Scalable Bilinear π Learning Using State and Action Features

Yichen Chen [1] **Lihong Li** [2] **Mengdi Wang** [3]

Abstract

Approximate linear programming (ALP) represents one of the major algorithmic families to solve large-scale Markov decision processes (MDP). In this work, we study a primal-dual formulation of the ALP, and develop a scalable, model-free algorithm called *bilinear π learning* for reinforcement learning when a sampling oracle is provided. This algorithm enjoys a number of advantages. First, it adopts linear and bilinear models to represent the high-dimensional value function and state-action distributions, respectively, using given state and action features. Its run-time complexity depends on the number of features, not the size of the underlying MDPs. Second, it operates in a fully online fashion without having to store any sample, thus having minimal memory footprint. Third, we prove that it is sample-efficient, solving for the optimal policy to high precision with a sample complexity linear in the dimension of the parameter space.

1. Introduction

Reinforcement learning lies at the intersection between control, machine learning, and stochastic processes (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998). The objective is to learn an optimal policy of a controlled system from interaction data. The most studied model for a controlled stochastic system is the Markov decision process (MDP), i.e., a controlled random walk over a state space $\mathcal{S}$, where in each state $s \in \mathcal{S}$ one can choose an action a from an action space $\mathcal{A}$ so that the random walk transitions to another state $s' \in \mathcal{S}$ with probability $P_a(s, s')$. In this paper, we do not assume the MDP model is explicitly known, but consider the setting where a *generative model* is given (see, e.g., Azar et al. (2013)). In other words, there is an oracle that takes (s, a) as input and outputs a random s' with probability $P_a(s, s')$ and an immediate reward $r_a(s) \in [0, 1]$. This is also known as the simulator-defined MDP in some literatures (Dietterich et al., 2013; Taleghan et al., 2015). Our goal is to find an optimal policy which, when running on the MDP to generate an infinitely long trajectory, yields the highest average per-step reward in the limit.

Here, we focus on problems where the state and action spaces $\mathcal{S}$ and $\mathcal{A}$ are too large to be enumerated. In practice, it might be computationally challenging to even store a single state of the process (e.g., states could correspond to high-resolution images). Suppose that we are given a collection of state features $\phi : \mathcal{S} \mapsto \mathbb{R}^D$ and action features $\psi : \mathcal{A} \mapsto \mathbb{R}^U$. They map each state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$ to column vectors $\phi(s) = (\phi_1(s), \ldots, \phi_D(s))^T$ and $\psi(a) = (\psi_1(a), \ldots, \psi_U(a))^T$, respectively, where $D \ll S := |\mathcal{S}|$ and $U \ll A := |\mathcal{A}|$.

Our primary interest is to develop a sample-efficient and computationally scalable algorithm, which takes advantage of the given features to solve an MDP with very large state and action spaces. Given the feature maps, ϕ and ψ, we adopt linear and bilinear models for approximating both the value function and the stationary state-action distribution of the MDP. By doing so, we can represent the value functions and state-action distributions, which are high-dimensional quantities, using a much smaller number of parameters.

Contributions. Our main contribution is a tractable, model-free primal-dual π-learning algorithm for such compact parametric representations. It incrementally updates parameters as new transitions are observed. With given state and action features, its per-iteration complexity is low, depending on U and D, not on $|\mathcal{S}|$ or $|\mathcal{A}|$:

- The new algorithm is inspired by a saddle point formulation of policy optimization in MDPs, which we refer to as the *Bellman saddle point problem*. We show a strong relation between the parametric saddle point problem and the original Bellman equation. In particular, the difference between solutions to these two problems can be quantified using the ℓ_∞- and ℓ_1-errors of the parametric function classes that are used to approximate the optimal value function and state-action

[1]Department of Computer Science, Princeton University, Princeton, NJ, USA [2]Google Inc., Kirkland, WA, USA [3]Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ, USA. Correspondence to: Mengdi Wang <mengdiw@princeton.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

distribution, respectively. In the special case where the approximation error is zero (also known as the "realizable" scenario), solving the Bellman saddle point problem is equivalent to solving the original Bellman equation.

- Each iteration of the algorithm can be viewed as a stochastic primal-dual iteration for solving the Bellman saddle point problem, where the value and policy updates are coupled in light of strong duality. We study the sample complexity of the π learning algorithm by analyzing the coupled primal-dual convergence process. We show that finding an ϵ-optimal policy (comparing to the best approximate policy) requires a sample size that is linear in $\frac{DU}{\epsilon^2}$, ignoring logarithmic terms. The sample complexity depends only on the numbers of state and action features. It is invariant with respect to the actual sizes of the state and action spaces.

Notations. The following notations are used throughout the paper. For any integer n, we use $[n]$ to denote the set of integers $\{1, 2, \ldots, n\}$. For a matrix Φ of size $m \times n$ and $1 \le p \le \infty$, the matrix p-norm is defined as $\|\Phi\|_p = \max\{\|\Phi v\|_p : v \in \mathbb{R}^n \text{ with } \|v\|_p = 1\}$. The (a, b)-norm of Φ, denoted by $\|\Phi\|_{a,b}$, is defined as the ℓ_b-norm of the column vector that consists of ℓ_a-norm of the rows of Φ. We use Φ_{i*} and Φ_{*j} to denote the i-th row and the j-th column of Φ, respectively. For a vector v, we denote by $\mathrm{diag}(v)$ the diagonal matrix with v on its diagonal. We denote by $\mathbf{1}$ the all-one column vector. For two probability distributions, u and w, over a finite set X, we denote by $D_{KL}(u\|w)$ their Kullback-Leibler divergence, i.e., $D_{KL}(u\|w) = \sum_{x\in X} u(x) \log \frac{u(x)}{w(x)}$. For two functions $f(x)$ and $g(x)$, we say that $f(x) = \mathcal{O}(g(x))$ if there exists a constant C such that $|f(x)| \le Cg(x)$ for all x.

2. Preliminaries

We review the basics of infinite-horizon MDP with the average-reward criterion.

