NEAR-OPTIMAL DEPLOYMENT EFFICIENCY IN REWARD-FREE REINFORCEMENT LEARNING WITH LINEAR FUNCTION APPROXIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the problem of deployment efficient reinforcement learning (RL) with linear function approximation under the *reward-free* exploration setting. This is a well-motivated problem because deploying new policies is costly in real-life RL applications. Under the linear MDP setting with feature dimension d and planning horizon H, we propose a new algorithm that collects at most $\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$ trajectories within H deployments to identify ϵ-optimal policy for any (possibly data-dependent) choice of reward functions. To the best of our knowledge, our approach is the first to achieve optimal deployment complexity and optimal d dependence in sample complexity at the same time, even if the reward is known ahead of time. Our novel techniques include an exploration-preserving policy discretization and a generalized G-optimal experiment design, which could be of independent interest. Lastly, we analyze the related problem of regret minimization in low-adaptive RL and provide information-theoretic lower bounds for switching cost and batch complexity.

1 INTRODUCTION

In many practical reinforcement learning (RL) based tasks, limited computing resources hinder applications of fully adaptive algorithms that frequently deploy new exploration policy. Instead, it is usually cheaper to collect data in large batches using the current policy deployment. Take recommendation system (Afsar et al., 2021) as an instance, the system is able to gather plentiful new data in very short time, while the deployment of a new policy often takes longer time, as it requires extensive computing and human resources. Therefore, it is impractical to switch the policy based on instantaneous data as a typical RL algorithm would demand. A feasible alternative is to run a large batch of experiments in parallel and only decide whether to update the policy after the whole batch is complete. The same constraint also appears in other RL applications such as healthcare (Yu et al., 2021), robotics (Kober et al., 2013) and new material design (Zhou et al., 2019). In those scenarios, the agent needs to minimize the number of policy deployment while learning a good policy using (nearly) the same number of trajectories as its fully-adaptive counterparts. On the empirical side, Matsushima et al. (2020) first proposed the notion *deployment efficiency*. Later, Huang et al. (2022) formally defined *deployment complexity*. Briefly speaking, deployment complexity measures the number of policy deployments while requiring each deployment to have similar size. We measure the adaptivity of our algorithms via deployment complexity and leave its formal definition to Section 2.

Under the purpose of deployment efficiency, the recent work by Qiao et al. (2022) designed an algorithm that could solve reward-free exploration in $O(H)$ deployments. However, their sample complexity $\widetilde{O}(|\mathcal{S}|^2|\mathcal{A}|H^5/\epsilon^2)$, although being near-optimal under the tabular setting, can be unacceptably large under real-life applications where the state space is enormous or continuous. For environments with large state space, function approximations are necessary for representing the feature of each state. Among existing work that studies function approximation in RL, linear function approximation is arguably the simplest yet most fundamental setting. In this paper, we study deployment efficient RL with linear function approximation under the reward-free setting, and we consider the following question:

Question 1.1. *Is it possible to design deployment efficient and sample efficient reward-free RL algorithms with linear function approximation?*

Algorithms for reward-free RL	*Sample complexity*	*Deployment complexity*
Algorithm 1 & 2 in Wang et al. (2020)	$\widetilde{O}(\frac{d^3H^6}{\epsilon^2})$	$\widetilde{O}(\frac{d^3H^6}{\epsilon^2})$
FRANCIS (Zanette et al., 2020b)‡	$\widetilde{O}(\frac{d^3H^5}{\epsilon^2})$	$\widetilde{O}(\frac{d^3H^5}{\epsilon^2})$
RFLIN (Wagenmaker et al., 2022b)‡	$\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$	$\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$
Algorithm 2 & 4 in Huang et al. (2022)‡	$\widetilde{O}(\frac{d^3H^5}{\epsilon^2\nu^2_{\min}})^*$	H
LARFE (Qiao et al., 2022)†	$\widetilde{O}(\frac{S^2AH^5}{\epsilon^2})$	$2H$
Our Algorithm 1 & 2 (Theorem 5.1)‡	$\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$	H
Our Algorithm 1 & 2 (Theorem 7.1)*	$\widetilde{O}(\frac{S^2AH^5}{\epsilon^2})$	H
Lower bound (Wagenmaker et al., 2022b)	$\Omega(\frac{d^2H^2}{\epsilon^2})$	N.A.
Lower bound (Huang et al., 2022)	If polynomial sample	$\widetilde{\Omega}(H)$

Table 1: Comparison of our results (in blue) to existing work regarding sample complexity and deployment complexity. We highlight that our results match the best known results for both sample complexity and deployment complexity at the same time. ‡: We ignore the lower order terms in sample complexity for simplicity. $*$: ν_{min} is the problem-dependent reachability coefficient which is upper bounded by 1 and can be arbitrarily small. †: This work is done under tabular MDP and we transfer the $O(HSA)$ switching cost to $2H$ deployments. $\star$: When both our algorithms are applied under tabular MDP, we can replace one d in sample complexity by S.

Our contributions. In this paper, we answer the above question affirmatively by constructing an algorithm with near-optimal deployment and sample complexities. Our contributions are threefold.

- A new layer-by-layer type algorithm (Algorithm 1) for reward-free RL that achieves deployment complexity of H and sample complexity of $\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$. Our deployment complexity is optimal while sample complexity has optimal dependence in d and ϵ. In addition, when applied to tabular MDP, our sample complexity (Theorem 7.1) recovers best known result $\widetilde{O}(\frac{S^2AH^5}{\epsilon^2})$.
- We generalize G-optimal design and select near-optimal policy via uniform policy evaluation on a finite set of *representative policies* instead of using optimism and LSVI. Such technique helps tighten our sample complexity and may be of independent interest.
- We show that "No optimal-regret online learners can be deployment efficient" and deployment efficiency is incompatible with the highly relevant regret minimization setting. For regret minimization under linear MDP, we present lower bounds (Theorem 7.2 and 7.3) for other measurements of adaptivity: switching cost and batch complexity.

1.1 Closely related works

There is a large and growing body of literature on the statistical theory of reinforcement learning that we will not attempt to thoroughly review. Detailed comparisons with existing work on reward-free RL (Wang et al., 2020; Zanette et al., 2020b; Wagenmaker et al., 2022b; Huang et al., 2022; Qiao et al., 2022) are given in Table 1. For more discussion of relevant literature, please refer to Appendix A and the references therein. Notably, all existing algorithms under linear MDP either admit fully adaptive structure (which leads to deployment inefficiency) or suffer from sub-optimal sample complexity. In addition, when applied to tabular MDP, our algorithm has the same sample complexity and slightly better deployment complexity compared to Qiao et al. (2022).

