Design of Experiments for Stochastic Contextual Linear Bandits

Andrea Zanette*
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA
zanette@berkeley.edu

Kefan Dong*
Department of Computer Science
Stanford University
Stanford, CA
kefandong@stanford.edu

Jonathan Lee*
Department of Computer Science
Stanford University
Stanford, CA
jnl@stanford.edu

Emma Brunskill
Department of Computer Science
Stanford University
Stanford, CA
ebrun@cs.stanford.edu

Abstract

In the stochastic linear contextual bandit setting there exist several minimax procedures for exploration with policies that are *reactive* to the data being acquired. In practice, there can be a significant engineering overhead to deploy these algorithms, especially when the dataset is collected in a distributed fashion or when a human in the loop is needed to implement a different policy. Exploring with a single non-reactive policy is beneficial in such cases. Assuming some batch contexts are available, we design a single stochastic policy to collect a good dataset from which a near-optimal policy can be extracted. We present a theoretical analysis as well as numerical experiments on both synthetic and real-world datasets.

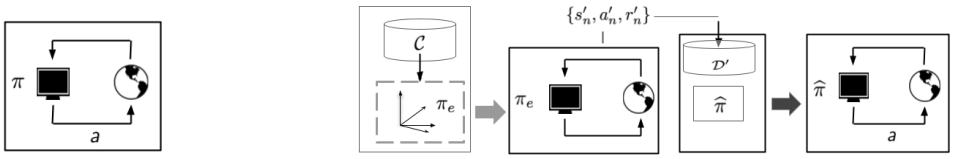
1 Introduction

Designers in many application areas can now deploy personalized decision policies. For example, recent work used manual classification of students to deploy semi-personalized literacy tips text messages which can be done easily by leveraging popular text messaging platforms. This work found significant improvements in young children's literacy scores (Doss et al. (2019)), but also noted the substantial variation in when and what type of text messages were most effective for different families (Cortes et al. (2019); Doss et al. (2019)).

This setting and many others might substantially benefit from data-driven contextualized decision policies that optimize the desired expected outcome (such as literacy). Online machine learning methods like multi-armed bandits and reinforcement learning, that adaptively change interventions

*The first three authors contributed equally. The work was fully completed while Andrea Zanette was a PhD Candidate in the Institute for Computational and Mathematical Engineering at Stanford University. Future revision of this article will be made available at https://arxiv.org/abs/2107.09912.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).



(a) Traditional online RL framework

(b) Design of experiment with an online and offline component

Figure 1: Comparison between the traditional RL setting and design of experiments

in response to outcomes in a closed loop process (see Figure 1a), may not yet be practical for such settings due to the expertise and infrastructure needed. However running an experiment with a fixed decision policy is likely to be both simple logistically (since such organizations already often design such decision policies by hand) and palatable, since many areas (education, healthcare, social sciences) commonly deploy experiments across a few conditions to find the best approach. For these reasons, a key opportunity is to design *static* or *non-reactive* policies that can be used to gather data to identify optimal contextualized decision policies.

More generally, a fixed data collection strategy is practically desirable (1) whenever multiple agents collect data asynchronously and communication to update their policy is difficult or impossible, and (2) whenever changing the policy requires a significant overhead either in the engineering infrastructure or in the training of human personnel. Indeed several works limit the number of policy switches with minimal sample complexity impact (Han et al., 2020; Ruan et al., 2020; Ren et al., 2020; Bai et al., 2019; Wang et al., 2021). In this work, motivated by the above settings, we go further and look for a *single*, *non-reactive* policy for data collection.

Setting and goal We consider the linear stochastic contextual bandit setting where each context $s \in \mathcal{S}$ is sampled from a distribution μ and a context-dependent action set $\mathcal{A}_s$ is made available to the learner. The bandit instance is defined by a feature extractor $\phi(s, a) \in \mathbb{R}^d$ and some unknown parameter $\theta^\star \in \mathbb{R}^d$. Upon choosing an action $a \in \mathcal{A}_s$, the linear reward function $r(s, a) = \phi(s, a)^\top \theta^\star + \eta$ is revealed to the learner corrupted by mean zero 1-subGaussian noise η. Our goal is to construct an exploration policy π_e to gather a dataset, such that after that dataset is gathered, we can extract a near optimal policy $\widehat{\pi}$ from it.

Perhaps surprisingly, there has been relatively little work on this setting. Prior work on exploration to quickly identify a near-optimal policy focuses on best-arm identification using adaptive policies that react to the observed rewards (Soare et al., 2014; Tao et al., 2018; Jedra and Proutiere, 2020) or design of experiments that produces a non-reactive policy for data collection (Kiefer and Wolfowitz, 1960; Esfandiari et al., 2019; Lattimore and Szepesvari, 2020); both lines of work assume that a *single*, repeated context with unchanging action set is presented to the learner. In contrast we are interested in identifying near-optimal context-specific decision policies. The closest related work (Ruan et al., 2020) investigates our task as a subtask for online regret learning, but requires an amount of data that scales as $\Omega(d^{16})$, which is impractical in applications with even moderate d.[2]

Without any apriori information about the problem, no algorithm can do much better than deploying a random policy, which can require an amount of data that scales exponentially in d, see appendix G.1. However, in many common applications, *prior data about the context distribution* $\mu(s)$ and the state–action feature representation ϕ is available. For example, an organization will often know information about its customers and specify the feature representation used for state–action spaces in advance of trying out such actions.[3] The initially available state contexts are referred to as *offline* (state) contexts data.

Our algorithm leverages historical context data to enable data efficient design of experiments for stochastic linear contextual bandits. It uses offline context data $\mathcal{C}$ to design a non-reactive policy π_e to collect new, *online* data where reward feedback is observed (see Figure 1b), and uses the resulting dataset $\mathcal{D}'$ to learn a near-optimal on average decision policy $\widehat{\pi}$ for future use. We highlight that the algorithm does not get to adjust the exploratory policy π_e while the online data is being collected.

[2]Note that the number of offline data in (Ruan et al., 2020) is independent of $1/\epsilon$. But for most of the practical settings, our bound d^3/ϵ^2 can be much smaller than d^{16}.