2.1. Infinite-Horizon Average-Reward MDP

Most of the paper focuses on the infinite-horizon average-reward Markov Decisions Problem (MDP), in which one aims to make an infinite sequence of decisions and optimize the average per-time-step reward. An instance of the MDP can be described by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r})$, where $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, $\mathcal{P}$ is the collection of state-to-state transition probabilities $\mathcal{P} = \{P_a(s, s') \mid s, s' \in \mathcal{S}, a \in \mathcal{A}\}$, $\mathbf{r}$ is the collection of immediate reward functions $\mathbf{r} = \{r_a(s) \mid s \in \mathcal{S}, a \in \mathcal{A}\}$ with $r_a(s) \in [0, 1]$. In every step of the decision process, the system is in some state s, and an action a is chosen by a control policy. The system then transitions to a next-state s' with probability $P_a(s, s')$ with an immediate reward $r_a(s)$.

2.2. Policy Optimization

A randomized stationary policy can be represented by a matrix $\pi \in \Pi \subset \mathbb{R}^{S\times A}$, where Π consists of non-negative matrices whose s-th row, denoted by π_{s*}, corresponds to a probability distribution over $\mathcal{A}$ for state s. The policy optimization problem is to maximize the infinite-horizon average reward over stationary policies:

$$\max_{\pi\in\Pi} \left\{ \bar{v}^\pi = \lim_{T\to\infty} \mathbf{E}^\pi \left[\frac{1}{T} \sum_{t=1}^{T} r_{a_t}(s_t) \right] \right\}, \qquad (1)$$

where $(s_1, a_1, s_2, a_2, \ldots, s_t, a_t, \ldots)$ are state-action transitions generated by the Markov decision process under π from an arbitrary initial distribution, and the expectation $\mathbf{E}^\pi[\cdot]$ is taken over the entire trajectory. The limit in (1) exists and is invariant with respect to the initial state if the Markov decision process is irreducible under any π.

We denote by P^π the transition probability matrix of the MDP under a fixed policy π, i.e.,

$$P^\pi(s, s') = \sum_{a\in\mathcal{A}} \pi_{s,a} P_a(s, s')$$

for all $s, s' \in \mathcal{S}$. Note that the policy optimization problem (1) is equivalent to the following optimization problem (Puterman, 2014):

$$\max_{\pi\in\Pi, \xi\in\mathbb{R}^{|\mathcal{S}|}} \sum_{s\in\mathcal{S}, a\in\mathcal{A}} \xi_s \pi_{s,a} r_a(s)$$
$$\text{s.t. } \xi^\top P^\pi = \xi^\top, \ \xi \ge 0, \ \xi^\top \mathbf{1} = 1,$$

where the constraint ensures that ξ is the stationary distribution of states under P^π, and the objective is an explicit expression of the average reward $\bar{v}^\pi$. By defining $\mu_{s,a} = \xi_s \pi_{s,a}$, the policy optimization becomes a linear program

$$\max_{\mu\in\mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}} \sum_{s\in\mathcal{S}, a\in\mathcal{A}} \mu_{s,a} r_a(s)$$
$$\text{s.t.} \quad \sum_{a\in\mathcal{A}} \mu_{*a}^\top P_a = \sum_{a\in\mathcal{A}} \mu_{*a}^\top, \ \mu \ge 0, \ \|\mu\|_{1,1} = 1, \qquad (2)$$

where the constraint ensures that μ is a *stationary state-action distribution* of the MDP under some policy. We denote by μ^* the optimal solution to (2). The optimal policy can be obtained by $\pi^*_{s,a} = \frac{\mu^*_{s,a}}{\|\mu^*_{s*}\|_1}$. It is known that there exists an optimal basic solution, which corresponds to a deterministic policy (Puterman, 2014).

2.3. Bellman Equation

It is known from the theory of dynamic programming (Puterman, 2014; Bertsekas, 1995) that $\bar{v}^*$ is the optimal average

reward to the MDP $\mathcal{M}$ if and only if it satisfies the following *Bellman equation*:

$$\bar{v}^* + v^*(s) = \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S}} P_a(s, s') v^*(s') + r_a(s) \right\}, \forall\, s \in \mathcal{S}, \tag{3}$$

for some vector $v^* \in \mathbb{R}^{|\mathcal{S}|}$. Here, v^* is known as the difference-of-value vector, which we also refer to as the *value vector* or *value function* for short. Note that there exist infinitely many solutions of v^* to (3) (by adding any constant shift), which does not affect our analysis. The results of the paper hold for any such v^* unless stated otherwise. A stationary policy π^* is an optimal policy of the MDP if it attains the elementwise maximization in the Bellman equation (Puterman, 2014, Theorem 8.4.5). The Bellman equation also admits an equivalent linear program:

$$\begin{aligned} &\min_{\bar{v} \in \mathbb{R}, v \in \mathbb{R}^{|\mathcal{S}|}} \ \bar{v} \\ &\text{subject to } \ \bar{v} \cdot \mathbf{1} + (I - P_a)\, v - r_a \geq 0, \qquad \forall\, a \in \mathcal{A}. \end{aligned} \tag{4}$$

It is well known that the linear program (4) is the dual of the policy optimization problem (2) (Puterman, 2014). Informally speaking, the Bellman equation and the policy optimization problem are dual to each other.

3. Bilinear Model Reduction of MDP using State and Action Features

In this section, we study how to utilize given state and action features for dimension reduction of the Bellman saddle point problem. We propose to use a linear model for approximating the optimal value vector and a *bilinear* additive model for approximating the state-action distribution. Then we formulate a reduced-order primal-dual policy optimization problem, which we refer to as the parametric Bellman saddle point problem.

3.1. Using State and Action Features As Bilinear Bases

Suppose that we are given feature functions $\phi : \mathcal{S} \mapsto \mathbb{R}^D$ and $\psi : \mathcal{A} \mapsto \mathbb{R}^U$, which map the state and action spaces into low-dimensional spaces, respectively. For simplicity of analysis, we assume the following:

Assumption 1 (State and Action Features)**.** *The state features $\phi_1(\cdot), \ldots, \phi_D(\cdot)$ and action features $\psi_1(\cdot), \ldots, \psi_U(\cdot)$ are probability density functions over the state space $\mathcal{S}$ and the action space $\mathcal{A}$, respectively. Besides, the state density functions and the action density functions are both linearly independent.*

Note that this assumption is not very strong: for finite state and action spaces, any nonnegative nonzero feature vector can be normalized to be a nonnegative vector whose entries sum to 1. Similarly, for general state and action spaces, any nonnegative and nonzero integrable feature functions can be normalized to become a probability density function. Such normalization is needed for the convenience of analysis and sampling, which does not affect the scaling of relevant quantities in our complexity bounds.

