Sample-Optimal Parametric Q-Learning Using Linearly Additive Features

Lin F. Yang [1] **Mengdi Wang** [1]

Abstract

Consider a Markov decision process (MDP) that admits a set of state-action features, which can linearly express the process's probabilistic transition model. We propose a parametric Q-learning algorithm that finds an approximate-optimal policy using a sample size proportional to the feature dimension K and invariant with respect to the size of the state space. To further improve its sample efficiency, we exploit the monotonicity property and intrinsic noise structure of the Bellman operator, provided the existence of anchor state-actions that imply implicit non-negativity in the feature space. We augment the algorithm using techniques of variance reduction, monotonicity preservation, and confidence bounds. It is proved to find a policy which is ϵ-optimal from any initial state with high probability using $\widetilde{O}(K/\epsilon^2(1-\gamma)^3)$ sample transitions for arbitrarily large-scale MDP with a discount factor $\gamma \in (0,1)$. A matching information-theoretical lower bound is proved, confirming the sample optimality of the proposed method with respect to all parameters (up to polylog factors).

1. Introduction

Markov decision problems (MDP) are known to suffer from the curse of dimensionality. A basic theoretical question is: Suppose that one can query sample transitions from any state of the system using any action, how many samples are needed for learning a good policy? In the tabular setting where the MDP has S states and A actions, the necessary and sufficient sample size for finding an approximate-optimal policy is $\widetilde{\Theta}(\frac{SA}{(1-\gamma)^3})$ [1] where $\gamma \in (0,1)$ is a discount factor (Azar et al., 2013; Sidford et al., 2018a). However, this theoretical-sharp result does not generalize to practical problems where S, A can be arbitrarily large or infinite.

Let us consider MDP with structural knowledges. Suppose that each state-action pair (s,a) admits a feature vector $\phi(s,a) \in \mathbb{R}^K$ that can express the transition dynamics conditioning on (s,a). In practice, the abstract state variable s can be a sequence of historical records or a raw-pixel image, containing much information that is not related to the decision process. More general settings of MDP with structural knowledges have been considered in Azizzadenesheli et al. (2016); Jiang et al. (2017) and references therein.

In this paper, we focus on an important and very basic class of structured MDP, where the features can represent transition distributions $P(\cdot \mid \cdot)$ through an unknown *linear additive model.* The feature-based linear transition model is related to the commonly used linear Q-function model. We show that they are essentially equivalent when there is zero Bellman error (a notion introduced in Munos & Szepesvári (2008)). A similar argument has been made in Parr et al. (2008). It also contains as a special case the soft state aggregation model (Singh et al., 1995; Duan et al., 2018). In this setting, we will study the theoretic sample complexity for learning a good policy by querying state-transition samples. We also aim to develop efficient policy learning algorithms with provable sample efficiency. We study the following two questions:

Q1: How many observations of state-action-state transitions are *necessary* for finding an ϵ-optimal policy?

Q2: How many samples are *sufficient* for finding an ϵ-optimal policy with high probability and *how* to find it?

To answer **Q1**, an information-theoretic lower bound is provided (Theorem 1), suggesting that, regardless of the learning algorithm, the necessary sample size for finding a good policy with high probability is $\widetilde{\Omega}\big(\frac{K}{(1-\gamma)^3\cdot\epsilon^2}\big)$ where K is the dimension of feature space.

To answer **Q2**, we develop Q-learning-like algorithms that take as input state-transition samples and output a parameterized policy. A basic parametric Q-learning algorithm performs approximate value-iteration estimates on a few points of the Q function, so that actual updates happen on the parameters. This idea originates from the phased Q-learning (Kearns & Singh, 1999) and the fitted value iteration (Munos

[*]Equal contribution [1]Department of Operations Research and Financial Engineering, Princeton University. Correspondence to: Lin Yang <lin.yang@princeton.edu>, Mengdi Wang <mengdiw@princeton.edu>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

[1] $\widetilde{f(\cdot)}$ ignores poly log $f(\cdot)$ factors.

& Szepesvári, 2008; Antos et al., 2008a;b). Our algorithm is simpler and does not require function fitting. Convergence and approximation error analysis is provided even when the MDP cannot be fully expressed using the features. Despite its simplicity, the basic algorithm has complexity $\widetilde{O}\big(\frac{K}{(1-\gamma)^7\cdot\epsilon^2}\big)$, which is not sample-optimal.

Furthermore, we develop an accelerated version of parametric Q-learning that involves taking mini-batches, computing confidence bounds, and using monotonicity-preserving and variance reduction techniques. It uses some ideas from fast solvers of tabular MDP (Sidford et al., 2018b;a). To fully exploit the monotonicity property of the Bellman operator in the algorithm, we need an additional "anchor" assumption, i.e., there exists a (small) set of state-actions that can represent the remaining ones using convex combinations. The "anchors" can be viewed as vertices of the state-action space, and implies an intrinsic nonnegativity in the feature space which is needed for monotonic policy improvement. We show that the algorithm takes just enough samples per update to keep the value/policy iterates within a sequence of narrow confidence regions that monotonically improve to the near-optimal solutions. It finds an ϵ-optimal policy (regardless of the initial state) with probability at least $1-\delta$ using

$$\widetilde{\Theta}\bigg(\frac{K}{(1-\gamma)^3\cdot\epsilon^2}\cdot\log\frac{1}{\delta}\bigg)$$

samples. It matches the information-theoretic lower bound up to $\log(\cdot)$ factors, thus the algorithm is nearly sample-optimal. If $\gamma = 0.99$, this algorithm is $(1-\gamma)^{-4} = 10^8$ times faster than the basic algorithm.

Our model, algorithms and analyses relate to previous literatures on the sample complexity of tabular MDP, reinforcement learning with function approximation, linear models and etc. A detailed account for the related literatures is given in Section 6. All technical proofs are given in the appendix. To our best knowledge, this work provides the first sample-optimal algorithm and sharp complexity analysis (up to polylog factors) for MDP with linear models.

2. Markov Decision Process, Features, Linear Models

In this section we introduce the basics of Markov decision process and the feature-based linear transition model.

