On Reward-Free RL with Kernel and Neural Function Approximations: Single-Agent MDP and Markov Game

Shuang Qiu [1] **Jieping Ye** [1] **Zhaoran Wang** [2] **Zhuoran Yang** [3]

Abstract

To achieve sample efficiency in reinforcement learning (RL), it necessitates to efficiently explore the underlying environment. Under the offline setting, addressing the exploration challenge lies in collecting an offline dataset with sufficient coverage. Motivated by such a challenge, we study the reward-free RL problem, where an agent aims to thoroughly explore the environment without any pre-specified reward function. Then, given any extrinsic reward, the agent computes the optimal policy via offline RL with data collected in the exploration stage. Moreover, we tackle this problem under the context of function approximation, leveraging powerful function approximators. Specifically, we propose to explore via an optimistic variant of the value-iteration algorithm incorporating kernel and neural function approximations, where we adopt the associated exploration bonus as the exploration reward. Moreover, we design exploration and planning algorithms for both single-agent MDPs and zero-sum Markov games and prove that our methods can achieve $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity for generating a ε-suboptimal policy or ε-approximate Nash equilibrium when given an arbitrary extrinsic reward. To the best of our knowledge, we establish the first provably efficient reward-free RL algorithm with kernel and neural function approximators.

1. Introduction

While reinforcement learning (RL) with function approximations has achieved great empirical success (Mnih et al., 2015; Silver et al., 2016; 2017; Vinyals et al., 2019), its application is mostly enabled by massive interactions with the unknown environment, especially when the state space is large and function approximators such as neural networks are employed. To achieve sample efficiency, any RL algorithm needs to accurately learn the transition model either explicitly or implicitly, which brings the need of efficient exploration.

Under the setting of offline RL, the agent aims to learn the optimal policy only from an offline dataset collected a priori, without any interactions with the environment. Thus, the collected offline dataset should have sufficient coverage of the trajectory generated by the optimal policy. However, in real-world RL applications, the reward function is often designed by the learner based on the domain knowledge. The learner might have a set of reward functions to choose from or use an adaptive algorithm for reward design (Laud, 2004; Grzes, 2017). In such a scenario, it is often desirable to collect an offline dataset that covers all the possible optimal trajectories associated with a reward function. With such a benign offline dataset, for any arbitrary reward function, the RL agent has sufficient information to estimate the corresponding policy.

To study such a problem in a principled manner, we focus on the framework of reward-free RL, which consists of an exploration phase and a planning phase. Specifically, in the exploration phase, the agent interacts with the environment without accessing any pre-specified reward and collects empirical trajectories for the subsequent planning stage. During the planning phase, using the offline data collected in the exploration phase, the agent computes the optimal policy given an extrinsic reward function, without further interactions with the environment.

Recently, many works focus on designing provably sample-efficient reward-free RL algorithms. For the single-agent tabular case, Jin et al. (2020a); Kaufmann et al. (2020); Ménard et al. (2020); Zhang et al. (2020) achieve $\widetilde{\mathcal{O}}(\mathrm{poly}(H, |\mathcal{S}|, |\mathcal{A}|)/\varepsilon^2)$ sample complexity for obtaining ε-suboptimal policy, where $|\mathcal{S}|, |\mathcal{A}|$ are the sizes of state and action space, respectively. In view of the large action and state spaces, the works Zanette et al. (2020b); Wang et al. (2020a) attempt to theoretically analyze reward-free RL by applying the linear function approximation for the single-agent Markov decision process (MDP), which achieve $\widetilde{\mathcal{O}}(H^6\mathfrak{d}^3/\varepsilon^2)$ sample complexity (Wang et al., 2020a) with

[1]University of Michigan [2]Northwestern University [3]Princeton University. Correspondence to: Shuang Qiu <qiush@umich.edu>, Jieping Ye <jpye@umich.edu>, Zhaoran Wang <zhaoranwang@gmail.com>, Zhuoran Yang <zy6@princeton.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

$\mathfrak{d}$ denoting the dimension of the feature space. However, RL algorithms combined with nonlinear function approximators such as kernel and neural function approximators have shown great empirical successes in a variety of application problems (e.g., Duan et al. (2016); Silver et al. (2016; 2017); Wang et al. (2018); Vinyals et al. (2019)), thanks to their expressive power. On the other hand, although reward-free RL algorithms for the multi-player Markov games in the tabular case has been studied in Bai & Jin (2020); Liu et al. (2020), there still lack works theoretically studying multi-agent scenarios with the function approximation. Thus, the following question remains open:

Can we design provably efficient RL algorithms with kernel and neural function approximations for both single-agent MDPs and Markov games?

The main challenges of answering the above question lie in how to appropriately integrate nonlinear approximators into the framework of reward-free RL and how to incentivize the exploration by designing exploration rewards that fit such approximation. In this paper, we provide an affirmative answer to the above question by tackling these challenges, and our contributions are summarized as follows:

Contributions. In this paper, we first propose a provable sample and computationally efficient reward-free RL algorithm with *nonlinear* kernel and neural function approximations for the single-agent MDP setting. Our exploration algorithm is an optimistic variant of the least-squares value iteration algorithm, incorporating kernel and neural function approximators, which adopts the associated exploration bonus as the intrinsic reward. The proposed algorithm is shown to achieve $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity for generating an ε-suboptimal policy, given *arbitrary* extrinsic reward functions. Furthermore, we extend the proposed method for the single-agent setting to the zero-sum Markov game setting such that the algorithm can achieve $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity to generate a policy pair which is an ε-approximate Nash equilibrium. In the planning phase for Markov games, our proposed algorithm only involves finding the Nash equilibrium of matrix games formed by Q function, that can be solved efficiently, to generate policies, which is of an independent interest. The sample complexities of our methods match the $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ results in existing works for tabular or linear function approximation settings. To the best of our knowledge, we establish the first provably efficient reward-free RL algorithm with kernel and neural function approximators for both single-agent and multi-agent scenarios.

Related Work. There have been a lot of works focusing on designing provably efficient reward-free RL algorithms for both single-agent and multi-agent RL problems. For the single-agent scenario, Jin et al. (2020a) formalize the reward-free RL for the tabular setting and provide theoretical analysis for the proposed algorithm with an $\widetilde{\mathcal{O}}(\mathrm{poly}(H, |\mathcal{S}|, |\mathcal{A}|)/\varepsilon^2)$ sample complexity for achieving ε-suboptimal policy. The sample complexity for the tabular setting is further improved in several recent works (Kaufmann et al., 2020; Ménard et al., 2020; Zhang et al., 2020). Recently, Zanette et al. (2020b); Wang et al. (2020a) study the reward-free RL from the perspective of the linear function approximation. For the multi-agent setting, Bai & Jin (2020) studies the reward-free exploration for the zero-sum Markov game for the tabular case. Liu et al. (2020) further proposes provable reward-free RL algorithms for multi-player general-sum games.

Our work is also closely related to a line of works that study RL algorithms with function approximations. There are many works (Yang & Wang, 2019; 2020; Cai et al., 2019; Zanette et al., 2020a; Jin et al., 2020b; Wang et al., 2019; Ayoub et al., 2020; Zhou et al., 2020; Kakade et al., 2020) studying different RL problems with the (generalized) linear function approximation. Furthermore, Wang et al. (2020b) studies a optimistic LSVI algorithm for general function approximation. Our work is most closely related to the recent work (Yang et al., 2020), which studies optimistic LSVI algorithms with kernel and neural function approximation. However, this paper studies an online single-agent RL problem where the exploration is executed with reward feedbacks, which cannot be directly applied to the reward-free RL problem. Inspired by Yang et al. (2020), our work extends the idea of kernel and neural function approximations to the reward-free RL setting and the Markov games.

2. Preliminaries

In this section, we introduce the basic notations and problem backgrounds for this paper.

2.1. Markov Decision Process

Consider an episodic single-agent MDP defined by the tuple $(\mathcal{S}, \mathcal{A}, H, \mathbb{P}, r)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ is the action space of the agent, H is the length of each episode, $\mathbb{P} = \{\mathbb{P}_h\}_{h=1}^H$ is the transition model with $\mathbb{P}_h(s'|s,a)$ denoting the transition probability at the h-th step from the state $s \in \mathcal{S}$ to the state $s' \in \mathcal{S}$ when the agent takes action $a \in \mathcal{A}$, and $r_h : \mathcal{S} \times \mathcal{A} \mapsto [0,1]$ denotes the reward function at the h-step. Specifically, we assume that the true transition model $\mathbb{P}$ is *unknown* to the agent which necessitates the reward-free exploration. The policy of an agent is a collection of probability distributions $\pi = \{\pi_h\}_{h=1}^H$ where $\pi_h : \mathcal{S} \mapsto \Delta_{\mathcal{A}}$ with $\Delta_{\mathcal{A}}$ denoting a probability simplex defined on the space $\mathcal{A}$.

