Linear Reinforcement Learning with Ball Structure Action Space

Zeyu Jia [*] ZYJIA@MIT.COM
Massachusetts Institute of Technology
Randy Jia RANDYJIA@AMAZON.COM
Amazon
Dhruv Madeka MADED@AMAZON.COM
Amazon
Dean P. Foster FOSTER@AMAZON.COM
Amazon

Editors: Shipra Agrawal and Francesco Orabona

Abstract

We study the problem of Reinforcement Learning (RL) with linear function approximation, i.e. assuming the optimal action-value function is linear in a known d-dimensional feature mapping. Unfortunately, however, based on only this assumption, the worst case sample complexity has been shown to be exponential, even under a generative model. Instead of making further assumptions on the MDP or value functions, we assume that our action space is such that there always exist playable actions to explore any direction of the feature space. We formalize this assumption as a "ball structure" action space, and show that being able to freely explore the feature space allows for efficient RL. In particular, we propose a sample-efficient RL algorithm (BallRL) that learns an ϵ-optimal policy using only $\tilde{\mathcal{O}}\left(\frac{H^5 d^3}{\epsilon^3}\right)$ number of trajectories.

Keywords: Markov Decision Process, Reinforcement Learning

1. Introduction

Reinforcement Learning (RL) is a well-studied framework for sequential decision making that has been successfully applied to real-world problems in fields such as game-play (Atari, AlphaGo, Starcraft), robotics, operations management, and more (Mnih et al., 2013; Silver et al., 2016; Vinyals et al., 2017; Kober et al., 2013). However, many of the existing theoretical results can not be applied to many practical applications due to intractably-large number of states and/or actions. A common modeling assumption to address this issue is to assume the existence of a known feature mapping that maps state and action to a d-dimensional feature vector, and that either the underlying MDP dynamics or value functions is linear in this feature mapping. In this work, we consider the common setting where the optimal action-value function (or Q^*-function) is linear and can be written as the inner product of the feature mapping of state-action pairs and some unknown parameter vector. The primary goal is to determine whether there exists algorithms that can achieve a near-optimal policy using an efficient number of samples. Here, efficient sample complexity refers to a polynomial number of samples with respect to the feature dimension d, the horizon H, and the size of the action set $|\mathcal{A}|$.

This setting has garnered much attention recently, however, in the general case pessimistic results have been shown in Weisz et al. (2021b, 2022); Du et al. (2019); Wang et al. (2021), which

[*] This work was done while Zeyu interned at Amazon.

indicate that this problem is exponentially hard in the horizon H or the size of the action set $|\mathcal{A}|$. Furthermore, this pessimistic result is true even with access to a generative model that allows for arbitrary state "resets." Recently, several works have made further assumptions on the MDP that allow for efficient learning when the Q^* function is linear (Jin et al., 2020; Amortila et al., 2022). These typically include additional assumptions on the transition or reward model, or access to additional side information such as expert queries. However, many of these assumptions are restrictive, unrealistic, or unfeasible for many practical use cases, since in the real world, we typically do not we have well-behaved transition models or access to expert oracles.

We seek a general yet practical assumption that is novel, realistic, and amenable to efficient learning. Our work is motivated by the observation that, in some difficult real-world RL applications such as game-play and operations management, it may be easier to think of actions (or consecutive actions) in feature space rather than state space. For example, in a typical dungeon-survival game with various tasks such as fighting monsters, eating food, or searching for treasure, the feature space could include combat statistics, health, and special items. Instead of actions consisting of low-level controls (e.g. movement, engage, run, etc.), we would consider higher-level "feature space" actions (e.g. fight monster, eat food, dig for treasure). Now, we conjecture that if a learning algorithm is always able to play actions to property explore the feature space, then, combined with a d-dimensional feature mapper that exists in the case of linear RL, it should be able to learn a near-optimal policy efficiently. In order to mathematically characterize this property, we introduce the concept of a "ball structure" action space. This assumes that our action space always lies within a d-dimensional ball of radius ρ, so that every direction of the feature space has a corresponding action that can be taken, and therefore at any time step, we are able to explore in any direction of the feature space. However, a perfect ball-shaped action space may be somewhat unrealistic, therefore, we allow some flexibility on the degree of exploration in each direction by considering less-restrictive settings, such as when the action is instead contained within a convex set, or when the radius of the ball is allowed to differ from one time step to the next.

Our main result is the BallRL (pronounced *baller*) algorithm that leverages the exploration capabilities of the ball structure assumption and achieves sample-efficient bounds on learning. The results hold under very mild trajectory learning/PAC learning setting, i.e. we do not assume have access to the action space, and each sampled trajectory gives us information only about action sets along the trajectory, together with total rewards. The algorithm takes advantage of the ball structure action space for exploration, which can be shown to be efficient using the closed form solution of the optimal Bellman Equation, and enjoys a sample complexity bound of $\tilde{\mathcal{O}}(H^5 d^3/\epsilon^2)$ for an ϵ-optimal policy. Furthermore, a similar algorithm and complexity bound hold in the case of convex action set instead of a ball action set, under additional mild assumptions. All together, our results show that with a ball structure action set, we can achieve an exponential improvement in comparison to algorithms to linear Q^* problem without the ball structure assumption. We also demonstrate that our algorithm is easy to implement and is computationally efficient as well.

1.1. Organization of the Paper

The rest of the paper is structured as follows: in Section 2 and 3 we will introduce the problem setting and review prior work in the literature for the linear RL problem. In Section 4 we present our learning algorithm where we demonstrate that efficient learning is possible assuming a ball structure action set. In particular, we present two special generalizations of the assumption: in

Section 4.1 consider when every state in step h shares the same convex action set, and in Section 4.2 we assume the ball structure action set is allowed to vary by state. Note that the simpler ball structure assumption is a special case of both settings. We finally conclude in Section 5 with some discussion.

1.2. Notations

We will use $\langle \cdot, \cdot \rangle$ and $\| \cdot \|_2$ to denote the inner product and the 2-norm in $\mathbb{R}^d$, respectively. Let $B_2(\rho) = \left\{ x \in \mathbb{R}^d | \|x\|_2 \leq \rho \right\}$ represent the L_2-ball of radius ρ in $\mathbb{R}^d$. The expectation $\mathbb{E}^\pi$ will denote the expectations over all trajectories obtained according to π and the underlying transition models and reward functions. We also follow standard big-Oh notation, that is, we will write $A = \mathcal{O}(B)$ if there exists some positive constant c such that $A \leq cB$, and write $A = \tilde{\mathcal{O}}(B)$ if there exists some $c = \text{polylog}(d, H, \delta, 1/\epsilon)$ such that $A \leq cB$. Here, d is the dimension of the feature space, H the time horizon of an episode, δ the high probability parameter, and ϵ the near-optimality parameter of the learned policy.

2. Background

2.1. Preliminaries

A Markov Decision Process (Sutton and Barto, 2018; Puterman, 2014) is a well-known model of the typical reinforcement learning environment. We consider finite-horizon MDPs which are defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, H, r, \mu)$, where the horizon $H \in \mathbb{N}$ and the state space $\mathcal{S} = \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_H$ is known to the learner, but the action space $\mathcal{A}(s)$ of each state $s \in \mathcal{S}$, the transition model $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, the reward function $r : \mathcal{S} \to \mathbb{R}$, and initial state distribution μ are not known. To avoid confusion, without loss of generality we assume that $\mathcal{S}_1, \cdots, \mathcal{S}_H$ have no intersection between each other.