[3]In other words, given a set of previously observed states $s_1, \ldots, s_M$, and a known state–action representation ϕ, for any potential action a, we can compute the resulting representation $\phi(s, a)$.

Contributions We make the following contributions.

- Using past state contexts only, we design a single, non-reactive policy to acquire online data that can be used to compute a context-dependent decision policy that is near-optimal in expectation across the contexts with high probability, for future use.
- In order to identify an ϵ-optimal policy, our algorithm achieves the minimax lower bound $\Omega(\min\{d \log \sum_s \mathcal{A}_s, d^2)\}/\epsilon^2)$ on the number of online samples (ignoring log and constants), while keeping the number of offline state contexts required polynomially small ($O(d^2/\epsilon^2)$ or $O(d^3/\epsilon^2)$).
- Our experiments on a simulation and on a learning to rank Yahoo dataset show our strategic design of experiments approach can learn better decision policies with less exploration data compared to standard exploration strategies that fail to exploit structure encoded in the historical contexts.

2 Related Work

There is an extensive line of literature on linear bandits, with most papers focusing on online regret. Research on sample complexity results has focused primarily on adaptive algorithms for non-contextual bandits with a simple or vectored action space (Soare et al., 2014; Tao et al., 2018; Xu et al., 2018; Fiez et al., 2019; Jedra and Proutiere, 2020; Degenne et al., 2020). Their sample complexity bounds depend on the suboptimality gap of the action set, and are therefore, instance-dependent. The design of experiments literature (Kiefer and Wolfowitz, 1960; Esfandiari et al., 2019; Lattimore and Szepesvari, 2020; Ruan et al., 2020) has also focused on non-contextual bandits, but in designing a non-adaptive policy such that the resulting dataset can be used to identify an optimal arm with high probability.

A few recent papers consider learning the optimal policy for contextual bandits in the pure exploration setting (Deshmukh et al., 2018; Ruan et al., 2020). Deshmukh et al. (2018) propose an algorithm that yields an online reactive policy for when the reward is generated by a RKHS function on the input space: in contrast we focus on non-adaptive policies for stochastic linear contextual bandits.

The most related paper to this work is by Ruan et al. (2020). The main focus of Ruan et al. (2020) is to minimize the number of policy switches in the online phase while keeping the expected regret optimal. Directly applying Theorem 6 of Ruan et al. (2020) yields a $\widetilde{O}(d^2/\epsilon^2)$ online sample complexity and $\widetilde{O}(d^{16})$ offline sample complexity. Note that the offline sample complexity is independent of ϵ. As a trade-off, Ruan et al. (2020) have a highly suboptimal dependence on d, which leads to burn-in phase too large to be practically useful; their algorithm also suffers from a high computational complexity. However, note their work is focused on a different objective than what we have here.

3 Setup

We consider the stochastic linear contextual bandit model with stochastic contexts. A bandit instance is characterized by a tuple $\langle \mathcal{S}, \mathcal{A}, \mu, r \rangle$ where $\mathcal{S}$ is the context space and μ is the context distribution. For a context $s \in \mathcal{S}$, the action space is denoted by $\mathcal{A}_s$. The feature map $\phi : (s, a) \mapsto \phi(s, a) \in \mathbb{R}^d$ is assumed to be known and defines the linear reward model $r(s, a) = \phi(s, a)^\top \theta^\star + \eta$ for some $\theta^\star \in \mathbb{R}^d$ parameter and some mean-zero 1-subgaussian random variable η.

We occasionally use the $\widetilde{O}$ notation to suppress $\mathrm{polylog}$ factors of the input parameters $d, \lambda, \frac{1}{\delta}$. We write $f \lesssim g$ if $f = O(g)$, and $f \lessapprox g$ if $f = \widetilde{O}(g)$. We say $f = \Omega(g)$ if there exists a positive constant c such that $f \geq cg$. For a positive semi-definite matrix $\Sigma \in \mathbb{R}^{d \times d}$ and a vector $x \in \mathbb{R}^d$, let $\|x\|_\Sigma = \sqrt{x^\top \Sigma x}$. For two symmetric matrices A, B, we say $A \preccurlyeq B$ if $B - A$ is positive semi-definite.

Our analysis consists of an offline and online component. We often add $'$ to indicate the online quantities, e.g., we denote with $s_1, s_2, \ldots$ the offline contexts and with $s'_1, s'_2, \ldots$ the online contexts.

3.1 Objective and Error Decomposition

As depicted in Figure 1, our approach is to leverage offline state contexts $\mathcal{C} = \{s_1, \dots, s_M\}$ where $s_m \sim \mu$ to design a single stochastic policy π_e to acquire data from an online data stream $\mathcal{C}' = \{s'_1, \dots, s'_N\}$. This generates a dataset $\mathcal{D}' = \{(s'_n, a'_n, r'_n)\}_{n=1,\dots,N}$ where $s'_n \sim \mu, a'_n \sim \pi_e(s'_n)$ and $r'_n = r(s'_n, a'_n)$. Using $\mathcal{D}'$, the least-square predictor $\widehat{\theta}$ and the corresponding greedy policy $\widehat{\pi}$ can be extracted

$$\widehat{\theta} = \left(\Sigma'_N\right)^{-1} \sum_{i=1}^{N} \phi(s'_n, a'_n) r'_n, \qquad \widehat{\pi}(s) = \arg\max_{a \in \mathcal{A}_s} \phi(s,a)^\top \widehat{\theta} \tag{1}$$

where $\Sigma'_N = \left(\sum_{n=1}^{N} \phi(s'_n, a'_n)\phi(s'_n, a'_n)^\top + \lambda_{reg} I\right)$ is the empirical cumulative covariance matrix with regularization level λ_{reg}. The quality of the dataset $\mathcal{D}'$ (and of the whole two-step procedure) is measured by the suboptimality of the extracted policy $\widehat{\pi}$ obtained after data collection:

$$\mathbb{E}_{s\sim\mu}[\max_a \phi(s,a)^\top \theta^\star - \phi(s, \widehat{\pi}(s))^\top \theta^\star]. \tag{2}$$

Note that Eq. (2) measures the expectation over the contexts of the suboptimality between the resulting decision policy $\widehat{\pi}$ and the optimal policy. This is a looser criteria than a maximum norm bound which evaluates the error over any possible context s: in general this latter error may not be easily reduced if certain directions in feature space are rarely available.

