Management Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Speed Up the Cold-Start Learning in Two-Sided Bandits with Many Arms

Mohsen Bayati, Junyu Cao, Wanning Chen

To cite this article:
Mohsen Bayati, Junyu Cao, Wanning Chen (2026) Speed Up the Cold-Start Learning in Two-Sided Bandits with Many Arms. Management Science

Published online in Articles in Advance 26 Jun 2026

. https://doi.org/10.1287/mnsc.2022.03394

https://pubsonline.informs.org/journal/mnsc

MANAGEMENT SCIENCE
Articles in Advance, pp. 1–20
ISSN 0025-1909 (print), ISSN 1526-5501 (online)

Speed Up the Cold-Start Learning in Two-Sided Bandits with Many Arms

Mohsen Bayati,[a] Junyu Cao,[b] Wanning Chen[c,*]

[a] Graduate School of Business, Stanford University, Palo Alto, California 94305; [b] McCombs School of Business, The University of Texas at Austin, Austin, Texas 78712; [c] Foster School of Business, University of Washington, Seattle, Washington 98195

*Corresponding author

Contact: bayati@stanford.edu, https://orcid.org/0000-0002-7280-912X (MB); junyu.cao@mccombs.utexas.edu, https://orcid.org/0000-0001-9235-1411 (JC); wnchen@uw.edu, https://orcid.org/0000-0003-0167-9257 (WC)

Received: November 2, 2022
Revised: December 17, 2024; November 4, 2025
Accepted: November 10, 2025
Published Online in Articles in Advance: June 26, 2026

https://doi.org/10.1287/mnsc.2022.03394

Abstract. Multiarmed bandit (MAB) algorithms are efficient approaches to reduce the opportunity cost of online experimentation and are used by companies to find the best product from periodically refreshed product catalogs. However, these algorithms face the so-called *cold-start* at the onset of the experiment due to a lack of knowledge of customer preferences for new products, requiring an initial data collection phase known as the *burn-in period*. During this period, standard MAB algorithms operate like randomized experiments, incurring large burn-in costs which scale with the large number of products. We attempt to reduce the burn-in by identifying that many products can be cast into *two-sided products* and then naturally model the rewards of the products with a matrix, whose rows and columns represent the two sides, respectively. Next, we design *two-phase* bandit algorithms that first use subsampling and low-rank matrix estimation to obtain a substantially smaller targeted set of products and then apply a Upper Confidence Bound procedure on the target products to find the best one. We theoretically show that the proposed algorithms lower costs and expedite the experiment in cases when there is limited experimentation time along with a large product set. Our analysis also provides insights into three experiment regimes of long, short, and ultra-short horizon, determined by the dimensions of the matrix. Empirical evidence from both synthetic data and a real-world data set on music streaming services validates the superior performance suggested by our theory.

History: Accepted by J. George Shanthikumar, data science.

Supplemental Material: The online appendix and data files are available at https://doi.org/10.1287/mnsc.2022.03394.

Keywords: online experimentation • multiarmed bandit with many arms • cold-start • low-rank matrix

1. Introduction

Online experimentation has become a major force in assessing different versions of a product, finding better ways to engage customers, and generating more revenue. These experiments happen on a recurring basis with periodically refreshed product catalogs. One family of online experimentation is the so-called multiarmed bandit (MAB) algorithms, where each arm models a version of the product. At each time period, the decision maker tests out a version, learns from the observed outcome, and proceeds to the next time period. It is well known that MAB algorithms balance exploration and exploitation to handle the lack of prior information at the onset of an experiment. They can dynamically allocate experimentation effort to versions that are performing well, in the meantime allocating less effort to versions that are underperforming. As a result, MAB algorithms can identify the optimal product under relatively low opportunity costs incurred from selecting suboptimal products. Consequently, fewer customers are subjected to potentially inferior experiences. We want to highlight that what is treated as a product in our context is not confined to merchandise in the e-commerce setting. It includes broader categories from ads campaign product in online advertisement to drug product in drug development.

However, when there is a large product set, even at the onset of the experimentation, MAB algorithms would face what is known as the *cold-start problem*: Many versions of a product are brand new to the customers. It is not yet clear to the algorithms how to draw any useful inferences before gathering some information. Therefore,

many MAB algorithms involve an initial data collection phase, known as the *burn-in period* (Lu 2019).[1] During this period, MAB algorithms operate as nearly equivalent to randomized experiments, incurring costs that scale with the number of products. Under such circumstances, the cumulative cost, or regret, of current practices is huge due to a costly and lengthy burn-in period.[2]

Our goal is to come up with a framework that can be cost-effective, which is achieved by first identifying a two-sided structure of the products. In many applications, because the products have two major components, the bandit arms can be viewed as versions of a *two-sided product*. To be more specific, let us provide two examples that will reappear in the later empirical studies.

Example 1. Consider an online platform that wants to select one user segment and one content creator segment that interact the most to target an advertisement campaign (Lops et al. 2011, Geng et al. 2020, Bhargava 2022). Such an advertisement campaign is a two-sided product where user segment and content creator segment are its two sides. There are many user segments and many content creator segments. In each time period, the platform tests a pair of user segments and content creator segments and observes the amount of interaction between them, based on which, the platform decides which pair to test next. The number of products scales quickly: 10 user segments and 10 content creator segments already give rise to 100 pairs, and this is just an underestimate of how many segments there are for each side.

Example 2. Consider an online platform that wants to design a homepage with a headline and an image. The platform's goal is to select a pair of headlines and images that attracts the most amount of user engagement. The homepage design is a two-sided product with headline and image as two sides. Similar to Example 1, the number of pairs can easily be more than several hundred.

Besides the two examples above, there are plenty of other two-sided products: Airbnb experiences (activities and locations), Stitch Fix personal styling (tops and pants), Expedia travel (hotels and flights), car sales (packages and prices), drug development (composition of ingredients and dosages), and so on.

The task of finding the best version of a two-sided product can be translated into finding the maximum entry of a partially observed reward matrix, which can help accelerate the learning process. That is, we can model different choices of one side to be rows of the matrix, different choices of the other side to be the columns, and the unknown rewards for the corresponding row-column pairs (e.g., the amount of interaction in Example 1 and the amount of user engagement in Example 2) to be the matrix entries. Mathematically, suppose the first side of the product has d_r number of choices and the second side has d_c number of choices, then the reward matrix contains the rewards of $d_r d_c$ versions of the two-sided product as entries. As mentioned previously, current algorithms will randomize on the initial $\Omega(d_r d_c)$ number of periods, leading to a very costly burn-in period because d_r and d_c are often very large. Furthermore, when the time horizon T is rather small compared with the order $d_r d_c$, a large cumulative regret that scales with $d_r d_c$ would arise because of the costly burn-in period.

By casting the rewards into a matrix and considering its *low-rank* structure, we can expedite the experiment and reduce the regret by orders of magnitude. To understand the low-rank structure, we note that the rewards for different versions of a two-sided product depend on the interactions between some latent features of the two sides. Low-rank simply means that, suppose $\mathfrak{r}$ is the rank of the matrix, then we only need as few as $\mathfrak{r}$ latent features to explain each side, such that $\mathfrak{r} \ll \min(d_r, d_c)$. Because of such a low-rank property and with the help from low-rank matrix estimators, we only need to collect a small amount of data to have an estimate of the rewards for all arms of interest in the burn-in period. Such an estimate will be accurate enough to help screen out majority of the suboptimal arms, thus requiring substantially less subsequent exploration and greatly reducing the burn-in cost.

1.1. Our Contributions

1.1.1. Algorithm. We design a new bandit algorithm called the low-rank bandit (LRB). The algorithm can efficiently choose from a large set of two-sided products given a short time horizon. It is split into two phases: "pure exploration phases" and "targeted exploration + exploitation phases." At each period, the algorithm follows a schedule to enter either of the two phases. If it is in a pure exploration phase, a forced sample is selected uniformly at random and is used for the low-rank matrix estimation (i.e., the forced-sample estimates). The forced-sample estimates are used to screen out arms that are far from the optimal arm. The remaining arms form a targeted set whose size is much smaller than the initial product set. If the algorithm is in a targeted exploration + exploitation phase, we choose a product based on an off-the-shelf stochastic MAB algorithm (e.g., Upper Confidence Bound (UCB), Thompson sampling) applied on the current targeted set as a subroutine. For illustrative purposes, we use UCB as the subroutine throughout the paper.

Moreover, we propose submatrix-sampled LRB (ss-LRB) that adds a subsampling prestep to our regular LRB algorithm when we are extremely sensitive about time. Given an ultra-short horizon (formally defined in Section 3.3.3) in the face of a huge number of arms, we can instead sample a submatrix of smaller dimensions and apply the LRB to it. The intuition behind this is as follows: Even though the best entry in the

submatrix might be suboptimal, because the horizon is ultra-short, the cost we save from not exploring the much larger matrix compensates for the regret we incur from picking a suboptimal entry.

1.1.2. Theory. We establish nonasymptotic dependent and independent regret bounds of LRB as functions of a parameter we term the "filtering resolution" and demonstrate how to optimize such bounds. Simply put, the filtering resolution controls the number of arms that will enter the targeted exploration + exploitation phase. Specifically, we show that our algorithm achieves a strictly better regret than the typical $O(\sqrt{d_r d_c T})$ for standard noncontextual bandits when T is small. By leveraging a structure of the reward distribution in the low-rank matrix characterized by a "shrinkage rate," we show a further improvement on the regret. For the ss-LRB algorithm we propose, we derive the optimal subsampling ratio and an improved regret bound, both of which depend on the form of a "subsampling cost" function.

Our theory sheds light on customized strategies given different time horizon lengths and the product set sizes, distinct from most existing bandit theoretical analysis that studies dependence of the regret on the time horizon asymptotically and results in suboptimal results for short horizons.

1.1.3. Empirics. Empirical evidence from synthetic data validates the superior performance of our algorithm. In addition, we illustrate the practical relevance of our algorithms by proposing a data-driven approach to select experiment-dependent hyperparameters. We evaluate such an approach on an advertisement targeting problem for the music streaming service, where we want to learn the best combination of user group and creator group to target an advertisement campaign. We show that our bandit algorithm significantly outperforms existing ones in finding the best combination efficiently under various time horizon lengths.

1.1.4. Reduction Model for Contextual Bandits. Additionally, we propose a new modeling framework to transform a stochastic linear bandit setting (a general class of bandit problems that contains contextual bandits) to the scope of our low-rank bandit problem. This allows us to empirically showcase LRB's effectiveness of reducing the burn-in in contextual settings by comparing it with a standard linear bandit method called optimism in the face of uncertainty linear bandit algorithm (OFUL) (Abbasi-Yadkori et al. 2011).

1.2. Related Literature

We first review the bandit literature relevant to online decision making under bandit feedback and then turn to related works on low-rank models and matrix-structured data.

Our work focuses on regret minimization in noncontextual bandit problems, with canonical algorithms including Thompson sampling (Thompson 1933, Agrawal and Goyal 2012, Gopalan et al. 2014, Russo et al. 2018) and upper confidence bound (Lai 1987, Kaelbling 1993, Katehakis and Robbins 1995, Auer et al. 2002a). Like ours, these algorithms explore and exploit without contextual information; that is, they make no assumptions on dependence of the arm rewards. Beyond the noncontextual setting, a large body of literature studies bandit problems with contextual information, where arm rewards are parametric functions of the observed contexts. In particular, extensive research has focused on the linear bandit (Auer et al. 2002b, Dani et al. 2008, Li et al. 2010, Rusmevichientong and Tsitsiklis 2010, Abbasi-Yadkori et al. 2011, Chu et al. 2011, Agrawal and Devanur 2014) and its extension under generalized linear models (GLMs; Filippi et al. 2010, Li et al. 2017, Kveton et al. 2020), in which the expected reward is a linear (or generalized linear) function of the context. We refer the readers to Lattimore and Szepesvári (2020) for a comprehensive overview.

Classical analysis of bandits focuses on asymptotic behavior of the regret bound under long experimentation horizons, assuming the number of arms is much smaller than the horizon. Specifically, the regret bound of a k-armed bandit consists of two terms: one linear in k but constant in T plus an order $\sqrt{kT}$ term.[3] The former is typically ignored because, for large T, it is dominated by the latter term. But for large k problems such as our two-sided bandit setting with $k = d_r d_c$, this linear term, $d_r d_c$, can be as large as the latter term. This challenge, which creates a significant cold-start problem in short-horizon settings, has been the motivation for several approaches. These include combining clustering and MAB algorithms (Miao et al. 2022, Keskin et al. 2024), warm starting the bandit problem using historical data (Banerjee et al. 2022), or using an integer program to constrain the action set (Bastani et al. 2022).

A separate line of work on many-armed or infinitely armed bandit problems (Berry et al. 1997, Bonald and Proutiere 2013) addresses the cold-start challenge by employing subsampling to reduce the opportunity cost (Wang et al. 2009, Carpentier and Valko 2015, Chaudhuri and Kalyanakrishnan 2018, Bayati et al. 2020). The key distinction of our work from this line of research lies in leveraging the inherent matrix structure of the two-sided products, which allows for a more informed arm selection for targeted exploration + exploitation. Another way in the infinitely armed bandit literature to mitigate the cold-start issue is to impose structural assumptions on the reward function, such as linearity over a compact action set, as explored in Mersereau et al. (2009) and Rusmevichientong and Tsitsiklis (2010). In such cases as in the linear

bandit setting, pulling *any action* provides information about the rewards of *all actions*.

Low-rank matrix estimators have become established tools for solving offline cold-start problems in recommendation systems (Lika et al. 2014, Zhang et al. 2014, Volkovs et al. 2017), and they are building blocks for many applications that have inherent low-rank models (Athey et al. 2021, Farias et al. 2022, Agarwal et al. 2023). We refer the readers to Davenport and Romberg (2016) and Chen et al. (2020) for a detailed discussion on low-rank matrix estimators.