Let (v^*, π^*) be a pair of optimal difference-of-value vector and optimal policy of the MDP. We adopt a (bi)linear model to approximate the value function and state-action distribution:

- The value function is approximated by a linear model, where we hope to find $\tilde{v} \in \mathbb{R}^D$ such that

$$v^*(\cdot) \approx \sum_{i=1}^{D} \tilde{v}_i \phi_i(\cdot).$$

- Representing the randomized policy π is trickier, as it is a collection of conditional distributions. We will use an implicit representation. Specifically, we use a bilinear additive model to represent μ^*, the stationary distribution of state-action pairs under the optimal policy, specified as follows.

$$\mu^*_{s,a} \approx \sum_{i=1}^{D} \sum_{u=1}^{U} \tilde{\mu}_{i,u} \phi_i(s) \psi_u(a),$$

where $\tilde{\mu} \in \mathbb{R}^{D \times U}$ is a matrix of parameters. The optimal policy is then approximated by

$$\pi^*_{s,a} \propto \sum_{i=1}^{D} \sum_{u=1}^{U} \tilde{\mu}_{i,u} \phi_i(s) \psi_u(a)\,,$$

such that $\sum_a \pi^*_{s,a} = 1$ for all $s \in \mathcal{S}$. Since ϕ_i's and ψ_u's are probability density functions, the bilinear model for μ^* is essentially an additive mixture model over the product space spanned by outer products between two classes of density functions.

Remark. Consider the case where $\mathcal{S}, \mathcal{A}$ are both large but finite sets. Let $\Phi \in \mathbb{R}^{|\mathcal{S}| \times D}$, $\Psi \in \mathbb{R}^{|\mathcal{A}| \times U}$ be the matrices consisting of column state features and column action features, respectively. We can write our models in the following matrix forms

$$\begin{aligned} v^* &\approx \Phi \tilde{v}\,, \\ \mu^* &\approx \Phi \tilde{\mu} \Psi^\top\,. \end{aligned}$$

In this case, Assumption 1 requires that the columns of Φ and Ψ be nonnegative vectors whose entries sum to 1, and that Φ, Ψ be full-rank matrices.

3.2. Reduced-Order Bellman Saddle Point Problem

The policy optimization problem (1) and the Bellman equation (3) admit linear programs (2) and (4), which are dual to each other. As suggested by Wang (2017a), they can be equivalently formulated as a minimax problem:

$$\min_{v} \max_{\mu \geq 0, \|\mu\|_{1,1}=1} \sum_{a \in \mathcal{A}} \mu_{*a}^{\top} \big((P_a - I)v + r_a\big). \tag{5}$$

Its saddle point(s) coincide with the pair(s) of optimal value function and the corresponding state-action distribution. The optimal minimax value is equal to the optimal average reward $\bar{v}^* = \sum_{a \in \mathcal{A}} (\mu_{*a}^*)^{\top} r_a$.

With the linear and bilinear models described in the previous subsection, we approximate the high-dimensional saddle point problem (5) with the following optimization problem that involves lower-dimensional variables:

$$\min_{\tilde{v} \in \mathcal{V}} \max_{\tilde{\mu} \in \mathcal{U}} \sum_{a \in \mathcal{A}} \Psi_{a*} \tilde{\mu}^{\top} \Phi^{\top} \big((P_a - I)\Phi\tilde{v} + r_a\big), \tag{6}$$

where $\mathcal{V}$ and $\mathcal{U}$ are certain primal and dual constraints to be specified later. Here, $\tilde{\mu}$ is of dimension $D \times U$. The primal and dual variables $(\tilde{v}, \tilde{\mu})$ are parameters of the bilinear models for the value function and state-action distribution. Equivalently, one can rewrite (6) into a sampling-friendly form:

$$\min_{\tilde{v} \in \mathcal{V}} \max_{\tilde{\mu} \in \mathcal{U}} \sum_{i=1}^{D} \sum_{u=1}^{U} \tilde{\mu}_{i,u} \mathbf{E}_{\phi_i, \psi_u} \left[\phi_{s'}^T \tilde{v} - \phi_s^T \tilde{v} + r_a(s)\right], \tag{7}$$

where the expectation is taken over the random variables (s, a, s') where $s \sim \phi_i$, $a \sim \psi_u$, $s' \sim P_a(s, \cdot)$.

3.3. Realizability of The Reduced Model

We introduce the following notion of realizability regarding the state and action features.

Definition 1 (Realizability). *An MDP $\mathcal{M}$ is realizable using state and action features ϕ and ψ, if there exist $\tilde{v} \in \mathbb{R}^D$ and $\tilde{\mu} \in \mathbb{R}_+^{D \times U}$ so that $v^* = \Phi\tilde{v}$ and $\mu^* = \Phi\tilde{\mu}\Psi^{\top}$.*

We have the following result:

Theorem 1. *Let $\mathcal{M}$ be an MDP that is realizable using ϕ and ψ. Then, the optimal saddle point $(\tilde{v}^*, \tilde{\mu}^*)$ to problem (7) satisfies*

$$v^* = \Phi\tilde{v}^*, \qquad \mu^* = \Phi\tilde{\mu}^*\Psi^{\top}.$$

The proof is straightforward by noting that (7) is a more restricted version of (5).

A natural question one may ask is what happens when the realizability condition does not hold. We wonder how well one can solve the high-dimensional MDP by solving the reduced-order saddle point problem (6) instead. We leave this question to Section 5.4, where we provide approximation guarantees for solving the misspecified saddle point problem, when realizability does not hold.