A related objective is to minimize the maximum prediction error on the linear bandit instance. By least square regression analyses (Lattimore and Szepesvári, 2020), with probability at least $1-\delta$ we have

$$\mathbb{E}_{s\sim\mu} \max_a |\phi(s,a)^\top (\theta^\star - \widehat{\theta})| \leq \sqrt{\beta}\, \mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{(\Sigma'_N)^{-1}}, \tag{3}$$

where

$$\sqrt{\beta} = \min\Big\{ \underbrace{\sqrt{2\ln 2 |\sum_s \mathcal{A}_s| + \ln\frac{1}{\delta}}}_{\text{Small } \sum_s \mathcal{A}_s}, \underbrace{2\sqrt{2d\ln 6 + \ln\frac{1}{\delta}}}_{\text{Large } \sum_s \mathcal{A}_s} \Big\} + \underbrace{\sqrt{\lambda_{reg}}\|\theta^\star\|_2}_{\text{Regularization Effect}}. \tag{4}$$

The above expression assumes that the state-action-rewards (s'_n, a'_n, r'_n) are drawn i.i.d. from a fixed distribution. This is satisfied in our setting as the data-collection policy π_e is non-reactive to the online data. The parameter β governs the sample complexity as a function of the size of the state-action space and also highlights the impact of the regularization bias.

Small predictive error (Eq. (3)) can be used to bound the suboptimal gap of the greedy policy (Eq. (2)). Therefore to obtain good performance, it is sufficient to bound $\mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{(\Sigma'_N)^{-1}}$. This can be achieved by designing an appropriate sampling policy π_e to yield a set of (s'_n, a'_n, r'_n) tuples whose empirical cumulative covariance matrix Σ'_N is as 'large' as possible.

4 Algorithms

Reward-free LinUCB First, assume it is acceptable to have an algorithm that updates its policy reactively. In order to reduce $\mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{(\Sigma'_n)^{-1}}$ we could use an algorithm that, every time a context $s \sim \mu$ is observed, chooses the action $\arg\max_{a\in\mathcal{A}_s} \|\phi(s,a)\|_{(\Sigma'_n)^{-1}}$ where the norm $\|\phi(s,a)\|_{(\Sigma'_n)^{-1}}$ that represents the uncertainty is highest (cf. section 3.1). Such procedure is related (although not identical) to the well known globally optimal design of experiment applied to linear bandits (i.e., Lattimore and Szepesvari (2020)). Algorithmically, it corresponds to running the LinUCB algorithm (Abbasi-Yadkori et al., 2011) with the empirical reward function set to zero. In other words, the planner is essentially an *offline, reward-free* version of LinUCB. We emphasize that the associated non-stationary policy is generated offline and is not actively played. One can show that after $\approx d^2/\epsilon^2$ iterations the uncertainty $\sqrt{\beta}\, \mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{(\Sigma'_n)^{-1}} \leq \epsilon$; if the algorithm stores the observed reward $r(s,a)$ in every visited context s and chosen action a, the greedy policy that can be extracted from this dataset (of size $\approx d^2/\epsilon^2$) is ϵ-optimal (cf. Eqs. (2),(3)), as desired.

Unfortunately we cannot run this reward-free algorithm online, as its policy is *reactive* to the online stream of observed online contexts s and selected actions a, while we want a non-reactive policy.

Algorithm 1 PLANNER (Reward-free LINUCB)

Input: Contexts $\mathcal{C} = \{s_1, \ldots, s_M\}$, reg. λ_{reg}
$\Sigma_1 = \lambda_{reg} I$
$m = 1$
for $m = 1, 2, \ldots M$ **do**
  if $\det(\Sigma_m) > 2\det(\Sigma_{\underline{m}})$ or $m = 1$ **then**
    $\underline{m} \leftarrow m$
    $\Sigma_{\underline{m}} \leftarrow \Sigma_m$
  end if
  Define $\pi_m : s \mapsto \arg\max_{a \in \mathcal{A}_s} \|\phi(s,a)\|_{\Sigma_{\underline{m}}^{-1}}$
  $\Sigma_{m+1} = \Sigma_m + \alpha \phi_m \phi_m^\top$; $\phi_m = \phi(s_m, \pi_m(s_m))$
end for
return policy mixture π_e of $\{\pi_1, \ldots, \pi_M\}$

Algorithm 2 SAMPLER

Input: $\pi_e = \{\pi_1, \ldots, \pi_M\}$, reg. λ_{reg}
Set $\mathcal{D}' = \emptyset$
for $n = 1, 2, \ldots N$ **do**
  Receive context $s'_n \sim \mu$
  Sample $m \in [M]$ uniformly at random
  Select action $a'_n = \pi_m(s'_n)$
  Receive feedback reward r'_n
  Store feedback $\mathcal{D}' = \mathcal{D}' \cup \{s'_n, a'_n, r'_n\}$
end for
return dataset $\mathcal{D}'$

Our algorithm leverages this idea and consists of two subroutines: 1) the *planner* (Alg. 1) which operates on offline contexts and identifies a mixture policy π_e (this is the exploratory policy π_e mentioned in the introduction) 2) the *sampler* (Alg. 2) which runs π_e online to finally gather the dataset. This way, π_e is non-reactive to the online data.

Planner The purpose of the planner is to use past contexts to compute the exploratory policy. The planner runs the reward-free version of LINUCB on the offline context dataset $\mathcal{C}$ as described earlier in this section. This way, the planner selects the action a in the current offline context s_m that maximizes the uncertainty encoded in $\|\phi(s_m, a)\|_{\Sigma_{\underline{m}}^{-1}}$ where $\Sigma_{\underline{m}}$ is a scaled, regularized, cumulative covariance over the contexts parsed so far and the actions selected. Note this procedure is possible since the state–action function $\phi(s_m, a)$ is assumed to be known for any input (s_m, a) pair, and no actual rewards are observed. The doubling schedule yields a short descriptor for the planner's policies $\{\pi_1, \ldots, \pi_M\}$. The variable $\underline{m}$ indicates the last doubling update before iteration m.