The deployment efficient setting is slightly different from other measurements of adaptivity. The low switching setting (Bai et al., 2019) restricts the number of policy updates, while the agent can decide whether to update the policy after collecting every single trajectory. This can be difficult to implement in practical applications. A more relevant setting, the batched RL setting (Perchet et al., 2016) requires decisions about policy changes to be made at only a few (often predefined) checkpoints. Compared to batched RL, the requirement of deployment efficiency is stronger by requiring each deployment to collect the same number of trajectories. Therefore, deployment efficient algorithms are easier to deploy in parallel (see, e.g., Huang et al., 2022, for a more elaborate discussion). Lastly, we remark that our algorithms also work under the batched RL setting by running in H batches.

Technically, our method is inspired by optimal experiment design – a well-developed research area from statistics. In particular, a major technical contribution of this paper is to solve a variant of G-optimal experiment design while solving exploration in RL at the same time. Zanette et al. (2020b); Wagenmaker et al. (2022b) choose policy through *online experiment design*, i.e., running no-regret online learners to select policies adaptively for approximating the optimal design. Those online approaches, however, cannot be applied under our problem due to the requirement of deployment efficiency. To achieve deployment complexity of H, we can only deploy one policy for each layer, so we need to decide the policy based on sufficient exploration for only previous layers. Therefore, our approach requires *offline experiment design* and thus raises substantial technical challenge.

A remark on technical novelty. The general idea behind previous RL algorithms with low adaptivity is optimism and doubling schedule for updating policies that originates from UCB2 (Auer et al., 2002). The doubling schedule, however, can not provide optimal deployment complexity. Different from those approaches, we apply layer-by-layer exploration to achieve the optimal deployment complexity, and our approach is highly non-trivial. Since we can only deploy one policy for each layer, there are two problems to be solved: the existence of a single policy that can explore all directions of a specific layer and how to find such policy. We generalize G-optimal design to show the existence of such *explorative policy*. Besides, we apply exploration-preserving *policy discretization* for approximating our generalized G-optimal design. We leave detailed discussions about these techniques to Section 3.

2 Problem Setup

Notations. Throughout the paper, for $n \in \mathbb{Z}^+$, $[n] = \{1, 2, \cdots, n\}$. We denote $\|x\|_\Lambda = \sqrt{x^\top \Lambda x}$. For matrix $X \in \mathbb{R}^{d\times d}$, $\|\cdot\|_2$, $\|\cdot\|_F$, $\lambda_{\min}(\cdot)$, $\lambda_{\max}(\cdot)$ denote the operator norm, Frobenius norm, smallest eigenvalue and largest eigenvalue, respectively. For policy π, $\mathbb{E}_\pi$ and $\mathbb{P}_\pi$ denote the expectation and probability measure induced by π under the MDP we consider. For any set U, $\Delta(U)$ denotes the set of all possible distributions over U. In addition, we use standard notations such as O and Ω to absorb constants while $\widetilde{O}$ and $\widetilde{\Omega}$ suppress logarithmic factors.

Markov Decision Processes. We consider finite-horizon episodic *Markov Decision Processes* (MDP) with non-stationary transitions, denoted by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, P_h, r_h)$ (Sutton & Barto, 1998), where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space and H is the horizon. The non-stationary transition kernel has the form $P_h : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \mapsto [0, 1]$ with $P_h(s'|s, a)$ representing the probability of transition from state s, action a to next state s' at time step h. In addition, $r_h(s, a) \in \Delta([0, 1])$ denotes the corresponding distribution of reward.[1] Without loss of generality, we assume there is a fixed initial state s_1.[2] A policy can be seen as a series of mapping $\pi = (\pi_1, \cdots, \pi_H)$, where each π_h maps each state $s \in \mathcal{S}$ to a probability distribution over actions, *i.e.* $\pi_h : \mathcal{S} \to \Delta(\mathcal{A})$, $\forall\, h \in [H]$. A random trajectory $(s_1, a_1, r_1, \cdots, s_H, a_H, r_H, s_{H+1})$ is generated by the following rule: s_1 is fixed, $a_h \sim \pi_h(\cdot|s_h), r_h \sim r_h(s_h, a_h), s_{h+1} \sim P_h(\cdot|s_h, a_h), \forall\, h \in [H]$.

Q-values, Bellman (optimality) equations. Given a policy π and any $h \in [H]$, the value function $V_h^\pi(\cdot)$ and Q-value function $Q_h^\pi(\cdot,\cdot)$ are defined as: $V_h^\pi(s) = \mathbb{E}_\pi[\sum_{t=h}^H r_t|s_h = s], Q_h^\pi(s, a) = \mathbb{E}_\pi[\sum_{t=h}^H r_t|s_h, a_h = s, a]$, $\forall\, s, a \in \mathcal{S}\times\mathcal{A}$. Besides, the value function and Q-value function with respect to the optimal policy $\pi^\star$ is denoted by $V_h^\star(\cdot)$ and $Q_h^\star(\cdot,\cdot)$. Then Bellman (optimality) equation follows $\forall\, h \in [H]$:

$$Q_h^\pi(s, a) = r_h(s, a) + P_h(\cdot|s, a)V_{h+1}^\pi, \quad V_h^\pi = \mathbb{E}_{a\sim\pi_h}[Q_h^\pi],$$
$$Q_h^\star(s, a) = r_h(s, a) + P_h(\cdot|s, a)V_{h+1}^\star, \; V_h^\star = \max_a Q_h^\star(\cdot, a).$$

In this work, we consider the reward-free RL setting, where there may be different reward functions. Therefore, we denote the value function of policy π with respect to reward r by $V^\pi(r)$. Similarly, $V^\star(r)$ denotes the optimal value under reward function r. We say that a policy π is ϵ-optimal with respect to r if $V^\pi(r) \geq V^\star(r) - \epsilon$.

Linear MDP (Jin et al., 2020b). An episodic MDP $(\mathcal{S}, \mathcal{A}, H, P, r)$ is a linear MDP with known feature map $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$ if there exist H unknown signed measures $\mu_h \in \mathbb{R}^d$ over $\mathcal{S}$ and H unknown reward vectors $\theta_h \in \mathbb{R}^d$ such that

$$P_h(s' \mid s, a) = \langle \phi(s, a), \mu_h(s')\rangle, \quad r_h(s, a) = \langle \phi(s, a), \theta_h\rangle, \quad \forall\,(h, s, a, s') \in [H]\times\mathcal{S}\times\mathcal{A}\times\mathcal{S}.$$

[1] We abuse the notation r so that r also denotes the expected (immediate) reward function.

[2] The generalized case where the initial distribution is an arbitrary distribution can be recovered from this setting by adding one layer to the MDP.