2.1. Preliminaries

In a *discounted Markov decision process* (DMDP or MDP for short), there is a finite set of *states* $\mathcal{S}$, a finite set of *actions* $\mathcal{A}$. Let $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$. At any state $s \in \mathcal{S}$, an agent is allowed to play an action $a \in \mathcal{A}$. She receives an immediate reward $r(s,a) \in [0,1]$ after playing a at s, and then the process will transition to the next state $s' \in \mathcal{S}$ with probability $P(s'|s,a)$, where P is the collection of *transition distributions*. The full instance of MDP can be described by the tuple $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$. The agent would like to find a *policy* $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the long-term expected reward starting from every state s, i.e.,

$$v^\pi(s) := \mathbb{E}\bigg[\sum_{t=0}^{\infty} \gamma^t r(s^t, \pi(s^t))|s^0 = s\bigg]$$

where $\gamma \in (0,1)$ is a discount factor. We call $v^\pi \in \mathbb{R}^{\mathcal{S}}$ the *value function* of policy π. A policy π^* is said to be *optimal* if it attains the maximal possible value at every state. In fact, it is known (see e.g. Puterman (2014)) that there is a *unique* optimal value function v^* such that

$$\forall s \in \mathcal{S}: \quad v^*(s) = \max_\pi v^\pi(s) = v^{\pi^*}(s).$$

A policy π is said to be *ϵ-optimal* if it achieves near-optimal cumulative reward from *any* initial state, i.e.,

$$v^\pi(s) \geq v^*(s) - \epsilon, \qquad \forall\, s \in \mathcal{S},$$

or equivalently $\|v^\pi - v^*\|_\infty \leq \epsilon$ for short. We denote the Bellman operator $\mathcal{T} : \mathbb{R}^{\mathcal{S}} \to \mathbb{R}^{\mathcal{S}}$ as

$$\forall s \in \mathcal{S}: \quad [\mathcal{T}v](s) = \max_{a\in\mathcal{A}}[r(s,a) + \gamma P(\cdot|s,a)^\top v].$$

A vector v^* is the optimal value of the DMDP if and only if it satisfies the *Bellman equation* $v = \mathcal{T}v$.

The Q-function of a policy π is defined as $Q^\pi(s,a) = r(s,a) + \gamma\sum_{s'} P(s'|s,a)v^\pi(s')$, and the optimal Q-function is denoted by $Q^* = Q^{\pi^*}$. We overload the notation $\mathcal{T}$ to also denote the Bellman operator in the space of Q-functions, i.e., $\mathcal{T} : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ such that

$$\mathcal{T}Q(s,a) = r(s,a) + \gamma P(\cdot|s,a)^\top \max_{a'} Q(\cdot, a').$$

A vector $Q^* \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is the optimal Q-function if and only if it satisfies the Bellman equation $Q = \mathcal{T}Q$.

We use O,Ω, Θ to denote leading orders, and we use $\tilde{O}$,$\tilde{\Omega}$, $\tilde{\Theta}$ to omit polylog factors. We use $\lesssim$ to denote "approximately less than" by ignoring non-leading order terms, constant and polylog factors.

2.2. Feature-based Linear Transition Model

We study Markov decision processes with structural knowledges. Suppose that the learning agent is given a set of K feature functions $\phi_1, \phi_2, \ldots, \phi_K : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$. The feature ϕ maps the raw state and action (s,a) into the K-dimensional vector

$$\phi(s,a) = [\phi_1(s,a), \phi_2(s,a), \ldots, \phi_K(s,a)] \in \mathbb{R}^K.$$

Suppose the feature vector $\phi(s,a)$ is sufficient to express the future dynamics of the process conditioning on the current raw state and action. In particular, we focus on a basic linear model given below.

Definition 1 (Feature-based Linear Transition Model). *Consider a DMDP instance $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ and a feature map $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^K$. We say that M admits a linear feature representation ϕ if for every s, a, s',*

$$P(s'|s, a) = \sum_{k \in [K]} \phi_k(s, a)\psi_k(s').$$

for some functions $\psi_1, \ldots, \psi_K : \mathcal{S} \to \mathbb{R}$. We denote the set of all such MDP instances as $\mathcal{M}^{trans}(\mathcal{S}, \mathcal{A}, \gamma, \phi)$. We denote $\mathcal{M}_K^{trans}(\mathcal{S}, \mathcal{A}, \gamma)$ the set of all DMDP instances that admits a K-dimensional feature representation.

Remark 1 (Independence of rewards). The feature representations $\phi(s, a)$ in Definition 1 capture the transition dynamics of the Markov process under different actions. It is a form of structural knowledge about the environment. It has nothing to do with the rewards $r(s, a)$.

Remark 2 (Combining state features and action features). In many settings one may be given a state-only feature map ϕ_1 and an action-only feature map ϕ_2. In this case, one can construct the joint state-action feature by $\phi(s, a) = \phi_1(s)\phi_2(a)$. As long as the MDP admits a linear transition model in both ϕ_1, ϕ_2, it also admits a linear representation in the product feature $\phi = \phi_1 \times \phi_2$.

Remark 3 (Relation to soft state aggregation). The feature-based linear transition model (Definition 1) contains a worth-noting special case. When each $\phi(s, a) \in \mathbb{R}^K$ and $\psi_k \in \mathbb{R}^S$ is a probability density function, the linear transition model reduces to a soft state aggregation model (Singh et al., 1995; Duan et al., 2018). In the soft state aggregation model, each state can be represented by a mixture of latent meta-states, through aggregation and disaggregation distributions. There would be K meta-states, which can be viewed as the leading "modes" of the process. In contrast, our feature-based transition model is much more general. Our feature map ϕ can be anything as long as it is representative of the transition distributions. It captures information about not only the states but also the actions.

2.3. Relation to Linear Q-function Model

Linear models are commonly used for approximating value functions or Q-functions using given features (sometimes referred to as basis functions). The proposed linear transition model is closely related to the linear Q-function model, where Q^π's are assumed to admit a linear representation.

First it is easy to see that if the MDP admits a linear transition model using ϕ, the Q-functions admit a linear model.

Proposition 1. *Let $M \in \mathcal{M}^{trans}(\mathcal{S}, \mathcal{A}, \gamma, \phi)$. Then $Q^\pi \in$ Span(r, ϕ) for all π.*

Next we show that the two models are essentially "equivalent" in terms of expressibility. A similar conclusion was made by Parr et al. (2008), which showed that a particular solution obtained using the linear value model is equivalent to a solution obtained using a linear transition model. Recall a notion of Bellman error that was firstly introduced in (Munos & Szepesvári, 2008).