A key choice is the parameter $\alpha \in (0, 1]$ in the cumulative covariance matrix update. The rationale is that when $\alpha < 1$ each rank-one update $\phi_m \phi_m^\top$ to the cumulative covariance gets *discounted*. The smaller α is, the more offline samples the planner needs to get to a sufficiently positive definite covariance matrix Σ_M. This choice effectively averages the updates and increases the estimation accuracy of the planner's covariance with respect to its conditional expectation.

Sampler Upon termination, the planner identifies a sequence of policies $\pi_1, \ldots, \pi_M$. Now consider the average policy π_e: every time an action is needed, π_e samples one index $m \in [M]$ uniformly at random and plays π_m. This is the policy that the sampler (Alg. 2) implements for $N = \alpha M \leq M$ fresh online contexts. Intuitively, executing the mixture policy π_e in the online phase produces the same covariance matrix as Σ_M (in expectation).

Upon playing π_m in state s'_n, the corresponding reward is observed and the tuple $(\{s'_n, a'_n, r'_n\})$ is stored. Since π_e is the average policy played by the planner, we expect that running π_e on the online dataset produces a covariance matrix Σ'_N similar to the planner's Σ_M. In a sense, the covariance matrix is a proxy for the amount of information acquired by the algorithm. This means the sampler acquires the same information that the planner would have acquired if it was acting online. Since the planner is the reward-free LINUCB algorithm, we expect its policy to efficiently reduce our uncertainty over the reward parameters and learn a near optimal policy; our analysis will make this intuition precise. Note the sampler's policy is non-reactive to the online stream of data.

Our work uses mixture policies; mixture policies are not essential, although they arise naturally from the way the algorithms are formulated.

5 Main Result

Theorem 1. *Fix $\epsilon > 0$, and consider running Alg. 1 for $M = \widetilde{\Omega}(\frac{d^2\beta}{\lambda_{reg}\epsilon^2})$ iterations and Alg. 2 for $N = \widetilde{\Omega}(\frac{d\beta}{\epsilon^2})$ iterations with regularization $\lambda_{reg} \in (\Omega(\ln(d/\delta)), d]$. Let $\widehat{\theta}, \widehat{\pi}$ be as in Eq. (1). For*

Sample Complexity Bounds				
		Small $\sum_s \mathcal{A}_s$	Large $\sum_s \mathcal{A}_s$	Large $\sum_s \mathcal{A}_s$
Offline Data	M	d^3/ϵ^2	d^3/ϵ^2	d^2/ϵ^2
Online Data	N	$(d \ln \sum_s \lvert\mathcal{A}_s\rvert)/\epsilon^2$	d^2/ϵ^2	d^2/ϵ^2
Regularization	λ_{reg}	1	1	d

Table 1: Sample complexity bounds (ignoring constants and log terms) to obtain an ϵ-optimal policy

any $\epsilon \leq 1$, with probability at least $1-\delta$ the expected maximum uncertainty

$$\mathbb{E}_{s\sim\mu} \max_{a\in\mathcal{A}_s} |\phi(s,a)^\top(\theta^\star - \widehat{\theta})| \leq \epsilon \tag{5}$$

and the suboptimality of the greedy policy $\widehat{\pi}$ satisfies

$$\mathbb{E}_{s\sim\mu} \max_{a\in\mathcal{A}_s} \left(\phi(s,a) - \phi(s,\widehat{\pi}(s))\right)^\top \theta^\star \leq 2\epsilon. \tag{6}$$

In Table 1 we instantiate the bounds from Theorem 1 in different settings, ignoring constants and log terms. This highlights different tradeoffs between the regularization λ_{reg} and the number of offline contexts ($M = N/\alpha$) needed to achieve the online minimax sample complexity lower bound $\approx \min\{d, \ln \sum_s |\mathcal{A}_s|\} \times d/\epsilon^2$ (Chu et al., 2011; Abbasi-Yadkori et al., 2011). [4] In the large action regime $\ln|\sum_s \mathcal{A}_s| \gtrapprox d$, a regularization level $\lambda_{reg} \approx d$ gives the optimal online sample complexity $N \approx d^2/\epsilon^2$ while requiring a context dataset of the same size ($M \approx d^2/\epsilon^2$, by choosing $\alpha = 1$).

More often, a lower level of regularization can be preferable to introduce less bias. For example $\lambda_{reg} = 1$ is a common choice in linear bandits (Abbasi-Yadkori et al., 2011; Chu et al., 2011). When the cumulative covariance matrix is less regularized, we may need to compensate with additional offline data to ensure the covariance matrices are accurately estimated, which is achieved by setting $M = N/\alpha \approx dN$ (i.e., $\alpha \approx 1/d$). In this way, additional samples are collected by the planner to maintain its covariance estimation accuracy (and hence its planning accuracy) despite the lower regularization.

Note that our algorithm achieves optimal sample complexity for the online data. Even with adaptive policies, the best known minimax upper bound is d^2/ϵ (Abbasi-Yadkori et al., 2011; Chu et al., 2011). Although Chu et al. (2011); Abbasi-Yadkori et al. (2011) consider regret minimization instead of sample complexity. But their results can be easily translated to sample complexity bounds (see also Jin et al. (2018, Section 3.1)). [5] However, the optimal rate of the number of offline data is unclear. We believe our results can be improved by new tools in matrix concentration, and we leave this direction for future work. Interestingly, in our experiments the algorithm performed well numerically even when $\lambda_{reg} < 1$ and $\alpha = 1$.

More generally, the amount of regularization should be set following the classical bias-variance tradeoff; its choice is mostly a statistical learning question, and its optimal value is normally problem dependent. As a result, the problem dependent tradeoff between different values of λ_{reg} is not reflected in our minimax analyses but can be appreciated in our numerical experiments on the real world dataset.