For a specific policy $\{\pi_h\}_{h\in[H]}$ and reward function $\{r_h\}_{h\in[H]}$, under the transition model $\{\mathbb{P}_h\}_{h\in[H]}$, we define the associated value function $V_h^\pi(s,r) : \mathcal{S} \mapsto \mathbb{R}$ at the h-th step as $V_h^\pi(s,r) := \mathbb{E}[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}) \,|\, s_h = s, \pi, \mathbb{P}]$. The corresponding action-value function (Q-

function) $Q_h^\pi : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is then defined. We have $Q_h^\pi(s,a,r) := \mathbb{E}[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}) \,|\, s_h = s, a_h = a, \pi, \mathbb{P}]$. Therefore, we have the Bellman equation as $V_h^\pi(s,r) = \langle Q_h^\pi(s,\cdot,r), \pi_h(\cdot|s)\rangle_{\mathcal{A}}$ and $Q_h^\pi(s,a,r) = r_h(s,a) + \langle \mathbb{P}_h(\cdot|s,a), V_{h+1}^\pi(\cdot,r)\rangle_{\mathcal{S}}$, where we let $\langle\cdot,\cdot\rangle_{\mathcal{S}}$, $\langle\cdot,\cdot\rangle_{\mathcal{A}}$ denote the inner product over the spaces $\mathcal{S}$, $\mathcal{A}$. The above Bellman equation holds for all $h \in [H]$ with setting $V_{H+1}^{\pi,\nu}(s) = 0, \forall s \in \mathcal{S}$. In the rest of this paper, for simplicity of the notation, we rewrite $\langle \mathbb{P}_h(\cdot|s,a), V_{h+1}(\cdot,r)\rangle_{\mathcal{S}} = \mathbb{P}_h V_{h+1}(s,a,r)$ for any transition probability $\mathbb{P}_h$ and value function $V(\cdot,r)$. Moreover, we denote π_r^* as the optimal policy w.r.t. r, such that π_r^* maximize $V_1^\pi(s_1,r)$[1]. Then, we have $Q_h^*(s,a,r) = Q_h^{\pi_r^*}(s,a,r)$ as well as $V_h^*(s,r) = V_h^{\pi_r^*}(s,r)$. We say $\widetilde{\pi}$ is ε-suboptimal policy if it satisfies

$$V_1^*(s_1,r) - V_1^{\widetilde{\pi}}(s_1,r) \le \varepsilon.$$

2.2. Zero-Sum Markov Game

In this paper, we also consider an episodic zero-sum Markov game characterized by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{B}, H, \mathbb{P}, r)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ and $\mathcal{B}$ are the action spaces for the two players, H is the length of each episode, $\mathbb{P} = \{\mathbb{P}_h\}_{h=1}^H$ is the transition model with $\mathbb{P}_h(s'|s,a,b)$ denoting the transition probability at the h-th step from the state s to the state s' when Player 1 takes action $a \in \mathcal{A}$ and Player 2 takes action $b \in \mathcal{B}$, $r_h : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \mapsto [0,1]$ denotes the reward function at the h-step. Similarly, we assume the transition model $\mathbb{P} = \{\mathbb{P}_h\}_{h=1}^H$ *unknown* to both players. The policy of Player 1 is a collection of probability distributions $\pi = \{\pi_h\}_{h=1}^H$ with $\pi : \mathcal{S} \mapsto \Delta_{\mathcal{A}}$. Analogously, the policy of Player 2 is a collection of probability distributions $\nu = \{\nu_h\}_{h=1}^H$ with $\nu : \mathcal{S} \mapsto \Delta_{\mathcal{B}}$. Here $\Delta_{\mathcal{A}}$ and $\Delta_{\mathcal{B}}$ are probability simplex defined on the spaces $\mathcal{A}$ and $\mathcal{B}$.

For a specific policy π and ν and reward function $\{r_h\}_{h\in[H]}$, under the transition model $\{\mathbb{P}_h\}_{h\in[H]}$, we define the value function $V_h^{\pi,\nu}(s,r) : \mathcal{S} \mapsto \mathbb{R}$ at the h-th step as $V_h^{\pi,\nu}(s,r) := \mathbb{E}[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}, b_{h'}) \,|\, s_h = s, \pi, \nu, \mathbb{P}]$. We further define the corresponding Q-function $Q_h^{\pi,\nu} : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \mapsto \mathbb{R}$, where we have $Q_h^{\pi,\nu}(s,a,b,r) := \mathbb{E}[\sum_{h'=h}^H r_{h'}(s_{h'}, a_{h'}, b_{h'}) \,|\, s_h = s, a_h = a, b_h = b, \pi, \mathbb{P}]$. Therefore, we have the Bellman equation as $V_h^{\pi,\nu}(s,r) = \mathbb{E}_{a\sim\pi_h, b\sim\nu_h}[Q_h^{\pi,\nu}(s,a,b,r)]$ and $Q_h^{\pi,\nu}(s,a,b,r) = r_h(s,a,b) + \mathbb{P}_h V_{h+1}^{\pi,\nu}(s,a,b,r)$, where, for simplicity, we also let $\mathbb{P}_h V_{h+1}^{\pi,\nu}(s,a,b,r) = \big\langle \mathbb{P}_h(\cdot|s,a,b), V_{h+1}^{\pi,\nu}(\cdot,r)\big\rangle_{\mathcal{S}}$.

We define the Nash Equilibrium (NE) $(\pi^\dagger, \nu^\dagger)$ as a solution to $\max_\pi \min_\nu V_1^{\pi,\nu}(s_1)$, where we have $V_1^{\pi^\dagger,\nu^\dagger}(s_1,r) = \max_\pi \min_\nu V_1^{\pi,\nu}(s_1) = \min_\nu \max_\pi V_1^{\pi,\nu}(s_1)$. For simplicity, we let $V_h^\dagger(s,r) = V_h^{\pi^\dagger,\nu^\dagger}(s,r)$ and also $Q_h^\dagger(s,a,b,r) = Q_h^{\pi^\dagger,\nu^\dagger}(s,a,b,r)$ denote the value function and Q function under the NE $\pi^\dagger, \nu^\dagger$ at h-th step. We further define the best response for Player 1 with policy π as $\mathrm{br}(\pi) := \operatorname{argmin}_\nu V_1^{\pi,\nu}(s_1,r)$ and the best response for Player 2 with policy ν as $\mathrm{br}(\nu) := \operatorname{argmax}_\pi V_1^{\pi,\nu}(s_1,r)$. Thus, we say $(\widetilde{\pi}, \widetilde{\nu})$ is ε-approximate NE if it satisfies

$$V_1^{\mathrm{br}(\widetilde{\nu}),\widetilde{\nu}}(s_1,r) - V_1^{\widetilde{\pi},\mathrm{br}(\widetilde{\pi})}(s_1,r) \le \varepsilon,$$

where $V_1^{\mathrm{br}(\widetilde{\nu}),\widetilde{\nu}}(s_1) \ge V_1^\dagger(s_1) \ge V_1^{\widetilde{\pi},\mathrm{br}(\widetilde{\pi})}(s_1)$ always holds. On the other hand, we let $V_1^*(s,r) = \max_{\pi,\nu} V_1^{\pi,\nu}(s,r)$, namely the maximal value function when $h = 1$. Then, we have the associated value function and Q function for the h-th step $V_h^*(s,r)$ and $Q_h^*(s,a,b,r)$.

2.3. Reproducing Kernel Hilbert Space

In our paper, we use reproducing kernel Hilbert space (RKHS) as the function space for approximation. Here, we abuse the notion a little by letting $\mathcal{Z} = \mathcal{S} \times \mathcal{A}$ for the single-agent MDP setting and $\mathcal{Z} = \mathcal{S} \times \mathcal{A} \times \mathcal{B}$ for the zero-sum game setting, such that $z = (s,a) \in \mathcal{Z}$ or $z = (s,a,b) \in \mathcal{Z}$ for different cases. We assume that the space $\mathcal{Z}$ is the input space of the function approximation, where $\mathcal{Z}$ is a compact space on $\mathbb{R}^d$. This can also be achieved if there is an preprocessing method to embed (s,a) or (s,a,b) into the space $\mathbb{R}^d$. We denote $\mathcal{H}$ as RKHS defined on the space $\mathcal{Z}$ with the kernel function $\ker : \mathcal{Z} \times \mathcal{Z} \mapsto \mathbb{R}$. We define the inner product on the RKHS $\mathcal{H}$ as $\langle\cdot,\cdot\rangle_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \mapsto \mathbb{R}$ and the norm $\|\cdot\| : \mathcal{H} \mapsto \mathbb{R}$. We have a feature map $\phi : \mathcal{Z} \mapsto \mathcal{H}$ on the RKHS $\mathcal{H}$ and define the function $f(z) := \langle f, \phi(z)\rangle_{\mathcal{H}}$ for $f \in \mathcal{H}$. Then the kernel is defined as

$$\ker(z,z') := \langle \phi(z), \phi(z')\rangle_{\mathcal{H}}, \forall z, z' \in \mathcal{Z}.$$

We assume that $\sup_{z\in\mathcal{Z}} \ker(z,z) \le 1$ such that $\|\phi(z)\|_{\mathcal{H}} \le 1$ for any $z \in \mathcal{Z}$.