For a given MDP, a policy $\pi : \mathcal{S} \to \mathcal{A}$ is a mapping from state space to the action space, where $\pi(s) \in \mathcal{A}(s)$ for all $s \in \mathcal{S}$. For a given policy π, we define its value functions (V) and Q functions according to the following iterative equations:

$$
\begin{aligned}
V^\pi_{H+1}(s_{h+1}) = 0, \quad Q_{H+1}(s_{H+1}, a_{H+1}) = 0, \quad \forall s_{H+1}, a_{H+1}, \\
Q^\pi_h(s_h, a_h) = r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) V^\pi_{h+1}(s_{h+1}), \\
V^\pi_h(s_h) = Q^\pi_h(s_h, \pi(s_h)).
\end{aligned}
$$

We further define the optimal Q and V function as:

$$
Q^*_h(s_h, a_h) = \max_\pi Q^\pi_h(s_h, a_h), \quad V^*_h(s_h) = \max_\pi V^\pi_h(s_h).
$$

For the optimal Q^* function we have the optimal Bellman Equations, so that for all $1 \leq h \leq H, s_h \in \mathcal{S}_h, a_h \in \mathcal{A}(s_h)$,

$$
Q^*_h(s_h, a_h) = r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \max_{a_{h+1}} Q^*_{h+1}(s_{h+1}, a_{h+1}). \tag{1}
$$

A typical reinforcement learning problem objective is to determine an algorithm that recovers a policy π that performs well relative to the unknown optimal policy π^*; performance is generally

defined by comparing the learned policy's and optimal policy's value functions. In the following section we detail our specific problem setting and objective.

2.2. Our problem setting

Due to the intractability of dealing with extremely high-dimensional state spaces, we make the standard assumption of a linear Q^* function for our problem; that is, the optimal Q^* function is linear in a d-dimensional feature mapping of the state and action:

Assumption 1 (Non-Stationary Linear Q^* Assumption) *For each state-action pair (s, a), there exists a feature vector $\varphi(s, a) \in \mathbb{R}^d$. There are also H unknown parameters $\theta_1^*, \cdots, \theta_H^*$, such that the Q^*-function of state-action pairs has the following parametrization*

$$Q_h^*(s_h, a_h) = \langle \varphi(s_h, a_h), \theta_h^* \rangle, \quad \forall 1 \le h \le H, s_h \in \mathcal{S}_h, a_h \in \mathcal{A}(s_h).$$

While Assumption 1 appears to be a very strong statement on the optimal Q function, it is known that by itself, the assumption is not enough to guarantee efficient learning. Therefore, we present our ball structure assumption that we will show will allow for sample-efficient RL under the linear Q^* assumption.

Assumption 2 (Ball Structure Action Set) *Define the L_2 ball with radius $\rho > 0$ as*

$$B_2(\rho) \triangleq \left\{ x \in \mathbb{R}^d \middle| \|x\|_2 \le \rho \right\}.$$

For each state s, there a feature vector $\varphi(s) \in \mathbb{R}^d$ and a positive number $\rho(s)$ such that

$$\{\varphi(s, a) | a \in \mathcal{A}_s\} = \varphi(s) + B_2(\rho(s)).$$

Remark 1 *Without loss of generality, we can assume that*

$$\mathcal{A}(s) = B_2(\rho_h(s)) \triangleq \left\{ a \in \mathbb{R}^d \middle| \|a\|_2 \le \rho_h(s) \right\},$$

and also

$$\varphi(s, a) = \varphi(s) + a.$$

This is because if there exists two actions a_1, a_2 such that $\varphi(s, a_1) = \varphi(s, a_2)$, then Assumption 1 implies that $Q^*(s, a_1) = Q^*(s, a_2)$. Hence if we remove a_2 from the action set, the value of $V^*(s_1)$ will remain the same. Therefore, if we remove these redundant actions and find a near-optimal policy of the MDP, this policy must also be a near-optimal policy of the original MDP.

By above, after removing redundant actions, we can assume that $a \to \varphi(s, a)$ is an injection, meaning $a \to \varphi(s, a)$ is a one-to-one mapping from $\mathcal{A}(s)$ to $B_2(\rho(s)) + \varphi(s)$. Hence we can replace every action a with $\varphi(s, a) - \varphi(s)$, and then we will have property $\varphi(s, a) = \varphi(s) + a$. Thus, without loss of generality, in the rest of the paper, we will assume $\varphi(s, a) = \varphi(s) + a$ always holds.

Because the transition model and reward function are unknown at the beginning, the learner will only be able to access samples, or realizations, of them by directly interacting with the environment. That is, the learner must execute a policy to actually observe the outcome of those actions. We will consider the following *trajectory learning* setting:

Definition 2 (Trajectory Learning) *At every iteration, the learner first picks a policy (a function mapping every state $s \in \mathcal{S}$ to some action in $\mathcal{A}(s)$), and then a trajectory $(s_1, s_2, \cdots, s_H)$ is sampled according to the true underlying MDP. Only the following two pieces of information are revealed to the learner:*

1. *$\mathcal{A}(s_h)$: The action sets of each state in the trajectory;*
2. *$\sum_{h=1}^{H} R(s_h, a_h)$: The sum of total reward of the trajectory, where $R(s_h, a_h)$ denotes the instant reward obtained by taking action a_h at state s_h, which satisfies $\mathbb{E}[R(s_h, a_h)] = r(s_h, a_h)$.*

Remark 3 *Note that our trajectory learning setting is weaker than the standard PAC learning setting in the literature, where it is assumed that all the information of the trajectories is revealed, including the states $s_1, \cdots, s_H$ and the instantaneous rewards $R(s_h, a_h)$. Our algorithm also does not require the use of a generative model that is standard in some linear Q^* works. Therefore, our algorithm applies to both the common PAC learning setting and generative model setting. For more information about this, please refer to Section 3.*

Finally, in order to measure the performance of our learner's policy, we define the closeness to optimality of a policy via the standard notion of an ϵ-optimal policy:

Definition 4 (ϵ-optimal policy) *If a policy π satisfies*

$$|V^{\pi}(s_0) - V^{*}(s_0)| \leq \epsilon,$$

then we call the policy π an ϵ-optimal policy. Here, V^{π} is the value function with respect to following the policy π, and V^{*} is the value function of the true optimal policy.

Our objective in this work is to develop an algorithm which can find an ϵ-optimal policy with high probability, by using a polynomial number (in d, H and $1/\epsilon$) of trajectory learning iterations.

3. Related Literature

The linear Q^* problem is one of the simplest and most intuitive ways to describe reinforcement learning with parametrization. Many works have studied this setting of RL with the goal to develop a sample-efficient algorithm to learn a near-optimal policy. However, in the most general case, recent work has yielded only pessimistic results related to this problem. In Weisz et al. (2021b, 2022); Du et al. (2019); Wang et al. (2021); Foster et al. (2021), the linear Q^* problem has been shown to be exponentially hard in d or H or $|\mathcal{A}|$, even when the number of actions are small. Their main idea revolves around showing a lower exponential bound by constructing a needle in haystack-type MDP, i.e., among exponentially many actions there is only one action that induces rewards, hence in order to find the optimal action the learner must run policies an exponential number of times. Additionally, they also adopt the Johnson-Lindenstrauss lemma to show that they can choose these actions such that every two actions are sufficiently far away from each other, so that querying non-optimal actions gives limited information of the optimal action.

Apart from pessimistic results, there are many works which demonstrate that the linear Q^* problem is polynomially solvable with added additional assumptions. Assumptions are quite varied and numerous, and we attempt to give an overview of the different types that have allowed for

efficient learning. If for all policies π, the Q-function Q^π can be linearly parameterized, then the problem is polynomially solvable by using approximate policy iteration (Lattimore et al., 2020). If both the transition model and reward function are deterministic, then the problem is polynomially solvable by eliminating functions that does not satisfy the linear Q^* function assumptions (Wen and Van Roy, 2013). If a 'core set' (that is, features of every state action pairs can be written as the convex combinations of features in the core set) exists for the MDP, then the problem is polynomially solvable (Zanette et al., 2019; Shariff and Szepesvári, 2020). In comparison to our assumption, our algorithm has access to an orthogonal basis at first, which is similar to the idea of core set. However, the core set cannot capture our setting, since the ball cannot be written as convex combination of basis vectors - simply adopting their algorithm would induce exponential sample complexity. Under the assumption that the action set is finite, the TensorPlan Algorithm in Weisz et al. (2021a) can obtain an ϵ-optimal policy using $\text{poly}\left(\left(\frac{dH}{\epsilon}\right)^{|\mathcal{A}|}\right)$ number of samples. Alternatively, if we assume access to an expert oracle which gives the value of $Q^*(s,a)$ when queried at state (s,a), the DELPHI algorithm can solve these linear Q^* problem in polynomial time using no more than $\mathcal{O}(d)$ calls of expert queries (Amortila et al., 2022).

