ACHIEVING SUBLINEAR REGRET IN FIXED-SIZE SKETCHING LINEAR BANDITS: A CLUSTERING APPROACH

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the problem of computational efficiency for linear bandits in high-dimensional settings with a finite arm set. In linear bandits, the increase in the dimension d of the feature vectors leads to growing computational costs of $O(d^2)$ at each round of update. Traditional sketching-based methods such as SOFUL reduce computation via fixed-size matrix sketching, yet run the risk of incurring vacuous linear regret when the spectral tail of the data is heavy and the sketch size is inadequately selected. To guarantee regret convergence and effectively reduce computational costs, we introduce a clustering mechanism and propose the Cluster Sketch Linear Bandit (CS-LB) algorithm. Our method preserves the full covariance information in each cluster to guarantee robust sublinear regret without spectral-tail vulnerabilities, performs cluster switching by assigning a sentinel for each cluster, and reduces per-round update computation to $O(l^2 d)$ via a tunable sketch size $l < d$. Experiments on synthetic datasets demonstrate that our method consistently maintains a favorable trade-off between efficiency and regret.

1 INTRODUCTION

The stochastic linear contextual bandit is a fundamental framework for sequential decision-making under uncertainty, where an agent repeatedly selects an action from a finite set based on its associated context vector, and observes a noisy reward that is linear in the unknown parameter. This model has found widespread applications in personalized recommendation (Li et al., 2010), online advertising (Tang et al., 2013), and mobile health (Tewari & Murphy, 2017), among others. Classical algorithms such as LinUCB (Chu et al., 2011; Abbasi-Yadkori et al., 2011) and linear Thompson Sampling (Agrawal & Goyal, 2013) achieve near-optimal regret guarantees, but incur a prohibitive $O(d^2)$ per-round time complexity due to the need to maintain and invert a $O(d \times d)$ covariance matrix, where d is the feature dimension.

As data scales have grown rapidly in recent years, both the feature dimension d and the number of arms N can be extremely large, rendering exact methods computationally infeasible. To reduce this update computation, recent works have turned to matrix sketching techniques, which maintain a low-dimensional approximation of the covariance matrix. SOFUL (Kuzborskij et al., 2019) applies Frequent Directions (Ghashami et al., 2016) to compress the covariance matrix from dimension d to $l < d$, reducing per-round complexity to $O(ld)$. The CBSCFD (Chen et al., 2021) algorithm improves upon SOFUL by leveraging the accumulated information of its truncated singular values, thereby achieving better regret performance. While these methods offer substantial computational speedups, they suffer from a critical drawback: the global sketching of the entire action-history covariance matrix inevitably introduces approximation errors that accumulate over time. This spectral distortion can inflate the confidence ellipsoid, degrade arm-selection accuracy, and in some cases lead to linear regret or even non-convergence, particularly when the covariance spectrum has a heavy tail. In other words, existing sketching methods sacrifice linear regression accuracy in favor of computational efficiency. However, this trade-off struggles to guarantee the convergence of regret, which may render them ineffective in systems that require stability guarantees. Wen et al. (2024) improved upon SOFUL and proposed DBSLinUCB, which ensures sublinear regret through a multi-scale, adaptively adjusted sketch size. Although DBSLinUCB resolves the non-convergence issue of SOFUL, its theoretical guarantees rely on adaptive sketch-size growth and cannot be extended to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the fixed-sketch-size setting. Moreover, its computational efficiency is predicated on low-rank data; on full-rank data, it reverts to $O(d^2)$ complexity, losing the speed advantage of sketching.

To achieve regret convergence in computation-constrained linear bandits, we propose the Cluster Sketch Linear Bandit (CS-LB). To our knowledge, it is the first sketching-based bandit strategy that reduces computational cost while guaranteeing regret convergence under a fixed sketch size. Our main contributions are as follows:

- **Cluster Sketching Framework.** We propose a clustering-assisted sketching framework for finite-action linear bandits. Unlike prior global sketching approaches, our method partitions the action set into multiple clusters and maintains a dedicated sketch for each cluster. This design preserves the information integrity of each cluster's sketch matrix, which serves as the foundation for our regret convergence guarantee.
- **Sentinel-Guided Cluster Selection.** We introduce a sentinel mechanism atop the cluster-sketch structure, where the sentinel of each cluster is defined as the maximum upper confidence bound among its arms. This mechanism enforces global optimism in the spirit of OFUL, aligning the algorithm's behavior with OFUL and enabling a clean theoretical analysis of regret.
- **Regret Convergence and Computational Efficiency.** We prove that CS-LB achieves a regret bound of $\widetilde{O}(l\sqrt{N_{\mathcal{A}}^l T})$ and a per-round computational cost of $O(l^2 d)$, excluding the additional overhead incurred during the warm-up phase, offering a clear efficiency gain when $l^2 < d$. Notably, unlike previous sketching-based methods, our regret bound is free of the spectral error term Δ_T—the primary source of linear regret in existing approaches—thus ensuring rigorous regret convergence. Moreover, our theoretical framework remains valid under both a fixed sketch size and full-rank data settings.
- **Empirical Validation.** Experiments on synthetic data show that CS-LB consistently outperforms SOFUL and CBSCFD in cumulative regret, with the largest improvements observed when the action set exhibits a natural clustering structure. In addition, CS-LB achieves lower running times than OFUL, confirming its computational advantage.

2 Related Work

Linear Bandits Contextual bandits generalize the finite-arm setting by allowing the exploitation of side information through linear models, thereby giving rise to the linear bandit framework. The study of regret minimization in linear bandits was first studied by Abe & Long (1999), and has attracted extensive interest in the development of various algorithms (Rusmevichientong & Tsitsiklis, 2010; Chu et al., 2011). Auer (2003) provided an early algorithm (LinRel) and theoretical analysis for this problem by introducing confidence-bound-based techniques, improving the regret bound from the previous $\widetilde{O}(T^{3/4})$ to $\widetilde{O}(\sqrt{T})$. Subsequently, Dani et al. (2008) extended the theoretical framework to more general compact action sets. Li et al. (2010) proposed the LinUCB algorithm and successfully applied it to personalized recommendation. Chu et al. (2011) conducted further theoretical investigations of linear bandits by maintaining a regularized least-squares estimate of the unknown parameter and constructing a confidence ellipsoid centered on this estimate. By selecting the action that maximizes the upper confidence bound, their work demonstrated that LinUCB achieves a near-optimal regret of $\widetilde{O}(d\sqrt{T})$ under relatively mild assumptions. Abbasi-Yadkori et al. (2011) further refined the analysis by developing a tighter confidence set based on self-normalized martingale inequalities, and introduced the classic OFUL strategy. Agrawal & Goyal (2013) adapted Thompson Sampling (Thompson, 1933) to linear bandits, and this method was later further refined by Abeille & Lazaric (2016). Linear Thompson sampling randomly draws a parameter from the posterior distribution and selects the action that maximizes the expected reward under this sample. Its performance is comparable to that of OFUL up to logarithmic factors.

Sketching Bandits Sketching is a classic dimensionality reduction technique (Song et al., 2017; Andoni et al., 2018; Deb et al., 2025) that has been introduced into linear bandits to reduce the computational cost of matrix inversion (Yu et al., 2017; Kuzborskij et al., 2019). Ghashami et al. (2016) proposed Frequent Directions (FD), a sketching method with strong theoretical guarantees in streaming settings, making it particularly suitable for online environments. Kuzborskij et al.

(2019) first combined FD with the classic OFUL algorithm in linear contextual bandits, yielding the SOFUL algorithm and reducing per-round update costs. Building on SOFUL, Chen et al. (2021) developed a variant of FD for linear bandits that achieves a tighter regret bound. To overcome the potential non-convergence issue of traditional sketching methods, Wen et al. (2024) proposed DB-SLinUCB, a multi-scale sketching framework that adaptively adjusts the sketch size during learning, thus avoiding the linear regret trap of fixed-size sketches. Unlike previous single-scale approaches, DBSLinUCB dynamically distributes the sketch budget across blocks to guarantee sublinear regret without prior knowledge of the data spectrum. While DBSLinUCB features an adaptive sketch size, our work focuses on the fixed-sketch-size setting.