Definition 2 (Bellman Error). *Let $\mathcal{F} \subset \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$ be a class of Q functions. Given the Bellman operator $\mathcal{T}$, the Bellman error of $\mathcal{F}$ is $d(\mathcal{T}\mathcal{F}, \mathcal{F}) = \sup_{g \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - \mathcal{T}g\|$.*

We show that the linear transition model is equivalent to the linear Q-function model with zero Bellman error.

Proposition 2 (Equivalence to Zero Bellman Error). *Let $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ be an MDP instance with the Bellman operator $\mathcal{T}$. Let $\phi : \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$ be a feature map, and let $\mathcal{F} =$ Span(r, ϕ). If $r \in \mathcal{F}$, then*

$$d(\mathcal{T}\mathcal{F}, \mathcal{F}) = 0 \quad \textit{if \& only if} \quad M \in \mathcal{M}^{trans}(S, A, \gamma, \phi).$$

Suppose $Q^\pi \in \text{Span}(r, \phi)$ for all π's. However, value-iteration-based method would still fail if the Bellman operator $\mathcal{T}$ does not preserve the (r, ϕ) representation. In contrast, if the Q-functions admit linear representations using ϕ but the transition kernel P does not, the Bellman error can be arbitrarily large. The Bellman error may be large even after projection or function fitting - a common source of unstable and oscillating behaviors in approximate dynamic programming (Tsitsiklis & Van Roy, 1996; Munos & Szepesvári, 2008).

3. Information-Theoretic Sample Complexity

Let us study the feature-based MDP model (Definition 1). It comes with the structural knowledge that each state-action pair (s, a) can be represented by the feature vector $\phi(s, a) \in \mathbb{R}^K$. However, this model can *not* be parameterized by a small number of parameters. The full transition model with known feature map ϕ can not be specified unless all the unknown parameters $\psi_k(s')$, for $s' \in S, k \in [K]$ are given. Its model size is $S \times K$, which can be arbitrarily large for arbitrarily large S.

Given the state-action features, we aim to learn a near-optimal parametrized policy using a small number of samples, which hopefully depends on K but not S. Suppose that we are given a *generative model* (Kakade, 2003) where the agent is able to query transition samples and reward from any state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$. Such a generative model is commonly available in simulation systems. To this end, we ask how many samples are necessary to obtain an approximate-optimal policy? Our first theorem provides a firm answer.

Theorem 1 (Sample Complexity Lower Bound). *Let $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ be an instance of DMDP, and let $\mathcal{A}$ be any algorithm that queries sample transitions of M and outputs a policy. Let $\pi^{\mathcal{A},M,N}$ be the output of $\mathcal{A}$ using N samples.*

Then

$$\inf_{\mathcal{A}} \sup_{M\in\mathcal{M}_K^{trans}(\mathcal{S},\mathcal{A},\gamma)} \mathbb{P}\bigg(\sup_{s\in\mathcal{S}}(v^*(s) - v^{\pi^{\mathcal{A},M,N}}(s)) \geq \epsilon\bigg)$$
$$\geq 1/3, \quad \textit{if} \quad N = O\bigg(\frac{K}{(1-\gamma)^3\cdot\epsilon^2\cdot\log\epsilon^{-1}}\bigg),$$

provided $\epsilon \leq \epsilon_0$ for some $\epsilon_0 \geq 0$.

Theorem 1 suggests that, in order to solve the feature-based MDP to precision level ϵ with probability at least $2/3$, any algorithm needs at least $\widetilde{\Omega}\bigg(\frac{K}{(1-\gamma)^3\cdot\epsilon^2}\bigg)$ sample transitions in the worst case.

In the tabular setting without any feature, the sample complexity lower bound is known to be $\Omega(\frac{SA}{(1-\gamma)^3\epsilon^2})$ (Azar et al., 2013). Our lower bound can be proved by constructing a reduction from the feature-based model to a smaller-size tabular MDP with K state-action pairs. In the reduction, the features are constructed as indicator functions corresponding to a K-partition of the state space. We postpone the proof to the appendix.

4. A Basic Parametric Q-Learning Method

We develop a Q-learning algorithm for MDP admitting feature representations provided with a generative model.

4.1. Algorithm

Recall that phased Q-Learning (Kearns & Singh, 1999) takes the form $Q(s,a) \leftarrow r(s,a) + \frac{\gamma}{m}\sum_{i=1}^m \max_{a'} Q(s_i', a')$, where s_i''s are sample states generated from $P(\cdot \mid s,a)$. In the tabular setting, one needs to keep track of all the $Q(s,a)$ values.

Given the feature map $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^K$, we parameterize the Q-functions, value functions and policies using $w \in \mathbb{R}^K$ by

$$Q_w(s,a) := r(s,a) + \gamma\phi(s,a)^\top w, \tag{1}$$
$$V_w(s) := \max_{a\in\mathcal{A}} Q_w(s,a), \tag{2}$$
$$\pi_w(s) := \arg\max_{a\in\mathcal{A}} Q_w(s,a). \tag{3}$$

A scalable learning algorithm should keep track of only the parameters w, from which one can decode the high-dimensional value and policy functions according to (1-3).

Algorithm 1 gives a parametric phased Q-learning method. It queries state-action transitions and makes Q-learning-like updates on the parameter w. Each iteration picks a small set of state-action pairs $\mathcal{K}$, and performs approximate value iteration on $\mathcal{K}$. The set $\mathcal{K}$ can be picked almost arbitrarily. To obtain a convergence bound, we assume that the state-action pairs in $\mathcal{K}$ cannot be too alike, i.e., the regularity condition (4) holds for some value $L > 0$.

Assumption 1 (Representative States and Regularity of Features). *There exists a representative state-action set $\mathcal{K} \subset \mathcal{S}\times\mathcal{A}$ with $|\mathcal{K}| = K$ and a scalar $L > 0$ such that*

$$\|\phi(s,a)^T\Phi_{\mathcal{K}}^{-1}\|_1 \leq L, \qquad \forall\,(s,a) \tag{4}$$

where $\Phi_{\mathcal{K}} \in \mathbb{R}^{K\times K}$ is the collection of row feature vectors $\phi(s,a)$ where $(s,a)\in\mathcal{K}$ and $L \geq 1$.