2.4. Overparameterized Neural Network

This paper further considers the function approximator using the overparameterized neural network. Overparameterized neural network has drawn a lot of attention recently in both theory and practice (Neyshabur et al., 2018; Allen-Zhu et al., 2018; Arora et al., 2019; Gao et al., 2019; Bai & Lee, 2019). Specifically, in this work, we have a two-layer neural network $f(\cdot; b, W) : \mathcal{Z} \mapsto \mathbb{R}$ with $2m$ neurons and weights $(\boldsymbol{v}, W)$, which can be represented as

$$f(z; \boldsymbol{v}, W) = \frac{1}{\sqrt{2m}} \sum_{i=1}^{2m} v_i \cdot \mathtt{act}(W_i^\top z), \qquad (1)$$

where $\mathtt{act}$ is the activation function, and $\boldsymbol{v} = [v_1, \cdots, v_{2m}]^\top$ and $W = [W_1, W_2, \cdots, W_{2m}]$. Here, we assume that $z = (s,a)$ or $z = (s,a,b)$ with $z \in \mathcal{Z}$ satisfies

[1] Without loss of generality, in this paper, we assume the agent starts from a fixed state s_1 at $h = 1$. We also assume this for the Markov game setting.

$\|z\|_2 = 1$, i.e., z is normalized on a unit hypersphere in $\mathbb{R}^d$. Let $W^{(0)}$ be the initial value of W and $\boldsymbol{v}^{(0)}$ be the initialization of $\boldsymbol{v}$. The initialization step for the above model is performed as follows: we let $v_i \sim \text{Unif}(\{-1, 1\})$ and $W_i^{(0)} \sim N(0, I_d/d)$ for all $i \in [m]$, where I_d is an identity matrix in $\mathbb{R}^{d\times d}$, and $v_i^{(0)} = -v_{i-m}^{(0)}$, $W_i^{(0)} = W_{i-m}^{(0)}$ for all $i \in \{m+1, 2m\}$. Here we let $N(0, I_d/d)$ denote Gaussian distribution In this paper, we let $\boldsymbol{v}$ be fixed as $\boldsymbol{v}^{(0)}$ and we only learn W for the ease of theoretical analysis. Thus, we represent $f(z; , \boldsymbol{v}, W)$ by $f(z; W)$ to simplify the notation. This neural network model is widely studied in recent paper on the analysis of neural networks, e.g., Gao et al. (2019); Bai & Lee (2019). When the model is overparameterized, i.e., m is sufficiently large, we can characterized the dynamics of the training such neural network by neural tangent kernel (NTK) (Jacot et al., 2018). We define

$$\varphi(z; W) = [\nabla_{W_1} f(z; W)^\top, \cdots, \nabla_{W_{2m}} f(z; W)^\top]^\top, \quad (2)$$

where we let $\nabla_{W_i} f(z; W)$ be a column vector such that $\varphi(z; W) \in \mathbb{R}^{2md}$. Thus, when conditioned on the randomness in the initialization of W by $W^{(0)}$, we define the kernel

$$\ker_m(z, z') = \langle \varphi(z; W^{(0)}), \varphi(z'; W^{(0)}) \rangle, \forall z, z' \in \mathcal{Z}.$$

In addition, we consider a linearization of the model $f(z, W)$ at the initial value $W^{(0)}$, such that we have $f_{\texttt{lin}}(z; W) := f(z; W^{(0)}) + \langle \varphi(z; W^{(0)}), W - W^{(0)} \rangle$ and furthermore the following holds $f_{\texttt{lin}}(z; W) = \langle \varphi(z; W^{(0)}), W - W^{(0)} \rangle$, since $f(z; W^{(0)}) = 0$ by the initialization scheme. As we can see, the linearized function $f_{\texttt{lin}}(z; W)$ is a function on RKHS with the kernel $\ker_m(z, z')$. When the model is overparamterized with $m \to \infty$, the kernel $\ker_m(z, z')$ converges to a NTK kernel, which is defined as $\ker_{\texttt{ntk}} = \mathbb{E}_{\boldsymbol{\omega}\sim N(0, I_d/d)}[\texttt{act}'(\boldsymbol{\omega}^\top z) \cdot \texttt{act}'(\boldsymbol{\omega}^\top z') \cdot z^\top z']$, where $\texttt{act}'$ is the derivative of $\texttt{act}$.

3. Single-Agent MDP Setting

In this section, we introduce the algorithms under the single-agent MDP setting with kernel and neural function approximation. Then, we present their theoretical results.

3.1. Kernel Function Approximation

Our proposed method includes the reward-free exploration phase and planning phase with the given true reward function. The exploration phase and planning phase are summarized in Algorithm 1 and Algorithm 2.

Specifically, the exploration algorithm is an optimistic variant of the value-iteration algorithm with the function approximation. In Algorithm 1, we use Q_h^k and V_h^k to denote the optimistic Q-function and value function for the exploration rewards. During the exploration phase, the agent does not access the true reward function and explore the environment for K episodes based on the policy $\{\pi_h^k\}_{(h,k)\in[H]\times[K]}$ determined by the value function V_h^k, and collect the trajectories $\{s_h^k, a_h^k\}_{(h,k)\in[H]\times[K]}$ for the subsequent planning phase. Thus, instead of approximating the Q-function directly, we seek to approximate $\mathbb{P}_h V_{h+1}^k$ by a clipped kernel function $f_h^k(s, a)$ for any $(s, a) \in \mathcal{S} \times \mathcal{A}$, where $f_h^k(\cdot, \cdot)$ is estimated by solving a regularized kernel regression problem as below. Based on this kernel approximation, we construct an associated UCB bonus term u_h^k to facilitate exploration, whose form is specified by the kernel function approximator. Moreover, although the true reward is not available to the agent, to guide the exploration, we construct the exploration reward by scaling the bonus u_h^k, guiding the agent to explore state-action pairs with high uncertainties characterized by u_h^k. Then, the Q function Q_h^k is a combination of $r_h^k(s, a)$, $f_h^k(s, a)$, and $u_h^k(s, a)$ as shown in Line 8 of Algorithm 1. In this paper, we define a clipping operator $\Pi_{[0,H]}[x] := \min\{\max\{x, 0\}, H\}$. Note that the exploration phase in Algorithm 1 is a general framework that is not restricted to kernel cases and can be combined with other approximators.

Algorithm 1 Exploration Phase for Single-Agent MDP

1: **Initialize:** $\delta > 0$ and $\varepsilon > 0$.
2: **for** episode $k = 1, \ldots, K$ **do**
3: Let $V_{H+1}^k(\cdot) = \mathbf{0}$ and $Q_{H+1}^k(\cdot, \cdot) = \mathbf{0}$
4: **for** step $h = H, H-1, \ldots, 1$ **do**
5: Construct bonus term $u_h^k(\cdot, \cdot)$
6: Compute exploration reward $r_h^k(\cdot, \cdot) = u_h^k(\cdot, \cdot)/H$
7: Compute approximation function $f_h^k(\cdot, \cdot)$
8: $Q_h^k(\cdot, \cdot) = \Pi_{[0,H]}[(f_h^k + r_h^k + u_h^k)(\cdot, \cdot)]$
9: $V_h^k(\cdot) = \max_{a\in\mathcal{A}} Q_h^k(\cdot, a)$
10: $\pi_h^k(\cdot) = \operatorname{argmax}_{a\in\mathcal{A}} Q_h^k(\cdot, a)$
11: **end for**
12: Take actions following $a_h^k \sim \pi_h^k(s_h^k)$, $\forall h \in [H]$.
13: **end for**
14: **Return:** $\{(s_h^k, a_h^k)\}_{(h,k)\in[H]\times[K]}$.

At the k-th episode, given the visited trajectories $\{s_h^\tau, a_h^\tau\}_{\tau=1}^{k-1}$, we construct the approximation for each $h \in [H]$ by solving the following regularized nonlinear kernel regression problem

$$\widehat{f}_h^k = \min_{f\in\mathcal{H}} \sum_{\tau=1}^{k-1} [V_{h+1}^k(s_{h+1}^\tau) - f(z_h^\tau)]^2 + \lambda \|f\|_{\mathcal{H}}^2,$$

where $f(z_h^\tau) = \langle f, \phi(z_h^\tau) \rangle_{\mathcal{H}}$ with $z_h^\tau = (s_h^\tau, a_h^\tau)$, and λ is a hyperparameter to be determined later. As we will discuss in Lemma B.1 in the supplementary material, the closed form solution to the above problem is $\widehat{f}_h^k(z) = \langle \widehat{f}_h^k, \phi(z) \rangle_{\mathcal{H}} = \psi_h^k(z)^\top (\lambda \cdot I + \mathcal{K}_h^k)^{-1} \mathbf{y}_h^k$, where we define $\psi_h^k(z) := [\ker(z, z_h^1), \cdots, \ker(z, z_h^{k-1})]^\top$, $\mathbf{y}_h^k := [V_{h+1}^k(s_{h+1}^1), \cdots, V_{h+1}^k(s_{h+1}^{k-1})]^\top$, and also $\mathcal{K}_h^k := [\psi_h^k(z_h^1), \cdots, \psi_h^k(z_h^{k-1})]$ (recalling that $z = (s, a)$).

