Journal of Machine Learning **ISSN**: 2790-2048(e), 2790-203X(p)

Reinforcement Learning with Function Approximation: From Linear to Nonlinear

Jihao Long * [1] and Jiequn Han † [2]

[1]Program of Applied and Computational Mathematics, Princeton University, Princeton, NJ 08544, USA
[2]Center for Computational Mathematics, Flatiron Institute, New York, NY 10010, USA

Abstract. Function approximation has been an indispensable component in modern reinforcement learning algorithms designed to tackle problems with large state spaces in high dimensions. This paper reviews recent results on error analysis for these reinforcement learning algorithms in linear or nonlinear approximation settings, emphasizing approximation error and estimation error/sample complexity. We discuss various properties related to approximation error and present concrete conditions on transition probability and reward function under which these properties hold true. Sample complexity analysis in reinforcement learning is more complicated than in supervised learning, primarily due to the distribution mismatch phenomenon. With assumptions on the linear structure of the problem, numerous algorithms in the literature achieve polynomial sample complexity with respect to the number of features, episode length, and accuracy, although the minimax rate has not been achieved yet. These results rely on the L^∞ and UCB estimation of estimation error, which can handle the distribution mismatch phenomenon. The problem and analysis become substantially more challenging in the setting of nonlinear function approximation, as both L^∞ and UCB estimation are inadequate for bounding the error with a favorable rate in high dimensions. We discuss additional assumptions necessary to address the distribution mismatch and derive meaningful results for nonlinear RL problems.

Keywords:
Reinforcement Learning,
Function Approximation,
High-Dimensionality Analysis,
Distribution Mismatch.

Article Info.:
Volume: 2
Number: 3
Pages: 161 - 193
Date: September/2023
doi.org/10.4208/jml.230105

Article History:
Received: 05/01/2023
Accepted: 17/05/2023

Communicated by:
Weinan E

1 Introduction

Reinforcement learning (RL) studies how an agent can learn, through interaction with the environment, an optimal policy that maximizes his/her long-term reward [52]. When the problem involves a finite set of states and actions of moderate size, the corresponding value or policy functions can be represented precisely as a table, which is called the tabular setting. However, when the problem contains an enormous number of states or continuous states, often in high dimensions, function approximation must be introduced to approximate the involved value or policy functions. With the rapid development of machine learning techniques for function approximation, modern reinforcement learning (RL) algorithms increasingly rely on function approximation tools to tackle problems with growing complexity, including video games [41], Go [50], and robotics [32].

Despite the astonishing practical success of RL with function approximation when applied to challenging high-dimensional problems, the theoretical understanding of RL al-

*Corresponding author. jihaol@princeton.edu
†jiequnhan@gmail.com

gorithms with function approximation remains relatively limited, particularly when compared to the theoretical results in the tabular setting. In the tabular setting, roughly speaking, we can achieve the minimax sample complexity up to the logarithm term: we need samples of the order of $H^3|\mathcal{S}||\mathcal{A}|/\epsilon^2$ to obtain an ϵ-optimal policy, where H denotes the episode length, $|\mathcal{S}|$ and $|\mathcal{A}|$ denote the size of the state space and action space (see [8,13,25] for detailed discussions). Apparently, these kinds of results become vacuous when $|\mathcal{S}|$ (and/or $|\mathcal{A}|$) is extremely large or infinite. Therefore, the study of sample complexity in the presence of function approximation has received considerable attention in recent years in the RL community. Relatively simple function approximation methods, such as the linear model in [29, 57] or generalized linear model in [37, 56] have been examined in the context of RL algorithms. Meanwhile, nonlinear forms like kernel approximation [15, 39, 40, 58, 59] have also been studied in RL problems to further bridge the gap between theoretical results under restrictive assumptions and practice.

In this paper, we review recent theoretical results in RL with function approximation, from linear setting to nonlinear setting. We mainly focus on the results regarding approximation error and estimation error/sample complexity, which are errors introduced by function approximation and finite datasets, respectively. We first review the basic concepts of RL in Section 2 and introduce two categories of RL algorithms: value-based methods and policy-based methods in Section 3. We are interested in these algorithms when combined with function approximation. In Section 4, we give a general framework for the theoretical analysis of RL with function approximation. We adopt the concepts of approximation error, estimation error, and optimization error from supervised learning to RL and discuss the crucial challenges of analyzing these errors in RL. In Section 5, we introduce RL algorithms with linear function approximation, as it is the simplest function approximation. We introduce the basic linear MDP assumption [29], which assumes that both reward function and transition probability are linear with respect to d known features. Under this or similar assumptions, the Q-value function can be represented as a linear function with respect to the features, and numerous algorithms in the literature can achieve polynomial sample complexity with respect to the number of features d, episode length H, and accuracy ϵ. However, the minimax sample complexity has not been achieved yet.

In Section 6, we further discuss RL with nonlinear function approximation. We first introduce the theoretical results of supervised learning on reproducing kernel Hilbert space, neural tangent kernel, and Barron space, and then discuss how to analyze the approximation error in RL problems with nonlinear function approximation. We then focus on the distribution mismatch phenomenon, which is a crucial challenge of RL compared to supervised learning when analyzing the estimation error in the presence of function approximation. In tabular and linear settings, the distribution mismatch is handled by the L^∞ and UCB estimation. However, as we will point out, both L^∞ and UCB estimation suffer from the curse of dimensionality for various function spaces, including neural tangent kernel, Barron space, and many common reproducing kernel Hilbert spaces. This challenge reveals an essential difficulty of RL problems with nonlinear function approximation, and thus additional assumptions are needed to derive meaningful results for nonlinear RL in the literature, including assumptions on the fast eigenvalue decay of the kernel and assumptions on the finite concentration coefficient. We finally introduce the perturbational

complexity by distribution mismatch in [39], which quantifies the difficulty of a large class of the RL problems in the nonlinear setting, as it can give both lower bound and upper bound of the sample complexity of these RL problems. Directions for future work are discussed in Section 7.

Notations. Given $\mathcal{X}$ as an arbitrary subset of Euclidean space, we use $C(\mathcal{X})$ to denote the bounded continuous function space on $\mathcal{X}$, and $\mathcal{P}(\mathcal{X})$ to denote the probability distribution space on $\mathcal{X}$. For any random variable X, we use $\mathcal{L}(X)$ to denote its law. Given a positive integer H, $[H]$ denotes the set $\{1, \ldots, H\}$. I_d denotes the identity matrix of size d. The notation $\tilde{O}(\cdot)$ ignores poly-logarithmic factors.

2 Preliminary

2.1 Markov decision processes

Throughout this article, otherwise explicitly stated, we mainly focus on the finite-horizon Markov decision process (MDP) $M = (\mathcal{S}, \mathcal{A}, H, P, r, \mu)$ with general time-inhomogeneity as the mathematical model for the RL problem. The specifications are the following:

- $\mathcal{S}$ is the state space and we assume $\mathcal{S}$ is a subset of a Euclidean space.
- $\mathcal{A}$ is the action space and we assume $\mathcal{A}$ is a compact subset of a Euclidean space.
- H is the length of each episode.
- $P\colon [H] \times \mathcal{S} \times \mathcal{A} \mapsto \mathcal{P}(\mathcal{S})$ is the state transition probability. For each $(h, s, a) \in [H] \times \mathcal{S} \times \mathcal{A}$. $P(\,\cdot\,|\,h, s, a)$ denotes the transition probability for the next state at step h if the current state is s and action a is taken.
- $r\colon [H] \times \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, denoting the reward at step h if we choose action a at the state s. Unless explicitly stated, we assume r is deterministic and the range of r is a subset of $[0, 1]$. We also assume that r is a continuous function.
- $\mu \in \mathcal{P}(\mathcal{S})$ is the initial distribution.

We denote a policy by $\pi = \{\pi_h\}_{h=1}^H \in \mathcal{P}(\mathcal{A}\,|\,\mathcal{S}, H)$, where

$$\mathcal{P}(\mathcal{A}\,|\,\mathcal{S}, H) = \Big\{\{\pi_h(\,\cdot\,|\,\cdot\,)\}_{h=1}^H \colon \pi_h(\,\cdot\,|\,s) \in \mathcal{P}(\mathcal{A}) \ \text{ for any } \ s \in \mathcal{S} \ \text{ and } \ h \in [H]\Big\}.$$

In some cases, we need to work with time-homogeneous, infinite-horizon MDP $M = (\mathcal{S}, \mathcal{A}, \gamma, P, r, \mu)$, where

- $\mathcal{S}, \mathcal{A}$ and μ are the same with the finite-horizon case,
- $\gamma \in [0, 1)$ is the discount factor,
- $P : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{P}(\mathcal{S})$ is the state transition probability,
- $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the deterministic reward function.

2.2 Total reward, value function and Bellman equation

Given an MDP M and a policy π, the agent's total reward is defined according to the following interaction protocol with the MDP. The agent starts at an initial states $S_0 \sim \mu$; at each time step $h \in [H]$ the agent takes action $A_h \sim \pi_h(\cdot \mid S_h)$, obtains the reward $r(h, S_h, A_h)$ and observes the next state $S_{h+1} \sim P(\cdot \mid h, S_h, A_h)$. In this way, we generate a trajectory $(S_0, A_0, \ldots, S_H, A_H)$ and we will use $\mathbb{P}_{M,\pi}$ and $\mathbb{E}_{M,\pi}$ to denote the probability and expectation of the trajectory generated by policy π on the MDP M. The expected total reward under policy π is defined by

$$J(M, \pi) = \mathbb{E}_{M,\pi}\left[\sum_{h=1}^{H} r(h, S_h, A_h)\right],$$

and our goal is to find a policy π to maximize $J(M, \pi)$ for a fixed MDP M. We will use

$$J^*(M) = \sup_{\pi \in \mathcal{P}(\mathcal{A}|\mathcal{S},H)} J(M, \pi)$$

to denote the optimal value. For ease of notation in analysis, we will use $\rho_{h,P,\pi,\mu}$ to denote the distribution of (S_h, A_h) under transition P, policy π and initial distribution μ. Moreover, we use $\Pi(h, P, \mu)$ to denote the set of all the possible distributions of $\rho_{h,P,\pi,\mu}$

$$\Pi(h, P, \mu) = \left\{\rho_{h,P,\pi,\mu} \colon \pi \in \mathcal{P}(\mathcal{A} \mid \mathcal{S}, H)\right\},$$

and let

$$\Pi(P, \mu) = \bigcup_{h \in [H]} \Pi(h, P, \mu).$$

Given an MDP M, a policy π, a state $s \in \mathcal{S}$, an action $a \in \mathcal{A}$ and time step $h \in [H]$, we define the value function and Q-value function as follows:

$$V_h^\pi(s) = \mathbb{E}_{M,\pi}\left[\sum_{h'=h}^{H} r(h', S'_h, A'_h) \mid S_h = s\right],$$

$$Q_h^\pi(s, a) = \mathbb{E}_{M,\pi}\left[\sum_{h'=h}^{H} r(h', S'_h, A'_h) \mid S'_h = s, A'_h = a\right],$$

as the expected cumulative reward of the MDP starting from step h. We have the following Bellman equation:

$$Q_h^\pi(s, a) = r(h, s, a) + \mathbb{E}_{s' \sim P(\cdot \mid h,s,a)}\left[V_{h+1}^\pi(s')\right],$$

$$V_h^\pi(s) = \int_{\mathcal{A}} Q_h^\pi(s, a)\, \mathrm{d}\pi_h(a|s),$$

where we define $V_{H+1}^\pi = 0$. The optimal value function and the optimal Q-value function are defined by

$$V_h^*(s) = \sup_{\pi \in \Pi(\mathcal{S},\mathcal{A},H)} V_h^\pi(s),$$

$$Q_h^*(s, a) = \sup_{\pi \in \Pi(\mathcal{S},\mathcal{A},H)} Q_h^\pi(s, a).$$

The famous Bellman optimality equation gives,

$$\begin{aligned} Q_h^*(s,a) &= r(h,s,a) + \mathbb{E}_{s' \sim P(\cdot \mid h,s,a)} \left[V_{h+1}^*(s') \right], \\ V_h^\pi(s) &= \max_{a \in \mathcal{A}} Q_h(s,a), \end{aligned} \tag{2.1}$$

where we again define $V_{H+1}^* = 0$. We will use π^* to denote an optimal policy of Q_h^* satisfying the following greedy condition:

$$\operatorname{supp}\big(\pi_h^*(\cdot \mid s)\big) \subset \big\{ a \in \mathcal{A} : Q_h^*(s,a) = V_h^*(s) \big\}$$

for any $(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}$. The optimal policy π^* satisfies that $V_h^{\pi^*} = V_h^*$ and $Q_h^{\pi^*} = Q_h^*$. We refer readers to [44] for an in-depth discussion of the value function, Bellman equation, and optimal policy.

In the case of the time-homogenous MDP, our goal is to find a policy to maximize the discounted total reward

$$\mathbb{E}_{M,\pi} \left[\sum_{h=1}^{\infty} \gamma^{h-1} r(S_h, A_h) \right].$$

We can then introduce the value function, Bellman equation, and optimal policy similarly as in the time-inhomogeneous case, and we refer [44] for details.