4. Bilinear π Learning

This section develops the Bilinear π-Learning algorithm based on given state and action features. We discuss its implementation and run-time efficiency. Its sample efficiency is the subject of the next section.

4.1. The Algorithm

We propose to use a primal-dual algorithm to solve problem (6), which is given as in Algorithm 1. The algorithm makes updates to $\tilde{v}$ and $\tilde{\mu}$ while sampling random coordinates and state transitions. Let us denote by $(i_t, u_t, s_t, a_t, r_t, s_t')$ the sample at iteration t. Let $\mathcal{U}, \mathcal{V}, M, \alpha, \beta$ be input parameters to be specified later. At the t-th iteration, the algorithm updates according to

$$\tilde{\mu}^{t+1} = \Pi_{\mathcal{U},KL}\left(\frac{\tilde{\mu}^t \exp\left(\beta\Delta_{\mu}^{t+1}\right)}{\|\tilde{\mu}^t \exp\left(\beta\Delta_{\mu}^{t+1}\right)\|_{1,1}}\right),$$
$$\tilde{v}^{t+1} = \Pi_{\mathcal{V}}\left(\tilde{v}^t - \alpha\Delta_v^{t+1}\right),$$

where $\Pi_{\mathcal{U},KL}(\mu) = \operatorname{argmin}_{\mu' \in \mathcal{U}} D_{KL}(\mu' \| \mu)$ denotes the information projection onto $\mathcal{U}$ with regard to the Kullback-Leibler divergence, $\Pi_{\mathcal{V}}(v) = \operatorname{argmin}_{v' \in \mathcal{V}} \|v - v'\|_2$ denotes the Euclidean projection onto $\mathcal{V}$, $\Delta_{\mu}^{t+1} \in \mathbb{R}^{D \times U}$ denotes the noisy gradient with respect to the dual variable, given by

$$\Delta_{\mu}^{t+1}(i,u) = \begin{cases} \frac{\left(\phi(s_t') - \phi(s_t)\right)^{\top} \tilde{v}^t + r_t - M}{\tilde{\mu}_{i_t,u_t}^t}, & \text{if } (i,u) = (i_t, u_t) \\ 0, & \text{otherwise,} \end{cases}$$

and $\Delta_v^{t+1} \in \mathbb{R}^D$ denotes the noisy partial gradient with respect to the primal variable:

$$\Delta_v^{t+1} = \phi(s_t')^{\top} - \phi(s_t)^{\top}.$$

Essentially, each iteration is a stochastic primal-dual iteration (using $\|\cdot\|_2$ and KL divergence as the primal and dual Bregman divergence functions, respectively) for solving the saddle point problem. The algorithm has several interesting features that are worth noting:

- Upon obtaining a new sample, the π-learning algorithm makes multiplicative updates on dual variables, which resembles the exponentiated gradient methods used in bandit problems (Kivinen & Warmuth, 1997; Auer et al., 2002).

- The updates on dual variables involve projection with respect to the KL divergence. Similar Bregman divergence functions have been used for regularization for policy optimization (Schulman et al., 2015; Fox et al., 2016).
- The algorithm does not need to take the sample transitions as input, which can be high-dimensional quantities. Instead, it is sufficient to take as input the feature values $\phi(s), \psi(a)$ of the state and action, so that all the computation can be carried out in the low-dimensional parameter space.

Algorithm 1 Bilinear π Learning (Average Reward)

Input: The number of iterations $T > 0$, ϕ, ψ
Input: Stepsizes α, β and offset parameter M
Set $\tilde{v}_i^1 = 0$, $\tilde{\mu}_{i,u}^1 = \frac{1}{DU}, i \in [D], u \in [U]$
Let $\tilde{\mu} = \tilde{\mu}^1$
for $t = 1, 2, 3, \ldots, T$ **do**
  Sample (i, u) according to distribution $\tilde{\mu}$
  Sample s according to distribution ϕ_i
  Sample a according to distribution ψ_u
  Sample s' and get $r_a(s)$ using (s, a) and the generative model
  $\tilde{v}^{t+1} = \Pi_{\mathcal{V}} \left(\tilde{v}^t - \alpha \left(\phi(s') - \phi(s) \right)^\top \right)$
  $\tilde{\mu}_{i,u} = \tilde{\mu}_{i,u} \cdot \exp \left\{ \beta \cdot \frac{\left(\phi(s') - \phi(s) \right)^\top \tilde{v}^t + r_a(s) - M}{\tilde{\mu}_{i,u}} \right\}$
  $\tilde{\mu} = \Pi_{\mathcal{U}, KL} \left(\frac{\tilde{\mu}}{\|\tilde{\mu}\|_{1,1}} \right)$
  $\tilde{\mu}^{t+1} = \tilde{\mu}$
end for
$\hat{\mu} = \frac{1}{T} \sum_{t=1}^T \tilde{\mu}^t$
Let $\hat{\pi}_{s,a} = \frac{\Phi_{s*} \hat{\mu} \Psi_{a*}^\top}{\sum_{b \in \mathcal{A}} \Phi_{s*} \hat{\mu} \Psi_{b*}^\top}$
Ouput: $\hat{\pi}$

4.2. Computational Efficiency

Algorithm 1 is model-free in the sense that it never attempts to estimate the transition probability model of the MDP. Instead, it makes direct updates to the parameters of the value function and state-action distribution. It is worth emphasizing that it outputs a policy, not a value function.

Scalability is a significant advantage of Algorithm 1. The algorithm uses low-dimensional memory. It maintains two variables, the value parameters $\tilde{v}$ and the policy parameters $\tilde{\mu}$. The memory size is $\mathcal{O}(DU)$, which is the dimension of the policy parameters and does not depend on the actual dimension of the MDP. Furthermore, the algorithm operates in a purely streaming mode: for example, in the case where the decision process is a sequence of images, the proposed algorithm does not need to store any past image. Each iteration of the algorithm makes sparse updates. When the sets $\mathcal{V}$,$\mathcal{U}$ are of simple forms (see the next section for examples), each iteration can be implemented in runtime that is polynomial with respect to D, U.