Algorithm 2 Planning Phase for Single-Agent MDP

Initialize: Reward function $\{r_h\}_{h\in[H]}$ and exploration data $\{(s_h^k, a_h^k)\}_{(h,k)\in[H]\times[K]}$
for step $h = H, H-1, \ldots, 1$ **do**
  Compute bonus term $u_h(\cdot,\cdot)$
  Compute approximation function $f_h(\cdot,\cdot)$
  $Q_h(\cdot,\cdot) = \Pi_{[0,H]}[(f_h + r_h + u_h)(\cdot,\cdot)]$
  $V_h(\cdot) = \max_{a\in\mathcal{A}} Q_h(\cdot, a)$
  $\pi_h(\cdot) = \operatorname{argmax}_{a\in\mathcal{A}} Q_h(\cdot, a)$
end for
Return: $\{\pi_h\}_{h\in[H]}$

We let $f_h^k(z) = \Pi_{[0,H]}[\widehat{f}_h^k(z)]$ by clipping operation to guarantee $f_h^k(z) \in [0, H]$ such that in Algorithm 1, we let

$$f_h^k(z) = \Pi_{[0,H]}[\psi_h^k(z)^\top(\lambda\cdot I + \mathcal{K}_h^k)^{-1}\mathbf{y}_h^k], \quad (3)$$

In addition, the associated bonus term is defined as

$$u_h^k(z) := \min\{\beta\cdot w_h^k(z), H\} \quad (4)$$

where β is a hyperparameter to be determined and we set

$$w_h^k(z) = \lambda^{-\frac{1}{2}}[\ker(z,z) - \psi_h^k(z)^\top(\lambda I + \mathcal{K}_h^k)^{-1}\psi_h^k(z)]^{\frac{1}{2}}.$$

The planning phase can be viewed as a single-episode version of optimistic value iteration algorithm. Using all the collected trajectories $\{s_h^k, a_h^k\}_{(h,k)\in[H]\times[K]}$, we can similarly construct the approximation of $\mathbb{P}_h V_{h+1}$ by solving

$$\widehat{f}_h = \operatorname*{argmin}_{f\in\mathcal{H}} \sum_{\tau=1}^{K}[V_{h+1}(s_{h+1}^\tau) - f(z_h^\tau)]^2 + \lambda\|f\|_{\mathcal{H}}^2. \quad (5)$$

Thus, the kernel approximation function can be estimated as

$$f_h(z) = \Pi_{[0,H]}[\widehat{f}_h(z)] = \Pi_{[0,H]}[\psi_h(z)^\top(\lambda\cdot I + \mathcal{K}_h)^{-1}\mathbf{y}_h],$$

and the bonus term is

$$u_h(z) := \min\{\beta\cdot w_h(z), H\}$$

with setting

$$w_h(z) = \lambda^{-\frac{1}{2}}[\ker(z,z) - \psi_h(z)^\top(\lambda I + \mathcal{K}_h)^{-1}\psi_h(z)]^{\frac{1}{2}},$$

where we define $\psi_h(z) := [\ker(z, z_h^1), \cdots, \ker(z, z_h^K)]^\top$, $\mathbf{y}_h := [V_{h+1}(s_{h+1}^1), \cdots, V_{h+1}(s_{h+1}^K)]^\top$, and also $\mathcal{K}_h := [\psi_h(z_h^1), \cdots, \psi_h(z_h^K)]$. Given all the true reward function r_h, with the kernel approximation f_h and the bonus u_h, one can compute the optimistic Q-function Q_h and the associated value function V_h. The learned policy π_h is obtained by value iteration based on the optimistic Q-function. Algorithm 2 is also a general planning scheme that can be generalized to any other function approximator, for example, the neural function approximator.

Remark 3.1. *Note that in the kernel function approximation setting, we directly define the kernel* $\ker(z, z')$ *for the algorithms instead of the feature map* $\phi(z)$ *which potential lies in an infinite dimensional space.*

3.2. Neural Function Approximation

For the neural function approximation setting, the agent also runs Algorithm 1 for exploration and Algorithm 2 for planning. Different from the kernel function approximation, in the exploration phase, at the k-th episode, given the visitation history $\{s_h^\tau, a_h^\tau\}_{\tau=1}^{k-1}$, we construct the approximation for each $h \in [H]$ by solving the following regularized regression problem, i.e., W_h^k is the global minimizer of

$$\min_{W\in\mathbb{R}^{2md}} \sum_{\tau=1}^{k-1}[V_{h+1}^k - f(z_h^\tau; W)]^2 + \lambda\|W - W^{(0)}\|_F^2, \quad (6)$$

where we assume that there exists an optimization oracle that can return the global optimizer of the above problem. The initialization of $W^{(0)}$ and $\boldsymbol{v}^{(0)}$ for the function $f(z; W)$ follows the schemes as we discussed in Section 2.4. As shown in many recent works (Du et al., 2019; 2018; Arora et al., 2019), when m is sufficiently large, with random initialization, some common optimizers, e.g., gradient descent, can find the global minimizer of the empirical loss efficiently at a linear convergence rate. Once we obtain W_h^k, the approximation function is constructed as $f_h^k(z) = \Pi_{[0,H]}[f(z; W_h^k)]$. The related exploration bonus u_h^k is of the form $u_h^k(z) := \min\{\beta\cdot w_h^k(z), H\}$ where

$$w_h^k(z) = [\varphi(z; W_h^k)^\top(\Lambda_h^k)^{-1}\varphi(z; W_h^k)]^{\frac{1}{2}}, \quad (7)$$

where we define the invertible matrix $\Lambda_h^k = \lambda I_{2md} + \sum_{\tau=1}^{k-1}\varphi(z_h^\tau; W_h^k)\varphi(z_h^\tau; W_h^k)^\top$ with $\varphi(z_h^\tau; W)$ as (2).

In the planning stage, given the collection of trajectories in K episodes of exploration phase, we construct the neural approximation of $\mathbb{P}_h V_{h+1}(z)$ as solving a least square problem, i.e., W_h is the global optimizer of

$$\min_{W\in\mathbb{R}^{2md}} \sum_{\tau=1}^{K}[V_{h+1} - f(z_h^\tau; W)]^2 + \lambda\|W - W^{(0)}\|_F^2, \quad (8)$$

such that $f_h(z) = \Pi_{[0,H]}[f(z; W_h)]$. Analogously, the bonus term for the planning phase is of the form $u_h(z) := \min\{\beta\cdot w_h(z), H\}$ where

$$w_h(z) = [\varphi(z; W_h)^\top(\Lambda_h)^{-1}\varphi(z; W_h)]^{\frac{1}{2}}, \quad (9)$$

where we define the invertible matrix $\Lambda_h := \lambda I_{2md} + \sum_{\tau=1}^{K}\varphi(z_h^\tau; W_h)\varphi(z_h^\tau; W_h)^\top$.

3.3. Theoretical Results for Single-Agent MDP

Kernel Function Setting. In this subsection, we first present the result for the kernel function approximation setting. We make the following assumptions.

Assumption 3.2. *We assume that for any value function* $V : \mathcal{S} \mapsto \mathbb{R}$, *we have* $\mathbb{P}_h V(z)$ *is in a form of* $\langle\phi(z), \mathbf{w}_h\rangle_{\mathcal{H}}$ *for some* $\mathbf{w}_h \in \mathcal{H}$. *In addition, we assume there exists a fixed constant* R_Q *such that* $\|\mathbf{w}_h\|_{\mathcal{H}} \le R_Q H$.

One example for this assumption is that the transition model is in a form of $\mathbb{P}_h(s'|z) = \langle\phi(z), \mathbf{w}'_h(s')\rangle_{\mathcal{H}}$ such that $\mathbb{P}_h V(z) = \int_{\mathcal{S}} V_{h+1}(s')\langle\phi(z), \mathbf{w}'_h(s')\rangle_{\mathcal{H}} \mathrm{d}s'$ where we can write $\mathbf{w}_h = \int_{\mathcal{S}} V_{h+1}(s')\mathbf{w}'_h(s')\mathrm{d}s'$. This example can be viewed as an generalization of the linear transition model (Jin et al., 2020b) to the RKHS.

In our work, we use maximal information gain (Srinivas et al., 2009) to measure the function space complexity, i.e.,

$$\Gamma(\mathfrak{C}, \sigma; \mathrm{ker}) = \sup_{\mathcal{D}\subseteq\mathcal{Z}} 1/2 \cdot \log\det(I + \mathcal{K}_{\mathcal{D}}/\sigma),$$

where the supremum is taken over all $\mathcal{D} \subseteq \mathcal{Z}$ with $|\mathcal{D}| \leq \mathfrak{C}$, and $\mathcal{K}_{\mathcal{D}}$ is the Gram matrix induced by $\mathcal{D}$ based on some kernel ker of RKHS. The value of $\Gamma(\mathfrak{C}, \sigma; \mathrm{ker})$ reflects how fast the the eigenvalues of $\mathcal{H}$ decay to zero and can be viewed as a proxy of the dimension of $\mathcal{H}$ when $\mathcal{H}$ is infinite-dimensional. To characterize the results in our paper, we define a specific Q-function class $\overline{\mathcal{Q}}$ of the form

$$\overline{\mathcal{Q}}(c, R, B) = \{Q : Q \text{ satisfies the form of } Q^\sharp\}. \quad (10)$$

where we define $Q^\sharp$ in the following form $Q^\sharp(z) = \min\{c(z) + \Pi_{[0,H]}[\langle\mathbf{w}, \phi(z)\rangle] + g(z), H\}^+$ with some $\mathbf{w}$ satisfying $\|\mathbf{w}\|_{\mathcal{H}} \leq R$ and also $g(z) = B \cdot \min\{\|\phi(z)\|_{\Lambda_{\mathcal{D}}^{-1}}, H/\beta\}^+$. Here $\Lambda_{\mathcal{D}}$ is an adjoint operator with the form $\Lambda_{\mathcal{D}} = \lambda I_{\mathcal{H}} + \sum_{z'\in\mathcal{D}} \phi(z')\phi(z')^\top$ with $I_{\mathcal{H}}$ denoting identity mapping on $\mathcal{H}$ and $\mathcal{D} \subseteq \mathcal{Z}$ with $|\mathcal{D}| \leq K$. Here we define the ς-covering number of the class $\overline{\mathcal{Q}}$ w.r.t. the ℓ_∞-norm as $\overline{\mathcal{N}}_\infty(\varsigma; R, B)$ with a upper bound $\mathcal{N}_\infty(\varsigma; R, B)$. As formally discussed in Section A of the supplementary material, we compute the covering number upper bound $\mathcal{N}_\infty(\varsigma; R, B)$. As we can see in Algorithms 1 and 2, we have $Q_h^k \in \overline{\mathcal{Q}}(\mathbf{0}, R, (1 + 1/H)\beta)$ and $Q_h \in \overline{\mathcal{Q}}(r_h, R', \beta)$ for some R and R'. Based on the above assumptions and definitions, we have the following result.