Incorporating the low-rank matrix structure in bandit settings has been seen in many other works, which all consider different settings than ours. For example, Katariya et al. (2017) and Trinh et al. (2020) consider only rank 1 matrices, whereas our algorithm works for general low-rank matrices; Kveton et al. (2017) allow a decision maker to noisily choose and observe every entry of a $r \times r$ submatrix (where r is the rank of the underlying unknown matrix) in each period, but we only allow the selection of one entry each time and we assume the rank r is unknown, which is a harder problem. Lu et al. (2018) provide an ensemble sampling based algorithm that lacks theoretical guarantees. Jun et al. (2019) address the contextual bilinear bandit setting where the reward function is a bilinear function of two feature vectors and an unknown low-rank parameter matrix. They propose an OFUL-based algorithm that combines with low-rank matrix estimation. Lu et al. (2021) propose a similar algorithm for low-rank generalized linear models. The Explore-Subspace-Then-Refine (ESTR) algorithm in Jun et al. (2019) and LowESTR algorithm in Lu et al. (2021) use a novel way of exploiting the subspace so as to reduce the problem to linear bandits after exploration. They consider a more general setting and their algorithms can be adapted to solve our problems, by taking the canonical basis vectors as the two feature vectors. We show in Section EC.9 in the Online Appendix that, when narrowing down to the specific noncontextual setting we focus on, our algorithm has better theoretical and empirical performance. Lu et al. (2021), in addition, propose an exponentially weighted average forecaster-based algorithm, which is statistically efficient but not computationally efficient as they have put it. Kallus and Udell (2020) impose a low-rank structure to their underlying parameter matrix for the dynamic assortment personalization problem in which a decision maker picks a best subset of products for a user. Hamidi et al. (2019) also impose a low-rank structure on the arm parameter set to expedite the learning process, but in their setting, contexts are observed that are different from ours. Nakamura (2015) proposes a heuristic UCB-like strategy of collaborative filtering for a recommendation problem that selects the best user-item pairs, which is called the direct mail problem, but no theoretical guarantee for that approach is known. Sam et al. (2023) study a class of Markov decision processes that exhibits latent low-dimensional structure with respect to the relationship between a large number of states and actions. Zhu et al. (2022) consider learning the Markov transition matrix under the low-rank structure. Pal and Jain (2023) propose an explore-then-commit (ETC) approach for the online low-rank matrix completion problem, where the algorithm recommends one item per user at each round. Zhou et al. (2025) propose low-rank tensor bandit algorithms for tensors that are at least three dimensional. For more LRB works, we refer the readers to the literature sections of these related works.

Although the low-rank structure is utilized in all the above-mentioned works, we want to emphasize that our work focuses on a short-horizon learning problem with many arms, and the purpose of utilizing the low-rank property is to efficiently filter suboptimal arms in order to construct a more focused targeted set, under very limited experimentation time.

There are recent papers such as Gupta and Rusmevichientong (2021), Gupta and Kallus (2022), Allouah et al. (2023), and Besbes and Mouchtaki (2023) that focus on practical small data regime with limited sample size: The former group studies how to optimize decision making in a large number of small-data problems, whereas the latter group studies classical pricing and newsvendor problems and shows how a small number of samples can provide valuable information, leveraging the structure of the problems. Even though we focus on the burn-in period with small data, our bandit formulation with two-sided products and low-rank structure is completely different from their offline setups. Our problem is also different from the recent work of Xu and Bastani (2021) that studies how one can learn across many contextual bandit problems, because the contexts in our model are hidden and will be estimated through learning of the low-rank structure.

Finally, recent works such as Bajari et al. (2021) and Johari et al. (2022) also look at online experimentation for two-sided settings, but from a completely different angle. Specifically, their objective is to remove bias due to the spillover effect by using two-sided randomization.

1.3. Organization of the Paper

The remainder of the paper is organized as follows. We describe the problem formulation in Section 2. We present the LRB and our main result on the algorithm's performance in Section 3. In Section 4, we present the submatrix sampling technique to further enhance LRB's performance when the time horizon is ultra-short. Finally, empirical results on simulated data and our evaluation on real music streaming data for the task of advertisement targeting are presented in Section 5. It also includes numerical comparison against linear bandits in contextual settings.

2. Model and Problem

In this section, we formulate our problem as a stochastic multiarmed bandit with (two-sided) arms and introduce notations and assumptions. These arms are represented as row-column pairs of a matrix, and the arm mean rewards are represented as entries of the matrix. We impose a low-rank structure to the reward matrix, which is a building block of our LRB algorithm. Throughout the paper, unless otherwise specified, we use bold capital letters (e.g., $\mathbf{B}$) for matrices and nonbold capital letters for vectors (e.g., V, except for T, which denotes the total time horizon). For a positive integer m, we denote the set of integers $\{1,\dots,m\}$ by $[m]$. For a matrix $\mathbf{B}$, $\mathbf{B}_{ij}$ refers to its entry (i,j) and $\mathbf{B}_j$ refers to its jth row; $\|\mathbf{B}\|_\infty$ denotes the largest entry of a matrix $\mathbf{B}$, that is, $\|\mathbf{B}\|_\infty = \max_{(j,k)\in[d_r]\times[d_c]}|\mathbf{B}_{jk}|$; and $\|\mathbf{B}\|_{2,\infty}$ denotes the largest l_2 norm of all rows of a matrix $\mathbf{B}$.

2.1. Rewards as a Low-Rank Matrix

We work with a $d_r \times d_c$ ground truth mean reward matrix $\mathbf{B}^*$, and our goal is to find the biggest reward entry as we sequentially observe noisy version of picked entries. As introduced earlier, the rows of this matrix correspond to different choices for the first side of the two-sided products and columns correspond to different choices for the second side. Entries of $\mathbf{B}^*$ correspond to the mean rewards for different combinations of the two sides and we assume they are bounded as in the canonical bandit literature. This assumption is standard and makes sure that the maximum regret at any time step is bounded. It aligns with the real-world setting because user preferences are bounded in practice.

Assumption 1 (Boundedness). *Assume* $\|\mathbf{B}^*\|_\infty \le b^*$. *Without loss of generality, we further assume* $b^* \le 1$.

We use the two-sided product in Example 1, ads campaign, as a running example. In this case, the marketer needs to pick a pair of user segment and content creator segment to target the ads. Then we can model user segments as rows of matrix $\mathbf{B}^*$ and content creator segments as columns of matrix $\mathbf{B}^*$. The amounts of interaction between different combinations of user segments and content creator segments are the entries of $\mathbf{B}^*$. The goal under this scenario is to find the pair that interacts the most to target the ads by experimenting with different pairs.

We model the ground truth mean reward matrix $\mathbf{B}^*$ as a low-rank matrix. That is, let $\mathfrak{r}$ be the rank of $\mathbf{B}^*$, and we have $\mathfrak{r} \ll \min\{d_r, d_c\}$. This means the reward for the entry corresponding to row j and column k is a dot product of two $\mathfrak{r}$-dimensional latent feature vectors. Therefore, our reward model is a contextual reward with unobserved features. In Section 5.3, we show an alternative contextual bandit representation of our reward function that is motivated by the standard linear bandit setting framework.

We introduce an incoherence parameter μ and the condition number κ, which is known to be crucial for reliable recovery of the matrix in the low-rank matrix completion literature (Keshavan et al. 2010, Candes and Recht 2012, Chen 2015, Chen et al. 2020). The former measures how much information observing an entry gives about other entries in a matrix and the latter quantifies the numerical stability of a matrix.

Definition 1. For a matrix $\mathbf{B}^* \in \mathbb{R}^{d_r \times d_c}$ of rank $\mathfrak{r}$ with singular value decomposition $\mathbf{B}^* = \mathbf{U}^*\mathbf{D}^*\mathbf{V}^{*\top}$, we denote the *incoherence* parameter as $\mu := \mu(\mathbf{B}^*)$ such that $\|\mathbf{U}^*\|_{2,\infty} \le \sqrt{\mu\mathfrak{r}/d_r}$ and $\|\mathbf{V}^*\|_{2,\infty} \le \sqrt{\mu\mathfrak{r}/d_c}$.

Intuitively, smaller μ means that we can get more information about other products from observing just a few entries, and the matrix completion problem is more tractable. Consequently, less experimentation cost is incurred while exploring a large product set. In contrast, matrices with bigger μ have most of their mass in a relatively small number of elements and are much harder to recover under limited observations.

Definition 2. For a matrix $\mathbf{B}$ of rank $\mathfrak{r}$ with $\mathfrak{r}$ nonzero singular values $\sigma_{\max} = \sigma_1 > \sigma_2 > \cdots > \sigma_{\mathfrak{r}} = \sigma_{\min}$, we denote the *condition number* as $\kappa := \kappa(\mathbf{B})$ such that $\kappa = \sigma_{\max}/\sigma_{\min}$.

Both μ and κ are assumed to be constants.

2.2. Bandit Under the Matrix-Shaped Rewards

We consider the following T-period horizon experiment with bandit feedback: At each time step t, the marketer has access to $d_r d_c$ arms (ads campaigns), and each arm yields an uncertain reward (amount of interaction). At time t, if the marketer pulls arm $(j_t,k_t) \in [d_r]\times[d_c]$, it yields reward $y_t := \mathbf{B}^*_{j_t k_t} + \epsilon_t$, where we assume the observation noises to be independent sub-Gaussian random variables with sub-Gaussian norm at most σ (see definition 5.7 in Vershynin (2010)). Natural examples of sub-Gaussian random variables include any centered and bounded random variable or a centered Gaussian.

Our task is to design a sequential decision-making policy π that learns the underlying arm rewards (parameter matrix $\mathbf{B}^*$) over time in order to maximize the expected reward. Let $\pi_t = (j_t,k_t)$, an element of $[d_r]\times[d_c]$, and denote the arm chosen by policy π at time $t \in [T]$. We compare with an oracle policy π^* that already knows $\mathbf{B}^*$ and thus always chooses the best arm (in expectation), $\pi^* = (j^*,k^*) = arg\ max_{(j,k)\in[d_r]\times[d_c]}\mathbf{B}^*_{jk}$. Thus, if the arm $\pi_t = (j_t,k_t)$ is chosen at time t, expected regret incurred is $\mathbb{E}[\mathbf{B}^*_{j^*k^*} - \mathbf{B}^*_{j_t k_t}]$, where the expectation is with respect to the randomness of ϵ_t and potential

randomness introduced by the policy π. This is simply the difference in expected rewards of π^* and π_t. We seek a policy π that minimizes the cumulative expected regret

$$\text{Regret}_T = \sum_{t=1}^{T} \mathbb{E}_\pi[\mathbf{B}^*_{j^*k^*} - \mathbf{B}^*_{j_t k_t}].$$

Note that throughout this paper, we assume the matrix $\mathbf{B}^*$ is deterministic, which means our notion of regret is *frequentist*.

Together with policy π, the observation model can be characterized by a trace regression model (Hastie et al. 2015). For a given subset of cardinality n, $\mathcal{I} = \{t_1, \ldots, t_n\}$ of $[T]$, we define $Y = \mathfrak{X}^\pi_{\mathcal{I}}(\mathbf{B}^*) + E$, with observation operator $\mathfrak{X}^\pi_{\mathcal{I}}$ (defined below), vector of observed values $Y \in \mathbb{R}^n$ equal to $[y_{t_1}, y_{t_2}, \ldots, y_{t_n}]^\top$, and noise vector $E \in \mathbb{R}^n$ equal to $[\epsilon_{t_1}, \epsilon_{t_2}, \ldots, \epsilon_{t_n}]^\top$. The observation operator $\mathfrak{X}^\pi_{\mathcal{I}}(\cdot)$ takes in a matrix $\mathbf{B}$ and outputs a vector of dimension n (i.e., $\mathfrak{X}^\pi_{\mathcal{I}} : \mathbb{R}^{d_r \times d_c} \to \mathbb{R}^n$). Elements of the output vector are the entries of $\mathbf{B}$ at n observed locations, determined by $\pi_{t_1}, \ldots, \pi_{t_n}$. Specifically, $\mathfrak{X}^\pi_{\mathcal{I}}(\mathbf{B})$ is a vector in $\mathbb{R}^n$, and for each i in $[T]$, its ith entry is defined by $[\mathfrak{X}^\pi_{\mathcal{I}}(\mathbf{B})]_i := \langle \mathbf{B}, \mathbf{X}^\pi_{t_i} \rangle$, where *design matrix* $\mathbf{X}^\pi_{t_i} \in \mathbb{R}^{d_r \times d_c}$ has all of its entries equal to zero, except for the $\pi_{t_i} = (j_{t_i}, k_{t_i})$th entry, which is equal to one. The notation $\langle \cdot, \cdot \rangle$ refers to the trace inner product of two matrices that is defined by $\langle \mathbf{B}_1, \mathbf{B}_2 \rangle := \text{Tr}(\mathbf{B}_1 \mathbf{B}_2^\top)$. The observation operator will appear in the matrix estimator we use.

3. LRB Algorithm

We first give an overview of our two-phase LRB algorithm before describing it in detail in Section 3.1. Then in Section 3.2, we carefully incorporate a matrix element-wise error bound borrowed from the literature to guide our regret analysis. In Section 3.3, we further explain how the so-called "filtering resolution" h balances regrets from the two phases to achieve tightened bounds, with a discussion on how LRB can be effective under different horizon lengths.

Intuitively, our LRB takes advantage of the low-rank reward structure and yields a good estimation of all the arms of interest by only observing a few arms. Thus, we can speed up the cold-start given a relatively short horizon in the face of many arms. As aforementioned, most prior literature operates under a typical but unrealistic formulation of this problem: The set of arms is assumed to be "small" relative to the time horizon T. In particular, in standard asymptotic analysis of the MAB setting, the horizon T scales to infinity while the number of arms remains constant. However, as discussed earlier, the number of arms could be large relative to the time horizon of interest in many practical situations. Standard MAB algorithms will incur at least $\Omega(d_r d_c)$ regret in the initialization stage by forced-sampling each arm once. Leveraging the low-rank matrix structure of the arms, our forced-sampling regret becomes $\tilde{O}(\mathfrak{r}^2(d_r + d_c)/h^2)$. By carefully selecting the filtering resolution h, we can show that the total regret upper bound can be strictly smaller than that of a standard noncontextual bandit for specified horizon lengths.

As mentioned in Section 1.1, the LRB algorithm is constructed using a two-phase approach—It is split into "pure exploration phases" (also called the first phase) and "targeted exploration + exploitation phases" (also called the second phase). This approach partially builds on the ideas in Goldenshluger and Zeevi (2013) that are later extended by Hamidi et al. (2019) and Bastani and Bayati (2020). Specifically, those studies centered around the contextual bandit problem. They build two sets of estimators (namely, forced-sample and all-sample estimators) in such a way that the former estimator is used to construct a targeted set that filters suboptimal arms that fall outside the confidence region of the optimal reward, and the latter estimator is used to select the best arm in the targeted set. The first step that is a pure exploration procedure aims at ensuring that, during a constant proportion of time periods, only a single arm is left in the targeted set, so that the algorithm can collect independent and identically distributed (i.i.d.) samples in the second phase, despite that the second phase is pure exploitation.