3 Problem Setting

3.1 Notation and Definition

In our paper, every d-dimensional vector is represented as a column vector. For matrix $\boldsymbol{A}$, $[\boldsymbol{A}]_k$ denotes the matrix formed by taking the first k rows of matrix $\boldsymbol{A}$. For matrix $\boldsymbol{A} \in \mathbb{R}^{d_1 \times d_2}$, the computed SVD of $\boldsymbol{A}$ is defined as $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U}$ is $d_1 \times d_2$ matrix, $\boldsymbol{\Sigma}$ is $d_2 \times d_2$ diagonal matrix, $\boldsymbol{V}$ is $d_2 \times d_2$ matrix. We use $SVD(\boldsymbol{A})$ to denote the singular value decomposition of matrix $\boldsymbol{A}$ and obtain the corresponding $\boldsymbol{U}$, $\boldsymbol{V}^\top$ and $\boldsymbol{\Sigma}$. Let $\sigma_i(\boldsymbol{A})$ be the i-th largest singular value of $\boldsymbol{A}$. For a positive integer N, let $[N] = \{1, ..., N\}$. For a set $\mathcal{A}$, $|\mathcal{A}|$ denotes the number of elements contained in the set. For theoretical bounds, the notation $\widetilde{O}(\cdot)$ omits logarithmic factors.

3.2 Stochastic Linear Bandits

In the MAB model, the learner needs to select an appropriate arm to pull at each round; in stochastic linear bandits (SLB), the expected reward of each arm is a linear function with an unknown parameter $\theta^* \in \mathbb{R}^d$.

For classical linear bandits, the learner is given an arm set $\mathcal{A} = [N]$, which corresponds to known arm vectors $\{\boldsymbol{x}^1, ..., \boldsymbol{x}^N\} \subset \mathbb{R}^{d\times 1}$. In round t, the learner chooses an action $\boldsymbol{x}_t \in \mathcal{A}$ and receives reward $r_t = \langle \boldsymbol{x}_t, \boldsymbol{\theta}^* \rangle + \eta_t$, where η_t is conditionally independent of $\boldsymbol{x}_t$ given $\boldsymbol{x}_{1:t-1}, r_{1:t-1}$. For our model, we currently consider LSB in the finite-set setting.

According to Abbasi-Yadkori et al. (2011), we make the following assumption:

Assumption 1. *The noise η_t is R-sub-Gaussian conditioning on $\boldsymbol{x}_{1:t-1}, r_{1:t-1}$.*

$$\mathbb{E}[e^{\lambda \eta_t} | \boldsymbol{x}_{1:t-1}, r_{1:t-1}] \leq exp(R^2\lambda^2/2).$$

Assumption 2. $\|\boldsymbol{\theta}^*\|_2 \leq S$, $\|\boldsymbol{x}\|_2 \leq L$ *for all action $\boldsymbol{x}_t \in \mathcal{A}$, for all $t \in [T]$.*

The goal of the linear bandits is to collaboratively minimize the cumulative regret defined as

$$Regret(T) := \sum_{t=1}^{T} \left(\max_{\boldsymbol{x}\in\mathcal{A}} \langle \boldsymbol{x}, \boldsymbol{\theta}^* \rangle - \langle \boldsymbol{x}_t, \boldsymbol{\theta}^* \rangle \right)$$

where $\boldsymbol{x}^* = argmax_{\boldsymbol{x}\in\mathcal{A}} \langle \boldsymbol{x}, \boldsymbol{\theta}^* \rangle$ is the optimal arm.

We use $\boldsymbol{X}_t = [\boldsymbol{x}_1, ..., \boldsymbol{x}_t]^\top$ to define the matrix of all actions selected up to round t.

3.3 Computation efficiency

Although the update overhead of linear bandits can be reduced to $O(d^2)$, such computational cost remains substantial in high-dimensional scenarios, especially in systems emphasizing real-time performance where the per-round computation budget may be strictly constrained. Unlike offline training pipelines that can leverage distributed computing and batching, online bandit algorithms demand immediate responses, and any computation beyond the allocated budget directly translates into service degradation or revenue loss. This practical constraint motivates the development of algorithms that offer a flexible trade-off between computational cost and statistical accuracy, allowing practitioners to adapt to varying system loads and latency requirements.

3.4 SKETCH

In the study of computational overhead in linear bandits, Kuzborskij et al. (2019) first proposed Sketched OFUL (SOFUL), which effectively reduces the computational cost of per-round updates. In traditional linear bandits, the update of the linear regression strategy relies on computing the inverse of the matrix $\boldsymbol{V}_t = \lambda \boldsymbol{I} + \boldsymbol{X}_t^\top \boldsymbol{X}_t$. This updating always incurs $O(d^2)$ computation overhead. Let $l < d$ be the sketch size. The goal of sketching is to keep an approximate matrix $\boldsymbol{S}_T \in \mathbb{R}^{l \times d}$ in place of $\boldsymbol{X}_T$ for policy updates, where $\boldsymbol{S}_T$ satisfies

$$\boldsymbol{S}_T^\top \boldsymbol{S}_T \approx \boldsymbol{X}_T^\top \boldsymbol{X}_T.$$

The classical sketch algorithm Frequent Directions (FD) (Ghashami et al., 2016) truncates the smallest singular value after performing an SVD of the covariance matrix at every update. In each round t, FD updates as follows:

$$[\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^\top] = SVD\left([\boldsymbol{S}_{t-1}^\top; \boldsymbol{x}_t]^\top\right),$$
$$\delta_t = \sigma_l^2(\boldsymbol{\Sigma}), \boldsymbol{S}_t = \sqrt{\boldsymbol{\Sigma}^2 - \delta_t \boldsymbol{I}}\boldsymbol{V}^\top,$$

where $\sigma_i(\boldsymbol{\Sigma})$ is the i-th largest singular value of $\boldsymbol{\Sigma}$.

FD method can reduce the computational cost from $O(d^2)$ to $O(ld)$ at the expense of doubling the space used by the algorithm. The FD method can effectively reduce the computational cost; however, it incurs information loss due to singular value truncation, thereby leading to non-convergence. The sketch matrix serves as an approximate representation of the original covariance matrix; some information is lost during the sketching process (e.g., the truncation of singular values), resulting in a discrepancy between the sketch and the original covariance matrix.

To address the information loss and instability caused by FD, a variant called Spectral Compensation Frequent Directions (SCFD) (Chen et al., 2021) introduces a compensation mechanism that preserves the positive definite monotonicity of the covariance sequence. Unlike FD, which directly discards the truncated spectral mass δ_t, SCFD accumulates this lost information into a scalar $\alpha_t = \alpha_{t-1} + \delta_t$ and explicitly adds it as a diagonal correction $\alpha_t \boldsymbol{I}_d$ to the sketched approximation. That is, SCFD maintains

$$\hat{\boldsymbol{V}}_t = \boldsymbol{S}_t^\top \boldsymbol{S}_t + \alpha_t \boldsymbol{I}_d \approx \boldsymbol{X}_T^\top \boldsymbol{X}_T + \lambda \boldsymbol{I},$$

which ensures $\hat{\boldsymbol{V}}_t \succeq \hat{\boldsymbol{V}}_{t-1}$ for all t, and achieves better-conditioned approximations. Although SCFD achieves improved regret bounds and stronger robustness, it still retains the same error bound as FD. The non-convergence issue caused by heavy spectral tails in sketching methods persists.

4 ALGORITHM

In this section, we analyze the limitations of existing approaches for improving computational efficiency in linear bandits, and present our proposed algorithm, Cluster Sketch Linear Bandits (CS-LB). The CS-LB algorithm not only enhances computational efficiency with a fixed sketch size, but also ensures the convergence of the corresponding regret.

4.1 LIMITATIONS OF EXISTING APPROACHES

OFUL (Abbasi-Yadkori et al., 2011). As a classic algorithm for linear bandits, OFUL enjoys rigorous theoretical guarantees on regret convergence. However, its per-round update cost is prohibitively large in high-dimensional settings, which makes it difficult to deploy OFUL in computation-constrained environments.

SOFUL (Kuzborskij et al., 2019) / CBSCFD (Chen et al., 2021). SOFUL reduces OFUL's computation to $O(dl)$ via fixed-size matrix sketching. Its regret bound, however, depends critically on the spectral error Δ_T:

$$\text{Regret}_T^{SOFUL} = \tilde{O}\left((1+\Delta_T)^{3/2}\left(l + d\log(1+\Delta_T)\right)\sqrt{T}\right).$$

When the data exhibit heavy spectral tails, Δ_T grows with T, and the regret collapses to linear. Thus, SOFUL trades statistical reliability for efficiency—a compromise that fails precisely when robust

performance is most needed. Although CBSCFD reduces the spectral error term from $O(\Delta_T^{3/2})$ to $O(\sqrt{\Delta_T})$, it still carries the risk of linear regret (Wen et al., 2024).