2.3 Simulator models

In RL problems, the exact form of the transition probability P and reward function r is unknown, and we can only interact with the MDP to obtain a near-optimal policy. Obviously, the sample complexity of an algorithm depends on the way in which we are allowed to interact with the MDP (the interaction can be different from the interaction according to which the total reward is defined, as described above). There are two main types of simulators for the MDP, which specify the allowed interaction: the generative model setting and the episodic setting. We describe these settings below, and the results in both settings will be reviewed in this paper.

Generative model setting. In the generative model setting, one can take any time-state-action tuple (h,s,a) as the input of the simulator and obtain a sample $s' \sim P(\cdot \mid h,s,a)$ and the reward $r(h,s,a)$. In this sense, the MDP simulator works as a general generative model.

Episodic setting. The episodic setting is a more restrictive scenario compared to the generative model setting. In the episodic setting, one can only decide an initial state s and the action on each time step to obtain from the simulator a trajectory $(S_1, A_1, \ldots, S_H, A_H)$ and the rewards on the trajectory $r(1, S_1, A_1), \ldots, r(H, S_H, A_H)$. In the episodic setting, it is common to consider the regret: the cumulative difference between the obtained reward and the optimal reward over K episodes

$$\text{Regret}(K) = \sum_{k=1}^{K} \left[J^*(M) - J(M, \pi_k) \right]$$

DOI https://doi.org/10.4208/jml.230105 | Generated on 2025-04-20 07:59:17

for any given policy sequence $(\pi_1, \ldots, \pi_K)$ and we aim to minimize the regret over K episodes. If we define a random policy $\bar{\pi}$, which uniformly chooses a policy among $\pi_1, \ldots, \pi_K$ and apply it to the MDP, then

$$J^*(M) - J(M, \bar{\pi}) = \frac{1}{K}\text{Regret}(K).$$

Therefore, a policy sequence with low regret can generate a near-optimal policy.

3 RL algorithms with function approximation

In this section, we introduce some typical RL algorithms with function approximation. They can be divided into two categories: valued-based methods and policy-based methods.

3.1 Value-based method

Value-based methods approximate the value or Q-value functions and use the Bellman optimality equation (2.1) to learn the optimal value or Q-value functions. The near-optimal policy can then be obtained through the greedy policy with respect to the optimal Q-value function.

If we have access to a generative model, one typical value-based algorithm with function approximation is the fitted Q-iteration algorithm [4, 12, 22, 42]. Its main idea is as follows: noticing that the conditional expectation minimizes the L^2-loss corresponding to the Bellman optimality equation (2.1), we know that for any function space $\mathcal{F}$ such that $Q_h^* \in \mathcal{F}$, Q_h^* is the minimizer of the following optimization problem:

$$\min_{f \in \mathcal{F}} \mathbb{E}_{(s,a)\sim\mu, s'\sim\mathbb{P}_h(\cdot\,|\,s,a)} |f(s,a) - r(h,s,a) - V_{h+1}^*(s')|^2. \tag{3.1}$$

Therefore, in the fitted Q-iteration algorithm, we compute the Q_h^* backwardly through the Bellman optimality equation (2.1). At each step h, we choose n state-action pairs $\{(S_h^i, A_h^i)\}_{1\le i\le n}$, submit the queries $\{(S_h^i, A_h^i, h)\}_{1\le i\le n}$ to the generative model, and obtain the reward and next state $\{(r_h^i, \hat{S}_{h+1}^i)\}_{1\le i\le n}$. We solve an empirical version of the least-square problem (3.1) for $h = H, H-1, \ldots, 1$ backwardly. The pseudocode of the fitted Q-iteration is presented in Algorithm 1. We comment that the performance of the fitted Q-iteration algorithm relies on the samples of state-action pairs $\{(S_h^i, A_h^i)\}_{1\le i\le n}$: we hope these samples are representative enough among those encountered under the optimal policy.

In contrast to the generative model setting where we can directly choose arbitrary state-action pairs to query at each step, when we work in the episodic setting, we need to decide how to take action on each step in order to balance the trade-off between exploitation and exploration. Exploitation concerns taking actions with large estimated Q-values in order to obtain large rewards, while exploration concerns taking actions with high uncertainty in their Q-values in order to obtain more accurate estimates. On the one hand, taking

Algorithm 1 Fitted Q-iteration algorithm

Input: MDP $(\mathcal{S}, \mathcal{A}, H, P, r, \mu)$, function classes $\{\mathcal{F}_h\}_{h=1}^H$, regularization terms $\{\Lambda_h\}_{h=1}^H$, number of samples n, state-action pairs $\{(S_h^i, A_h^i)\}_{h\in[H], i\in[n]}$.

Initialize: $Q_{H+1}(x,a) = 0$ for any $(x,a) \in \mathcal{S}\times\mathcal{A}$.

for $h = H, H-1, \ldots, 1$ **do**

Send $(S_h^1, A_h^1, h), \ldots, (S_h^n, A_h^n, h)$ to the generative model and obtain the rewards and next states $(r_h^1, \hat{S}_{h+1}^1), \ldots, (r_h^n, \hat{S}_{h+1}^n)$ for all the state-action pairs.

Compute $y_h^i = r_h^i + \max_{a\in\mathcal{A}} Q_{h+1}(\hat{S}_{h+1}^i, a)$.

Compute $\hat{Q}_h$ as the minimizer of the optimization problem

$$\min_{f\in\mathcal{F}_h} \frac{1}{n}\sum_{i=1}^n \left|y_h^i - f(S_h^i, A_h^i)\right|^2 + \Lambda_h(f). \tag{3.2}$$

Set $Q_h = \max\{0, \min\{\hat{Q}_h, H\}\}$.

end

Output: $\hat{\pi}$ as the greedy policies with respect to $\{Q_h\}_{h=1}^H$.

actions with large estimated Q-values can lead to large estimated total rewards in this episode, but it can also prevent the agent from discovering better actions. On the other hand, taking actions with high uncertainty can lead to more accurate Q-value estimates, but it may also result in lower total rewards in this episode. Finding the proper balance between exploitation and exploration is an important challenge in RL. The upper confidence bound (UCB) method is a common approach used in reinforcement learning to balance such a trade-off. In the UCB method, a bonus function is added to the estimated Q-value function to reflect the uncertainty in the Q-value estimates. The action that is chosen is the one with the highest sum of the estimated Q-value and the bonus. Since the estimated Q-value and the bonus reflect exploitation and exploration, respectively, the UCB method is widely used in reinforcement learning algorithms and often achieves good performance. The UCB method allows the algorithm to balance the need for exploitation, in order to obtain large immediate rewards, with the need for exploration, in order to obtain more accurate Q-value estimates. We present the pseudocode of one typical such algorithm, the value iteration algorithm [29], in Algorithm 2.

3.2 Policy-based method

In policy-based methods, the policy function is approximated and optimized based on cumulative reward using stochastic gradient descent. A key step in this process is calculating the gradient with respect to the cumulative reward. This relies on the policy gradient theorem [53]

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{M,\pi_\theta}\left[\sum_{h=1}^H \nabla_\theta \log \pi_\theta(A_h \mid S_h) \sum_{h'=h}^T r(h', S_{h'}, A_{h'})\right].$$

DOI https://doi.org/10.4208/jml.230105 | Generated on 2025-04-20 07:59:17

Algorithm 2 Value iteration with function approximation and bonus term

Input: MDP $(\mathcal{S}, \mathcal{A}, H, P, r, \mu)$, number of episodes K, function classes $\{\mathcal{F}_h\}_{h=1}^H$, regularization terms $\{\Lambda_h\}_{h=1}^H$.

Initialize: $Q_h^1(s,a) = 0$ for any $(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}$, $Q_{H+1}^k(s,a) = 0$ for any $(k,s,a) \in [K] \times \mathcal{S} \times \mathcal{A}$.

for $k = 1, \ldots, K$ **do**

  Sample S_1^k from the initial state distribution μ.

  for $h = 1, \ldots, H$ **do**

    Take action $A_h^k = \arg\max_{a \in \mathcal{A}} Q_h^k(S_h^k, A_h^k)$ and observe the reward $r_h^k = r(h, S_h^k, A_h^k)$ and next state $S_{h+1}^k \sim P(\cdot \mid h, S_h^k, A_h^k)$.

  end

  if $k < K$ **then**

    for $h = H, H-1, \ldots, 1$ **do**

      Compute $\hat{Q}_h^{k+1}$ as the minimizer of the optimization problem

$$\min_{f \in \mathcal{F}_h} \left\{ \frac{1}{n} \sum_{i=1}^{k} \left| r_h^i + \max_{a \in \mathcal{A}} Q_{h+1}^{k+1}(S_{h+1}^i, a) - f(S_h^i, A_h^i) \right|^2 + \Lambda_h(f) \right\}. \tag{3.3}$$

      Compute the bonus function $b_h^{k+1} : \mathcal{S} \times \mathcal{A} \to [0, +\infty)$ based on $\{(S_h^i, A_h^i)\}_{i=1}^k$ and $\mathcal{F}_h$. Set $Q_{h+1}^{k+1} = \max\{0, \min\{\hat{Q}_h^{k+1} + b_h^{k+1}, H\}\}$.

    end

  end

end

Output: $\hat{\pi}^k$ as the greedy policies with respect to $\{Q_h^k\}_{h=1}^H$ for $k = 1, \ldots, K$.

Here we only state the naive method to compute the gradient. Several variants can be used to replace $\sum_{h'=h}^T r(h', S_{h'}, A_{h'})$ to reduce the variance, see [48] for a detailed discussion. Algorithm 3 gives the pseudocode of the policy gradient method. Besides the vanilla policy gradient method, several variants of the policy gradient method have been proposed by adding various regularization terms to the cumulative reward as the target of the optimization, including the natural policy gradient [30], proximal policy optimization [49], and trust region policy optimization [47].

4 General framework of theoretical analysis on RL with function approximation

In this section, we will discuss how to distinguish and quantify different sources that affect the performance of RL algorithms that use function approximation. To begin, we will provide a brief overview of supervised learning and error decomposition in that context. Then, we will examine how to adapt these concepts for application in the analysis of RL algorithms with function approximation.

Algorithm 3 Policy gradient method

Input: MDP $(\mathcal{S}, \mathcal{A}, H, P, r, \mu)$, parametrization of policy π_θ, initialization θ_0, batch size N, iteration step K, learning rate η.

Initialize: Set $\theta = \theta_0$.

for $k = 1, \ldots, K$ **do**

Sample N i.i.d. states $\{S_1^i\}_{i=1}^N$ from μ and collect N trajectories $\{(S_1^i, A_1^i, \ldots, S_H^i, A_H^i)\}_{i=1}^N$ using policy π_θ.

Estimate the gradient

$$g_k = \frac{1}{N} \sum_{i=1}^{N} \sum_{h=1}^{H} \nabla_\theta \log \pi_\theta(A_h^i \mid S_h^i) \sum_{h'=h}^{T} r(h', S_{h'}^i, A_{h'}^i).$$

Update $\theta \leftarrow \theta + \eta g_k$.

end

Output: π_θ.

In supervised learning, the goal is to estimate the target function f^* based on a finite training set

$$\mathcal{D} = \{x_i, y_i\}_{i=1}^n,$$

where $x_1, \ldots, x_n$ are i.i.d. sampled from a fixed distribution μ,

$$y_i = f^*(x_i) + \epsilon_i$$

and the noises $\epsilon_1, \ldots, \epsilon_n$ are i.i.d. standard normal distribution independent of $x_1, \ldots, x_n$. We aim to find an estimator $\hat{f}$ with a small population loss

$$\mathcal{R}(\hat{f}) = \mathbb{E}_{x \sim \mu} |f^*(x) - \hat{f}(x)|^2.$$

In a standard procedure of supervised learning, one first chooses a hypothesis space or a set of trial functions $\mathcal{H}_m = \{f(x; \theta) : \theta \in \Theta_m\}$, where θ denotes the parameters and m denotes the number of parameters in $\mathcal{H}_m$. Common choices of the hypothesis space include linear functions, kernel functions, and neural networks. The next step is to choose a loss function and formulate an optimization problem. The loss function is typically composed of the empirical loss $\mathcal{R}_n(\theta)$ and a regularization term $\Lambda(\theta)$

$$\mathcal{L}_n(\theta) = \mathcal{R}_n(\theta) + \Lambda(\theta) = \frac{1}{n} \sum_{i=1}^{n} |y_i - f(x_i; \theta)|^2 + \Lambda(\theta).$$

The last step is to solve the optimization problem that aims to minimize the above loss function. It is usually solved by the gradient descent method, stochastic gradient descent method, or their variants.