5. Sample Complexity

We now turn to the sample-complexity analysis of Algorithm 1. We first analyze the convergence of of an inexact duality gap associated with the primal and dual iterates. Then we use the duality gap bound to derive the number of samples needed to find an ϵ-optimal policy, provided that the realizability condition holds. Finally, we extend the analysis to the unrealizable case where the approximation error may be nonzero.

For simplicity of exposition, we focus on the case where the state and action spaces, $\mathcal{S}$ and $\mathcal{A}$, are finite. We will show that our sample complexity bounds do not scale with the sizes of the state and action spaces.

5.1. Assumptions

We make the following assumptions in the rest of the paper. They require that the state-action distributions of the MDP are within certain ranges and that the process is rapidly mixing. For convenience, we define $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$.

Assumption 2 (Uniformly Bounded Ergodicity)**.** *The Markov decision process is ergodic under any stationary policy π, and there exists $\tau > 1$ such that*

$$\frac{1}{\sqrt{\tau} S} \cdot \mathbf{1} \leq \nu^\pi \leq \frac{\sqrt{\tau}}{S} \cdot \mathbf{1},$$

where ν^π is the stationary distribution over the state space of the MDP under the policy π.

This assumption is also used by Wang (2017a). It implies that the MDP is unichain (Puterman, 2014).

Assumption 3 (Fast Mixing Time)**.** *There exists a constant $t_{mix} > 0$ such that for any stationary policy π*

$$t_{mix} \geq \min_t \left\{ t \,\middle|\, \|(P^\pi)^t(s, \cdot) - \nu^\pi\|_{\mathrm{TV}} \leq \frac{1}{4}, \forall s \in \mathcal{S} \right\},$$

where $\|\cdot\|_{\mathrm{TV}}$ stands for the total variation.

Under this assumption, there exists an optimal difference-of-value vector satisfying $\|v^*\|_\infty \leq 2t_{mix}$, according to Lemma 1 in (Wang, 2017a). *In the rest of the analysis, we focus on such v^*.*

In what follows, we choose the primal constraint $\mathcal{V}$ as

$$\mathcal{V} = \{\tilde{v} \in \mathbb{R}^D \big| \|\Phi \tilde{v}\|_\infty \leq 2t_{mix}\},$$

and the dual constraint $\mathcal{U}$ as

$$\mathcal{U} = \left\{ \tilde{\mu} \,\middle|\, \tilde{\mu} \geq 0, \|\tilde{\mu}\|_{1,1} = 1, \sum_{a \in \mathcal{A}} \Phi \tilde{\mu} \Psi^\top \geq \frac{\mathbf{1}}{\sqrt{\tau} S} \right\}.$$

Note that $\mathcal{U}$ is guaranteed to be nonempty as long as a constant-valued state feature function is included.

In the algorithm, we choose the parameters as follows:

$$M = 4t_{mix} + 1\,,$$
$$\beta = \frac{1}{5t_{mix}}\sqrt{\frac{\log(DU)}{TDU}}\,, \tag{8}$$
$$\alpha = \frac{t_{mix}}{\lambda_{\min}(\Phi^\top\Phi)\|\Phi\|_{2,\infty}}\sqrt{\frac{D}{T}}\,.$$

5.2. Primal-Dual Convergence

In this section, we analyze the convergence of Algorithm 1. It can be viewed as a stochastic approximation algorithm, in which we use only a single sample per iteration.

Let $(\check{v}, \check{\mu})$ be the best approximation to (v^*, μ^*) using the linear models and given features ϕ, ψ:

$$\check{\mu} = \text{argmin}_{\tilde{\mu}\in\mathcal{U}}\|\Phi\tilde{\mu}\Psi^\top - \mu^*\|_{1,1},$$
$$\check{v} = \text{argmin}_{\tilde{v}\in\mathcal{V}}\|\Phi\tilde{v} - v^*\|_\infty.$$

Note that $\|\cdot\|_{1,1}$ is equivalent to the total variation distance between $\Phi\tilde{\mu}\Psi^\top$ and μ^*, if viewed as distributions.

Our first result concerns the convergence of an inexact duality gap that is associated with the primal-dual iterates.

Theorem 2 (Inexact Duality Gap). *Let $\Phi \in \mathbb{R}^{S\times D}$ and $\Psi \in \mathbb{R}^{A\times U}$ satisfy Assumption 1. Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r})$ be an arbitrary MDP satisfying Assumptions 2 and 3. Then the sequence of iterates $\{\tilde{\mu}^t, \tilde{v}^t\}_{t=1}^T$ generated by Algorithm 1 satisfies the following:*

$$\sum_{a\in\mathcal{A}} r_a^\top\Phi\check{\mu}\Psi_{a*}^\top + \frac{1}{T}\sum_{t=1}^T \mathbf{E}\Big[\sum_{a\in\mathcal{A}}((I-P_a)\Phi\check{v} - r_a)^\top\Phi\tilde{\mu}^t\Psi_{a*}^\top - \sum_{a\in\mathcal{A}}(\Phi\check{\mu}\Psi_{a*}^\top)^\top(I-P_a)\Phi\tilde{v}^t\Big] = \mathcal{O}\left(t_{mix}\left(c_\Phi + \sqrt{U\log(DU)}\right)\sqrt{\frac{D}{T}}\right), \tag{9}$$

where $c_\Phi = \frac{\|\Phi\|_{2,\infty}}{\lambda_{\min}(\Phi^\top\Phi)}$ is a constant, and $\lambda_{\min}(\Phi^\top\Phi)$ is the smallest nonnegative eigenvalue of $\Phi^\top\Phi$.

The constant $c_\Phi = \frac{\|\Phi\|_{2,\infty}}{\lambda_{\min}(\Phi^\top\Phi)}$ is feature-dependent, but it does *not* scale with S. Recall that each column of Φ is a nonnegative vector that sums to 1, therefore the ratio between $\|\Phi\|_{2,\infty}$ and $\lambda_{\min}(\Phi^\top\Phi)$ does not scale with S. The constant c_Φ would be finite and dimension-free as long as $\lambda_{\min}(\Phi^\top\Phi)$ is bounded from below - a form of "restricted eigenvalue condition" that is commonly used for analysis of linear models.