Without loss of generality, we assume $\|\phi(s,a)\|_2 \leq 1$ for all s, a; and for all $h \in [H]$, $\|\mu_h(\mathcal{S})\|_2 \leq \sqrt{d}$, $\|\theta_h\|_2 \leq \sqrt{d}$.

For policy π, we define $\Lambda_{\pi,h} := \mathbb{E}_\pi[\phi(s_h,a_h)\phi(s_h,a_h)^\top]$, the *expected covariance matrix* with respect to policy π and time step h (here s_h, a_h follows the distribution induced by policy π). Let $\lambda^\star = \min_{h\in[H]} \sup_\pi \lambda_{\min}(\Lambda_{\pi,h})$. We make the following assumption regarding explorability.

Assumption 2.1 (Explorability of all directions)**.** *The linear MDP we have satisfies $\lambda^\star > 0$.*

We remark that Assumption 2.1 only requires the existence of a (possibly non-Markovian) policy to visit all directions for each layer and it is analogous to other explorability assumptions in papers about RL under linear representation (Zanette et al., 2020b; Huang et al., 2022; Wagenmaker & Jamieson, 2022). In addition, the parameter $\lambda^\star$ only appears in lower order terms of sample complexity bound and our algorithms do not take $\lambda^\star$ as an input.

Reward-Free RL. The reward-free RL setting contains two phases, the exploration phase and the planning phase. Different from PAC RL[3] setting, the learner does not observe the rewards during the exploration phase. Besides, during the planning phase, the learner has to output a near-optimal policy for any valid reward functions. More specifically, the procedure is:

1. Exploration phase: Given accuracy ϵ and failure probability δ, the learner explores an MDP for $K(\epsilon,\delta)$ episodes and collects the trajectories without rewards $\{s_h^k, a_h^k\}_{(h,k)\in[H]\times[K]}$.
2. Planning phase: The learner outputs a function $\widehat{\pi}(\cdot)$ which takes reward function as input. The function $\widehat{\pi}(\cdot)$ satisfies that for any valid reward function r, $V^{\widehat{\pi}(r)}(r) \geq V^\star(r) - \epsilon$.

The goal of reward-free RL is to design a procedure that satisfies the above guarantee with probability at least $1-\delta$ while collecting as few episodes as possible. According to the definition, any procedure satisfying the above guarantee is provably efficient for PAC RL setting.

Deployment Complexity. In this work, we measure the adaptivity of our algorithm through deployment complexity, which is defined as:

Definition 2.2 (Deployment complexity (Huang et al., 2022))**.** *We say that an algorithm has deployment complexity of M, if the algorithm is guaranteed to finish running within M deployments. In addition, the algorithm is only allowed to collect at most N trajectories during each deployment, where N should be fixed a priori and cannot change adaptively.*

We consider the deployment of non-Markovian policies (i.e. mixture of deterministic policies) (Huang et al., 2022). The requirement of deployment efficiency is stronger than batched RL (Perchet et al., 2016) or low switching RL (Bai et al., 2019), which makes deployment-efficient algorithms more practical in real-life applications. For detailed comparison between these definitions, please refer to Section 1.1 and Appendix A.

3 Technique Overview

In order to achieve the optimal deployment complexity of H, we apply layer-by-layer exploration. More specifically, we construct a single policy π_h to explore layer h based on previous data. Following the general methods in reward-free RL (Wang et al., 2020; Wagenmaker et al., 2022b), we do exploration through minimizing uncertainty. As will be made clear in the analysis, given exploration dataset $\mathcal{D} = \{s_h^n, a_h^n\}_{h,n\in[H]\times[N]}$, the uncertainty of layer h with respect to policy π can be characterized by $\mathbb{E}_\pi\|\phi(s_h,a_h)\|_{\Lambda_h^{-1}}$, where $\Lambda_h = I + \sum_{n=1}^N \phi(s_h^n,a_h^n)\phi(s_h^n,a_h^n)^\top$ is (regularized and unnormalized) *empirical covariance matrix*. Note that although we can not directly optimize Λ_h, we can maximize the expectation $N_{\pi_h}\cdot\mathbb{E}_{\pi_h}[\phi_h\phi_h^\top]$ (N_{π_h} is the number of trajectories we apply π_h) by optimizing the policy π_h. Therefore, to minimize the uncertainty with respect to some policy set Π, we search for an *explorative policy* π_0 to minimize $\max_{\pi\in\Pi}\mathbb{E}_\pi\phi(s_h,a_h)(\mathbb{E}_{\pi_0}\phi_h\phi_h^\top)^{-1}\phi(s_h,a_h)$.

3.1 Generalized G-optimal design

For the minimization problem above, traditional G-optimal design handles the case where each deterministic policy π generates some ϕ_π at layer h with probability 1 (i.e. we directly choose ϕ instead of choosing π), as is the case under deterministic MDP. However, traditional G-optimal design

[3] Also known as reward-aware RL, which aims to identify near optimal policy given reward function.

cannot tackle our problem since under general linear MDP, each π will generate a distribution over the feature space instead of a single feature vector. We generalize G-optimal design and show that for any policy set Π, the following Theorem 3.1 holds. More details are deferred to Appendix B.

Theorem 3.1 (Informal version of Theorem B.1)**.** *If there exists policy* $\pi_0 \in \Delta(\Pi)$ *such that* $\lambda_{\min}(\mathbb{E}_{\pi_0}\phi_h\phi_h^\top) > 0$*, then* $\min_{\pi_0\in\Delta(\Pi)} \max_{\pi\in\Pi} \mathbb{E}_\pi \phi(s_h, a_h)(\mathbb{E}_{\pi_0}\phi_h\phi_h^\top)^{-1}\phi(s_h, a_h) \leq d$*.*

Generally speaking, Theorem 3.1 states that for any Π, there exists a single policy from $\Delta(\Pi)$ (i.e., mixture of several policies in Π) that can efficiently reduce the uncertainty with respect to Π. Therefore, assume we want to minimize the uncertainty with respect to Π and we are able to derive the solution π_0 of the minimization above, we can simply run π_0 repeatedly for several episodes.

However, there are two gaps between Theorem 3.1 and our goal of reward free RL. First, under the Reinforcement Learning setting, the association between policy π and the corresponding distribution of ϕ_h is unknown, which means we need to approximate the above minimization. It can be done by estimating the two expectations and we leave the discussion to Section 3.3. The second gap is about choosing appropriate Π in Theorem 3.1, for which a natural idea is to use the set of all policies. It is however infeasible to simultaneously estimate the expectations for all π accurately. The size of $\{\text{all policies}\}$ is infinity and $\Delta(\{\text{all policies}\})$ is even bigger. It seems intractable to control its complexity using existing uniform convergence techniques (e.g., a covering number argument).