Let f_m be the minimizer of the population loss $\mathcal{R}(f)$, the best approximation to f^* in $\mathcal{H}_m$, $f_{m,n}$ be the minimizer of the loss function $\mathcal{L}_n(\theta)$ in the hypothesis space $\mathcal{H}_m$ and $\hat{f}$ be the output of the optimization algorithm. Then the total error between the true target

function f^* and the output of the supervised learning algorithm $\hat{f}$ can be decomposed into three parts, where $\|\cdot\|_{L^2(\mu)}$ denotes the L^2-norm under the distribution μ

$$\|f^* - \hat{f}\|_{L^2(\mu)} \leq \underbrace{\|f^* - f_m\|_{L^2(\mu)}}_{\text{approximation error}} + \underbrace{\|f_m - f_{m,n}\|_{L^2(\mu)}}_{\text{estimation error}} + \underbrace{\|f_{m,n} - \hat{f}\|_{L^2(\mu)}}_{\text{optimization error}}.$$

The first part is the approximation error $\|f^* - f_m\|_{L^2(\mu)}$, which arises because the hypothesis space $\mathcal{H}_m$ may not be able to represent the true function f^* exactly. The second part is the estimation error $\|f_m - f_{m,n}\|_{L^2(\mu)}$, which arises because we only have a finite dataset and may not be able to find the best approximation f_m. The third part is the optimization error $\|f_{m,n} - \hat{f}\|_{L^2(\mu)}$, which arises because the optimization algorithm may not converge to the true minimizer of the empirical loss. We will then discuss the approximation error, estimation error, and optimization error in the context of RL with function approximation.

4.1 Approximation error

To investigate the approximation error in the context of RL, we aim to understand the requirements for the transition probability and reward function needed to accurately approximate the Q-value function for value-based methods and the policy function for policy-based methods. Note that the value function V_h^* is less significant, as we cannot directly compute the optimal policy based on it. The subsequent theorem demonstrates that when the action space is finite and the optimal Q-value function can be accurately approximated, the optimal policy can also be accurately approximated by the corresponding softmax policy. Consequently, our primary focus is on the conditions that ensure the Q-value function can be effectively approximated.

Theorem 4.1. *Assume that $\mathcal{A}$ is a finite set. Given any $\beta > 0$ and continuous functions $Q = \{Q_h\}_{h=1}^H : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, let*

$$\pi_h^{Q,\beta}(a \mid s) = \frac{\exp(\beta Q_h(s,a))}{\sum_{a' \in \mathcal{A}} \exp(\beta Q_h(s,a'))}.$$

Then,

$$\begin{aligned} 0 &\leq J^*(M) - J(M, \pi^{Q,\beta}) \\ &\leq \frac{H \log |\mathcal{A}|}{\beta} + 2\beta H \sum_{h=1}^{H} \mathbb{E}_{M, \pi^{Q^*,\beta}} \max_{a \in \mathcal{A}} |Q_h^*(S_h, a) - Q_h(S_h, a)|. \end{aligned}$$

In supervised learning, a common approach to investigating the approximation error involves proving that the target function resides in specific function spaces, such as reproducing kernel Hilbert space (RKHS) and Barron space. These spaces are ideal for particular approximation schemes, like kernel function and neural network approximation, due to the direct and inverse approximation theorems present within these function

spaces. In other words, any function within the space can be approximated using the selected approximation method at a specific rate of convergence, and any function that can be approximated at a particular rate belongs to that function space. We will provide a comprehensive introduction to RKHS and Barron space in Section 6.1.

In RL, our primary focus is on whether the Q-value function, as a function of state and action, lies within the specific function space. Depending on the specific algorithm we analyze, we need to determine the conditions of P and r under which the MDP satisfies the following properties:

Property 1. The optimal Q-value function Q_h^* lies in the specific function space.

Property 2. The Q-value function Q_h^π lies in the specific function space for any policy π. This is often required in the policy iteration algorithm with function approximation (see, e.g., [34]).

Property 3. The Bellman optimal operator

$$(\mathcal{T}_h^* f)(s,a) = r(h,s,a) + \mathbb{E}_{s'\sim P(\,\cdot\,|\,h,s,a)}\big[\max_{a'\in\mathcal{A}} f(s',a')\big]$$

maps the specific function space to itself and $\|\mathcal{T}_h^* f\| \le C[1+\|f\|]$ for a constant $C>0$. This is often required in the fitted Q-iteration and value iteration algorithm with function approximation (see, e.g., [58–60]).

Property 4. The Bellman operator

$$(\mathcal{T}_h f)(s,a) = r(h,s,a) + \mathbb{E}_{s'\sim P(\,\cdot\,|\,h,s,a)}[f(s')]$$

maps any bounded function in $C(\mathcal{S})$ to the specific function spaces and $\|\mathcal{T}_h f\| \le C[1+\|f\|_{C(\mathcal{S})}]$ for a constant $C>0$.

In these questions, we assume that the function space is a Banach space with norm $\|\cdot\|$ and a subset of the space of all bounded functions in $C(\mathcal{S}\times\mathcal{A})$. Noticing that for any policy π, we have that $Q_h^\pi = \mathcal{T}_h V_{h+1}^\pi$ and $V_{h+1}^\pi \in [0,H]$, we know that Property 4 implies Property 2 and hence implies Property 1. Observing that for any bounded function f in $C(\mathcal{S}\times\mathcal{A})$, $\max_{a'\in\mathcal{A}} f(s',a')$ is a bounded function in $C(\mathcal{S})$, we know that Property 4 implies Property 3. Moreover, since $Q_h^* = \mathcal{T}_h Q_{h+1}^*$, Property 3 implies Property 1. Finally, we remark that Property 2 and Property 3 cannot be inferred from each other [60, Proposition 5]. We introduce Property 4 because it is the strongest one among those properties and the conditions which imply Property 4 are easy to analyze due to the linearity of $\mathcal{T}_h$.

Proof of Theorem 4.1. By the definition of the optimal value, we have

$$J^*(M) - J\big(M,\pi^{Q,\beta}\big) \ge 0.$$

Noticing that

$$\begin{aligned} J^*(M) - J\big(M,\pi^{Q,\beta}\big) &= J^*(M) - J\big(M,\pi^{Q^*,\beta}\big) + J\big(M,\pi^{Q^*,\beta}\big) - J(M,\pi^{Q,\beta}) \\ &=: I_1 + I_2, \end{aligned}$$

we will estimate I_1 and I_2 respectively.

Using the classical performance difference lemma [31], we have

$$\begin{aligned} I_1 &= \sum_{h=1}^{H} \mathbb{E}_{M,\pi^{Q^*,\beta}} \sum_{a\in\mathcal{A}} Q_h^*(S_h,a)\left[\pi_h^*(a\mid S_h) - \pi_h^{Q^*,\beta}(a\mid S_h)\right] \\ &= \sum_{h=1}^{H} \mathbb{E}_{M,\pi^{Q^*,\beta}} \left[\max_{a\in\mathcal{A}} Q_h^*(S_h,a) - \frac{\sum_{a\in\mathcal{A}} Q_h^*(S_h,a)\exp(\beta Q_h^*(S_h,a))}{\sum_{a\in\mathcal{A}}\exp(\beta Q_h^*(S_h,a))}\right]. \end{aligned}$$

We will then prove that for any $q = (q_a)_{a\in\mathcal{A}}$,

$$q_m - \frac{\sum_{a\in\mathcal{A}} q_a \exp(\beta q_a)}{\sum_{a\in\mathcal{A}}\exp(\beta q_a)} \le \frac{\log|\mathcal{A}|}{\beta},$$

where $q_m = \max_{a\in\mathcal{A}} q_a$. We can then conclude that

$$I_1 \le \frac{H\log|\mathcal{A}|}{\beta}. \tag{4.1}$$

Noticing that

$$q_m - \frac{\sum_{a\in\mathcal{A}} q_a \exp(\beta q_a)}{\sum_{a\in\mathcal{A}}\exp(\beta q_a)} = -\frac{(q_a - q_m)\exp(\beta(q_a - q_m))}{\exp(\beta(q_a - q_m))}.$$

Define $\phi : \mathbb{R}^{|\mathcal{A}|} \to \mathbb{R}$

$$\phi(x) = \beta^{-1}\log\left(\sum_{a\in\mathcal{A}}\exp(\beta x_a)\right).$$

Then

$$q_m - \frac{\sum_{a\in\mathcal{A}} q_a \exp(\beta q_a)}{\sum_{a\in\mathcal{A}}\exp(\beta q_a)} = -(q-q_m)^{\mathrm{T}}\nabla\phi(q-q_m).$$

Noticing that ϕ is a convex function, we have [24]

$$\begin{aligned} -(q-q_m)^{\mathrm{T}}\nabla\phi(q-q_m) &= [0-(q-q_m)]^{\mathrm{T}}\nabla\phi(q-q_m) \\ &\le \phi(0) - \phi(q-q_m) \\ &= \frac{\log|\mathcal{A}|}{\beta} - \phi(q-q_m). \end{aligned}$$

Noticing that there exists an $a\in\mathcal{A}$ such that $q_a - q_m = 0$, we have

$$\phi(q-q_m) \ge 0.$$

Therefore,

$$q_m - \frac{\sum_{a\in\mathcal{A}} q_a \exp(\beta q_a)}{\sum_{a\in\mathcal{A}}\exp(\beta q_a)} \le \frac{\log|\mathcal{A}|}{\beta}.$$

DOI https://doi.org/10.4208/jml.230105 | Generated on 2025-04-20 07:59:17

For I_2, we again use the performance difference lemma to obtain that

$$\begin{aligned}
I_2 &= \sum_{h=1}^{H} \mathbb{E}_{M,\pi^{Q^*,\beta}} \sum_{a\in\mathcal{A}} Q_h^{\pi^{Q^*,\beta}}(S_h,a)\left[\pi_h^{Q^*,\beta}(a\,|\,S_h) - \pi_h^{Q,\beta}(a\,|\,S_h)\right] \\
&\le H\sum_{h=1}^{H} \mathbb{E}_{M,\pi^{Q^*,\beta}} \sum_{a\in\mathcal{A}} \left|\pi_h^{Q^*,\beta}(a\,|\,S_h) - \pi_h^{Q,\beta}(a\,|\,S_h)\right| \\
&= H\sum_{h=1}^{H} \mathbb{E}_{M,\pi^{Q^*,\beta}} \sum_{a\in\mathcal{A}} \left|\frac{\exp(\beta Q_h^*(S_h,a))}{\sum_{a'\in\mathcal{A}}\exp(\beta Q_h^*(S_h,a'))} - \frac{\exp(\beta Q_h(S_h,a))}{\sum_{a'\in\mathcal{A}}\exp(\beta Q_h(S_h,a'))}\right|.
\end{aligned}$$

Given $q=\{q_a\}_{a\in\mathcal{A}}$ and $\bar{q}=\{\bar{q}_a\}_{a\in\mathcal{A}}$, we have

$$\begin{aligned}
&\sum_{a\in\mathcal{A}}\left|\frac{\exp(\beta q_a)}{\sum_{a'\in\mathcal{A}}\exp(\beta q_{a'})} - \frac{\exp(\beta \bar{q}_a)}{\sum_{a'\in\mathcal{A}}\exp(\beta \bar{q}_{a'})}\right| \\
&\le \frac{\sum_{a\in\mathcal{A}}\sum_{a'\in\mathcal{A}}|\exp(\beta q_a+\beta\bar{q}_{a'}) - \exp(\beta\bar{q}_a+\beta q_{a'})|}{\sum_{a\in\mathcal{A}}\exp(\beta q_a)\sum_{a\in\mathcal{A}}\exp(\beta\bar{q}_a)} \\
&\le \beta\max_{a\in\mathcal{A}}|q_a-\bar{q}_a|\frac{\sum_{a\in\mathcal{A}}\sum_{a'\in\mathcal{A}}[\exp(\beta q_a+\beta\bar{q}_{a'})+\exp(\beta\bar{q}_a+\beta q_{a'})]}{\sum_{a\in\mathcal{A}}\exp(\beta q_a)\sum_{a\in\mathcal{A}}\exp(\beta\bar{q}_a)} \\
&= 2\beta\max_{a\in\mathcal{A}}|q_a-\bar{q}_a|,
\end{aligned}$$

where we used $|e^x-e^y|\le(e^x+e^y)|x-y|/2$. Therefore,

$$I_2 \le 2\beta H\sum_{h=1}^{H}\mathbb{E}_{M,\pi^{Q^*,\beta}}\max_{a\in\mathcal{A}}\left|Q_h^*(S_h,a)-Q_h(S_h,a)\right|.$$

Combining the above estimation and inequality (4.1), we conclude our proof. □

4.2 Estimation error

In the context of RL, sample complexity, i.e., the number of samples required to obtain a near-optimal policy, is often used to refer to the estimation error and plays a central role in the theoretical analysis of RL. However, in contrast to the estimation error in supervised learning, which can be characterized by the gap between the empirical loss and the population loss, the estimation error in RL is much more complex. The main challenge in the RL problem is the so-called distribution mismatch phenomenon. Take value-based methods as an example. Assume that we have an estimation of the optimal Q-value function Q_h^*, denoted by $\hat{Q}_h^*$, which is close to Q_h^* in the sense of $L^2(\nu)$ for a prespecified distribution ν. Then, we consider the performance of the greedy policy $\hat{\pi}$ with respect to $\hat{Q}_h^*$. Using the performance difference lemma, we have

$$0 \le J(M,\pi^*) - J(M,\pi) = \mathbb{E}_{M,\hat{\pi}}\sum_{h=1}^{H}\sum_{a\in\mathcal{A}} Q_h^*(S_h,a)\left[\pi_h^*(a\,|\,S_h) - \hat{\pi}_h(a\,|\,S_h)\right]$$

$$\leq \mathbb{E}_{M,\hat{\pi}} \sum_{h=1}^{H} \sum_{a\in\mathcal{A}} \left[Q_h^*(S_h,a) - \hat{Q}_h^*(S_h,a)\right] \left[\pi_h^*(a \mid S_h) - \hat{\pi}_h(a \mid S_h)\right],$$

where in the last inequality, we use that $\sum_{a\in\mathcal{A}} \hat{Q}_h^*(s,a)[\pi_h^*(a \mid s) - \hat{\pi}_h(a \mid s)] \leq 0$ since $\hat{\pi}_h$ is the greedy policy with respect to $\hat{Q}_h^*$. Therefore, we need to control the difference between Q_h^* and $\hat{Q}_h^*$ under the state distribution generated by the policy $\hat{\pi}$, which is unknown before we obtain $\hat{Q}_h^*$. We refer to this phenomenon as the distribution mismatch: a mismatch between the distribution ν for estimation and the distribution for evaluation that is unknown a priori. This phenomenon is ubiquitous in the analysis of RL; see, e.g., [31, Section 6].