Algorithm 1 Phased Parametric Q-Learning (PPQ-Learning)

```
1: Input: A DMDP $\mathcal{M} = (\mathcal{S},\mathcal{A},P,r,\gamma)$ with a generative model
2: Input: Integer $N > 0$
3:
4: Initialize: $R \leftarrow \Theta\big[\frac{\log N}{1-\gamma}\big]$, $w \leftarrow 0 \in \mathbb{R}^K$;
5: Repeat:
6: for $t = 1, 2, \ldots, R$ do
7:     Pick a representative set $\mathcal{K} \subset \mathcal{S}\times\mathcal{A}$ satisfying (4).
8:     $Q \leftarrow 0 \in \mathbb{R}^K$;
9:     for $(s,a) \in \mathcal{K}$ do
10:        Obtain $\frac{N}{KR}$ samples $\{s^{(j)}\}$ i.i.d. from $P(\cdot|s,a)$;
11:        $Q[(s,a)] \leftarrow \frac{KR}{N}\sum_{j=1}^{N/KR}\Pi_{[0,(1-\gamma)^{-1}]}[V_w(s^{(j)})]$;
12:                $\triangleright$ $\Pi_{[a,b]}$ projects a number onto $[a,b]$
13:    end for
14:    $w \leftarrow \Phi_{\mathcal{K}}^{-1}Q$;
15: end for
16: Output: $w \in \mathbb{R}^K$
```

4.2. Error Bound and Sample Complexity

We show that the basic parametric Q-learning method enjoys the following error bound.

Theorem 2 (Convergence of Algorithm 1)**.** *Suppose Assumption 1 holds. Suppose that the DMDP instance $M = (\mathcal{S},\mathcal{A},P,r,\gamma)$ has an approximate transition model $\widetilde{P}$ that admits a linear feature representation ϕ (Defn. 1), such that for some $\xi\in[0,1]$, $\|P(\cdot\mid s,a) - \widetilde{P}(\cdot\mid s,a)\|_{TV} \leq \xi$, $\forall\,(s,a)$. Let Algorithm 1 takes $N > 0$ samples and outputs a parameter $w\in\mathbb{R}^K$. Then, with probability at least $1-\delta$, $\|v^{\pi_w} - v^*\|_\infty$*

$$\leq L\cdot\Bigg(\sqrt{\frac{K}{N\cdot(1-\gamma)}} + \xi\Bigg)\cdot\frac{\text{poly}\log(NK\delta^{-1})}{(1-\gamma)^3}.$$

Remark 4 (Policy optimality guarantee). Our bound applies to v^{π_w}, i.e., the actual performance of the policy π_w in the real MDP. It is for the ℓ_∞ norm, i.e., the policy is ϵ-optimal from every initial state. This is the strongest form of optimality guarantee for solving MDP.

Remark 5 (Approximation error due to model misspecification). When the feature-based transition model is inexact up to ξ total variation, there is an approximation gap in the policy's performance $O\Big[L \cdot \xi \cdot \frac{\text{poly}\log(NK\delta^{-1})}{(1-\gamma)^3}\Big]$. It suggests that, even if the observed feature values $\phi(s,a)$ cannot fully express the state and action, the Q-learning method can still find approximate-optimal policies. The level of degradation depends on the total-variation divergence between the true transition distribution and its closest feature-based transition model.

Remark 6 (**Sample complexity of Algorithm 1).** When the MDP is fully realizable under the features, we have $\xi = 0$. Then the number of samples needed for achieving ϵ policy error is

$$\tilde{O}\bigg[\frac{KL^2}{(1-\gamma)^7\epsilon^2}\bigg].$$

It is independent of size of the original state space, but depends linearly on K. Its dependence on $\frac{1}{1-\gamma}$ matches the tabular phased Q-learning (Kearns & Singh, 1999) which has complexity $O(\frac{SA}{(1-\gamma)^7\epsilon^2})$ (Sidford et al., 2018a). Despite the fact that the MDP model has $S \times K$ unknown parameters, the basic parametric Q-learning method can produce good policies even with small data. However, there remains a gap between the current achievable sample complexity (Theorem 2) and the lower bound (Theorem 1).

5. Sample-Optimal Parametric Q-Learning

In this section we will accelerate the basic parametric Q-learning algorithm to maximize its sample efficiency. To do so, we need to modify the algorithm in nontrivial ways in order to take full advantage of the MDP's structure.

5.1. Anchor States and Monotonicity

In order to use samples more efficiently, we need to leverage monotonicity of the Bellman operator (i.e., $\mathcal{T}v_1 \le \mathcal{T}v_2$ if $v_1 \le v_2$). However, when the Q function is parameterized as a linear function in w, noisy updates on w may easily break the pointwise monotonicity in the Q space. To remedy this issue, we will impose an additional assumption to ensure that monotonicity can be preserved implicitly.

Assumption 2 (Anchor State-Action Pairs). *There exists a set of anchor state-action pairs $\mathcal{K}$ such that for any $(s,a) \in \mathcal{S} \times \mathcal{A}$, its feature vector can be represented as a convex combination of the anchors $\{(s_k, a_k) \mid k \in \mathcal{K}\}$:*

$$\exists\{\lambda_k\} : \phi(s,a) = \sum_{k\in\mathcal{K}} \lambda_k \phi(s_k,a_k), \quad \sum_{k\in\mathcal{K}} \lambda_k = 1, \lambda_k \ge 0.$$

The anchoring (s_k, a_k)'s can be viewed as "vertices" of the state-action space. They imply that the transition kernel P admits a nonnegative factorization, which can be seen by transforming ϕ linearly such that each anchor corresponds to a unit feature vector. This implicit non-negativity is a key to pointwisely monotonic policy/value updates.

The notion of "anchor" is a natural analog of the anchor word condition from topic modeling (Arora et al., 2012) and nonnegative matrix factorization (Donoho & Stodden, 2004). A similar notion of "anchor state" has been studied in the context of soft state aggregation models to uniquely identify latent meta-states (Duan et al., 2018). Under the anchor assumption, without loss of generality, we will assume that ϕ's are nonnegative, each $\phi(s,a)$ is a vector of probabilities, and there are K anchors with unit feature vectors.