DSBLinUCB (Wen et al., 2024). DBSLinUCB employs a multi-scale, adaptively adjusted sketch size to ensure that the global spectral error is bounded by a fixed parameter ε, thereby guaranteeing sublinear regret. This effectively addresses the long-standing linear regret issue inherent in SOFUL and CBSCFD. However, DBSLinUCB still has several limitations. First, it maintains gradually growing blocks to adaptively increase the sketch size so as to ensure regret convergence, which makes it inapplicable in computation-constrained scenarios where a fixed sketch size is required. Second, DBSLinUCB is built on the assumption that the data is low-rank. When the data is full-rank and heavy-tailed, DBSLinUCB degrades to exact rank-1 updates, in which case its time complexity increases to $O(d^2)$—the same as OFUL—thereby losing the speed advantage of sketching.

4.2 Cluster Sketch Linear Bandits

We address the regret non-convergence problem in traditional sketching methods and propose the Cluster Sketch Linear Bandits (CS-LB) algorithm to improve the computational efficiency of updates for linear bandits with a finite arm set, as shown in Algorithm 3. Moreover, our method still guarantees regret convergence even in the presence of full-rank data and under a fixed sketch size. CS-LB is primarily composed of the following components:

FD Sketching. As mentioned above, FD is a deterministic sketching algorithm that maintains the l-th smallest singular value of the sketch matrix and truncates the remaining singular values. This approach strictly bounds the computational complexity of SVD decomposition at $O(l^2 d)$, and by leveraging the acceleration techniques in FD, the average per-round update cost can be further reduced to $O(ld)$, where the sketch size $l < d$. While this method achieves a significant reduction in computational overhead, when the matrix X_T is not low-rank, the FD method discards much of the information necessary for accurately computing linear regression, ultimately leading to non-convergent regret in SOFUL (Kuzborskij et al., 2019) and CBSCFD (Chen et al., 2021). Although the FD method may suffer from non-convergence, its simplicity and efficiency lead us to adopt it for reducing the update cost of linear bandits, with further optimizations built on top. Here we make some modifications to the update procedure in Ghashami et al. (2016), and present the resulting FD sketching update procedure in Algorithm 1.

Algorithm 1 FD sketching: $FD(\boldsymbol{S}, \boldsymbol{x})$

Input: $\boldsymbol{S}$, $\boldsymbol{x}$
Parameters: sketch size l
Output:

1: compute SVD: $[\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^T] = SVD(\text{vstack}[\boldsymbol{S}, \boldsymbol{x}^\top])$
where $\boldsymbol{U} \in \mathbb{R}^{(l+1)\times(l+1)}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{(l+1)\times d}$, $\boldsymbol{V}^T \in \mathbb{R}^{d\times d}$
2: $k = rank(\boldsymbol{\Sigma})$
3: $\hat{\boldsymbol{\Sigma}} = [\boldsymbol{\Sigma}]_{l\times d}$
4: $\boldsymbol{S'} = \sqrt{\hat{\boldsymbol{\Sigma}}}\boldsymbol{V}^T \in \mathbb{R}^{l\times d}$
5: $\boldsymbol{H} = ([\hat{\boldsymbol{\Sigma}}]_{l\times l} + \lambda \boldsymbol{I})^{-1} \in \mathbb{R}^{l\times l}$
6: **return** $\boldsymbol{S'}, \boldsymbol{H}, k$

The vstack function performs row-wise concatenation of the sketch matrix $\boldsymbol{S}$ and the vector $\boldsymbol{x}$. The FD sketch algorithm considered herein receives an $l \times d$ sketch matrix and a $d \times 1$ update vector $\boldsymbol{x}$ as inputs, and outputs the rank k resulting from the concatenation, the updated $l \times d$ sketch matrix S', as well as the matrix H' needed for inverse computation. Let $\bar{\boldsymbol{V}} = \lambda \boldsymbol{I} + \boldsymbol{S'}^\top \boldsymbol{S'}$. According to Kuzborskij et al. (2019), $\bar{\boldsymbol{V}}_t^{-1} = \frac{1}{\lambda}(\boldsymbol{I} - \boldsymbol{S'}^\top \boldsymbol{H} \boldsymbol{S'})$ can be efficiently computed.

Cluster sketching Warm-up. We propose a strategy that integrates clustering with matrix sketching, where each cluster maintains an independent sketch matrix to preserve the local covariance structure of arms within the cluster, rather than relying on a global unified sketch. We partition the original arm set into clusters based on the rank information obtained from FD sketch updates, ensuring that the rank of the matrix composed of the arm vectors within each cluster does not exceed l. During this clustering process, an arm is assigned to a cluster if its inclusion does not cause the

cluster's rank to exceed l; otherwise, we search for another suitable cluster or form a new cluster consisting solely of that arm.

Algorithm 2 presents our cluster sketch partitioning scheme. We use $\mathcal{C}^{all}$ to denote the set of all clusters. Each cluster $\mathcal{C}$ is a disjoint subset of the arm set, and each cluster maintains its own sketch matrix $\mathbf{S}_{\mathcal{C}}$. Algorithm 2 takes an arm set $\mathcal{A}$ as input and returns a partitioned cluster set $\mathcal{C}^{all}$ after $t = N$ rounds.

In the warm-up phase, FD sketching update process itself corresponds to computing the rank within the cluster (i.e. the matrix $\boldsymbol{X}_{\mathcal{C},t}$ obtained by concatenating the arms contained in the cluster as row vectors). By leveraging FD sketching, the rank of $\boldsymbol{X}_{\mathcal{C},t}$ can be obtained with a computational cost of $O(l^2 d)$. Through cluster sketch-based warm-up of the arm set, the rank of the matrix $\boldsymbol{X}_{\mathcal{C},N}$ is bounded by the sketch size l. Drawing upon Property 3 in Kuzborskij et al. (2019), we ensure that $\boldsymbol{S}_{\mathcal{C},N}^{\top}\boldsymbol{S}_{\mathcal{C},N} = \boldsymbol{X}_{\mathcal{C},N}^{\top}\boldsymbol{X}_{\mathcal{C},N}$ holds within each cluster. Consequently, the FD sketching update incurs no information loss inside any cluster, thereby guaranteeing the convergence of regret.

Algorithm 2 Cluster Sketching Warm-up: $CSW(\mathcal{A}, l)$

Input: $\mathcal{A}$, sketch size l
Initialization: $\mathcal{C}^{all} = \phi$, Find=False, $t = 0$
Output:

for $\boldsymbol{x} \in \mathcal{A}$ **do**
  for $\mathcal{C} \in \mathcal{C}^{all}$ **do**
    $\boldsymbol{S}_{\mathcal{C},t+1}, \boldsymbol{H}_{\mathcal{C},t+1}, k_{\mathcal{C},t+1} = FD(\boldsymbol{S}_{\mathcal{C},t}, \boldsymbol{x})$
    if $k_{\mathcal{C},t+1} \leq l$ **then**
      Find=True
      $\mathcal{C} = \mathcal{C} \cup \{\boldsymbol{x}_t\}$
      break
    else
      Find=False
      $\boldsymbol{S}_{\mathcal{C},t+1} = \boldsymbol{S}_{\mathcal{C},t}$
    end if
  end for
  if not Find **then**
    $\mathcal{C}_{new} = \{\boldsymbol{x}\}$
    $\boldsymbol{S}_{\mathcal{C}_{new},t+1} = \text{vstack}[\boldsymbol{x}^{\top}, \mathbf{0}_{(l-1)\times d}] \in \mathbb{R}^{l\times d}$.
    $\mathcal{C}^{all}$.append($\mathcal{C}_{new}$), len($\mathcal{C}^{all}$)+=1
  end if
  t+=1
end for
return $\mathcal{C}^{all}$

Confidence Ellipsoid. Under our cluster sketch partitioning strategy, FD sketching updates within each cluster incur no information loss. Consequently, the linear regression estimates $\hat{\boldsymbol{\theta}}_{\mathcal{C},t}$ within each cluster can accurately identify the optimal arm corresponding to that cluster. Following Abbasi-Yadkori et al. (2011) and Kuzborskij et al. (2019), we construct the confidence interval as:

$$\beta_t(\delta) = R\sqrt{l\ln(1+\frac{tL^2}{l\lambda}) + 2\ln\frac{N}{l\delta}} + S\sqrt{\lambda}. \tag{1}$$

Lemma 1. *With probability at least $1-\delta$, for any $0 < t \leq T$ and $\mathcal{C} \in \mathcal{C}^{all}$, we have*

$$\left\|\hat{\boldsymbol{\theta}}_{\mathcal{C},t} - \boldsymbol{\theta}^*\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C},t}} \leq \beta_t(\delta),$$

where $\bar{\boldsymbol{V}}_{\mathcal{C},t} = \lambda\boldsymbol{I} + \boldsymbol{S}_{\mathcal{C},t}^{\top}\boldsymbol{S}_{\mathcal{C},t}$ is the sketched regularized correlation matrix for cluster $\mathcal{C}$ in round t.