4.3 Optimization error

In the context of RL, the optimization error differs significantly between value-based methods and policy-based methods. In value-based methods, the optimization error arises during the optimization process at each iteration, such as in (3.2) or (3.3) in Algorithm 1 or Algorithm 2. As these optimization problems typically have a similar form to those in supervised learning, the analysis of the optimization error in RL is largely comparable to the analysis of the optimization error in supervised learning. On the other hand, the optimization error in policy-based methods, particularly the rate at which the algorithm's performance converges as the number of iterations increases, is a key focus in the theoretical analysis of these methods. The analysis of the optimization problem in policy-based methods is more challenging than in supervised learning due to the shift of the distribution of the trajectories $\{S_1^i, A_1^i, \ldots, S_H^i, A_H^i\}_{i=1}^N$ in Algorithm 3 during the optimization process in policy-based methods.

5 Linear setting

The simplest form of function approximation is linear function approximation. It is the setting under which the most recent theoretical results in RL are derived with function approximation. In the linear setting, we do not assume that the state space and action space are finite, and hence we need to make some structural assumptions to obtain meaningful results. The most common one is the linear MDP assumption introduced in [29].

Definition 5.1 (Linear MDP)**.** *We say an MDP$(\mathcal{S},\mathcal{A},H,P,r,\mu)$ is a linear MDP with a feature map $\boldsymbol{\phi} : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$, if for any $h \in [H]$, there exists d unknown signed measures $\boldsymbol{\mu}_h = (\mu_h^1, \ldots, \mu_h^d)$ over $\mathcal{S}$ and an unknown vector $\boldsymbol{\theta}_h \in \mathbb{R}^d$, such that for any $(s,a) \in \mathcal{S}\times\mathcal{A}$*

$$P(\cdot \mid h,s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\mu}_h(\cdot),$$
$$r(h,s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\theta}_h.$$

We shall notice that the tabular MDP is a special case of the linear MDP if we set $d = |\mathcal{S}||\mathcal{A}|$, index each coordinate of $\mathbb{R}^d$ by state-action pair $(s,a) \in \mathcal{S}\times\mathcal{A}$, choose $\boldsymbol{\phi}(s,a)$ as

the canonical basis in $\mathbb{R}^d$ and set

$$(\boldsymbol{\theta}_h)_{s,a} = r(h,s,a), \quad \big(\boldsymbol{\mu}_h(\cdot)\big)_{s,a} = P(\cdot \,|\, h,s,a)$$

for any $(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}$.

5.1 Approximation error

The next theorem demonstrates that the linear MDP assumption is the necessary and sufficient condition of the Property 4 holding true when discussing the approximation error in Section 4.1. Noticing that Property 4 is the strongest one among the four, we know that under linear MDP assumption, for any policy π, the corresponding Q-value function Q_h^π lies in the linear space, i.e., there exist weights $\{\boldsymbol{w}_h^\pi\}_{h\in[H]}$ such that for any $(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}$,

$$Q_h^\pi(s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{w}_h^\pi.$$

Theorem 5.1. *Assume that $\mathcal{S} \times \mathcal{A}$ is a compact set and $\boldsymbol{\phi} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ is a feature map. Then the following two statements are equivalent:*

1. *There exist d signed measures $\boldsymbol{\mu}_h = (\mu_h^1, \ldots, \mu_h^d)$ over $\mathcal{S}$ and a vector $\boldsymbol{\theta}_h \in \mathbb{R}^d$, such that for any $(s,a) \in \mathcal{S} \times \mathcal{A}$*

$$P(\cdot \,|\, h,s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\mu}_h(\cdot),$$
$$r(h,s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\theta}_h.$$

2. *For any $f \in C(\mathcal{S})$, there exists $\boldsymbol{\omega}_{f,h} \in \mathbb{R}^d$ such that*

$$(\mathcal{T}_h f)(s,a) = r(h,s,a) + \mathbb{E}_{s'\sim P(\cdot \,|\, h,s,a)}[f(s')] = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\omega}_{f,h}$$

and $\|\boldsymbol{\omega}_{f,h}\| \le C[\|f\|_{C(\mathcal{S})} + 1]$ for a constant $C > 0$, where we use $\|\cdot\|$ to denote the l^2-norm on $\mathbb{R}^d$.

The linear MDP assumption completely ensures Property 4 (in Section 4.1) in the linear setting, but it rules out non-trivial deterministic MDPs that are frequently encountered in real-world situations. This is because $P(\cdot \,|\, h,s,a)$ is a delta distribution and hence the supports of $\mu_h^1, \ldots, \mu_h^d$ are all single point sets. Therefore, the MDP can only visit at most d state when $h \ge 2$. Hence, there are some studies that direct assume Property 2 [2] or Property 3 [60]. However, there has been little investigation into the concrete conditions of the transition probability and reward function under which the MDP satisfies Property 2 or Property 3.

Proof of Theorem 5.1. $1 \Rightarrow 2$: For any $f \in C(\mathcal{S})$, we have

$$\mathcal{T}_h f = r(h,s,a) + \mathbb{E}_{s'\sim P(\cdot \,|\, h,s,a)}[f(s')] = \phi^{\mathrm{T}}(s,a)\left[\boldsymbol{\theta_h} + \int_{\mathcal{S}} f(s')\,\mathrm{d}\boldsymbol{\mu}_h(s')\right].$$

Therefore, let

$$\boldsymbol{\omega}_{f,h} = \boldsymbol{\theta}_h + \int_{\mathcal{S}} f(s')\,\mathrm{d}\boldsymbol{\mu}_h(s').$$

Hence,

$$\|\boldsymbol{\omega}_{f,h}\| \leq \|\boldsymbol{\theta}_h\| + \||\boldsymbol{\mu}_h|_{TV}\|\|f\|_{C(\mathcal{S}\times\mathcal{A})},$$

where $|\boldsymbol{\mu}_h|_{TV} = (|\mu_h^1|_{TV}, \ldots, |\mu_h^d|_{TV})$ is the total variation of signed measures $\boldsymbol{\mu}_h$. We can then choose $C = \max\{\|\boldsymbol{\theta}_h\|, \||\mu|_{TV}\|\}$.

$2 \Rightarrow 1$: Let $f = 0$, we know that we can choose $\boldsymbol{\theta}_h = \boldsymbol{\omega}_{0,h}$ such that $r(h,s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\theta_h$. Moreover $\|\boldsymbol{\theta}_h\| \leq C$.

Let

$$\mathcal{W}_0 = \left\{\boldsymbol{\omega} \in \mathbb{R}^d : \boldsymbol{\phi}^{\mathrm{T}}(s,a)\cdot\boldsymbol{\omega} = 0, \forall (s,a) \in \mathcal{S}\times\mathcal{A}\right\}.$$

Noticing that $\mathcal{W}_0$ is a subspace of $\mathbb{R}^d$, we can define $\mathcal{W}$ as the orthogonal complement of $\mathcal{W}_0$. For any $f \in C(\mathcal{S})$, let $\boldsymbol{\omega}'_{f,h}$ be the orthogonal projection of $\boldsymbol{\omega}_{f,h}$ to $\mathcal{W}$. Then by the definition of $\mathcal{W}_0$, we have

$$(\mathcal{T}_h f)(s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\omega}_{f,h} = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\omega}'_{f,h}.$$

On the other hand, for any $\boldsymbol{\omega} \in \mathcal{W}$ such that $(\mathcal{T}_h f)(s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\omega}$ holds for any $(s,a) \in \mathcal{S}\times\mathcal{A}$, by the definition of N_0, we know that $\omega = \omega'_{f,h}$. We can then define a mapping from f to $\boldsymbol{\omega}'_{f,h}$ and $\mathcal{B} : C(\mathcal{S}\times\mathcal{A}) \to N$

$$\mathcal{B}f = \boldsymbol{\omega}'_{f,h} - \boldsymbol{\theta}'_h,$$

where $\boldsymbol{\theta}'_h$ is the orthogonal projection of $\boldsymbol{\theta}_h$ to $\mathcal{W}$. Then,

$$\mathbb{E}_{s'\sim P(\,\cdot\,|\,h,s,a)}[f(s')] = \phi^{\mathrm{T}}(s,a)(\mathcal{B}f).$$

We can then prove that $\mathcal{B}f$ is a linear mapping and

$$\|\mathcal{B}f\| \leq \|\boldsymbol{\omega}'_{f,h}\| + \|\boldsymbol{\theta}'_h\| \leq \|\boldsymbol{\omega}_{f,h}\| + \|\boldsymbol{\theta}_h\| \leq C\big(\|f\|_{C(\mathcal{S}\times\mathcal{A})} + 2\big).$$

Then for any $\|f\|_{C(\mathcal{S}\times\mathcal{A})} = 1$, we have $\|\mathcal{B}f\| \leq 3C$, which means that

$$\|\mathcal{B}f\| \leq 3C\|f\|_{C(\mathcal{S}\times\mathcal{A})}.$$

Therefore, $\mathcal{B}f$ is a bounded linear mapping from $C(\mathcal{S}\times\mathcal{A})$ to $\mathcal{W}$. Noticing that $\mathcal{W}$ is a finite-dimensional linear space, we can use the Risez representation theorem on $C(\mathcal{S}\times\mathcal{A})$ [45] to show that there exists d signed measure $\boldsymbol{\mu}_h = (\mu_h^1, \ldots, \mu_h^d)$ such that for any $f \in C(\mathcal{S}\times\mathcal{A})$,

$$\begin{aligned}\mathbb{E}_{s'\sim P(\,\cdot\,|\,h,s,a)}[f(s')] &= \boldsymbol{\phi}^{\mathrm{T}}(s,a)(\mathcal{B}f) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\int_{\mathcal{S}} f(s')\,\mathrm{d}\boldsymbol{\mu}_h(s') \\ &= \int_{\mathcal{S}} f(s')\,\mathrm{d}\boldsymbol{\phi}^{\mathrm{T}}(s,a)\mu_h(s'),\end{aligned}$$

which means that

$$P(\,\cdot\,|\,h,s,a) = \boldsymbol{\phi}^{\mathrm{T}}(s,a)\boldsymbol{\mu}_h(\cdot).$$

The proof is complete. □

5.2 Estimation and optimization error

Under the linear MDP assumption, the above analysis justifies our use of a linear function to approximate the Q-value function. We can then apply the fitted Q-iteration algorithm or value iteration methods in Section 3.1 with linear function approximation. We can also use the linear function to approximate the Q-value function and then use the softmax policy to approximate the optimal policy to apply the policy gradient algorithms in Section 3.2. Here we take the value iteration (Algorithm 2) in the episodic setting [29] as an example to discuss the estimation error. In [29], they set

$$\mathcal{F}_h = \left\{ \boldsymbol{\phi}^{\mathrm{T}}(s,a)\theta, \theta \in \mathbb{R}^d \right\},$$

$$\Lambda_h(f) = \lambda \|\theta\|^2,$$

$$b_h^{k+1}(s,a) = \beta \left[\boldsymbol{\phi}(s,a)^{\mathrm{T}} \left(\sum_{i=1}^{k} \boldsymbol{\phi}(S_h^i, A_h^i) \boldsymbol{\phi}^{\mathrm{T}}(S_h^i, A_h^i) + n\lambda \mathrm{I} \right)^{-1} \boldsymbol{\phi}(s,a) \right]^{\frac{1}{2}}.$$

Intuitively speaking, the bonus term b_h^{k+1} is proportional to the standard deviation of the $\hat{Q}_h^{k+1}$. It is proved that [29, Lemma B.5], with high probability,

$$Q_h^*(s,a) \leq Q_h^{k+1}(s,a), \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}.$$

In this sense, Q_h^{k+1} is the uniformly upper confidence bound of Q_h^*. The introduction of the bonus term b_h^{k+1} and resulting UCB estimation is the crucial step to address the distribution mismatch phenomenon. By properly choosing the parameters β and λ, [29] prove that Algorithm 2 can achieve the following regret bounds (recalling that K denotes the number of episodes):

$$\mathrm{Regret}(K) = \tilde{O}(\sqrt{d^3 H^4 K}),$$

which implies that to obtain an ϵ-optimal policy, the algorithm needs at most $\tilde{O}(d^3 H^5 / \epsilon^2)$ samples. Such a result is independent of the number of states and actions and is much more general than the tabular setting.