However, in our study, the two-phase procedure serves a different purpose. More specifically, in the first phase, we utilize the low-rank structure to filter suboptimal arms that are $h/2$ away from the highest estimated reward by forced-sample estimators. Although our first step seems similar to the aforementioned papers, our second phase is itself a UCB algorithm that both explores and exploits the targeted set to find the optimal arm. This distinction results in a different requirement for the first phase: The targeted set does not need to contain just one element; rather, it only needs to be much smaller than $d_r d_c$ to meaningfully reduce the exploration cost of a regular UCB algorithm. This fact results in different selection criteria for the filtering resolution h. As we discuss in detail later, the optimal value of h is determined by both the matrix size and the total time horizon.

Furthermore, we want to emphasize that the two-phase approach in the aforementioned studies cannot be directly applied to our problem. They focused on long enough experimentation time horizons so that their algorithms can take time to learn arms well and leave only the optimal choice in the targeted set for some fixed h. However, as aforementioned, we face a short horizon under the many-arm circumstances. In our analysis, we will show that the filtering resolution h controls not only the size of the targeted set, but also the estimation error of using the forced samples. We need smaller h to filter out more arms to have a smaller

targeted set, which in turn requires more forced-sampling rounds to achieve a good enough estimation to be more confident about the suboptimality of the filtered arms. Thus, it is unlikely to construct a targeted set with fine-enough filtering resolution that only contains the optimal choice as the single arm, because forced-sample estimation is not accurate enough under limited experimentation time to achieve that.

Finally, we note that one cannot use the low-rank estimator to construct confidence sets for the UCB algorithm because existing bounds for the low-rank estimators do not work in the presence of adaptively collected data. The only exception (to the best of our knowledge) is the all-sampling estimator of Hamidi et al. (2019), but as discussed above, their approach would not work when the targeted set has more than one arm.

3.1. Description of the LRB Algorithm

The main idea of our two-phase LRB algorithm is to reduce the initially large set of arms based on a low-rank estimator and pass a smaller set of arms to a UCB algorithm. Specifically, we *forced-sample* arms at prescribed times and use them to train a low-rank estimator of $\mathbf{B}^*$, which has a sharp entry-wise error bound. Then we use the entry-wise estimations to select a targeted set of arms which, with high probability, contains the optimal arm.

The LRB takes as input the matrix row and column dimensions d_r and d_c, the forced-sampling rule $f_\cdot$, and a filtering resolution $h > 0$ (discussed further in Section 3.1.1 and Section 3.3.1). The complete procedure is in Algorithm 1. We highlight the key components of our algorithm below.

3.1.1. Forced-Sampling Rule. Our forced-sampling rule is $f : \mathbb{N}^+ \to ([d_r] \times [d_c]) \cup \{\emptyset\}$. At time t, the forced-sampling rule decides between forcing the arm $f_t \in [d_r] \times [d_c]$ to be pulled or exploiting the past data, indicated by $f_t = \emptyset$. We use $\mathcal{F}_t$ to denote the set of time periods when an arm is forced to be pulled up to time t, that is, $\mathcal{F}_t := \{t' \le t : f_{t'} \in [d_r] \times [d_c]\}$. The forced-sampling rule that we use is a randomized function that picks an arm $(j,k) \in [d_r] \times [d_c]$ with probability

$$\mathbb{P}(f_t = (j,k)) = \begin{cases} \dfrac{1}{d_r d_c} & \text{if } t \le 2\rho\log(\rho), \\ \dfrac{\rho}{d_r d_c[t - \rho\log(\rho) + 1]} & \text{if } t > 2\rho\log(\rho), \end{cases} \tag{3.1}$$

and $f_t = \emptyset$ otherwise. We call ρ the forced-sampling parameter and $2\rho\log(\rho)$ the initialization period, which we denote as t_0. In Lemma EC.1 shown in Section EC.1 in the Online Appendix, we prove that the number of forced-sampling rounds $|\mathcal{F}_t|$ is of order $O(\rho\log(t))$ with high probability. The choice of ρ will be discussed in Section 3.1.1.

3.1.2. Estimators. At any time t, LRB maintains two sets of parameter estimates for $\mathbf{B}^*$:

1. The forced-sample estimates $\hat{\mathbf{B}}^F$ based only on forced samples.
2. The UCB estimator $\hat{\mathbf{B}}^{UCB}_{jk}$ based on all samples observed for each arm (j,k), where $\hat{\mathbf{B}}^{UCB}_{jk} = \overline{x}_{(j,k)} + \sqrt{2\log w(t)/n_{(j,k)}}$ is a combination of the empirical mean reward estimation $\overline{x}_{(j,k)}$ and a term $\sqrt{2\log w(t)/n_{(j,k)}}$ that quantifies the uncertainty of the estimates, $n_{(j,k)}$ is the number of times entry (j,k) has been played so far, and the function $w(t)$ is defined by $w(t) = 1 + t\log^2(t)$ for the empirical experiment and $w(t) = t$ for the analysis part for simplicity.

3.1.3. Low-Rank Estimator. As mentioned before, our forced-sample estimator is based on a low-rank estimator. We use nuclear norm penalized least squares as our example for a low-rank estimator.

Definition 3 (Nuclear Norm Penalized Least Squares). Given a regularization parameter $\lambda \ge 0$, the nuclear norm penalized least square estimator is

$$\hat{\mathbf{B}} = \underset{\mathbf{B} \in \mathbb{R}^{d_r \times d_c}}{\arg\min} \left(\frac{1}{|\mathcal{F}_t|} \|Y - \mathfrak{X}^{\pi}_{\mathcal{F}_t}(\mathbf{B})\|_2^2 + \lambda \|\mathbf{B}\|_* \right).$$

However, any low-rank estimator that satisfies the following tail bound can be used as our forced-sample estimator. For simplicity of presentation, we set $d_r = d_c = d$.

Proposition 1 (Tail Bound for Low-Rank Estimators (Chen et al. 2020)). *Fix any $\varsigma > 0$. By taking $\lambda = C_\lambda \sigma \sqrt{1/(nd)}$ where n denotes the number of i.i.d. samples which are sampled uniformly for some large enough constant $C_\lambda > 0$, it holds that*

$$\mathbb{P}\left(\|\hat{\mathbf{B}} - \mathbf{B}^*\|_\infty \ge C_\varsigma \sqrt{\kappa^3 \mu r} \frac{\sigma}{\sigma_{\min}} \sqrt{\frac{d\log d}{n}} \right) \le \frac{1}{d^\varsigma},$$

when $n \ge C\kappa^4 \mu^2 r^2 d\log^3 d$, and C and C_ς are positive constants which are independent of d and n.

Remark 1. Chen et al. (2020) provide the tail bound for square matrices to facilitate a clear presentation. In their paper, they note that it is straightforward to extend their discussions to general rectangular matrices of size $d_r \times d_c$. Similarly, we focus on square matrices for simplicity of presentation, whereas our proposed framework is not limited to square matrices. Given a tail bound for rectangular low-rank matrices of $d_r \times d_c$, all subsequent discussions can be easily extended accordingly. In addition, theorem 1 in Chen et al. (2020) proves for $\varsigma = 3$, but their result can be easily generalized to any large enough $\varsigma \ge 3$.

3.1.4. Execution. We follow a *two-phase* process: If the current time t is in $\mathcal{F}_t$, then we are in the "pure exploration phases," and arm f_t is played; otherwise, we are in the "targeted exploration + exploitation phases," and this phase consists of the following two steps. As a preprocessing step, we use the forced-sample estimates to find the highest estimated reward achievable across all $d_r d_c$ entries. We select a subset of arms $\mathcal{C} \subset [d_r] \times [d_c]$ whose estimated rewards are within $h/2$ of the maximum achievable (in step 6 of Algorithm 1). After this preprocessing step, we apply UCB on the targeted set $\mathcal{C}$ by selecting the arm that maximizes the UCB value (in step 7 of Algorithm 1).

Algorithm 1 (LRB)

1: Input: Matrix dimensions d_r, d_c, forced-sampling rule f, filtering resolution h
2: **for** $t = 1, 2, \ldots, T$ **do**
3: **if** $f_t \neq \emptyset$ **then** "pure exploration phase"
4: $\pi_t \leftarrow f_t$ (forced-sampling)
5: **else**"targeted exploration + exploitation phases"
6: Define the set of near-optimal arms $\mathcal{C}$ according to the forced-sample estimator:

$$\mathcal{C} = \Big\{ (j,k) \in [d_r] \times [d_c] \,|\, \hat{\mathbf{B}}^F_{jk}(\mathcal{F}_{t-1}) \geq \max_{(l,m)\in[d_r]\times[d_c]} \hat{\mathbf{B}}^F_{lm}(\mathcal{F}_{t-1}) - \frac{h}{2} \Big\}. \quad (3.2)$$

7: Choose the best arm within $\mathcal{C}$ according to the UCB estimator, that is,

$$\pi_t \leftarrow (j_t, k_t) = \underset{(j,k)\in\mathcal{C}}{arg\ max}\ \hat{\mathbf{B}}^{UCB}_{jk},$$

where $\hat{\mathbf{B}}^{UCB}_{jk} = \overline{x}_{(j,k)} + \sqrt{\frac{2\log w(t)}{n_{(j,k)}}}$ is such that $\overline{x}_{(j,k)}$ is the average observed value for entry (j,k), and $n_{(j,k)}$ is the number of times entry (j,k) has been played so far.
8: **end if**
9: Play arm $\pi_t = (j_t, k_t)$, observe $y_t = \mathbf{B}^*_{j_t k_t} + \epsilon_t$.
10: **end for**

3.1.5. Filtering Resolution *h*. In our preprocessing step described above, we used h to filter the arm set as well as control how accurate the forced-sample estimator needs to be. As mentioned earlier, smaller h includes fewer arms in the targeted set, and thus more accurate forced-sample estimation is needed to have the optimal arm in the targeted set. Vice versa, bigger h leads to a larger targeted set that allows for more crude forced-sample estimation.

Recall that the parameter ρ controls the quality of the forced-sample estimator. As the next lemma shows, the smaller h is, the bigger ρ needs to be to ensure more forced samples. Its proof (in Section EC.1 in the Online Appendix) builds on Proposition 1 and Lemma EC.1 (in Section EC.1, which characterizes how many forced samples we can have given the forced-sampling parameter ρ). In our subsequent analysis, we assume $T = d^\beta$ for some $\beta \in \mathbb{R}$, thus smaller values of β correspond to shorter horizons.

Lemma 1. *For all* $t \geq 2\rho\log(\rho)$ *where* $\rho \geq 160\max\{C, C_\varsigma^2\}\kappa^4\mu^2 r^2(\sigma/\sigma_{\min})^2 d\log^3(d)h^{-2}$ *with* $\varsigma = 3\beta$ *where* $T = d^\beta$, *it holds with probability at least* $1 - 2t^{-3}$ *that,* $\|\hat{\mathbf{B}}^F - \mathbf{B}^*\|_\infty \leq h/4$.

The above result states that the forced-sample estimator $\hat{\mathbf{B}}^F$ satisfies $\|\hat{\mathbf{B}}^F - \mathbf{B}^*\|_\infty \leq O(1)$ with probability at least $1 - O(1/t)$ for all arms when the forced-sampling parameter ρ is big enough. To this end, we set

$$\rho(h;d) = \alpha r^2 d\log^3(d)h^{-2}, \quad (3.3)$$

where $\alpha = 160\max\{C, C_\varsigma^2\}\kappa^4\mu^2(\sigma/\sigma_{\min})^2$, so that the sample size requirement for the low-rank estimator is satisfied and we do not overexplore. Based on the forced-sampling parameter $\rho(h;d)$ derived in Equation (3.3), we need $T \geq d$ (i.e., $\beta \geq 1$) to make the low-rank estimator effective. We discuss the case when $\beta < 1$ in Section 4. For ease of notation, we use $G(\cdot)$ to refer to the event $G(\mathcal{F}_t) := \{\|\hat{\mathbf{B}}^F(\mathcal{F}_{t-1}) - \mathbf{B}^*\|_\infty \leq h/4\}$.

***3.1.5.1. Range of* h.** Filtering resolution h has a selection range. The upper bound of this range is controlled by the range of the arm rewards (entries of $\mathbf{B}^*$) and is one per Assumption 1. To derive the lower bound, we notice that, given the time horizon T, we have $O(\rho\log T)$ number of forced samples per Lemma EC.1 in the Online Appendix. Because $\rho\log\rho \leq T/2$ is a required condition for Lemma EC.1, we can let ρ be such that $\rho\log T \leq T/2$ to derive the minimum value that h can take, denoted by $\underline{h}(T)$:

$$\underline{h}(T) = \sqrt{\frac{2\alpha r^2 d\log^3 d\log T}{T}}. \quad (3.4)$$

This shows that $\underline{h}(T)$ is of order $\Omega(\sqrt{d/T})$.

We notice that the range of h gets bigger with the time horizon, because $\underline{h}(T)$ decreases in the time horizon. Hence, for shorter time horizons, the best achievable accuracy of the low-rank estimator is less refined than that for longer time horizons, but more arms will be included in the targeted set to ensure that high-reward arms are not filtered out due to a lower quality estimation. Such intuition is formalized in Proposition 2 in Section 3.3.1, where we examine the optimality of h. We discuss further about the range of h when comparing our bound to the existing literature in Section 3.3.2.

3.1.5.2. Near-Optimal Arms. An important step in our regret analysis is to show the following: When the estimation is accurate enough, the targeted set will be a subset of "near-optimal" arms, which we define below

as a function of h. Notation-wise, we use $\mathcal{S}^h_{opt}(\mathcal{I}_r, \mathcal{I}_c)$ to denote the near-optimal set of arms parameterized by an arbitrary h, where $\mathcal{I}_r \subseteq [d_r]$ denotes a subset of row indices and $\mathcal{I}_c \subseteq [d_c]$ denotes a subset of column indices; $\mathcal{I}_r \times \mathcal{I}_c \subseteq [d_r] \times [d_c]$ denotes the set of entries of the submatrix constructed from $\mathcal{I}_r$ and $\mathcal{I}_c$.