Sentinel Strategy. Combining with the clustering partition, we propose CS-LB (Algorithm 3) to perform arm selection and update at each round. We assign a sentinel $U(\mathcal{C})$ to each cluster, initialized to positive infinity (or the upper bound on rewards). During subsequent updates, the sentinel $U(\mathcal{C})$ is set to the maximum upper confidence bound of the reward among all arms within that cluster.

At the beginning of each round, the algorithm selects the cluster with the largest sentinel value as the current active cluster $\mathcal{W}_t$, and then performs LinUCB selection within $\mathcal{W}_t$. Specifically, it chooses the current arm $\boldsymbol{x}_t$ with the maximum upper confidence bound of the linear estimate to pull. Subsequently, FD sketching is updated to incorporate $\boldsymbol{x}_t$ into the sketch matrix $\boldsymbol{S}_{\mathcal{W}_t,t}$. Note that after warm-up, the rank within each cluster remains below l, and the selected arm belongs to the current cluster. Therefore, the FD sketching update incurs no information loss, and within the current cluster $\mathcal{W}_t$, it effectively behaves as a rank-l LinUCB update. Moreover, owing to the adoption of the FD sketching update, the inverse matrix $\bar{\boldsymbol{V}}^{-1}_{\mathcal{W}_t,t}$ can be obtained via matrix multiplication without the need for separate computation. According to Kuzborskij et al. (2019), the computational cost per round stems from the SVD decomposition, yielding a per-round complexity of $O(l^2 d)$. Compared with the $O(d^2)$ cost incurred by computing the inverse matrix in the traditional OFUL algorithm, CS-LB achieves a significant reduction in computational overhead when $l < \sqrt{d}$.

The sentinel strategy allows our method to preserve the OFUL policy at the global level, while enabling the adoption of the LinUCB policy within individual clusters after warm-up. The integration of these two strategies constitutes our CS-LB algorithm. Overall, the CS-LB algorithm maintains the optimism-in-the-face-of-uncertainty principle globally and executes a rank-l LinUCB method inside each cluster, thereby substantially lowering the per-round computational cost without compromising regret convergence.

Algorithm 3 Cluster Sketch Linear Bandits (CS-LB)

Input: $\mathcal{A}$, sketch size l, δ
Initialization: $\mathcal{C}^{all} \longleftarrow CSW(\mathcal{A}, l)$
$U(\mathcal{C}) = +\infty, \boldsymbol{S}_{\mathcal{C},1} = \boldsymbol{0}_{l\times d}, \hat{\boldsymbol{\theta}}_{\mathcal{C},1} = \boldsymbol{0}_{1\times d}, \bar{\boldsymbol{V}}^{-1}_{\mathcal{C},1} = \frac{1}{\lambda}\boldsymbol{I}_{d\times d}, \forall \mathcal{C} \in \mathcal{C}^{all}$
Output:

for round $t = 1, ..., T$ **do**
  $\mathcal{W}_t = argmax_{\mathcal{C}\in\mathcal{C}^{all}} U(\mathcal{C})$
  $\boldsymbol{x}_t = argmax_{\boldsymbol{x}\in\mathcal{W}_t} \left\{ \boldsymbol{x}^\top \hat{\boldsymbol{\theta}}_{\mathcal{W}_t,t} + \beta_t(\delta) \left\|\boldsymbol{x}\right\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{W}_t,t}} \right\}$
  Play $\boldsymbol{x}_t$ and receive y_t
  $\boldsymbol{S}_{\mathcal{W}_t,t+1}, \boldsymbol{H}_{\mathcal{W}_t,t+1}, k_{\mathcal{W}_t,t+1} = FD(\boldsymbol{S}_{\mathcal{W}_t,t}, \boldsymbol{x}_t)$
  $\bar{\boldsymbol{V}}^{-1}_{\mathcal{W}_t,t+1} = \frac{1}{\lambda}\left(\boldsymbol{I} - \boldsymbol{S}^T_{\mathcal{W}_t,t+1}\boldsymbol{H}_{\mathcal{W}_t,t+1}\boldsymbol{S}_{\mathcal{W}_t,t+1}\right)$
  $\boldsymbol{\beta}_{\mathcal{W}_t,t+1} = \boldsymbol{\beta}_{\mathcal{W}_t,t} + y_t\boldsymbol{x}_t$
  $\hat{\boldsymbol{\theta}}_{\mathcal{W}_t,t+1} = \bar{\boldsymbol{V}}^{-1}_{\mathcal{W}_t,t+1}\boldsymbol{\beta}_{\mathcal{W}_t,t+1}$
  $U(\mathcal{W}_t) = \max_{\boldsymbol{x}\in\mathcal{W}_t} \left\{ \boldsymbol{x}^\top \hat{\boldsymbol{\theta}}_{\mathcal{W}_t,t+1} + \beta_{t+1}(\delta) \left\|\boldsymbol{x}\right\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{W}_t,t+1}} \right\}$
  for $\mathcal{C} \in \mathcal{C}^{all}$ and $\mathcal{C} \neq \mathcal{W}_t$ **do**
    $\boldsymbol{S}_{\mathcal{C},t+1} = \boldsymbol{S}_{\mathcal{C},t}, \boldsymbol{H}_{\mathcal{C},t+1} = \boldsymbol{H}_{\mathcal{C},t}, k_{\mathcal{C},t+1} = k_{\mathcal{C},t}.$
    $\bar{\boldsymbol{V}}^{-1}_{\mathcal{C},t+1} = \bar{\boldsymbol{V}}^{-1}_{\mathcal{C},t}, \boldsymbol{\beta}_{\mathcal{C},t+1} = \boldsymbol{\beta}_{\mathcal{C},t}, \hat{\boldsymbol{\theta}}_{\mathcal{C},t+1} = \hat{\boldsymbol{\theta}}_{\mathcal{C},t}$
  end for
end for

5 Theoretical Results

5.1 Regret Analysis

Theorem 1. *Given the sketch size l, if $\beta_t(\delta)$ is chosen as specified in Equation 1 and set $\lambda \geq \max\{1, L^2\}$, then with probability at least $1-\delta$, the expected cumulative regret of Algoritm 3 is upper bounded by*

$$Regret(T) \leq 2\beta_T(\delta)\sqrt{2lN^l_{\mathcal{A}}T\ln\left(1+\frac{TL^2}{\lambda l}\right)} \tag{2}$$

$$= \widetilde{O}\left(l\sqrt{N^l_{\mathcal{A}}T}\right) \tag{3}$$

where $N^l_{\mathcal{A}} = |\mathcal{C}^{all}|$ is the number of clusters produced by Cluster sketch warm-up on arm set $\mathcal{A}$ with sketch size l.

Compared with the results of SOFUL/CBSCFD in the finite arm set setting, our regret bound eliminates the influence of the spectral error term Δ_T, rigorously guaranteeing the convergence of regret for any sketch size $l > 1$. This leads to an overall improvement in the algorithm's performance. The detailed proof is provided in the Appendix.

Since each cluster has a rank of l, when the arms are processed sequentially according to Algorithm 2, each cluster can contain at least l arms. Therefore, we have $N_{\mathcal{A}}^l < \lceil N/l \rceil$. In this case, the regret bound of CS-LB is obtained as $\widetilde{O}(\sqrt{lNT})$, which implies that in the worst case, the regret of CS-LB depends on the data scale N. Here, $N_{\mathcal{A}}^l$ is a quantity determined by the composition of the dataset $\mathcal{A}$ and the sketch size l. If the dataset exhibits strong linear correlation, the number of arms contained in each cluster increases, and consequently the number of cluster $N_{\mathcal{A}}^l$ becomes substantially smaller. For datasets with a strong linear structure, by choosing an appropriate l, it is possible to achieve $l\sqrt{N_{\mathcal{A}}^l} < d$, thereby reducing computational overhead while attaining better regret performance than OFUL. We will demonstrate this in the subsequent experiments.

5.2 Computation Analysis

The computational cost of CS-LB mainly consists of two parts: the cost incurred during the warm-up phase (Algorithm 2), where the arm set is partitioned into clusters, and the per-round update cost during the subsequent online learning phase (Algorithm 3).

