Pareto Optimal Model Selection in Linear Bandits

Yinglun Zhu
University of Wisconsin-Madison

Robert Nowak
University of Wisconsin-Madison

Abstract

We study model selection in linear bandits, where the learner must adapt to the dimension (denoted by $d_\star$) of the smallest hypothesis class containing the true linear model while balancing exploration and exploitation. Previous papers provide various guarantees for this model selection problem, but have limitations; i.e., the analysis requires favorable conditions that allow for inexpensive statistical testing to locate the right hypothesis class or are based on the idea of "corralling" multiple base algorithms, which often performs relatively poorly in practice. These works also mainly focus on upper bounds. In this paper, we establish the first lower bound for the model selection problem. Our lower bound implies that, even with a fixed action set, adaptation to the unknown dimension $d_\star$ comes at a cost: There is no algorithm that can achieve the regret bound $\widetilde{O}(\sqrt{d_\star T})$ simultaneously for all values of $d_\star$. We propose Pareto optimal algorithms that match the lower bound. Empirical evaluations show that our algorithm enjoys superior performance compared to existing ones.

1 INTRODUCTION

Model selection considers the problem of choosing an appropriate hypothesis class to conduct learning, and the hope is to optimally balance two types of error: the approximation error and the estimation error. In the supervised learning setting, the learner is provided with a (usually nested) sequence of hypothesis classes $\mathcal{H}_d \subset \mathcal{H}_{d+1}$. As an example, $\mathcal{H}_d$ could be the hypothesis class consisting of polynomials of degree at most d. The goal is to design a learning algorithm that adaptively selects the best of these hypothesis classes, denoted by $\mathcal{H}_\star$, to optimize the trade-off between approximation error and estimation error. Structural Risk Minimization (SRM) (Vapnik and Chervonenkis, 1974; Vapnik, 1995; Shawe-Taylor et al., 1998) provides a principled way to conduct model selection in the standard supervised learning setting. SRM can automatically adapt to the complexity of the hypothesis class $\mathcal{H}_\star$, with only additional logarithmic factors in sample complexity. Meanwhile, cross-validation (Stone, 1978; Craven and Wahba, 1978; Shao, 1993) serves as a helpful tool to conduct model selection in practice.

Despite the importance and popularity of model selection in the supervised learning setting, only very recently have researchers started to study on model selection problems in interactive/sequential learning setting with bandit feedback. Two additional difficulties are highlighted in such bandit setting (Foster et al., 2019): (1) decisions/actions must be made online/sequentially without seeing the entire dataset; and (2) the learner's actions influence what data is observed, i.e., we only have partial/bandit feedback. In the simpler online learning setting with full information feedback, model selection results analogous to those in the supervised learning setting are obtained by several parameter-free online learning algorithms (McMahan and Abernethy, 2013; Orabona, 2014; Koolen and Van Erven, 2015; Luo and Schapire, 2015; Orabona and Pál, 2016; Foster et al., 2017; Cutkosky and Boahen, 2017; Cutkosky and Orabona, 2018).

The model selection problem for (contextual) linear bandits is first introduced by Foster et al. (2019). They consider a sequence of nested linear classifiers in $\mathbb{R}^{d_i}$ as the set of hypothesis classes, with $d_1 < d_2 < \cdots < d_M = d$. The goal is to adapt to the smallest hypothesis class, with apriori *unknown* dimension $d_\star$, that preserves linearity in rewards. Equivalently, one can think of the model selection problem as learning a true reward parameter $\theta_\star \in \mathbb{R}^d$, but only the first $d_\star$ entries of $\theta_\star$ contain non-zero values. The goal is to design algorithms that could automatically adapt to the intrinsic dimension $d_\star$, rather than suffering the am-

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

bient dimension d. In favorable scenarios when one can cheaply test linearity, Foster et al. (2019) provide an algorithm with regret guarantee that scales as $\widetilde{O}(K^{1/4}T^{3/4}/\gamma^2 + \sqrt{TKd_\star}/\gamma^4)$, where K is the number of arms and γ is the smallest eigenvalue of the expected design matrix. The core idea therein is to conduct a sequential test, with sublinear sample complexity, to determine whether to step into a larger hypothesis class on the fly. Although this provides the first guarantee for model selection in the linear bandits, the regret bound is proportional to the number of arms K and the reciprocal of the smallest eigenvalue, i.e., γ^{-1}. Both K and γ^{-1} can be quite large in practice, thus limiting the application of their algorithm. Recall that, when provided with the optimal hypothesis class, the classical algorithm LinUCB (Chu et al., 2011; Auer, 2002) for linear bandit achieves a regret bound $\widetilde{O}(\sqrt{d_\star T})$, with only polylogarithmic dependence on K and no dependence on γ^{-1}.

The model selection problem in linear bandits was further studied in many subsequent papers. We roughly divide these methods into the following two subcategories:

1. **Testing in Favorable Scenarios.** The algorithm in Ghosh et al. (2020) conducts a sequence of statistical tests to gradually estimate the true support (non-zero entries) of $\theta_\star$, and then applies standard linear bandit algorithms on identified support. The regret bound of their algorithm scales as $\widetilde{O}(d^2/\gamma^{4.65} + d_\star^{1/2}T^{1/2})$, where $\gamma = \min\{|\theta_{\star,i}| : \theta_{\star,i} \neq 0\}$ is the minimum magnitude of non-zero entries in $\theta_\star$. Their regret bound not only depends on the ambient dimension d but also scales inversely proportional to a small quantity γ. Their guarantee becomes vacuous when d and/or γ^{-1} are large. Chatterji et al. (2020) consider a different model selection problem where the rewards come from either a linear model or a model with K independent arms. Their algorithm also relies on sequential statistical testing, which requires assumptions stronger than the ones used in Foster et al. (2019) (thus suffering from similar problems).

2. **Corralling Multiple Base Algorithms.** Another approach maintains multiple base learners and use a master algorithm to determine sample allocation among base learners. This type of algorithm is initiated by the Corral algorithm (Agarwal et al., 2017). Focusing on our model selection setting, the base learners are usually constructed using standard linear bandit algorithms with respect to different hypothesis classes (dimensions). To give an example of the Corral-type of algorithm, the Smooth Corral algorithm developed in Pacchiano et al. (2020b) enjoys regret guarantees $\widetilde{O}(d_\star\sqrt{T})$ or $\widetilde{O}(d_\star^{1/2}T^{2/3})$. Other algorithms of this type, including some concurrent works, can be found in Abbasi-Yadkori et al. (2020); Arora et al. (2020); Pacchiano et al. (2020a); Cutkosky et al. (2020, 2021).

Note that above algorithms either only work in favorable scenarios when some critical parameters, e.g., γ^{-1} and K, are not too large or must balance over multiple base algorithms which often hurts the empirical performance. They also mainly focus on developing upper bounds for the model selection problem in linear bandits. In this paper, we explore the fundamental limits (lower bounds) of the model selection problem and design algorithms with matching guarantees (upper bounds). We establish a lower bound, using only a fixed action set, indicating that adaptation to the unknown intrinsic dimension $d_\star$ comes at a cost: There is no algorithm that can achieve the regret bound $\widetilde{O}(\sqrt{d_\star T})$ simultaneously for all values of $d_\star$. We also develop a Pareto optimal algorithm, with ideas fundamentally different from "testing" (Foster et al., 2019; Ghosh et al., 2020) and "corralling" (Pacchiano et al., 2020b; Agarwal et al., 2017), to bear on the model selection problem in linear bandits. Our algorithm is built upon the construction of virtual mixture-arms, which is previously studied in continuum-armed bandits (Hadiji, 2019) and K-armed bandits (Zhu and Nowak, 2020). We adapt their methods to our setting, with new techniques developed to deal with the linear structure, e.g., the construction of virtual dimensions.

1.1 Contribution and Outline

We briefly summarize our contributions as follows.

- We review the model selection problem in linear bandits, and additionally define a new parameter (in Section 2) that reflects the tension between time horizon and the intrinsic dimension. This parameter provides a convenient way to analyze high-dimensional linear bandits.

- We establish the first lower bound for the model selection problem in Section 3. Our lower bound indicates that the model selection problem is strictly harder than the problem with given optimal hypothesis class: There is no algorithm that can achieve the non-adaptive $\widetilde{O}(\sqrt{d_\star T})$ regret bound simultaneously for all values of $d_\star$. We additionally characterize the exact Pareto frontier of the model selection problem.

- In Section 4, we develop a Pareto optimal algorithm that is fundamentally different from existing ones relying on "testing" or "corralling". Our algorithm

is built on the construction of virtual mixture-arms and virtual dimensions. Although our main algorithm is analyzed under a mild assumption, we also provide a workaround.

- We conduct experiments in Section 5 to evaluate our algorithms. Our main algorithm shows superior performance compared to existing ones. We also show that our main algorithm is fairly robust to the existence of the assumption used in our analysis.

1.2 Other Related Work

Bandit with Large/Continuous Action Space. Adaptivity issues naturally arises in bandit problems with large or infinite action space. In continuum-armed bandit problems (Agrawal, 1995), actions are embedded into a bounded subset $\mathcal{X} \subseteq \mathbb{R}^d$ with a smooth function f governing the mean payoff for each arm. Achievable theoretical guarantees are usually influenced by some smoothness parameters, and an important question is to design algorithms that adapt to these *unknown* parameters, as discussed in Bubeck et al. (2011). Locatelli and Carpentier (2018) show that, however, no strategy can be optimal simultaneously over all smoothness classes. Hadiji (2019) establishes the Pareto frontier for continuum-armed bandits with Hölder reward functions. Adaptivity is also studied in the discrete case with a large action space (Wang et al., 2008; Lattimore, 2015; Chaudhuri and Kalyanakrishnan, 2018; Russo and Van Roy, 2018; Zhu and Nowak, 2020). Lattimore (2015) studies the Pareto frontier in standard K-armed bandits. Zhu and Nowak (2020) develop Pareto optimal algorithms for the case with multiple best arms.

High-Dimensional Linear Bandits. As more and more complex data are being used and analyzed, modern applications of linear bandit algorithms usually involve dealing with ultra-high-dimensional data, sometimes with dimension even larger than time horizon (Deshpande and Montanari, 2012). To make progress in this high-dimensional regime, one natural idea is to study (or assume) sparsity in the reward vector and try to adapt to the unknown true support (non-zero entries). The sparse bandit problem is strictly harder than the model selection setting considered here due to the absence of the hierarchical structures. Consequently, a lower bound on the regret of the form $\Omega(\sqrt{dT})$, which scales with the ambient dimension d, is indeed unavoidable in the sparse linear bandit problem (Abbasi-Yadkori et al., 2012; Lattimore and Szepesvári, 2020). Other papers deal with the sparsity setting with additional feature feedback (Oswal et al., 2020) or further distributional/structual assumptions (Carpentier and Munos, 2012; Hao et al., 2020) to circumvent the lower bound. These high-dimensional linear bandit problems motivate our investigation of the relationship between time horizon and data dimension.

2 PROBLEM SETTING

We consider a linear bandit problem with a finite action set $\mathcal{A} \subseteq \mathbb{R}^d$ where $|\mathcal{A}| = K$ (Auer, 2002; Chu et al., 2011). (The feature representation of) Each arm/action $a \in \mathcal{A}$ is viewed as a d dimensional vector, and its expected reward $f(a)$ is linear with respect to a reward parameter $\theta_\star \in \mathbb{R}^d$, i.e., $f(a) = \langle a, \theta_\star \rangle$. As standard in the literature (Lattimore and Szepesvári, 2020), we assume $\max_{a\in\mathcal{A}} \|a\| \leq 1$ and $\|\theta_\star\| \leq 1$. The bandit instance is said to have intrinsic dimension $d_\star$ if $\theta_\star$ only has non-zero entries on its first $d_\star \leq d$ coordinates. The model selection problem aims at designing algorithm that can automatically adapt to the *unknown* intrinsic dimension $d_\star$ in the interactive learning setting with bandit feedback.