3.2 Discretization of policy set

The key realization towards a solution to the above problem is that we do not need to consider the set of all policies. It suffices to consider a smaller subset Π that is more amenable to an ϵ-net argument. This set needs to satisfy a few conditions.

(1) Due to condition in Theorem 3.1, Π should contain explorative policies covering all directions.

(2) Π should contain a *representative policy set* Π^{eval} such that it contains a near-optimal policy for any reward function.

(3) Since we apply *offline experimental design* via approximating the expectations, Π must be "small" enough for a uniform-convergence argument to work.

We show that we can construct a finite set Π with $|\Pi|$ being small enough while satisfying Condition (1) and (2). More specifically, given the feature map $\phi(\cdot,\cdot)$ and the desired accuracy ϵ, we can construct the *explorative policy set* Π_ϵ^{exp} such that $\log(|\Pi_{\epsilon,h}^{exp}|) \leq \widetilde{O}(d^2\log(1/\epsilon))$, where $\Pi_{\epsilon,h}^{exp}$ is the policy set for layer h. In addition, when ϵ is small compared to $\lambda^\star$, we have $\sup_{\pi\in\Delta(\Pi_\epsilon^{exp})} \lambda_{\min}(\mathbb{E}_\pi\phi_h\phi_h^\top) \geq \widetilde{\Omega}(\frac{(\lambda^\star)^2}{d})$, which verifies Condition (1).[4] Plugging in Π_ϵ^{exp} and approximating the minimization problem, after the exploration phase we will be able to estimate the value functions of all $\pi \in \Pi_\epsilon^{exp}$ accurately.

It remains to check Condition (2) by formalizing the *representative policy set* discussed above. From Π_ϵ^{exp}, we can further select a subset, and we call it *policies to evaluate*: Π_ϵ^{eval}. It satisfies that $\log(|\Pi_{\epsilon,h}^{eval}|) = \widetilde{O}(d\log(1/\epsilon))$ while for any possible linear MDP with feature map $\phi(\cdot,\cdot)$, Π_ϵ^{eval} is guaranteed to contain one ϵ-optimal policy. As a result, it suffices to estimate the value functions of all policies in Π_ϵ^{eval} and output the greedy one with the largest estimated value.[5]

3.3 New approach to estimate value function

Now that we have a discrete policy set, we still need to estimate the two expectations in Theorem 3.1. We design a new algorithm (Algorithm 4, details can be found in Appendix E) based on the technique of LSVI (Jin et al., 2020b) to estimate $\mathbb{E}_\pi r(s_h, a_h)$ given policy π, reward r and exploration data. Algorithm 4 can estimate the expectations accurately simultaneously for all $\pi \in \Pi^{exp}$ and r (that appears in the minimization problem) given sufficient exploration of the first $h-1$ layers. Therefore, under our layer-by-layer exploration approach, after adequate exploration for the first $h-1$ layers, Algorithm 4 provides accurate estimations for $\mathbb{E}_{\pi_0}\phi_h\phi_h^\top$ and $\mathbb{E}_\pi[\phi(s_h,a_h)^\top(\widehat{\mathbb{E}}_{\pi_0}\phi_h\phi_h^\top)^{-1}\phi(s_h,a_h)]$. As a result, the (1) we solve serves as an accurate approximation of the minimization problem in Theorem 3.1 and the solution π_h of (1) is provably efficient in exploration.

[4] For more details about explorative policies, please refer to Appendix C.3.

[5] For more details about policies to evaluate, please refer to Appendix C.2.

Finally, after sufficient exploration of all H layers, the last step is to estimate the value functions of all policies in Π^{eval}. We design a slightly different algorithm (Algorithm 3, details in Appendix D) for this purpose. Based on LSVI, Algorithm 3 takes $\pi \in \Pi^{eval}$ and reward function r as input, and estimates $V^{\pi}(r)$ accurately given sufficient exploration for all H layers.

4 ALGORITHMS

In this section, we present our main algorithms. The algorithm for the exploration phase is Algorithm 1 which formalizes the ideas in Section 3, while the planning phase is presented in Algorithm 2.

Algorithm 1 Layer-by-layer Reward-Free Exploration via Experimental Design (Exploration)

Input: Accuracy ϵ. Failure probability δ.
Initialization: $\iota = \log(dH/\epsilon\delta)$. Error budget for each layer $\bar{\epsilon} = \frac{C_1\epsilon}{H^2\sqrt{d}\cdot\iota}$. Construct $\Pi^{exp}_{\epsilon/3}$ as in Section 3.2. Number of episodes for each deployment $N = \frac{C_2 d\iota}{\bar{\epsilon}^2} = \frac{C_2 d^2 H^4 \iota^3}{C_1^2\epsilon^2}$. Dataset $\mathcal{D} = \emptyset$.
for $h = 1, 2, \cdots, H$ **do**
  Solve the following optimization problem.

$$\pi_h = \operatorname*{argmin}_{\pi \in \Delta(\Pi^{exp}_{\epsilon/3}) \text{ s.t. } \lambda_{\min}(\widehat{\Sigma}_\pi) \geq C_3 d^2 H \bar{\epsilon}\iota} \max_{\widehat{\pi} \in \Pi^{exp}_{\epsilon/3}} \widehat{\mathbb{E}}_{\widehat{\pi}} \left[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_\pi)^{-1} \phi(s_h, a_h) \right], \quad (1)$$

  where $\widehat{\Sigma}_\pi$ is $\widehat{\mathbb{E}}_\pi[\phi(s_h, a_h)\phi(s_h, a_h)^\top] = \mathsf{EstimateER}(\pi, \phi(s,a)\phi(s,a)^\top, A = 1, h, \mathcal{D}, s_1)$,
  $\widehat{\mathbb{E}}_{\widehat{\pi}} \left[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_\pi)^{-1} \phi(s_h, a_h) \right] = \mathsf{EstimateER}(\widehat{\pi}, \phi(s,a)^\top (N \cdot \widehat{\Sigma}_\pi)^{-1} \phi(s,a), A = \frac{\bar{\epsilon}}{C_2 d^3 H \iota^2}, h, \mathcal{D}, s_1)$. // Both expectations are estimated via Algorithm 4.
  for $n = 1, 2, \cdots, N$ **do**
    Run π_h and add trajectory $\{s_i^n, a_i^n\}_{i \in [H]}$ to $\mathcal{D}$. // Run Policy π_h for N episodes.
  end for
end for
Output: Dataset $\mathcal{D}$.

