wen, Brian Eriksson, and S. Muthukrishnan. Large-scale optimistic adaptive submodularity. In Proceedings of the 28th AAAI Conference on Artificial Intelligence, 2014.

Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. IEEE/ACM Transactions on Networking, 20(5):1466–1478, 2012.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems (TOIS), 20(4):422–446, 2002.

Raghuveer M. N. Kesavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. Information Theory, IEEE Transactions on, 56(6):2980–2998, 2010.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. IEEE Computer, 42(8):30–37, 2009.

Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Brian Eriksson. Matroid bandits: Fast combinatorial optimization with learning. CoRR, abs/1403.5045, 2014a.

Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Michal Valko. Polymatroid bandits. CoRR, abs/1405.7752, 2014b.

Shyong Lam and Jon Herlocker. MovieLens 1M Dataset. http://www.grouplens.org/node/12, 2013.

Gergely Neu and Gbor Bartk. An efficient algorithm for learning with semi-bandit feedback. In Sanjay Jain, Rmi Munos, Frank Stephan, and Thomas Zeugmann, editors, Algorithmic Learning Theory, volume 8139 of Lecture Notes in Computer Science, pages 234–248. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40934-9. doi: 10.1007/978-3-642-40935-6_17. URL http://dx.doi.org/10.1007/978-3-642-40935-6_17

Christos Papadimitriou and Kenneth Steiglitz. Combinatorial Optimization. Dover Publications, Mineola, NY, 1998.

Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. CoRR, abs/1301.2609, 2013.

Daniel Russo and Benjamin Van Roy. An information-theoretic analysis of thompson sampling. CoRR, abs/1403.5341, 2014.

W.R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika, 25(3/4):285–294, 1933.

Zheng Wen, Branislav Kveton, Brian Eriksson, and Sandilya Bhamidipati. Sequential Bayesian search. In Proceedings of the 30th International Conference on Machine Learning, pages 977–983, 2013.
A Proof for Theorem 1

A.1 Bound on $\sum_{t=1}^{n} \mathbb{E}[g(A^*, \theta^*) - U_t(A^*)]$

We first prove that if we choose

$$c \geq \min \left\{ \sqrt{\ln\left(\frac{\lambda L n}{\sqrt{2\pi}}\right)}, \sqrt{d \ln\left(\frac{2dKn\lambda}{\sqrt{2\pi}}\right)} \right\}, \quad (14)$$

then $\sum_{t=1}^{n} \mathbb{E}[g(A^*, \theta^*) - U_t(A^*)] \leq 1$. To prove this result, we use the following inequality for truncated Gaussian distribution.

**Lemma 1.** If $X \sim N(\mu, \sigma^2)$, then we have

$$\mathbb{E}[X 1\{X \geq 0\}] = \mu \left[1 - \Phi_G\left(\frac{-\mu}{\sigma}\right)\right] + \frac{\sigma}{\sqrt{2\pi}} \exp\left(-\frac{\mu^2}{2\sigma^2}\right),$$

where $\Phi_G$ is the cumulative distribution function (CDF) of the standard Gaussian distribution $N(0, 1)$. Furthermore, if $\mu \leq 0$, we have $\mathbb{E}[X 1\{X \geq 0\}] \leq \frac{\sigma}{\sqrt{2\pi}} \exp\left(-\frac{\mu^2}{2\sigma^2}\right)$.

Based on Lemma 1, we can prove the following lemmas:

**Lemma 2.** If $c \geq \sqrt{\ln\left(\frac{\lambda L n}{\sqrt{2\pi}}\right)}$, then we have $\sum_{t=1}^{n} \mathbb{E}[g(A^*, \theta^*) - U_t(A^*)] \leq 1$.