Regarding the per-round update cost in the online learning phase, the analysis in the preceding sections has shown that each update occurs within a cluster whose rank does not exceed l, and the dominant cost arises from the SVD decomposition. Hence, the per-round update cost is $O(l^2 d)$. Since l can be tuned in advance according to the requirements of the environment, the setting $l^2 < d$ is readily achievable, and thus CS-LB achieves a significant improvement in computational efficiency.

Regarding the computational cost of the warm-up phase, it mainly arises from the fact that each arm must perform an FD sketch update on each existing cluster to verify whether it can be added to that cluster. In the worst case, this incurs a total computational cost of $O(N \cdot N_{\mathcal{A}}^l \cdot l^2 d)$, where $N_{\mathcal{A}}^l$ denotes the number of clusters. Since both the number of clusters and the rank of each cluster grow gradually during the construction process, the actual computational cost is smaller than this worst-case bound. On the other hand, the additional cost incurred during the warm-up phase is one-time only and does not appear in the subsequent online learning stage. When the horizon T is sufficiently large, this one-time warm-up overhead becomes merely a constant term in the total computational cost. Thus, in terms of total computation, CS-LB achieves a significant efficiency gain.

6 Numerical Experiments

We evaluate the performance of CS-LB and compare it with OFUL (Abbasi-Yadkori et al., 2011), SOFUL (Kuzborskij et al., 2019) and CBSCFD (Chen et al., 2021). We generated different synthetic datasets for different scenarios to facilitate comparison.

6.1 Baseline

OFUL. As a classic algorithm for solving linear bandits, OFUL enjoys a tight regret guarantee, but inevitably incurs a computational cost of $O(d^2)$ per round. Our method reduces the per-round computational cost to $O(l^2 d)$ (excluding cluster partitioning), where l can be adjusted as needed.

SOFUL. SOFUL is the first method to adopt a sketching strategy to reduce computational costs in linear bandits, with both low time and space overhead. However, it faces the risk of linear regret when the sketch size is set inappropriately. Our method retains full matrix information within each cluster without truncating singular values, thereby ensuring convergence.

CBSCFD. CBSCFD improves upon SOFUL and achieves a tighter regret bound theoretically; however, it still fails to address the issue of linear regret arising from a heavy spectral tail.

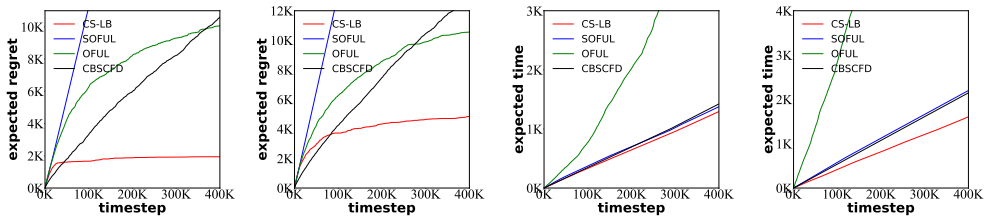
6.2 Experimental Setup

We set the additive random noise of the reward for each arm to follow a truncated Gaussian distribution with variance $R = 1$, and the truncation interval is $[\mu_{\boldsymbol{x}} - 3R, \mu_{\boldsymbol{x}} + 3R]$, where $\mu_{\boldsymbol{x}}$ denotes the expected reward corresponding to arm $\boldsymbol{x} \in \mathcal{A}$. Each component of every arm vector is drawn independently from a uniform distribution over $(0, 1)$. Each vector is then multiplied by a random integer coefficient and subsequently normalized to ensure that $\|\boldsymbol{x}\| \leq 1$. The ground-truth parameter $\boldsymbol{\theta}^*$ is generated in the same manner. Accordingly, we set $L = 1$, $S = 1$ and $\lambda = 1$.

6.3 Regret Convergence Verification

We set up two separate synthetic datasets as follows: (1) $N = 60, d = 50, l = 5$, (2) $N = 100, d = 50, l = 10$, where N is the number of arms, d is the dimension, and l is the sketch size.

We compare the regret and running time (seconds) of our method against SOFUL, CBSCFD, and OFUL on these two simulated datasets. All reported results are averaged over five independent runs. Averaged experimental results are shown in Figures 1.



(a) Regret on (1) (b) Regret on (2) (c) Running Time on (1) (d) Running Time on (2)

Figure 1: Average Regret and Running Time on Datasets (1) and (2)

For Dataset (1), according to Theorem 1, the maximum number of clusters is $N/l = 12$, so we have $l\sqrt{N_{\mathcal{A}}^l} < 5 \times \sqrt{12} < d = 50$. Therefore, CS-LB is expected to achieve the smallest regret among all compared algorithms. Meanwhile, SOFUL and CBSCFD suffer from non-convergence due to the chosen sketch size. The same reasoning applies to Dataset (2). The experimental results are consistent with our theoretical analysis.

We also observe that CS-LB achieves slightly lower running time than SOFUL and CBSCFD. We attribute this to the fact that CS-LB restricts the search for the arm to be played to the current cluster, whose size is significantly smaller than the entire arm set of size N. Consequently, on these datasets, CS-LB not only reduces computational cost but also lowers the search overhead.

In addition, we include a comparison with methods that directly apply traditional UCB strategies. The experimental results demonstrate the necessity of adopting linear models in large-scale scenarios, as well as the greater advantage of CS-LB. Details are provided in the appendix.

7 Conclusion

This paper addresses the non-convergence regret caused by cumulative spectral errors in sketching-based linear bandits. We propose Cluster Sketch Linear Bandits (CS-LB), the first sketching strategy that achieves convergence under a fixed sketch size, via clustering that partitions the action set, maintains lossless per-cluster sketches, and uses a sentinel to select the optimistic cluster each round. We prove that our method achieves sublinear regret with guaranteed convergence, and reduce per-round computation from $O(d^2)$ to $O(ld)$. Experimental evaluations on synthetic data show that CS-LB converges reliably, lowers computational overhead, and achieves superior regret performance.

We acknowledge that our current method still has limitations. Since the warm-up phase of the algorithm relies on reading the entire dataset, CS-LB is not yet applicable to more general linear contextual bandits with a varying arm set. We plan to extend our approach in future work.

AI USE STATEMENT

In this work, we used generative AI tools for assisting with translation. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods and interpreting results; and cleaning and reformatting dataset, supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for drafting parts of a research paper (section 1,2), creating part of software code (code implementation of sub-gaussian feedback setup for bandits), sourcing/searching for information (part of related work), editing a research paper to improve readability and identifying relevant literature. We have reviewed all AI-assisted work. We checked the LLM translations for errors, whether the LLM-generated content matched the method expressed in our paper, whether the polishing of our paper contained grammatical errors, and whether the references in our literature review were accurate. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The experimental code for our paper is available at the anonymous link `https://anonymous.4open.science/r/anonymous_11F014C`, and the experimental data and related experimental content can be reproduced using the experimental source code.

REFERENCES

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvari. Improved algorithms for linear stochastic bandits. In *Neural Information Processing Systems*, 2011.

Naoki Abe and Philip M. Long. Associative reinforcement learning using linear probabilistic concepts. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pp. 3–11, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1558606122.

Marc Abeille and Alessandro Lazaric. Linear thompson sampling revisited. In *International Conference on Artificial Intelligence and Statistics*, 2016. URL `https://api.semanticscholar.org/CorpusID:8385089`.

Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pp. III–1220–III–1228. JMLR.org, 2013.

Alexandr Andoni, Chengyu Lin, Ying Sheng, Peilin Zhong, and Ruiqi Zhong. Subspace embedding and linear regression with orlicz norm. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 224–233. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/andoni18a.html`.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, 3(null):397–422, March 2003. ISSN 1532-4435.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2–3):235–256, May 2002. ISSN 0885-6125. doi: 10.1023/A:1013689704352. URL `https://doi.org/10.1023/A:1013689704352`.

Cheng Chen, Luo Luo, Weinan Zhang, Yong Yu, and Yijiang Lian. Efficient and robust high-dimensional linear contextual bandits. IJCAI'20, 2021. ISBN 9780999241165.

Wei Chu, Lihong Li, L. Reyzin, and Robert E. Schapire. Contextual bandits with linear payoff functions. In *International Conference on Artificial Intelligence and Statistics*, 2011. URL `https://api.semanticscholar.org/CorpusID:1452971`.

Varsha Dani, Thomas P. Hayes, and Sham M. Kakade. Stochastic linear optimization under bandit feedback. In *Annual Conference Computational Learning Theory*, 2008. URL https://api.semanticscholar.org/CorpusID:9134969.