Beyond the linear Q^* problem, there are also several works which achieve polynomial sample complexity under general assumptions of the MDP's underlying properties. If the transition model can be linearly parametrized, then the MDP problem becomes polynomially solvable as shown in Jin et al. (2020); Yang and Wang (2019, 2020); Jia et al. (2020). However, a linear transition model is a fairly strong assumption and generally not a very practical assumption, as most systems do not behave as such. There are also works focused on generalized function approximations, e.g. Eluder Dimensions (Ayoub et al., 2020; Wang et al., 2020), Bellman Rank (Jiang et al., 2017), Bellman Eluder Dimension (Jin et al., 2021), Bilinear Class (Du et al., 2021), Bellman Closeness (Jin et al., 2021; Zanette et al., 2020). However, again these assumptions on the models are either hard to verify in practice or generally do not occur in real world systems, which makes the use of these algorithms difficult to justify in practice.

4. The BallRL Algorithm

We now present the main result of our paper, that is, an algorithm that achieves polynomial sample complexity in the linear Q^* setting under the assumption of a ball structure action space. Before proceeding with the details, we highlight two versions of our algorithm, Convex-BallRL and DiffR-BallRL, both of which are essentially extensions of the standard ball structure assumption (Assumption 2). In the first case, we consider convex action sets where the action sets are identical across state. While every state necessarily has the same set of actions to take, the magnitude to which one can explore different directions is permitted to vary, so long the overall action set is convex. This can be seen as a slightly more realistic version of the standard ball assumption, as in practice it may be difficult to guarantee the magnitude of every feature direction to be the same. In the second case, we consider the standard ball structure action set but allow the action set to vary depending on the current state. The motivation behind these two slightly different settings is to represent a more realistic generalization of the original ball structure presented earlier, as in practical settings action spaces may not always be uniformly a perfect ball.

4.1. Identical Convex Action Sets within One Step

In this section, we make the assumption that the action set $\mathcal{A}(s_h)$ is identical for every $s_h \in \mathcal{S}_h$, and moreover, we assume the action sets are regular convex sets, which is a generalization of the ball structure presented in Assumption 2. Intuitively, the action set is contained between a smaller radius and a larger radius ball.

Definition 5 (Regular Convex Set) *We call a set $\mathcal{M} \subset \mathbb{R}^d$ a regular convex set with parameter $B \geq 1$ if there exists $\eta \geq \rho > 0$ such that $\frac{\eta}{\rho} = B$ and*

$$B_2(\rho) \subset \mathcal{M} \subset B_2(\eta).$$

Remark 6 *Regular convex sets include many different types of structures such as balls, cubes, ellipsoids, etc. Some specific examples include:*

1. *All balls are regular convex sets with parameter 1;*
2. *Cubes in d dimension are regular convex sets with parameter $\sqrt{d}$;*
3. *Ellipsoids are regular convex sets with parameter $\frac{a_{\max}}{a_{\min}}$, where $a_{\max}, a_{\min}$ are the longest and shortest axes.*

Let us formally characterize our assumption for the setting with convex action sets.

Assumption 3 (Identical Convex Action Sets within One Step) *For every $1 \leq h \leq H$, there exists a regular convex set $\mathcal{A}_h$ with parameter B, such that for all $s_h \in \mathcal{S}_h$, $\mathcal{A}(s_h) = \mathcal{A}_h$. Specifically, there exists $\rho_1, \cdots, \rho_H, \eta_1, \cdots, \eta_H$, such that for every $1 \leq h \leq H$ we have*

$$\frac{\eta_h}{\rho_h} = B, \qquad B_2(\rho_h) \subset \mathcal{A}_h \subset B_2(\eta_h).$$

Without loss of generality, we also assume that the features still satisfy $\varphi(s, a) = \varphi(s) + a$.

We develop an algorithm, Convex-BallRL, that works in the trajectory learning setting (Definition 2) under Assumption 1 and 3, and is guaranteed to find an ϵ-optimal policy using a polynomial number of trajectories.

4.1.1. INTUITION AND KEY IDEAS

Before presenting the algorithm itself, we provide some intuition on the key ideas behind our algorithm. With loss of generality, we assume that we know the value of $\rho_1, \cdots, \rho_H$ at the beginning. Otherwise, we can run one trajectory according to any policy, then all the action sets $\mathcal{A}_1, \cdots, \mathcal{A}_H$ will be revealed to us, from which we can determine the values of $\rho_1, \cdots, \rho_H$.

We start by observing the following equation due to telescoping of Bellman Equation (1):

$$\mathbb{E}\left[\langle \varphi(s_1), \theta_1^* \rangle\right] + \mathbb{E}\left[\sum_{h=1}^{H} \langle a_h, \theta_h^* \rangle\right] = \sum_{h=1}^{H} \mathbb{E}[R(s_h, a_h)] + \sum_{h=1}^{H} \rho_{h+1} \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle a_{h+1}, \theta_{h+1}^* \rangle. \quad (2)$$

Our next observation is that the first term of LHS and the second term of RHS in (2) are identical for every policy. Hence, if we compare (2) between two different policies, we can obtain information

of θ_h^* (according to $\langle a_h, \theta_h^* \rangle$) based on the first term in RHS, which can be estimated through sampled trajectories. Formally, we choose π_0 to be the all-zero policy:

$$\pi_0(s_h) = \mathbf{0} \in \mathbb{R}^d, \quad \forall 1 \leq h \leq H, \tag{3}$$

and $\pi_{h,i}$ $(1 \leq h \leq H, 1 \leq i \leq d)$ to be the following policy: for $1 \leq h' \leq H$ and $s_{h'} \in \mathcal{S}_{h'}$,

$$\pi_{h,i}(s_{h'}) = \begin{cases} \mathbf{0} & \text{if } h' \neq h, \\ \rho_h \mathbf{e}_i & \text{if } h' = h \end{cases}, \tag{4}$$

where $\mathbf{e}_i$ is the i-th basis vector in $\mathbb{R}^d$. Comparing (2) according to policy π_0 and also policy $\pi_{h,i}$, we obtain that

$$\rho_h \langle \mathbf{e}_i, \theta_h^* \rangle = \mathbf{E}^{\pi_{h,i}} \left[\sum_{h=1}^{H} R(s_h, a_h) \right] - \mathbf{E}^{\pi_0} \left[\sum_{h=1}^{H} R(s_h, a_h) \right].$$

The right hand side can be estimated according to trajectories from policy π_0 and $\pi_{h,i}$, which leads to the estimate of i-th component of θ_h^*.

Finally after getting accurate enough estimations $\hat{\theta}_h \in \mathbb{R}^d$ on θ_h^*, we adopt the greedy policy, i.e.

$$\pi(s_h) = \arg\max_{a_h \in \mathcal{A}_h} \langle a_h, \hat{\theta}_h \rangle, \tag{5}$$

and then show that this policy is a nearly optimal policy.

4.1.2. ALGORITHM AND SAMPLE COMPLEXITY

The pseudocode for BallRL with convex action sets is given in Algorithm 1. The main result of this section is the following theorem about its sample complexity, in particular, that it has polynomial sample complexity. The complete proof details are provided in Appendix A.