6 Proof Sketch

We give a brief proof sketch where we ignore constants and log factors; the full analysis can be found in the appendix. First we introduce some notation. We define the scaled cumulative covariance Σ_M

[4] The minimax lower bounds in the literature are stated for the regret setting, but can nonetheless be adapted to derive sample complexity results. See Appendix A for detailed discussion.

[5] Specifically, Abbasi-Yadkori et al. (2011) achieve a regret after N timesteps of at most $d\sqrt{N}$, meaning that $\sum_{t=1}^N (r(s_t, \pi^\star(s_t)) - r(s_t, \pi_t(s_t))) \leq O(d\sqrt{N})$. Note that π_t is independent of s_t, by the Azuma martingale concentration bound we get $\sum_{t=1}^N (\mathbb{E}_s r(s, \pi^\star(s)) - \mathbb{E}_s r(s, \pi_t(s))) \leq O(d\sqrt{N})$. Consider the average policy $\pi(s) = \pi_t(s)$ w.p. $1/N$. We get $\mathbb{E}_\pi \mathbb{E}_s[r(s,\pi(s))] \geq \mathbb{E}_s[r(s,\pi^\star(s))] - O(d/\sqrt{N})$. So when $N \geq \Omega(\frac{d^2}{\epsilon^2})$ the average policy played from timestep 1 to N is at most ϵ-suboptimal.

at step m for the planner and the cumulative covariance Σ'_n for the sampler at step n:

$$\Sigma_m = \alpha \sum_{j=1}^{m-1} \phi(s_j, a_j)\phi(s_j, a_j)^\top + \lambda_{reg} I, \qquad \Sigma'_n = \sum_{j=1}^{n-1} \phi(s'_j, a'_j)\phi(s'_j, a'_j)^\top + \lambda_{reg} I \tag{7}$$

where the planner's j actions is $a_j = \pi_j(s_j)$ and likewise the sampler's j actions is $a'_j \sim \pi_e(s'_j)$.

Uncertainty Stochastic Process We represent the value of the uncertainty though a stochastic process. Let us define the filtration for the planner $\mathcal{F}_m = \sigma(s_1, \dots, s_{m-1})$ at stage m and for the sampler $\mathcal{F}'_n = \sigma(s_1, \dots, s_M, s'_1, \dots, s'_{n-1})$ at stage n to represent the amount of information available. Let us also define the observed uncertainty U_m for the planner and likewise U'_n for the sampler:

$$U_m \stackrel{def}{=} \max_{a \in \mathcal{A}_{s_m}} \|\phi(s_m, a)\|_{(\Sigma_m)^{-1}}, \qquad U'_n \stackrel{def}{=} \max_{a \in \mathcal{A}_{s_n}} \|\phi(s'_n, a)\|_{(\Sigma'_n)^{-1}}. \tag{8}$$

We ultimately want to bound $\mathbb{E}'_n U'_n = \mathbb{E}[U'_n \mid \mathcal{F}_n] = \mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{(\Sigma'_n)^{-1}}$ when $n = N$, i.e., the average uncertainty at the end, see Section 3. Likewise, let us define $\mathbb{E}_m U_m = \mathbb{E}[U_m \mid \mathcal{F}_m] = \mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{(\Sigma_m)^{-1}}$.

We show that $\mathbb{E}'_N U'_N$ is minimized in two steps: 1) we show that the planner's uncertainty $\mathbb{E}_M U_M$ can be bounded 2) since the sampler implements the planner's average policy, $\mathbb{E}'_N U'_N$ cannot be too far from $\mathbb{E}_M U_M$.

We highlight that $M \geq N$ in general, i.e., the stochastic processes proceed at different speed. In fact, the planner needs more data than the sampler ($M \geq N$) as the planner's policy is reactive to the offline contexts; this forces us to use more data-intensive concentration inequalities for the planner.

6.1 Offline Uncertainty

The next lemma is formally presented in Lemma 4 and Lemma 6 in the appendix and examines the reduction in the planner's expected uncertainty $\mathbb{E}_M U_M$.

Lemma 1 (Offline Expected Uncertainty)**.** *With high probability we have*

$$\mathbb{E}_M U_M \leq \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}_m U_m \lessapprox \frac{1}{M} \sum_{m=1}^{M} U_m \lessapprox \sqrt{\frac{d}{\alpha M}}.$$

Proof sketch. We start by modifing a result from the classical bandit literature (e.g., Abbasi-Yadkori et al. (2011)) to bound the average realized uncertainty U_m while accounting for the extra α scaling factor contained in the covariance matrix Σ_M:

$$\frac{1}{M} \sum_{m=1}^{M} U_m = \frac{1}{M} \sum_{m=1}^{M} \|\phi(s_m, a_m)\|_{\Sigma_m^{-1}} \lessapprox \sqrt{\frac{d}{\alpha M}}.$$

The parameter α on the right hand side above arises from its inclusion in the definition of cumulative covariance Eq. 7. This justifies the last inequality in the lemma's statement.

While we bounded $\frac{1}{M}\sum_{m=1}^{M} U_m$, we need to bound the average conditional expectations $\frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_m U_m$. Bernstein's inequality would bound the actual deviation $\sum_{m=1}^{M} U_m$ by the conditional variances (or expectations $\sum_{m=1}^{M} \mathbb{E}_m U_m$ given our boundness assumptions); here we need the opposite, and so we 'reverse' the inequality in Theorem 3 (*Reverse Bernstein for Martingales*) to claim $\sum_{m=1}^{M} \mathbb{E}_m U_m \lessapprox \sum_{m=1}^{M} U_m$ with high probability to conclude. □

6.2 Online Uncertainty

In the prior section we showed that the planner would be successful in reducing its final uncertainty $\mathbb{E}_M U_M$ if it was collecting reward information; moreover, its sequence of policies $\pi_1, \dots, \pi_M$ only depends on the observed contexts. Because of this, if the sampler runs the planner's average policy π_e, we would expect a similar reduction in the uncertainty $\mathbb{E}'_N U'_N$ after $N = \alpha M$ iterations (the α factor simply arises because the planner's information are discounted by α). The argument is formalized in Lemma 3 (*Relations between Offline and Online Uncertainty*) in the appendix, which we preview here. We let K be the number of policy switches by the sampler which is bounded by $K \lessapprox d$ in Lemma 15 (*Number of Switches*) in appendix.