Rohan Deb, Qiaobo Li, Mayank Shrivastava, and Arindam Banerjee. Beyond johnson-lindenstrauss: Uniform bounds for sketched bilinear forms, 2025. URL https://arxiv.org/abs/2509.21847.

Mina Ghashami, Edo Liberty, Jeff M. Phillips, and David P. Woodruff. Frequent directions: Simple and deterministic matrix sketching. *SIAM J. Comput.*, 45(5):1762–1792, January 2016. ISSN 0097-5397. doi: 10.1137/15M1009718. URL https://doi.org/10.1137/15M1009718.

Ilja Kuzborskij, Leonardo Cella, and Nicolò Cesa-Bianchi. Efficient linear bandits through matrix sketching. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 177–185. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/kuzborskij19a.html.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pp. 661–670, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605587998. doi: 10.1145/1772690.1772758. URL https://doi.org/10.1145/1772690.1772758.

Paat Rusmevichientong and John N. Tsitsiklis. Linearly parameterized bandits. *Math. Oper. Res.*, 35(2):395–411, May 2010. ISSN 0364-765X. doi: 10.1287/moor.1100.0446. URL https://doi.org/10.1287/moor.1100.0446.

Zhao Song, David P. Woodruff, and Peilin Zhong. Low rank approximation with entrywise l1-norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pp. 688–701, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345286. doi: 10.1145/3055399.3055431. URL https://doi.org/10.1145/3055399.3055431.

Liang Tang, Romer Rosales, Ajit Singh, and Deepak Agarwal. Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pp. 1587–1594, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450322638. doi: 10.1145/2505515.2514700. URL https://doi.org/10.1145/2505515.2514700.

Ambuj Tewari and Susan A. Murphy. From ads to interventions: Contextual bandits in mobile health. In *Mobile Health - Sensors, Analytic Methods, and Applications*, 2017. URL https://api.semanticscholar.org/CorpusID:18778220.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285–294, 1933. URL https://api.semanticscholar.org/CorpusID:120462794.

Dongxie Wen, Hanyan Yin, Xiao Zhang, Peng Zhao, Lijun Zhang, and Zhewei Wei. Revisiting matrix sketching in linear bandits: Achieving sublinear regret via dyadic block sketching. 2024. URL https://api.semanticscholar.org/CorpusID:273346688.

Xiaotian Yu, Michael R. Lyu, and Irwin King. Cbrap: Contextual bandits with random projection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. doi: 10.1609/aaai.v31i1.10888. URL https://ojs.aaai.org/index.php/AAAI/article/view/10888.

A Appendix: Additional Experiments

We have added a comparison with UCB1 (Auer et al., 2002), the classic UCB strategy for the traditional setting with a finite number of arms, to demonstrate the necessity of considering a linear model in the finite-arm case and to show that traditional UCB strategies cannot be directly applied in large-scale scenarios, where our method offers greater advantages.

UCB1. Although the classic UCB1 method does not involve matrix computations and thus incurs low computational cost, UCB1 ignores the linear structure of the dataset. When the number of arms N is large, its regret grows linearly with N. In contrast, our method exploits the correlations among arms via the linear structure within each cluster, and therefore outperforms traditional UCB1 methods on datasets with a strong linear structure.

A.1 Linear Correlation

To demonstrate the advantages of the linear model, we incorporate linear correlation into the generation of our synthetic dataset. In generating the data, we introduce a linear correlation coefficient m^l, representing the number of linearly correlated vectors generated in succession. The arm set generation procedure incorporating m^l is as follows: Assuming that N arms are to be generated, these arms are divided into N/m^l groups for generation. Each group consists of m^l vectors that share the same direction but differ in their coefficients. The resulting dataset exhibits the following characteristics:

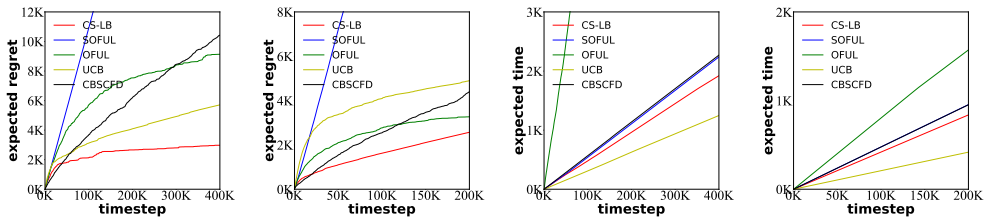
$$\alpha_{1,1}\boldsymbol{x}^1, ..., \alpha_{1,m^l}\boldsymbol{x}^1,$$
$$\alpha_{2,1}\boldsymbol{x}^2, ..., \alpha_{2,m^l}\boldsymbol{x}^2,$$
$$...$$
$$\alpha_{N/m^l,1}\boldsymbol{x}^{N/m^l}, ..., \alpha_{N/m^l,m^l}\boldsymbol{x}^{N/m^l},$$

where $\alpha_{1,1} > 0$ is a randomly generated real-valued coefficient. Finally, the dataset generated above is normalized by $\boldsymbol{x}/\|\boldsymbol{x}\|$.

A.2 Comparing with UCB1

For the fixed-arm-set setting, a straightforward approach is the classic UCB1 algorithm, which incurs no matrix operations and thus achieves the fastest running time. However, UCB1 fails to exploit the internal structure of the linear model, causing its regret to deteriorate as the number of arms grows. To demonstrate that the linear bandit problem is not trivially solvable by a simple UCB1 strategy, we generate the following two synthetic datasets according to the experimental setup for linear correlation described above: (3) $N = 100, d = 50, l = 10, m^l = 2$, (4) $N = 200, d = 50, l = 5, m^l = 10$.

Based on the definition of linear correlation coefficient m^l above, Dataset (4) has higher linear correlation and a smaller rank than Dataset (3). All reported results are averaged over five independent runs. Averaged experimental results are shown in Figures 2.



(a) Regret on (3) (b) Regret on (4) (c) Running Time on (3) (d) Running Time on (4)

Figure 2: Average Regret and Running Time on Datasets (3) and (4)

Experimental results demonstrate that, while UCB1 is computationally cheaper, OFUL and CS-LB achieve superior convergence rates as the arm set expands, by effectively exploiting the linear

structure. Hence, sketching strategies retain considerable practical value for fixed-arm-set linear bandits, particularly when the arm set is large.

B Appendix: Supplementary Proof

B.1 Notation in Proof

To facilitate a formal proof, we provide the following definitions.

Let $\mathcal{C}^{all} \longleftarrow CSW(\mathcal{A}, l)$ and $\boldsymbol{x}_t$ is arm selected in round t. We use $\mathcal{C}^{\boldsymbol{x}} \in \mathcal{C}^{all}$ to denote the cluster that $\boldsymbol{x}$ belongs to after running Algorithm 2.

When running CS-LB, for each cluster $\mathcal{C} \in \mathcal{C}^{all}$, we define:

$$\boldsymbol{X}_{\mathcal{C},t} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_t]^\top_{x_i \in \mathcal{C}, 1 \leq i \leq t}, \tag{4}$$

$$\boldsymbol{V}_{\mathcal{C},t} = \lambda \boldsymbol{I} + \sum_{s=1, \boldsymbol{x}_s \in \mathcal{C}}^{t} \boldsymbol{x}_s \boldsymbol{x}_s^\top, \tag{5}$$

$$\bar{\boldsymbol{V}}_{\mathcal{C},t} = \lambda \boldsymbol{I} + \boldsymbol{S}_{\mathcal{C},t}^\top \boldsymbol{S}_{\mathcal{C},t}, \tag{6}$$

$$\boldsymbol{\beta}_{\mathcal{C},t} = \sum_{s=1, \boldsymbol{x}_s \in \mathcal{C}}^{t} y_s \boldsymbol{x}_s, \tag{7}$$

$$\hat{\boldsymbol{\theta}}_{\mathcal{C},t} = \bar{\boldsymbol{V}}_{\mathcal{C},t}^{-1} \boldsymbol{\beta}_{\mathcal{C},t}. \tag{8}$$

We use $T_{\mathcal{C},t}$ to denote the number of times each cluster has been selected up to round t, i.e.

$$T_{\mathcal{C},t} = \sum_{s=1}^{t} \mathbb{I}\{\boldsymbol{x}_s \in \mathcal{C}\} = \sum_{s=1, \boldsymbol{x}_s \in \mathcal{C}}^{t} 1. \tag{9}$$

It is obvious that $\sum_{\mathcal{C} \in \mathcal{C}^{all}} T_{\mathcal{C},t} = t$.