5.2. Achieving The Optimal Sample Complexity

We develop a sample-optimal algorithm which is implemented in Algorithm 2. Let us explain the features that enable it to find more accurate policies. Some of the ideas are due to (Sidford et al., 2018b;a), where they were used to develop fast solvers for the tabular MDP.

Parametrization. For the purpose of preserving monotonicity, Algorithm 2 employs a new parametric form. It uses a collection of parameters $\theta = \{w^{(i)}\}_{i=1}^{Z}$ instead of a single vector, with $Z = \widetilde{O}(\frac{1}{1-\gamma})$. The parameterized policy and value functions take the form

$$V_\theta(s) := \max_{h\in[Z]} \max_{a\in\mathcal{A}} \big(r(s,a) + \gamma\phi(s,a)^\top \cdot w^{(h)}\big) \quad \text{and}$$

$$\pi_\theta(s) \in \arg\max_{a\in\mathcal{A}} \max_{h\in[Z]} \big(r(s,a) + \gamma\phi(s,a)^\top \cdot w^{(h)}\big). \quad (5)$$

Given θ, one can compute $V_\theta(s), \pi_\theta(s)$ by solving an one-step optimization problem. If a takes continuous values, it needs to solve a nonlinear optimization problem.

Computing confidence bounds. In Step 13 and Step 18, the algorithm computes confidence bounds $\epsilon^{(i,j)}$'s for the estimated values of PV_θ. These bounds tightly measure the distance from V_θ to the desired solution path, according to probaiblistic concentration arguments. With these bounds, we can precisely shift our estimator downwards so that certain properties would hold (e.g. monotonicity to be explained later) while not incurring additional error.

Monotonicity preservation. The algorithm guarantees that the following condition holds throughout:

$$V_\theta \le \mathcal{T}_{\pi_\theta} V_\theta, \qquad \text{pointwise.}$$

We call this property the *monotonicity* property, which together with monotonicity of the Bellman operator guarantees that (by an induction proof)

$$V_\theta \le \mathcal{T}_{\pi_\theta} V_\theta \le \mathcal{T}^2_{\pi_\theta} V_\theta \le \cdots \le \mathcal{T}^\infty_{\pi_\theta} V_\theta = v^{\pi_\theta} \le v^*, \quad \text{pointwise.}$$

Algorithm 2 uses two algorithmic tricks to preserve the monotonicity property throughout the iterations. First, the parametric forms of V_θ and π_θ (eq.(5)) take the maximum across all previous parameters (indexed by $h = (i,j)$). It guarantees that V_θ is monotonically improving throughout the outer and inner iterations. Second, the algorithm shifts all the estimated V_θ downwards by a term corresponding to its confidence bound (last equation of Line 13 and Line 18 of Algorithm 2). As a result, the estimated expectation is always smaller than the true expected value. By virtue of the nonnegativity (due to Assumption 2), the estimate, $\phi(s,a)^\top \overline{w}^{(i,j)}$, of the exact inner product $P(\cdot|s,a)^\top V^{(i,j-1)}$ for arbitrary (s,a) is also shifted downwards. Then we have

$$\phi(s,a)^\top \overline{w}^{(i,j)} \le P(\cdot|s,a)^\top V^{(i,j-1)} \le P(\cdot|s,a)^\top V^{(i,j)}.$$

By maximizing the lefthandside over a, we see that the monotonicity property is preserved inductively. See Lemma 8 for a more detailed proof.

Variance reduction. The algorithm uses an outer loop and an inner loop for approximately iterating the Bellman operator. Each outer iteration performs pre-estimation of a reference vector $PV_{\theta^{(i,0)}}$ (Step 13), which is used throughout the inner loop. For instance, let $\theta^{(i,j)}$ be the parameters at outer iteration i and inner iteration j. To obtain an entry $Q^{(i,j)}(s,a)$ of the new Q-function, we need to estimate $P(\cdot|s,a)^\top V_{\theta^{(i,j-1)}}$ with sufficient accuracy, so we have

$$\begin{aligned} P(\cdot|s,a)^\top V_{\theta^{(i,j-1)}} &= P(\cdot|s,a)^\top (V_{\theta^{(i,j-1)}} - V_{\theta^{(i,0)}}) \\ &\quad + P(\cdot|s,a)^\top V_{\theta^{(i,0)}}. \end{aligned}$$

Note that the reference $P(\cdot|s,a)^\top V_{\theta^{(i,0)}}$ is already approximated with high accuracy in Step 13. This allows the inner loop to successively refine the value and policy, while each inner iteration uses a smaller number of sample transitions to estimate the offset $P(\cdot|s,a)^\top (V_{\theta^{(i,j-1)}} - V_{\theta^{(i,0)}})$ (Step 18).

Putting together the preceding techniques, Algorithm 2 performs carefully controlled Bellman updates so that the estimated value-policy functions monotonically improve to the optimal ones. The algorithm contains $R' = \Theta(\log[\epsilon^{-1}(1-\gamma)^{-1}])$ many outer loops. Each outer loop (indexed by i) starts with a policy $\|v^* - V_{\theta^{(i,0)}}\|_\infty \lesssim H/2^i$ and ends with a policy $\|v^* - V_{\theta^{(i+1,0)}}\|_\infty \lesssim H/2^{i+1}$.The algorithm takes multiple rounds of mini-batches, where the sample size of each mini-batch is picked just enough to guarantee the accumulation of total error is within ϵ. The algorithm fully exploits the monotonicity property of the Bellman operator as well as the error accumulation in the Markov process (to be explained later in the proof outline).

5.3. Optimal Sample Complexity Guarantee

In this section, we analyze the sample complexity of the algorithm provided in the last section.

Theorem 3 (Near-Optimal Sample Complexity). *Suppose $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ is an MDP instance admitting the feature representation $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^K$. Suppose that Assumption 2 holds. Let $\delta, \epsilon \in (0,1)$ be parameters. Then Algorithm 2 takes*

$$N = \Theta\left[\frac{K}{(1-\gamma)^3 \cdot \epsilon^2} \cdot \log^{4/3}\frac{K}{\epsilon\delta(1-\gamma)} \cdot \log^2 \frac{1}{\epsilon(1-\gamma)}\right]$$

samples and outputs θ such that π_θ is ϵ-optimal from every initial state with probability at least $1-\delta$.