At each time step $t \in [T]$,[1] the algorithm selects an action $A_t \in \mathcal{A}$ based on previous observations and receives a reward $X_t = \langle A_t, \theta_\star \rangle + \eta_t$, where η_t is an independent 1-sub-Gaussian noise. We define the pseudo regret (which is random, due to randomness in A_t) over time horizon T as $\widehat{R}_T = \sum_{t=1}^T \langle \theta_\star, a_\star - A_t \rangle$, where $a_\star$ corresponds to the best action in action set, i.e., $a_\star = \arg\max_{a\in\mathcal{A}} \langle a, \theta_\star \rangle$. We measure the performance of any algorithm by its expected regret $R_T = \mathbb{E}[\widehat{R}_T] = \mathbb{E}[\sum_{t=1}^T \langle \theta_\star, a_\star - A_t \rangle]$.

We primarily focus on the high-dimensional linear bandit setting with ambient dimension d close to or even larger than (the allowed) time horizon T. We use $\mathcal{R}(T, d_\star)$ to denote the set of regret minimization problems with time horizon T and any bandit instance with intrinsic dimension $d_\star$. We emphasize that T is part of the problem instance, which was largely neglected in previous work focusing on the low dimensional regime where $T \gg d_\star$. To model the tension between the allowed time horizon and the intrinsic dimension, we define the hardness level as

$$\psi\left(\mathcal{R}(T, d_\star)\right) = \min\{\alpha \geq 0 : d_\star \leq T^\alpha\} = \log d_\star / \log T.$$

$\psi(\mathcal{R}(T, d_\star))$ is used here since it precisely captures the regret over the set of regret minimization problem $\mathcal{R}(T, d_\star)$, as discussed later in our review of the LinUCB algorithm and the lower bound. Since smaller $\psi(\mathcal{R}(T, d_\star))$ indicates easier problem, we define the family of regret minimization problems with *hardness*

[1] Throughout the paper, we denote $[n] = \{1, 2, \ldots, n\}$ for any positive integer n.

level at most α as

$$\mathcal{H}_T(\alpha) = \{\cup \mathcal{R}(T, d_\star) : \psi(\mathcal{R}(T, d_\star)) \leq \alpha\},$$

where $\alpha \in [0, 1]$. Although T is necessary to define a regret minimization problem, the hardness of the problem is encoded into a single parameter α: Problems with different time horizons but the same α are equally difficult in terms of the regret achieved by LinUCB (the exponent of T). We explore the connection $d_\star \leq T^\alpha$ in the rest of this paper and focus on (polynomial) dependence on T (i.e., the dependence on $d_\star$ is translated into the dependence on T^α). We are interested in designing algorithms with worst case guarantees over $\mathcal{H}_T(\alpha)$, but *without* the knowledge of α.

LinUCB and Upper Bounds. In the standard setting where $d_\star$ is known, LinUCB Chu et al. (2011); Auer (2002) achieves $\widetilde{O}(\sqrt{d_\star T})$ regret.[2] For any problem in $\mathcal{H}_T(\alpha)$ with *known* α, one could run LinUCB on the first $\lfloor T^\alpha \rfloor$ coordinates and achieve $\widetilde{O}(T^{(1+\alpha)/2})$ regret. The goal of model selection is to achieve the $\widetilde{O}(T^{(1+\alpha)/2})$ regret but without the knowledge of α.

Lower Bounds. In the case when $d_\star \leq \sqrt{T}$, Chu et al. (2011) prove a $\Omega(\sqrt{d_\star T})$ lower bound for linear bandits. When $d_\star \geq \sqrt{T}$ is the case, a lower bound $\Omega(K^{1/4}T^{3/4})$ is developed in Abe et al. (2003).

3 LOWER BOUND AND PARETO OPTIMALITY

We study lower bounds for model selection in this section. We show that simultaneously adapting to all hardness levels is impossible. Such fundamental limitation leads to the established of Pareto frontier.

Our lower bound is constructed by relating the regrets between two (sets of) closely related problems: We show that any algorithm achieves good performance on one of them necessarily performs bad on the other one. Similar ideas are previously explored in continuum-armed bandit and K-armed bandits (Locatelli and Carpentier, 2018; Hadiji, 2019; Zhu and Nowak, 2020). We study the linear case with model selection and establish the following lower bound.[3] We use $\omega \in \mathcal{H}_T(\alpha)$ to represent any bandit regret minimization problem with time horizon T and hardness level at most α (i.e., $d_\star \leq T^\alpha$).

[2]Technically, the regret bound is only achieved by a more complicated algorithm SupLinUCB. However, it's common to use LinUCB as the practical algorithm. See Chu et al. (2011) for detailed discussion.

[3]Our lower bound is quantitatively similar to the one studied in K-armed bandits with multiple best arms (Zhu and Nowak, 2020).

Theorem 1. *Consider any $0 \leq \alpha' < \alpha \leq 1$ and $B > 0$ satisfying $T^\alpha \leq B$ and $\lfloor T^\alpha/2 \rfloor \geq \max\{T^\alpha/4, T^{\alpha'}, 2\}$. If an algorithm is such that $\sup_{\omega \in \mathcal{H}_T(\alpha')} R_T \leq B$, then the regret of the same algorithm must satisfy*

$$\sup_{\omega \in \mathcal{H}_T(\alpha)} R_T \geq c T^{1+\alpha} B^{-1}, \tag{1}$$

with a universal constant c.

Our lower bound delivers important messages to the model selection problem in linear bandits. Most of the previous efforts and open problems (Foster et al., 2019; Pacchiano et al., 2020b) are made to match the usual non-adaptive regret with known $d_\star$ (or α). Our lower bound, however, provides a negative answer towards the open problem of achieving regret guarantees $\widetilde{O}(T^{(1+\alpha)/2})$ simultaneously for all hardness levels α. We interpret this result next.

Interpretation of Theorem 1. Fix any linear bandit algorithm. We consider two problem instances with different hardness levels $0 \leq \alpha' < \alpha \leq 1$ (and satisfy the constrains in Theorem 1). On one hand, if the algorithm is such that $\sup_{\omega \in \mathcal{H}_T(\alpha')} R_T = \widetilde{\omega}(T^{(1+\alpha')/2})$, we know that this algorithm is already sub-optimal over problems with hardness level α'. On the other hand, suppose that the algorithm achieves the desired regret $\widetilde{O}(T^{(1+\alpha')/2})$ over $\mathcal{H}_T(\alpha')$. Eq. (1) then tells us that $\sup_{\omega \in \mathcal{H}_T(\alpha)} R_T = \widetilde{\Omega}(T^{(1+2\alpha-\alpha')/2})$, which is clearly larger than the desired regret $\widetilde{O}(T^{(1+\alpha)/2})$ over problems with hardness level α.

If we aim at providing regret bounds with only poly-logarithmic dependence on K in linear bandits (which is usually the case for linear bandits with finite action set (Auer, 2002; Chu et al., 2011)). our lower bound also provides a negative answer to the open problem of achieving a weaker guarantee $\widetilde{O}(T^\gamma d_\star^{1-\gamma}) = \widetilde{O}(T^{\gamma+\alpha(1-\gamma)})$, with $\gamma \in [1/2, 1)$ (Foster et al., 2019), simultaneously for all $d_\star$ (or α).

In the model selection setting, the performance of any algorithm should be a function of the hardness level α: The algorithm needs to adapt the *unknown* α. To further explore the fundamental limit for model selection in linear bandits, following Hadiji (2019); Zhu and Nowak (2020), we define rate function to capture the performance of any algorithm (in terms of its regret dependence on polynomial terms of T).

Definition 1. *Let $\theta : [0, 1] \to [0, 1]$ denote a non-decreasing function. An algorithm achieves the rate function θ if*

$$\forall \varepsilon > 0, \forall \alpha \in [0, 1], \quad \limsup_{T \to \infty} \frac{\sup_{\omega \in \mathcal{H}_T(\alpha)} R_T}{T^{\theta(\alpha)+\varepsilon}} < +\infty.$$

Since there may not always exist a pointwise ordering over rate functions, we consider the notion of Pareto optimality over rate functions.

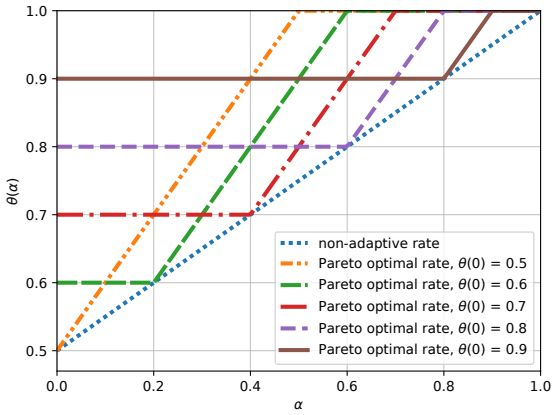


Figure 1: Pareto Optimal Rates for Model Selection in Linear Bandits.

Definition 2. *A rate function θ is Pareto optimal if it is achieved by an algorithm, and there is no other algorithm achieving a strictly smaller rate function θ' in the pointwise order. An algorithm is Pareto optimal if it achieves a Pareto optimal rate function.*

We establish the following lower bound for any rate function that can be achieved by an algorithm designed for model selection in linear bandits.

Theorem 2. *Suppose a rate function θ is achieved by an algorithm, then we must have*

$$\theta(\alpha) \geq \min\{\max\{\theta(0), 1+\alpha-\theta(0)\}, 1\}, \qquad (2)$$

with $\theta(0) \in [1/2, 1]$.

Fig. 1 illustrates the Pareto frontiers for the model selection problem in linear bandits: The blue dashed line represents the non-adaptive rate function achieved by LinUCB with *known* α; Other curves represent Pareto optimal rate functions (achieved by Pareto optimal algorithms introduced in Section 4) for the model selection problem in linear bandits. Fig. 1 implies that no algorithm can achieve the non-adaptive rate simultaneously for all α: any Pareto optimal curve has to be higher than the non-adaptive curve at least at some points.

Pareto Optimality of Corral-Type of Algorithms. We remark that, accompanied with our lower bound, the Smooth Corral algorithm presented in Pacchiano et al. (2020b) is also Pareto optimal. While only a $\widetilde{O}(d_\star\sqrt{T})$ regret bound is presented for the Smooth Corral algorithm, upon inspection of their analysis, we find that Smooth Corral can actually match the lower bound in Eq. (2) by setting the learning rate as $\eta = T^{-\theta(0)}$, for any $\theta(0) \in [1/2, 1)$. See Appendix C.3 for a detailed discussion.

Although the Corral-type of algorithm (e.g., Smooth Corral) is Pareto optimal, they may not be effective in problems with specific structures (Papini et al., 2021). We introduce a new Pareto optimal algorithm in the next section, which is shown to be more practical than Smooth Corral regarding model selection problems in linear bandits (see Section 5).

4 PARETO OPTIMALITY WITH NEW IDEAS

We develop a Pareto optimal algorithm LinUCB++ (Algorithm 1) that operates fundamentally different from algorithms rely on "testing" (Foster et al., 2019; Ghosh et al., 2020) or "corralling" (Pacchiano et al., 2020b; Agarwal et al., 2017). Our algorithm is built upon the construction of virtual mixture-arms (Hadiji, 2019; Zhu and Nowak, 2020) and virtual dimensions.

We first introduce some additional notations. For any vector $a \in \mathbb{R}^d$ and $0 \leq d_i \leq d$, we use $a^{(d_i)} \in \mathbb{R}^{d_i}$ to represent the truncated version of a that only keeps the first d_i dimensions. We also use $[a_1; a_2]$ to represent the concatenated vector of a_1 and a_2. We denote $\mathcal{A}^{(d_i)} \subseteq \mathbb{R}^{d_i}$ as the "truncated" action (multi-) set, i.e., $\mathcal{A}^{(d_i)} = \{a^{(d_i)} \in \mathbb{R}^{d_i} : a \in \mathcal{A}\}$. One can always *manually* construct the truncated action set $\mathcal{A}^{(d_i)}$ and *pretend* to work with arms with truncated feature representations (though their expected rewards may not be aligned with the truncated feature representations).

Algorithm 1 LinUCB++

Input: Time horizon T and a user-specified parameter $\beta \in [1/2, 1)$.