We use

$$\boldsymbol{V}_t = \lambda \boldsymbol{I} + \sum_{s=1}^{t} \boldsymbol{x}_s \boldsymbol{x}_s^\top, \tag{10}$$

$$\boldsymbol{\beta}_t = \sum_{s=1}^{t} y_s \boldsymbol{x}_s, \tag{11}$$

$$\boldsymbol{\theta}_t = \boldsymbol{V}_t^{-1} \boldsymbol{\beta}_t \tag{12}$$

to denote the statistical information maintained during the execution of OFUL.

B.2 Technical Lemmas

This section collects the key, previously established properties and lemmas that we will use in our proofs.

Lemma 2. *For any positive semi-definite matrix $\boldsymbol{A}$ and any vector $\boldsymbol{x}, \boldsymbol{y}$, we have $|\boldsymbol{x}^\top \boldsymbol{y}| \leq \|\boldsymbol{x}\|_{\boldsymbol{A}} \|\boldsymbol{y}\|_{\boldsymbol{A}^{-1}}$.*

Proof.

$$|\boldsymbol{x}^\top \boldsymbol{y}| = \boldsymbol{x}^\top \boldsymbol{A}^{\frac{1}{2}} \boldsymbol{A}^{-\frac{1}{2}} \boldsymbol{y} \tag{13}$$

$$\leq \left\| \boldsymbol{A}^{\frac{1}{2}} \boldsymbol{x} \right\|_2 \left\| \boldsymbol{A}^{-\frac{1}{2}} \boldsymbol{y} \right\|_2 \text{ (Cauchy-Schwartz inequality)} \tag{14}$$

$$= \|\boldsymbol{x}\|_{\boldsymbol{A}} \|\boldsymbol{y}\|_{\boldsymbol{A}^{-1}}. \tag{15}$$

□

Lemma 3. *(Theorem 2 in Abbasi-Yadkori et al. (2011)) For any $\delta > 0$, with probability at least $1-\delta$, for all $t \geq 0$, $\boldsymbol{\theta}^*$ lies in the set*

$$\left\{ \boldsymbol{\theta} \in \mathbb{R}^d : \left\| \hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta} \right\|_{\boldsymbol{V}_t} \leq R\sqrt{2\log\left(\frac{det(\boldsymbol{V}_t)^{1/2} det(\lambda \boldsymbol{I})^{-1/2}}{\delta}\right)} + \sqrt{\lambda} S \right\} \tag{16}$$

Lemma 4. *According to Abbasi-Yadkori et al. (2011) and Kuzborskij et al. (2019), with sketch size l, we have*

$$\begin{aligned} \sum_{t=1}^{T} \min\{1, \|\boldsymbol{x}_t\|^2_{\boldsymbol{V}_t^{-1}}\} &\leq 2\ln \frac{det(\boldsymbol{V}_t)}{det(\lambda \boldsymbol{I})} \\ &\leq 2d\ln(1+\varepsilon_l) + 2l\ln\left(1 + \frac{tL^2}{l\lambda}\right). \end{aligned}$$

where $\varepsilon_l = \min_{k=0,\ldots,l-1} \frac{\lambda_{d-k}+\lambda_{d-k+1}+\ldots+\lambda_d}{\lambda(l-k)}$ and $\lambda_1 > \ldots > \lambda_d$ are the eigenvalues of the correlation matrix $\boldsymbol{V}_t = \lambda \boldsymbol{I} + \sum_{s=1}^{t} \boldsymbol{x}\boldsymbol{x}^\top$, ε_l is upper bounded by the spectral tail of the covariance matrix under l sketch size.

B.3 Proof of Lemma 1

We first derive the following lemma from Lemma 10 in Abbasi-Yadkori et al. (2011):

Lemma 5. *(Sketch-Trace Inequality) For a given sketch size l and any cluster $\mathcal{C} \in \mathcal{C}^{all}$, we have*

$$det(\boldsymbol{V}_{\mathcal{C},t}) = det(\bar{\boldsymbol{V}}_{\mathcal{C},t}) \leq \left(\lambda + tL^2/l\right)^l \cdot \lambda^{d-l}. \tag{17}$$

Proof. First, according to Proposition 3 in Kuzborskij et al. (2019), no singular value truncation is performed during the sketch update process of CS-LB. Therefore, we have

$$\boldsymbol{S}_{\mathcal{C},t}^\top \boldsymbol{S}_{\mathcal{C},t} = \boldsymbol{X}_{\mathcal{C},t}^\top \boldsymbol{X}_{\mathcal{C},t} = \sum_{s=1, \boldsymbol{x}_s \in \mathcal{C}}^{t} \boldsymbol{x}_s \boldsymbol{x}_s^\top. \tag{18}$$

Then we have

$$\boldsymbol{V}_{\mathcal{C},t} = \bar{\boldsymbol{V}}_{\mathcal{C},t}.$$

and according to the process of Algorithm 2, the rank of $\boldsymbol{S}_{\mathcal{C},t}^\top \boldsymbol{S}_{\mathcal{C},t}$ is less than l.

Let $\alpha_1, ..., \alpha_d$ be the eigenvalues of $\boldsymbol{V}_{\mathcal{C},t}$ and $\widetilde{\alpha}_1, ..., \widetilde{\alpha}_d$ be the eigenvalues of $\boldsymbol{S}_{\mathcal{C},t}^\top \boldsymbol{S}_{\mathcal{C},t}$. Because $rank(\boldsymbol{S}_{\mathcal{C},t}^\top \boldsymbol{S}_{\mathcal{C},t}) \leq l$, then, by the properties of singular value decomposition, we obtain

$$\alpha_1, ..., \alpha_l = \lambda + \widetilde{\alpha}_1, ..., \lambda + \widetilde{\alpha}_d,$$

and

$$\alpha_{l+1} = ... = \alpha_d = \lambda.$$

Then $det(\boldsymbol{V}_{\mathcal{C},t}) = \prod_{i=1}^{l} \alpha_i \cdot \lambda^{d-l}$. We have

$$\prod_{i=1}^{l} \alpha_i \leq \left(\frac{\sum_{i=1}^{l} \alpha_i}{l}\right)^l \tag{19}$$

$$= \left(\frac{l\lambda + \sum_{i=1}^{l} \widetilde{\alpha}_i}{l}\right)^l \tag{20}$$

$$= \left(\frac{l\lambda + trace(\boldsymbol{S}_{\mathcal{C},t}^\top \boldsymbol{S}_{\mathcal{C},t})}{l}\right)^l \tag{21}$$

$$= \left(\frac{l\lambda + trace(\boldsymbol{X}_{\mathcal{C},t}^\top \boldsymbol{X}_{\mathcal{C},t})}{l}\right)^l. \tag{22}$$

The first inequality is because AM-GM inequality. According to Lemma 10 in Abbasi-Yadkori et al. (2011), we have $trace(\boldsymbol{X}_{\mathcal{C},t}^{\top}\boldsymbol{X}_{\mathcal{C},t}) \leq \sum_{s=1,\boldsymbol{x}_s\in\mathcal{C}}^{t} L^2 \leq T_{\mathcal{C},t}L^2 \leq tL^2$. Substituting the inequality yields

$$det(\boldsymbol{V}_{\mathcal{C},t}) = det(\bar{\boldsymbol{V}}_{\mathcal{C},t}) \leq \left(\lambda + tL^2/l\right)^l \cdot \lambda^{d-l}. \tag{23}$$

□

Combining Lemma 3 and Lemma 5, because $\boldsymbol{V}_{\mathcal{C},t} = \bar{\boldsymbol{V}}_{\mathcal{C},t}$, we obtain that for each cluster $\mathcal{C}$, with probability at least $1-\delta'$, the following holds:

$$\left\|\hat{\boldsymbol{\theta}}_{\mathcal{C},t} - \boldsymbol{\theta}^*\right\|_{\boldsymbol{V}_{\mathcal{C},t}} \leq R\sqrt{2\log\left(\frac{det(\boldsymbol{V}_{\mathcal{C},t})^{1/2}det(\lambda\boldsymbol{I})^{-1/2}}{\delta'}\right)} + \sqrt{\lambda}S \tag{24}$$

$$\leq R\sqrt{2\log\left(\frac{\left(\lambda^{d-l}\left(\lambda + tL^2/l\right)^l\right)^{1/2}\lambda^{-d/2}}{\delta'}\right)} + \sqrt{\lambda}S \tag{25}$$

$$= R\sqrt{2l\log(1+\frac{tL^2}{l\lambda}) + 2\log\frac{1}{\delta'}} + S\sqrt{\lambda}. \tag{26}$$

Let $\delta' = \frac{\delta}{N_{\mathcal{A}}^l}$, then for any $\mathcal{C} \in \mathcal{C}^{all}$, with probability at least $1-\delta$, we have

$$\left\|\hat{\boldsymbol{\theta}}_{\mathcal{C},t} - \boldsymbol{\theta}^*\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C},t}} = \left\|\hat{\boldsymbol{\theta}}_{\mathcal{C},t} - \boldsymbol{\theta}^*\right\|_{\boldsymbol{V}_{\mathcal{C},t}} \leq R\sqrt{2l\log(1+\frac{tL^2}{l\lambda}) + 2\log\frac{N_{\mathcal{A}}^l}{\delta}} + S\sqrt{\lambda}.$$

Because $N_{\mathcal{A}}^l \leq N/r$, we finish the proof of Lemma 1.