**Proof.** We have the following naive bound:

$$g(A^*, \theta^*) - U_t(A^*) = \sum_{e \in A^*} \left[\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e}\right]$$

$$\leq \sum_{e \in A^*} \left[\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e}\right] 1\left\{\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e} \geq 0\right\}$$

$$\leq \sum_{e \in E} \left[\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e}\right] 1\left\{\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e} \geq 0\right\}.$$

Notice that conditioning on $\mathcal{H}_t$, $\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e}$ is a Gaussian random variable with mean $-c\sqrt{\phi_e^T \Sigma_t \phi_e}$ and variance $\phi_e^T \Sigma_t \phi_e$. Thus, from Lemma 1 we have

$$\mathbb{E}_{\mathcal{H}_t} g(A^*, \theta^*) - U_t(A^*)|\mathcal{H}_t|$$

$$\leq \sum_{e \in E} \mathbb{E}_{\mathcal{H}_t} \left[\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e}\right] 1\left\{\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c\sqrt{\phi_e^T \Sigma_t \phi_e} \geq 0\right\}$$

$$\leq \sum_{e \in E} \sqrt{\phi_e^T \Sigma_t \phi_e} \exp\left(-\frac{c^2}{2}\right)$$

$$\leq \exp\left(-\frac{c^2}{2}\right) \sum_{e \in E} \frac{\lambda ||\phi_e||}{\sqrt{2\pi}} \leq \exp\left(-\frac{c^2}{2}\right) \frac{\lambda L}{\sqrt{2\pi}}, \quad (15)$$

where the last two inequalities follow from the fact that $\phi_e^T \Sigma_t \phi_e \leq \phi_e^T \Sigma \phi_e \leq \lambda^2 ||\phi_e||^2 \leq \lambda^2$, since $||\phi_e|| \leq 1$ by assumption. Thus we have

$$\mathbb{E} \left[\sum_{t=1}^{n} [g(A^*, \theta^*) - U_t(A^*)]\right] \leq \exp\left(-\frac{c^2}{2}\right) \frac{n\lambda L}{\sqrt{2\pi}}. \quad (16)$$

If we choose $c \geq \sqrt{2 \ln\left(\frac{\lambda L n}{\sqrt{2\pi}}\right)}$, then we have $E[\sum_{t=1}^{n} [g(A^*, \theta^*) - U_t(A^*)]] \leq 1.$
Lemma 3. If \( c \geq \sqrt{d \ln \left( \frac{24K\eta \lambda}{2d\sigma} \right)} \), then we also have \( \sum_{i=1}^{n} E[g(A^*, \theta^*) - U_t(A^*)] \leq 1 \).

Proof. We use \( v_1, \ldots, v_d \) to denote a fixed set of \( d \) orthonormal eigenvectors of \( \Sigma_t \), and \( \Lambda_1^2, \ldots, \Lambda_d^2 \) to denote the associated eigenvalues. Notice that for \( i \neq j \), we have \( v_i^T \Sigma_t v_j = \Lambda_i^2 v_i^T v_j = 0 \). \( \forall i = 1, \ldots, d \), we define \( v_{i+d} = -v_i \) and \( \Lambda_{i+d} = \Lambda_i \), which allows us to define the following conic decomposition:

\[
\phi_e = \sum_{i=1}^{2d} \alpha_{ei} v_i, \quad \forall e \in E,
\]

subject to the constraints that \( \alpha_{ei} \geq 0, \forall (e, i) \). Notice that \( \alpha_{ei}'s \) are uniquely determined. Furthermore, for \( i \) and \( j \) s.t. \( |i - j| = d \), by definition of conic decomposition, we have \( \alpha_{ei} \alpha_{ej} = 0 \). In other words, \( \alpha_e \) is a \( d \)-sparse vector.