To prove Theorem 2, we first establish recursions on the KL divergence $D_{KL}(\check{\mu}\|\tilde{\mu}^t) = \sum_{u=1}^U\sum_{i=1}^D \check{\mu}_{i,u}\log\frac{\check{\mu}_{i,u}}{\tilde{\mu}_{i,u}^t}$ and on the squared distance $\|\tilde{v}^t - \check{v}\|_2^2$ via the following two lemmas. We denote by $\mathcal{F}_t$ the collection of all the random variables revealed up to the t-th iteration of Algorithm 1.

Lemma 1. *Given $\mathcal{F}_t$, the KL divergence $D_{KL}(\check{\mu}\|\tilde{\mu}^{t+1})$ satisfies for $t \geq 1$ that*

$$\mathbf{E}[D_{KL}(\check{\mu}\|\tilde{\mu}^{t+1}) \mid \mathcal{F}_t] - D_{KL}(\check{\mu}\|\tilde{\mu}^t) \leq 50\beta^2 DUt_{mix}^2 + \beta\sum_{a\in\mathcal{A}}\Psi_{a*}(\tilde{\mu}^t - \check{\mu})^\top\Phi^\top((P_a - I)\Phi\tilde{v}^t + r_a)\,. \tag{10}$$

Moreover, since $\tilde{\mu}_{i,u}^1 = \frac{1}{DU}$ for all $u \in [U], i \in [D]$ as in Algorithm 1, we have $D_{KL}(\check{\mu}\|\tilde{\mu}^1) \leq \log(DU)$.

Lemma 2. *Given $\mathcal{F}_t$, the squared error $\|\tilde{v}^t - \check{v}\|_2^2$ satisfies*

$$\mathbf{E}[\|\tilde{v}^{t+1} - \check{v}\|_2^2 \mid \mathcal{F}_t] - \|\tilde{v}^t - \check{v}\|_2^2 \leq 2\alpha\sum_{a\in\mathcal{A}}\Psi_{a*}(\tilde{\mu}^t)^\top\Phi^\top(I - P_a)\Phi(\tilde{v}^t - \check{v}) + 4\alpha^2\|\Phi\|_{2,\infty}^2, \tag{11}$$

and $\|\tilde{v}^1 - \check{v}\|_2^2 \leq \frac{4Dt_{mix}^2}{\lambda_{\min}^2(\Phi^\top\Phi)}$.

The proofs of Lemmas 1 and 2 are deferred to the appendix. Using these two lemmas, we prove Theorem 2 as follows.

Proof of Theorem 2. We first define $\mathcal{G}_t = \sum_{a\in\mathcal{A}}(\Phi\tilde{\mu}^t\Psi_{a*}^\top)^\top((I-P_a)\Phi\check{v} - r_a) + \sum_{a\in\mathcal{A}} r_a^\top\Phi\check{\mu}\Psi_{a*}^\top - \sum_{a\in\mathcal{A}}(\Phi\check{\mu}\Psi_{a*}^\top)^\top(I-P_a)\Phi\tilde{v}^t$. If we take a weighted sum between (10) and (11), we obtain

$$\mathcal{G}_t \leq \frac{D_{KL}(\check{\mu}\|\tilde{\mu}^t) - \mathbf{E}[D_{KL}(\check{\mu}\|\tilde{\mu}^{t+1}) \mid \mathcal{F}_t]}{\beta} + 50\beta DUt_{mix}^2 + \frac{\|\tilde{v}^t - \check{v}\|_2^2 - \mathbf{E}[\|\tilde{v}^{t+1} - \check{v}\|_2^2 \mid \mathcal{F}_t]}{2\alpha} + 2\alpha\|\Phi\|_{2,\infty}^2\,.$$

Applying the stepsizes in (8), taking the average over t, and using iterated expectations, we have

$$\frac{1}{T}\sum_{t=1}^T\mathbf{E}[\mathcal{G}_t] \leq \frac{D_{KL}(\check{\mu}\|\tilde{\mu}^1)}{T\beta} + 50\beta DUt_{mix}^2 + \frac{\|\tilde{v}^1 - \check{v}\|_2^2}{2T\alpha} + 2\alpha\|\Phi\|_{2,\infty}^2 \leq 15t_{mix}\sqrt{\frac{DU\log(DU)}{T}} + \frac{4t_{mix}\|\Phi\|_{2,\infty}}{\lambda_{\min}(\Phi^\top\Phi)}\sqrt{\frac{D}{T}},$$

where the second inequality is due to Lemmas 1 and 2. Recall that $\frac{1}{T}\sum_{t=1}^T\mathbf{E}[\mathcal{G}_t]$ is exactly the LHS of equation (9). The proof is thus completed. □

5.3. Convergence To Optimal Policy: The Realizable Case

We now study the sample complexity of Algorithm 1 for obtaining a near-optimal policy in the realizable case. Recall that we are interested in finding an approximately-optimal policy which performs comparably to the optimal policy in the Markov decision process. Therefore, we wish to quantify the policy error.

Theorem 3 (Convergence with Realizability). *Let $\Phi \in \mathbb{R}^{S\times D}$ and $\Psi \in \mathbb{R}^{A\times U}$ satisfy Assumption 1. Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r})$ be an arbitrary MDP satisfying Assumptions 2 and 3. Suppose that the MDP $\mathcal{M}$ is realizable using state and action features Φ, Ψ. Then, the policy $\hat{\pi}$ generated by Algorithm 1 after running T time steps satisfies*

$$\bar{v}^* - \mathbf{E}[\bar{v}^{\hat{\pi}}] = \mathcal{O}\left(\tau t_{mix}\left(c_\Phi + \sqrt{U\log(DU)}\right)\sqrt{\frac{D}{T}}\right),$$

where $\bar{v}^{\hat{\pi}}$ is the average reward of the policy $\hat{\pi}$.