Lemma 2 (Relations between Offline and Online Uncertainty)**.** *If* $\lambda_{reg} = \Omega(\ln \frac{d}{\delta})$ *and* $M = \Omega\left(\frac{KN}{\lambda_{reg}} \ln \frac{dNK}{\lambda_{reg}\delta}\right)$*, upon termination of Alg. 1 and 2 it holds with high probability that*

$$\mathbb{E}'_N U'_N \lesssim \mathbb{E}_M U_M.$$

Proof. Let $d_1, \dots, d_M$ be the conditional distributions of the feature vectors sampled at timesteps $1, \dots, M$ in Alg. 1 after the algorithm has terminated. Conditioned on $\mathcal{F}_M$, the d_i's are non-random. Then the conditional expected covariance matrix given $\mathcal{F}_M$ can be defined as

$$\overline{\Sigma} = \alpha \sum_{i=1}^{M} \mathbb{E}_{\phi \sim d_i} \phi\phi^\top + \lambda_{reg} I \tag{9}$$

Now, our argument relies on some matrix concentration inequalities which hold if

$$\lambda_{reg} = \Omega(\ln \frac{d}{\delta}), \qquad \frac{1}{\alpha} = \Omega\left(\frac{K}{\lambda_{reg}} \ln \frac{dNK}{\lambda_{reg}\delta}\right). \tag{10}$$

Precisely, from Lemma 13 (*Matrix Upper Bound Offline Phase*) and Lemma 14 (*Matrix Upper Bound Online Phase*) with high probability we have that

$$\forall x, \|x\|_2 \le 1: \qquad \|x\|_{(\Sigma'_N)^{-1}} \lesssim \|x\|_{\overline{\Sigma}^{-1}} \qquad \text{and} \qquad \|x\|_{\overline{\Sigma}^{-1}} \lesssim \|x\|_{\Sigma_M^{-1}}. \tag{11}$$

In words, we can relate the planner's (random) scaled covariance Σ_M to its conditional expectation $\overline{\Sigma}$, with the guarantee that it won't be very different from the sampler's covariance Σ'_N (which implements the planner's policy). These concentration inequalities are key to the proof and are proved in the appendix.

Define the policy maximizing the online uncertainty $\pi'_n(s) = \arg\max_{a \in \mathcal{A}_s} \|\phi(s,a)\|_{(\Sigma'_n)^{-1}}$. Under the event in Eq. (11) we can write

$$\begin{aligned}\mathbb{E}'_N U'_N &\stackrel{def}{=} \mathbb{E}_{s\sim\mu} \max_{a\in\mathcal{A}_s} \|\phi(s,a)\|_{(\Sigma'_N)^{-1}} = \mathbb{E}_{s\sim\mu} \|\phi(s,\pi'_N(s))\|_{(\Sigma'_N)^{-1}} \lesssim \mathbb{E}_{s\sim\mu} \|\phi(s,\pi'_N(s))\|_{\overline{\Sigma}^{-1}} \\ &\lesssim \mathbb{E}_{s\sim\mu} \|\phi(s,\pi'_N(s))\|_{\Sigma_M^{-1}} \le \mathbb{E}_{s\sim\mu} \max_a \|\phi(s,a)\|_{\Sigma_M^{-1}} = \mathbb{E}_M U_M.\end{aligned}$$

□

6.3 Conclusion and Tradeoffs

We can now tune the parameters α, λ_{reg}. Starting from Eq. (3), we can write

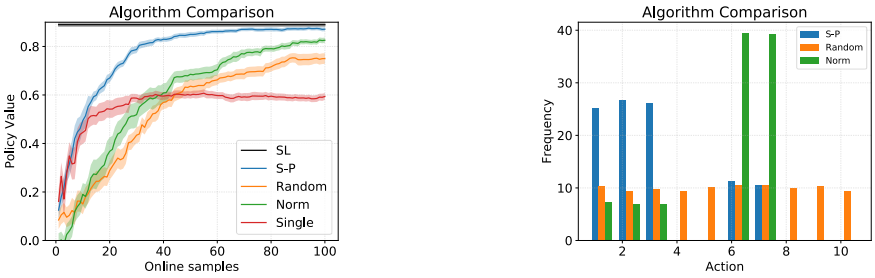
$$\mathbb{E}_{s\sim\mu} \max_a \phi(s,a)^\top(\theta^\star - \widehat{\theta}) \le \sqrt{\beta}\, \mathbb{E}'_N U'_N \lesssim \sqrt{\beta}\, \mathbb{E}_M U_M \lesssim \sqrt{\frac{\beta d}{\alpha M}} = \sqrt{\frac{\beta d}{N}}. \tag{12}$$

Thus, the rate-optimal online sample complexity $N \approx \frac{\beta d}{\epsilon^2}$ always suffices. However, we need a different number of offline contexts depending on the regularization that we want to adopt. Since $K \lessapprox d$, the reader can verify that if $\lambda_{reg} \approx d$ then the preconditions in Eq. (10) are verified with $M \approx N$, so $\alpha \approx 1$. If a lower value for the regularization $\lambda \approx 1$ is desired, $M \approx dN$ would be required (thus $\alpha \approx \frac{1}{d}$).

The reason why we might need $M \ge N$ (i.e., $\alpha \le 1$) is that the planner's policy is adaptive to the observed offline contexts. This means that the planner's cumulative covariance Σ_M is not a sum of i.i.d. rank-one random matrices; in this case, we need either a more data-hungry concentration inequality or heavier regularization, leading to the the precondtion on α in Eq. (10).

7 Experiments

In this section, we study the empirical properties of the planner and sampler in both a synthetic dataset and a real-world dataset from the Yahoo! Learning to Rank challenge (Chapelle and Chang, 2011). The objective of these experiments is to compare the performance of the planner-sampler pair with alternatives and analyze its sensitivity to regularization.