Definition 4 (Near-Optimal Set). We define the near-optimal set as the following:

$$\mathcal{S}^h_{opt}(\mathcal{I}_r, \mathcal{I}_c) = \{(j,k) \in \mathcal{I}_r \times \mathcal{I}_c \,|\, \mathbf{B}^*_{jk} \geq \max_{(j',k') \in \mathcal{I}_r \times \mathcal{I}_c} \mathbf{B}^*_{j'k'} - h\}.$$

That is, it contains elements in $\mathcal{I}_r \times \mathcal{I}_c$ that are at most h smaller than the best arm in $\mathcal{I}_r \times \mathcal{I}_c$.

We next introduce the near-optimal function $g(h; \mathcal{I}_r, \mathcal{I}_c)$, which characterizes the cardinality of the near-optimal set.

Definition 5 (Near-Optimal Function). The near-optimal function $g(h; \mathcal{I}_r, \mathcal{I}_c) = |\mathcal{S}^h_{opt}(\mathcal{I}_r, \mathcal{I}_c)|$ denotes the number of elements that are at most h smaller than the best arm in $\mathcal{I}_r \times \mathcal{I}_c$.

In such cases, for brevity, we often use shortened notation $\mathcal{S}^h_{opt}$ and $g(h)$ for $\mathcal{S}^h_{opt}([d_r], [d_c])$ and $g(h; [d_r], [d_c])$, respectively. The more general case with $\mathcal{I}_r \times \mathcal{I}_c \neq [d_r] \times [d_c]$ will be considered in Section 4. When entries of $\mathbf{B}^*$ are sampled from certain distributions, we can describe the expected value of the near-optimal function in a closed form (see Section EC.6 in the Online Appendix). We also provide empirical examples, on the near-optimal functions for low-rank matrices in Section EC.7 in the Online Appendix.

The following lemma shows that a targeted set is a subset of the near-optimal set when the forced-sample estimation is accurate enough. Its proof is in Section EC.2.3.3 in the Online Appendix.

Lemma 2. *If $G(\mathcal{F}_{t-1}) = 1$, then the largest entry (j^*, k^*) belongs to the targeted set $\mathcal{C}$ defined in* Equation (3.2) *of Algorithm 1. Furthermore, $\mathcal{C} \subseteq \mathcal{S}_{opt}$.*

Our next section provides the regret analysis of the LRB algorithm.

3.2. Regret Analysis of LRB

We present both a *gap-dependent* bound and a *gap-independent* bound on our LRB algorithm. The key idea of our regret analysis is to decompose the total regret into the regret incurred from the pure exploration phase and that incurred from the targeted exploration + exploitation phase, and bound each, respectively. The *gap-dependent bound* depends on the term Δ, which is defined to be the difference between the biggest element and the second biggest element of $\mathbf{B}^*$, that is, $\Delta := \min_{(j,k) \in [d_r] \times [d_c]} \{(\mathbf{B}^*_{j^*k^*} - \mathbf{B}^*_{jk}) \mathbb{I}[\mathbf{B}^*_{j^*k^*} - \mathbf{B}^*_{jk} > 0]\}$, where $\mathbb{I}[\cdot]$ is the indicator function. The independent bound does not depend on Δ^{-1} which would be helpful when Δ is very small. We present the main results below and relegate the detailed derivations to Section EC.2 in the Online Appendix.

Theorem 1 (Gap-Dependent Bound). *For any h such that $h \geq \underline{h}(T)$, the cumulative regret of Algorithm 1, represented by* $\mathsf{Regret}_T(h)$, *is upper bounded by*

$$\mathsf{Regret}_T(h) \leq C_1 \mathrm{r}^2 d \log^3(d) h^{-2} \log T + \min\left\{hT, \frac{8g(h)\log T}{\Delta} + \left(1 + \frac{\pi^2}{3}\right) h g(h)\right\} + C_2, \tag{3.5}$$

where $C_1 = b^* \alpha(2t_0 + 12)$ and $C_2 = 100 b^*$.

Theorem 2 (Gap-Independent Bound). *For any $h \geq \underline{h}(T)$, the cumulative regret at time T, represented by* $\mathsf{Regret}_T(h)$, *is upper bounded by*

$$\mathsf{Regret}_T(h) \leq C_1 \mathrm{r}^2 d \log^3(d) h^{-2} \log T + \min\left\{hT, 8\sqrt{2Tg(h)\log T}\right\} + C_2,$$

where $C_1 = b^* \alpha(2t_0 + 12)$ and $C_2 = 100 b^*$.

As emphasized earlier, we use low-rank estimators rather than the empirical mean of the arms to select the targeted arms in the first phase so that we can learn the arms faster and keep the regret low. Indeed, to ensure that the optimal arm is selected into the targeted set, we need to control the estimation error to be within $O(h)$ (Lemma 2). When there are d^2 number of arms, low-rank estimators only use $\tilde{O}(\mathrm{r}^2 d/h^2)$ number of explorations to achieve the aforementioned estimation accuracy (shown earlier by Lemma 1), whereas empirical mean estimator needs $\tilde{O}(d^2/h^2)$ number of explorations. This is because empirical mean estimator for each arm only focuses on information for that particular arm, ignoring helpful information from other arms. We elaborate further on this point in Remark 2 in Section 3.3.2 after more in-depth analysis.

3.3. Balancing Regrets of Two Phases with h

Our algorithm can adjust the filtering resolution h to balance the cost of forced-sampling and the cost of targeted exploration + exploitation after the filtering. Finer filtering resolution incurs more costs in the first phase due to more forced-sampling rounds to achieve more accurate estimates but alleviates the burden in the second phase as more suboptimal arms have been filtered out. Vice versa, less refined filtering resolution results in a larger targeted set and thus incurs more costs in the second phase, but saves costs in the first phase.

To explain further mathematically, recall the regret decomposition we have discussed in the beginning of Section 3.2. According to the proof of Theorem 2 in

Section EC.2 in the Online Appendix (we use the gap-independent regret analysis to illustrate), the regret from the pure exploration phase can be bounded above by ϕ_1 and the regret from the targeted exploration + exploitation phase can be bounded above by ϕ_2, where

$$\phi_1(h;d,t) = C_1 \mathfrak{r}^2 d \log^3(d) h^{-2} \log(t) \quad \text{and}$$
$$\phi_2(h;d,t) = \min\{8\sqrt{2tg(h)\log(t)}, ht\}.$$

Note that we ignore the constant term C_2 for ϕ_2 above (and in subsequent expressions we omit this term for simplicity). The upper bound of the regret, $\overline{\mathsf{Regret}}_T(h)$, can be expressed as

$$\overline{\mathsf{Regret}}_T(h) = \phi_1(h;d,T) + \phi_2(h;d,T) \geq \mathsf{Regret}_T(h). \tag{3.6}$$

Note that the quantity $\phi_1(h)$ increases in h. Indeed, as described in Lemma 1, the bigger the h, the fewer rounds of forced-sampling are needed to achieve the forced-sample estimation accuracy. Consequently, less regret is incurred in the first phase which leads to smaller $\phi_1(h)$. The quantity $\phi_2(h)$ depends on h via the near-optimal function $g(h)$, which is nondecreasing in h. This is because more arms may be included in the targeted set as h gets bigger. Consequently, more regret is incurred in the second phase to find the best arm in the targeted set, which makes $\phi_2(h)$ bigger. Next, we discuss how to choose h optimally.

3.3.1. Optimality of h. Based on the interaction between ϕ_1 and ϕ_2, we characterize the best filtering resolution that optimizes the regret bound in the following lemma.

Lemma 3 (Optimal Filtering Resolution). *Given the total time horizon T, let the optimal h be defined as $h^* = \arg\min_{h\in[\underline{h}(T),2b^*]} \overline{\mathsf{Regret}}_T(h)$. We consider the following three cases:*

1. *If $\phi_1(h;d,T) < \phi_2(h;d,T)$ for any $h \in [\underline{h}(T), 2b^*]$, then $\tilde{h} = \underline{h}(T)$ is optimal up to a factor of two, that is, $\overline{\mathsf{Regret}}_T(\tilde{h}) \leq 2\overline{\mathsf{Regret}}_T(h^*)$.*
2. *If $\phi_1(h;d,T) = \phi_2(h;d,T)$ for some $h = \tilde{h} \in [\underline{h}(T), 2b^*]$, then $\tilde{h}$ is optimal up to a factor of two.*
3. *If $\phi_1(h;d,T) > \phi_2(h;d,T)$ for any $h \in [\underline{h}(T), 2b^*]$, then $\tilde{h} = 2b^*$ is optimal up to a factor of two.*

Similar arguments hold for the gap-dependent bound in (3.5).

The proof of Lemma 3 can be found in Section EC.3 in the Online Appendix. Although Lemma 3 only analyzes the regret *upper bounds* ϕ_1 and ϕ_2 rather than the actual regret of the two phases, we empirically show that the actual regret follows the same behavior in numerical simulations. In Section EC.5 in the Online Appendix, we plot the cumulative regret from the two phases for low-rank matrices that are 100×100 with rank 3 with respect to a sequence of h, and observe that $h = 1$ with 225 number of forced samples gives the lowest cumulative regret. Further, the regret curves of the two phases cross at around $h = 1.4$, and the cumulative regret at $h = 1.4$ is within a factor of two of the lowest cumulative regret.

We discussed in Section 3.1.1 that the range of h becomes bigger in the time horizon T due to the functional form of $\underline{h}(T)$. Next, we show that the optimal filtering resolution in fact monotonically decreases in the time horizon T.

Proposition 2. *The optimal filtering resolution $\tilde{h}$ monotonically decreases in the time horizon T when $T \geq 10$.*

The proof of Proposition 2 is in Section EC.3 in the Online Appendix. This proposition aligns with our earlier finding that finer filtering resolution is achieved over longer time horizons.

Next, we show that our upper regret bounds improve on those in the existing literature by balancing ϕ_1 and ϕ_2 through h.

3.3.2. Benefits of Low-Rank Structure and the Targeted Exploration + Exploitation. We compare the gap-independent bound in Theorem 2 against the upper regret bounds for standard noncontextual bandits to show that our bound achieves better rates in Proposition 3. This analysis also holds for comparison against gap-dependent bounds in the literature.

From this section, we focus on the order of d and T and treat $\mathfrak{r}$ as a constant when we discuss the regret order because $\mathfrak{r} \ll d$. We first show in Proposition 3(1) that our regret bound under any time horizon is no worse than the regret bound $\min\{O(T), \tilde{O}(d\sqrt{T})\}$ of those standard noncontextual bandit algorithms (e.g., UCB), which would treat each entry as an independent arm. Intuitively, when $T \geq d^2$, the regret bound takes the form $\tilde{O}(d\sqrt{T})$ because $T \geq d\sqrt{T}$. When $T \leq d^2$, because standard noncontextual bandits like UCB need to randomly sample each of $d_r d_c$ arms at least once to initialize, it will incur $O(T)$ regret. Secondly, we show in Proposition 3(2) how to optimize for the choice of h to achieve a strictly smaller regret upper bound under time horizons $d^2 \leq T < d^4$.

Proposition 3. *Recall that $T = d^\beta$. For any structure of $g(h)$:*

1. *For all values of β, Algorithm 1 achieves no worse regret than $\min\{O(T), \tilde{O}(d\sqrt{T})\}$.*
2. *In particular, when $2 \leq \beta < 4$, Algorithm 1 can achieve regret of order $o(d\sqrt{T})$. More specifically, it can achieve the regret of order $\tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})$.*

The proof of Proposition 3 is relegated to Section EC.3.2 in the Online Appendix. We want to emphasize

that the improvement of our algorithm stems from leveraging the low-rank structure. Without assuming any special structure, no algorithm is able to achieve a lower regret than the regret lower bound of $d\sqrt{T}$ of the standard noncontextual bandit algorithms like UCB. Below, we further discuss how we benefit from a low-rank estimator, in comparison with using the empirical mean estimator.

Remark 2 (Benefit of Low-Rank Estimator). As aforementioned, without the low-rank structure, we need d^2/h^2 rounds to achieve an optimality gap of h for d^2 entries with an empirical mean estimator. As a result, the order of regret bound on the first phase increases from $\mathrm{r}^2 d$ to d^2 (ignoring the logarithmic order), and the gap-independent bound in Theorem 2 thus becomes $C_1' d^2 h^{-2} + \min\{hT, 8\sqrt{2Tg(h)}\}$. Adapting the analysis for Proposition 3 to the empirical mean estimator, we obtain a regret order of $\tilde{O}(d^{\frac{2}{3}}T^{\frac{2}{3}})$, which is strictly higher than $\tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})$ derived in Proposition 3 with the low-rank estimator used for the first phase.

In Proposition 3, we have not yet considered any structure of $g(h)$ in our discussion and have only worked with a very loose bound of $g(h) \le d^2$ for any h to show how the sole use of low-rank estimation and the element-wise error bound already helps with reducing regret upper bound of the existing literature.

We illustrate how to further tighten the regret bound by leveraging the structure of $g(h)$ determined by its shrinkage rate ζ defined below.

Definition 6 (Shrinkage Rate of Near-Optimal Functions). The shrinkage rate of a near-optimal function $g(h;\mathcal{I}_r,\mathcal{I}_c)$ is $\zeta = \sup\{\zeta' \ge 0 \,|\, \mathbb{E}_{|\mathcal{I}_r|=|\mathcal{I}_c|=m}[\sqrt{g(h;\mathcal{I}_r,\mathcal{I}_c)}] \le mh^{\zeta'/2},\ \forall m \le d\}$.

The inequality in the set constraint naturally holds when $\zeta = 0$ because $g(h;\mathcal{I}_r,\mathcal{I}_c) \le m^2$. Because $h \le 1$, then $g(h;\mathcal{I}_r,\mathcal{I}_c)$ shrinks faster for larger values of ζ, which implies that the low-rank structure can filter out more arms. When $\mathcal{I}_r = [d] = \mathcal{I}_c$, Definition 6 yields $g(h;[d],[d]) \le d^2 h^{\zeta}$.

Theorem 3 then works with Definition 6 to give a tighter regret bound for our algorithm. We defer its proof to Section EC.3.2 in the Online Appendix.

Theorem 3. *For* $\vartheta = (\frac{2\sqrt{2}\zeta(2\alpha\mathrm{r}^2)^{\frac{\zeta+4}{4}}}{C_1\mathrm{r}^2})^{\frac{4}{\zeta+2}}$, *it holds that*

1. *When* $T \ge \vartheta d^{\frac{\zeta+4}{\zeta+1}}$, $\mathrm{Regret}_T = \tilde{O}(dT^{\frac{2}{\zeta+4}})$; *and*
2. *when* $T \le \vartheta d^{\frac{\zeta+4}{\zeta+1}}$, $\mathrm{Regret}_T = \tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})$.