Since we assume that \( \|\phi_e\| \leq 1 \), we have that \( \sum_{i=1}^{2d} \alpha_{ei}^2 \leq 1, \forall e \in E \). Thus, for any \( e \), we have that \( \langle \phi_e, \theta^* - \bar{\theta}_t \rangle = \sum_{i=1}^{2d} \alpha_{ei} \langle v_i, \theta^* - \bar{\theta}_t \rangle \) and

\[
\phi_e^T \Sigma_t \phi_e = \left( \sum_{i=1}^{2d} \alpha_{ei} v_i^T \right) \Sigma_t \left( \sum_{j=1}^{2d} \alpha_{ej} v_j \right) = \sum_{i=1}^{2d} \sum_{j=1}^{2d} \alpha_{ei} \alpha_{ej} v_i^T v_j.
\]

Notice that for \( i \neq j \), if \( |i - j| \neq d \), then \( v_i^T \Sigma_t v_j = 0 \); on the other hand, if \( |i - j| = d \), \( \alpha_{ei} \alpha_{ej} = 0 \). Thus, if \( i \neq j \), we have \( \alpha_{ei} \alpha_{ej} v_i^T \Sigma_t v_j = 0 \). Consequently,

\[
\phi_e^T \Sigma_t \phi_e = \sum_{i=1}^{2d} \alpha_{ei}^2 v_i^T \Sigma_t v_i = \sum_{i=1}^{2d} \alpha_{ei}^2 \Lambda_i^2.
\]

Thus we have

\[
\sqrt{\phi_e^T \Sigma_t \phi_e} = \sqrt{\sum_{i=1}^{2d} \alpha_{ei}^2 \Lambda_i^2} \geq \frac{1}{\sqrt{d}} \sum_{i=1}^{2d} \alpha_{ei} \Lambda_i,
\]

where the inequality follows from Cauchy-Schwartz inequality, specifically, define \( s_i = 1 \) if \( \alpha_{ei} \Lambda_i \neq 0 \), and \( s_i = 0 \) if \( \alpha_{ei} \Lambda_i = 0 \), then we have

\[
\sum_{i=1}^{2d} \alpha_{ei} \Lambda_i = \sum_{i=1}^{2d} \alpha_{ei} \Lambda_i s_i \leq \sqrt{\sum_{i=1}^{2d} s_i^2} \sqrt{\sum_{i=1}^{2d} \alpha_{ei}^2 \Lambda_i^2} \leq \sqrt{d} \sqrt{\sum_{i=1}^{2d} \alpha_{ei}^2 \Lambda_i^2},
\]

where the last inequality follows from the fact that \( \alpha_e \) is \( d \)-sparse. Thus, for any \( e \), we have that

\[
\langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c \sqrt{\phi_e^T \Sigma_t \phi_e} \leq \sum_{i=1}^{2d} \alpha_{ei} \langle v_i, \theta^* - \bar{\theta}_t \rangle - c \frac{1}{\sqrt{d}} \sum_{i=1}^{2d} \alpha_{ei} \Lambda_i.
\]

Consequently, we have

\[
\sum_{e \in A^*} \langle \phi_e, \theta^* - \bar{\theta}_t \rangle - c \sqrt{\phi_e^T \Sigma_t \phi_e} \leq \sum_{i=1}^{2d} \left( \langle v_i, \theta^* - \bar{\theta}_t \rangle - \frac{c \Lambda_i}{\sqrt{d}} \right) \left( \sum_{e \in A^*} \alpha_{ei} \right).
\]