In Algorithm 2, the optimization problem (3.3) is a ridge regression problem whose solution can be exactly computed, so there is no error introduced in this optimization step. However, we need to compute $\max_{a \in \mathcal{A}} Q_h^k(s,a)$ many times. If $|\mathcal{A}|$ is finite, the maximum can be exactly computed. However, if $|\mathcal{A}|$ is infinite, numerical errors may be introduced when calculating the maximum value.

There are other works in the RL literature studying RL problems in the linear setting. [36, 54, 57] consider the linear setting with a generative model in the time-homogeneous case. These works use L^∞ estimation instead of UCB estimation to handle the distribution mismatch phenomenon. [60] considers a similar assumption with [29] but provides an algorithm with a tighter regret bound. [61] also considers the RL problem in a linear setting but with respect to discounted MDP with infinite horizons. [1, 2, 10] study the policy-based methods for the RL problem in the linear setting. Most of these results establish the polynomial sample complexity with respect to the number of features d, the length

of the episode H, and the accuracy ϵ under similar assumptions. However, so far, the gap between the lower bound and upper bound in sample complexity still exists in the linear setting, except for results in [54, 57], which require a generative model and a very restrictive assumption called anchor state-action pairs assumption. This gap is evident by noticing that the tabular MDP is a special case of the linear MDP, and the lower bound in the tabular MDP implies a naive lower bound dH^3/ϵ^2; see [29, Section 5] for a detailed discussion. Bridging the gap is an important direction for future work.

6 Nonlinear setting

6.1 RKHS, NTK and Barron space

As powerful function approximation tools (particularly in high dimensions), kernel functions and neural networks are now widely used in various machine learning tasks, including RL problems. Theoretical analysis of RL algorithms involving function approximations hinges on the proper choice of function spaces and a deep understanding of these spaces. In this subsection, we will briefly introduce the concepts of reproducing kernel Hilbert space (RKHS), neural tangent kernel (NTK), and Barron space, as they are suitable function spaces associated with kernel function and neural network approximation. In particular, we introduce the theoretical results of supervised learning algorithms in these function spaces, which will be the foundation for analyzing RL algorithms with function approximation.

RKHS. Suppose k is a continuous positive definite kernel that satisfies:

1. $k(x, x') = k(x', x), \forall x, x' \in \mathcal{X}$;
2. For all $m \geq 1$, $x_1, \ldots, x_m \in \mathcal{X}$ and $a_1, \ldots, a_m \in \mathbb{R}$, we have
$$\sum_{i=1}^{m} \sum_{j=1}^{m} a_i a_j k(x_i, x_j) \geq 0.$$

Then, there exists a Hilbert space $\mathcal{H}_k \subset C(\mathcal{X})$ such that:

1. For all $x \in \mathbb{R}^d$, $k(x, \cdot) \in \mathcal{H}_k$;
2. For all $x \in \mathbb{R}^d$ and $f \in \mathcal{H}_k$, $f(x) = \langle f, k(x, \cdot)\rangle_{\mathcal{H}_k}$.

k is called the reproducing kernel of $\mathcal{H}_k$ [5], and we use $\|\cdot\|_{\mathcal{H}_k}$ to denote the norm of the Hilbert space $\mathcal{H}_k$. Common examples of reproducing kernels include the Gaussian kernel $k(x, x') = \exp(-\alpha\|x - x'\|^2)$ and the Laplacian kernel $k(x, x') = \exp(-\alpha\|x - x'\|)$ $(\alpha > 0)$.

The kernel method can efficiently learn functions in the RKHS with finite data. In the kernel method, we set the hypothesis space as $\mathcal{H}_k$, the entire RKHS. In this sense, since the target function f^* lies in $\mathcal{H}_k$, the approximation error is zero. We can then define the loss function
$$\mathcal{L}_n(f) = \frac{1}{n} \sum_{i=1}^{n} |y_i - f(x_i)|^2 + \lambda \|f\|_{\mathcal{H}_k}^2.$$

We can then obtain the kernel ridge estimator $\hat{f} = \arg\min_{f\in\mathcal{H}_k} \mathcal{L}_n(f)$. If we choose $\lambda = n^{-1/2}$, then the estimation error

$$\|f^* - \hat{f}\|_{L^2(\mu)} \leq \mathcal{O}\big([1 + \|f\|_{\mathcal{H}_k}]n^{-\frac{1}{4}}\big).$$

We can then show that the kernel ridge estimator can be computed exactly, and hence the optimization error is zero. First, using [43, Proposition 4.2], we know that

$$\min_{f\in\mathcal{H}_k} \mathcal{L}_n(f) = \min_{f=\sum_{i=1}^n \alpha_i k(\cdot,x_i)} \mathcal{L}_n(f).$$

Then we only need to compute the $\alpha_1, \ldots, \alpha_n$ to minimize the loss function. Moreover, if $f = \sum_{i=1}^n \alpha_i k(\cdot, x_i)$, then

$$\mathcal{L}_n(f) = \frac{1}{n}[\boldsymbol{y} - K_n\boldsymbol{\alpha}]^{\mathrm{T}}[\boldsymbol{y} - K_n\boldsymbol{\alpha}] + \lambda\alpha^{\mathrm{T}}K_n\boldsymbol{\alpha},$$

where $\boldsymbol{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^{\mathrm{T}}$ and $K_n = (k(x_i, x_j))_{1\leq i,j\leq n}$ is an $n \times n$ matrix. Therefore,

$$\hat{\alpha} = (K_n + \lambda n \mathrm{I}_d)^{-1}\boldsymbol{y},$$

which can be computed directly. See [11, 46, 51] for more details on the kernel method.

NTK. Neural tangent kernel (NTK) was first introduced to study the overparameterized neural networks [27]. For the input data $x \in \mathbb{R}^d$, we consider a two-layer ReLU neural network with m neurons

$$f(x;\theta) = \frac{1}{\sqrt{m}} \sum_{i=1}^m a_i \sigma\big(\omega_i^{\mathrm{T}} x\big),$$

where $\sigma(x) = \max\{x, 0\}$ is the ReLU activation function. Here θ denotes the collection of all the parameters $(a_1, \omega_1, \ldots, a_m, \omega_m)$, with $a_i \in \mathbb{R}, \omega_i \in \mathbb{R}^d, i = 1, \ldots, m$. If we initialize θ according to the following rule:

$$a_i \overset{i.i.d.}{\sim} \mathcal{N}(0, 1),\ \omega_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \mathrm{I}_d/d), \tag{6.1}$$

then the fully trained neural networks approximate the kernel ridge regression on the RKHS with respect to the NTK k_{NTK} when the width m goes to the infinity [6], where

$$k_{\mathrm{NTK}}(x, x') = \mathbb{E}_{\omega\sim\mathcal{N}(0,\mathrm{I}_d/d)}\Big[x^{\mathrm{T}}x'\sigma'(\omega^{\mathrm{T}}x)\sigma'(\omega^{\mathrm{T}}x') + \sigma(\omega^{\mathrm{T}}x)\sigma(\omega^{\mathrm{T}}x')\Big]$$

and $\sigma'(x) = 1_{x>0}$ is the derivative of σ. The theory of NTK establishes a connection between neural network and kernel methods and shows that it is sufficient to study the corresponding NTK if we are interested in the overparameterized neural networks with the NTK scaling (6.1). Therefore, we will not discuss RL with neural network approximation under the NTK regime, as it is covered by the kernel function approximation. For more details on the NTK theory, including the NTK corresponding to multi-layer neural networks, see [3, 6, 7, 18, 27].

Barron space. Barron space is introduced to study the overparameterized neural networks as well but with a different scaling. We consider the two-layer ReLU neural network with mean-field scaling

$$f(x;\theta) = \frac{1}{m}\sum_{i=1}^{m} a_i \sigma\big(\omega_i^{\mathrm{T}} x\big).$$

The function in the Barron space serves as the continuous analog of the two-layer neural network as the width m goes to infinity

$$f(x) = \int_{\mathbb{R}^d} a(\omega)\sigma(\omega^{\mathrm{T}} x)\,\mathrm{d}\rho(\omega). \tag{6.2}$$

The Barron space is defined as follows:

$$\mathcal{B} = \left\{ f(x) = \int_{\mathbb{R}^d} a(\omega)\sigma(\omega^{\mathrm{T}} x)\,\mathrm{d}\rho(\omega),\ \rho \in \mathcal{P}(\mathbb{R}^d) \ \text{ and } \ \int_{\mathbb{R}^d} |a(\omega)|\,\mathrm{d}\rho(\omega) < +\infty \right\}.$$

Due to the scaling invariance of the ReLU function, the norm of the Barron space can be defined by

$$\|f\|_{\mathcal{B}} = \inf_{a,\rho} \int_{\mathbb{S}^{d-1}} |a(\omega)|\,\mathrm{d}\rho(\omega),$$

where the infimum is taken over all possible $\rho \in \mathcal{P}(\mathbb{S}^{d-1})$ and $a \in L^1(\rho)$ such that Eq. (6.2) is satisfied. We have the following relationship between the RKHS and Barron space:

$$\mathcal{B} = \bigcup_{\pi \in \mathcal{P}(\mathbb{S}^{d-1})} \mathcal{H}_{k_\pi}, \tag{6.3}$$

where $k_\pi(x,y) = \mathbb{E}_{\omega\sim\pi}[\sigma(\omega^{\mathrm{T}} x)\sigma(\omega^{\mathrm{T}} y)]$. We shall also point out that compared to the RKHS, the Barron space is much larger in high dimensions, see [21, Example 4.3]. We refer to [17, 20] for more details and properties of the Barron space. For a detailed comparison between the NTK and the Barron space, see [16, 18].

We can approximate the target function in Barron space with the two-layer neural networks

$$\mathcal{H}_m = \left\{ f(x,\theta) = \frac{1}{m}\sum_{i=1}^{m} a_i \sigma\big(\omega_i^{\mathrm{T}} x\big) : \theta = (a_1,\omega_1,\ldots,a_m,\omega_m), a_i \in \mathbb{R}, \omega_i \in \mathbb{S}^{d-1}, 1 \le i \le m \right\}.$$

Then we have that for any $f^* \in \mathcal{B}$ and $\mu \in \mathcal{P}(\mathbb{R}^d)$

$$\inf_{f_m \in \mathcal{H}_m} \mathbb{E}_{x\sim\mu}|f^*(x) - f_m(x)|^2 \le \frac{\|f\|_{\mathcal{B}}^2}{m}.$$

Similar to the kernel ridge regression, we can define the following loss function:

$$\mathcal{L}_n(\theta) = \frac{1}{n}\sum_{j=1}^{n} \max\left\{ (\ln n)^2, [y_i - f(x_i,\theta)]^2 \right\} + \frac{\lambda}{m}\sum_{i=1}^{m} |a_i|\|\omega_i\|$$

to obtain the estimator $\hat{f}$. Under proper condition, we can obtain that

$$\|f^* - \hat{f}\|_{L^2(\mu)} = \tilde{O}\left(\|f\|_{\mathcal{B}}\left[m^{-\frac{1}{2}} + n^{-\frac{1}{4}}\right]\right),$$

if we set $\lambda = \tilde{O}(n^{-1/2})$, see [19] for details. However, it is still not clear how to efficiently compute the estimator $\hat{f}$, see, e.g., [16,18]. More work is needed to further understand the optimization error in this case.

6.2 Reinforcement learning with nonlinear function approximation

We first discuss the approximation error in the nonlinear setting. To this end, we generalize Theorem 5.1 to the case of RKHS.

Theorem 6.1. *Assume that $\mathcal{S} \times \mathcal{A}$ is a compact set. Let ρ be a probability distribution on $\mathcal{S} \times \mathcal{A}$, we will use $\{\lambda_i\}_{i=1}^{+\infty}$ and $\{\psi_i\}_{i=1}^{+\infty}$ to denote the eigenvalues and eigenfunctions of the operator*

$$(\mathcal{K}_\rho g)(x) := \int_{\mathcal{S}\times\mathcal{A}} k(x, x')g(x')\,\mathrm{d}\rho(z')$$

from $L^2(\rho)$ to $L^2(\rho)$. We further require that $\{\lambda_i\}_{i=1}^{+\infty}$ is nonincreasing and $\{\psi_i\}_{i=1}^{+\infty}$ is orthonormal in $L^2(\rho)$. Then, the following statements are equivalent:

1. *For any $h \in [H]$, $r(h,\cdot) \in \mathcal{H}_k$ and there exist signed measures $\{\mu_h^i\}_{i=1}^{+\infty}$ over $\mathcal{S}$, such that for any (s,a)*

$$P(\cdot\,|\,h,s,a) = \sum_{i=1}^{+\infty} \psi_i(s,a)\mu_h^i(\cdot)$$

in the sense that for any $f \in C(\mathcal{S})$

$$\int_{\mathcal{S}} f(s')\,\mathrm{d}P(s'|h,s,a) = \sum_{i=1}^{+\infty} \int_{\mathcal{S}} f(s')\,\mathrm{d}\mu_h^i(s')\psi_i(s,a),$$

where the convergence is in the sense of $L^2(\rho)$. Moreover, for any $f \in C(\mathcal{S})$

$$\sum_{i=1}^{+\infty} \frac{1}{\lambda_i}\left|\int_{\mathcal{S}} f(s')\,\mathrm{d}\mu_h^i(s')\right|^2 \le C\|f\|_{C(\mathcal{S})}^2$$

for a constant $C > 0$.