Theorem 3.3. *Suppose that β satisfies the condition that $16H^2\big[R_Q^2 + \log\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta) + 2\Gamma(K, \lambda; \mathrm{ker}) + 6\log(2KH) + 5\big] \leq \beta^2$. Under the kernel function approximation setting with a kernel ker, letting $\lambda = 1 + 1/K$, $R_K = 2H\sqrt{\Gamma(K, \lambda; \mathrm{ker})}$, and $\varsigma^* = H/K$, with probability at least $1 - (2K^2H^2)^{-1}$, the policy generated via Algorithm 2 satisfies $V_1^*(s_1, r) - V_1^\pi(s_1, r) \leq \mathcal{O}(\beta\sqrt{H^4[\Gamma(K, \lambda; \mathrm{ker}) + \log(KH)]}/\sqrt{K})$, after K episodes of exploration with Algorithm 1.*

The covering number $\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta)$ and the information gain $\Gamma(K, \lambda; \mathrm{ker})$ reflect the function class complexity. To understand the result in Theorem 3.3, we consider kernels ker with two different types of eigenvalue decay conditions: (i) γ-finite spectrum and (ii) γ-exponential spectral decay.

For the case of γ-finite spectrum with $\gamma \in \mathbb{Z}_+$, we have $\beta = \mathcal{O}(\gamma H\sqrt{\log(\gamma KH)})$, $\log\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta) = \mathcal{O}(\gamma^2\log(\gamma KH))$, and $\Gamma(K, \lambda; \mathrm{ker}) = \mathcal{O}(\gamma\log K)$, which further implies that to achieve $V_1^*(s_1, r) - V_1^\pi(s_1, r) \leq \varepsilon$, it requires $\widetilde{\mathcal{O}}(H^6\gamma^3/\varepsilon^2)$ rounds of exploration, where $\widetilde{\mathcal{O}}$ hides the logarithmic dependence on γ and $1/\varepsilon$.

Therefore, when the problem reduces to the setting of linear function approximation, the above result becomes $\widetilde{\mathcal{O}}(H^6\mathfrak{d}^3/\varepsilon^2)$ by letting $\gamma = \mathfrak{d}$, where $\mathfrak{d}$ is the feature dimension. This is consistent with the result in Wang et al. (2020a), which studies the linear approximation setting for reward-free RL. Furthermore, the sample complexity becomes $\widetilde{\mathcal{O}}(H^6|\mathcal{S}|^3|\mathcal{A}|^3/\varepsilon^2)$ by setting $\gamma = |\mathcal{S}||\mathcal{A}|$, when the problem reduces to the tabular setting.

For the case of γ-exponential spectral decay with $\gamma > 0$, we have $\log\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta) = \mathcal{O}((\log K)^{1+2/\gamma} + (\log\log H)^{1+2/\gamma})$, $\beta = \mathcal{O}(H\sqrt{\log(KH)}(\log K)^{1/\gamma})$, and also $\Gamma(K, \lambda; \mathrm{ker}) = \mathcal{O}((\log K)^{1+1/\gamma})$. Therefore, to obtain an ε-suboptimal policy, it requires $\mathcal{O}(H^6C_\gamma \cdot \log^{4+6/\gamma}(\varepsilon^{-1})/\varepsilon^2) = \widetilde{\mathcal{O}}(H^6C_\gamma/\varepsilon^2)$ rounds of exploration, where C_γ is some constant depending on $1/\gamma$. Please see Section A for detailed definitions and discussions.

Neural Function Setting Next, we present the result for the neural function approximation setting.

Assumption 3.4. *We assume that for any value function V, we have $\mathbb{P}_h V(z)$ can be represented by $\mathbb{P}_h V(z) = \int_{\mathbb{R}^d} \mathtt{act}'(\boldsymbol{\omega}^\top z) \cdot z^\top\boldsymbol{\alpha}_h(\boldsymbol{\omega})\mathrm{d}p_0(\boldsymbol{\omega})$ for some $\boldsymbol{\alpha}_h(\boldsymbol{\omega})$ with $\boldsymbol{\alpha} : \mathbb{R}^d \mapsto \mathbb{R}^d$ and $\sup_{\boldsymbol{\omega}} \|\boldsymbol{\alpha}(\boldsymbol{\omega})\| \leq R_Q H/\sqrt{d}$. Here p_0 is the density of Gaussian distribution $N(0, I_d/d)$.*

As discussed in Gao et al. (2019); Yang et al. (2020), the function class characterized by $f(z) = \int_{\mathbb{R}^d} \mathtt{act}'(\boldsymbol{\omega}^\top z) \cdot z^\top\boldsymbol{\alpha}_h(\boldsymbol{\omega})\mathrm{d}p_0(\boldsymbol{\omega})$ is an expressive subset of RKHS $\mathcal{H}$. One example is that the transition model can be written as $\mathbb{P}_h(s'|z) = \int_{\mathbb{R}^d} \mathtt{act}'(\boldsymbol{\omega}^\top z) \cdot z^\top\boldsymbol{\alpha}'_h(\boldsymbol{\omega}; s')\mathrm{d}p_0(\boldsymbol{\omega})$ such that we have $\boldsymbol{\alpha}_h(\boldsymbol{\omega}) = \int_{\mathcal{S}} \boldsymbol{\alpha}'_h(\boldsymbol{\omega}; s')V_{h+1}(s')\mathrm{d}s'$. This example generalizes the linear transition model (Jin et al., 2020b) to the overparameterized neural network setting. For analysis, we define a special Q function based on $f_{\mathtt{lin}}(z; W)$, and thus we also use the notations $\overline{\mathcal{Q}}$ and $\mathcal{N}_\infty$ w.r.t. $\varphi(z, W^{(0)})$ and $\mathrm{ker}_m(z)$ (See Lemma C.2 for details).

Theorem 3.5. *Suppose that β satisfies the condition that $8H^2[R_Q^2(1 + \sqrt{\lambda/d})^2 + 4\Gamma(K, \lambda; \mathrm{ker}_m) + 10 + 4\log\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta) + 12\log(2KH)] \leq \beta^2$ with $m = \Omega(K^{19}H^{14}\log^3 m)$. Under the overparameterized neural function approximation setting, letting $\lambda = C(1+1/K)$ for some constant $C \geq 1$, $R_K = H\sqrt{K}$, and $\varsigma^* = H/K$, with probability at least $1 - (2K^2H^2)^{-1} - 4m^{-2}$, the policy generated via Algorithm 2 satisfies $V_1^*(s_1, r) - V_1^\pi(s_1, r) \leq \mathcal{O}(\beta\sqrt{H^4[\Gamma(K, \lambda; \mathrm{ker}_m) + \log(KH)]}/\sqrt{K} + H^2\beta\iota)$ with $\iota = 5K^{7/12}H^{1/6}m^{-1/12}\log^{1/4} m$, after K episodes of exploration with Algorithm 1.*

In Theorem 3.5, there is an error term $H^2\beta\iota$ that depends on $m^{-1/12}$. In the regime of overparameterization, when m

is sufficiently large, this term can be extremely small and $\iota \to 0, \ker_m \to \ker_{\mathtt{ntk}}$ if $m \to \infty$. Here $\Gamma(K, \lambda; \ker_m)$ and $\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta)$ characterize the intrinsic complexity of the function class. In particular, when m is large, the overparamterized neural function setting can be viewed as a special case of RKHS with a misspecification error. If the eigenvalues of the kernel $\ker_m$ satisfy finite spectrum or exponential spectral decay, we know that β, $\Gamma(K, \lambda; \ker_m)$, and $\log \mathcal{N}_\infty(\varsigma^*; R_K, 2\beta)$ are of the same orders to the ones in the discussion after Theorem 3.3. Moreover, if m is sufficiently large such that $H^2\beta\iota \le \varepsilon$, we similarly have an $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity to achieve an $O(\varepsilon)$-suboptimal policy.