Theorem 7 *For any $\delta > 0$, if we choose*

$$M = \frac{8H^2B^2d\log(2dH/\delta)}{\epsilon^2},$$

then with probability at least $1 - \delta$, the output policy from the above algorithm is an ϵ-optimal policy. The total number of trajectories used in this algorithm is

$$\frac{16H^3B^2d^2\log(2dH/\delta)}{\epsilon^2}.$$

Remark 8 *If we assume all action sets have ball structure, then all action sets are regular convex sets with parameter 1. Hence the above algorithm is guaranteed to find an ϵ-optimal policy using $\tilde{\mathcal{O}}\left(\frac{H^3d^2}{\epsilon^2}\right)$ number of trajectories.*

Algorithm 1 Convex-BallRL

Input: d, H, M, ρ_h;
  for $h = 1 : H$ **do**
    for $i = 1 : d$ **do**
      Choose policy π_0 such that $\pi_0(s_{h'}) = \mathbf{0}$ for any $s_{h'}, h' \in [H]$.
      Choose policy $\pi_{h,i}$ such that $\pi_{h,i}(s_{h'}) = \mathbf{0}$ if $h' \neq h$ and $\pi_{h,i}(s_h) = \rho_h \mathbf{e}_i$ for all s_h.
      Get M trajectories from policy π_0 and $\pi_{h,i}$ respectively, and calculate the average of total rewards: R_0 and $R_{h,i}$.
    end
  Let $\hat{\theta}_h$ to be the vector whose i-th component to be $\frac{R_{h,i}-R_0}{\rho_h}$.
  Let $a_h = \arg\max_{a \in \mathcal{A}_h} \langle a_h, \hat{\theta}_h \rangle$.
  end
Output: Policy π: $\pi(s_h) = a_h$ for any $s_h, h \in [H]$.

4.2. Different Radius

In this section, we abandon the assumption that all states s_h in step h share identical action sets, and allow for the action set to vary depending on the state. However, we again assume that the action set corresponding to each state is a ball as in Assumption 2. We further assume that the norm of θ_h are all the same for $1 \leq h \leq H$, and also that the norm of features, rewards and radius are bounded:

Assumption 4 (Boundedness) *For each state $s \in \mathcal{S}$, action $a \in \mathcal{A}(s)$, we have*

$$\|\varphi(s, a)\|_2 \leq 1;$$

For some $\Theta \in [0, 1]$, we have

$$\|\theta_1\|_2 = \cdots = \|\theta_H\|_2 = \Theta;$$

For every trajectories $(s_1, a_1, \cdots, s_H, a_H)$, we have

$$0 \leq \sum_{h=1}^{H} R(s_h, a_h) \leq 1, \quad 0 \leq \sum_{h=1}^{H} \rho(s_h) \leq 1.$$

We again aim to develop an algorithm that works under Definition 2 (trajectory learning), but under Assumption 1 (Linear Q^* assumption), 2 (Ball Structure Assumption) and 4 (Boundedness Assumption).

4.2.1. INTUITION AND KEY IDEAS

We begin by presenting the following key ideas of our algorithm:

To Exploit the Ball Structure Action space Similar to (2) in Convex-BallRL, our algorithm is again based on the telescoping of Bellman Equation (1), which exploits the ball structure of the action space:

$$\langle \varphi(s_1), \theta_1^* \rangle + \mathbb{E}^{\pi} \left[\sum_{h=1}^{H} \langle a_h, \theta_h^* \rangle \right] = \mathbb{E}^{\pi} \left[\sum_{h=1}^{H} R(s_h, a_h) \right] + \Theta \cdot \mathbb{E}^{\pi} \left[\sum_{h=1}^{H} \rho(s_{h+1}) \right]. \tag{6}$$

Estimation of Norm by Grid Search According to (6), we can estimate $\theta_1^*, \cdots, \theta_H^*$ in the LHS based on the RHS. However, Θ, which is the norm of the unknown parameters $\theta_1^*, \cdots, \theta_H^*$, is difficult to estimate. Hence in our algorithm, we adopt a grid search method for the value of Θ: choosing $\xi = l\varepsilon$ for $1 \leq l \leq \frac{1}{\varepsilon}$, so that at least one such ξ is ε-close to the true Θ. Therefore, if we develop our policy based on these ξ, then at least one policy will necessarily be an ϵ-optimal policy.

Hierarchical Exploration For the exploration in our algorithm, we will choose actions to be $\rho(s_h)\mathbf{e}_i$ for $1 \leq i \leq d$ in order to give information about the i-th component of θ_h^*. However, one problem is that this estimation has accuracy at most $1/\rho(s_h)$, which will explode as $\rho(s_h)$ goes to zero. To deal with this problem, we consider a hierarchical exploration method:

Suppose the policy we currently use for exploration is π_e, and the greedy policy we calculated is π. We can show that the exploration will guarantee $1/(\mathbb{E}^{\pi_e}[\rho(s_h)]\sqrt{M})$ accuracy on θ_h^* (up to logarithmic factors), and hence the error of π is $\mathbb{E}^{\pi}[\rho(s_h)]/(\mathbb{E}^{\pi_e}[\rho(s_h)]\sqrt{M})$. Therefore, if for every $1 \leq h \leq H$ we all have $\mathbb{E}^{\pi}[\rho(s_h)] \leq 2\mathbb{E}^{\pi_e}[\rho(s_h)]$, then the error of the greedy policy is of order $2/\sqrt{M}$, which can be bounded by choosing some proper M. Otherwise, we use the greedy policy π to construct another exploration policy as follows:

$$\pi_{h,0}(s_{h'}) = \begin{cases} \pi(s_{h'}) & \text{if } 1 \leq h' < h; \\ 0 & \text{if } h' \geq h; \end{cases}$$
$$\pi_{h,i}(s_{h'}) = \begin{cases} \pi(s_{h'}) & \text{if } 1 \leq h' < h; \\ \rho(s_h)\mathbf{e}_i & \text{if } h' = h; \\ 0 & \text{if } h' \geq h. \end{cases} \tag{7}$$

Then these new policies $\pi_{h,i}$ will guarantee that $\mathbb{E}^{\pi_{h,i}}[\rho(s_h)] \geq 2\mathbb{E}^{\pi_e}[\rho(s_h)]$, i.e. the value of $\mathbb{E}^{\pi_e}[\rho(s_h)]$ becomes at least twice of its previous value. Therefore, we can show that this process will end in at most $H\log(1/\varepsilon)$ number of times, provided that the initial value of $\mathbb{E}^{\pi_e}[\rho(s_h)]$ is at least ε.

Ignore Small Radius We will show that if within a policy π, the expected radius $\mathbb{E}^{\pi}[\rho(s_h)]$ at step h is smaller than ε, then the effect of different actions within this step can be ignored, and we do not need to carry out the above exploration in this step.

4.2.2. ALGORITHM AND SAMPLE COMPLEXITY

Combine these ideas together, we construct the following Algorithm 2.

Algorithm 2 DiffR-BallRL

Input: $d, H, M_1, M_2, \varepsilon, \eta, L = 1/\eta$;
Let $\rho_h = 0, \rho_h^l = 1$ for all $1 \le h \le H, 1 \le l \le L$.
Let policy π_l' to be the policy such that $\pi(s_h) = 0$ for all $s_h \in \mathcal{S}_h$ and $1 \le h \le H$ and $1 \le l \le L$.
while $\exists 1 \le h \le H$ *and* $1 \le l \le L$ *such that* $\rho_h^l \ge 2\rho_h$ *and* $\rho_h^l \ge \varepsilon$ **do**
Fix h, l to be the one that $\rho_h^l \ge \rho_h$, and construct Policy $\pi_{h,0}$ and $\pi_{h,i}$ $(1 \le i \le d)$ based on policy π_l' according to (7) (use $\pi = \pi_l'$ in (7)).
Collect M_1 trajectories according to policy $\pi_{h,i}$ for $0 \le i \le d$ each, and calculate the average of total reward $\sum_{h'=1}^{H} R(s_h, a_h)$, and average of $\sum_{h=2}^{H} \rho(s_h)$ as $R_{h,i}$ and $s_{h,i}$, respectively.
$\rho_h \leftarrow \rho_h^l$.
for $l = 1 : L$ **do**
Let $\xi = l\eta$.
Calculate $\hat{\theta}_{h,i}^l = \frac{(s_{h,i} - s_{h,0})\xi + R_{h,i} - R_{h,0}}{\rho_h}$ for $1 \le i \le d$.
end
Let $\hat{\theta}_h^l = \sum_{i=1}^{d} \hat{\theta}_{h,i}^l \mathbf{e}_i$.
Construct policy π_l':

$$\pi_l'(s_{h'}) = \underset{a_{h'} \in B_2(\rho(s_{h'}))}{\arg\max} \left\langle a_{h'}, \hat{\theta}_{h'}^l \right\rangle, \quad \forall s_{h'} \in \mathcal{S}_{h'}, 1 \le h' \le H.$$

For $1 \le l \le L$, run policy π_l' each for M_2 times, and calculate the average total reward $R_1, \cdots, R_L$, and also calculate the average of $\rho(s_{h'})$ as $\rho_{h'}^1, \cdots, \rho_{h'}^L$ for $1 \le h' \le H$.
Let $l = \arg\max R_l$ and $\pi = \pi_l'$.
end
Output: Policy π.