Exploration Phase. We apply layer-by-layer exploration and π_h is the stochastic policy we deploy to explore layer h. For solving π_h, we approximate generalized G-optimal design via (1). For each candidate π and $\widehat{\pi}$, we estimate the two expectations by calling $\mathsf{EstimateER}$ (Algorithm 4, details in Appendix E). $\mathsf{EstimateER}$ is a generic subroutine for estimating the value function under a particular reward design. We estimate the two expectations of interest by carefully choosing one specific reward design for each coordinate separately, so that the resulting value function provides an estimate to the desired quantity in that coordinate. [6] As mentioned above and will be made clear in the analysis, given adequate exploration of the first $h-1$ layers, all estimations will be accurate and the surrogate policy π_h is sufficiently explorative for all directions at layer h.

The restriction on $\lambda_{\min}(\widehat{\Sigma}_\pi)$ is for technical reason only, and we will show that under the assumption in Theorem 5.1, there exists valid solution of (1). Lastly, we remark that solving (1) is inefficient in general. Detailed discussions about computation are deferred to Section 7.2.

Algorithm 2 Find Near-Optimal Policy Given Reward Function (Planning)

Input: Dataset $\mathcal{D}$ from Algorithm 1. Feasible linear reward function $r = \{r_h\}_{h \in [H]}$.
Initialization: Construct $\Pi^{eval}_{\epsilon/3}$ as in Section 3.2. // The set of policies to evaluate.
for $\pi \in \Pi^{eval}_{\epsilon/3}$ **do**
  $\widehat{V}^{\pi}(r) = \text{EstimateV}(\pi, r, \mathcal{D}, s_1)$. // Estimate value functions using Algorithm 3.
end for
$\widehat{\pi} = \arg\max_{\pi \in \Pi^{eval}_{\epsilon/3}} \widehat{V}^{\pi}(r)$. // Output the greedy policy w.r.t $\widehat{V}^{\pi}(r)$.
Output: Policy $\widehat{\pi}$.

[6] For $\widehat{\Sigma}_\pi$, what we need to handle is matrix reward $\phi_h \phi_h^\top$ and stochastic policy $\pi \in \Delta(\Pi^{exp}_{\epsilon/3})$, we apply generalized version of Algorithm 4 to tackle this problem as discussed in Appendix F.1.

Planning Phase. The output dataset $\mathcal{D}$ from the exploration phase contains sufficient information for the planning phase. In the planning phase (Algorithm 2), we construct a set of policies to evaluate and repeatedly apply Algorithm 3 (in Appendix D) to estimate the value function of each policy given reward function. Finally, Algorithm 2 outputs the policy with the highest estimated value. Since $\mathcal{D}$ has acquired sufficient information, all possible estimations in line 4 are accurate. Together with the property that there exists near-optimal policy in $\Pi^{eval}_{\epsilon/3}$, we have that the output $\widehat{\pi}$ is near-optimal.

5 MAIN RESULTS

In this section, we state our main results, which formalize the techniques and algorithmic ideas we discuss in previous sections.

Theorem 5.1. *We run Algorithm 1 to collect data and let Planning$(\cdot)$ denote the output of Algorithm 2. There exist universal constants $C_1, C_2, C_3, C_4 > 0$[7] such that for any accuracy $\epsilon > 0$ and failure probability $\delta > 0$, as well as $\epsilon < \frac{H(\lambda^\star)^2}{C_4 d^{7/2}\log(1/\lambda^\star)}$, with probability $1-\delta$, for any feasible linear reward function r, Planning(r) returns a policy that is ϵ-optimal with respect to r. In addition, the deployment complexity of Algorithm 1 is H while the number of trajectories is $\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$.*

The proof of Theorem 5.1 is sketched in Section 6 with details in the Appendix. Below we discuss some interesting aspects of our results.

Near optimal deployment efficiency. First, the deployment complexity of our Algorithm 1 is optimal up to a log-factor among all reward-free algorithms with polynomial sample complexity, according to a $\Omega(H/\log_d(NH))$ lower bound (Theorem B.3 of Huang et al. (2022)). In comparison, the deployment complexity of RFLIN (Wagenmaker et al., 2022b) can be the same as their sample complexity (also $\widetilde{O}(d^2H^5/\epsilon^2)$) in the worst case.

Near optimal sample complexity. Secondly, our sample complexity matches the best-known sample complexity $\widetilde{O}(d^2H^5/\epsilon^2)$ (Wagenmaker et al., 2022b) of reward-free RL even when deployment efficiency is not needed. It is also optimal in parameter d and ϵ up to lower-order terms, when compared against the lower bound of $\Omega(d^2H^2/\epsilon^2)$ (Theorem 2 of Wagenmaker et al. (2022b)).

Dependence on $\lambda^\star$. A striking difference of our result comparing to the closest existing work (Huang et al., 2022) is that the sample complexity is independent to the explorability parameter $\lambda^\star$ in the small-ϵ regime. And we do not need to know $\lambda^\star$. This is highly desirable because we only require a non-zero $\lambda^\star$ to exist, and smaller $\lambda^\star$ does not affect the sample complexity asymptotically. In contrast, the best existing result (Algorithm 2 of Huang et al. (2022)) requires the knowledge of explorability parameter $\nu_{\min}$[8] and $\widetilde{O}(1/\epsilon^2\nu^2_{\min})$ trajectories even when ϵ gets close to 0.

Sample complexity in the large-ϵ regime. For the case when ϵ is larger than the threshold: $\frac{H(\lambda^\star)^2}{C_4 d^{7/2}\log(1/\lambda^\star)}$, we can run the procedure with $\epsilon = \frac{H(\lambda^\star)^2}{C_4 d^{7/2}\log(1/\lambda^\star)}$, and the sample complexity will be $\widetilde{O}(\frac{d^9H^3}{(\lambda^\star)^4})$. So the overall sample complexity for any $\epsilon > 0$ can be bounded by $\widetilde{O}(\frac{d^2H^5}{\epsilon^2} + \frac{d^9H^3}{(\lambda^\star)^4})$. This effectively says that the algorithm requires a "Burn-In" period before getting non-trivial results. Similar limitations were observed for linear MDPs before (Huang et al., 2022; Wagenmaker & Jamieson, 2022) so it is not a limitation of our analysis.

Comparison to Qiao et al. (2022). Algorithm 4 (LARFE) of Qiao et al. (2022) tackles reward-free exploration under tabular MDP in $O(H)$ deployments while collecting $\widetilde{O}(\frac{S^2AH^5}{\epsilon^2})$ trajectories. We generalize their result to reward-free RL under linear MDP with the same deployment complexity. More importantly, although a naive instantiation of our main theorem to the tabular MDP only gives $\widetilde{O}(\frac{S^2A^2H^5}{\epsilon^2})$, a small modification to an intermediate argument gives the same $\widetilde{O}(\frac{S^2AH^5}{\epsilon^2})$, which matches the best-known results for tabular MDP. More details will be discussed in Section 7.1.