B.4 Proof of Theorem 1

According to Lemma 1 and sentinel strategy, with probability at least $1-\delta$,the upper confidence bound satisfies

$$\boldsymbol{x}^{\top}\boldsymbol{\theta}^* \leq \boldsymbol{x}^{\top}\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}},t} + \beta_t(\delta)\left\|\boldsymbol{x}\right\|_{\boldsymbol{V}_{\mathcal{C}^{\boldsymbol{x}},t}^{-1}} \tag{27}$$

$$= \boldsymbol{x}^{\top}\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}},t} + \beta_t(\delta)\left\|\boldsymbol{x}\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}},t}^{-1}} \tag{28}$$

$$\leq U(\mathcal{C}^{\boldsymbol{x}}). \tag{29}$$

According to the arm selection policy of CS-LB, we have

$$\boldsymbol{x}_t^{\top}\hat{\boldsymbol{\theta}}_{\mathcal{W}_t,t} + \beta_t(\delta)\left\|\boldsymbol{x}_t\right\|_{\bar{\boldsymbol{V}}_{\mathcal{W}_t,t}^{-1}} = U(\mathcal{W}_t) = \max\{U(\mathcal{C})\} = U(\mathcal{C}^{\boldsymbol{x}_t}) = \boldsymbol{x}_t^{\top}\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}_t},t} + \beta_t(\delta)\left\|\boldsymbol{x}_t\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}_t},t}^{-1}}. \tag{30}$$

Then

$$(\boldsymbol{x}^*)^{\top}\boldsymbol{\theta}^* \leq (\boldsymbol{x}^*)^{\top}\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}^*},t} + \beta_t(\delta)\left\|\boldsymbol{x}^*\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}^*},t}^{-1}} \tag{31}$$

$$\leq U(\mathcal{C}^{\boldsymbol{x}^*}) \tag{32}$$

$$\leq U(\mathcal{W}_t) \tag{33}$$

$$= \boldsymbol{x}_t^{\top}\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}_t},t} + \beta_t(\delta)\left\|\boldsymbol{x}_t\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}_t},t}^{-1}}. \tag{34}$$

Then, for the per-round regret $regret_t$ in round t, we can get

$$regret_t = \max_{\boldsymbol{x}\in\mathcal{A}}\langle\boldsymbol{x},\boldsymbol{\theta}^*\rangle - \langle\boldsymbol{x}_t,\boldsymbol{\theta}^*\rangle \tag{35}$$

$$= \langle\boldsymbol{x}^*,\boldsymbol{\theta}^*\rangle - \langle\boldsymbol{x}_t,\boldsymbol{\theta}^*\rangle \tag{36}$$

$$\leq \left\langle\boldsymbol{x}_t,\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}_t},t}\right\rangle - \langle\boldsymbol{x}_t,\boldsymbol{\theta}^*\rangle + \beta_t(\delta)\left\|\boldsymbol{x}_t\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}_t},t}^{-1}} \tag{37}$$

$$= \left\langle\boldsymbol{x}_t,\hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}_t},t} - \boldsymbol{\theta}^*\right\rangle + \beta_t(\delta)\left\|\boldsymbol{x}_t\right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}_t},t}^{-1}}. \tag{38}$$

According to Lemma 2, we have

$$\left\langle \boldsymbol{x}_t, \hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}_t},t} - \boldsymbol{\theta}^* \right\rangle \leq \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}} \left\| \hat{\boldsymbol{\theta}}_{\mathcal{C}^{\boldsymbol{x}_t},t} - \boldsymbol{\theta}^* \right\|_{\bar{\boldsymbol{V}}_{\mathcal{C}^{\boldsymbol{x}_t},t}} \leq \beta_t(\delta) \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}}. \tag{39}$$

Substituting into the above gives

$$regret_t \leq 2\beta_t(\delta) \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}}. \tag{40}$$

Since the regret $regret_t \leq 2LS$ and $\beta_t(\delta) > \sqrt{\lambda}S$, then

$$Regret(T) = \sum_{t=1}^{T} regret_t \tag{41}$$

$$\leq 2\sum_{t=1}^{T} \min\{\beta_t(\delta) \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}}, LS\} \tag{42}$$

$$\leq 2\max\{1, \frac{L}{\sqrt{\lambda}}\}\beta_T(\delta) \sum_{t=1}^{T} \min\{1, \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}}\}. \tag{43}$$

On the other hands, we have

$$\sum_{t=1}^{T} \min\{1, \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}}\} = \sum_{\mathcal{C}\in\mathcal{C}^{all}} \sum_{t=1,\boldsymbol{x}_t\in\mathcal{C}}^{T} \min\{1, \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C},t}}\} = \sum_{\mathcal{C}\in\mathcal{C}^{all}} \sum_{t=1,\boldsymbol{x}_t\in\mathcal{C}}^{T} \min\{1, \|\boldsymbol{x}_t\|_{\boldsymbol{V}^{-1}_{\mathcal{C},t}}\}. \tag{44}$$

By Cauchy-Schwartz inequality

$$\sum_{t=1,\boldsymbol{x}_t\in\mathcal{C}}^{T} \min\{1, \|\boldsymbol{x}_t\|_{\boldsymbol{V}^{-1}_{\mathcal{C},t}}\} \leq \sqrt{T_{\mathcal{C},T} \sum_{t=1,\boldsymbol{x}_t\in\mathcal{C}}^{T} \min\{1, \|\boldsymbol{x}_t\|^2_{\boldsymbol{V}^{-1}_{\mathcal{C},t}}\}}. \tag{45}$$

Since $rank(\boldsymbol{X}^\top_{\mathcal{C},t}\boldsymbol{X}_{\mathcal{C},t}) = rank(\boldsymbol{S}^\top_{\mathcal{C},t}\boldsymbol{S}_{\mathcal{C},t}) \leq l$ for each cluster $\mathcal{C}$, we have $\varepsilon_l = 0$ within that cluster. Thus according to Lemma 4 we have

$$\sum_{t=1,\boldsymbol{x}_t\in\mathcal{C}}^{T} \min\{1, \|\boldsymbol{x}_t\|^2_{\boldsymbol{V}^{-1}_{\mathcal{C},t}}\} \leq 2l \ln\left(1 + \frac{tL^2}{l\lambda}\right). \tag{46}$$

Combining the above inequalities yields

$$\sum_{t=1}^{T} \min\{1, \|\boldsymbol{x}_t\|_{\bar{\boldsymbol{V}}^{-1}_{\mathcal{C}^{\boldsymbol{x}_t},t}}\} \leq \sqrt{2l \ln\left(1 + \frac{tL^2}{l\lambda}\right)} \sum_{\mathcal{C}\in\mathcal{C}^{all}} \sqrt{T_{\mathcal{C},T}}. \tag{47}$$

By Cauchy-Schwartz inequality and the fact that $\sum_{\mathcal{C}\in\mathcal{C}^{all}} T_{\mathcal{C},T} = T$, we can get

$$\sum_{\mathcal{C}\in\mathcal{C}^{all}} \sqrt{T_{\mathcal{C},T}} \leq \sqrt{N^l_{\mathcal{A}} T}.$$

At the same time, we known that $\beta_T(\delta) = \widetilde{O}(\sqrt{l})$ and we set $\lambda \geq \max\{1, L^2\}$; with probability at least $1-\delta$, we get that the regret is

$$Regret(T) \leq 2\beta_T(\delta)\sqrt{2lN^l_{\mathcal{A}} T \ln\left(1 + \frac{TL^2}{l\lambda}\right)} = \widetilde{O}(l\sqrt{N^l_{\mathcal{A}} T}). \tag{48}$$

Thus, we complete the proof.