Theorem 3 is proved through a series of lemmas, which we defer to the appendix. Here we sketch the key ideas.

Proof Sketch. Let $H = \frac{1}{1-\gamma}$ for short. Each outer-loop iteration decreases the policy error upper bound by at least half. Suppose $\theta^{(i,0)}$ is the parameter when the ith outer iteration begins, we expect $\|V_{\theta^{(i,0)}} - v^*\|_\infty \le H/2^i$, with high probability. Therefore, after $R' = \log(H/\epsilon)$ iterations, we expect $\|V_{\theta^{(R',0)}} - v^*\|_\infty \le H/2^{R'} = O(\epsilon)$.

Now we analyze how many samples are sufficient within one outer-loop iteration. We show that the final error is mainly due to $\epsilon^{(i,0)}$, which comes from estimating the reference function $V_{\theta^{(i,0)}}$ (Line 13). This error is exemplified in the inner loop since $V_{\theta^{(i,0)}}$ is used repeatedly (line 18).

A key step of the proof is to show that the error contributed by $\epsilon^{(i,0)}$ throughout the inner-loop iterations is small. By using the monotonicity property, we can show that

$$\epsilon^{(i,0)}(s,a) \lesssim \sqrt{\sigma_{v^*}(s,a)/m}, \quad \forall\,(s,a),$$

where $\lesssim$ denotes "approximately less than" (ignoring non-leading terms), and $\sigma_{v^*} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is an intrinsic variance function of the MDP:

$$\sigma_{v^*}(s,a) := \mathrm{Var}_{s' \sim P(\cdot|s,a)}\left[v^*(s')\right].$$

By using the monotonicity property, we prove by induction:

$$\begin{aligned} v^* - v^{\pi_{\theta^{(i,R)}}} &\le v^* - V_{\theta^{(i,R)}} \lesssim \gamma P^{\pi^*}(v^* - V_{\theta^{(i,R-1)}}) + \epsilon_{\pi^*}^{(i,0)} \\ &\lesssim \ldots \le \gamma^R (v^* - V_{\theta^{(i,0)}}) + \sum_{i=0}^{R} \gamma^i (P^{\pi^*})^i \epsilon_{\pi^*}^{(i,0)} \\ &\lesssim (I - \gamma P^{\pi^*})^{-1} \epsilon_{\pi^*}^{(i,0)} \\ &\lesssim (I - \gamma P^{\pi^*})^{-1} \sqrt{\sigma_{v^*}^{\pi^*}}/\sqrt{m}, \qquad \text{pointwise w.h.p.,} \end{aligned}$$

where $\sigma_{v^*}^{\pi^*}(s) = \sigma_{v^*}(s, \pi^*(s))$, $\epsilon_{\pi^*}^{(i,0)}(s) = \epsilon^{(i,0)}(s, \pi^*(s))$, and m is the mini-batch size. Now we have found a connection between the error accumulation of the algorithm and the intrinsic variance of the MDP. By a form of conditional law of total variance of the Markov process (Lemma 7) and using the convex combination property (Assumption 2), one has

$$(I - \gamma P^{\pi^*})^{-1} \sqrt{\sigma_{v^*}^{\pi^*}} = \widetilde{O}\big(\sqrt{H^3}\big) \cdot \mathbf{1}.$$

Algorithm 2 Optimal Phased Parametric Q-Learning (OPPQ-Learning)

```
1:  **Input:** A DMDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ with anchor state-action pairs $\mathcal{K}$; feature map $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$;
2:  **Input:** $\epsilon, \delta \in (0, 1)$
3:  **Output:** $\theta \subset \mathbb{R}^K$ with $|\theta| = \Theta[(1-\gamma)^{-1} \log^2 \epsilon^{-1}]$
4:
5:  **Initialize:** $R' \leftarrow \Theta(\log[\epsilon^{-1}(1-\gamma)^{-1}])$, $R \leftarrow \Theta[R'(1-\gamma)^{-1}]$   ▷ initialize the numbers of iterations
6:              $\{w^{(i,j)}, \epsilon^{(i,j)}, \overline{w}^{(i,j)}\}_{i\in[0,R'],j\in[0,R]} \subset \mathbb{R}^K$ as 0 vectors   ▷ initialize parameters
7:              $m \leftarrow C \cdot \frac{1}{\epsilon^2} \cdot \frac{\log(R'RK\delta^{-1})^{4/3}}{(1-\gamma)^3}$,   ▷ mini-batch size for outer loop
                $m_1 \leftarrow C \cdot \frac{\log(R'RK\delta^{-1})}{(1-\gamma)^2}$ for some constant $C$;   ▷ mini-batch size for inner loop
8:              $\theta^{(0,0)} \leftarrow \{0\} \subset \mathbb{R}^K$   ▷ initialize the output to contain a single 0-vector
9:  **Iterates:**
10: ▷ Outer loop
11: **for** $i = 0, 1, \ldots, R'$ **do**
12:   **for** each $k \in [K]$ **do**
13:     Obtain state samples $x_k^{(1)}, x_k^{(2)}, \ldots, x_k^{(m)} \in \mathcal{S}$ from $P(\cdot|s_k, a_k)$ for $(s_k, a_k) \in \mathcal{K}$. Let
            $w^{(i,0)}(k) \leftarrow \frac{1}{m}\sum_{\ell=1}^{m} V_{\theta^{(i,0)}}(x_k^{(\ell)})$,   $z^{(i,0)}(k) \leftarrow \frac{1}{m}\sum_{\ell=1}^{m} V^2_{\theta^{(i,0)}}(x_k^{(\ell)})$
                ▷ empirical esitimate of $P_{\mathcal{K}} V_{\theta^{(i,0)}}$ and $P_{\mathcal{K}} V^2_{\theta^{(i,0)}}$
            $\sigma^{(i,0)}(k) \leftarrow z^{(i,0)}(k) - (w^{(i,0)}(k))^2$
                ▷ empirical esitimate of variance $P_{\mathcal{K}} V^2_{\theta^{(i,0)}} - (P_{\mathcal{K}} V_{\theta^{(i,0)}})^2$
            $\epsilon^{(i,0)}(k) \leftarrow \Theta\Big[\sqrt{\log[R'RK\delta^{-1}] \cdot \sigma^{(i,0)}(k) \cdot m^{-1}} + \log[R'RK\delta^{-1}](1-\gamma)^{-1}/m^{3/4}\Big]$
                ▷ estimate of the confidence bound of the emprical estimator $w^{(i,0)}$
            $\overline{w}^{(i,0)}(k) \leftarrow \max\{0, \quad \min\{w^{(i,0)}(k) - \epsilon^{(i,0)}(k), \quad (1-\gamma)^{-1}\}\}$   ▷ shift and clip the estimate
14:   **end for**
15:   ▷ Inner loop
16:   **for** $j = 1, 2, \ldots, R$ **do**
17:     **for** each $k \in [K]$ **do**
18:       Obtain state samples $x_k^{(1)}, x_k^{(2)}, \ldots, x_k^{(m_1)} \in \mathcal{S}$ from $P'(\cdot|s_k, a_k)$ for $(s_k, a_k) \in \mathcal{K}$. Let
              $w^{(i,j)}(k) \leftarrow \frac{1}{m_1}\sum_{\ell=1}^{m}\Big(V_{\theta^{(i,j-1)}}(x_k^{(\ell)}) - V_{\theta^{(i,0)}}(x_k^{(\ell)})\Big) + w^{(i,0)}(k)$   ▷ empirical esitimate of $P_{\mathcal{K}} V_{\theta^{(i,j-1)}}$
              $\epsilon^{(i,j)}(k) \leftarrow \epsilon^{(i,0)}(k) + \Theta(1-\gamma)^{-1} 2^{-i}\sqrt{\log(RR'K\delta^{-1})/m_1}$
                  ▷ approximate the confidence bound of $P_{\mathcal{K}} V_{\theta^{(i,j-1)}}$
              $\overline{w}^{(i,j)}(k) \leftarrow \max\Big\{0, \quad \min\{w^{(i,j)}(k) - \epsilon^{(i,j)}(k), \quad (1-\gamma)^{-1}\}\Big\}$   ▷ shift and clip the estimate
19:     **end for**
20:     $\theta^{(i,j)} \leftarrow \theta^{(i,j-1)} \cup \{\overline{w}^{(i,j)}\}$   ▷ attach the newly estimated parameter to $\theta$
21:   **end for**
22:   $\theta^{(i+1,0)} \leftarrow \theta^{(i,R)}$   ▷ prepare the next outer loop
23: **end for**
24: **Return** $\theta^{(R',R)}$
```