Proof. In the realizable case, we can write $\mu^* = \Phi\check{\mu}\Psi^\top$ and $v^* = \Phi\check{v}$. Note that μ^* is the stationary distribution under the optimal policy, which gives us that $\sum_{a\in\mathcal{A}}(\Phi\check{\mu}\Psi_{a*}^\top)^\top(I - P_a) = \sum_{a\in\mathcal{A}}(\mu_{*a}^*)^\top(I - P_a) = 0$, where μ_{*a}^* is the a-th column of μ^*. It is also known that $\sum_{a\in\mathcal{A}}(\mu_{*a}^*)^\top r_a = \bar{v}^*$. We can simplify the result of Theorem 2 to

$$\begin{aligned}\bar{v}^* + \frac{1}{T}\sum_{t=1}^{T}\mathbf{E}\Big[\sum_{a\in\mathcal{A}}((I - P_a)v^* - r_a)^\top\Phi\tilde{\mu}^t\Psi_{a*}^\top\Big] \\ = \mathcal{O}\left(t_{mix}\left(c_\Phi + \sqrt{U\log(DU)}\right)\sqrt{\frac{D}{T}}\right),\end{aligned} \tag{12}$$

where $\tilde{\mu}^t$ is the sequence of dual variables computed in Algorithm 1.

Recall that the probability of choosing action a at state s, for the policy output by Algorithm 1, is

$$\hat{\pi}_{s,a} = \frac{\Phi_{s*}\hat{\mu}\Psi_{a*}^\top}{\sum_{a'\in\mathcal{A}}\Phi_{s*}\hat{\mu}\Psi_{a'*}^\top},$$

where $\hat{\mu} = \frac{1}{T}\sum_{t=1}^{T}\tilde{\mu}^t$. We let $\xi = (\xi_1, \ldots, \xi_S)$ be such that $\xi_s = \sum_{a'\in\mathcal{A}}\Phi_{s*}\hat{\mu}\Psi_{a'*}^\top$ for $s \in \mathcal{S}$. Denote by $\nu^{\hat{\pi}}$ the stationary distribution of the MDP under the policy $\hat{\pi}$. Using the definition of the average reward, we have

$$\begin{aligned}\bar{v}^* - \bar{v}^{\hat{\pi}} &= \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\nu_s^{\hat{\pi}}\hat{\pi}_{s,a}(\bar{v}^* - r_a(s)) \\ &= \sum_{a\in\mathcal{A}}(\nu^{\hat{\pi}})^\top\mathrm{diag}(\hat{\pi}_{*a})((I - P_a)v^* + \bar{v}^*\cdot\mathbf{1} - r_a),\end{aligned}$$

where the last equality is because $\nu^{\hat{\pi}}$ is the stationary distribution and hence $\sum_{a\in\mathcal{A}}(\nu^{\hat{\pi}})^\top\mathrm{diag}(\hat{\pi}_{*a})(I - P_a) = \mathbf{0}$. From Assumption 2 and the definition of the dual constraint $\mathcal{U}$, we have $\nu_s^{\hat{\pi}} \le \frac{\sqrt{\tau}}{S} = \tau\cdot\frac{1}{\sqrt{\tau}S} \le \tau\xi_s$ for any state s, where the second inequality uses the fact that the dual iterates are always projected onto $\mathcal{U}$. Also, we have $(I - P_a)v^* + \bar{v}^*\cdot\mathbf{1} - r_a \ge 0$ by the Bellman equation. This gives us

$$\bar{v}^* - \bar{v}^{\hat{\pi}} \le \tau\sum_{a\in\mathcal{A}}\xi^\top\mathrm{diag}(\hat{\pi}_{*a})((I - P_a)v^* + \bar{v}^*\cdot\mathbf{1} - r_a).$$

By the definition of ξ, we have $\xi_s\hat{\pi}_{s,a} = \frac{1}{T}\sum_{t=1}^{T}\Phi_{s*}\tilde{\mu}^t\Psi_{a*}^\top$. Also note that $\sum_{a\in\mathcal{A}}\xi^\top\mathrm{diag}(\hat{\pi}_{*a})\cdot\mathbf{1} = 1$. It then follows from the last expression and equation (12) that

$$\begin{aligned}&\bar{v}^* - \mathbf{E}[\bar{v}^{\hat{\pi}}] \\ &\le \tau\left(\bar{v}^* + \frac{1}{T}\sum_{t=1}^{T}\mathbf{E}\left[\sum_{a\in\mathcal{A}}((I - P_a)v^* - r_a)^\top\Phi\tilde{\mu}^t\Psi_{a*}^\top\right]\right) \\ &= \mathcal{O}\left(\tau t_{mix}\left(c_\Phi + \sqrt{U\log(DU)}\right)\sqrt{\frac{D}{T}}\right),\end{aligned} \tag{13}$$

which completes the proof. □

Theorem 3 suggests that, the bilinear π learning algorithm achieves an ϵ-optimal policy by using the sample size

$$\mathcal{O}\left(\frac{DU\log(DU)}{\epsilon^2}\right),$$

with other parameters fixed. This sample complexity bound depends linearly on the dimension of policy parameters. It is scale-free with respect to the sizes of the underlying MDP.

5.4. Convergence To Approximately Optimal Policy: The Unrealizable Case

We finally turn to the case where the realizability may not hold. The following theorem generalizes the results of Theorem 3 to the unrealizable case.

Theorem 4 (Convergence without Realizability). *Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r})$ be an arbitrary MDP satisfying Assumptions 2 and 3. Let $\Phi \in \mathbb{R}^{S\times D}$ and $\Psi \in \mathbb{R}^{A\times U}$ satisfy Assumption 1. Then the policy computed by Algorithm 1 after running T time steps satisfies*

$$\begin{aligned}\bar{v}^* - \mathbf{E}[\bar{v}^{\hat{\pi}}] = \mathcal{O}\Bigg(&\tau t_{mix}\left(c_\Phi + \sqrt{U\log(DU)}\right)\sqrt{\frac{D}{T}} \\ &+ \tau\cdot\min_{\tilde{v}\in\mathcal{V}}\|\Phi\tilde{v} - v^*\|_\infty + \tau t_{mix}\cdot\min_{\tilde{\mu}\in\mathcal{U}}\|\Phi\tilde{\mu}\Psi^\top - \mu^*\|_{1,1}\Bigg),\end{aligned}$$

where $\bar{v}^{\hat{\pi}}$ is the average reward of the policy $\hat{\pi}$.