We next present Example 3 in which we plug in empirical estimated values of ζ to show more concrete improvement.

Example 3. Let the low-rank matrices $\mathbf{B} = \mathbf{U}\mathbf{V}^\top$, where the entries of $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{d\times \mathrm{r}}$ follow uniform distribution on [0,1]. From the empirical evidence in Section EC.7 in the Online Appendix, $g(h) \le d^2h^3$ when $d = 100$, that is, the shrinkage rate $\zeta \approx 3$. Then from Theorem 3, when $T \ge \vartheta d^{\frac{7}{4}}$, the regret is of order $\tilde{O}(dT^{\frac{2}{3}})$, which is lower than $\min\{\tilde{O}(d\sqrt{T}), T\}$; when $d \le T \le \vartheta d^{\frac{7}{4}}$, the regret is of order $\tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})$, lower than the regret needed by standard noncontextual bandit like UCB, which is at least $\min\{d^2, T\}$, incurred from pulling each arm once when $T \le d^2$.

Remark 3. The above example shows that we can construct the empirical $g(h)$ and thus fit ζ if we have access to the reward distribution from abundant related historical data. Consequently, we can obtain the optimal h via the recipe provided in the proof of Theorem 3. We elaborate on a more data-driven approach for obtaining the optimal h in Section 5.2, where we use a prespecified grid to search for optimal values based on a historical data set.

Remark 4 (Benefit of Targeted Exploration + Exploitation). We highlight the benefit of our targeted exploration + exploitation in the second phase by comparing against a naive ETC approach that we call low-rank ETC, based on the results from Theorem 3. As the name suggests and similar to ours, low-rank ETC undergoes the "pure exploration" phases and uses the forced samples to construct a low-rank estimator. However, unlike ours, it commits to the best estimated entry in the second phase rather than conducting targeted exploration + exploitation. Consequently, the regret upper bound on the second phase changes from $\min\{8\sqrt{2Tg(h)\log(T)}, hT\}$ to hT, resulting in a potentially higher gap-independent regret bound of $C_1\mathrm{r}^2 d\log^3(d)h^{-2}\log T + hT$. In particular, when $g(h)$ has a nice structure (i.e., $g(h)$ is small), our approach is strictly better. As shown in Theorem 3(1), the regret of our LRB achieves an order of $\tilde{O}(dT^{\frac{2}{\zeta+4}})$ for $T \ge \vartheta d^{\frac{\zeta+4}{\zeta+1}}$, which improves from low-rank ETC, whose regret is of order $\tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})$. A visualization of such a comparison in upper bounds can be found in Section 4.3. We additionally show the superior empirical performance of our approach against low-rank ETC in Section EC.10 in the Online Appendix.

3.3.3. Effectiveness of LRB Under Different Horizons. Proposition 3, Theorem 3, and Example 3 bring an important implication: Our algorithm LRB can adapt to different time horizons, even when the time horizon is short.

When $T > d^4$, Proposition 3 shows that our algorithm performs no worse than those standard noncontextual bandits, and we call this time regime the *long horizon*. Furthermore, when we know the near-optimal function $g(h)$ as in Example 3, LRB is strictly better in the long horizon.

When $d^2 \le T < d^4$, our algorithm achieves a regret upper bound that is strictly better than that of standard noncontextual bandits even without specifying $g(h)$ (as shown in Proposition 3), and we call such a time regime the *short horizon*. In such a *short horizon* regime, we have seen that LRB no longer sticks with the pure exploration phase but activates the targeted exploration + exploitation phase with a proper selection of the filtering resolution h (shown in the proof of Proposition 3 in Section EC.3.2 in the Online Appendix). Such flexibility of our algorithm can be very useful, because in real-world practices, the product set size and the experimentation time vary a lot by applications. Although some have a small product set and the experimentation can effectively occur over a long time horizon, others have a huge product set and the experimentation needs to be completed in a very short time.

The remaining time horizons, that is, $T \le d^2$, will be called the *ultra-short* regime. In this regime, UCB is known to only perform forced sampling (i.e., pure exploration). Our algorithm LRB also only does forced sampling when $T \le d$. When $d \le T \le d^2$, Example 3 shows that our algorithm LRB can perform better than pure exploration when $g(h)$ has a good structure. However, as in the general case shown in Proposition 3, although our algorithm is no worse than standard noncontextual bandits, a strictly better performance is not guaranteed under this regime. We thus propose a subsampling strategy for enhancing the performance when the time horizon is very limited in Section 4. Additionally, we show in Theorem 5 and Example 4 that the subsampling variation can be effective even under longer periods beyond the ultra-short horizons.

4. LRB with Submatrix Sampling

In this section, we propose to perform a subsampling prestep that can potentially incur less cost than a regular LRB policy when the horizon is (and possibly beyond) ultra-short (defined in Section 3.3.3). Under such a horizon, subsampling allows our regular LRB to leverage the low-rank structure of a smaller submatrix to effectively filter out enough suboptimal arms. Building on Section 3.3.2, we summarize and visualize the benefits of low-rank structure, targeted exploration + exploitation, and subsampling at the end of this section.

4.1. Description of the Submatrix-Sampled Low-Rank Bandit Algorithm

The subsampling prestep samples a submatrix of $m_r \times m_c$, and our LRB policy is used as a subroutine to explore and exploit the low-rank structure of that submatrix. We call this subsampling version of LRB the submatrix-sampled LRB (ss-LRB), and the specific procedures are summarized in Algorithm 2.

Algorithm 2 (ss-LRB Algorithm)

1: Decide the submatrix size m_r, m_c as a function of $d_r, d_c, \mathfrak{r}, T$.

2: Draw a set of m_r number of row indices, denotes as $\mathcal{I}_r$, and a set of m_c number of column indices, denoted as $\mathcal{I}_c$, uniformly at random (without replacement) from $[d]$, respectively.

3: Run LRB algorithm (Algorithm 1) on the submatrix indexed by $(\mathcal{I}_r, \mathcal{I}_c)$.

For simplicity of presentation, we set $m_r = m_c = m$, but we can optimize for selecting nonsquare submatrices. Note that ss-LRB only estimates the submatrix rather than the entire matrix, which is a much easier and quicker task if the submatrix dimension is small (because it has smaller sample complexity). We further notice that the forced-sampling parameter in ss-LRB becomes $\rho(h;m) = \alpha r^2 m \log^3(m) h^{-2}$, and the lower bound for h becomes $\underline{h}(T) = \sqrt{2\alpha r^2 m \log^3(m) \log T / T}$ according to Equation (3.4), such that $\underline{h}(T)$ is of order $\tilde{\Omega}(\sqrt{m/T})$ and the range of h becomes larger. Thus, the selection range of h expands. Applying Lemma 3 and deploying a similar proof strategy as for Proposition 2, we can conclude that, for a fixed time horizon, the optimal filtering resolution can be finer for a subsampling scheme with a smaller submatrix size, thus leading to a more accurate low-rank estimation.

Our motivation comes from Bayati et al. (2020), who suggest that subsampling may help when the time horizon is ultra-short. Their work states that, when the number of arms $k \ge \sqrt{T}$, it is optimal to sample a subset of $\Theta(\sqrt{T})$ arms and execute UCB on that subset. However, directly adopting random subsampling from Bayati et al. (2020) will demolish the helpful matrix structure. To preserve the potential advantage of working with a low-rank matrix, we integrate a slightly different subsampling strategy into our LRB, which is the aforementioned submatrix sampling.

Noticeably, we cannot ensure that the biggest element in this submatrix is necessarily the biggest element of the entire matrix. If not, then in each period, we would at least incur a subsampling cost defined as the difference between the biggest element of the entire matrix and the biggest element of a random submatrix. Such a subsampling cost is a function of the matrix size, the submatrix size and the rank of the full matrix.

Definition 7 (Subsampling Cost). Let $\mathcal{I}_r$ ($\mathcal{I}_c$, respectively) be an index set of i, where i is drawn uniformly from $[d]$. The subsampling cost function is defined as

$$\psi(m;d) := \max_{j,k} \mathbf{B}^*_{jk} - \mathbb{E}_{\mathcal{I}_r,\mathcal{I}_c:|\mathcal{I}_r|=|\mathcal{I}_c|=m}\Big[\max_{(j',k')\in\mathcal{I}_r\times\mathcal{I}_c} \mathbf{B}^*_{j'k'}\Big].$$

When the arm rewards follow certain common distributions, the subsampling cost function has closed forms, which is presented in Section EC.6 in the

Online Appendix. We also provide empirical evidence for how the subsampling cost functions look like for low-rank matrices in Section EC.7 in the Online Appendix.

4.2. Regret Analysis Under Submatrix Sampling

In this section, we first give a general regret upper bound analysis for ss-LRB in Theorem 4. Then, we impose Assumption 2 on the subsampling cost function to derive further results. This assumption applies to matrices whose entries follow common distributions and is supported by empirical evidence for low-rank matrices. We show in Theorem 5 and Example 4 that ss-LRB indeed achieves a smaller expected cumulative regret upper bound than the regular LRB under a possibly big range of time horizons. That is, even if the biggest entry in the submatrix might be suboptimal, the cost we save from not exploring the much larger matrix compensates for the regret incurred from picking a suboptimal entry. As a result, such a submatrix sampling procedure may be desirable beyond the ultra-short horizons, and we specify when that would be in our subsequent analysis.

Recall that $\phi_1(h;m,t) = C_1 r^2 m \log^3(m) h^{-2} \log t$, where $C_1 > 0$ is a constant. Similarly, we define

$$\phi_2(h;m,t) = \min\left\{8\sqrt{2t\log t} \cdot \mathbb{E}_{|\mathcal{I}_r|=|\mathcal{I}_c|=m}\left[\sqrt{g(h;\mathcal{I}_r,\mathcal{I}_c)}\right], ht\right\},$$

where the expectation is taken over all submatrices with both the number of rows and columns equal to m. For a fixed m, we use h^*_m (which is characterized in Lemma 3) to denote the optimal h that minimizes the regret upper bound relative to the largest entry in the submatrix (not considering the subsampling cost), which is defined in Equation (3.6) as $\overline{\mathsf{Regret}}_T(h;m) = \phi_1(h;m,T) + \phi_2(h;m,T)$. The following theorem gives the gap-independent bound for our algorithm with submatrix sampling.

Theorem 4. *The total regret of* ss-LRB*, represented by* $\mathsf{Regret}_T(h^*_m;m)$*, is bounded by*

$$\mathsf{Regret}_T(h^*_m;m) \le \psi(m;d)T + \phi_1(h^*_m;m,T) + \phi_2(h^*_m;m,T). \tag{4.1}$$

The first term $\psi(m;d)T$ describes the regret incurred from the difference between the biggest element in the submatrix and the biggest element in the entire matrix, per the definition of the subsampling cost function ψ (Definition 7). The remaining sum, $\phi_1(h^*_m;m,T) + \phi_2(h^*_m;m,T)$, bounds the regret of running LRB on the submatrix when the value of the filtering resolution is h^*_m. The gap-dependent bound for the submatrix sampling algorithm can be constructed similarly, such that the first term $\psi(m;d)T$ stays the same, and for the remaining terms, we use the gap-dependent regret bound on the submatrix. In the extreme case where $m = d$, the regret is reduced to that in Theorem 2. Hence, LRB can be viewed as a special case of ss-LRB. We visualize the benefits of subsampling based on the regret upper bound (4.1) in Section 4.3.

To bring the unknown subsampling cost function $\psi(m;d)$ into the analysis, we let $m = \eta d$, where η is called the subsampling ratio which is a decision variable. We assume a structure of the subsampling cost in Assumption 2 such that the subsampling cost is equal to zero when $\eta = 1$; that is, there is no subsampling cost if the submatrix is the entire matrix. Further, the subsampling cost increases when η reduces.

Assumption 2 (Subsampling Cost). *Suppose that* $\psi(\eta d;d) \le c_s(1-\eta)^{\gamma_1}\eta^{\gamma_2}d^\iota$*, where* $\gamma_1 \ge 1$*,* $\gamma_2 \le 0$*, and* $c_s \ge 0$*.*

We show in Section EC.6 of the Online Appendix that subsampling cost functions for matrices whose entries are drawn from some common distributions satisfy Assumption 2, and we provide further empirical evidence in Section EC.7 of the Online Appendix to show that Assumption 2 holds for a variety of low-rank matrices.

Under Assumption 2, we provide regret upper bounds of the subsampling version of LRB, namely the ss-LRB, in Theorem 5. From the analysis, we show that it can be more efficient to work with a submatrix in (and possibly beyond) the ultra-short horizon regime, even though the first term grows linearly with time horizon T in Equation (4.1).

Theorem 5. *Suppose Assumption 2 holds. Recall that* ϑ *is defined in Theorem 3.*

1. *When* $T \ge \vartheta d^{\frac{\zeta+4}{\zeta+1}}$, $\mathsf{Regret}^{SS}_T = \tilde{O}(\min\{d^\iota T, dT^{\frac{2}{\zeta+4}}\})$.
2. *When* $T \le \vartheta d^{\frac{\zeta+4}{\zeta+1}}$, $\mathsf{Regret}^{SS}_T = \tilde{O}(\min\{d^\iota T, d^{\frac{1}{3}}T^{\frac{2}{3}}\})$.

Moreover, to achieve this regret order, we select the subsampling ratio in the following way:

i. *When* $T \ge \vartheta d^{\frac{\zeta+4}{\zeta+1}}$:
   a. *If* $T^{\frac{\zeta+2}{\zeta+4}} \le d^{1-\iota}$*, we choose* $\eta = d^{\iota-1}T^{\frac{\zeta+2}{\zeta+4}}$*;*
   b. *If* $T^{\frac{\zeta+2}{\zeta+4}} \ge d^{1-\iota}$*, we choose* η *such that* $(1-\eta)^{\gamma_1}\eta^{\gamma_2} = d^{1-\iota}T^{-\frac{\zeta+2}{\zeta+4}}$*.*

ii. *When* $T \le \vartheta d^{\frac{\zeta+4}{\zeta+1}}$:
   a. *If* $T \ge d^{1-3\iota}$*, we choose* η *such that* $(1-\eta)^{\gamma_1}\eta^{\gamma_2} = d^{\frac{1}{3}-\iota}T^{-\frac{1}{3}}$*;*
   b. *If* $T \le d^{1-3\iota}$*, we choose* $\eta = d^{3\iota-1}T$*.*

The proof of Theorem 5 is relegated to Section EC.4 in the Online Appendix. Compared with Theorem 3, the subsampling strategy has reduced the regret from Regret_T in Theorem 3 to $\mathsf{Regret}^{SS}_T = \min\{\mathsf{Regret}_T, \tilde{O}(d^\iota T)\}$. We notice that, although the optimal subsampling ratio η depends on all parameters in Assumption 2

(i.e.,γ_1, γ_2, ι, and ζ), the improvement of the subsampling strategy depends solely on the value of ι. Next, we provide an example to present more concrete improvement.