Finally, we arrive at our main result - that DiffR-BallRL is sample efficient. The proof details are provided in Appendix B.

Theorem 9 *For any* $0 < \delta < 1$*, with the choice*

$$\varepsilon = \frac{\epsilon}{8H}, \quad \delta' = \frac{\delta}{(d + 3HL)(1 + H\log_2(1/\varepsilon))}, \quad \eta = \frac{\epsilon}{8Hd},$$
$$M_2 = 2\log(1/\delta') \cdot \frac{16(2 + 4H + 2Hd)^2}{\epsilon^2}, \quad M_1 = 2\log(1/\delta') \cdot \frac{256H^2d^2}{\epsilon^2}, \quad L = \frac{1}{\eta}$$

Algorithm 2 will output an ϵ*-optimal policy with probability at least* $1 - \delta$*. This algorithm will use at most*

$$\tilde{\mathcal{O}}\left(\frac{H^5 d^3}{\epsilon^3}\right)$$

number of trajectories.

Proof Sketch of Theorem 9. Our first step of the proof is to use Bellman Equation to prove (6):

$$\langle \varphi(s_1), \theta_1^* \rangle + \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} \langle a_h, \theta_h^* \rangle\right] = \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} R(s_h, a_h)\right] + \Theta \cdot \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} \rho(s_{h+1})\right],$$

which can be obtained through telescoping the following closed form of Bellman Equation at step h:

$$\mathbb{E}^{\pi}\left[\langle\varphi(s_h),\theta_h^*\rangle+\langle a_h,\theta_h^*\rangle\right]=\mathbb{E}^{\pi}\left[R(s_h,a_h)+\rho(s_{h+1})\cdot\|\theta_{h+1}\|_2+\langle\varphi(s_{h+1}),\theta_{h+1}^*\rangle\right].$$

Our second step is a result bounding the value function error of the greedy policies:

$$\mathbb{E}[V_1^*(s_1)]-\mathbb{E}[V_1^{\pi}(s_1)]\leq 2\sum_{h=1}^{H}\mathbb{E}^{\pi}[\rho(s_h)]\cdot\left\|\hat{\theta}_h-\theta_h^*\right\|_2.$$

Therefore, if $\mathbb{E}^{\pi}[\rho(s_h)]$ is small for some h (say less than ϵ), then we can ignore this term, since it will never make big difference on the error. In the following, we assume that $\mathbb{E}^{\pi}[\rho(s_h)]\geq\epsilon$.

Next, we observe that there exists some $\Theta'=l_0\eta$ such that $|\Theta'-\Theta|\leq\eta$. And for the iteration l_0, according to Hoeffding inequality we can get

$$\rho_h\left\|\theta_h-\hat{\theta}_h^{l_0}\right\|_2\leq\eta d+2d\sqrt{\frac{2\log(1/\delta')}{M_1}}+d\sqrt{\frac{2\log(1/\delta')}{M_2}}$$

with high probability, where ρ_h is the expectation of $\rho(s_h)$ according to the exploration policy.

Finally, if $\rho_h^{l_0}$ of the greedy policy satisfies that $\rho_h^{l_0}\leq\rho_h$, then the above inequality can guarantee that this greedy policy is near optimal. Otherwise, the value of ρ_h will become twice as before according to our algorithm, and this process will terminate in $\log(1/\epsilon)$ number of iterations, since according to our assumption the initial value of ρ_h is at least ϵ, and ρ_h cannot be large than 1.

Combining these steps together, we can show that the algorithm will end in certain number of iterations, and when the algorithm ends, it will output a near optimal policy with high probability.

5. Conclusion

We presented the BallRL reinforcement learning algorithm that provides sample-efficient learning guarantees when the optimal action-value function is linear and actions exhibit a ball structure. We further generalized the ball structure to both convex actions sets and changing ball radius between states. Our techniques demonstrate that there is hope for efficient learning in linear RL when actions can sufficiently explore the feature space. The ball structure assumption itself is a sufficient, but not fully necessary condition to ensure full exploration of the feature space. We believe that the idea of the action set allowing for sufficient exploration can be achieved (perhaps approximately) in many practical settings.

An interesting research direction is to dive deeper into assuming the actions lie in convex sets instead of a pure ball structure. While the problem can be solved when the action set is consistent between all actions, it remains to be shown if convex sets can vary between states. Additionally, with different radii, our algorithm is polynomially efficient when unknown parameters across the horizon share the same norm. It would be valuable to see whether this assumption can be removed and parameters allowed to have different norms.

Acknowledgments

We thank Philip Amortila for helpful discussions.

References

Philip Amortila, Nan Jiang, Dhruv Madeka, and Dean P Foster. A few expert queries suffices for sample-efficient rl with resets and linear value approximation. *arXiv preprint arXiv:2207.08342*, 2022.

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*, pages 463–474. PMLR, 2020.

Simon Du, Sham Kakade, Jason Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear classes: A structural framework for provable generalization in rl. In *International Conference on Machine Learning*, pages 2826–2836. PMLR, 2021.

Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. Is a good representation sufficient for sample efficient reinforcement learning? *arXiv preprint arXiv:1910.03016*, 2019.

Dylan J Foster, Sham M Kakade, Jian Qian, and Alexander Rakhlin. The statistical complexity of interactive decision making. *arXiv preprint arXiv:2112.13487*, 2021.

Zeyu Jia, Lin Yang, Csaba Szepesvari, and Mengdi Wang. Model-based reinforcement learning with value-targeted regression. In *Learning for Dynamics and Control*, pages 666–686. PMLR, 2020.

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low bellman rank are pac-learnable. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2017.

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

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.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Roshan Shariff and Csaba Szepesvári. Efficient planning in large mdps with weak linear function approximation. *Advances in Neural Information Processing Systems*, 33:19163–19174, 2020.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

Ruosong Wang, Russ R Salakhutdinov, and Lin Yang. Reinforcement learning with general value function approximation: Provably efficient approach via bounded eluder dimension. *Advances in Neural Information Processing Systems*, 33:6123–6135, 2020.

Yuanhao Wang, Ruosong Wang, and Sham Kakade. An exponential lower bound for linearly realizable mdp with constant suboptimality gap. *Advances in Neural Information Processing Systems*, 34:9521–9533, 2021.

Gellért Weisz, Philip Amortila, Barnabás Janzer, Yasin Abbasi-Yadkori, Nan Jiang, and Csaba Szepesvári. On query-efficient planning in mdps under linear realizability of the optimal state-value function. In *Conference on Learning Theory*, pages 4355–4385. PMLR, 2021a.

Gellért Weisz, Philip Amortila, and Csaba Szepesvári. Exponential lower bounds for planning in mdps with linearly-realizable optimal action-value functions. In *Algorithmic Learning Theory*, pages 1237–1264. PMLR, 2021b.

Gellért Weisz, Csaba Szepesvári, and András György. Tensorplan and the few actions lower bound for planning in mdps under linear realizability of optimal value functions. In *International Conference on Algorithmic Learning Theory*, pages 1097–1137. PMLR, 2022.

Zheng Wen and Benjamin Van Roy. Efficient exploration and value function generalization in deterministic systems. *Advances in Neural Information Processing Systems*, 26, 2013.