Therefore the inner loop accumulates error $\widetilde{O}(\sqrt{H^3/m})$, so $m = O(H^3) = O((1-\gamma)^{-3})$ number of samples is enough.

Finally, we prove by induction that all the desired events happen with sufficiently high probability, so that the iterates improve monotonically to the optimal solution within a sequence of carefully controlled error bars. The total number of outer iterations is nearly constant, therefore the total sample size needed scales with $O((1-\gamma)^{-3})$. □

Remark 7 (Sample Optimality of Algorithm 2). Theorem 3 matches the information-theoretic lower bound of Theorem 1 up to polylog factors with respect to all parameters $S, A, K, \epsilon, 1-\gamma$ (note that Theorem 1 still holds under the anchor restriction). Therefore it is a sample-optimal method for solving the feature-based MDP. No other method can outperform it by more than polylog factors.

Remark 8 (About Anchor State-Actions). The proof of Theorem 3 relies on the anchor assumption. The monotonicity property can be preserved because the anchor state-action pairs imply an implicit non-negative factorization of the transition kernel. The convex combination property of anchor state-actions is used in analyzing the error accumulation, needed by the conditional law of total variance. Anchor condition is commonly believed to be a key to identifying nonnegative models; see for example (Donoho & Stodden, 2004). We believe this is the first observation that it also relates to sample-optimal reinforcement learning.

Note that it is possible that the number of anchors is greater than the number of features K, then one can append new (dependent) features to make them equal. In this sense Assumption 2 *always* holds and the actual sample complexity depends on the number of anchors (instead of features). In addition, the anchors can be pre-computed as long as the ϕ feature map is known.

Remark 9 (Significance of $(1-\gamma)^{-4}$ Improvement). Let us compare the sample complexities of Algorithms 1, 2. They differ by a multiplicative gap $(1-\gamma)^{-4}$. Recall that $\gamma \in (0,1)$ is the discount factor. One can view $(1-\gamma)^{-1} = 1+\gamma+\gamma^2+\cdots$ as an approximate horizon. If $\gamma = 0.99$, the MDP essentially has 100 time steps, and

$$(1-\gamma)^{-4} = 10^8,$$

i.e., Algorithm 2 is 10^8 times faster. It only needs a tiny portion $(1/10^8)$ of the samples as needed by the basic algorithm. We see that clever algorithmic usage of monotonicity and variance structures of the MDP saves big.

6. Related Literatures

There is a body of works studying the sample complexity of tabular DMDP (i.e., the finite-state finite-action case without structural knowledge). Sample-based algorithms for learning value and policy functions have been studied in Kearns & Singh (1999); Kakade (2003); Singh & Yee (1994); Azar et al. (2011b; 2013); Sidford et al. (2018b;a) and many others. Among these papers, Azar et al. (2013) obtains the first tight sample bound for finding an ϵ-optimal value function, Sidford et al. (2018a) obtains the first tight sample bound for finding an ϵ-optimal policy; both complexities are of the form $\widetilde{O}[|\mathcal{S}||\mathcal{A}|(1-\gamma)^{-3}]$. Lower bounds have been shown in Azar et al. (2011a); Even-Dar et al. (2006) and Azar et al. (2013). Azar et al. (2013) gives the first tight lower bound $\Omega[|\mathcal{S}||\mathcal{A}|(1-\gamma)^{-3}]$.

Our result is relevant to the large body of works using linear models and basis functions to approximate value and Q functions. For instance, Tsitsiklis & Van Roy (1997); Nedić & Bertsekas (2003); Lagoudakis & Parr (2003); Melo et al. (2008); Parr et al. (2008); Sutton et al. (2009); Lazaric et al. (2012); Tagorti & Scherrer (2015) and Maei et al. (2010) studies both policy evaluation and optimization by assuming values are from a linear space. Tsitsiklis & Van Roy (1997) studied the convergence of the temporal-difference learning algorithm for approximating the value function for a fixed policy. Nedić & Bertsekas (2003) studies the policy evaluation problem using least square. Parr et al. (2008) studies the relationships of using linear functions to represent values and to represent transition models. Melo et al. (2008) studies the almost sure convergence of Q-learning-like methods using linear function approximation. Sutton et al. (2009) shows off-policy temporal-difference learning is convergent with linear function approximation. These earlier works primarily focused on convergence using linear function approximation, without analyzing the sample complexity.