[7] C_1, C_2, C_3 are the universal constants in Algorithm 1.

[8] $\nu_{\min}$ in Huang et al. (2022) is defined as $\nu_{\min} = \min_{h\in[H]}\min_{\|\theta\|=1}\max_\pi\sqrt{\mathbb{E}_\pi[(\phi_h^\top\theta)^2]}$, which is also measurement of explorability. Note that $\nu_{\min}$ is always upper bounded by 1 and can be arbitrarily small.

6 PROOF SKETCH

In this part, we sketch the proof of Theorem 5.1. Notations ι, $\bar{\epsilon}$, C_i ($i \in [4]$), Π^{exp}, Π^{eval}, $\widehat{\Sigma}_\pi$ and $\widehat{\mathbb{E}}_\pi$ are defined in Algorithm 1. We start with the analysis of deployment complexity.

Deployment complexity. Since for each layer $h \in [H]$, we only deploy one stochastic policy π_h for exploration, the deployment complexity is H. Next we focus on the sample complexity.

Sample complexity. Our proof of sample complexity bound results from induction. With the choice of $\bar{\epsilon}$ and N from Algorithm 1, suppose that $\Lambda_{\tilde{h}}^k$ is empirical covariance matrix from data up to the k-th deployment[9], we assume $\max_{\pi \in \Pi_{\epsilon/3}^{exp}} \mathbb{E}_\pi[\sum_{\tilde{h}=1}^{h-1} \sqrt{\phi(s_{\tilde{h}}, a_{\tilde{h}})^\top (\Lambda_{\tilde{h}}^{h-1})^{-1} \phi(s_{\tilde{h}}, a_{\tilde{h}})}] \leq (h-1)\bar{\epsilon}$ holds and prove that with high probability, $\max_{\pi \in \Pi_{\epsilon/3}^{exp}} \mathbb{E}_\pi[\sqrt{\phi(s_h, a_h)^\top (\Lambda_h^h)^{-1} \phi(s_h, a_h)}] \leq \bar{\epsilon}$.

Note that the induction condition implies that the uncertainty for the first $h-1$ layers is small, we have the following key lemma that bounds the estimation error of $\widehat{\Sigma}_\pi$ from (1).

Lemma 6.1. *With high probability, for all $\pi \in \Delta(\Pi_{\epsilon/3}^{exp})$, $\|\widehat{\Sigma}_\pi - \mathbb{E}_\pi \phi_h \phi_h^\top\|_2 \leq \frac{C_3 d^2 H \bar{\epsilon} \iota}{4}$.*

According to our assumption on ϵ, the *optimal* policy for exploration $\bar{\pi}_h^\star$ [10] satisfies that $\lambda_{\min}(\mathbb{E}_{\bar{\pi}_h^\star} \phi_h \phi_h^\top) \geq \frac{5 C_3 d^2 H \bar{\epsilon} \iota}{4}$. Therefore, $\bar{\pi}_h^\star$ is a feasible solution of (1) and it holds that:

$$\max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \widehat{\mathbb{E}}_{\widehat{\pi}} \left[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_{\pi_h})^{-1} \phi(s_h, a_h)\right] \leq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \widehat{\mathbb{E}}_{\widehat{\pi}} \left[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_{\bar{\pi}_h^\star})^{-1} \phi(s_h, a_h)\right].$$

Moreover, due to matrix concentration and the Lemma 6.1 we derive, we can prove that $(\frac{4}{5}\Sigma_{\bar{\pi}_h^\star})^{-1} \succcurlyeq (\widehat{\Sigma}_{\bar{\pi}_h^\star})^{-1}$ and $(N \cdot \widehat{\Sigma}_{\pi_h})^{-1} \succcurlyeq (2\Lambda_h^h)^{-1}$. [11] In addition, similar to the estimation error of $\widehat{\Sigma}_\pi$, the following lemma bounds the estimation error of $\widehat{\mathbb{E}}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_\pi)^{-1} \phi(s_h, a_h)]$ from (1).

Lemma 6.2. *With high probability, for all $\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}$, $\pi \in \Delta(\Pi_{\epsilon/3}^{exp})$ such that $\lambda_{\min}(\widehat{\Sigma}_\pi) \geq C_3 d^2 H \bar{\epsilon} \iota$,*

$$\left|\widehat{\mathbb{E}}_{\widehat{\pi}} \left[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_\pi)^{-1} \phi(s_h, a_h)\right] - \mathbb{E}_{\widehat{\pi}} \left[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_\pi)^{-1} \phi(s_h, a_h)\right]\right| \leq \frac{\bar{\epsilon}^2}{2d^2} \leq \frac{\bar{\epsilon}^2}{8}.$$

With all the conclusions above, we have (Σ_π is short for $\mathbb{E}_\pi[\phi_h \phi_h^\top]$):

$$\begin{aligned}
&\frac{3\bar{\epsilon}^2}{8} \geq \frac{5d}{4N} + \frac{\bar{\epsilon}^2}{8} \geq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \mathbb{E}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (\frac{4N}{5} \cdot \Sigma_{\bar{\pi}_h^\star})^{-1} \phi(s_h, a_h)] + \frac{\bar{\epsilon}^2}{8} \\
&\geq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \mathbb{E}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_{\bar{\pi}_h^\star})^{-1} \phi(s_h, a_h)] + \frac{\bar{\epsilon}^2}{8} \\
&\geq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \widehat{\mathbb{E}}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_{\bar{\pi}_h^\star})^{-1} \phi(s_h, a_h)] \geq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \widehat{\mathbb{E}}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_{\pi_h})^{-1} \phi(s_h, a_h)] \\
&\geq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \mathbb{E}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (N \cdot \widehat{\Sigma}_{\pi_h})^{-1} \phi(s_h, a_h)] - \frac{\bar{\epsilon}^2}{8} \geq \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \mathbb{E}_{\widehat{\pi}}[\phi(s_h, a_h)^\top (2\Lambda_h^h)^{-1} \phi(s_h, a_h)] - \frac{\bar{\epsilon}^2}{8} \\
&\geq \frac{1}{2} \max_{\widehat{\pi} \in \Pi_{\epsilon/3}^{exp}} \left(\mathbb{E}_{\widehat{\pi}} \sqrt{\phi(s_h, a_h)^\top (\Lambda_h^h)^{-1} \phi(s_h, a_h)}\right)^2 - \frac{\bar{\epsilon}^2}{8}.
\end{aligned}$$

As a result, the induction holds. Together with the fact that $\Pi_{\epsilon/3}^{eval}$ is subset of $\Pi_{\epsilon/3}^{exp}$, we have $\max_{\pi \in \Pi_{\epsilon/3}^{eval}} \mathbb{E}_\pi[\sum_{h=1}^H \sqrt{\phi(s_h, a_h)^\top (\Lambda_h)^{-1} \phi(s_h, a_h)}] \leq H\bar{\epsilon}$. We have the following lemma.