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

Lin Yang and Mengdi Wang. Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound. In *International Conference on Machine Learning*, pages 10746–10756. PMLR, 2020.

Andrea Zanette, Alessandro Lazaric, Mykel J Kochenderfer, and Emma Brunskill. Limiting extrapolation in linear approximate value iteration. *Advances in Neural Information Processing Systems*, 32, 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*, pages 10978–10989. PMLR, 2020.

Appendix A. Proof of Theorem 7

Proof [Proof of Theorem 7]

First of all, according to Bellman Equation (1), we have

$$\begin{aligned}
&\langle \varphi(s_h), \theta_h^* \rangle + \langle a_h, \theta_h^* \rangle = Q_h^*(s_h, a_h) \\
&= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) V_{h+1}^*(s_{h+1}) \\
&= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle \varphi(s_{h+1}) + a_{h+1}, \theta_{h+1}^* \rangle \\
&= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \rho_{h+1} \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle a_{h+1}, \theta_{h+1}^* \rangle \\
&= \mathbb{E}\left[R(s_h, a_h) + \langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle + \rho_{h+1} \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle a_{h+1}, \theta_{h+1}^* \rangle \middle| s_h, a_h \right],
\end{aligned}$$

where $R(s_h, a_h)$ is the instant reward we obtain after choosing action a_h from state s_h ($R(s_h, a_h)$ has mean $r(s_h, a_h)$). Hence for a given fixed policy π, suppose a trajectory following this policy is $(s_1, a_1, \cdots, s_H, a_H)$, then we have

$$\mathbb{E}^\pi \left[\langle \varphi(s_h), \theta_h^* \rangle \right] + \mathbb{E}^\pi [\langle a_h, \theta_h^* \rangle] = \mathbb{E}^\pi [\langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle] + \mathbb{E}^\pi \left[R(s_h, a_h) \right] + \rho_{h+1} \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle a_{h+1}, \theta_{h+1}^* \rangle.$$

Summing this up from $h = 1$ to $h = H$ and noticing that $\theta_{h+1}^* = 0$, we obtain that

$$\mathbb{E}^\pi \left[\langle \varphi(s_1), \theta_1^* \rangle \right] + \mathbb{E}^\pi \left[\sum_{h=1}^H \langle a_h, \theta_h^* \rangle \right] = \sum_{h=1}^H \mathbb{E}^\pi [R(s_h, a_h)] + \sum_{h=1}^H \rho_{h+1} \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle a_{h+1}, \theta_{h+1}^* \rangle.$$

With our choice of π_0 (the policy which choose action 0 at any state and step), we obtain

$$\mathbb{E}^{\pi_0} \left[\langle \varphi(s_1), \theta_1^* \rangle \right] = \sum_{h=1}^H \mathbb{E}^{\pi_0} [R(s_h, a_h)] + \sum_{h=1}^H \rho_{h+1} \max_{a_{h+1} \in \mathcal{A}_{h+1}} \langle a_{h+1}, \theta_{h+1}^* \rangle.$$

We notice that for every policy π, $\mathbb{E}^\pi \left[\langle \varphi(s_1), \theta_1^* \rangle \right]$ are identical. Hence after subtracting the above two equations, we get

$$\mathbb{E}^\pi \left[\sum_{h=1}^H \langle a_h, \theta_h^* \rangle \right] = \mathbb{E}^\pi \left[\sum_{h=1}^H R(s_h, a_h) \right] - \mathbb{E}^{\pi_0} \left[\sum_{h=1}^H R(s_h, a_h) \right]. \tag{8}$$

With our choice of policy $\pi_{h,i}$, the above equation indicates that

$$\rho_h \theta_{h,i}^* = \rho_h \langle \mathbf{e}_i, \theta_h^* \rangle = \mathbb{E}^{\pi_{h,i}} \left[\sum_{h=1}^H R(s_h, a_h) \right] - \mathbb{E}^{\pi_0} \left[\sum_{h=1}^H R(s_h, a_h) \right],$$

where $\theta_{h,i}^*$ is the i-th component of θ_h^*. According to our algorithm, we have $\rho_h \hat{\theta}_{h,i} = R_{h,i} - R_{h,0}$, and we also have

$$\mathbf{E} \left[R_{h,i} - R_{h,i} \right] = \mathbb{E}^{\pi_{h,i}} \left[\sum_{h=1}^H R(s_h, a_h) \right] - \mathbb{E}^{\pi_0} \left[\sum_{h=1}^H R(s_h, a_h) \right] = \rho_h \theta_{h,i}^*.$$

Therefore, according to Hoeffding inequality, with probability at least $1-2\delta'$ we have,

$$\rho_h \left|\hat{\theta}_{h,i} - \theta^*_{h,i}\right| \leq 2\sqrt{\frac{\log(1/\delta')}{2M}} = \sqrt{\frac{2\log(1/\delta')}{M}},$$

if we assume that $\sum_{h=1}^H R(s_h, a_h) \in [0,1]$ always holds. This indicates that with probability at least $1-2dH\delta'$, for every $1 \leq h \leq H$,

$$\rho_h \left\|\hat{\theta}_h - \theta^*_h\right\|_2 \leq \sqrt{\frac{2d\log(1/\delta')}{M}}.$$

Moreover, since $a_h = \arg\max_{a\in\mathcal{A}_h} \langle a_h, \hat{\theta}_h\rangle$, we have

$$\begin{aligned}
\langle a_h, \theta^*_h\rangle - \langle a^*_h, \theta^*_h\rangle &= \langle a_h, \theta^*_h - \hat{\theta}_h\rangle + \langle a_h, \hat{\theta}_h\rangle - \langle a^*_h, \hat{\theta}_h\rangle + \langle a^*_h, \hat{\theta}_h - \theta^*_h\rangle \\
&\leq \langle a_h, \theta^*_h - \hat{\theta}_h\rangle + \langle a^*_h, \hat{\theta}_h - \theta^*_h\rangle \\
&\leq 2\eta_h \cdot \|\theta_h - \hat{\theta}_h\|_2 \\
&\leq 2B \cdot \sqrt{\frac{2d\log(1/\delta')}{M}},
\end{aligned}$$

where the first inequality is due to $a_h = \arg\max_{a\in\mathcal{A}_h} \langle a_h, \hat{\theta}_h\rangle$, and the second inequality is due to $a_h, a^*_h \in \mathcal{A}_h \subset B_2(\eta_h)$. Therefore, according to (8) we have

$$V^*(s_1) - V^\pi(s_1) \leq 2BH\sqrt{\frac{2d\log(1/\delta')}{M}}.$$

With our choice of $\delta' = \frac{\delta}{2dH}$ and $M = \frac{8H^2B^2d\log(2dH/\delta)}{\epsilon^2}$, we have with probability at least $1-\delta$, the output policy π satisfies that

$$V^*(s_1) - V^\pi(s_1) \leq \epsilon.$$

■

Appendix B. Proof of Theorem 9

To prove the complexity of our algorithm, we first show the following several lemmas:

Lemma 10 *For each policy π, we have the following equation*

$$\langle \varphi(s_1), \theta^*_1\rangle + \mathbb{E}^\pi\left[\sum_{h=1}^H \langle a_h, \theta^*_h\rangle\right] = \mathbb{E}^\pi\left[\sum_{h=1}^H R(s_h, a_h)\right] + \Theta \cdot \mathbb{E}^\pi\left[\sum_{h=1}^H \rho(s_{h+1})\right], \tag{9}$$

where we use the notation that $\rho(s_{H+1}) = 0$.