Fitted value iteration (VI) applies to more general function approximators of the value function (Munos & Szepesvári, 2008; Antos et al., 2008a; Farahmand et al., 2010; Antos et al., 2008b), where v is approximated within a low-dimensional function space $\mathcal{F}$. They have shown that the error of the fitted-VI is affected by the Bellman error of the space $\mathcal{F}$. Their result applies to a general set of functional spaces, where the statistical error depends on a polynomial of $1/\epsilon, 1/(1-\gamma)$ and the intrinsic dimension of the functional space. It appears that their result works for the ℓ_p norm of the policy error, which is proportional to $\epsilon^{-\Theta(p)}$ with high probability. Their result does not apply to the ℓ_∞ policy error which is the focus of the current paper.

More recently, Lazaric et al. (2012); Tagorti & Scherrer (2015) analyzes the sample complexity of temporal difference least square for evaluating a fixed policy. Recently, a work by Jiang et al. (2017) studies the case when a form of Bellman error' is decomposable and has a small rank. They show that the number of trajectories needed depends on the Bellman rank rather than the number of states. Chen et al. (2018) proposes a primal-dual method for policy learning that uses linear models and state-action features for both the value and state-action distribution. To our best knowledge, there is no existing result that solves the linear-model MDP with provable-optimal sample complexity.

7. Remarks

The paper studies the information-theoretic sample complexity for solving MDP with feature-based linear transition model. It provides the first sharp sample complexity upper and lower bounds for learning the policy using a generative model. It also provides a sample-optimal parametric Q-learning method that involves computing confidence bounds, variance reduction and monotonic improvement. We hope that establishing sharp results for the basic linear model would shed lights on more general structured models and motivate faster solutions.

Acknowledgment

We thank Zhuoran Yang for pointing out a flaw in the initial proof of Proposition 2. We thank the anonymous reviewers for the helpful comments.

References

Antos, A., Szepesvári, C., and Munos, R. Fitted Q-iteration in continuous action-space mdps. In *Advances in neural information processing systems*, pp. 9–16, 2008a.

Antos, A., Szepesvári, C., and Munos, R. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008b.

Arora, S., Ge, R., and Moitra, A. Learning topic models–going beyond svd. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pp. 1–10. IEEE, 2012.

Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. Reinforcement learning with a near optimal rate of convergence. 2011a.

Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. Speedy q-learning. In *Advances in neural information processing systems*, 2011b.

Azar, M. G., Munos, R., and Kappen, H. J. Minimax pac bounds on the sample complexity of reinforcement learning with a generative model. *Machine learning*, 91(3): 325–349, 2013.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. Reinforcement learning in rich-observation mdps using spectral methods. *arXiv preprint arXiv:1611.03907*, 2016.

Bertsekas, D. P. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 2005.

Chen, Y., Li, L., and Wang, M. Scalable bilinear pi learning using state and action features. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 834–843, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Donoho, D. and Stodden, V. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in neural information processing systems*, pp. 1141–1148, 2004.

Duan, Y., Ke, Z. T., and Wang, M. State aggregation learning from markov transition data. *arXiv preprint arXiv:1811.02619*, 2018.

Even-Dar, E., Mannor, S., and Mansour, Y. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research*, 7(Jun):1079–1105, 2006.

Farahmand, A.-m., Szepesvári, C., and Munos, R. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems*, pp. 568–576, 2010.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. Contextual decision processes with low bellman rank are pac-learnable. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1704–1713. JMLR. org, 2017.

Kakade, S. M. *On the sample complexity of reinforcement learning*. PhD thesis, University of London London, England, 2003.

Kearns, M. J. and Singh, S. P. Finite-sample convergence rates for q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems*, pp. 996–1002, 1999.

Lagoudakis, M. G. and Parr, R. Least-squares policy iteration. *Journal of machine learning research*, 4(Dec): 1107–1149, 2003.

Lazaric, A., Ghavamzadeh, M., and Munos, R. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13(Oct):3041–3074, 2012.

Maei, H. R., Szepesvári, C., Bhatnagar, S., and Sutton, R. S. Toward off-policy learning control with function approximation. In *ICML*, pp. 719–726, 2010.

Melo, F. S., Meyn, S. P., and Ribeiro, M. I. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on Machine learning*, pp. 664–671. ACM, 2008.

Munos, R. and Szepesvári, C. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9 (May):815–857, 2008.

Nedić, A. and Bertsekas, D. P. Least squares policy evaluation algorithms with linear function approximation. *Discrete Event Dynamic Systems*, 13(1-2):79–110, 2003.

Parr, R., Li, L., Taylor, G., Painter-Wakefield, C., and Littman, M. L. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 752–759. ACM, 2008.

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

Sidford, A., Wang, M., Wu, X., Yang, L., and Ye, Y. Near-optimal time and sample complexities for solving markov decision processes with a generative model. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 5192–5202, 2018a.

Sidford, A., Wang, M., Wu, X., and Ye, Y. Variance reduced value iteration and faster algorithms for solving markov decision processes. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 770–787. Society for Industrial and Applied Mathematics, 2018b.

Singh, S. P. and Yee, R. C. An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16(3):227–233, 1994.

Singh, S. P., Jaakkola, T., and Jordan, M. I. Reinforcement learning with soft state aggregation. In *Advances in neural information processing systems*, pp. 361–368, 1995.

Sutton, R. S., Maei, H. R., and Szepesvári, C. A convergent $o(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in neural information processing systems*, pp. 1609–1616, 2009.

Tagorti, M. and Scherrer, B. On the rate of convergence and error bounds for LSTD(λ). In *International Conference on Machine Learning*, pp. 1521–1529, 2015.

Tsitsiklis, J. N. and Van Roy, B. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1-3):59–94, 1996.

Tsitsiklis, J. N. and Van Roy, B. Analysis of temporal-diffference learning with function approximation. In *Advances in neural information processing systems*, pp. 1075–1081, 1997.