Define \( X_i = \langle v_i, \theta^* - \bar{\theta}_t \rangle - \frac{c \Lambda_i}{\sqrt{d}} \), notice that conditioning on \( H_t \), we have \( X_i | H_t \sim N \left( -\frac{c \Lambda_i}{\sqrt{d}}, \Lambda_i^2 \right) \).
Hence we have
\[
\sum_{e \in A^*} \langle \phi_e, \theta^* - \tilde{\theta} \rangle - c \sqrt{\phi_e^T \Sigma_t \phi_e} \leq \sum_{i=1}^{2d} X_i \left[ \sum_{e \in A^*} \alpha_{ei} \right]
\]
\[
\leq \sum_{i=1}^{2d} X_i \mathbb{1}\{X_i \geq 0\} \left[ \sum_{e \in A^*} \alpha_{ei} \right],
\]
where the inequality (b) follows from the fact that \(X_i \leq X_i \mathbb{1}\{X_i \geq 0\}\) and \(\left[ \sum_{e \in A^*} \alpha_{ei} \right] \geq 0\). On the other hand, notice that \(|A^*| \leq K\)
\[
\sum_{e \in A^*} \alpha_{ei} \leq |A^*| \sqrt{\sum_{e \in A^*} \alpha_{ei}^2} \leq |A^*| \sqrt{\sum_{e \in A^*} \sum_{j=1}^{d} \alpha_{ej}^2} \leq |A^*| \sqrt{\sum_{e \in A^*} 1} = |A^*| \leq K.
\]
Since \(X_i \mathbb{1}\{X_i \geq 0\} \geq 0\), we have
\[
\sum_{e \in A^*} \langle \phi_e, \theta^* - \tilde{\theta} \rangle - c \sqrt{\phi_e^T \Sigma_t \phi_e} \leq K \sum_{i=1}^{2d} X_i \mathbb{1}\{X_i \geq 0\},
\]
notice that the RHS does not include \(A^*\). Hence we have
\[
E_{\theta^*}[g(A^*, \theta^*) - U_t(A^*)|H_t] = E_{\theta^*}\left[ \sum_{e \in A^*} \langle \phi_e, \theta^* - \tilde{\theta} \rangle - c \sqrt{\phi_e^T \Sigma_t \phi_e} |H_t \right]
\]
\[
\leq K \sum_{i=1}^{2d} E_{\theta^*}[X_i \mathbb{1}\{X_i \geq 0\}|H_t]
\]
\[
\leq K \sum_{i=1}^{2d} \frac{\Lambda_i}{\sqrt{2\pi}} \exp \left( -\frac{c^2}{2d} \right) \leq \frac{2dK\lambda}{\sqrt{2\pi}} \exp \left( -\frac{c^2}{2d} \right),
\]
where the last inequality follows from the fact that \(\Lambda_i \leq \lambda\). Hence we have
\[
\sum_{t=1}^{n} E[g(A^*, \theta^*) - U_t(A^*)] \leq \frac{2dK\lambda}{\sqrt{2\pi}} \exp \left( -\frac{c^2}{2d} \right),
\]
if we choose \(c \geq \sqrt{2d \ln \left( \frac{2dK\lambda}{\sqrt{2\pi}} \right)}\), then we have \(\sum_{t=1}^{n} E[f(A^*, \theta^*) - U_t(A^*)] \leq 1\).

Combining the results from Lemma 2 and 3, we have proved that if
\[
c \geq \min \left\{ \sqrt{\ln \left( \frac{\Lambda L n}{\sqrt{2\pi}} \right)}, \sqrt{d \ln \left( \frac{2dK\lambda}{\sqrt{2\pi}} \right)} \right\},
\]
then \(\sum_{t=1}^{n} E[g(A^*, \theta^*) - U_t(A^*)] \leq 1\).

### A.2 Bound on \(\sum_{t=1}^{n} E\left[ \sum_{e \in A^t} \sqrt{\phi_e^T \Sigma_t \phi_e} \right]\)

In this subsection, we derive a bound on \(\sum_{t=1}^{n} E\left[ \sum_{e \in A^t} \sqrt{\phi_e^T \Sigma_t \phi_e} \right]\). Our analysis is motivated by the analysis in Dani et al. [2008]. Specifically, \(\forall t = 1, \ldots, n\) and \(\forall k = 1, \ldots, |A^t|\), we use \(a_k^t\) to denote the \(k\)th item in \(A^t\). To simplify the exposition, we define
\[
z_{t,k} = \sqrt{\phi_k^t \Sigma_t \phi_k^t} = \frac{\phi_k^t \Sigma_t \phi_k^t}{\sqrt{\phi_k^t \Sigma_t \phi_k^t}}.
\]
Hence, we have \(\sum_{t=1}^{n} E\left[ \sum_{e \in A^t} \sqrt{\phi_e^T \Sigma_t \phi_e} \right] = E\left[ \sum_{t=1}^{n} \sum_{k=1}^{|A^t|} z_{t,k} \right]\).
In this subsection, we provide a bound on $\sum_{t=1}^{n} \sum_{k=1}^{\lfloor A^t \rfloor} z_{t,k}$, for any realization of random variable $w_t$, $\theta_t$, $A^t$, $A^*$, and $\theta^*$. First, notice that $\Sigma_t^{-1}$ is the Gramian matrix and satisfies