1: **Set:** $p = \lceil \log_2 T^\beta \rceil$, $d_i = \min\{2^{p+2-i}, d\}$ and $\Delta T_i = \min\{2^{p+i}, T\}$.
2: **for** $i = 1, \ldots, p$ **do**
3: Run LinUCB on a set of arms S_i for ΔT_i rounds, where S_i contains all arms in $\mathcal{A}^{(d_i)}$ *and* a set of virtual mixture-arms constructed from previous iterations, i.e., $\{\widetilde{\nu}_j\}_{j<i}$. LinUCB is operated with respect to an modified linear bandit problem with added virtual dimensions.
4: Construct a virtual mixture-arm $\widetilde{\nu}_i$ based on empirical sampling frequencies of LinUCB above.
5: **end for**

We present LinUCB++ in Algorithm 1. LinUCB++ operates in iterations with geometrically increasing length, and it invokes LinUCB (SupLinUCB) (Chu et al., 2011; Auer, 2002) with (roughly) geometrically decreasing dimensions. The core steps of LinUCB++ are summa-

rized at lines 3 and 4 in Algorithm 1, which consists of construction of virtual mixture-arms and virtual dimensions (the modified linear bandit problem). We next explain in detail these two core ideas.

The Virtual Mixture-Arm. After each iteration j, let $\widehat{p}_j$ denote the vector of empirical sampling frequencies of the arms in that iteration, i.e., the k-th element of $\widehat{p}_j$ is the number of times arm k, including all previously constructed virtual mixture-arms, was sampled in iteration j divided by the total number of time steps ΔT_j. The virtual mixture-arm for iteration j is the $\widehat{p}_j$-mixture of the arms played in iteration j, denoted by $\widetilde{\nu}_j$. When LinUCB samples from $\widetilde{\nu}_j$, it first draws a real arm $j_t \sim \widehat{p}_j$ with feature representation A_t,[4] then pull the real arm A_t to obtain a reward $X_t = \langle \theta_\star, A_t \rangle + \eta_t$. The expected reward of virtual mixture-arm $\widetilde{\nu}_j$ can be expressed as $\langle \theta_\star, a_\star \rangle - R_{\Delta T_j}/\Delta T_j$, where we use $R_{\Delta T_j}$ to denote the expected regret suffered in iteration j. Virtual mixture-arms $\widetilde{\nu}_j$ provide a convenient summary of the information gained in the j-th iterations so that we don't need to explore arms in the (effectively) d_j dimensional space again.

Linear Bandits with Added Virtual Dimensions. We consider the linear bandit problem in iteration i, where each arm in $\mathcal{A}^{(d_i)}$ is viewed as a vector in $\mathbb{R}^{d_i}$. Besides this simple truncation, we lift the feature representation of each arm into a slightly higher dimensional space to include the $i-1$ virtual mixture-arms constructed in previous iterations (i.e., adding virtual dimensions). More specifically, we augment $i-1$ zeros to the feature representation of each truncated real arm $a \in \mathcal{A}^{(d_i)}$; we also view each virtual mixture-arm $\widetilde{\nu}_j$ as a $d_i + i - 1$ dimensional vector $\widetilde{\nu}_j^{\langle d_i \rangle}$ with its $(d_i + j)$-th entry being 1 and all other entries being 0. As a result, LinUCB will operate on an modified linear bandit problem with action set $\mathcal{A}^{\langle d_i \rangle} \subseteq \mathbb{R}^{d_i+i-1}$, where $\mathcal{A}^{\langle d_i \rangle} = \{[a^{(d_i)}; 0] \in \mathbb{R}^{d_i+i-1} : a \in \mathcal{A}\} \cup \{\widetilde{\nu}_j^{\langle d_i \rangle}\}$, and $|\mathcal{A}^{\langle d_i \rangle}| = K + i - 1$. Working with added virtual dimensions allows us to incorporate information stored in virtual mixture-arms without too much additional cost since $i \le p = O(\log T)$.

Remark 1. *Previous application of the virtual mixture-arms only works in continuum-armed bandits or K-armed bandits (Zhu and Nowak, 2020; Hadiji, 2019), where no further modifications are needed to incorporate information stored in virtual mixture-arms. Besides the construction of the virtual dimension, we also provide another way to incorporate the virtual mixture-arms in Section 4.2. These modifications are important for the linear bandit case.*

[4] If the index of another virtual mixture-arm is returned, we sample from that virtual mixture-arm until a real arm is returned.

4.1 Analysis

We first analyze LinUCB++ with the following assumption. A modified version of LinUCB++ (Algorithm 2) is provided in Section 4.2 and analyzed without the assumption.

Assumption 1. *An action set $\mathcal{A} \subseteq \mathbb{R}^d$ is expressive if we have $a^{[d_i]} = [a^{(d_i)}; 0] \in \mathcal{A}$ for any $a \in \mathcal{A}$ and $d_i < d$.*

Assumption 1 is naturally satisfied when certain combinatorial structure and ranking information are associated with the action set. This is best explained with an example. Suppose the arms are consumer products and each has a subset of d possible features, i.e., the arms are *binary vectors* in $\mathbb{R}^d$ indicating the features of the product (the combinatorial aspect). Think of the features as being ordered from base-level features to high-end features (the ranking information). In this case, Assumption 1 means that if a product $a \in \mathcal{A}$, then $\mathcal{A}$ also contains all products with fewer high-end features, i.e., truncations of action a. We also make the following two comments regarding Assumption 1.

1. The action set we used to construct the lower bound in Theorem 1 can be made expressive, as noted in Remark 2 in Appendix A.1;
2. Although the original version of LinUCB++ is analyzed with Assumption 1, it shows strong empirical performance even without such assumption (see Section 5).

Equipped with Assumption 1, we can replace the "truncated" action set $\mathcal{A}^{(d_i)}$ with real arms that actually exist in the action set. As a result, the linearity in rewards is preserved in the modified linear bandit problem in $\mathbb{R}^{d_i+i-1}$ with added virtual dimensions. The modified linear bandit problem is associated with reward vector $\theta_\star^{\langle d_i \rangle} = [\theta_\star^{(d_i)}; \widetilde{\mu}_1; \ldots; \widetilde{\mu}_{i-1}] \in \mathbb{R}^{d_i+i-1}$, where we use $\widetilde{\mu}_i = \langle \theta_\star, a_\star \rangle - R_{\Delta T_i}/\Delta T_i$ to denote the expected reward of mixture-arm $\widetilde{\nu}_i$. In the i-th iteration of LinUCB++, we invoke LinUCB to learn reward vector $\theta_\star^{\langle d_i \rangle} \in \mathbb{R}^{d_i+i-1}$, which takes worst case regret proportional to $d_i + i - 1$ instead of the ambient dimension d.

Since there are at most $O(\log T)$ iterations of LinUCB++, we only need to upper bound its regret at each iteration. Suppose S_i is the set of actions that LinUCB++ is working on at iteration i. We use $a_{S_i} = \arg\max_{a \in S_i} \langle \theta_\star, a \rangle$ to denote the arm with the highest expected reward; and decompose the regret into

approximation error and learning error:

$$R_{\Delta T_i} = \underbrace{\mathbb{E}\left[\Delta T_i \cdot \langle \theta_\star, a_\star - a_{S_i}\rangle\right]}_{\text{expected approximation error due to the selection of } S_i} + \underbrace{\mathbb{E}\left[\sum_{t=1}^{\Delta T_i} \langle \theta_\star, a_{S_i} - A_t\rangle\right]}_{\text{expected learning error due to the sampling rule } \{A_t\}_{t=1}^T}. \tag{3}$$

The Learning Error. At each iteration i, LinUCB++ invokes LinUCB on a linear bandit problem in $\mathbb{R}^{d_i+i-1}$ for ΔT_i time steps, where d_i and ΔT_i are specifically chosen such that $d_i \Delta T_i \le \widetilde{O}(T^{2\beta})$. The learning error is then upper bounded by $\widetilde{O}(\sqrt{d_i \Delta T_i}) = \widetilde{O}(T^\beta)$ based on the regret bound of LinUCB (the norm of reward vector $\theta_\star^{\langle d_i\rangle}$ increases with iteration i due to added virtual dimensions, we deal with that in Appendix B.2).

The Approximation Error. Let $i_\star \in [p]$ denote the largest integer such that $d_{i_\star} \ge d_\star$. For iterations $i \le i_\star$, since $\theta_\star$ only has its first $d_\star \le d_i$ coordinates being non-zero, we have $\max_{a \in \mathcal{A}^{\langle d_i\rangle}} \{\langle \theta_\star^{\langle d_i\rangle}, a\rangle\} = \langle \theta_\star, a_\star\rangle$ and the expected approximation error equals zero. As a result, we upper bound the expected regret for iteration $i \le i_\star$ by its expected learning error, i.e., $R_{\Delta T_i} \le \widetilde{O}(T^\beta)$. Now consider any iteration $i > i_\star$. Since the virtual mixture-arm $\widetilde{\nu}_{i_\star}$ is constructed by then, and its expected reward is $\widetilde{\mu}_{i_\star} = \langle \theta_\star, a_\star\rangle - R_{\Delta T_{i_\star}}/\Delta T_{i_\star}$, we can further bound the expected approximation error by $\Delta T_i R_{\Delta T_{i_\star}}/\Delta T_{i_\star} = \widetilde{O}(T^{1+\alpha-\beta})$ (detailed in Appendix B.5).

We now present the formal guarantees of LinUCB++.

Theorem 3. *Run LinUCB++ with time horizon T and any user-specified parameter $\beta \in [1/2, 1)$ leads to the following upper bound on the expected regret:*

$$\sup_{\omega \in \mathcal{H}_T(\alpha)} R_T \le O\left(\log^{7/2}(KT\log T) \cdot T^{\min\{\max\{\beta, 1+\alpha-\beta\}, 1\}}\right).$$

The next theorem shows that LinUCB++ is Pareto optimal with *any* input $\beta \in [1/2, 1)$.

Theorem 4. *The rate function achieved by LinUCB++ with any input $\beta \in [1/2, 1)$, i.e.,*

$$\theta_\beta : \alpha \mapsto \min\{\max\{\beta, 1+\alpha-\beta\}, 1\}, \tag{4}$$

is Pareto optimal.

4.2 Removing Assumption 1

Assumption 1 is used to preserve linearity when working with truncated action sets. In general, one should not expect to deal with misspecified linear bandits without extra cost: Lattimore et al. (2020) develop a regret lower bound $\Omega(\varepsilon\sqrt{d}T)$ for misspecified linear bandits with misspecification level ε. The lower bound scales linearly with T if there is no extra control/assumptions on the misspecified level ε.

Going back to our algorithm, however, we notice that there is a special structure in the source of misspecifications: the virtual-mixture arms are never misspecified. We explore this fact and provide a modified version of Algorithm 1 (i.e., Algorithm 2) that works *without* Assumption 1 and is Pareto optimal. The modified algorithm is less practical since it invokes Smooth Corral as a subroutine (see Section 5).

Algorithm 2 LinUCB++ with Corral

Input: Time horizon T and a user-specified parameter $\beta \in [1/2, 1)$.

1: **Set:** $p = \lceil \log_2 T^\beta \rceil$, $d_i = \min\{2^{p+2-i}, d\}$ and $\Delta T_i = \min\{2^{p+i}, T\}$.
2: **for** $i = 1, \ldots, p$ **do**
3: Construct two (smoothed) base algorithms: (1) a LinUCB algorithm working with action set $\mathcal{A}^{(d_i)}$; and (2) a UCB algorithm working with the set of virtual-mixture arms (if any), i.e., $\{\widetilde{\nu}_j\}_{j<i}$. Invoke Smooth Corral as the master algorithm with learning rate $\eta = 1/\sqrt{d_i \Delta T_i}$.
4: Construct a virtual mixture-arm $\widetilde{\nu}_i$ based on the empirical sampling frequencies.
5: **end for**

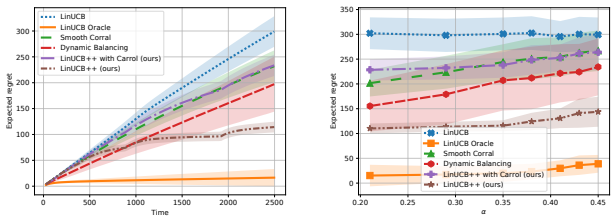
We defer detailed discussion on Algorithm 2 and Smooth Corral to Appendix C. We state the guarantee of Algorithm 2 next.