Lemma 6.3. *With high probability, for all $\pi \in \Pi_{\epsilon/3}^{eval}$ and r, $|\widehat{V}^\pi(r) - V^\pi(r)| \leq \widetilde{O}(H\sqrt{d}) \cdot H\bar{\epsilon} \leq \frac{\epsilon}{3}$.*

Finally, since $\Pi_{\epsilon/3}^{eval}$ contains $\epsilon/3$-optimal policy, the greedy policy with respect to $\widehat{V}^\pi(r)$ is ϵ-optimal. We discuss about how we get rid of the $d/\nu_{\min}^2$ dependence in Huang et al. (2022) in Appendix G.

[9] Detailed definition is deferred to Appendix F.4.

[10] Solution of the actual minimization problem, detailed definition in (39).

[11] $\Sigma_{\bar{\pi}_h^\star} = \mathbb{E}_{\bar{\pi}_h^\star}[\phi_h \phi_h^\top]$. The proof is through direct calculation, details are deferred to Appendix F.6.

7 SOME DISCUSSIONS

In this section, we discuss some interesting extensions of our main results.

7.1 APPLICATION TO TABULAR MDP

Under the special case where the linear MDP is actually a tabular MDP and the feature map is canonical basis (Jin et al., 2020b), our Algorithm 1 and 2 are still provably efficient. Suppose the tabular MDP has discrete state-action space with cardinality $|\mathcal{S}| = S$, $|\mathcal{A}| = A$, let $d_m = \min_h \sup_\pi \min_{s,a} d_h^\pi(s,a) > 0$ where d_h^π is occupancy measure, then the following theorem holds.

Theorem 7.1 (Informal version of Theorem H.2)**.** *With minor revision to Algorithm 1 and 2, when ϵ is small compared to d_m, our algorithms can solve reward-free exploration under tabular MDP within H deployments and the sample complexity is bounded by $\widetilde{O}(\frac{S^2AH^5}{\epsilon^2})$.*

The detailed version and proof of Theorem 7.1 are deferred to Appendix H.1 due to space limit. We highlight that we recover the best known result from Qiao et al. (2022) under mild assumption about reachability to all (state,action) pairs. The replacement of one d by S is mainly because under tabular MDP, there are A^S different deterministic policies for layer h and the log-covering number of Π_h^{eval} can be improved from $\widetilde{O}(d)$ to $\widetilde{O}(S)$. In this way, we effectively save a factor of A.

7.2 COMPUTATIONAL EFFICIENCY

We admit that solving the optimization problem (1) is inefficient in general, while this can be solved approximately in exponential time by enumerating π from a tight covering set of $\Delta(\Pi_{\epsilon/3}^{exp})$. Note that the issue of computational tractability arises in many previous works (Zanette et al., 2020a; Wagenmaker & Jamieson, 2022) that focused on information-theoretic results under linear MDP, and such issue is usually not considered as a fundamental barrier. For efficient surrogate of (1), we remark that a possible method is to apply softmax (or other differentiable) representation of the policy space and use gradient-based optimization techniques to find approximate solution of (1).

7.3 POSSIBLE EXTENSIONS TO REGRET MINIMIZATION WITH LOW ADAPTIVITY

In this paper, we tackle the problem of deployment efficient reward-free exploration while the optimal adaptivity under regret minimization still remains open. We remark that deployment complexity is not an ideal measurement of adaptivity for this problem since the definition requires all deployments to have similar sizes, which forces the deployment complexity to be $\widetilde{\Omega}(\sqrt{T})$ if we want regret bound of order $\widetilde{O}(\sqrt{T})$. Therefore, the more reasonable task is to design algorithms with near optimal switching cost or batch complexity. We present the following two lower bounds whose proof is deferred to Appendix H.2. Here the number of episodes is K and the number of steps $T := KH$.

Theorem 7.2. *For any algorithm with the optimal $\widetilde{O}(\sqrt{poly(d,H)T})$ regret bound, the switching cost is at least $\Omega(dH \log\log T)$.*

Theorem 7.3. *For any algorithm with the optimal $\widetilde{O}(\sqrt{poly(d,H)T})$ regret bound, the number of batches is at least $\Omega(\frac{H}{\log_d T} + \log\log T)$.*

To generalize our Algorithm 1 to regret minimization, what remains is to remove Assumption 2.1. Suppose we can do accurate uniform policy evaluation (as in Algorithm 2) with low adaptivity without assumption on explorability of policy set, then we can apply iterative policy elimination (i.e., eliminate the policies that are impossible to be optimal) and do exploration with the remaining policies. Although Assumption 2.1 is common in relevant literature, it is not necessary intuitively since under linear MDP, if some direction is hard to encounter, we do not necessarily need to gather much information on this direction. Under tabular MDP, Qiao et al. (2022) applied absorbing MDP to ignore those "hard to visit" states and we leave generalization of such idea as future work.

8 CONCLUSION

In this work, we studied the well-motivated deployment efficient reward-free RL with linear function approximation. Under the linear MDP model, we designed a novel reward-free exploration algorithm that collects $\widetilde{O}(\frac{d^2H^5}{\epsilon^2})$ trajectories in only H deployments. And both the sample and deployment complexities are near optimal. An interesting future direction is to design algorithms to match our lower bounds for regret minimization with low adaptivity. We believe the techniques we develop (generalized G-optimal design and exploration-preserving policy discretization) could serve as basic building blocks and we leave the generalization as future work.

REFERENCES

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pp. 2312–2320, 2011.

M Mehdi Afsar, Trafford Crump, and Behrouz Far. Reinforcement learning based recommender systems: A survey. *arXiv preprint arXiv:2101.06286*, 2021.

Alekh Agarwal, Sham Kakade, Akshay Krishnamurthy, and Wen Sun. Flambe: Structural complexity and representation learning of low rank mdps. *Advances in neural information processing systems*, 33:20095–20107, 2020.

Shipra Agrawal and Randy Jia. Posterior sampling for reinforcement learning: worst-case regret bounds. In *Advances in Neural Information Processing Systems*, pp. 1184–1194, 2017.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin Yang. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pp. 463–474. PMLR, 2020.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 263–272. JMLR. org, 2017.