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{1}{\sigma^2} \sum_{k=1}^{\lfloor A^t \rfloor} \phi_{a_t^k} \phi_{a_t^k}^T.$$  \hfill (22)

Hence for any $t$, $k$, we have that

$$\det[\Sigma_{t+1}^{-1}] \geq \det[\Sigma_{t}^{-1} + \frac{1}{\sigma^2} \phi_{a_t^k} \phi_{a_t^k}^T] = \det[\Sigma_t^{-1} \left(I + \frac{1}{\sigma^2} \Sigma_t^{\frac{1}{2}} \phi_{a_t^k} \phi_{a_t^k}^T \Sigma_t^{\frac{1}{2}}\right)] = \det[\Sigma_t^{-1} \left(I + \frac{1}{\sigma^2} \phi_{a_t^k} \phi_{a_t^k}^T \Sigma_t\right)]$$

$$= \det[\Sigma_t^{-1} \left(I + \frac{\mu^2 k}{\sigma^2}\right)].$$ \hfill (23)

Hence we have that

$$\left(\det[\Sigma_t^{-1}]\right)^{|A^t|} \geq \left(\det[\Sigma_t^{-1}]\right)^{|A^t|} \prod_{t=1}^{n} \left(1 + \frac{\mu^2 k}{\sigma^2}\right).$$ \hfill (24)

**Remark 1.** This is where the extra $O(\sqrt{K})$ factor arises. Notice that this extra factor is purely due to linear generalization. Specifically, if $\Phi = I$, then $\Sigma_t$’s and $\Sigma_t^{-1}$’s will be diagonal, and we have

$$\det[\Sigma_t^{-1}] = \det[\Sigma_t^{-1}] \prod_{t=1}^{n} \left(1 + \frac{\mu^2 k}{\sigma^2}\right).$$ \hfill (25)

Notice that Equation (24) further implies that

$$\left(\det[\Sigma_t^{-1}]\right)^K \geq \left(\det[\Sigma_t^{-1}]\right)^K \prod_{t=1}^{n} \left(1 + \frac{\mu^2 k}{\sigma^2}\right),$$ \hfill (26)

since $\det[\Sigma_{t+1}] \geq \det[\Sigma_t]$, and $|A^t| \leq K$. Recall that $\det[\Sigma_t^{-1}] = \left(\frac{1}{\lambda}\right)^d$, we have that

$$\left(\det[\Sigma_{n+1}^{-1}]\right)^K \geq \left(\det[\Sigma_t^{-1}]\right)^K \prod_{t=1}^{n} \prod_{k=1}^{\lfloor A^t \rfloor} \left(1 + \frac{\mu^2 k}{\sigma^2}\right) = \frac{1}{\lambda^{2dK}} \prod_{t=1}^{n} \prod_{k=1}^{\lfloor A^t \rfloor} \left(1 + \frac{\mu^2 k}{\sigma^2}\right).$$ \hfill (27)

On the other hand, we have

$$\text{trace} \left[\Sigma_{n+1}^{-1}\right] = \text{trace} \left[\frac{1}{\lambda^2} I + \sum_{t=1}^{n} \sum_{k=1}^{\lfloor A^t \rfloor} \phi_{a_t^k} \phi_{a_t^k}^T\right] = \frac{d}{\lambda^2} + \sum_{t=1}^{n} \sum_{k=1}^{\lfloor A^t \rfloor} \|\phi_{a_t^k}\|^2 \leq \frac{d}{\lambda^2} + nK,$$ \hfill (28)

where the last inequality follows from the assumption that $\|\phi_c\| \leq 1$, $\forall c \in E$ and $|A^t| \leq K$. From the trace-determinant inequality, we have

$$\frac{1}{d} \text{trace} \left[\Sigma_{n+1}^{-1}\right] \geq \left(\det[\Sigma_{n+1}^{-1}]\right)^{\frac{1}{d}},$$