Overall, the above results show that with the kernel function approximation and overparameterized neural function approximation, Algorithms 3 and 4 guarantee $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity for achieving ε-suboptimal policy, which matches existing $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ results for the single-agent MDP for the tabular case or with linear function approximation.

4. Zero-Sum Game Setting

In this section, we introduce the algorithms under the Markov game setting with kernel and neural function approximation. We further present their theoretical results on the sample complexity.

4.1. Kernel Function Approximation

The exploration phase and planning phase for the zero-sum game are summarized in Algorithm 3 and Algorithm 4.

Specifically, in the exploration phase, the exploration policy for both players is obtained by taking maximum on Q-function over both action space. Thus, Algorithm 3 in essence is an extension of Algorithm 1 and performs the same exploration steps, if we view the pair (a, b) as a single action $\boldsymbol{a} = (a, b)$ on the action space $\mathcal{A} \times \mathcal{B}$ and regard the exploration policy pair $(\pi_h^k(s), \nu_h^k(s))$ as a product policy $(\pi_h^k \otimes \nu_h^k)(s)$. Thus, the approximator $f_h^k(z)$ and the bonus term $u_h^k(z)$ share the same forms as (3) and (4) if we slightly abuse the notation by letting $z = (s, a, b)$.

In the planning phase, the algorithm generates the policies for two players in a separate manner. While maintaining two Q-functions, their policies are generated by solving NE of two games with payoff matrices $\overline{Q}$ and $\underline{Q}$ respectively, namely $(\pi_h(s), \overline{D}_0(s))$ is the solution to $\max_{\pi'} \min_{\nu'} \mathbb{E}_{a,\sim\pi',b\sim\nu'}[\overline{Q}_h(s, a, b)]$ and $(\underline{D}_0(s), \nu_h(s))$ is the solution to $\max_{\pi'} \min_{\nu'} \mathbb{E}_{a,\sim\pi',b\sim\nu'}[\underline{Q}_h(s, a, b)]$, which can be solved efficiently in computation by many existing algorithms (e.g., Koller et al. (1994)).

Moreover, we construct the approximation function for Player 1 and Player 2 similarly via (5) by letting $z = (s, a, b)$ and placing the value function with $\overline{V}$ and $\underline{V}$ separately such

Algorithm 3 Exploration Phase for Zero-Sum Game

Initialize: $\delta > 0$ and $\varepsilon > 0$.
for episode $k = 1, \ldots, K$ **do**
  Let $V_{H+1}^k(\cdot) = \mathbf{0}$ and $Q_{H+1}^k(\cdot, \cdot, \cdot) = \mathbf{0}$
  for step $h = H, H-1, \ldots, 1$ **do**
    Construct bonus term $u_h^k(\cdot, \cdot, \cdot)$
    Exploration reward $r_h^k(\cdot, \cdot, \cdot) = u_h^k(\cdot, \cdot, \cdot)/H$
    Compute approximation function $f_h^k(\cdot, \cdot, \cdot)$
    $Q_h^k(\cdot, \cdot, \cdot) = \min\{(f_h^k + r_h^k + u_h^k)(\cdot, \cdot, \cdot), H\}^+$
    $V_h^k(\cdot) = \max_{a\in\mathcal{A}, b\in\mathcal{B}} Q_h^k(\cdot, a, b)$
    $(\pi_h^k(\cdot), \nu_h^k(\cdot)) = \operatorname{argmax}_{a\in\mathcal{A}, b\in\mathcal{B}} Q_h^k(\cdot, a, b)$
  end for
  Take actions following $a_h^k \sim \pi_h^k(s_h^k)$ and also $u_h^k \sim \nu_h^k(s_h^k), \forall h \in [H]$
end for
Return: $\{(s_h^k, a_h^k, u_h^k)\}_{(h,k)\in[H]\times[K]}$

that we have

$$\overline{f}_h(s, a) = [\psi_h(s, a)^\top(\lambda \cdot I + \mathcal{K}_h)^{-1}\overline{\mathbf{y}}_h]^+,$$
$$\underline{f}_h(s, a) = [\psi_h(s, a)^\top(\lambda \cdot I + \mathcal{K}_h)^{-1}\underline{\mathbf{y}}_h]^+,$$

where $\overline{\mathbf{y}}_h := [\overline{V}_{h+1}(s_{h+1}^1), \cdots, \overline{V}_{h+1}(s_{h+1}^K)]^\top$ and $\underline{\mathbf{y}}_h := [\underline{V}_{h+1}(s_{h+1}^1), \cdots, \underline{V}_{h+1}(s_{h+1}^K)]^\top$. Then, for the bonus term, Players 1 and 2 share the one of the same form, i.e., $\overline{u}_h(z) = \underline{u}_h(z) = \min\{\beta \cdot w_h(z), H\}$ with

$$w_h(z) = \lambda^{-\frac{1}{2}}[\ker(z, z) - \psi_h(z)^\top(\lambda I + \mathcal{K}_h)^{-1}\psi_h(z)]^{\frac{1}{2}}.$$

4.2. Neural Function Approximation

The exploration and planning stages are performed by Algorithm 3 and 4. In the exploration stage, following the same discussion for the exploration algorithm with kernel function approximation, Algorithm 3 with the neural approximator is intrinsically the same as Algorithm 1. Thus, one can follow the same approaches to construct the neural function approximator $f_h^k(z) = \Pi_{[0,H]}[f(z; W_h^k)]$ and the bonus $u_h^k(z)$ as in (6) and (7) with only letting $z = (s, a, b)$.

For the planning phase, by letting $z = (s, a, b)$, we construct approximation functions separately for Player 1 and Player 2 via solving two regression problems, i.e., $\overline{W}_h$ and $\underline{W}_h$ are respectively the global optimizers of

$$\min_{W\in\mathbb{R}^{2md}} \sum_{\tau=1}^K [\overline{V}_{h+1} - f(z_h^\tau; W)]^2 + \lambda\|W - W^{(0)}\|_F^2,$$
$$\min_{W\in\mathbb{R}^{2md}} \sum_{\tau=1}^K [\underline{V}_{h+1} - f(z_h^\tau; W)]^2 + \lambda\|W - W^{(0)}\|_F^2,$$

such that we let $\overline{f}_h(z) = \Pi_{[0,H]}[f(z; \overline{W}_h)]$ and $\underline{f}_h(z) = \Pi_{[0,H]}[f(z; \underline{W}_h)]$. The bonus terms $\overline{u}_h$ and $\underline{u}_h$ for Players 1 and 2 are $\overline{u}_h(z) := \min\{\beta \cdot \overline{w}_h(z), H\}$ and $\underline{u}_h(z) :=$

Algorithm 4 Planning Phase for Zero-Sum Game

Initialize: Reward function $\{r_h\}_{h\in[H]}$ and exploration data $\{(s_h^k, a_h^k, u_h^k)\}_{(h,k)\in[H]\times[K]}$
for step $h = H, H-1, \ldots, 1$ **do**
  Compute bonus term $\overline{u}_h(\cdot,\cdot,\cdot)$ and $\underline{u}_h(\cdot,\cdot,\cdot)$.
  Compute approximations $\overline{f}_h(\cdot,\cdot,\cdot)$ and $\underline{f}_h(\cdot,\cdot,\cdot)$
  $\overline{Q}_h(\cdot,\cdot,\cdot) = \min\{(\overline{f}_h + r_h + \overline{u}_h)(\cdot,\cdot,\cdot), H\}^+$
  $\underline{Q}_h(\cdot,\cdot,\cdot) = \min\{(\underline{f}_h + r_h - \underline{u}_h)(\cdot,\cdot,\cdot), H\}^+$
  Let $(\pi_h(s), \overline{D}_0(s))$ be NE for $\overline{Q}_h(s,\cdot,\cdot), \forall s\in\mathcal{S}$
  Let $(\underline{D}_0(s), \nu_h(s))$ be NE for $\underline{Q}_h(s,\cdot,\cdot), \forall s\in\mathcal{S}$
  $\overline{V}_h(\cdot) = \mathbb{E}_{a\sim\pi_h(s), b\sim\overline{D}_0(s)}[\overline{Q}_h(\cdot,a,b)], \forall s\in\mathcal{S}$
  $\underline{V}_h(\cdot) = \mathbb{E}_{a\sim\underline{D}_0(s), b\sim\nu_h(s)}[\underline{Q}_h(s,a,b)], \forall s\in\mathcal{S}$
end for
Return: $\{\pi_h\}_{h\in[H]}, \{\nu_h\}_{h\in[H]}$

$\min\{\beta\cdot\underline{w}_h(z), H\}$ with

$$\overline{w}_h(z) = [\varphi(z;\overline{W}_h)^\top(\overline{\Lambda}_h)^{-1}\varphi(z;\overline{W}_h)]^{\frac{1}{2}},$$
$$\underline{w}_h(z) = [\varphi(z;\underline{W}_h)^\top(\underline{\Lambda}_h)^{-1}\varphi(z;\underline{W}_h)]^{\frac{1}{2}},$$

where we define the invertible matrices $\overline{\Lambda}_h := \lambda I_{2md} + \sum_{\tau=1}^K \varphi(z_h^\tau;\overline{W}_h)\varphi(z_h^\tau;\overline{W}_h)^\top$ and $\underline{\Lambda}_h := \lambda I_{2md} + \sum_{\tau=1}^K \varphi(z_h^\tau;\underline{W}_h)\varphi(z_h^\tau;\underline{W}_h)^\top$.