(a) (b)

Figure 2: Synthetic Setting. In the left figure, the policy value of the sampler-planner (S-P) is compared with a random algorithm (Random), a largest norm algorithm (Norm), and an algorithm that always chooses the same single action (Single). Supervised learning (SL) represents an approximate upper bound; however, we don't expect the algorithms to be able to reach the SL performance as they all receive bandit feedback compared to the full feedback available to SL (10,000 samples). In the right figure, the empirical distributions over the actions chosen by the algorithms are compared, with the exception of Single, which always chooses action 1.

7.1 Synthetic dataset

We first conduct a synthetic validation experiment in order to verify that the algorithms behave consistently with the intuition and theory developed in the previous sections. Towards this end, we constructed a simple linear contextual bandit problem with $d = 20$ and $\mathcal{A} = \{1, \ldots, 10\}$. At each time, features were generated randomly through the following procedure. An category I is sampled uniformly from the set $\{1, 2, 3\}$. For actions $a = 1, 2, 3$ and categories $i \in \{1, 2, 3\}$, we defined diagonal covariance matrices $\Sigma_{a,i} \in \mathbb{R}^{d \times d}$ such that $\Sigma_{a,i}$ has a single coordinate along the diagonal equal to 1 while all others are equal to 10^{-9}. The position of the coordinate equal to 1 is determined by the pair (a, i). Details about the exact assignments can be found in the appendix. Conditioned on the category I, features are distributed as $\phi(s, a) \sim \mathcal{N}(0, \Sigma_{a,I})$. This ensures variance only in certain directions depending on the category sampled and the action taken. Regardless of the category I sampled, for actions $a = 6, 7$, features are distributed as $\phi(s, a) \sim \mathcal{N}(0, \mathrm{diag}(10^{-9}, \ldots, 10^{-9}, 5)$ independently and thus there is always large variance in last coordinate regardless of the other actions. All other actions lead to features that are identically zero. The first $d - 1$ coordinates of θ were chosen randomly from $\{-1, 1\}^{d-1}$. The last coordinate is always zero.

We considered four algorithms: (1) a random algorithm that always chooses actions uniformly from $\mathcal{A}$, (2) a largest norm algorithm that always chooses the feature with the largest norm, (3) an algorithm that always chooses action 1 and (4) the sampler-planner algorithms proposed in this work. The planner is first run on an independent dataset of the same size as the one to be used for training in order to experience contexts from the same distribution that generates the online contexts. All algorithms are then run in the online phase with reward feedback. The algorithms are then evaluated based on their resulting policy values during the online phase on a held-out test set. Policy value is defined as $\mathbb{E}_s \phi(s, \hat{\pi}(s))^\top \theta^\star$ where $\hat{\pi}$ is the learned policy for a given algorithm. All algorithms used $\lambda = 1$ and the planner used $\alpha = 1$ as these worked well. The appendix contains more λ settings.

The synthetic dataset is designed in such a way that most actions are uninformative and that it is often most prudent to get information by selecting judiciously from actions $\{1, 2, 3, 6, 7\}$. Thus a random algorithm should perform poorly, spending time on irrelevant actions. The largest norm algorithm also fails: while much is learned about the last coordinate of θ by typically taking actions $\{6, 7\}$, little is learned about the others which are necessary for making good decisions. Finally, the single action algorithm fails to account for the categories. Figure 2 shows the policy values plotted over the number of samples taken and also the action distributions of the algorithms. Each line represents the mean of 20 trials and the error bars represent the standard error.

7.2 Learning to rank dataset

We evaluate the planner-sampler performance on real-world data from the Yahoo! Learning to Rank challenge (Chapelle and Chang, 2011), which has been used previously (Foster et al., 2018).

The ranking dataset is structured as follows. Each datapoint (row) is a 700-dimensional feature vector associated with a ranking relevance score that takes on values in $\{0, 1, 2, 3, 4\}$. Each datapoint is also associated with a single context, termed a query. Multiple datapoints matched with the same

query correspond to actions (documents) that the learner can choose when presented with that query. The dataset is already divided into training, validation, and testing data.

The objective is to choose actions, among those presented for a given query, that maximize the relevance scores. Unlike the previous synthetic dataset, the reward function here is misspecified, i.e., it does not exactly follow a linear model. To reduce the computational burden, we randomly subsampled coordinates of the original 700-dimensional features down to 300-dimensional features. We then normalized them to ensure the norms are at most 1. Similar to prior work, we also limit the number actions (documents) available at any given context to $K = 20$.

As before, we compare the planner-sampler pair to a random algorithm and a supervised learning oracle that observes all the context-action features and relevance scores, which is 200,000 training samples in total. To simulate streaming the data online to the algorithms, we sequentially iterate through the contexts and present the associated features to the learner as actions. During the offline phase, we run the planner on the validation set where none of the relevance scores are observed in order to generate the sampling policies. During the online phase, we run both the sampler and the random algorithm on the training data such that they observe the same contexts but may take different actions. The maximum number of distinct contexts we can iterate over is about 3000. The resulting policies are evaluated on the full test set after every 20 training observations. This experiment was repeated with varying values of the regularization parameter $\lambda \in \{0.1, 1, 10\}$. The planner uses $\alpha = 1$; we did not find significant improvements with $\alpha < 1$. Additional values for λ are reported in the supplementary material.

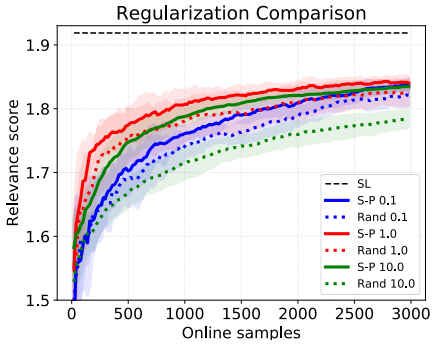


Figure 3: The sampler-planner (S-P) pair is compared with a random sampler (Rand) with varying levels of regularization λ. The supervised learning oracle (SL), which observes state-action pairs in the training set, is shown as baseline.