Theorem 5. *With any input $\beta \in [1/2, 1)$, the rate function achieved by Algorithm 2 (without Assumption 1) is Pareto optimal.*

5 EXPERIMENTS

We empirically evaluate our algorithms LinUCB++ and LinUCB++ with Corral in this section. We find that LinUCB++ enjoys superior performance compared to existing algorithms. Although Assumption 1 is needed in the analysis of LinUCB++, our experiments show that LinUCB++ is fairly robust to the existence of such assumption.

We compare LinUCB++ and LinUCB++ with Corral with four baselines: LinUCB (Chu et al., 2011), LinUCB Oracle, Smooth Corral (Pacchiano et al., 2020b) and Dy-



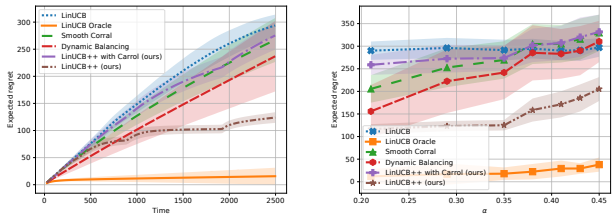
(a) (b)

Figure 2: Experiments *without* Assumption 1: (a) Regret Curve Comparison with $\alpha \approx 0.32$. (b) Regret Comparison with Different α.

namic Balancing (Cutkosky et al., 2021). LinUCB is the standard linear bandit algorithm that works in the ambient dimension $\mathbb{R}^d$. LinUCB Oracle represents the oracle version of LinUCB: it takes the knowledge of the instrinsic dimension $d_\star$ and works in $\mathbb{R}^{d_\star}$. Smooth Corral and Dynamic Balancing are implemented with $M = \lceil \log_2 d \rceil$ base LinUCB learners with different dimensions $d_i \in \{2^0, 2^1, \ldots, 2^{M-1}\}$; their master algorithms conduct corraling/regret balancing on top of these base learners. We set $\beta = 0.5$ in LinUCB++ and LinUCB++ with Corral.[5] The regularization parameter λ for least squares in (all subroutines/base learners of) LinUCB is set as 0.1.

We first conduct experiments *without* an expressive action set (i.e., without Assumption 1). We consider a regret minimization problem with time horizon $T = 2500$ and a bandit instance consists of $K = 1200$ arms selected uniformly at random in the $d = 600$ dimensional unit ball. We set reward parameter $\theta_\star = [1/\sqrt{d_\star}, \ldots, 1/\sqrt{d_\star}, 0, \ldots, 0]^\top \in \mathbb{R}^d$ for any intrinsic dimension $d_\star$ (see Appendix D for experiments with other choices of $\theta_\star$). To prevent lengthy exploration over exploitation, we consider Gaussian noises with zero means and 0.1 standard deviations. We evaluate each algorithm on 100 independent trials and average the results. Fig. 2a shows how regret curves of different algorithms increase. The experiment is run with intrinsic dimension $d_\star = 12$, which corresponds to a hardness level $\alpha \approx 0.32$. LinUCB++ outperforms all other algorithms (except LinUCB Oracle), and enjoys the smallest variance. LinUCB++ (almost) flatten its regret curve at early stages, indicating that it has learned the true reward parameter. Fig. 2b illustrates the performance of algorithms with respect to different intrinsic dimensions. We run experiments with $d_\star \in \{5, 10, 15, 20, 25, 30, 35\}$, and mark the corresponding α values in the plot. Across all α values, LinUCB++ shows superior performance compared to LinUCB, Smooth Corral, Dynamic Balancing and LinUCB++ with Corral. These results indicate that LinUCB++ can be practically applied without an expressive action set (thus without Assumption 1).

The empirically poor performance of Corral-type of algorithms might be due to the fact that they need to balance over multiple base algorithms. On the other hand, LinUCB++ invokes only one LinUCB subroutine at each iteration. Although the subroutine is restarted at the beginning of each iteration, it runs on (roughly) geometrically decreasing dimensions. Such efficient learning procedure is backed by our construction of virtual mixture-arms and virtual dimensions.



(a) (b)

Figure 3: Similar Experiment Setups to Those Shown in Fig. 2, but with An Expressive Action Sets.

We also run experiments *with* expressive action sets. We first generate $K = 800$ arms uniformly at random from a $d = 400$ dimensional unit ball. The action set is then made *expressive* by adding actions with truncated features.[6] We provide the expressive action set to all algorithms since the best reward could be achieved by a truncated arm. Other experimental setups are similar to the ones described before. The shape of curves appearing in both Fig. 3a and Fig. 3b are resembles the ones in Fig. 2, and LinUCB++ outperforms LinUCB, Smooth Corral, Dynamic Balancing and LinUCB++ with Corral. One slight difference is that Smooth Corral, Dynamic Balancing, LinUCB++ with Corral and LinUCB++ have relatively worse performance when as α increases: The regret curves (in Fig. 3b) increase at faster speeds. Smooth Corral, Dynamic Balancing and LinUCB++ with Corral are outperformed by the standard LinUCB when the hardness level α gets large.

6 DISCUSSION

We study the model selection problem in linear bandits where the goal is to adapt to the *unknown* intrinsic

[5] In practice, we recommend taking $\beta = (1 + \widehat{\alpha})/2$ if an estimation $\widehat{\alpha}$ (of α) is available; otherwise, we empirically find that taking $\beta = 0.5$ works well.

[6] We only truncate actions with respect to d_i s selected by LinUCB++ to avoid the computational burden of dealing with a large number of actions.

dimension $d_\star$, rather than suffering from regret proportional to the ambient dimension d. We establish a lower bound indicating that adaptation to the unknown intrinsic dimension $d_\star$ comes at a cost: There is no algorithm that can achieve the regret bound $\widetilde{O}(\sqrt{d_\star T})$ simultaneously for all values of $d_\star$. Under a mild assumption, we design a Pareto optimal algorithm, with ideas fundamentally different from "testing" (Foster et al., 2019; Ghosh et al., 2020) and "corralling" (Pacchiano et al., 2020b; Agarwal et al., 2017), to bear on the model selection problem in linear bandits. We also provide a workaround to remove the assumption. Experimental evaluations show superior performance of our main algorithm compared to existing ones.

Although linear bandits with a fixed action set are commonly studied in the literature (Lattimore et al., 2020; Wagenmaker et al., 2021), an interesting direction is to generalize LinUCB++ to the contextual setting. The current version of LinUCB++ works in the setting with adversarial contexts under the following two additional assumptions: (1) we have a nested sequence of action sets $\mathcal{A}_t \subseteq \mathcal{A}_{t+1}$ with $|\mathcal{A}_T| \leq K$; and (2) one of the best/near-optimal arm belongs to $\mathcal{A}_1$. How to remove/weaken these assumptions is left to future work. We also remark that, after our initial (arXiv) publication, Marinov and Zimmert (2021) established the Pareto frontier for general contextual bandits, providing a negative answer to open problems raised in Foster et al. (2020).

Acknowledgements

We thank anonymous reviewers for helpful comments. This work is partially supported by NSF grant 1934612 and ARMY MURI grant W911NF-15-1-0479.

References

Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Online-to-confidence-set conversions and application to sparse stochastic bandits. In *Artificial Intelligence and Statistics*, pages 1–9, 2012.

Yasin Abbasi-Yadkori, Aldo Pacchiano, and My Phan. Regret balancing for bandit and rl model selection. *arXiv preprint arXiv:2006.05491*, 2020.

Naoki Abe, Alan W Biermann, and Philip M Long. Reinforcement learning with immediate rewards and linear hypotheses. *Algorithmica*, 37(4):263–293, 2003.

Alekh Agarwal, Haipeng Luo, Behnam Neyshabur, and Robert E Schapire. Corralling a band of bandit algorithms. In *Conference on Learning Theory*, pages 12–38. PMLR, 2017.

Rajeev Agrawal. The continuum-armed bandit problem. *SIAM journal on control and optimization*, 33 (6):1926–1951, 1995.

Raman Arora, Teodor V Marinov, and Mehryar Mohri. Corralling stochastic bandit algorithms. *arXiv preprint arXiv:2006.09255*, 2020.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Sébastien Bubeck, Gilles Stoltz, and Jia Yuan Yu. Lipschitz bandits without the lipschitz constant. In *International Conference on Algorithmic Learning Theory*, pages 144–158. Springer, 2011.

Alexandra Carpentier and Rémi Munos. Bandit theory meets compressed sensing for high dimensional stochastic linear bandit. In *Artificial Intelligence and Statistics*, pages 190–198, 2012.

Niladri Chatterji, Vidya Muthukumar, and Peter Bartlett. Osom: A simultaneously optimal algorithm for multi-armed and linear contextual bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 1844–1854, 2020.

Arghya Roy Chaudhuri and Shivaram Kalyanakrishnan. Quantile-regret minimisation in infinitely many-armed bandits. In *UAI*, pages 425–434, 2018.

Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214, 2011.

Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische mathematik*, 31 (4):377–403, 1978.

Ashok Cutkosky and Kwabena Boahen. Online learning without prior information. *arXiv preprint arXiv:1703.02629*, 2017.

Ashok Cutkosky and Francesco Orabona. Black-box reductions for parameter-free online learning in banach spaces. In *Conference On Learning Theory*, pages 1493–1529, 2018.

Ashok Cutkosky, Abhimanyu Das, and Manish Purohit. Upper confidence bounds for combining stochastic bandits. *arXiv preprint arXiv:2012.13115*, 2020.

Ashok Cutkosky, Christoph Dann, Abhimanyu Das, Claudio Gentile, Aldo Pacchiano, and Manish Purohit. Dynamic balancing for model selection in bandits and rl. In *International Conference on Machine Learning*, pages 2276–2285. PMLR, 2021.

Yash Deshpande and Andrea Montanari. Linear bandits in high dimension and recommendation systems. In *2012 50th Annual Allerton Conference*

on Communication, Control, and Computing (Allerton), pages 1750–1754. IEEE, 2012.

Dylan J Foster, Satyen Kale, Mehryar Mohri, and Karthik Sridharan. Parameter-free online learning via model selection. In *Advances in Neural Information Processing Systems*, pages 6020–6030, 2017.

Dylan J Foster, Akshay Krishnamurthy, and Haipeng Luo. Model selection for contextual bandits. In *Advances in Neural Information Processing Systems*, pages 14741–14752, 2019.

Dylan J Foster, Akshay Krishnamurthy, and Haipeng Luo. Open problem: Model selection for contextual bandits. *arXiv preprint arXiv:2006.10940*, 2020.

Avishek Ghosh, Abishek Sankararaman, and Kannan Ramchandran. Problem-complexity adaptive model selection for stochastic linear bandits. *arXiv preprint arXiv:2006.02612*, 2020.

Hédi Hadiji. Polynomial cost of adaptation for x-armed bandits. In *Advances in Neural Information Processing Systems*, pages 1027–1036, 2019.

Yanjun Han, Zhengqing Zhou, Zhengyuan Zhou, Jose Blanchet, Peter W Glynn, and Yinyu Ye. Sequential batch learning in finite-action linear contextual bandits. *arXiv preprint arXiv:2004.06321*, 2020.

Botao Hao, Tor Lattimore, and Mengdi Wang. High-dimensional sparse linear bandits. *arXiv preprint arXiv:2011.04020*, 2020.

Wouter M Koolen and Tim Van Erven. Second-order quantile methods for experts and combinatorial games. In *Conference on Learning Theory*, pages 1155–1175, 2015.

Tor Lattimore. The pareto regret frontier for bandits. *arXiv preprint arXiv:1511.00048*, 2015.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. Learning with good feature representations in bandits and in rl with a generative model. In *International Conference on Machine Learning*, pages 5662–5670. PMLR, 2020.

Andrea Locatelli and Alexandra Carpentier. Adaptivity to smoothness in x-armed bandits. In *Conference on Learning Theory*, pages 1463–1492, 2018.

Haipeng Luo and Robert E Schapire. Achieving all with no parameters: Adanormalhedge. In *Conference on Learning Theory*, pages 1286–1304, 2015.

Teodor Vanislavov Marinov and Julian Zimmert. The pareto frontier of model selection for general contextual bandits. *Advances in Neural Information Processing Systems*, 34, 2021.

Brendan McMahan and Jacob Abernethy. Minimax optimal algorithms for unconstrained linear optimization. *Advances in Neural Information Processing Systems*, 26:2724–2732, 2013.