4.3. Theoretical Results for Zero-Sum Game

In this section, we present the results for zero-sum Markov game. Particularly, in the subsection, we make the same assumptions as in Section 3.3 with only letting $z = (s, a, b)$. Moreover, we also use the same Q-function class $\overline{\mathcal{Q}}$ as (10), such that we can see in Algorithms 3 and 4, $Q_h^k \in \overline{\mathcal{Q}}(\mathbf{0}, R, (1+1/H)\beta)$ for some R, and $\overline{Q}_h \in \overline{\mathcal{Q}}(r_h, R', \beta)$ for some R'. To characterize the space of $\underline{Q}_h$, we define a specific Q-function class $\underline{\mathcal{Q}}$ of the form

$$\underline{\mathcal{Q}}(c, R, B) = \{Q : Q \text{ satisfies the form of } Q^\flat\}, \quad (11)$$

where $Q^\flat(z) = \min\{c(z) + \Pi_{[0,H]}[\langle\mathbf{w}, \phi(z)\rangle] - g(z), H\}^+$ for some $\mathbf{w}$ satisfying $\|\mathbf{w}\|_{\mathcal{H}} \le R$ and also $g(z) = B\cdot\max\{\|\phi(z)\|_{\Lambda_{\mathcal{D}}^{-1}}, H/\beta\}^+$. Thus, we have $\underline{Q}_h \in \underline{\mathcal{Q}}(r_h, R', \beta)$. As we show in Section A of the supplementary material, $\overline{\mathcal{Q}}(c, R, B)$ and $\underline{\mathcal{Q}}(c, R, B)$ have the same covering number upper bound w.r.t $\|\cdot\|_\infty$. Then, we can use the same notation $\mathcal{N}_\infty$ to denote such upper bound. Thus, we have the following result for kernel setting.

Theorem 4.1. *Suppose that β satisfies the condition that $16H^2\big[R_Q^2 + \log\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta) + 2\Gamma(K,\lambda;\ker) + 6\log(4KH) + 5\big] \le \beta^2$. Under the kernel function approximation setting with a kernel* ker*, letting $\lambda = 1 + 1/K$, $R_K = 2H\sqrt{\Gamma(K,\lambda;\ker)}$, and $\varsigma^* = H/K$, with probability at least $1-(2K^2H^2)^{-1}$, the policy pair generated via Algorithm 4 satisfies $V_1^{\mathrm{br}(\nu),\nu}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r) \le \mathcal{O}(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker) + \log(KH)]}/\sqrt{K})$, after K episodes of exploration with Algorithm 3.*

We further obtain the result for reward-free Markov game with the neural function approximation.

Theorem 4.2. *Suppose that β satisfies the condition that $8H^2[10 + 12\log(4K/\delta) + R_Q^2(1+\sqrt{\lambda/d})^2 + 4\log\mathcal{N}_\infty(\varsigma^*; R_K, 2\beta) + 4\Gamma(K,\lambda;\ker_m)] \le \beta^2$ with $m = \Omega(K^{19}H^{14}\log^3 m)$. Under the overparameterized neural function approximation setting, letting $\lambda = C(1+1/K)$ for some constant $C \ge 1$, $R_K = H\sqrt{K}$, and $\varsigma^* = H/K$, with probability at least $1-(2K^2H^2)^{-1} - 4m^{-2}$, the policy pair generated via Algorithm 4 satisfies $V_1^{\mathrm{br}(\nu),\nu}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r) \le \mathcal{O}(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker_m) + \log(KH)]}/\sqrt{K} + H^2\beta\iota)$ with $\iota = 5K^{7/12}H^{1/6}m^{-1/12}\log^{1/4} m$, after K episodes of exploration with Algorithm 3.*

Following the same discussion as in Section 3.3, the above results show that with the kernel function approximation and overparameterized neural function approximation, Algorithms 3 and 4 guarantee an $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity to achieve an ε-approximate NE. In particular, when our problem reduces to the Markov game with linear function approximation, the algorithm requires $\widetilde{\mathcal{O}}(H^6\mathfrak{d}^3/\varepsilon^2)$ sample complexity to achieve an ε-approximate NE, where $\mathfrak{d}$ is the feature dimension. For the tabular case, Bai & Jin (2020) gives an $\widetilde{\mathcal{O}}(H^5|\mathcal{S}|^2|\mathcal{A}||\mathcal{B}|)$ sample complexity and Liu et al. (2020) gives an $\widetilde{\mathcal{O}}(H^4|\mathcal{S}||\mathcal{A}||\mathcal{B}|)$ sample complexity. Our analysis gives an $\widetilde{\mathcal{O}}(H^6|\mathcal{S}|^3|\mathcal{A}|^3|\mathcal{B}|^3/\varepsilon)$ sample complexity by simply letting $\mathfrak{d} = |\mathcal{S}||\mathcal{A}||\mathcal{B}|$, which matches the existing results in terms of ε. Though the dependence on $H, |\mathcal{S}|, |\mathcal{A}|, |\mathcal{B}|$ is not tight as existing results, our work present a more general analysis for the function approximation settings which is not fully studied in previous works.

5. Theoretical Analysis

5.1. Proof Sketches of Theorem 3.3 and Theorem 3.5

We first show the proof sketches for Theorem 3.3. Our goal is to bound the term $V_1^*(s_1,r) - V_1^\pi(s_1,r)$. By optimistic updating rule in the planning phase, according to Lemma B.7, we have $V_1^*(s_1,r) \le V_1(s_1)$ such that $V_1^*(s_1,r) - V_1^\pi(s_1,r) \le V_1(s_1) - V_1^\pi(s_1,r)$. Then we only need to consider to upper bound $V_1(s_1) - V_1^\pi(s_1,r)$. Further by this lemma, for any $h\in[H]$, we have

$$\begin{aligned} &V_h(s) - V_h^\pi(s,r) \\ &\quad\le r_h(s,\pi_h(s)) + \mathbb{P}_h V_{h+1}(s,\pi_h(s)) \\ &\qquad + 2u_h(s,\pi_h(s)) - Q_h^\pi(s,\pi_h(s),r) \\ &\quad= \mathbb{P}_h V_{h+1}(s,\pi_h(s)) - \mathbb{P}_h V_{h+1}^\pi(s,\pi_h(s),r) \\ &\qquad + 2u_h(s,\pi_h(s)). \end{aligned} \quad (12)$$

where we use the fact that $Q_h^\pi(s,\pi_h(s),r) = r_h(s,\pi_h(s)) + \mathbb{P}_h V_{h+1}^\pi(s,\pi_h(s),r)$. Recursively applying the above in-

equality and also using $V_{H+1}^{\pi}(s,r) = V_{H+1}(s) = 0$ give

$$V_1(s_1) - V_1^{\pi}(s_1, r) \leq \mathbb{E}_{\mathbb{P}}[\sum_{h=1}^{H} 2u_h(s_h, \pi_h(s_h))|s_1] = 2H \cdot V_1^{\pi}(s_1, u/H).$$

Moreover, by Lemma B.8, we build a connection between the exploration and planing phase, which is $V_1^{\pi}(s_1, u/H) \leq K^{-1}\sum_{k=1}^{K} V_1^*(s_1, r^k)$. Therefore, combining the above results together, we eventually obtain

$$V_1^*(s_1,r) - V_1^{\pi}(s_1,r) \leq 2H/K \cdot \sum_{k=1}^{K} V_1^*(s_1, r^k) \leq \mathcal{O}\big(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker) + \log(KH)]}/\sqrt{K}\big),$$

where the last inequality is by Lemma B.5 and the fact that $\beta \geq H$. This completes the proof of Theorem 3.3. Please see detailed proof in Section B.2.

Next, we show the proof sketches of Theorem 3.5. By Lemma C.5, we have $V_1^*(s_1,r) \leq V_1(s_1) + H\beta\iota$ by optimism, such that $V_1^*(s_1,r) - V_1^{\pi}(s_1,r) \leq V_1(s_1) - V_1^{\pi}(s_1,r) + H\beta\iota$. Note that different from the proof of Theorem 3.3, there is an extra bias term $H\beta\iota$ introduced by the neural function approximation. Further by Lemma C.5, and using the same argument as (12), we have

$$V_h(s) - V_h^{\pi}(s,r) \leq 2u_h(s,\pi_h(s)) + \beta\iota + \mathbb{P}_h V_{h+1}(s,\pi_h(s)) - \mathbb{P}_h V_{h+1}^{\pi}(s,\pi_h(s),r),$$

which introducing another bias $\beta\iota$. Recursively applying the above inequality with $V_{H+1}^{\pi}(s,r) = V_{H+1}(s) = 0$ gives

$$V_1(s_1) - V_1^{\pi}(s_1,r) = 2H \cdot V_1^{\pi}(s_1, u/H) + H\beta\iota.$$

Thus, with Lemma C.6 connecting the exploration and planning such that $V_1^{\pi}(s_1,u/H) \leq K^{-1}\sum_{k=1}^{K} V_1^*(s_1,r^k) + 2\beta\iota$, combining all the above results eventually yields

$$V_1^*(s_1,r) - V_1^{\pi}(s_1,r) \leq 2H/K \cdot \sum_{k=1}^{K} V_1^*(s_1,r^k) + 4H\beta\iota \leq \mathcal{O}(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker_m) + \log(KH)]}/\sqrt{K} + H^2\beta\iota),$$

where the second inequality and the last inequality is by Lemma C.3 and the fact that $\beta \geq H$. This completes the proof. Please see detailed proof in Section C.2.