Figure 3 depicts the results. Each point on the graph represents the mean of 10 independent trials and the error bands represent the standard deviation. The randomness is due only to internal randomness in the algorithms and the randomized order in which the contexts are revealed. The sampler-planner consistently outperforms the random algorithm in most regimes, including $\lambda = 1$ where the pinnacle performance of both algorithms is achieved. The gap between the algorithms is widened for larger choices of λ. We notice that for $\lambda = 0.1$ the curves appear to catch up as more data is gathered; it is possible that they would outperform larger value of regularization if more samples were available. The optimal choice for λ depends on the bias-variance tradeoff for such problem. Note that despite misspecification, our sampler-planner approach consistently matches or significantly exceeds the performance of the baseline approach, particularly when the number of samples is small.

8 Conclusion

Assuming access to a set of offline contexts, this work proposes an algorithm to find a single stochastic policy to acquire data strategically. The procedure is sound as validated by a theoretical analysis and by experiments even when the reward model is not linear.

Our work opens several interesting directions: (1) can we devise a procedure that works across all values of regularization without severely impacting the amount of offline data we need? (2) how does the performance of the algorithm degrade if the online context distribution differs from the offline one? (3) can we incorporate prior reward data to reduce the sample complexity, and how is the benefit of such data impacted by the size of the state-action space, and the decision policy used to gather the prior reward data, which might be an adversarial procedure? (4) what is the price to pay for being non-reactive to data, i.e., what is the trade-off between an adaptive and a non-adaptive procedure like the one that we propose here?

Our work presents little direct negative potential societal impacts though whether contextualized decision policies are beneficial or harmful depends on the setting and the optimization (reward function) chosen by a system designer.

Acknowledgments

JL is supported by the NSF GRFP.

References

Abbasi-Yadkori, Y., Pal, D., and Szepesvari, C. (2011). Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems (NIPS)*.

Bai, Y., Xie, T., Jiang, N., and Wang, Y. X. (2019). Provably efficient q-learning with low switching cost. *Advances in Neural Information Processing Systems*, 32.

Beygelzimer, A., Langford, J., Li, L., Reyzin, L., and Schapire, R. (2011). Contextual bandit algorithms with supervised learning guarantees. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 19–26. JMLR Workshop and Conference Proceedings.

Chapelle, O. and Chang, Y. (2011). Yahoo! learning to rank challenge overview. In *Proceedings of the learning to rank challenge*, pages 1–24. PMLR.

Chu, W., Li, L., Reyzin, L., and Schapire, R. (2011). Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214. JMLR Workshop and Conference Proceedings.

Cortes, K. E., Fricke, H. D., Loeb, S., Song, D. S., and York, B. N. (2019). When behavioral barriers are too high or low–how timing matters for parenting interventions. Technical report, National Bureau of Economic Research.

Degenne, R., Ménard, P., Shang, X., and Valko, M. (2020). Gamification of pure exploration for linear bandits. In *International Conference on Machine Learning*, pages 2432–2442. PMLR.

Deshmukh, A. A., Sharma, S., Cutler, J. W., Moldwin, M., and Scott, C. (2018). Simple regret minimization for contextual bandits. *arXiv preprint arXiv:1810.07371*.

Doss, C., Fahle, E. M., Loeb, S., and York, B. N. (2019). More than just a nudge supporting kindergarten parents with differentiated and personalized text messages. *Journal of Human Resources*, 54(3):567–603.

Esfandiari, H., Karbasi, A., Mehrabian, A., and Mirrokni, V. (2019). Regret bounds for batched bandits. *arXiv preprint arXiv:1910.04959*.

Fiez, T., Jain, L., Jamieson, K., and Ratliff, L. (2019). Sequential experimental design for transductive linear bandits. *arXiv preprint arXiv:1906.08399*.

Foster, D., Agarwal, A., Dudik, M., Luo, H., and Schapire, R. (2018). Practical contextual bandits with regression oracles. In *International Conference on Machine Learning*, pages 1539–1548. PMLR.

Han, Y., Zhou, Z., Zhou, Z., Blanchet, J., Glynn, P. W., and Ye, Y. (2020). Sequential batch learning in finite-action linear contextual bandits. *arXiv preprint arXiv:2004.06321*.

Jedra, Y. and Proutiere, A. (2020). Optimal best-arm identification in linear bandits. *Advances in Neural Information Processing Systems*, 33.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. (2018). Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4863–4873.

Kiefer, J. and Wolfowitz, J. (1960). The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12:363–366.

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

Lattimore, T. and Szepesvari, C. (2020). Learning with good feature representations in bandits and in rl with a generative model. In *International Conference on Machine Learning (ICML)*.

Mannor, S. and Tsitsiklis, J. N. (2004). The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5(Jun):623–648.

Ren, Z., Zhou, Z., and Kalagnanam, J. R. (2020). Batched learning in generalized linear contextual bandits with general decision sets. *IEEE Control Systems Letters*.

Ruan, Y., Yang, J., and Zhou, Y. (2020). Linear bandits with limited adaptivity and learning distributional optimal design. *arXiv preprint arXiv:2007.01980*.

Soare, M., Lazaric, A., and Munos, R. (2014). Best-arm identification in linear bandits. In *Advances in Neural Information Processing Systems (NIPS)*, pages 828–836.

Tao, C., Blanco, S., and Zhou, Y. (2018). Best arm identification in linear bandits with linear dimension dependency. In *International Conference on Machine Learning*, pages 4877–4886. PMLR.

Tropp, J. A. (2012). User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434.

Wang, T., Zhou, D., and Gu, Q. (2021). Provably efficient reinforcement learning with linear function approximation under adaptivity constraints. *arXiv preprint arXiv:2101.02195*.

Xu, L., Honda, J., and Sugiyama, M. (2018). A fully adaptive algorithm for pure exploration in linear bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 843–851. PMLR.

Zanette, A., Cheng, C.-A., and Agarwal, A. (2021). Cautiously optimistic policy optimization and exploration with linear function approximation. *arXiv preprint arXiv:2103.12923*.