Francesco Orabona. Simultaneous model selection and optimization through parameter-free stochastic learning. *Advances in Neural Information Processing Systems*, 27:1116–1124, 2014.

Francesco Orabona and Dávid Pál. Coin betting and parameter-free online learning. *Advances in Neural Information Processing Systems*, 29:577–585, 2016.

Urvashi Oswal, Aniruddha Bhargava, and Robert Nowak. Linear bandits with feature feedback. In *AAAI*, pages 5331–5338, 2020.

Aldo Pacchiano, Christoph Dann, Claudio Gentile, and Peter Bartlett. Regret bound balancing and elimination for model selection in bandits and rl. *arXiv preprint arXiv:2012.13045*, 2020a.

Aldo Pacchiano, My Phan, Yasin Abbasi-Yadkori, Anup Rao, Julian Zimmert, Tor Lattimore, and Csaba Szepesvari. Model selection in contextual stochastic bandit problems. *arXiv preprint arXiv:2003.01704*, 2020b.

Matteo Papini, Andrea Tirinzoni, Marcello Restelli, Alessandro Lazaric, and Matteo Pirotta. Leveraging good representations in linear contextual bandits. *arXiv preprint arXiv:2104.03781*, 2021.

Daniel Russo and Benjamin Van Roy. Satisficing in time-sensitive bandit learning. *arXiv preprint arXiv:1803.02855*, 2018.

Jun Shao. Linear model selection by cross-validation. *Journal of the American statistical Association*, 88 (422):486–494, 1993.

John Shawe-Taylor, Peter L Bartlett, Robert C Williamson, and Martin Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE transactions on Information Theory*, 44(5): 1926–1940, 1998.

M Stone. Cross-validation: A review. *Statistics: A Journal of Theoretical and Applied Statistics*, 9(1): 127–139, 1978.

Vladimir Vapnik and Alexey Chervonenkis. Theory of pattern recognition, 1974.

Vladimir N Vapnik. The nature of statistical learning theory, 1995.

Andrew Wagenmaker, Julian Katz-Samuels, and Kevin Jamieson. Experimental design for regret minimization in linear bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 3088–3096. PMLR, 2021.

Yizao Wang, Jean-Yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. *Advances in Neural Information Processing Systems*, 21, 2008.

Yinglun Zhu and Robert Nowak. On regret with multiple best arms. *arXiv preprint arXiv:2006.14785*, 2020.

Supplementary Material: Pareto Optimal Model Selection in Linear Bandits

A OMITTED PROOFS FOR SECTION 3

Besides specific treatments for linear bandits (e.g., the lower bound construction for model selection), our proofs for this section largely follow the ones developed in Hadiji (2019); Zhu and Nowak (2020). We provide details here for completeness.

A.1 Proof of Theorem 1

We consider $K+1$ linear bandit instances such that each is characterized by a reward vector $\theta_i \in \mathbb{R}^d$, $0 \le i \le K$, with different intrinsic dimensions $d_\star$ (or equivalently α). For any action $a \in \mathbb{R}^d$, we obtain a reward $r = \langle \theta_i, a\rangle + \eta$ where η is an independent $(1/2)$-sub-Gaussian noise. Time horizon T is fixed and the ambient dimension d is assumed to be large enough to avoid some trivial conflicts in the following construction (e.g., we need $d \ge T^\alpha$ to construct θ_i) . For any $0 \le \alpha' < \alpha \le 1$ so that $T^\alpha/2 \ge T^{\alpha'}$, we now provide an explicit construction of $\{\theta_i\}_{i=0}^K$ as followings, with $\Delta \in \mathbb{R}$ to be specified later.

1. Let $\theta_0 \in \mathbb{R}^d$ be any vector such that it is only supported on one of its first $\lfloor T^{\alpha'} \rfloor$ coordinates and $\|\theta_0\|_2 = \Delta/2$. The regret minimization problem with respect to θ_0 belongs to $\mathcal{H}_T(\alpha')$ by construction.
2. For any $i \in [K]$, let $\theta_i = \theta_0 + \Delta \cdot e_{\rho(i)}$ where e_j is the j-th canonical base and $\rho(i) = \lfloor T^\alpha/2 \rfloor + i$. We set $K = \lfloor T^\alpha/2 \rfloor = \Theta(T^\alpha)$ so that the regret minimization problem with respect to any θ_i belongs to $\mathcal{H}_T(\alpha)$.

We consider a common *fixed* action set $\mathcal{A} = \{a_i\}_{i=0}^K = \{\theta_0/\|\theta_0\|\} \cup \{e_{\rho(i)}\}_{i=1}^K$ for all regret minimization problems (we set $a_0 = \theta_0/\|\theta_0\|$ and $a_i = e_{\rho(i)}$ for convenience). We could notice that a_0 is the best arm with respect to θ_0, which has expected reward $\Delta/2$; and a_i is the best arm with respect to θ_i, which has expected reward Δ.

Remark 2. *The action set $\mathcal{A}$ can be made expressive by augmenting the action set with an all-zero action. The all-zero action will not affect our analysis since it always has zero expected reward.*

Remark 3. *One can also add other canonical bases into the action set $\mathcal{A}$ so that $\{\theta_i\}_{i=1}^K$ becomes the unique reward vector for corresponding problems. These additional actions will not affect our analysis as well since they all have zero expected reward.*

For any $t \in [T]$, the tuple of random variables $H_t = (A_1, X_1, \ldots, A_t, X_t)$ is the outcome of an algorithm interacting with an bandit instance up to time t. Let $\Omega_t = \prod_{i=1}^t (\mathcal{A} \times \mathbb{R})$ and $\mathcal{F}_t = \mathfrak{B}(\Omega_t)$; one could then define a measurable space $(\Omega_t, \mathcal{F}_t)$ for H_t. The random variables $A_1, X_1, \ldots, A_t, X_t$ that make up the outcome are defined by their coordinate projections:

$$A_t(a_1, x_1, \ldots, a_t, x_t) = a_t \quad \text{and} \quad X_t(a_1, x_1, \ldots, a_t, x_t) = x_t.$$

For any fixed algorithm/policy π and bandit instance θ_i, we are now constructing a probability measure $\mathbb{P}_{i,t}$ over $(\Omega_t, \mathcal{F}_t)$. Note that a policy π is a sequence $(\pi_t)_{t=1}^T$, where π_t is a probability kernel from $(\Omega_{t-1}, \mathcal{F}_{t-1})$ to $(\mathcal{A}, 2^{\mathcal{A}})$ with the first probability kernel $\pi_1(\omega, \cdot)$ being defined arbitrarily over $(\mathcal{A}, 2^{\mathcal{A}})$, to model the selection of the first action. For each i, we define another probability kernel $p_{i,t}$ from $(\Omega_{t-1} \times \mathcal{A}, \mathcal{F}_{t-1} \otimes 2^{\mathcal{A}})$ to $(\mathbb{R}, \mathfrak{B}(\mathbb{R}))$ that models the reward. Since the reward is distributed according to $\mathcal{N}(\theta_i^\top a_t, 1/4)$, we gives its explicit expression for any $B \in \mathfrak{B}(\mathbb{R})$ as following

$$p_{i,t}\big((a_1, x_1, \ldots, a_t), B\big) = \int_B \sqrt{\frac{2}{\pi}} \exp\big(-2(x - \theta_i^\top a_t)\big) dx.$$

The probability measure over $\mathbb{P}_{i,t}$ over $(\Omega_t, \mathcal{F}_t)$ could then be define recursively as $\mathbb{P}_{i,t} = p_{i,t}\big(\pi_t \mathbb{P}_{i,t-1}\big)$. We use $\mathbb{E}_i$ to denote the expectation taken with respect to $\mathbb{P}_{i,T}$. We have the following lemmas.

Lemma 1 (Lattimore and Szepesvári (2020))**.**

$$\mathrm{KL}\left(\mathbb{P}_{0,T}, \mathbb{P}_{i,T}\right) = \mathbb{E}_0\left[\sum_{t=1}^{T} \mathrm{KL}\left(\mathcal{N}(\theta_0^\top A_t, 1/4), \mathcal{N}\left(\theta_i^\top A_t, 1/4\right)\right)\right]. \tag{5}$$

Lemma 2 (Hadiji (2019))**.** *Let $\mathbb{P}$ and $\mathbb{Q}$ be two probability measures. For any random variable $Z \in [0,1]$, we have*

$$|\mathbb{E}_{\mathbb{P}}[Z] - \mathbb{E}_{\mathbb{Q}}[Z]| \leq \sqrt{\frac{\mathrm{KL}(\mathbb{P}, \mathbb{Q})}{2}}.$$

Theorem 1. *Consider any $0 \leq \alpha' < \alpha \leq 1$ and $B > 0$ satisfying $T^\alpha \leq B$ and $\lfloor T^\alpha/2 \rfloor \geq \max\{T^\alpha/4, T^{\alpha'}, 2\}$. If an algorithm is such that $\sup_{\omega \in \mathcal{H}_T(\alpha')} R_T \leq B$, then the regret of the same algorithm must satisfy*

$$\sup_{\omega \in \mathcal{H}_T(\alpha)} R_T \geq c\, T^{1+\alpha} B^{-1}, \tag{1}$$

with a universal constant c.

Proof. Let $N_i(T) = \sum_{t=1}^{T} \mathbb{1}\left(A_t = a_i\right)$ denote the number of times the algorithm π selects arm a_i up to time T. Let $R_{i,T}$ define the expected regret achieved by algorithm π interacting with the bandit instance θ_i. Based on the construction of bandit instances, we have

$$R_{0,T} \geq \frac{\Delta}{2} \sum_{i=1}^{K} \mathbb{E}_0\left[N_i(T)\right], \tag{6}$$

and for any $i \in [K]$

$$R_{i,T} \geq \frac{\Delta}{2}\left(T - \mathbb{E}_i[N_i(T)]\right) = \frac{T\Delta}{2}\left(1 - \frac{\mathbb{E}_i[N_i(T)]}{T}\right). \tag{7}$$

According to Lemma 1 and the calculation of KL-divergence between two Gaussian distributions, we further have

$$\begin{aligned}
\mathrm{KL}(\mathbb{P}_{0,T}, \mathbb{P}_{i,T}) &= \mathbb{E}_0\left[\sum_{t=1}^{T} \mathrm{KL}\left(\mathcal{N}(\theta_0^\top A_t, 1/4), \mathcal{N}\left(\theta_i^\top A_t, 1/4\right)\right)\right] \\
&= \mathbb{E}_0\left[\sum_{t=1}^{T} 2\left\langle \theta_i - \theta_0, A_t \right\rangle^2\right] \\
&= 2\mathbb{E}_0\left[N_i(T)\right]\Delta^2,
\end{aligned} \tag{8}$$

where Eq. (8) comes from the fact that $\theta_i = \theta_0 + \Delta \cdot e_{\rho(i)}$ and the only arm in $\mathcal{A}$ with non-zero value on the $\rho(i)$-th coordinate is $a_i = e_{\rho(i)}$, with $\langle \theta_i - \theta_0, a_i \rangle = \Delta$.

We now consider the average regret over $i \in [K]$:

$$\begin{aligned}
\frac{1}{K}\sum_{i=1}^{K} R_{i,T} &\geq \frac{T\Delta}{2}\left(1 - \frac{1}{K}\sum_{i=1}^{K}\frac{\mathbb{E}_i[N_i(T)]}{T}\right) \\
&\geq \frac{T\Delta}{2}\left(1 - \frac{1}{K}\sum_{i=1}^{K}\left(\frac{\mathbb{E}_0[N_i(T)]}{T} + \sqrt{\frac{\mathrm{KL}(\mathbb{P}_{i,T}, \mathbb{P}_{0,T})}{2}}\right)\right) && (9) \\
&= \frac{T\Delta}{2}\left(1 - \frac{1}{K}\frac{\sum_{i=1}^{K}\mathbb{E}_0[N_i(T)]}{T} - \frac{1}{K}\sum_{i=1}^{K}\sqrt{\mathbb{E}_0\left[N_i(T)\right]\Delta^2}\right) && (10) \\
&\geq \frac{T\Delta}{2}\left(1 - \frac{1}{K} - \sqrt{\frac{\sum_{i=1}^{K}\mathbb{E}_0\left[N_i(T)\right]\Delta^2}{K}}\right) && (11) \\
&\geq \frac{T\Delta}{2}\left(1 - \frac{1}{K} - \sqrt{\frac{2\Delta R_{0,T}}{K}}\right) && (12) \\
&\geq \frac{T\Delta}{2}\left(\frac{1}{2} - \sqrt{\frac{2\Delta B}{K}}\right), && (13)
\end{aligned}$$

where Eq. (9) comes from applying Lemma 2 with $Z = N_i(T)/T$ and $\mathbb{P} = \mathbb{P}_{i,T}$ and $\mathbb{Q} = \mathbb{P}_{0,T}$; Eq. (10) comes from Lemma 1; Eq. (11) comes from concavity of $\sqrt{\cdot}$; Eq. (12) comes from Eq. (6); and finally Eq. (13) comes from the fact that $K \geq 2$ by construction and the assumption that $R_{0,T} \leq B$.