Yu Bai, Tengyang Xie, Nan Jiang, and Yu-Xiang Wang. Provably efficient q-learning with low switching cost. *Advances in Neural Information Processing Systems*, 32, 2019.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

Nicolo Cesa-Bianchi, Ofer Dekel, and Ohad Shamir. Online learning with switching costs and other adaptive adversaries. In *Advances in Neural Information Processing Systems*, pp. 1160–1168, 2013.

Xiaoyu Chen, Jiachen Hu, Lin F Yang, and Liwei Wang. Near-optimal reward-free exploration for linear mixture mdps with plug-in solver. *arXiv preprint arXiv:2110.03244*, 2021.

Christoph Dann, Lihong Li, Wei Wei, and Emma Brunskill. Policy certificates: Towards accountable reinforcement learning. In *International Conference on Machine Learning*, pp. 1507–1516. PMLR, 2019.

Hossein Esfandiari, Amin Karbasi, Abbas Mehrabian, and Vahab Mirrokni. Regret bounds for batched bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7340–7348, 2021.

Minbo Gao, Tianle Xie, Simon S Du, and Lin F Yang. A provably efficient algorithm for linear markov decision process with low switching cost. *arXiv preprint arXiv:2101.00494*, 2021.

Zijun Gao, Yanjun Han, Zhimei Ren, and Zhengqing Zhou. Batched multi-armed bandits problem. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Pihe Hu, Yu Chen, and Longbo Huang. Nearly minimax optimal reinforcement learning with linear function approximation. In *International Conference on Machine Learning*, pp. 8971–9019. PMLR, 2022.

Jiawei Huang, Jinglin Chen, Li Zhao, Tao Qin, Nan Jiang, and Tie-Yan Liu. Towards deployment-efficient reinforcement learning: Lower bound and optimality. In *International Conference on Learning Representations*, 2022.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(4), 2010.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pp. 4863–4873, 2018.

Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning. In *International Conference on Machine Learning*, pp. 4870–4879. PMLR, 2020a.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pp. 2137–2143. PMLR, 2020b.

Chi Jin, Qinghua Liu, and Sobhan Miryoosefi. Bellman eluder dimension: New rich classes of rl problems, and sample-efficient algorithms. *Advances in neural information processing systems*, 34:13406–13418, 2021.

Emilie Kaufmann, Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Edouard Leurent, and Michal Valko. Adaptive reward-free exploration. In *Algorithmic Learning Theory*, pp. 865–891. PMLR, 2021.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

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

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

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

Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. In *International Conference on Learning Representations*, 2020.

Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Emilie Kaufmann, Edouard Leurent, and Michal Valko. Fast active learning for pure exploration in reinforcement learning. In *International Conference on Machine Learning*, pp. 7599–7608. PMLR, 2021.

Yifei Min, Tianhao Wang, Dongruo Zhou, and Quanquan Gu. Variance-aware off-policy evaluation with linear function approximation. *Advances in neural information processing systems*, 34: 7598–7610, 2021.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems*, 26, 2013.

Vianney Perchet, Philippe Rigollet, Sylvain Chassang, and Erik Snowberg. Batched bandit problems. *The Annals of Statistics*, 44(2):660–681, 2016.

Dan Qiao, Ming Yin, Ming Min, and Yu-Xiang Wang. Sample-efficient reinforcement learning with loglog(T) switching cost. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 18031–18061. PMLR, 2022.

Yufei Ruan, Jiaqi Yang, and Yuan Zhou. Linear bandits with limited adaptivity and learning distributional optimal design. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 74–87, 2021.

David Simchi-Levi and Yunzong Xu. Phase transitions and cyclic phenomena in bandits with switching constraints. *Advances in Neural Information Processing Systems*, 32, 2019.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Andrew Wagenmaker and Kevin Jamieson. Instance-dependent near-optimal policy identification in linear mdps via online experiment design. *arXiv preprint arXiv:2207.02575*, 2022.

Andrew J Wagenmaker, Yifang Chen, Max Simchowitz, Simon Du, and Kevin Jamieson. First-order regret in reinforcement learning with linear function approximation: A robust estimation approach. In *International Conference on Machine Learning*, pp. 22384–22429. PMLR, 2022a.

Andrew J Wagenmaker, Yifang Chen, Max Simchowitz, Simon Du, and Kevin Jamieson. Reward-free rl is no harder than reward-aware rl in linear markov decision processes. In *International Conference on Machine Learning*, pp. 22430–22456. PMLR, 2022b.

Ruosong Wang, Simon S Du, Lin Yang, and Russ R Salakhutdinov. On reward-free reinforcement learning with linear function approximation. *Advances in neural information processing systems*, 33:17816–17826, 2020.

Tianhao Wang, Dongruo Zhou, and Quanquan Gu. Provably efficient reinforcement learning with linear function approximation under adaptivity constraints. *Advances in Neural Information Processing Systems*, 34, 2021.

Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International Conference on Machine Learning*, pp. 6995–7004. PMLR, 2019.

Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

Andrea Zanette and Emma Brunskill. Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds. In *International Conference on Machine Learning*, pp. 7304–7312. PMLR, 2019.

Andrea Zanette, Alessandro Lazaric, Mykel Kochenderfer, and Emma Brunskill. Learning near optimal policies with low inherent bellman error. In *International Conference on Machine Learning*, pp. 10978–10989. PMLR, 2020a.

Andrea Zanette, Alessandro Lazaric, Mykel J Kochenderfer, and Emma Brunskill. Provably efficient reward-agnostic navigation with linear value iteration. *Advances in Neural Information Processing Systems*, 33:11756–11766, 2020b.

Weitong Zhang, Dongruo Zhou, and Quanquan Gu. Reward-free model-based reinforcement learning with linear function approximation. *Advances in Neural Information Processing Systems*, 34: 1582–1593, 2021a.

Xuezhou Zhang, Adish Singla, et al. Task-agnostic exploration in reinforcement learning. *Advances in Neural Information Processing Systems*, 2020a.

Zihan Zhang, Simon S Du, and Xiangyang Ji. Nearly minimax optimal reward-free reinforcement learning. *arXiv preprint arXiv:2010.05901*, 2020b.

Zihan Zhang, Yuan Zhou, and Xiangyang Ji. Almost optimal model-free reinforcement learning via reference-advantage decomposition. *Advances in Neural Information Processing Systems*, 33: 15198–15207, 2020c.

Zihan Zhang, Jiaqi Yang, Xiangyang Ji, and Simon S Du. Variance-aware confidence set: Variance-dependent bound for linear bandits and horizon-free bound for linear mixture mdp. *arXiv preprint arXiv:2101.12745*, 2021b.

Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Conference on Learning Theory*, pp. 4532–4576. PMLR, 2021.

Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):1–10, 2019.