2. *For any $f \in C(\mathcal{S})$ and $h \in [H]$,*

$$(\mathcal{T}_h f)(s,a) = r(h,s,a) + \mathbb{E}_{s'\sim P(\cdot\,|\,h,s,a)}[f(s')] \in \mathcal{H}_k$$

and $\|\mathcal{T}_h f\|_{\mathcal{H}_k} \le C'[\|f\|_{C(\mathcal{S})} + 1]$ for a constant $C' > 0$.

Proof. We will use the following representation of the RKHS norm:

$$\|g\|_k^2 = \sum_{i=1}^{+\infty} \frac{1}{\lambda_i}|\langle g, \psi_i\rangle_{L^2(\rho)}|^2. \tag{6.4}$$

See, e.g., [9, Section 2.1].

1 ⇒ 2: For any $f \in C(\mathcal{S})$,

$$(\mathcal{T}_h f)(s,a) = r(h,s,a) + \sum_{i=1}^{+\infty} \int_{\mathcal{S}} f(s') \, \mathrm{d}\mu_h^i(s') \psi_i(s,a).$$

Let

$$g_{f,h} = \sum_{i=1}^{+\infty} \int_{\mathcal{S}} f(s') \, \mathrm{d}\mu_h^i(s') \psi_i,$$

then

$$\langle g_{f,h}, \psi_i \rangle_{L^2(\rho)} = \int_{\mathcal{S}} f(s') \, \mathrm{d}\mu_h^i(s')$$

for any $i \in \mathbb{N}^+$. Therefore $g_{f,h} \in \mathcal{H}_k$ and

$$\|g_{f,h}\|_{\mathcal{H}_k} = \sqrt{\sum_{i=1}^{+\infty} \frac{1}{\lambda_i} \left| \int_{\mathcal{S}} f(s') \, \mathrm{d}\mu_i(s') \right|^2} \leq \sqrt{C} \|f\|_{C(\mathcal{S})}.$$

Hence,

$$\|\mathcal{T}_h f\|_{\mathcal{H}_k} \leq \|r(h,\cdot)\|_{\mathcal{H}_k} + \|g_{f,h}\|_{\mathcal{H}_k} \leq C' \left[1 + \|f\|_{C(\mathcal{S})}\right].$$

2 ⇒ 1: Choose $f = 0$ we know that $r(h,\cdot) \in \mathcal{H}_k$ and $\|r(h,\cdot)\|_{\mathcal{H}_k} \leq C'$. Let

$$g_{f,h} = \mathbb{E}_{s' \sim P(\,\cdot\,|h,s,a)}[f(s')],$$

then for any $f \in C(\mathcal{S})$, $g_{f,h} \in \mathcal{H}_k$. If $\|f\|_{C(\mathcal{S})} = 1$, we have

$$\|g_{f,h}\|_{\mathcal{H}_k} \leq \|\mathcal{T}_h f\|_{\mathcal{H}_k} + \|r(h,\cdot)\|_{\mathcal{H}_k} \leq 3C'.$$

Then by the linearity of $g_{f,h}$ with respect to f, we have that for any $f \in C(\mathcal{S})$,

$$\|g_{f,h}\|_{\mathcal{H}_k} \leq 3C' \|f\|_{C(\mathcal{S})}.$$

Noticing that $g_{f,h} \in \mathcal{H}_k \subset L^2(\rho)$, we know that for any $f \in C(\mathcal{S})$,

$$\int_{\mathcal{S}} f(s') \, \mathrm{d}P(s'|h,s,a) = g_{f,h} = \sum_{i=1}^{+\infty} \int_{\mathcal{S}\times\mathcal{A}} g_{f,h}(s',a') \psi_i(s',a') \, \mathrm{d}\rho(s',a') \psi_i(s,a).$$

Noticing that

$$\left| \int_{\mathcal{S}\times\mathcal{A}} g_{f,h}(s',a') \psi_i(s',a') \, \mathrm{d}\rho(s',a') \right| \leq \|g_{f,h}\|_{L^2(\rho)} \leq \sqrt{\lambda_1} \|g_{f,h}\|_{\mathcal{H}_k} \leq 3C' \sqrt{\lambda_1} \|f\|_{C(\mathcal{S})}.$$

We can then use the Riesz representation theorem to obtain that there exists signed measures $\{\mu_h^i\}_{i=1}^{+\infty}$ such that

$$\int_{\mathcal{S}} g_{f,h}(s',a')\psi_i(s',a')\,\mathrm{d}\rho(s',a') = \int_{\mathcal{S}} f(s')\,\mathrm{d}\mu_h^i(s').$$

Therefore,

$$P(\,\cdot\,|\,h,s,a) = \sum_{i=1}^{+\infty} \mu_h^i(\cdot)\psi_i(s,a).$$

Moreover,

$$\|g_{f,h}\|_{\mathcal{H}_k}^2 = \sum_{i=1}^{+\infty} \frac{1}{\lambda_i}\left|\int_{\mathcal{S}} f(s')\,\mathrm{d}\mu_h^i(s')\right|^2 \le 9(C')^2\|f\|_{C(\mathcal{S})}^2 =: C\|f\|_{C(\mathcal{S})}^2.$$

The proof is complete. □

Similar to the discussions in the linear setting, Theorem 6.1 provides a necessary and sufficient condition to ensure Property 4 in the RKHS setting. The extension of Theorem 6.1 to the Barron space is not yet clear, mainly due to the lack of a representation of the Barron norm that is similar to (6.4). Nevertheless, it is still worthwhile to consider only sufficient conditions for Property 4 in Barron space. [40] introduce one such condition: for any $h \in [H]$,

$$\|r(h,\cdot)\|_{\mathcal{B}} < +\infty, \quad \sup_{s'\in\mathcal{S}} \|p(h,s',\cdot)\|_{\mathcal{B}} < +\infty, \tag{6.5}$$

where $p(h,s',s,a) = \mathrm{d}P(s'\,|\,h,s,a)/\,\mathrm{d}\rho_h(s')$ and ρ_h is a probability distribution on $\mathcal{S}$. Similar to the situation in the linear setting, these conditions rule out many interesting deterministic MDPs, since these MDPs can only visit countably infinite states when $h \ge 2$. Additionally, there has been little investigation into the concrete conditions on the transition probability and reward function that ensure Property 2 and Property 3 in Section 4.1 in the nonlinear setting.

With theoretical results in supervised learning and analysis on the approximation error of RL in the nonlinear setting, it remains to develop tools to handle the distribution mismatch phenomenon in the nonlinear setting, which leads to a paramount difference in RL analysis between tabular/linear settings and nonlinear settings. In the tabular and linear settings, the L^∞ estimation and UCB estimation are used to handle the distribution mismatch. The L^∞ estimation or UCB estimation of the value function is obtained such that the error under any distribution can be controlled. However, as pointed out in [33,40] and the theorem below, both L^∞ and UCB estimations will suffer from the curse of dimensionality for high-dimensional NTK, Barron space, and many common RKHSs. This challenge reveals at least one essential difficulty of RL problems in the nonlinear setting compared to the tabular and linear settings.

Theorem 6.2. *Given an RKHS $\mathcal{H}_k$ on $\mathcal{X}$ associated with a continuous kernel k (assuming that $\sup_{x\in\mathcal{X}} k(x,x) \le 1$) and any $x_1,\dots,x_n \in \mathcal{X}$, let $\mathcal{H}_k^1$ be the unit ball of $\mathcal{H}_k$ and $\mathcal{G}_n : \mathcal{H}_k^1 \to C(\mathcal{X})$ be a mapping satisfying*

$$\mathcal{G}_n f = \mathcal{G}_n f', \quad \forall f,f' \in \mathcal{H}_k^1 \quad \textit{such that} \quad f(x_i) = f'(x_i), \quad i = 1,\dots,n. \tag{6.6}$$

For any given distribution ρ on $\mathcal{X}$, let $\{\lambda_i\}_{i=1}^{+\infty}$ be the nonincreasing eigenvalues of the mapping $K_\rho : L^2(\rho) \to L^2(\rho)$

$$(K_\rho g)(x) = \int_{\mathcal{X}} k(x, x')g(x')\,\mathrm{d}\rho(x').$$

The following two statements hold true:

1. *(L^∞ estimation)*

$$\sup_{f\in\mathcal{H}_k^1} \|f - \mathcal{G}_n f\|_\infty \geq \left(\sum_{i=n+1}^{+\infty} \lambda_i\right)^{\frac{1}{2}}.$$

2. *(UCB estimation) If $\mathcal{G}_n$ additionally satisfies that*

$$\mathcal{G}_n f(x) \geq f(x), \quad \forall f \in \mathcal{H}_k^1, \quad x \in \mathcal{X}, \tag{6.7}$$

then,

$$\sup_{f\in\mathcal{H}_k^1} \mathbb{E}_\rho[\mathcal{G}_n f - f] \geq \sum_{i=n+1}^{+\infty} \lambda_i.$$

We can interpret the mapping $\mathcal{G}_n$ in Theorem 6.2 as an abstraction of a function approximation algorithm that takes the function values at n points, $x_1, \ldots, x_n$ as input and returns a continuous function. The requirement in Theorem 6.2 is naturally satisfied: if two target functions have the same values at $x_1, \ldots, x_n$, the function approximation result will be identical. From the definition, UCB estimation must satisfy condition (6.7) as the UCB estimation should give a pointwise upper bound of the target function. Therefore, Theorem 6.2 gives the lower bound of the worst-case error of both L^∞ and UCB estimation based on the eigenvalue decay of the kernel. As pointed out in [40], if we choose $\mathcal{X} = \mathbb{S}^{d-1}$, the unit ball in $\mathbb{R}^d$ and ρ the uniform distribution on $\mathbb{S}^{d-1}$, the eigenvalue decay $\sum_{i=n+1}^{+\infty} \lambda_i$ of the following RKHSs:

$$k(x, x') = \begin{cases} k_{Lap}(x, x') = \exp(-\|x - x'\|), \\ k_{NTK}(x, x') = \mathbb{E}_{\omega\sim\pi}(x \cdot x')\sigma'(\omega \cdot x)\sigma'(\omega \cdot x'), \\ k_\pi(x, x') = \mathbb{E}_{\omega\sim\pi}\sigma(\omega \cdot x)\sigma(\omega \cdot x') \end{cases}$$

is $n^{-\alpha/d}$ for some universal constant α. Here π is also the uniform distribution on $\mathbb{S}^{d-1}$. Therefore, if the target function lies in the RKHS associated with the Laplacian kernel or NTK, according to the first argument in Theorem 6.2, the L^∞ estimation suffers from the curse of dimensionality: the number of points needed to achieve an error tolerance scales exponentially with respect to the dimension d. Since the $\mathcal{H}_{k_\pi}$ is the subspace of the Barron space $\mathcal{B}$ (Eq. (6.3)), the L^∞ and UCB estimation in the Barron space also suffer from the curse of dimensionality.

Proof of Theorem 6.2. We first prove that

$$\mathbb{E}_{x\sim\rho} \sup_{f\in\mathcal{H}_{k'}^1 f(x_1)=\cdots=f(x_n)=0} |f(x)|^2 \geq \sum_{l=n+1}^{+\infty} \lambda_l. \tag{6.8}$$

DOI https://doi.org/10.4208/jml.230105 | Generated on 2025-04-20 07:59:17

Notice that

$$\sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} f(x) = \sup_{\|f\|_{\mathcal{H}}\le 1, \langle f, k(x_i,\cdot)\rangle_{\mathcal{H}_k}=0, 1\le i\le n} \langle f, k(x,\cdot)\rangle_{\mathcal{H}_k}$$
$$= \inf_{c_1,\ldots,c_n} \left\| k(x,\cdot) - \sum_{i=1}^n c_i k(x_i,\cdot) \right\|_{\mathcal{H}_k}.$$

Then, let $\phi_1,\ldots,\phi_n$ be the Gram-Schmidt orthonormalization of $\{k(x_1,\cdot),\ldots,k(x_n,\cdot)\}$ in $\mathcal{H}$, then

$$\inf_{c_1,\ldots,c_n} \left\| k(x,\cdot) - \sum_{i=1}^n c_i k(x_i,\cdot) \right\|_{\mathcal{H}_k}^2 = k(x,x) - \sum_{i=1}^n \phi_i^2(x).$$

Therefore,

$$\mathbb{E}_{x\sim\rho} \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} |f(x)|^2 = \mathbb{E}_{x\sim\rho} k(x,x) - \sum_{i=1}^n \mathbb{E}_{x\sim\rho}\phi_i^2(x).$$

Let $\{\psi_l\}_{l=1}^{+\infty}$ be the eigenfunctions corresponding to $\{\lambda_l\}_{l=1}^{+\infty}$, which is an orthonormal basis in $L^2(\rho)$. Let

$$c_l = \sum_{i=1}^n \left(\mathbb{E}_{x\sim\rho}\psi_l(x)\phi_i(x)\right)^2,$$

then,

$$\lambda_l = \lambda_l^2 \|\psi_l\|_{\mathcal{H}_k}^2 \ge \lambda_l^2 \sum_{i=1}^n \left(\langle\psi_l,\phi_i\rangle_{\mathcal{H}_k}\right)^2 = \lambda_l^2 \sum_{i=1}^n \frac{(\mathbb{E}_{x\sim\rho}\psi_l(x)\phi_i(x))^2}{\lambda_l^2} = c_l \ge 0,$$

and

$$\sum_{l=1}^{+\infty} \frac{c_l}{\lambda_l} = \sum_{i=1}^n \sum_{l=1}^{+\infty} \frac{(\mathbb{E}_{x\sim\rho}\psi_l(x)\phi_i(x))^2}{\lambda_l} = \sum_{i=1}^n \|\phi_i\|_{\mathcal{H}}^2 = n.$$

Hence,

$$\sum_{i=1}^n \mathbb{E}_{x\sim\rho}\phi_i^2(x) = \sum_{l=1}^{+\infty} c_l \le \sum_{l=1}^n \lambda_l.$$

The famous Mercer decomposition states that

$$k(x,x') = \sum_{i=1}^{+\infty} \lambda_i \psi_i(x)\psi_i(x').$$

Therefore, with the observation that

$$\mathbb{E}_{x\sim\rho} k(x,x) = \sum_{l=1}^{+\infty} \lambda_l,$$

we obtain inequality (6.8).