To obtain a large value for Eq. (13), one could maximize Δ while still make sure $\sqrt{2\Delta B/K} \leq 1/4$. Set $\Delta = 2^{-5}KB^{-1}$, following Eq. (13), we obtain

$$\begin{aligned}
\frac{1}{K}\sum_{i=1}^{K} R_{i,T} &\geq 2^{-8}TKB^{-1} \\
&= 2^{-8}T\left\lfloor T^{\alpha}/2\right\rfloor B^{-1} && (14) \\
&\geq 2^{-10}T^{1+\alpha}B^{-1}, && (15)
\end{aligned}$$

where Eq. (14) comes from the construction of K; and Eq. (15) comes from the assumption that $\lfloor T^{\alpha}/2\rfloor \geq T^{\alpha}/4$.

It is clear that any action $a \in \mathcal{A}$ satisfies $\|a\| \leq 1$ by construction, we now only need to make sure that $\|\theta_i\| \leq 1$ as well. Notice that $\|\theta_i\| \leq \sqrt{5}\Delta/2$ by construction, we only need to make sure $\Delta = 2^{-5}KB^{-1} \leq 2/\sqrt{5}$. Since on one hand $K = \lfloor T^{\alpha}/2\rfloor \leq T^{\alpha}$, and on the other hand $T^{\alpha} \leq B$ by assumption, we have $\Delta = 2^{-5}KB^{-1} \leq 2^{-5} < 2/\sqrt{5}$, as desired. □

A.2 Proof of Theorem 2

Lemma 3. *Suppose an algorithm achieves rate function $\theta(\alpha)$ on $\mathcal{H}_T(\alpha)$, then for any $0 < \alpha \leq 1$ such that $\alpha \leq \theta(0)$, we have*

$$\theta(\alpha) \geq 1 + \alpha - \theta(0). \tag{16}$$

Proof. Fix $0 \leq \alpha \leq \theta(0)$. For any $\varepsilon > 0$, there exists constant c_1 and c_2 such that

$$\sup_{\omega \in \mathcal{H}_T(0)} R_T \leq c_1 T^{\theta(0)+\varepsilon} \quad \text{and} \quad \sup_{\omega \in \mathcal{H}_T(\alpha)} R_T \leq c_2 T^{\theta(\alpha)+\varepsilon},$$

for sufficiently large T. Let $B = \max\{c_1, 1\} \cdot T^{\theta(0)+\varepsilon}$, we could see that $T^{\alpha} \leq T^{\theta(0)} \leq B$ holds by assumption. For T large enough, the condition $\lfloor T^{\alpha}/2\rfloor \geq \max\{T^{\alpha}/4, T^0, 2\}$ of Theorem 1 holds, and we then have

$$c_2 T^{\theta(\alpha)+\varepsilon} \geq 2^{-10}T^{1+\alpha}\left(\max\{c_1, 1\} \cdot T^{\theta(0)+\varepsilon}\right)^{-1} = 2^{-10}T^{1+\alpha-\theta(0)-\varepsilon}/\max\{c_1, 1\}.$$

For T sufficiently large, we then must have

$$\theta(\alpha) + \varepsilon \geq 1 + \alpha - \theta(0) - \varepsilon.$$

Let $\varepsilon \to 0$ leads to the desired result. □

Theorem 2. *Suppose a rate function θ is achieved by an algorithm, then we must have*

$$\theta(\alpha) \geq \min\{\max\{\theta(0), 1 + \alpha - \theta(0)\}, 1\}, \tag{2}$$

with $\theta(0) \in [1/2, 1]$.

Proof. For any adaptive rate function θ achieved by an algorithm, we first notice that $\theta(\alpha) \geq \theta(\alpha')$ for any $0 \leq \alpha' \leq \alpha \leq 1$ as $\mathcal{H}_T(\alpha') \subseteq \mathcal{H}_T(\alpha)$, which also implies $\theta(\alpha) \geq \theta(0)$. From Lemma 3, we further obtain $\theta(\alpha) \geq 1 + \alpha - \theta(0)$ if $0 < \alpha \leq \theta(0)$. Thus, for any $\alpha \in (0, \theta(0)]$, we have

$$\theta(\alpha) \geq \max\{\theta(0), 1 + \alpha - \theta(0)\}. \tag{17}$$

Note that this indicates $\theta(\theta(0)) = 1$ since we trivially have $R_T \leq T$. For any $\alpha \in [\theta(0), 1]$, we have $\theta(\alpha) \geq \theta(\theta(0)) = 1$, which also leads to $\theta(\alpha) = 1$ for $\alpha \in [\theta(0), 1]$. To summarize, we obtain the desired result in Eq. (2). We have $\theta(0) \in [1/2, 1]$ as the minimax optimal rate among problems in $\mathcal{H}_T(0)$ is $1/2$ (Chu et al., 2011). □

B OMITTED PROOFS FOR SECTION 4

B.1 The Virtual-Mixture Arm

The expected reward of virtual mixture-arm $\widetilde{\nu}_j$ can be expressed as the total expected reward obtained in iteration j divided by the corresponding time horizon ΔT_j:

$$\widetilde{\mu}_j = \mathbb{E}[\widetilde{\nu}_j] = \mathbb{E}\left[\sum_{t \text{ in iteration } j} X_t\right] / \Delta T_j = \langle \theta_\star, a_\star \rangle - R_{\Delta T_j} / \Delta T_j \in [-1, 1], \tag{18}$$

where we use $R_{\Delta T_j}$ to denote the expected regret suffered in iteration j. Let X_t be the reward obtained by pulling the virtual arm $\widetilde{\nu}_j$ (with A_t being the feature representation of the drawn real arm), we then know that $X_t - \widetilde{\mu}_j$ is $\sqrt{2}$-sub-Gaussian since $X_t - \widetilde{\mu}_j = (X_t - \langle \theta_\star, A_t \rangle) + (\langle \theta_\star, A_t \rangle - \widetilde{\mu}_j) = \eta_t + (\langle \theta_\star, A_t \rangle - \widetilde{\mu}_j)$: η_t is 1-sub-Gaussian by assumption and $(\langle \theta_\star, A_t \rangle - \widetilde{\mu}_j)$ is 1-sub-Gaussian due to boundedness $\langle \theta_\star, A_t \rangle \in [-1, 1]$ and $\mathbb{E}[\langle \theta_\star, A_t \rangle] = \widetilde{\mu}_j$.

B.2 Modifications of LinUCB

Recall that, under Assumption 1, the linear reward structure is preserved in the modified linear bandit problem that LinUCB will be working on in Algorithm 1. Two main differences in the modified linear bandit problem from the original setting considered in Chu et al. (2011) are: (1) we will be working with $\sqrt{2}$-sub-Gaussian noise while they deal with strictly bounded noise; and (2) the norm of our reward parameter, i.e., $\|\theta_\star^{\langle d_i \rangle}\|$, could be as large as $1 + (p-1) = p = \lceil \log_2(T^\beta) \rceil \leq \log_2(T) + 1 \leq 2 \log T$ when $T \geq 2$.

To reduce clutters, we consider a d dimensional linear bandit with time horizon T and K actions. We consider the reward structure $X_t = \langle \theta_\star, A_t \rangle + \eta_t$, where η_t is an independent $\sqrt{2}$-sub-Gaussian noise, $\|\theta_\star\| \leq 2 \log T$ and $\|A_t\| \leq 1$. The following Theorem 6 takes care of these changes.

Theorem 6. *For the modified setting introduced above, run LinUCB with $\alpha = 2\sqrt{\log(2TK/\delta)}$ leads to an upper bound*

$$O\left(\log^2(KT \log(T)/\delta) \cdot \sqrt{dT}\right)$$

on the (pseudo) random regret with probability at least $1 - \delta$.

Corollary 1. *For the modified setting introduced above, run LinUCB with $\alpha = 2\sqrt{\log(2T^{3/2}K)}$ leads to an upper bound*

$$O\left(\log^2(KT \log(T)) \cdot \sqrt{dT}\right)$$

on the expected regret.

Proof. One can simply combine the result in Theorem 6 with $\delta = 1/\sqrt{T}$. □

It turns out that in order to prove Theorem 6, we mainly need to modify Lemma 1 in Chu et al. (2011), and the rest of the arguments go through smoothly. The changed exponent on the logarithmic term is due to $\|\theta_\star\| \le 2\log T$. We introduce the following notations. Let

$$V_0 = I \quad \text{and} \quad V_t = V_{t-1} + A_t A_t^\top$$

denote the design matrix up to time t; and let

$$\widehat{\theta}_t = V_t^{-1} \sum_{i=1}^{t} A_i X_i$$

denote the estimate of $\theta_\star$ at time t.

Lemma 4. *(modification of Lemma 1 in Chu et al. (2011)) Suppose for any fixed sequence of selected actions $\{A_i\}_{i\le t}$ the (random) rewards $\{X_i\}_{i\le t}$ are independent. Then we have*

$$\mathbb{P}\left(\forall A_{t+1} \in \mathcal{A}_{t+1} : |\langle \widehat{\theta}_t - \theta_\star, A_{t+1}\rangle| \le (\alpha + 2\log T)\sqrt{A_{t+1}^\top V_t^{-1} A_{t+1}}\right) \ge 1 - \delta/T. \tag{19}$$

Remark 4. *The requirement of (conditional) independence is guaranted by the SupLinUCB algorithm introduced in Chu et al. (2011), and is not satisfied by the vanilla LinUCB: the reveal/selection of a future arm A_{t+1} makes previous rewards $\{X_i\}_{i\le t}$ dependent. See Remark 4 in Han et al. (2020) for a detailed discussion.*

Proof. For any fixed A_t, we first notice that

$$\begin{aligned}
\left|\left\langle \widehat{\theta}_t - \theta_\star, A_{t+1}\right\rangle\right| &= \left|A_{t+1}^\top V_t^{-1} \sum_{i=1}^{t} A_i X_i - A_{t+1}^\top \theta_\star\right| \\
&= \left|A_{t+1}^\top V_t^{-1} \sum_{i=1}^{t} A_i X_i - A_{t+1}^\top V_t^{-1}\left(I + \sum_{i=1}^{t} A_i A_i^\top\right)\theta_\star\right| \\
&\le \left|\sum_{i=1}^{t} A_{t+1}^\top V_t^{-1} A_i \left(X_i - A_i^\top \theta_\star\right)\right| + \left|A_{t+1}^\top V_t^{-1}\theta_\star\right| \\
&\le \left|\sum_{i=1}^{t} A_{t+1}^\top V_t^{-1} A_i \left(X_i - A_i^\top \theta_\star\right)\right| + \left\|A_{t+1}^\top V_t^{-1}\right\| \cdot \|\theta_\star\|.
\end{aligned} \tag{20}$$

We next bound the two terms in Eq. (20) seperately.