Proof According to the Bellman Equation (1) at step h, we have

$$\begin{aligned}
Q_h^*(s_h, a_h) &= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) V_{h+1}^*(s_{h+1}) \\
&= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \max_{a_{h+1} \in \mathcal{A}(s_{h+1})} \langle \varphi(s_{h+1}) + a_{h+1}, \theta_{h+1}^* \rangle \\
&= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \max_{a_{h+1} \in B_2(\rho(s_{h+1}))} \langle \varphi(s_{h+1}) + a_{h+1}, \theta_{h+1}^* \rangle \\
&= r(s_h, a_h) + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \rho(s_{h+1}) \|\theta_{h+1}^*\|_2 + \sum_{s_{h+1}} P(s_{h+1}|s_h, a_h) \langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle \\
&= \mathbb{E}^\pi \left[R(s_h, a_h) + \rho_{h+1} \|\theta_{h+1}^*\|_2 + \langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle \middle| s_h, a_h \right] \\
&= \mathbb{E}^\pi \left[R(s_h, a_h) + \rho_{h+1} \cdot \Theta + \langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle \middle| s_h, a_h \right].
\end{aligned}$$

Since

$$Q_h^*(s_h, a_h) = \langle \varphi(s_h, a_h), \theta_h^* \rangle = \langle \varphi(s_h), \theta_h^* \rangle + \langle a_h, \theta_h^* \rangle,$$

if we take the expectation over s_h and a_h, we will have

$$\mathbb{E}^\pi \left[\langle \varphi(s_h), \theta_h^* \rangle + \langle a_h, \theta_h^* \rangle \right] = \mathbb{E}^\pi \left[R(s_h, a_h) + \rho(s_{h+1}) \cdot \Theta + \langle \varphi(s_{h+1}), \theta_{h+1}^* \rangle \right].$$

Telescoping this equation from $h = 1$ to $h = H$, we will get (9). ■

The following lemma indicates that if we have accurate enough estimation on the $\theta_1, \cdots, \theta_H$, then the greedy policy is a nearly optimal policy.

Lemma 11 *Suppose we have estimation $\hat{\theta}_1, \cdots, \hat{\theta}_H$, which satisfy*

$$\left\| \hat{\theta}_h - \theta_h^* \right\|_2 \leq \varepsilon_h, \quad \forall 1 \leq h \leq H.$$

If consider policy π to be the greedy policy with respect to $\hat{\theta}_1, \cdots, \hat{\theta}_H$, then we have, e.g.

$$\pi(s_h) = \arg\max_{a_h \in \mathcal{A}(s_h)} \left\langle \varphi(s_h, a_h), \hat{\theta}_h \right\rangle,$$

then we have

$$V_1^*(s_1) - V_1^\pi(s_1) \leq 2 \sum_{h=1}^H \mathbb{E}[\rho(s_h)] \varepsilon_h.$$

Proof We use $\pi^*(s)$ to denote the action of s in the best policy (which is deterministic). Then we have

$$V_1^*(s_1) = Q_1^*(s_1, \pi^*(s_1)) = \langle \varphi(s_1), \theta_1^* \rangle + \langle \pi^*(s_1), \theta_1^* \rangle.$$

We further have

$$\begin{aligned}
\langle \pi^*(s_1), \theta_1^* \rangle &= \left\langle \pi^*(s_1), \theta_1^* - \hat{\theta}_1 \right\rangle + \left\langle \pi^*(s_1), \hat{\theta}_1 \right\rangle \leq \rho(s_1) \cdot \left\| \theta_1^* - \hat{\theta}_1 \right\|_2 + \left\langle \pi^*(s_1), \hat{\theta}_1 \right\rangle \\
&\leq \rho(s_1) \cdot \left\| \theta_1^* - \hat{\theta}_1 \right\|_2 + \left\langle \pi(s_1), \hat{\theta}_1 \right\rangle = \rho(s_1) \cdot \left\| \theta_1^* - \hat{\theta}_1 \right\|_2 + \langle \pi(s_1), \theta_1^* \rangle + \left\langle \pi(s_1), \hat{\theta}_1 - \theta_1^* \right\rangle \\
&\leq 2\rho(s_1) \cdot \left\| \theta_1^* - \hat{\theta}_1 \right\|_2 + \langle \pi(s_1), \theta_1^* \rangle,
\end{aligned}$$

where the first and last inequality are due to $\pi(s_1), \pi^*(s_1) \in \mathcal{A}(s_1)$, and the second inequality is due to the definition of $\pi(s_1)$ (greedy policy $w.r.t\ \hat{\theta}_1$). Therefore, we obtain that

$$\begin{aligned}
V_1^*(s_1) - V^\pi(s_1) &\le 2\rho(s_1) \cdot \left\|\theta_1^* - \hat{\theta}_1\right\|_2 + \langle \pi(s_1), \theta_1^* \rangle + \langle \varphi(s_1), \theta_1^* \rangle - V^\pi(s_1) \\
&= 2\rho(s_1) \cdot \left\|\theta_1^* - \hat{\theta}_1\right\|_2 + Q_1^*(s_1, \pi(s_1)) - V^\pi(s_1) \\
&= 2\rho(s_1) \cdot \left\|\theta_1^* - \hat{\theta}_1\right\|_2 + r(s_1, \pi(s_1)) + \mathbb{E}^\pi[V_2^*(s_2)] - (r(s_1, \pi(s_1)) + \mathbb{E}^\pi[V_2^\pi(s_2)]) \\
&= 2\rho(s_1) \cdot \left\|\theta_1^* - \hat{\theta}_1\right\|_2 + \mathbb{E}^\pi\left[V_2^*(s_2) - V_2^\pi(s_2)\right].
\end{aligned}$$

We can continue the same process for V_2, and obtain that

$$\mathbb{E}^\pi\left[V_2^*(s_2) - V_2^\pi(s_2)\right] \le 2\mathbb{E}^\pi[\rho(s_2)] \cdot \left\|\theta_2^* - \hat{\theta}_2\right\|_2 + \mathbb{E}^\pi\left[V_3^*(s_3) - V_3^\pi(s_3)\right].$$

Keeping doing this until we get $V_{H+1}^*(s_{H+1}) - V_{H+1}^\pi(s_{H+1})$, which is zero, we obtain that

$$V_1^*(s_1) - V_1^\pi(s_1) \le 2\sum_{h=1}^{H} \mathbb{E}[\rho(s_h)] \cdot \left\|\theta_h^* - \hat{\theta}_h\right\|_2.$$

Therefore, when $\left\|\theta_h^* - \hat{\theta}_h\right\|_2 \le \varepsilon_h$, we will have

$$V_1^*(s_1) - V_1^\pi(s_1) \le 2\sum_{h=1}^{H} \mathbb{E}[\rho(s_h)]\varepsilon_h$$

■

Proof [Proof of Theorem 9] We will divide our proof in the following parts:

Concentration Inequalities In the following, we fix all parameters within one 'while' loop in the algorithm.

According to the Boundedness Assumption (Assumption 4), and Hoeffding inequality, we have that each of the following inequalities holds with probability at least $1-\delta'$ separately for $0 \le i \le d$,

$$\begin{aligned}
\left| R_{h,i} - \mathbb{E}^{\pi_{h,i}}\left[\sum_{h'=1}^{H} R(s_{h'}, a_{h'})\right]\right| &\le \sqrt{\frac{2\log(1/\delta')}{M_1}} \\
\left| s_{h,i} - \mathbb{E}^{\pi_{h,i}}\left[\sum_{h'=1}^{H} \rho(s_{h'+1})\right]\right| &\le \sqrt{\frac{2\log(1/\delta')}{M_1}}.
\end{aligned} \tag{10}$$

We further notice that the before step h, policy $\pi_{h,i}$ and policy $\pi_{h,0}$ are both same as policy π, and for every $h' > h$ and $s_{h'} \in \mathcal{S}_{h'}$, we both have $\pi_{h,i}(s_{h'}) = \pi_{h,0}(s_{h'}) = 0$. Therefore, according to