Example 4. Let the low-rank matrices $\mathbf{B} = \mathbf{U}\mathbf{V}^\top$, where the entries of $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{d\times r}$ follow uniform distribution on [0,1]. From the empirical evidence in Section EC.7 in the Online Appendix, $c_s \approx 2$, $\gamma_1 \approx 1$, $\gamma_2 \approx -0.5$, and $\iota \approx -\frac{1}{4}$. In this case, $\psi(\eta d; d) \le 2(1-\eta)\eta^{-\frac{1}{2}}d^{-\frac{1}{4}}$. Moreover, $\zeta \approx 3$ as discussed in Example 3, that is, $\mathbb{E}_{|\mathcal{I}_r|=|\mathcal{I}_c|=m}[\sqrt{g(h;\mathcal{I}_r,\mathcal{I}_c)}] \le mh^{3/2}$. By applying Theorem 5, when $T \ge \vartheta d^{\frac{7}{4}}$, the regret is of order $\min\left\{\tilde{O}(d^{-\frac{1}{4}}T), \tilde{O}(dT^{\frac{2}{7}})\right\}$; when $T \le \vartheta d^{\frac{7}{4}}$, the regret is of order $\min\left\{\tilde{O}(d^{-\frac{1}{4}}T), \tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})\right\}$. Compared with Example 3, when $T \le \vartheta d^{\frac{7}{4}}$, the regret can be reduced from $\tilde{O}(d^{\frac{1}{3}}T^{\frac{2}{3}})$ to $\tilde{O}(d^{-\frac{1}{4}}T)$ with the subsampling strategy.

The above example shows that ss-LRB performs better than LRB under the ultra-short horizons. In addition, Theorem 5 implies that ss-LRB can possibly improve LRB under longer horizons when ι is small. The algorithm balances between lower subsampling cost with larger submatrices and more efficient low-rank estimation with smaller submatrices. As the experimentation horizon increases, the algorithm wants to work with a larger submatrix so that it is more likely to include the largest element of the entire matrix to reduce the subsampling cost which grows linearly in time. Eventually, the algorithm will choose to work with the entire product set and utilize the low-rank structure on the entire matrix to filter out suboptimal arms when the time horizon is long enough. Such an intuition is implied by Theorem 5 ((i)(b)), where we see that, as T becomes large enough, the optimal η approaches one.

Remark 5. Similar to the comments we made in Remark 3, we see via Example 4 that we can estimate $\psi(\eta d; d)$ if we are given historical data and fit c_s, γ_1, γ_2 and ι to decide the optimal subsampling ratio η. We elaborate on a more data-driven approach in Section 5.2, where we search over a prespecified grid to find the optimal subsample size based on a historical data set.

4.3. Visualizing the Benefits of Low-Rank Structure, Targeted Exploration + Exploitation, and Subsampling

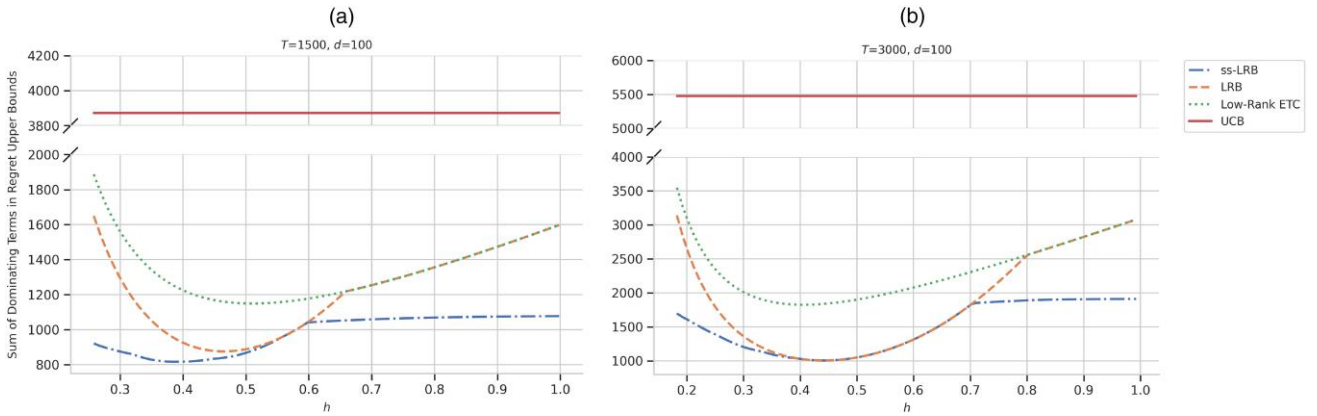
We visualize the potential benefits of leveraging low-rank structure, having a targeted exploration + exploitation phase, and subsampling, by plotting the sum of dominating terms (ignoring the constants) in the regret bound of several approaches, namely, $\psi(m;d)T + m/h^2 + \min\{\sqrt{T}\cdot\mathbb{E}_{|\mathcal{I}_r|=|\mathcal{I}_c|=m}[\sqrt{g(h;\mathcal{I}_r,\mathcal{I}_c)}], hT\}$ for ss-LRB, $d/h^2 + \min\{\sqrt{Tg(h)}, hT\}$ for LRB, $d/h^2 + hT$ for low-rank ETC, and $d\sqrt{T}$ for UCB, as they vary with h in Figure 1. We let $d = 100$ and set various values for T (more experiment details can be found in Section EC.8.1 in the Online Appendix). Note that the y-axis is the sum of the dominating terms and not the actual regret upper bounds or the actual regrets. This plot is intended to convey what we would expect from the theoretical bounds at a high level. We show with other empirical experiments that our methods outperform in terms of the actual regrets.

First of all, the curve for UCB hovers above all other approaches, showing that ignoring the low-rank structure can lead to substantially worse results.

Next, when comparing LRB and low-rank ETC, the gap between the two curves shows the value of having the targeted exploration + exploitation in the second phase. We elaborate more on different ranges of h under $T = 3{,}000$ as in Figure 1(b) first. For large h, the targeted set might be too large for efficient exploration + exploitation. Hence, targeted exploration + exploitation offers no advantage over committing to the best estimated arm under low-rank ETC, resulting in zero gap. For intermediate h, the benefit of targeted exploration + exploitation becomes evident as the gap increases. LRB attains its minimum as the estimation accuracy and the targeted set size are well balanced. That is, the time spent between pure exploration and targeted exploration + exploitation is balanced. For small h, more forced samples are needed to improve low-rank estimation, and as low-rank estimation becomes more accurate, the value of targeted exploration + exploitation decreases, shown by the shrinking gap between the rising curves. However, allocating too many forced-sampling periods to learn a highly accurate low-rank estimation is unnecessary, because our LRB can afford a larger h in the middle range to have more targeted exploration + exploitation periods and achieve a lower overall regret bound. When $T = 1{,}500$ in Figure 1(a), because of a shorter horizon, the range where targeted exploration + exploitation brings in benefits shrinks. As we plot for even shorter horizons shown in Section EC.8.1 in the Online Appendix, LRB becomes ineffective and coincides with low-rank ETC, motivating our design of ss-LRB for ultra-short horizons.

Finally, the gap between ss-LRB and LRB shows the benefit of subsampling. For ss-LRB, we plot the regret given by the optimal submatrix size for each h, where the submatrix size[4] is chosen from 5 to 100 with step size 5. The gap is zero when the full matrix size is the best submatrix size. In Figure 1(a), the lowest value of ss-LRB is achieved when it subsamples, showing that, in shorter time horizons, it is better to subsample first rather than working with regular LRB directly. In Figure 1(b), the lowest value of ss-LRB is achieved when it does not subsample, which is expected

Figure 1. (Color online) Regret Upper Bound Comparison Among ss-LRB, LRB, low-rank ETC, and UCB Under Different Values of h for Time Horizons (a) $T = 1{,}500$ and (b) $T = 3{,}000$



because the time horizon is relatively large. In both figures, when h is very small, working with a submatrix is less costly than working with a full one, because it potentially involves fewer forced-sampling rounds and a smaller targeted set. When h is very large, ss-LRB prefers smaller submatrix size, thereby reducing the time spent on pure exploration and allowing more time for targeted exploration + exploitation over a targeted set where few arms are filtered out.

5. Empirical Results

We compare the performance of LRB and ss-LRB against existing algorithms in this section. First, we present two sets of empirical results evaluating our algorithm on both synthetic data and a real world data set on NetEase Cloud Music App's impression-level data. Then, we consider a contextual bandit setting and show simulations supporting fast learning of LRB compared with an optimism-based contextual bandit algorithm.

5.1. Synthetic Data

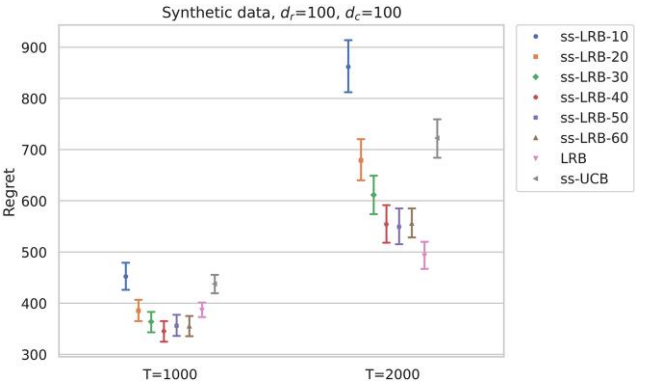
5.1.1. Synthetic Data Generation. In Figure 2, we evaluate LRB and ss-LRB over 200 simulations for two sets of parameters $T; d$: (a) $d = 100$, $\mathfrak{r} = 3$, $T = 1{,}000$ and (b) $d = 100$, $\mathfrak{r} = 3$, $T = 2{,}000$. In each case, we consider d^2 number of arms that form a d^2 matrix $\mathbf{B}^*$ of rank $\mathfrak{r}$. We compare the cumulative regret at period T. The error bars are 95% confidence intervals.

5.1.2. Results. We compare the LRB algorithm and its submatrix sampling version against the ss-UCB from Bayati et al. (2020), which lowers cost from UCB notably due to the presence of many arms. Our results demonstrate that the LRB and its submatrix sampling version significantly outperform the benchmark given different time horizon lengths. Specifically, in the shorter time horizon $T = 1{,}000$, we observe that subsampling improves the performance of LRB. In particular, ss-LRB with submatrix size equal to 40 achieves the lowest regret and reduces the LRB regret by 10%. Further, by utilizing the low-rank structure of the subsample matrix, ss-LRB with submatrix size equal to 40 reduces ss-UCB regret by 21%. In the longer time horizon $T = 2{,}000$, LRB algorithms perform better than the ss-LRB algorithm. In addition, LRB algorithm cuts the regret of ss-UCB by 31%, demonstrating that the benefit from the low-rank structure outweighs the benefit from subsampling. We specify the algorithm inputs and other details in Section EC.8.2 in the Online Appendix.

5.2. Data-Driven Approach to Select Experimentation Parameters for a Music Streaming App

In a real-world setting, we can tune the experimentation parameters in a data-driven manner by using

Figure 2. (Color online) Distribution of Cumulative Regrets at $T = 1{,}000$ and $T = 2{,}000$ for (1) ss-LRB (Algorithm 2) with Submatrix Size $m = 10$, (2) ss-LRB with $m = 20$, (3) ss-LRB with $m = 30$, (4) ss-LRB with $m = 40$, (5) ss-LRB with $m = 50$, (6) ss-LRB with $m = 60$, (7) LRB (Algorithm 1), and (8) ss-UCB with Subsampling Size $= \lfloor 4\sqrt{T} \rfloor$



related historical data. To illustrate, we utilize a real-world data set to simulate an advertising campaign setting as described in Example 1. That is, an advertiser needs to specify which pair of user group and content creator group to target for the advertisement campaign to be most effective.[5] The number of combinations of a user group and a creator group can be large, as we can distinguish user groups with many different information such as demographics, geographic information, and behavior. Likewise, we can distinguish creator groups with many distinct features. Our algorithm (and other benchmark algorithms) would seek to find the best combination of a user group and a creator group to maximize overall interaction between the two groups. As such, we choose an application in the music streaming industry, where we have access to user and creator interaction data. More details on the data can be found in Zhang et al. (2022). We suppress the interaction data information and let different algorithms learn such information over time. We provide more background on music streaming services in Section EC.8.4 in the Online Appendix.

5.2.1. Our Problem. The company has been providing services to older-aged users and now wants to expand their services to younger-aged ones. It has also attracted new creators to the platform and plans to run an experiment to find a younger-aged user group and a new creator group that interact the most to target an advertisement campaign. The experimentation parameters can be found via historical data on interaction between older-aged user groups and seasoned creator groups.

5.2.2. Bandit Formulation. To simulate such a setting, we gather the data that correspond to the older-aged users and seasoned creators as a "historical" data set and keep the data that correspond to the younger-aged users and new creators on the side as a "test" data set. For each data set, we cluster the samples into 53 user groups and 39 creator groups, so that we have a 2,067-armed stochastic bandit. We bucket users using different locations (provinces) and activity intensity levels. We bucket creators using the anonymous creator types and activity intensity levels. We obtain the same user groups and creator groups for the two data sets. For each (user group, creator group)-pair, we take the average of all interaction data between this user group and this creator group (including clicks, likes, shares, whether the user commented, whether the user viewed the comments, and whether the user visited the creator's homepage) and treat it as the reward for this pair.