which implies that

$$\left(\frac{1}{\lambda^2} + \frac{nK}{d}\right)^{dK} \geq \left(\frac{1}{d} \text{trace} \left[\Sigma_{n+1}^{-1}\right]\right)^{dK} \geq \left(\det[\Sigma_{n+1}^{-1}]\right)^K \geq \frac{1}{\lambda^{2dK}} \prod_{t=1}^{n} \prod_{k=1}^{\lfloor A^t \rfloor} \left(1 + \frac{\mu^2 k}{\sigma^2}\right).$$
Taking the logarithm, we have
\[
dK \log \left(1 + \frac{nK\lambda^2}{d}\right) \geq \sum_{t=1}^{n} \sum_{k=1}^{|A^t|} \log \left(1 + \frac{z_{t,k}^2}{\sigma^2}\right).
\]  
(29)

Notice that \(z_{t,k}^2 = \phi_{a_t}^T \Sigma_t \phi_{a_t}\), hence we have that \(0 \leq z_{t,k}^2 \leq \phi_{a_t}^T \Sigma_t \phi_{a_t} \leq \lambda^2 \|\phi_{a_t}\|^2 \leq \lambda^2\).

We have the following technical lemma:

**Lemma 4.** For any real number \(x \in [0, \lambda^2]\), we have \(x \leq \lambda^2 \frac{\log(1 + \lambda^2 \sigma^2)}{\log(1 + x \sigma^2)}\).

**Proof.** Define \(h(x) = \frac{\lambda^2}{\log(1 + \lambda^2 \sigma^2)} \log(1 + \frac{x \sigma^2}{1}) - x\), thus we only need to prove \(h(x) \geq 0\) for \(x \in (0, \lambda^2]\). Notice that \(h(x)\) is a strictly concave function for \(x \geq 0\), and \(h(\lambda^2) = 0\). From Jensen’s inequality, for any \(x \in (0, \lambda^2]\), we have \(h(x) > 0\).

Hence we have that
\[
\sum_{t=1}^{n} \sum_{k=1}^{|A^t|} z_{t,k}^2 \leq \lambda^2 \frac{\log(1 + \lambda^2 \sigma^2)}{\log(1 + \frac{\lambda^2 \sigma^2}{1})} \sum_{t=1}^{n} \sum_{k=1}^{|A^t|} \log \left(1 + \frac{z_{t,k}^2}{\sigma^2}\right) \leq \frac{dK \lambda^2 \log(1 + \frac{nK\lambda^2}{d})}{\log(1 + \frac{\lambda^2 \sigma^2}{1})}.
\]  
(30)

Finally, we have that
\[
\sum_{t=1}^{n} \sum_{k=1}^{|A^t|} z_{t,k} \leq \sqrt{nK} \sqrt{\sum_{t=1}^{n} \sum_{k=1}^{|A^t|} z_{t,k}^2} \leq K \lambda \sqrt{\frac{d \log(1 + \frac{nK\lambda^2}{d})}{\log(1 + \frac{\lambda^2 \sigma^2}{1})}}.
\]  
(31)

Recall that the above bound holds for any realization of random variables, thus, we have
\[
\mathbb{E} \left[ \sum_{t=1}^{n} \left[ U_t(A^t) - g(A^t, \theta^*) \right] \right] = c \mathbb{E} \left[ \sum_{t=1}^{n} \sum_{k=1}^{|A^t|} z_{t,k} \right] \leq cK \lambda \sqrt{\frac{d \log(1 + \frac{nK\lambda^2}{d})}{\log(1 + \frac{\lambda^2 \sigma^2}{1})}}.
\]

With
\[
c = \min \left\{ \sqrt{\ln \left( \frac{\lambda \ln n}{\sqrt{2\pi}} \right)}, \sqrt{\ln \left( \frac{2dKn\lambda}{\sqrt{2\pi}} \right)} \right\},
\]
(32)
and combining the results in the previous subsection, we have proved Theorem 1.