DOI https://doi.org/10.4208/jml.230105 | Generated on 2025-04-20 07:59:17

To prove the first argument, first notice that

$$\sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} \|f\|_\infty = \sup_{x\in\mathcal{X}} \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} |f(x)|$$
$$\geq \left(\mathbb{E}_{x\sim\rho} \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} |f(x)|^2\right)^{\frac{1}{2}} \geq \left(\sum_{l=n+1}^{+\infty} \lambda_l\right)^{\frac{1}{2}}.$$

Then, noticing that for any $f \in \mathcal{H}_k^1$ such that $f(x_1) = \cdots = f(x_n) = 0$, we have $\mathcal{G}_n f = \mathcal{G}_n(-f)$. Therefore,

$$\sup_{f\in\mathcal{H}_k^1} \|f - \mathcal{G}_n f\|_\infty = \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} \frac{\|f - \mathcal{G}_n f\|_\infty + \| - f - \mathcal{G}_n f\|_\infty}{2}$$
$$\geq \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} \|f\|_\infty,$$

which concludes the proof.

For the second argument, let $f_0 = 0$. For any $f \in \mathcal{H}_k^1$ such that $f(x_1) = \cdots = f(x_n) = 0$, we have

$$\mathcal{G}_n f_0(x) = \mathcal{G}_n f(x) \geq f(x)$$

for any $x \in \mathcal{X}$. Therefore,

$$\mathcal{G}_n f_0(x) \geq \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} f(x) = \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} |f(x)|.$$

Combining the fact that

$$\sup_{f\in\mathcal{H}_k^1} |f(x)| = \sup_{f\in\mathcal{H}_k^1} |\langle f, k(x,\cdot)\rangle_{\mathcal{H}_k}| = \|k(x,\cdot)\|_{\mathcal{H}_k} = k(x,x) \leq 1,$$

we know that

$$\mathcal{G}_n f_0(x) \geq \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} |f(x)|^2.$$

Therefore,

$$\sup_{f\in\mathcal{H}_k^1} \mathbb{E}_{x\sim\rho}[\mathcal{G}_n f - f] \geq \mathbb{E}_{x\sim\rho}\mathcal{G}_n f_0(x) \geq \mathbb{E}_{x\sim\rho} \sup_{f\in\mathcal{H}_k^1, f(x_1)=\cdots=f(x_n)=0} |f(x)|^2 \geq \sum_{i=n+1}^{+\infty} \lambda_i.$$

The proof is complete. □

Theorem 6.2 indicates that the L^∞ or UCB estimation is too strong as a requirement to pursue in the high-dimensional cases with nonlinear function approximation. Therefore, to obtain meaningful results in the nonlinear setting, besides the assumption that ensures the value or policy function can be approximated by the kernel or neural functions, some additional assumptions are needed to handle the distribution mismatch phenomenon. Based on the assumptions used, most of the existing works addressing this

difficulty can be divided into two categories. The first category [58, 59] assumes the fast eigenvalue decay of the kernel such that the L^∞ and UCB estimation still provide a meaningful bound in high dimensions. The second category [2, 12, 22, 23, 40, 55] requires the following concentration coefficient condition: for any $h \in [H]$, there exists a distribution ν_h such that for any policy π, the corresponding state-action distribution $\rho_{h,P,\pi,\mu}$ satisfies

$$\left\| \frac{\mathrm{d}\rho_{h,P,\pi,\mu}}{\mathrm{d}\nu_h} \right\|_{L^2(\nu_h)} \leq C,$$

where $C > 0$ is a universal constant. Under this assumption, an L^2 estimation under ν_h is sufficient to handle distribution mismatch since we can control the estimation error under the state-action distributions generated by all possible policies, including the optimal policy. This assumption is commonly used to study the convergence of the fitted Q-iteration algorithm (Algorithm 1) [12, 22, 23, 40]. In the episodic setting, due to the lack of a generative model, we need to additionally assume that ν_h is the state-action distribution $\rho_{h,P,\bar{\pi},\mu}$ for a policy $\bar{\pi}$ [55].

To better capture the influence of distribution mismatch in the RL problem, [39] introduce a quantity called perturbational complexity by distribution mismatch for a large class of the RL problems in the nonlinear setting when a generative model is accessible. This quantity can give both the lower bound and upper bound of the sample complexity of these RL problems and hence measure their difficulty. Moreover, both fast eigenvalue decay and finite concentration coefficient can lead to small perturbational complexity by distribution mismatch [39, Propositions 2 and 3] and hence the results in [39] generalize both categories of the previous results in the nonlinear setting.

The formal definition of the perturbational complexity by distribution mismatch is given as follows.

Definition 6.1. *(i) For any set Π consisting of probability distributions on $\mathcal{S} \times \mathcal{A}$, we define a semi-norm $\|\cdot\|_\Pi$ on $C(\mathcal{S} \times \mathcal{A})$*

$$\|g\|_\Pi := \sup_{\rho \in \Pi} \left| \int_{\mathcal{S}\times\mathcal{A}} g(s,a)\, \mathrm{d}\rho(s,a) \right|.$$

We call this semi-norm Π-norm.

(ii) Given a Banach space $\mathcal{B}$, a probability distribution $\nu \in \mathcal{P}(\mathcal{S} \times \mathcal{A})$ and a positive constant $\epsilon > 0$, we define a ν-perturbation space $\mathcal{B}_{\epsilon,\nu}$ with scale ϵ, as follows:

$$\mathcal{B}_{\epsilon,\nu} := \{g \in \mathcal{B} \colon \|g\|_{\mathcal{B}} \leq 1, \|g\|_{L^2(\nu)} \leq \epsilon\}.$$

(iii) The perturbation response by distribution mismatch is defined as the radius of $\mathcal{B}_{\epsilon,\nu}$ under Π-norm,

$$\mathcal{R}(\Pi, \mathcal{B}, \epsilon, \nu) := \sup_{g \in \mathcal{B}_{\epsilon,\nu}} \|g\|_\Pi.$$

We consider an RL problem whose underlying MDP belongs to a family of MDPs

$$\mathcal{M} = \{M_\theta = (\mathcal{S}, \mathcal{A}, P, r_\theta, H, \mu) \colon \theta \in \Theta\},$$

Algorithm 4 Fitted reward algorithm

Input: MDP family $\mathcal{M}$, generative model of MDP $(\mathcal{S}, \mathcal{A}, H, P, r_\theta, \mu)$, sampling distribution $\hat{\nu} = \arg\min_{\nu \in \mathcal{P}(\mathcal{S}\times\mathcal{A})} \mathcal{R}(\Pi(P_0, \mu), \mathcal{H}_k, n^{-1/4}, \nu)$.

for $h = 1, 2, \ldots, H$ **do**

Sample $(s_1, a_1), \ldots, (s_n, a_n)$ i.i.d. from $\hat{\nu}$.

Sample $r_h^1, \ldots, r_h^n$ from $\mathcal{N}(r_\theta(h, s_1, a_1), 1), \ldots, \mathcal{N}(r_\theta(h, s_n, a_n), 1)$, respectively.

Compute $\hat{r}_\theta(h, \cdot)$ as the minimizer of the optimization problem

$$\min_{\|r\|_{\mathcal{B}} \le 1} \sum_{i=1}^{n} \left[r(s_i, a_i) - r_h^i \right]^2.$$

end

Collect the fitted reward function to form the MDP $(\mathcal{S}, \mathcal{A}, H, P, \hat{r}_\theta, \mu)$, of which both reward function and transition are known. Denote it as $\hat{M}_\theta$.

Output: $\hat{\pi}_\theta$ as the optimal policy of $\hat{M}_\theta$.

where $\mathcal{S}, \mathcal{A}, P, H$ and μ are common state space, action space, transition probability[1], length of each episode, and initial distribution. The unknown reward function lies in the unit ball of a Banach space $\mathcal{B}$

$$\{r_\theta, \theta \in \Theta\} = \mathcal{B}^1.$$

In the generative model setting, [39] prove that any RL algorithm $J_n : \theta \to \mathbb{R}$ on $\mathcal{M}$ with at most n accesses to the generative model satisfies that

$$\sup_{\theta \in \Theta} \mathbb{E}|J_n(\theta) - J^*(M_\theta)| \ge \frac{1}{12} \Delta_{\mathcal{M}}\left(n^{-\frac{1}{2}}\right),$$

where

$$\Delta_{\mathcal{M}}(\epsilon) = \inf_{\nu \in \mathcal{P}(\mathcal{S}\times\mathcal{A})} \mathcal{R}\left(\Pi(P, \mu), \mathcal{B}, \epsilon, \nu\right).$$

Therefore, the perturbational complexity by distribution mismatch gives a lower bound for RL problems on $\mathcal{M}$. Note that in [39], it is assumed that we can only obtain a noisy reward with a standard normal noise in the generative model, rather than the exact reward. On the other hand, if $\mathcal{B}$ is the Barron space or an RKHS, then the output $\hat{\pi}_\theta$ of Algorithm 4 satisfies

$$\sup_{\theta \in \Theta} |J(M_\theta, \hat{\pi}_\theta) - J^*(M_\theta)| \le \tilde{O}\left(H \Delta_{\mathcal{M}}(n^{-\frac{1}{4}})\right).$$

Therefore, the perturbational complexity by distribution mismatch also gives an upper bound for RL problems on $\mathcal{M}$.

[1] In [39], the general case where the transition probability is unknown is also considered and fitted Q-iteration algorithm (Algorithm 1) with kernel function approximation is studied in this case. Here for brevity we only discuss the case where the transition probability is known.

DOI https://doi.org/10.4208/jml.230105 | Generated on 2025-04-20 07:59:17

The perturbational complexity by distribution mismatch can also be used to construct various RL problems that suffer from the curse of dimensionality [39]. The first example involves a state space $\mathcal{S}$ consisting of a single point s_0, while the action space $\mathcal{A}$ is $\mathbb{S}^{d-1}$ and $H = 1$. In this setting, the RL problem essentially aims to find the maximum value of the reward function lying in the unit ball of $\mathcal{B}$ based on the values of n points. We can prove that when $\mathcal{B}$ is the Barron space and the RKHS corresponding to the Laplacian kernel and NTK, the convergence rate can be bounded below by the eigenvalue decay. Therefore, if we consider the RKHS corresponding to the Laplacian kernel or neural tangent kernel, the convergence rate suffers from the curse of dimensionality. We can then conclude that if we want to solve RL problems with high dimensional action space, we need to assume the decay of the eigenvalue is fast enough to break the curse of dimensionality. The other example involves a high-dimensional state space and finite action space. For any dimension $d \geq 2$, length of each episode $H \in \mathbb{N}^+$ and positive constant $\delta > 0$, we define an MDP family $\mathcal{M}_{d,H,\delta}$ as follows:

$$\begin{aligned}
&\mathcal{S} = \mathbb{S}^{d-1}, \quad \mathcal{A} = \{0,1\}, \quad H = H, \quad \mu = \mathrm{Uniform}_{\mathbb{S}^{d-1}}, \\
&\{r_{\theta_r} : \theta_r \in \Theta_r\} = \big\{r : \|r(h,\cdot)\|_{\mathcal{H}_k} \leq 1, \forall h \in [H]\big\}, \\
&k\big((s,a),(s',a')\big) = \exp(-\|s-s'\|), \quad P(\cdot \,|\, h,s,a) = \delta_{T_{a,h}s}(\cdot), \\
&T_{a,h}s = \begin{cases} (\phi_1,\ldots,\phi_{h_d}+\delta,\ldots,\phi_d), & \text{when} \quad a = 0, \\ (\phi_1,\ldots,\phi_{h_d}-\delta,\ldots,\phi_d), & \text{when} \quad a = 1, \end{cases}
\end{aligned}$$

where $h_d = h \mod d$ and we use the spherical coordinates $(\phi_1,\ldots,\phi_d)$ to denote the points on $\mathbb{S}^{d-1}$. Then we can show that there exist no universal constants $\alpha, \beta > 0$ and constant $C_d > 0$ only depending on d such that

$$\sup_{\delta>0} \Delta_{\mathcal{M}_{d,H,\delta}}\big(n^{-\frac{1}{2}}\big) \leq C_d H^{\alpha} \left(\frac{1}{n}\right)^{\beta}$$

holds for all $n, H \in \mathbb{N}^+$ and $d \geq 2$. Therefore, the above RL problems cannot be solved without the curse of dimensionality.