For the first term in Eq. (20), since $\left(X_i - A_i^\top \theta_\star\right)$ is $\sqrt{2}$-sub-Gaussian and $\{X_i\}_{i\le t}$ are independent, we know that $\sum_{i=1}^{t} A_{t+1}^\top V_t^{-1} A_i \left(X_i - A_i^\top \theta_\star\right)$ is $\left(\sqrt{2\sum_{i=1}^{t}\left(A_{t+1}^\top V_t^{-1} A_i\right)^2}\right)$-sub-Gaussian. Since

$$\begin{aligned}
\sqrt{\sum_{i=1}^{t}\left(A_{t+1}^\top V_t^{-1} A_i\right)^2} &= \sqrt{\sum_{i=1}^{t} A_{t+1}^\top V_t^{-1} A_i A_i^\top V_t^{-1} A_{t+1}} \\
&\le \sqrt{A_{t+1}^\top V_t^{-1}\left(I + \sum_{i=1}^{t} A_i A_i^\top\right) V_t^{-1} A_{t+1}} \\
&= \sqrt{A_{t+1}^\top V_t^{-1} A_{t+1}},
\end{aligned}$$

according to a standard Chernoff-Hoeffding bound, we have

$$\begin{aligned}
\mathbb{P}\left(\left|\sum_{i=1}^{t} A_{t+1}^\top V_t^{-1} A_i \left(X_i - A_i^\top \theta_\star\right)\right| \ge \alpha\sqrt{A_{t+1}^\top V_t^{-1} A_{t+1}}\right) &\le 2\exp\left(-\frac{\alpha^2}{4}\right) \\
&= \frac{\delta}{TK},
\end{aligned} \tag{21}$$

where Eq. (21) is due to $\alpha = 2\sqrt{\log(2TK/\delta)}$.

For the second term in Eq. (20), we have

$$\begin{aligned}\left\|A_{t+1}^\top V_t^{-1}\right\| \cdot \|\theta_\star\| &\le 2\log T \sqrt{A_{t+1}^\top V_t^{-1} I V_t^{-1} A_{t+1}} \\ &\le 2\log T \sqrt{A_{t+1}^\top V_t^{-1}\left(I + \sum_{i=1}^{t} A_i A_i^\top\right) V_t^{-1} A_{t+1}} \\ &= 2\log T \sqrt{A_{t+1}^\top V_t^{-1} A_{t+1}}.\end{aligned} \tag{22}$$

where Eq. (22) comes from the fact that $\|\theta_\star\| \le 2\log T$.

The desired result in Eq. (19) follows from a union bound argument together with the two upper bounds derived above. □

Remark 5. *Technically, regret guarantees are for a more complicated version of LinUCB that ensures statistical independence (Chu et al., 2011). However, as recommended by Chu et al. (2011), we will use the more practical LinUCB as our subroutine.*

B.3 Notations and Preliminaries for Analysis of LinUCB++

We provide some notations and preliminaries for analysis of LinUCB++ that will be used in the following two subsections, i.e., the proofs of Lemma 5 and Theorem 3.

We define $T_i = \sum_{j=1}^{i} \Delta T_j$ so that the i-th iteration of LinUCB++ goes from $T_{i-1}+1$ to T_i. We first notice that Algorithm 1 is a valid algorithm in the sense that it selects an arm A_t for any $t \in [T]$, i.e., it does not terminate before time T: the argument is clearly true if there exists $i \in [p]$ such that $\Delta T_i = T$; otherwise, we can show that

$$T_p = \sum_{i=1}^{p} \Delta T_i = 2(2^{2p} - 1) \ge 2^{2p} \ge T,$$

for all $\beta \in [1/2, 1]$.

We use $R_{\Delta T_i} = \Delta T_i \cdot \mu_\star - \mathbb{E}[\sum_{t=T_{i-1}+1}^{T_i} X_t]$ to denote the expected cumulative regret at iteration i. Let $\mathcal{F}_i$ denote the information collected up to the end of iteration i, we further use $R_{\Delta T_i|\mathcal{F}_{i-1}}$ to represent the expected regret conditioned on $\mathcal{F}_{i-1}$ and have $\mathbb{E}[R_{\Delta T_i|\mathcal{F}_{i-1}}] = R_{\Delta T_i}$.

In the modified linear bandit problem at each iteration i, we will be applying LinUCB with respect to a $d_i + i - 1$ dimensional problem with an action set $\mathcal{A}^{\langle d_i\rangle}$ such that $|\mathcal{A}^{\langle d_i\rangle}| \le K + i - 1$. Let $a_\star^{\langle d_i\rangle} = \arg\max_{a \in \mathcal{A}^{\langle d_i\rangle}} \{\langle \theta_\star^{\langle d_i\rangle}, a\rangle\}$ denote the best arm in the i-th iteration. Applying Eq. (3) on $R_{\Delta T_i|\mathcal{F}_{i-1}}$ leads to

$$R_{\Delta T_i|\mathcal{F}_{i-1}} = \Delta T_i \cdot \left(\langle \theta_\star, a_\star\rangle - \langle \theta_\star^{\langle d_i\rangle}, a_\star^{\langle d_i\rangle}\rangle\right) + \mathbb{E}\left[\sum_{t=T_{i-1}+1}^{T_i} \langle \theta_\star^{\langle d_i\rangle}, a_\star^{\langle d_i\rangle} - A_t\rangle \,\middle|\, \mathcal{F}_{i-1}\right], \tag{23}$$

where $A_t \in \mathcal{A}^{\langle d_i\rangle}$ and $\langle \theta_\star^{\langle d_i\rangle}, A_t\rangle$ represents the expected reward of pulling arm A_t.

B.4 Proof of Lemma 5

The proof of Lemma 5 follows the notations and preliminaries introduced in Appendix B.3.

Lemma 5. *At each iteration $i \in [p]$, the learning error suffered from subroutine LinUCB is upper bounded by $O(\log^{5/2}(KT\log T) \cdot T^\beta)$.*

Proof. We focus on the second term in Eq. (23), i.e., the (conditional) learning error during iteration i. Conditioning on $\mathcal{F}_{i-1}$, both $\theta_\star^{\langle d_i\rangle}$ and $a_\star^{\langle d_i\rangle}$ can be treated as fixed quantities. Applying the regret bound in Corollary 1,

we have:

$$\mathbb{E}\left[\sum_{t=T_{i-1}+1}^{T_i}\langle\theta_\star^{\langle d_i\rangle}, a_\star^{\langle d_i\rangle} - A_t\rangle \,\middle|\, \mathcal{F}_{i-1}\right] \le O\left(\log^2\left((K+i-1)\Delta T_i \log(\Delta T_i)\right)\cdot\sqrt{(d_i+i-1)\Delta T_i}\right) \tag{24}$$

$$\le O\left(\log^2\left((K+p)\Delta T_i \log(\Delta T_i)\right)\cdot\sqrt{(d_i+p)\Delta T_i}\right) \tag{25}$$

$$\le O\left(\log^2\left((K+p)T\log T\right)\cdot\sqrt{2^{2p+2}+pT}\right) \tag{26}$$

$$\le O\left(\log^2\left(KT\log T\right)\cdot\sqrt{T^{2\beta}+\log T\cdot T}\right) \tag{27}$$

$$\le O\left(\log^{5/2}\left(KT\log T\right)\cdot T^{\beta}\right), \tag{28}$$

where Eq. (24) comes from the guarantee of LinUCB in Corollary 1; Eq. (25) uses the fact that $i \le p$; Eq. (26) comes from the definition of d_i and ΔT_i; Eq. (27) comes from the fact that $p = \lceil \log_2 T^\beta \rceil$; Eq. (28) comes from trivially bounding $\sqrt{T^{2\beta}+\log T\cdot T} \le O((\log T)^{1/2}\cdot T^\beta)$.[7] The desired result follows from taking another expectation over randomness in $\mathcal{F}_{i-1}$. □

B.5 Proof of Theorem 3

The proof of Theorem 3 follows the notations and preliminaries introduced in Appendix B.3.

Theorem 3. *Run LinUCB++ with time horizon T and any user-specified parameter $\beta \in [1/2, 1)$ leads to the following upper bound on the expected regret:*

$$\sup_{\omega\in\mathcal{H}_T(\alpha)} R_T \le O\left(\log^{7/2}\left(KT\log T\right)\cdot T^{\min\{\max\{\beta, 1+\alpha-\beta\}, 1\}}\right).$$

Proof. When $\alpha \ge \beta$, one could see that Theorem 3 trivially holds since $T^{1+\alpha-\beta} \ge T$. In the following, we only consider the case when $\alpha < \beta$.

Taking expectation on Eq. (23) and combining the result in Lemma 5, we obtain

$$R_{\Delta T_i} \le \Delta T_i \cdot \mathbb{E}\left[\left(\langle\theta_\star, a_\star\rangle - \langle\theta_\star^{\langle d_i\rangle}, a_\star^{\langle d_i\rangle}\rangle\right)\right] + O\left(\log^{5/2}\left(KT\log T\right)\cdot T^\beta\right). \tag{29}$$

We now focus on the first term, i.e., the expected approximation error over the i-th iteration. Notice that, according to the definition of $a_\star^{\langle d_i\rangle}$ and $\theta_\star^{\langle d_i\rangle}$, we have $\langle\theta_\star^{\langle d_i\rangle}, a_\star^{\langle d_i\rangle}\rangle = \langle\theta_\star, a_\star\rangle$ if $d_i \ge d_\star$, i.e., the optimal arm is contained in the action set $\mathcal{A}^{\langle d_i\rangle}$. Let $i_\star \in [p]$ be the largest integer such that $d_{i_\star} \ge d_\star$, we then have that, for any $i \le i_\star$ and in particular for $i = i_\star$,

$$R_{\Delta T_i} \le O\left(T^\beta \log^{5/2}\left(KT\log T\right)\right). \tag{30}$$

In the case when $\Delta T_{i_\star} = \min\{2^{p+i_\star}, T\} = T$ or $i_\star = p$, we know that LinUCB++ will in fact stop at a time step no larger than $T_{i_\star}$ (since the allowed time horizon is T), and incur no regret in iterations $i > i_\star$. In the following, we only consider the case when $\Delta T_{i_\star} = 2^{p+i_\star}$ and $i_\star < p$. To incooperate another possible corner case when $d_{i_\star} = \min\{2^{p+2-i_\star}, d\} = d$, we consider $d_{i_\star+1} = 2^{p+1-i_\star} < d_{i_\star}$. As a result, we have $d_{i_\star}\Delta T_{i_\star} > d_{i_\star+1}\Delta T_{i_\star} = 2^{2p+1}$, which leads to

$$\Delta T_{i_\star} > \frac{2^{2p+1}}{d_{i_\star}} > \frac{2^{2p}}{d_\star} = \frac{2^{2p}}{T^\alpha}, \tag{31}$$

where Eq. (31) comes from the fact that $d_{i_\star} < 2d_\star$ according to the definition of $i_\star$.[8]

[7] One can improve the bound to $\sqrt{T^{2\beta}+\log T\cdot T} \le O((\log T)^{1/2}\cdot T^\beta)$ in many cases, e.g., when $\beta > 1/2$ and T is large enough (with respect to β). However, we mainly focus on the polynomial terms here.

[8] We will have $\Delta T_{i_\star} \ge 2^{2p+1}/T^\alpha > 2^{2p}/T^\alpha$ if $d_{i_\star} = \min\{2^{p+2-i_\star}, d\} = 2^{p+2-i_\star}$.

We now analysis the expected approximation error for iteration $i > i_\star$. Since the sampling information during $i_\star$-th iteration is summarized in the virtual mixture-arm $\widetilde{\nu}_{i_\star}$, and its representation $\widetilde{\nu}_{i_\star}^{\langle d_i \rangle}$ is added to $\mathcal{A}^{\langle d_i \rangle}$. For any $i > i_\star$, we then have

$$\begin{aligned}
\Delta T_i \cdot \mathbb{E}\left[\left(\langle \theta_\star, a_\star \rangle - \langle \theta_\star^{\langle d_i \rangle}, a_\star^{\langle d_i \rangle} \rangle\right)\right] &\leq \Delta T_i \cdot \mathbb{E}\left[\left(\langle \theta_\star, a_\star \rangle - \langle \theta_\star^{\langle d_i \rangle}, \widetilde{\nu}_{i_\star}^{\langle d_i \rangle} \rangle\right)\right] \\
&= \Delta T_i \cdot (\langle \theta_\star, a_\star \rangle - \widetilde{\mu}_{i_\star}) \qquad (32) \\
&= \frac{\Delta T_i}{\Delta T_{i_\star}} \cdot R_{\Delta T_{i_\star}} \qquad (33) \\
&< \frac{\Delta T_i}{\frac{2^{2p}}{T^\alpha}} \cdot O\left(\log^{5/2}(KT \log T) \cdot T^\beta\right) \\
&\leq \frac{O\left(\log^{5/2}(KT \log T) \cdot T^{1+\alpha+\beta}\right)}{2^{2p}} \qquad (34) \\
&\leq O\left(\log^{5/2}(KT \log T) \cdot T^{1+\alpha-\beta}\right), \qquad (35)
\end{aligned}$$

where Eq. (32) comes from the formulation of the modified linear bandit problem; Eq. (33) comes from that fact that $\widetilde{\mu}_j = \mathbb{E}[\widetilde{\mu}_{j|\mathcal{F}_j}] = \langle \theta_\star, a_\star \rangle - R_{\Delta T_j}/\Delta T_j$ derived from Eq. (18); Eq. (34) comes from the fact that $\Delta T_i \leq T$ and some rewriting; Eq. (35) comes from the fact that $p = \lceil \log_2 T^\beta \rceil \geq \log_2 T^\beta$.