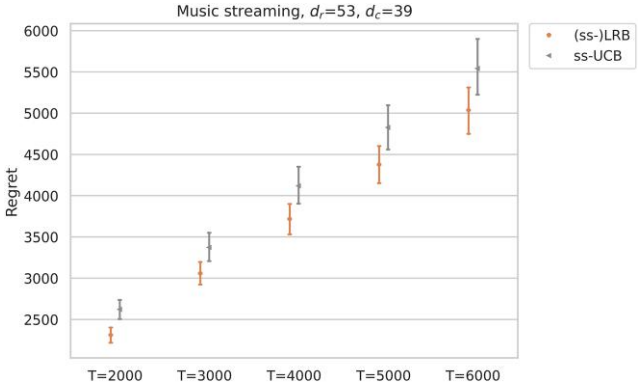
5.2.3. Selecting Experimentation Parameters. The experimentation on finding a pair of younger-aged user group and new creator group needs input parameters such as the number of forced samples, denoted as f, the filtering resolution h, and the submatrix sampling size m for different experimentation horizon length T. We work with a prespecified grid of parameters (listed in Section EC.8.3 in the Online Appendix) and simulate a bandit experiment for each combination of the parameters using the historical data set. We pick the set of parameters that achieves the lowest regret on the historical data set averaged over 30 trials. Essentially speaking, we assume that the historical data and test data share similar reward distributions; hence, the optimal parameters learned for the historical data can be used for the experiment we want to run for the test data.

5.2.4. Results and Evaluation. In Figure 3, we report the regret of using test data to conduct experiments using our algorithm (ss-LRB) versus the benchmark algorithm (ss-UCB). We consider 300 runs. The error bars are 95% confidence intervals. Our algorithm consistently reduces the regret of the benchmark ss-UCB by around 10% across all time horizons.

In this real data experiment, without knowing whether the data generation produces low-rank matrix of rewards, we still see the effectiveness of our algorithm. Further, even though the historical data distribution may differ from the test data, our experiment results show that using the optimal parameters learned from the historical data still yields better performance than the benchmark. This shows that our data-driven approach of selecting the experiment parameters provides a practical solution to real-world scenarios, such that we can train the parameters on related historical data sets prior to starting an experiment. In practice, when no past data are available to tune these parameters, one can select parameters using the insights provided in Section 3 and 4.

Thus far, we did not use any contextual information. In the next section, we show that LRB performs better

Figure 3. (Color online) Distribution of the Per-Instance Regret Under Different Time Horizons



Notes. Parameters of our algorithm (ss-)LRB are selected based on historical data. ss-UCB of Bayati et al. (2020) has subarm size $= 4\sqrt{T}$.

than linear bandits that use contextual information, when the time horizon is short, because linear bandits take a long time to learn all the contextual information to initialize.

5.3. Fast Learning in the Contextual Setting

In this section, we consider a classical contextual bandit setting and show how we use LRB to lower the cost in a synthetic experiment. We compare with a bandit algorithm, OFUL, that is designed to learn all the contextual information (Abbasi-Yadkori et al. 2011). This comparison is motivated by the observation that the regret of linear bandits at the initial rounds of an experiment increases almost linearly because the algorithm is essentially collecting initial data to perform a baseline estimate of the model parameters. Our algorithm deals with this so-called cold-start learning problem by identifying the low-rank structure behind the arm rewards, ignoring the learning of a large number of parameters. Even though our algorithm does not observe the contextual information, it learns latent feature vectors of the two-sided products quickly and incurs less cost.

5.3.1. Contextual Bandit Setting. We consider a bandit setting with $d_r d_c$ number of arms whose parameters are unknown. We index the $d_r d_c$ arms by (j,k) for every $j \in [d_r]$ and every $k \in [d_c]$. We denote each arm feature with a p-dimensional vector $A^*_{jk} \in \mathbb{R}^p$. At time t, by selecting arm (j,k), we observe a linear reward $A^{*\top}_{jk} X + \epsilon_t$, where $X \in \mathbb{R}^p$ is the (population-level) known context. The expected regret incurred at period t is $\max_{A^*_{j'k'}} [A^{*\top}_{j'k'} X - A^{*\top}_{jk} X]$. We want to minimize the cumulative expected regret and find the arm that has the highest reward.

To match this bandit formulation with an example, we consider the case of tailoring a homepage of an app to a user segment with a known context X mentioned in Example 2. The home page consists of a welcome text message and a picture background and thus can be considered as a two-sided product. We have d_r choices of welcome text messages and d_c choices for picture backgrounds. A^*_{jk} is the unknown ground truth arm feature of homepage indexed by (j,k).

5.3.2. Low-Rank Reward Matrix. The mean rewards can be shaped into the following matrix $\mathbf{B}^*$:

$$\mathbf{B}^* = [\mathbf{A}^*_1 X | \mathbf{A}^*_2 X | \cdots | \mathbf{A}^*_{d_c} X] \in \mathbb{R}^{d_r \times d_c}, \text{ where } \mathbf{A}^*_k = \begin{bmatrix} A^{*\top}_{1k} \\ A^{*\top}_{2k} \\ \vdots \\ A^{*\top}_{d_r k} \end{bmatrix} \in \mathbb{R}^{d_r \times p}.$$

The low-rank structure of $\mathbf{B}^*$ attributes to the modeling detailed next. Let $\mathbf{A}^*_k = \mathbf{U}\mathbf{V}^\top_k$, where $\mathbf{U} \in \mathbb{R}^{d_r \times \mathfrak{r}}$ and $\mathbf{V}_k \in \mathbb{R}^{p \times \mathfrak{r}}$. Then $\mathbf{B}^* = [\mathbf{U}\mathbf{V}^\top_1 X | \mathbf{U}\mathbf{V}^\top_2 X | \cdots | \mathbf{U}\mathbf{V}^\top_{d_c} X] = \mathbf{U}[\mathbf{V}^\top_1 X | \mathbf{V}^\top_2 X | \cdots | \mathbf{V}^\top_{d_c} X]$. Under this design, the mean reward matrix $\mathbf{B}^*$ is of rank $\mathfrak{r}$ because $\mathbf{U}$ is in $\mathbb{R}^{d_r \times \mathfrak{r}}$ and $[\mathbf{V}^\top_1 X | \mathbf{V}^\top_2 X | \cdots | \mathbf{V}^\top_{d_c} X]$ is in $\mathbb{R}^{\mathfrak{r} \times d_c}$. We assume $\mathfrak{r}$ is small, so the mean reward matrix is of low rank. This is because we only need very few features to explain each welcome text message and each picture background.

In the example of designing homepage, each row of $\mathbf{U}$ can be interpreted as the feature vector of a text message, and each column of $[\mathbf{V}^\top_1 X | \mathbf{V}^\top_2 X | \cdots | \mathbf{V}^\top_{d_c} X]$ can be interpreted as the feature vector of a picture background. Welcome text messages are more straightforward and the interpretation would be the same across different user segments, so they are assumed to be independent of X. Picture backgrounds, on the other hand, are more subjective and up to each user segment's interpretation. Thus, the representation depends on X.

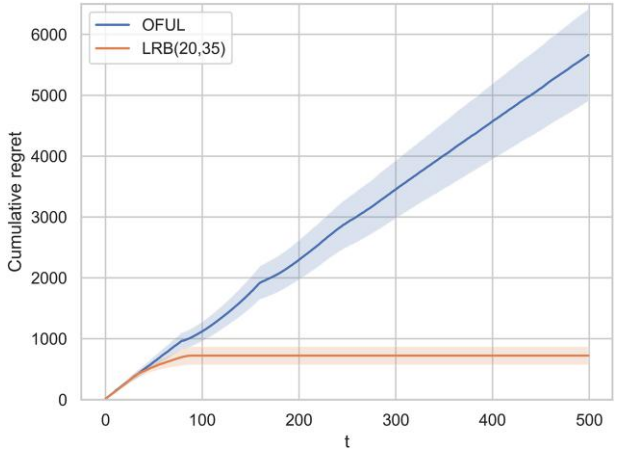
5.3.3. LRB Policy. To use our LRB method in Algorithm 1, we can treat the observed reward $A^{*\top}_{jk} X + \epsilon_t$ as the noisy observation of entry (j,k) in $\mathbf{B}^*$. Note that $A^{*\top}_{jk} X = \mathbf{B}^*_{jk}$.

5.3.4. OFUL Policy. To apply OFUL in this setting, we need the following transformation. We work with an arm set that is established by the known context X and an unknown vector $\Theta \in \mathbb{R}^{d_r d_c p}$ that is constructed by using A^*_{jk}. The arm set is $\{\tilde{A}_{jk} \in \mathbb{R}^{d_r d_c p} | j \in [d_r], k \in [d_c]\}$, such that $\tilde{A}_{jk} = [X^\top_{11}, X^\top_{12}, \ldots, X^\top_{jk'}, \ldots, X^\top_{d_r d_c}]^\top$, where $X_{jk} = X$ and, for $(j',k') \neq (j,k)$, $X_{j'k'}$ is the vector of all zeros in $\mathbb{R}^p$. The parameter Θ is defined by $\Theta = [A^{*\top}_{11}, A^{*\top}_{12}, \ldots, A^{*\top}_{jk}, \ldots, A^{*\top}_{d_r d_c}]^\top$, which gives $\tilde{A}^\top_{jk}\Theta = A^{*\top}_{jk} X$. At time t, by pulling arm (j,k) with known feature $\tilde{A}_{jk}$, it is equivalent to say that we observe a linear reward $\tilde{A}^\top_{jk}\Theta + \epsilon_t$.

5.3.5. Synthetic Experiment Setup. Our goal is to seek a policy that minimizes the cumulative expected regret over time horizon $T = 500$. We set $d_r = 8$, $d_c = 10$, $\mathfrak{r} = 3$, and $p = 7$. We evaluate different policies over 10 simulations. The error bars are 95% confidence intervals. Other experiment details can be found in Section EC.8.5 in the Online Appendix.

5.3.6. Results. In Figure 4, focusing on time $T = 500$, the average cumulative regret of OFUL is 5,662 and the average cumulative regret of LRB is 723. LRB lowers the regret of OFUL by nearly 87%. This result shows that, although OFUL takes time to learn each dimension of Θ with $d_r \times d_c \times p = 8 \times 10 \times 7 = 560$ dimensions, LRB efficiently transfers knowledge learned for some arms to other arms by taking advantage of the

Figure 4. (Color online) Cumulative Regrets of OFUL and Our Algorithm LRB with Number of Forced Samples Equal to 20 and the Filtering Resolution Equal to 35 Under the Contextual Setting



low-rank structure and thus learns much quicker and incurs much smaller costs. We show in Section EC.8.5 of the Online Appendix that our algorithm consistently outperforms OFUL by a big margin under all combinations of the experiment parameters we have tried.

It is possible to design a variation of OFUL as a benchmark that takes advantage of the special structure of the transformed arms (most dimensions of which are zeros). For example, the OFUL can learn a subset of dimensions in Θ or it can learn a sparse model, in similar spirit to Lasso bandit. However, the objective here is to show the advantage of looking at the low-rank reward structure compared with a standard linear bandit implementation.

6. Conclusion

This work introduced novel two-sided bandit algorithms, LRB and ss-LRB, to speed up the cold-start learning when there are many arms and the experimentation horizon is relatively short. Our theoretically and empirically efficient solution features leveraging the low-rank structure to shorten the *burn-in* period or the exploration phase, a filtering mechanism to filter out suboptimal arms, and the submatrix sampling to accommodate ultra-short horizons.

Our framework offers practical guidance for the design of large-scale online experiments. In many industrial settings, such as A/B testing systems, recommender algorithms, or marketing experiments, the number of potential interventions (e.g., user-item pairs, ad-audience combinations, or headline-image variants) is combinatorially large. A common workflow is to first use heuristics or manual curation to filter this vast action space down to a manageable subset and then run an experiment on this reduced set. Our LRB algorithm provides a formal alternative to this ad hoc filtering. Its initial forced-sampling phase serves as a principled, low-cost burn-in. Unlike a naive exploration that must sample all $d_r d_c$ arms, our phase is guided by the latent low-rank structure. It requires only a limited number of samples, scaling with the sum of the dimensions $(d_r + d_c)$ rather than their impractical product, to efficiently estimate the reward landscape and rapidly identify a high-quality targeted set. This formally justifies and improves upon the common heuristic practice.

6.1. Future Directions

Besides the low-rank estimator, prediction methods that leverage other problem structures may benefit from pairing with our targeted exploration + exploitation and/or subsampling strategy for enhanced performance as well. Such a heuristic may be of independent interest. For example, when we want to leverage more dimensions to characterize a product, we can formulate a tensor version of our LRB. As a real-life application, Airbnb Experiences can be modeled as three-sided products by including a temporal side in addition to the content and the location of an activity as mentioned in Section 1. On a related note, for settings such as Stitch Fix personalized styling, we have a unique reward profile for each user and different user reward matrices might share similarity. To find the best outfit for each user, we can form a tensor version of our LRB that allows information-sharing among different users, or we can transfer latent contextual information learned across different users. In addition, it would be interesting to see how our algorithm can be modified to tackle an online product assortment experiment: We have users and products as rows and columns, upon a user's arrival, we want to recommend a product (or an array of products) he/she likes the best.

Our work also gives new perspectives on how to leverage potential low-rank matrix structure for product bundling (Stigler 1963) because the basic two-product bundling case considered in canonical works (Adams and Yellen 1976, Schmalensee 1984, McAfee et al. 1989) can be viewed as two-sided products. The entries of the matrix can be the expected rewards or purchase probabilities of bundles of two products. Through customer's purchase behaviors, the preference regarding each bundle can be learned, which ultimately informs the pricing scheme.

Finally, our work can be extended to use offline collected historical data (if any) to warm start our two-sided bandit under limited online experiment horizon. For example, we can leverage the offline data for low-rank matrix estimation,[6] by adopting a meta-algorithm called artificial replay proposed by Banerjee et al. (2022) and previously cited in Section 1.2 to incorporate selected historical data into our bandit algorithm.

Acknowledgments

All authors contributed equally.

Endnotes

[1] We use this term to describe the initial nonadaptive data collection phase, where the algorithm operates like a randomized experiment. This is analogous to its use in the Markov chain Monte Carlo (MCMC) literature for the initial period before convergence (Brooks et al. 2011). We distinguish this from another common use of "burn-in" in online experimentation, which typically refers to a period for time-varying treatment effects (e.g., novelty effects) to stabilize; that is not the context here.

[2] It should be noted that, although standard MAB algorithms are recognized for *addressing* cold-start problems in certain studies (Ye et al. 2023) through effective exploration and exploitation, this capability is contingent on a relatively short *burn-in period* compared with the total experimentation horizon. As we have discussed, given a substantial number of arms and a constrained time horizon, standard MAB algorithms cannot manage the cold-start problem effectively.

[3] Theorem 7.2 of Lattimore and Szepesvári (2020).

[4] We treat LRB as a special case of ss-LRB by setting $m = d = 100$.

[5] In Section EC.8.4 of the Online Appendix, we discuss other related modes of advertisement campaigns that are not necessarily two-sided products.