5.2. Proof Sketches of Theorem 4.1 and Theorems 4.2

For the proof sketch of Theorem 4.1, we decompose the $V_1^{\mathrm{br}(\nu),\nu}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r)$ into two terms $V_1^{\dagger}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r)$ and $V_1^{\mathrm{br}(\nu),\nu}(s_1,r) - V_1^{\dagger}(s_1,r)$ and bound them separately. We first bound the first term. By Lemma D.4, we have $V_1^{\dagger}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r) \leq \overline{V}_1(s_1) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r)$. Note that by the updating rule for $\overline{V}_h$ in Algorithm 4, we have

$$\overline{V}_h(s) = \min_{\nu'} \mathbb{E}_{a\sim\pi_h, b\sim\nu'}[\overline{Q}_h(s,a,b)] \leq \mathbb{E}_{a\sim\pi_h, b\sim\mathrm{br}(\pi)_h}[\overline{Q}_h(s,a,b)],$$

such that further by Lemma D.4, there is

$$\begin{aligned}&\overline{V}_h(s_h) - V_h^{\pi,\mathrm{br}(\pi)}(s_h,r)\\ &\leq \mathbb{E}[(\mathbb{P}_h\overline{V}_{h+1} + r_h + 2u_h)(s_h,a_h,b_h)] - V_h^{\pi,\mathrm{br}(\pi)}(s_h,r)\\ &= \mathbb{E}[\overline{V}_{h+1}(s_{h+1}) - V_{h+1}^{\pi,\mathrm{br}(\pi)}(s_{h+1},r) + 2u_h(s_h,a_h,b_h)].\end{aligned}$$

where $\mathbb{E}$ in the inequality is taken over $a_h \sim \pi_h, b_h \sim \mathrm{br}(\pi)_h$ and $\mathbb{E}$ in the equality is taken over $a_h \sim \pi_h, b_h \sim \mathrm{br}(\pi)_h, s_{h+1} \sim \mathbb{P}_h(\cdot|s_h,a_h,b_h)$. The equality above also uses $V_h^{\pi,\mathrm{br}(\pi)}(s_h,r) = \mathbb{E}_{a_h\sim\pi_h, b_h\sim\mathrm{br}(\pi)_h}[r_h(s_h,a_h,b_h) + \mathbb{P}_h V_{h+1}^{\pi,\mathrm{br}(\pi)}(s_h,a_h,b_h,r)]$. Recursively applying the above inequality yields

$$\begin{aligned}&\overline{V}_1(s_1) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r)\\ &\leq \mathbb{E}_{\pi,\mathrm{br}(\pi),\mathbb{P}}[\sum_{h=1}^{H} 2u_h(s_h,a_h,b_h)|s_1]\\ &= 2H \cdot V_1^{\pi,\mathrm{br}(\pi)}(s_1,u/H).\end{aligned}$$

Combining the above results eventually gives

$$\begin{aligned}&V_1^{\dagger}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r)\\ &\leq 2H \cdot V_1^{\pi,\mathrm{br}(\pi)}(s_1,u/H) \leq \frac{2H}{K}\sum_{k=1}^{K} V_1^*(s_1,r^k)\\ &\leq \mathcal{O}(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker) + \log(KH)]}/\sqrt{K}),\end{aligned}$$

where the second inequality is due to Lemma D.5 and the last inequality is by Lemma D.2. The upper bound of the term $V_1^{\dagger}(s_1,r) - V_1^{\pi,\mathrm{br}(\pi)}(s_1,r)$ is also $\mathcal{O}(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker) + \log(KH)]}/\sqrt{K})$ with the similar proof idea. This completes the proof of Theorem 4.1. Please see Section D.2 for details.

The proof of Theorem 4.2 follows the same argument as above. The only difference is that the neural function approximation brings bias terms depending on ι as we discussed in the proof sketch of Theorem 3.5. Thus, the final bound is $\mathcal{O}(\beta\sqrt{H^4[\Gamma(K,\lambda;\ker_m) + \log(KH)]}/\sqrt{K} + H^2\beta\iota)$. Please see Section E.2 for the detailed proof.

6. Conclusion

In this paper, we study the reward-free RL algorithms with kernel and neural function approximators for both single-agent MDPs and zero-sum Markov games. We prove that our methods can achieve $\widetilde{\mathcal{O}}(1/\varepsilon^2)$ sample complexity for generating an ε-suboptimal policy or ε-approximate NE.

Acknowledgements

Zhaoran Wang acknowledges National Science Foundation (Awards 2048075, 2008827, 2015568, 1934931), Simons Institute (Theory of Reinforcement Learning), Amazon, J.P. Morgan, and Two Sigma for their supports. Zhuoran Yang acknowledges Simons Institute (Theory of Reinforcement Learning).

References

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pp. 322–332. PMLR, 2019.

Ayoub, A., Jia, Z., Szepesvari, C., Wang, M., and Yang, L. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pp. 463–474. PMLR, 2020.

Bai, Y. and Jin, C. Provable self-play algorithms for competitive reinforcement learning. In *International Conference on Machine Learning*, pp. 551–560. PMLR, 2020.

Bai, Y. and Lee, J. D. Beyond linearization: On quadratic and higher-order approximation of wide neural networks. *arXiv preprint arXiv:1910.01619*, 2019.

Cai, Q., Yang, Z., Jin, C., and Wang, Z. Provably efficient exploration in policy optimization. *arXiv preprint arXiv:1912.05830*, 2019.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pp. 1675–1685. PMLR, 2019.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. PMLR, 2016.

Gao, R., Cai, T., Li, H., Wang, L., Hsieh, C.-J., and Lee, J. D. Convergence of adversarial training in overparametrized neural networks. *arXiv preprint arXiv:1906.07916*, 2019.

Grzes, M. Reward shaping in episodic reinforcement learning. 2017.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

Jin, C., Krishnamurthy, A., Simchowitz, M., and Yu, T. Reward-free exploration for reinforcement learning. In *International Conference on Machine Learning*, pp. 4870–4879. PMLR, 2020a.

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

Kakade, S., Krishnamurthy, A., Lowrey, K., Ohnishi, M., and Sun, W. Information theoretic regret bounds for online nonlinear control. *arXiv preprint arXiv:2006.12466*, 2020.

Kaufmann, E., Ménard, P., Domingues, O. D., Jonsson, A., Leurent, E., and Valko, M. Adaptive reward-free exploration. *arXiv preprint arXiv:2006.06294*, 2020.

Koller, D., Megiddo, N., and Von Stengel, B. Fast algorithms for finding randomized strategies in game trees. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pp. 750–759, 1994.

Laud, A. D. Theory and application of reward shaping in reinforcement learning. Technical report, 2004.

Liu, Q., Yu, T., Bai, Y., and Jin, C. A sharp analysis of model-based reinforcement learning with self-play. *arXiv preprint arXiv:2010.01604*, 2020.

Ménard, P., Domingues, O. D., Jonsson, A., Kaufmann, E., Leurent, E., and Valko, M. Fast active learning for pure exploration in reinforcement learning. *arXiv preprint arXiv:2007.13442*, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. Towards understanding the role of overparametrization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*, 2018.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Steinwart, I. and Christmann, A. *Support vector machines*. Springer Science & Business Media, 2008.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wang, R., Du, S. S., Yang, L. F., and Salakhutdinov, R. On reward-free reinforcement learning with linear function approximation. *arXiv preprint arXiv:2006.11274*, 2020a.

Wang, R., Salakhutdinov, R., and Yang, L. F. Provably efficient reinforcement learning with general value function approximation. *arXiv preprint arXiv:2005.10804*, 2020b.

Wang, W. Y., Li, J., and He, X. Deep reinforcement learning for nlp. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pp. 19–21, 2018.

Wang, Y., Wang, R., Du, S. S., and Krishnamurthy, A. Optimism in reinforcement learning with generalized linear function approximation. *arXiv preprint arXiv:1912.04136*, 2019.

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

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

Yang, Z., Jin, C., Wang, Z., Wang, M., and Jordan, M. Provably efficient reinforcement learning with kernel and neural function approximations. *Advances in Neural Information Processing Systems*, 33, 2020.

Zanette, A., Brandfonbrener, D., Brunskill, E., Pirotta, M., and Lazaric, A. Frequentist regret bounds for randomized least-squares value iteration. In *International Conference on Artificial Intelligence and Statistics*, pp. 1954–1964. PMLR, 2020a.

Zanette, A., Lazaric, A., Kochenderfer, M. J., and Brunskill, E. Provably efficient reward-agnostic navigation with linear value iteration. *arXiv preprint arXiv:2008.07737*, 2020b.

Zhang, Z., Du, S. S., and Ji, X. Nearly minimax optimal reward-free reinforcement learning. *arXiv preprint arXiv:2010.05901*, 2020.

Zhou, D., He, J., and Gu, Q. Provably efficient reinforcement learning for discounted mdps with feature mapping. *arXiv preprint arXiv:2006.13165*, 2020.