7 Discussion and conclusion

In this paper, we review existing research on reinforcement learning with function approximation. The results in the tabular and linear settings are well-developed because methods such as L^∞ and UCB estimation can be used to handle the phenomenon of distribution mismatch. When a generative model is available, the perturbational complexity by distribution mismatch can be used to measure the impact of distribution mismatch and assess the difficulty of reinforcement learning problems in the nonlinear setting. However, it remains unclear how to extend these results to the episodic setting, and it is still an open question how to use perturbational complexity information to guide the design of efficient reinforcement learning algorithms in practice.

Approximation error is also an important topic in RL, especially in the nonlinear setting. Apart from the Theorem 5.1 for linear space, Theorem 6.1 for RKHS, and the condition (6.5) for Barron space, there are limited results in this area, particularly for deterministic MDPs. We remark that the solution of the continuous-time Hamilton-Jacobi-Bellman equation, which is the value function of continuous-time MDPs, can be approximated by neural networks, see, e.g., [26]. However, it is not clear whether this result can be applied to discrete-time MDPs. Computational issues are another important topic in reinforcement learning, particularly for reinforcement learning with neural function approximation. The convergence of the gradient descent method of neural networks in the mean field regime is still not well-understood. We hope that further research will be conducted on these topics.

Finally, a significant gap exists between the current theory and practice of reinforcement learning, even in the absence of function approximation. The majority of theoretical results focus on algorithms that employ strategic exploration, such as the UCB method [8, 28, 29, 59]. However, RL algorithms in practice often utilize the random exploration. Theoretical research suggests that, in the worst-case scenario, RL with random exploration exhibits exponential difficulty with respect to the horizon [14], which does not accurately explain practical performance. While some theoretical studies [35, 38] have examined instance-based bounds by identifying specific RL problem properties that lead to better performance than the worst case when random exploration is employed, these properties do not fully account for the success of all practical RL problems, nor do they address function approximation. Furthermore, many practical techniques, such as reward shaping, experience replay, and pre-trained policies, have not been sufficiently explored in theoretical research to explain their positive impact on RL algorithm performance. It is essential for future research to bridge the gap between theory and practice, particularly in the presence of function approximation.

References

[1] A. Agarwal, M. Henaff, S. Kakade, and W. Sun, PC-PG: Policy cover directed exploration for provable policy gradient learning, *Adv Neural Inf Process Syst*, **33**:13399–13412, 2020.

[2] A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan, On the theory of policy gradient methods: Optimality, approximation, and distribution shift, *J Mach Learn Res*, **22**(98):1–76, 2021.

[3] Z. Allen-Zhu, Y. Li, and Z. Song, A convergence theory for deep learning via over-parameterization. In: *International Conference on Machine Learning*, PMLR, 242–252, 2019.

[4] A. Antos, C. Szepesvári, and R. Munos, Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path, *Mach Learn*, **71**(1):89–129, 2008.

[5] N. Aronszajn, Theory of reproducing kernels, *Trans. Amer. Math. Soc.*, **68**(3):337–404, 1950.

[6] S. Arora, S. S. Du, W. Hu, Z. Li, R. R. Salakhutdinov, and R. Wang, On exact computation with an infinitely wide neural net, *Adv Neural Inf Process Syst*, **32**, 2019.

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

[8] M. G. Azar, I. Osband, and R. Munos, Minimax regret bounds for reinforcement learning. In: *International Conference on Machine Learning*, PMLR, 263–272, 2017.
[9] F. Bach, On the equivalence between kernel quadrature rules and random feature expansions, *J Mach Learn Res*, **18**(1):714–751, 2017.
[10] Q. Cai, Z. Yang, C. Jin, and Z. Wang, Provably efficient exploration in policy optimization. In: *International Conference on Machine Learning*, PMLR, 1283–1294, 2020.
[11] A. Caponnetto and E. De Vito, Optimal rates for the regularized least-squares algorithm, *Found. Comput. Math.*, **7**(3):331–368, 2007.
[12] J. Chen and N. Jiang, Information-theoretic considerations in batch reinforcement learning. In: *International Conference on Machine Learning*, PMLR, 1042–1051, 2019.
[13] C. Dann, T. Lattimore, and E. Brunskill, Unifying PAC and regret: Uniform PAC bounds for episodic reinforcement learning, *Adv Neural Inf Process Syst*, **30**:5713–5723, 2017.
[14] C. Dann, Y. Mansour, M. Mohri, A. Sekhari, and K. Sridharan, Guarantees for epsilon-greedy reinforcement learning with function approximation. In: *International Conference on Machine Learning*, PMLR, 4666–4689, 2022.
[15] O. D. Domingues, P. Ménard, M. Pirotta, E. Kaufmann, and M. Valko, Regret bounds for kernel-based reinforcement learning, *arXiv:2004.05599*, 2020.
[16] W. E, C. Ma, S. Wojtowytsch, and L. Wu, Towards a mathematical understanding of neural network-based machine learning: What we know and what we don't, *arXiv:2009.10713*, 2020.
[17] W. E, C. Ma, and L. Wu, Barron spaces and the compositional function spaces for neural network models, *arXiv:1906.08039*, 2019.
[18] W. E, C. Ma, L. Wu, A comparative analysis of optimization and generalization properties of two-layer neural network and random feature models under gradient descent dynamics, *Sci. China Math*, 2019.
[19] W. E, C. Ma, and L. Wu, A priori estimates of the population risk for two-layer neural networks, *Commun. Math. Sci.*, **17**:1407–1425, 2019.
[20] W. E, C. Ma, and L. Wu, The Barron space and the flow-induced function spaces for neural network models, *Constr. Approx.*, **55**(1):369–406, 2022.
[21] W. E, S. Wojtowytsch, Kolmogorov width decay and poor approximators in machine learning: Shallow neural networks, random feature models and neural tangent kernels, *Res. Math. Sci.*, **8**(1):1–28, 2021.
[22] J. Fan, Z. Wang, Y. Xie, and Z. Yang, A theoretical analysis of deep Q-learning. In: *Learning for Dynamics and Control*, PMLR, 486–489, 2020.
[23] A.-m. Farahmand, M. Ghavamzadeh, C. Szepesvári, and S. Mannor, Regularized policy iteration with nonparametric function spaces, *J Mach Learn Res*, **17**(139):1–66, 2016.
[24] B. Gao and L. Pavel, On the properties of the softmax function with application in game theory and reinforcement learning, *arXiv:1704.00805*, 2017.
[25] M. Gheshlaghi Azar, R. Munos, and H. J. Kappen, Minimax PAC bounds on the sample complexity of reinforcement learning with a generative model, *Mach Learn*, **91**(3):325–349, 2013.
[26] M. Hutzenthaler, A. Jentzen, and T. Kruse, Overcoming the curse of dimensionality in the numerical approximation of parabolic partial differential equations with gradient-dependent nonlinearities, *Found. Comput. Math.*, **22**(4):905–966, 2022.
[27] A. Jacot, F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks, *arXiv:1806.07572*, 2018.
[28] C. Jin, Z. Allen-Zhu, S. Bubeck, and M. I. Jordan, Is Q-learning provably efficient?, *Adv Neural Inf Process Syst*, **31**:4863–4873, 2018.
[29] C. Jin, Z. Yang, Z. Wang, and M. I. Jordan, Provably efficient reinforcement learning with linear function approximation. In: *Conference on Learning Theory*, 2137–2143, 2020.
[30] S. M. Kakade, A natural policy gradient, *Adv Neural Inf Process Syst*, **14**, 2001.
[31] S. Kakade and J. Langford, Approximately optimal approximate reinforcement learning. In: *Proceedings of 19th International Conference on Machine Learning*, Citeseer, 2002.
[32] J. Kober, J. A. Bagnell, and J. Peters, Reinforcement learning in robotics: A survey, *Int. J. Robot. Res.*, **32**(11):1238–1274, 2013.

[33] F. Y. Kuo, G. W. Wasilkowski, and H. Woźniakowski, Multivariate L_∞ approximation in the worst case setting over reproducing kernel Hilbert spaces, *J. Approx. Theory*, **152**(2):135–160, 2008.
[34] M. G. Lagoudakis and R. Parr, Least-squares policy iteration, *J Mach Learn Res*, **4**:1107–1149, 2003.
[35] C. Laidlaw, S. Russell, and A. Dragan, Bridging RL theory and practice with the effective horizon, *arXiv:2304.09853*, 2023.
[36] T. Lattimore, C. Szepesvari, and G. Weisz, Learning with good feature representations in bandits and in RL with a generative model. In: *International Conference on Machine Learning*, PMLR, 5662–5670, 2020.
[37] G. Li, Y. Chen, Y. Chi, Y. Gu, and Y. Wei, Sample-efficient reinforcement learning is feasible for linearly realizable MDPs with limited revisiting, *arXiv:2105.08024*, 2021.
[38] Y. Liu and E. Brunskill, When simple exploration is sample efficient: Identifying sufficient conditions for random exploration to yield pac RL algorithms, *arXiv:1805.09045*, 2018.
[39] J. Long and J. Han, Perturbational complexity by distribution mismatch: A systematic analysis of reinforcement learning in reproducing kernel Hilbert space, *J. Mach. Learn.*, **1**:1–34, 2022.
[40] J. Long, J. Han, and W. E, An L^2 analysis of reinforcement learning in high dimensions with kernel and neural network approximation, *CSIAM Trans. Appl. Math.*, **3**(2):191–220, 2022.
[41] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, Playing Atari with deep reinforcement learning, *arXiv:1312.5602*, 2013.
[42] R. Munos, Error bounds for approximate value iteration. In: *Proceedings of the 20th National Conference on Artificial Intelligence*, AAAI Press, Vol. 2, 1006–1011, 2005.
[43] V. I. Paulsen and M. Raghupathi, *An Introduction to the Theory of Reproducing Kernel Hilbert Spaces*, Vol. 152, Cambridge University Press, 2016.
[44] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, 2014.
[45] F. Riesz, *Sur les opérations functionnelles linéaires*, Gauthier-Vllars, 1909.
[46] A. Rudi, L. Carratino, and L. Rosasco, FALKON: An optimal large scale kernel method, *Adv Neural Inf Process Syst*, 30, 2017.
[47] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, Trust region policy optimization. In: *International Conference on Machine Learning*, PMLR, 1889–1897, 2015.
[48] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, High-dimensional continuous control using generalized advantage estimation, *arXiv:1506.02438*, 2015.
[49] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms, *arXiv:1707.06347*, 2017.
[50] D. Silver et al., Mastering the game of Go with deep neural networks and tree search, *Nature*, **529**(7587): 484–489, 2016.
[51] I. Steinwart, D. R. Hush, and C. Scovel, Optimal rates for regularized least squares regression. In: *Conference on Learning Theory*, 79–93, 2009.
[52] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT press, 2018.
[53] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, Policy gradient methods for reinforcement learning with function approximation, *Adv Neural Inf Process Syst*, **12**, 1999.
[54] B. Wang, Y. Yan, and J. Fan, Sample-efficient reinforcement learning for linearly-parameterized mdps with a generative model, *Adv Neural Inf Process Syst*, **34**:23009–23022, 2021.
[55] L. Wang, Q. Cai, Z. Yang, and Z. Wang, Neural policy gradient methods: Global optimality and rates of convergence, *arXiv:1909.01150*, 2019.
[56] Y. Wang, R. Wang, S. S. Du, and A. Krishnamurthy, Optimism in reinforcement learning with generalized linear function approximation. In: *International Conference on Learning Representations*, 2021.
[57] L. Yang and M. Wang, Sample-optimal parametric Q-learning using linearly additive features. In: *International Conference on Machine Learning*, PMLR, 6995–7004, 2019.
[58] Z. Yang, C. Jin, Z. Wang, M. Wang, and M. Jordan, Provably efficient reinforcement learning with kernel and neural function approximations, *Adv Neural Inf Process Syst*, **33**, 2020.
[59] Z. Yang, C. Jin, Z. Wang, M. Wang, and M. I. Jordan, On function approximation in reinforcement learning: Optimism in the face of large state spaces, *arXiv:2011.04622*, 2020.

[60] A. Zanette, A. Lazaric, M. Kochenderfer, and E. Brunskill, Learning near optimal policies with low inherent Bellman error. In: *International Conference on Machine Learning*, PMLR, 10978–10989, 2020.
[61] D. Zhou, J. He, and Q. Gu, Provably efficient reinforcement learning for discounted MDPs with feature mapping. In: *International Conference on Machine Learning*, PMLR, 12793–12802, 2021.