[6] Depending on whether the offline samples are uniformly or nonuniformly sampled, we can keep using the entry-wise bound from Chen et al. (2020) or switch to Xi et al. (2023) who have derived an entry-wise bound for sampling a matrix under a nonuniform pattern.

References

Abbasi-Yadkori Y, Pál D, Szepesvári C (2011) Improved algorithms for linear stochastic bandits. *Adv. Neural Inform. Processing Systems*, vol. 24 (Curran Associates Inc., Red Hook, NY), 2312–2320.

Adams WJ, Yellen JL (1976) Commodity bundling and the burden of monopoly. *Quart. J. Econom.* 90(3):475–498.

Agrawal S, Devanur NR (2014) Bandits with concave rewards and convex knapsacks. Babaioff M, Conitzer V, Easley D, Nisan N, eds. *Proc. 15th ACM Conf. Econom. Comput*. (ACM, New York), 989–1006.

Agrawal S, Goyal N (2012) Analysis of Thompson sampling for the multi-armed bandit problem. Mannor S, Srebro N, Williamson RC, eds. *Proc. Conf. Learn. Theory*, vol. 39 (PMLR, New York), 1–26.

Agarwal A, Dahleh M, Shah D, Shen D (2023) Causal matrix completion. Neu G, Rosasco L, eds. *Proc. 36th Ann. Conf. Learn. Theory* (PMLR, New York), 3821–3826.

Allouah A, Bahamou A, Besbes O (2023) Optimal pricing with a single point. *Management Sci.* 69(10):5866–5882.

Athey S, Bayati M, Doudchenko N, Imbens G, Khosravi K (2021) Matrix completion methods for causal panel data models. *J. Amer. Statist. Assoc.* 116(536):1716–1730.

Auer P, Cesa-Bianchi N, Fischer P (2002a) Finite-time analysis of the multiarmed bandit problem. *Machine Learn.* 47(2):235–256.

Auer P, Cesa-Bianchi N, Freund Y, Schapire RE (2002b) The nonstochastic multiarmed bandit problem. *SIAM J. Comput.* 32(1):48–77.

Bajari P, Burdick B, Imbens GW, Masoero L, McQueen J, Richardson T, Rosen IM (2021) Multiple randomization designs. Preprint, submitted December 27, https://arxiv.org/abs/2112.13495.

Banerjee S, Sinclair SR, Tambe M, Xu L, Yu CL (2022) Artificial replay: A meta-algorithm for harnessing historical data in bandits. Preprint, submitted October 1, https://arxiv.org/abs/2210.00025.

Bastani H, Bayati M (2020) Online decision making with high-dimensional covariates. *Oper. Res.* 68(1):276–294.

Bastani H, Harsha P, Perakis G, Singhvi D (2022) Learning personalized product recommendations with customer disengagement. *Manufacturing Service Oper. Management* 24(4):2010–2028.

Bayati M, Hamidi N, Johari R, Khosravi K (2020) Unreasonable effectiveness of greedy algorithms in multi-armed bandit with many arms. *Adv. Neural Inform. Processing Systems* 33:1713–1723.

Berry DA, Chen RW, Zame A, Heath DC, Shepp LA (1997) Bandit problems with infinitely many arms. *Ann. Statist.* 25(5):2103–2116.

Besbes O, Mouchtaki O (2023) How big should your data really be? Data-driven newsvendor: Learning one sample at a time. *Management Sci.* 69(10):5848–5865.

Bhargava HK (2022) The creator economy: Managing ecosystem supply, revenue sharing, and platform design. *Management Sci.* 68(7):5233–5251.

Bonald T, Proutiere A (2013) Two-target algorithms for infinite-armed bandits with Bernoulli rewards. *Adv. Neural Inform. Processing Systems* 26:2184–2192.

Brooks S, Gelman A, Jones G, Meng X-L (2011) *Handbook of Markov Chain Monte Carlo* (CRC Press, Boca Raton, FL).

Candes E, Recht B (2012) Exact matrix completion via convex optimization. *Comm. ACM* 55(6):111–119.

Carpentier A, Valko M (2015) Simple regret for infinitely many armed bandits. Bach F, Blei D, eds. *Proc. Internat. Conf. Machine Learn.* (PMLR, New York), 1133–1141.

Chaudhuri AR, Kalyanakrishnan S (2018) *Quantile-Regret Minimisation in Infinitely Many-Armed Bandits* (UAI).

Chen Y (2015) Incoherence-optimal matrix completion. *IEEE Trans. Inform. Theory* 61(5):2909–2923.

Chen Y, Chi Y, Fan J, Ma C, Yan Y (2020) Noisy matrix completion: Understanding statistical guarantees for convex relaxation via nonconvex optimization. *SIAM J. Optim.* 30(4):3098–3121.

Chu W, Li L, Reyzin L, Schapire R (2011) Contextual bandits with linear payoff functions. *Proc. 14th Internat. Conf. Artificial Intelligence Statist.* (PMLR, New York), 208–214.

Dani V, Hayes TP, Kakade SM (2008) Stochastic linear optimization under bandit feedback. Servedio RA, Zhang T, eds. *Proc. Conf. Learn. Theory* (Omnipress, Madison, WI), 355–366.

Davenport MA, Romberg J (2016) An overview of low-rank matrix recovery from incomplete observations. *IEEE J. Selected Topics Signal Processing* 10(4):608–622.

Farias V, Li AA, Peng T (2022) Uncertainty quantification for low-rank matrix completion with heterogeneous and sub-exponential noise. Camps-Valls G, Ruiz FJR, Guyon I, eds. *Proc. Internat. Conf. Artificial Intelligence Statist.* (PMLR, New York), 1179–1189.

Filippi S, Cappe O, Garivier A, Szepesvári C (2010) Parametric bandits: The generalized linear case. *Adv. Neural Inform. Processing Systems* 23:586–594.

Geng T, Lin X, Nair HS (2020) Online evaluation of audiences for targeted advertising via bandit experiments. *Proc. AAAI Conf. Artificial Intelligence* 34(1):13273–13279.

Goldenshluger A, Zeevi A (2013) A linear response bandit problem. *Stochastic Systems* 3(1):230–261.

Gopalan A, Mannor S, Mansour Y (2014) Thompson sampling for complex online problems. *Proc. Internat. Conf. Machine Learn.* (PMLR, New York), 100–108.

Gupta V, Kallus N (2022) Data pooling in stochastic optimization. *Management Sci.* 68(3):1595–1615.

Gupta V, Rusmevichientong P (2021) Small-data, large-scale linear optimization with uncertain objectives. *Management Sci.* 67(1):220–241.

Hamidi N, Bayati M, Gupta K (2019) Personalizing many decisions with high-dimensional covariates. *Adv. Neural Inform. Processing Systems* 32:11469–11480.

Hastie T, Tibshirani R, Wainwright M (2015) *Statistical Learning with Sparsity: The Lasso and Generalizations* (Taylor & Francis, Boca Raton, FL).

Johari R, Li H, Liskovich I, Weintraub GY (2022) Experimental design in two-sided platforms: An analysis of bias. *Management Sci.* 68(10):7069–7089.

Jun K-S, Willett R, Wright S, Nowak R (2019) Bilinear bandits with low-rank structure. Chaudhuri K, Salakhutdinov R, eds. *Proc. Internat. Conf. Machine Learn.* (PMLR, New York), 3163–3172.

Kaelbling LP (1993) *Learning in Embedded Systems* (MIT Press, Cambridge, MA).

Kallus N, Udell M (2020) Dynamic assortment personalization in high dimensions. *Oper. Res.* 68(4):1020–1037.

Katariya S, Kveton B, Szepesvari C, Vernade C, Wen Z (2017) Stochastic rank-1 bandits. Singh A, Zhu J, eds. *Proc. Artificial Intelligence Statist.* (PMLR, New York), 392–401.

Katehakis MN, Robbins H (1995) Sequential choice from several populations. *Proc. Natl. Acad. Sci. USA* 92(19):8584.

Keshavan RH, Montanari A, Oh S (2010) Matrix completion from a few entries. *IEEE Trans. Informs. Theory* 56(6):2980–2998.

Keskin NB, Li Y, Sunar N (2024) Data-driven clustering and feature-based retail electricity pricing with smart meters. *Oper. Res.* 73(5):2636–2660.

Kveton B, Szepesvári C, Rao A, Wen Z, Abbasi-Yadkori Y, Muthukrishnan S (2017) Stochastic low-rank bandits. Preprint, submitted December 13, https://arxiv.org/abs/1712.04644.

Kveton B, Zaheer M, Szepesvari C, Li L, Ghavamzadeh M, Boutilier C (2020) Randomized exploration in generalized linear bandits. *Proc. Internat. Conf. Artificial Intelligence Statist.*, 2066–2076.

Lai TL (1987) Adaptive treatment allocation and the multi-armed bandit problem. *Ann. Statist.* 15(3):1091–1114.

Lattimore T, Szepesvári C (2020) *Bandit Algorithms* (Cambridge University Press, Cambridge, MA).

Li L, Lu Y, Zhou D (2017) Provably optimal algorithms for generalized linear contextual bandits. Precup D, Teh YW, eds. *Proc. 34th Internat. Conf. Machine Learn.*, vol. 70 (PMLR, New York), 2071–2080.

Li L, Chu W, Langford J, Schapire RE (2010) A contextual-bandit approach to personalized news article recommendation. *Proc. 19th Internat. Conf. World Wide Web* (ACM, New York), 661–670.

Lika B, Kolomvatsos K, Hadjiefthymiades S (2014) Facing the cold start problem in recommender systems. *Expert Systems Appl.* 41(4):2065–2073.

Lops P, De Gemmis M, Semeraro G (2011) Content-based recommender systems: State of the art and trends. Ricci F, Rokach L, Shapira B, Kantor P, eds. *Recommender Systems Handbook* (Springer, Berlin), 73–105.

Lu S (2019) Beyond A/B testing: Multi-armed bandit experiments. *Medium* (April 4), https://medium.com/data-science/beyond-a-b-testing-multi-armed-bandit-experiments-1493f709f804.

Lu Y, Meisami A, Tewari A (2021) Low-rank generalized linear bandit problems. Banerjee A, Fukumizu K, eds. *Proc. Internat. Conf. Artificial Intelligence Statist.* (PMLR, New York), 460–468.

Lu X, Wen Z, Kveton B (2018) Efficient online recommendation via low-rank ensemble sampling. Pera S, Rendle S, Vlachos M, eds. *Proc. 12th ACM Conf. Recommender Systems* (ACM, New York), 460–464.

McAfee RP, McMillan J, Whinston MD (1989) Multiproduct monopoly, commodity bundling, and correlation of values. *Quart. J. Econom.* 104(2):371–383.

Mersereau AJ, Rusmevichientong P, Tsitsiklis JN (2009) A structured multiarmed bandit problem and the greedy policy. *IEEE Trans. Automated Control* 54(12):2787–2802.

Miao S, Chen X, Chao X, Liu J, Zhang Y (2022) Context-based dynamic pricing with online clustering. *Production Oper. Management* 31(9):3559–3575.

Nakamura A (2015) A UCB-like strategy of collaborative filtering. Bui TD, Ho TB, eds. *Proc. Asian Conf. Machine Learn.* (PMLR, New York), 315–329.

Pal S, Jain P (2023) Online low rank matrix completion. *Proc. 11th Internat. Conf. Learn. Representations* (ICLR, Appleton, WI).

Rusmevichientong P, Tsitsiklis JN (2010) Linearly parameterized bandits. *Math. Oper. Res.* 35(2):395–411.

Russo DJ, Van Roy B, Kazerouni A, Osband I, Wen Z (2018) A tutorial on thompson sampling. *Foundations Trends® Machine Learn.* 11(1):1–96.

Sam T, Chen Y, Yu CL (2023) Overcoming the long horizon barrier for sample-efficient reinforcement learning with latent low-rank structure. *Proc. ACM Measurment Anal. Comput. Systems* 7(2):1–60.

Schmalensee R (1984) Gaussian demand and commodity bundling. *J. Bus.* 57(1):S211–S230.

Stigler GJ (1963) United States v. Loew's Inc.: A note on block-booking. *Supreme Court Rev.* 1963:152–157.

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

Trinh C, Kaufmann E, Vernade C, Combes R (2020) Solving Bernoulli rank-one bandits with unimodal Thompson sampling. *Algorithmic Learning Theory* (PMLR, New York), 862–889.

Vershynin R (2010) Introduction to the non-asymptotic analysis of random matrices. Preprint, submitted November 12, https://arxiv.org/abs/1011.3027.

Volkovs M, Yu G, Poutanen T (2017) Dropoutnet: Addressing cold start in recommender systems. *Adv. Neural Inform. Processing Systems* 30:4957–4966.

Wang Y, Yves Audibert J, Munos R (2009) Algorithms for infinitely many-armed bandits. Koller D, Schuurmans D, Bengio Y, Bottou L, eds. *Adv. Neural Inform. Processing Systems*, vol. 21 (Curran Associates, Inc., Red Hook, NY), 1729–1736.

Xi X, Yu CL, Chen Y (2023) Entry-specific bounds for low-rank matrix completion under highly non-uniform sampling. *Proc. IEEE Internat. Sympos. Inform. Theory* (IEEE, Piscataway, NJ), 2625–2630.

Xu K, Bastani H (2021) Multitask learning and bandits via robust statistics. *Management Sci.* 71(9):7752–7773.

Ye Z, Zhang DJ, Zhang H, Zhang R, Chen X, Xu Z (2023) Cold start to improve market thickness on online advertising platforms: Data-driven algorithms and field experiments. *Management Sci.* 69(7):3838–3860.

Zhang M, Tang J, Zhang X, Xue X (2014) Addressing cold start in recommender systems: A semi-supervised co-training algorithm. Berkovsky S, Bennett PN, Murdock V, Moffat A, eds. *Proc. 37th Internat. ACM SIGIR Conf. Res. Development Informat. Retrieval* (ACM, New York), 73–82.

Zhang DJ, Hu M, Liu X, Wu Y, Li Y (2022) Netease cloud music data. *Manufacturing Service Oper. Management* 24(1):275–284.

Zhou J, Hao B, Wen Z, Zhang J, Sun WW (2025) Stochastic low-rank tensor bandits for multi-dimensional online decision making. *J. Amer. Statist. Assoc.* 120(549):198–211.

Zhu Z, Li X, Wang M, Zhang A (2022) Learning markov models via low-rank optimization. *Oper. Res.* 70(4):2384–2398.