Combining Eq. (35) and Eq. (29) for cases when $i > i_\star$ (or the corner case algorithm stops before $T_{i_\star}$ and incurs no regret in iterations $i \geq i_\star$), and together with Eq. (30) for cases when $i \leq i_\star$, we have that $\forall i \in [p]$,

$$R_{\Delta T_i} \leq O\left(\log^{5/2}(KT \log T) \cdot T^{\max\{\beta, 1+\alpha-\beta\}}\right).$$

Since the cumulative regret is non-decreasing in t, we have

$$\begin{aligned}
R_T &\leq \sum_{i=1}^{p} R_{\Delta T_i} \\
&= \sum_{i=1}^{p} O\left(\log^{5/2}(KT \log T) \cdot T^{\max\{\beta, 1+\alpha-\beta\}}\right) \\
&\leq O\left(\log^{7/2}(KT \log T) \cdot T^{\max\{\beta, 1+\alpha-\beta\}}\right),
\end{aligned}$$

where we use the fact that $p = \lceil \log_2(T^\beta) \rceil \leq O(\log T)$. Our results follows after noticing $R_T \leq T$ is a trivial upper bound. □

B.6 Proof of Theorem 4

Theorem 4. *The rate function achieved by LinUCB++ with any input $\beta \in [1/2, 1)$, i.e.,*

$$\theta_\beta : \alpha \mapsto \min\{\max\{\beta, 1 + \alpha - \beta\}, 1\}, \qquad (4)$$

is Pareto optimal.

Proof. From Theorem 3, we know that the rate in Eq. (4) is achieved by Algorithm 1 with input β. We only need to prove that no other algorithms achieve strictly smaller rates in pointwise order.

Suppose, by contradiction, we have θ' achieved by an algorithm such that $\theta'(\alpha) \leq \theta_\beta(\alpha)$ for all $\alpha \in [0, 1]$ and $\theta'(\alpha_0) < \theta(\alpha_0)$ for at least one $\alpha_0 \in [0, 1]$. We then must have $\theta'(0) \leq \theta_\beta(0) = \beta$. We consider the following two exclusive cases.

Case 1 $\theta'(0) = \beta$**.** According to Theorem 2, we must have $\theta' \geq \theta_\beta$, which leads to a contradiction.

Case 2 $\theta'(0) = \beta' < \beta$**.** According Theorem 2, we must have $\theta' \geq \theta_{\beta'}$. However, $\theta_{\beta'}$ is not strictly better than θ_β, e.g., $\theta_{\beta'}(2\beta - 1) = 2\beta - \beta' > \beta = \theta_\beta(2\beta - 1)$, which also leads to a contradiction. □

C ANALYSIS FOR SECTION 4.2

C.1 Discussion on Algorithm 2

We construct the following two (smoothed) base algorithms (Pacchiano et al., 2020b) at each iteration of LinUCB++: (1) a LinUCB algorithm that works with truncated feature representations in $\mathbb{R}^{d_i}$, with possible mis-specifications; and (2) a UCB algorithm that works only with virtual mixture-arms, if there exists any. We use Smooth Corral from Pacchiano et al. (2020b) as the master algorithm and always optimally tune it with respect to the LinUCB base, i.e., set the learning rate as $\eta = 1/\sqrt{d_i \Delta T_i}$. For iterations such that $d_i \geq d_\star$, the LinUCB is the optimal base and we incur $\widetilde{O}(\sqrt{d_i \Delta T_i}) = \widetilde{O}(T^\beta)$ regret; a good enough virtual mixture-arm $\widetilde{\nu}_{i_\star}$ is then constructed as before. For later iterations such that $d_i < d_\star$, Smooth Corral incurs regret $\widetilde{O}(\max\{T^{1+\alpha-\beta}, T^\beta\})$ thanks to guarantees of the UCB base: the $\widetilde{O}(T^{1+\alpha-\beta})$ term is due to the approximation error and the $\widetilde{O}(T^\beta)$ term is due to the learning error. Although the learning error of UCB is enlarged from $\widetilde{O}(T^{1/2})$ to $\widetilde{O}(T^\beta)$, as Smooth Corral is always tuned with respect to the LinUCB base, this won't affect the resulted Pareto optimality.

C.2 Proof of Theorem 5

Theorem 5. *With any input $\beta \in [1/2, 1)$, the rate function achieved by Algorithm 2 (without Assumption 1) is Pareto optimal.*

Proof. At each iteration $i \in [p]$ of LinUCB++, we applying Smooth Corral as the master algorithm with two smoothed base algorithms: (1) a LinUCB algorithm that works with truncated feature representations in $\mathbb{R}^{d_i}$, with possible mis-specifications; and (2) a UCB algorithm that works only with virtual mixture-arms, if there exists any. The learning rate of Smooth Corral is always optimally tuned with respect to the LinUCB base, i.e., $\eta = 1/\sqrt{d_i \Delta T_i}$. Since there are at most $p = O(\log T)$ iterations, we only need to bound the expected regret at each iteration $R_{\Delta T_i}$. As before, we use $i_\star \in [p]$ to denote the largest integer such that $d_{i_\star} \geq d_\star$.

For $i \leq i_\star$, the LinUCB base works on a well-specified linear bandit problem. Theorem 5.3 in Pacchiano et al. (2020b) gives the following guarantees:

$$R_{\Delta T_i} \leq \widetilde{O}\left(\sqrt{\Delta T_i} + \eta^{-1} + \Delta T_i \eta + \Delta T_i d_i \eta\right) = \widetilde{O}\left(\sqrt{d_i \Delta T_i}\right) = \widetilde{O}\left(T^\beta\right).$$

Good enough virtual mixture-arm $\widetilde{\nu}_{i_\star}$ is then constructed with conditional expectation $\widetilde{\mu}_{i_\star | \mathcal{F}_{i_\star}} = \mathbb{E}[\widetilde{\nu}_{i_\star} | \mathcal{F}_{i_\star}] = \langle \theta_\star, a_\star \rangle - \widehat{R}_{\Delta T_{i_\star}} / \Delta T_{i_\star}$.

We now analyze the regret incurred for iteration $i > i_\star$. Conditioning on past information $\mathcal{F}_{i-1}$ and let $r(\pi_t)$ denote the (conditional) expected reward of applying policy π_t, we have

$$\begin{aligned} R_{\Delta T_i | \mathcal{F}_{i-1}} &= \Delta T_i \cdot \left(\langle \theta_\star, a_\star \rangle - \widetilde{\mu}_{i_\star | \mathcal{F}_{i_\star}}\right) + \mathbb{E}\left[\sum_{t \text{ in iteration } i} \widetilde{\mu}_{i_\star | \mathcal{F}_{i_\star}} - r(\pi_t) \,\middle|\, \mathcal{F}_{i-1}\right] \\ &\leq \Delta T_i \cdot \left(\langle \theta_\star, a_\star \rangle - \widetilde{\mu}_{i_\star | \mathcal{F}_{i_\star}}\right) + \widetilde{O}\left(\sqrt{\Delta T_i} + \eta^{-1} + \Delta T_i \eta + \Delta T_i \eta\right), \end{aligned}$$

where the second term comes from the guarantee of Smooth Corral with respect to the UCB base. Taking expectation over randomness in $\mathcal{F}_{i-1}$ leads to

$$R_{\Delta T_i} \leq \widetilde{O}\left(T^{1+\alpha-\beta}\right) + \widetilde{O}\left(T^\beta\right),$$

where the first term follows from a similar analysis as in Eq. (35), and the second term follows by setting $\eta = 1/\sqrt{d_i \Delta T_i}$. A similar analysis as in Theorem 4 thus show Algorithm 2 is Pareto optimal, even without Assumption 1. □

C.3 Discussion on Smooth Corral

Pacchiano et al. (2020b) tackles the model selection problem in linear bandit by applying Smooth Corral with $O(\log d)$ base LinUCB learners working with different dimensions $d_i \in \{2^0, 2^1, \ldots, 2^{\lfloor \log d \rfloor}\}$. Let $d_{i_\star}$ denote the

smallest dimension that satisfies $d_{i_\star} \geq d_\star$. With respect to the base LinUCB working on the first $d_{i_\star}$ dimensions, Smooth Corral enjoys regret guarantee

$$R_T \leq \widetilde{O}\left(\sqrt{T} + \eta^{-1} + T\eta + Td_\star\eta\right). \tag{36}$$

Smooth Corral then achieves the rate function in Eq. (4) by setting the learning rate $\eta = T^{-\beta}$ (and also noticing that $d_\star \leq T^\alpha$).

D ADDITIONAL EXPERIMENT RESULTS

We conduct additional experiments with setups similar to the ones shown in Fig. 2b, but with different reward parameters $\theta_\star$. We set $\theta_\star$ as (the normalized version of) $[\frac{1}{\sqrt{1}}, \frac{1}{\sqrt{2}}, \ldots, \frac{1}{\sqrt{d_\star}}, 0, \ldots, 0]^\top \in \mathbb{R}^d$ in Fig. 4a; and $\theta_\star$ as (the normalized version of) $[\frac{1}{\sqrt{d_\star}}, \frac{1}{\sqrt{d_\star - 1}}, \ldots, \frac{1}{\sqrt{1}}, 0, \ldots, 0]^\top \in \mathbb{R}^d$ in Fig. 4b. With $\theta_\star$ selected in Fig. 4a, Dynamic Balancing shows comparable performance to LinUCB++ in terms of averaged regret (but with larger variance). LinUCB++ outperforms Dynamic Balancing when $\theta_\star$ is "flipped" (i.e., the one used in Fig. 4b) but with the same intrinsic dimension $d_\star$.

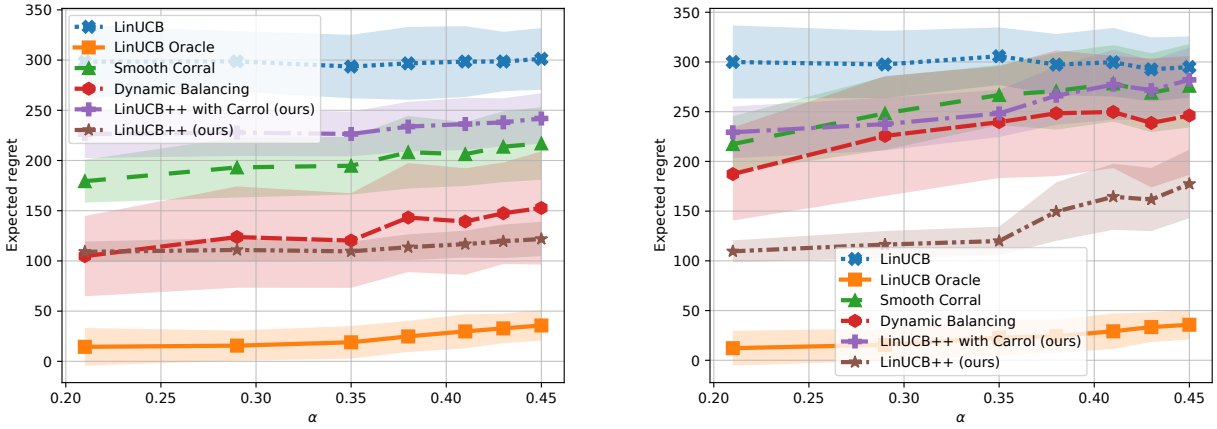


Figure 4: Similar Experiment Setups to Those Shown in Fig. 2b, but with Different reward parameters $\theta_\star$.