(9), we have

$$\begin{aligned}\mathbb{E}^{\pi}[\rho(s_h)]\cdot\langle \mathbf{e}_i,\theta_h^*\rangle &= \mathbb{E}^{\pi_{h,i}}\left[\sum_{h'=1}^{H}\langle a_{h'},\theta_{h'}^*\rangle\right]-\mathbb{E}^{\pi_{h,0}}\left[\sum_{h'=1}^{H}\langle a_{h'},\theta_{h'}^*\rangle\right]\\ &= \mathbb{E}^{\pi_{h,i}}\left[\sum_{h'=1}^{H}R(s_{h'},a_{h'})\right]+\Theta\cdot\mathbb{E}^{\pi_{h,i}}\left[\sum_{h'=1}^{H}\rho(s_{h'+1})\right]\\ &\quad-\mathbb{E}^{\pi_{h,0}}\left[\sum_{h'=1}^{H}R(s_{h'},a_{h'})\right]-\Theta\cdot\mathbb{E}^{\pi_{h,0}}\left[\sum_{h'=1}^{H}\rho(s_{h'+1})\right].\end{aligned}\tag{11}$$

According to (10), if we further assume $|\Theta-\xi|\le\eta$, then with probability at least $1-2(d+1)\delta'$, we have for every $1\le i\le d$,

$$|\text{RHS of (11)}-((s_{h,i}-s_{h,0})\xi+R_{h,i}-R_{h,0})|\le\eta+\sqrt{\frac{2\log(1/\delta')}{M_1}}+\sqrt{\frac{2\log(1/\delta')}{M_1}}=\eta+2\sqrt{\frac{2\log(1/\delta')}{M_1}},$$

where we have used the fact that $|s_{h,i}-s_{h,0}|\le 1$ and also $|\Theta|\le 1$. Moreover, in the last 'while' loop, with probability at least $1-LH\delta$, according to Hoeffding inequality we have

$$\left|\rho_h^l-\mathbb{E}^{\pi_l'}[\rho(s_h)]\right|\le\sqrt{\frac{2\log(1/\delta')}{M_2}},\quad\forall 1\le l\le L,1\le h\le H.$$

Hence we have with probability at least $1-(2d+3)\delta'$,

$$|\rho_h\langle\mathbf{e}_i,\theta_h^*\rangle-((s_{h,i}-s_{h,0})\xi+R_{h,i}-R_{h,0})|\le\eta+2\sqrt{\frac{2\log(1/\delta')}{M_1}}+\sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

According to our algorithm, there exists one l such that $|l\eta-\Theta|\le\eta$, we write this l as l_0. Then with $\xi=l_0\eta$, we have

$$\hat{\theta}_{h,i}^{l_0}=\frac{(s_{h,i}-s_{h,0})\xi+R_{h,i}-R_{h,0}}{\rho_h}.$$

Therefore, we obtain that

$$\left|\rho_h\left(\langle\mathbf{e}_i,\theta_h^*\rangle-\hat{\theta}_{h,i}^{l_0}\right)\right|\le\eta+2\sqrt{\frac{2\log(1/\delta')}{M_1}}+\sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

According to our construction of $\hat{\theta}_h^{l_0}$, we have

$$\rho_h\left\|\theta_h-\hat{\theta}_h^{l_0}\right\|_2\le\eta d+2d\sqrt{\frac{2\log(1/\delta')}{M_1}}+d\sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

If in the next 'while' loop, the condition does not satisfy, then before the next loop, for every $1\le h\le H$ and $1\le l\le L$, we will have $\rho_h^l\le\varepsilon$ or $\rho_h^l\le\rho_h$. Hence for l_0, we also have either $\rho_h^{l_0}\le\varepsilon$ or $\rho_h^{l_0}\le\rho_h$ for $1\le h\le H$. With loss of generality, we can assume $\left\|\hat{\theta}_h^{l_0}\right\|_2\le 1$,

otherwise we can consider $\hat{\theta}_h^{l_0} \leftarrow \hat{\theta}_h^{l_0}/\|\hat{\theta}_h^{l_0}\|_2$, which will induce the same policy, and $\left\|\theta_h - \hat{\theta}_h^{l_0}\right\|_2$ will decrease under this operation. According to Hoeffding inequality and union bound, we have with probability at least $1 - HL\delta'$, for every $1 \le l \le L$ and $1 \le h \le H$,

$$\left|\rho_h^l - \mathbb{E}^{\pi_l'}(\rho(s_h))\right| \le \sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

Hence according to Lemma 11, we have

$$V_1^*(s_1) - V_1^{\pi_{l_0}'}(s_1) \le 2\sum_{h=1}^{H} \mathbb{E}^{\pi_{l_0}'}[\rho(s_h)] \left\|\theta_h - \hat{\theta}_h^{l_0}\right\|_2.$$

If we assume the above high-probability event all hold, then we have

$$\text{RHS} \le 4H\sqrt{\frac{2\log(1/\delta')}{M_2}} + 2\sum_{h=1}^{H} \rho_h^{l_0} \left\|\theta_h - \hat{\theta}_h^{l_0}\right\|_2,$$

where we use the fact that $\left\|\theta_h - \hat{\theta}_h^{l_0}\right\|_2 \le 2$ since $\|\theta_h\|_2 \le 1$ and $\|\hat{\theta}_h^{l_0}\|_2 \le 1$ both hold. According to the condition that we do not enter the next 'while' loop, the last term above can be upper bounded by

$$2H \cdot \left(\varepsilon + \eta d + 2d\sqrt{\frac{2\log(1/\delta')}{M_1}} + d\sqrt{\frac{2\log(1/\delta')}{M_2}}\right).$$

Therefore, we obtain that with probability at least $1 - d\delta' - 2HL\delta'$

$$V_1^*(s_1) - V_1^{\pi_{l_0}'}(s_1) \le 2H\varepsilon + 2H\eta d + 4H\sqrt{\frac{2\log(1/\delta')}{M_2}} + 4Hd\sqrt{\frac{2\log(1/\delta')}{M_1}} + 2Hd\sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

Further again according to Hoeffding inequality, we know that with probability at least $1 - HL\delta$, we have

$$\left|R_l - V_1^{\pi_{l_0}'}(s_0)\right| \le \sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

If we assume

$$l_1 = \arg\max_l R_l,$$

then we have

$$V_1^{\pi_{l_0}'}(s_0) - V_1^{\pi}(s_0) = V_1^{\pi_{l_0}'}(s_0) - V_1^{\pi_{l_1}'}(s_0) \le 2\sqrt{\frac{2\log(1/\delta')}{M_2}} + R_{l_0} - R_{l_1} \le 2\sqrt{\frac{2\log(1/\delta')}{M_2}}.$$

Therefore, we have proved that if the 'while' loop ends, then with probability at least $1 - d\delta' - 3HL\delta'$, the output policy π satisfies that

$$V_1^*(s_1) - V_1^{\pi}(s_1) \le 2H\varepsilon + 2H\eta d + (2 + 4H + 2Hd)\sqrt{\frac{2\log(1/\delta')}{M_2}} + 4Hd\sqrt{\frac{2\log(1/\delta')}{M_1}}.$$

Bound on Iterations in 'While' Loop What left in this proof is to show that the 'while' loop will terminate within some certain number of iterations.

We notice that starting from the third iterations of the 'while' loop, if we choose some h which satisfies the loop condition, then the value of ρ_h will at least be twice of the previous loop. And the value of ρ_h will be at least ε in order to enter the loop. Moreover, the value of ρ_h will never exceed 1 due to the boundedness assumption. Therefore, the loop will at most run

$$1 + H \log_2 \left(\frac{1}{\varepsilon} \right)$$

times.

Therefore, choosing $\varepsilon = \frac{\epsilon}{8H}$, $\delta' = \frac{\delta}{(d+3HL)(1+H\log_2(1/\varepsilon))}$, $\eta = \frac{\epsilon}{8Hd}$, $M_2 = 2\log(1/\delta') \cdot \frac{16(2+4H+2Hd)^2}{\epsilon^2}$ and $M_1 = 2\log(1/\delta') \cdot \frac{256H^2d^2}{\epsilon^2}$ and $L = \frac{1}{\eta}$, then our algorithm will output an ϵ-optimal policy using at most

$$(M_1 Hd + M_2 HL) \cdot \left(1 + H \log_2 \left(\frac{1}{\varepsilon} \right) \right) = \tilde{\mathcal{O}} \left(\frac{H^5 d^3}{\epsilon^3} \right)$$

number of samples. ■