The proof requires a careful characterization of the approximation error, and is deferred to the appendix. The approximation error has two parts. The first part $\min_{\tilde{v}\in\mathcal{V}}\|\Phi\tilde{v} - v^*\|_\infty$ is the ℓ_∞ distance from the optimal value function v^* to the span of state features. The second part $\min_{\tilde{\mu}\in\mathcal{U}}\|\Phi\tilde{\mu}\Psi^\top - \mu^*\|_{1,1}$ is the total variation distance between the optimal state-action distribution and the best approximate distribution in the product space generated by state and action features.

6. Related Literatures

The primal-dual approach we developed in this paper is based on the well-known linear program (LP) formulation of the Bellman equation (d'Epenoux, 1963; Puterman, 2014). The exact LP has been extended to approximate linear programs (ALP) to tackle large-scale problems, by using compact representations of the value function. For example, Schweitzer & Seidmann (1985) and de Farias & Van Roy (2003) propose to represent the value function as a linear combination of basis functions, but the number of constraints depends on the number of state-action pairs, which motivated a lot of research on how to select a small subset of constraints without introducing much suboptimality in the reduced ALP (e.g., de Farias & Van Roy (2004)). Others have studied various approaches to *compress* the large constraint set into a smaller one (Taylor & Parr, 2012; Lakshminarayanan et al., 2017). These prior works focus on the *quality* of the ALP solution—the distance it is from the optimal value function, but not so much on developing efficient algorithms to solve it. An exception is Abbasi-Yadkori et al. (2014), who reduces the ALP to stochastic convex optimization. However, their algorithm requires certain knowledge of the transition probabilities that is not easily available for all problems. In contrast, our work provides a concrete algorithm to solve the ALP when a bilinear representation is used, with strong guarantees on its computation and sample complexity; the algorithm only requires a sampling oracle.

A similar saddle point approach has been studied by other authors. Wang & Chen (2016); Wang (2017b;a) propose provably fast solvers for finite MDPs, but those algorithms are not expected to scale well when the number of states/actions becomes large. Our algorithm's complexity depends on the number of basis functions, which has a much lower dimension than the number of states and actions. Dai et al. (2018) considers a related yet different primal-dual formulation that also allows compact representations, but no rate of convergence is provided. It should be noted that there is another line of research that studies saddle point formulations that result from the fixed-point (as opposed to LP) view of the Bellman equation for fixed policy *evaluation* (Mahadevan et al., 2014; Macua et al., 2015; Du et al., 2017; Dai et al., 2017). In contrast, we consider policy optimization, which is substantially more challenging. It is an interesting future direction to study the connections between these two saddle point formulations.

While our work is mostly related to the ALP family, another important class of methods to solve large-scale MDPs is approximate dynamic programming, or ADP (Bertsekas & Tsitsiklis, 1996). Linear basis functions have been used for both policy evaluation and optimization (Tsitsiklis & Van Roy, 1997; Nedić & Bertsekas, 2003; Lagoudakis & Parr, 2003; Melo et al., 2008; Sutton et al., 2009). However, except in certain special cases, ADP with linear approximation can be inherently unstable, leading to undesired situations including oscillation and even divergence. The Greedy-GQ is an interesting control algorithm with linear approximation that has convergence guarantees under a relatively mild assumption (Maei et al., 2010). But no finite-sample convergence is provided, and the algorithm requires two time-scale updates that can cause difficulties in practice. Our primal-dual algorithm not only is provably stable, but also enjoys strong finite-sample convergence rate.

Finally, our use of bilinear representations to compactly represent distributions over state-action pairs also appears new to the best of our knowledge. The most relevant model that has been studied is proposed by Elkan (2011), which used a bilinear state-action representation for Q-functions. Previous work also uses linear approximation of value functions (Bertsekas & Tsitsiklis, 1996; Elkan, 2011), or state transition probabilities (Wang et al., 2007; Yang et al., 2009). In contrast, our representation allows one to work with large state and/or action spaces: the state and action are first mapped to their respective low-dimensional spaces, and a weight matrix is used to compute the distribution of that state-action pair. Such a technique may find use in other context when a distribution over state-action needs to be represented explicitly.

7. Summary

We provide a scalable model-free method, bilinear π learning, for reinforcement learning, when a sampling oracle is provided. The method has its origin in a primal-dual formulation of the policy optimization problem. It adopts (bi)linear models to approximate the high-dimensional value functions and state-action distributions, using given state and action features. The approach enjoys a number of advantages. First, it is a compact method that has very small memory footprint and makes updates to low-dimensional variables. Its run-time and space complexities do not depend on the size of the MDP, so is scalable to large-scale reinforcement learning problems. Furthermore, the π-learning method is sample-efficient, which solves the approximate Bellman saddle point problem to high precision using a small number of observations. The sample complexity of the compact π-learning method is linear in the dimension of the reduced parameter space, and again does not depend on the size of the underlying MDP.

We mention a few exciting directions for future research. First, how to generalize our approach to nonlinear approximations such as neural networks? Second, how to adapt the algorithm to online reinforcement learning, with the learner following a single trajectory, with PAC (Strehl et al., 2009) or regret (Jaksch et al., 2010) guarantees? This will require explicitly addressing the exploration problem.

References

Abbasi-Yadkori, Y., Bartlett, P. L., and Malek, A. Linear programming for large-scale Markov decision problems. In *Proceedings of the 31st International Conference on Machine Learning*, pp. 496–504, 2014.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

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.

Bertsekas, D. P. *Dynamic programming and optimal control*, volume 1. Athena Scientific, Belmont, MA, 1995.

Bertsekas, D. P. and Tsitsiklis, J. N. *Neuro-Dynamic Programming*. Athena Scientific, September 1996. ISBN 1-886529-10-8.

Dai, B., He, N., Pan, Y., Boots, B., and Song, L. Learning from conditional distributions via dual embeddings. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1458–1467, 2017.

Dai, B., Shaw, A., He, N., Li, L., and Song, L. Boosting the actor with dual critic. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018. arXiv:1712.10282.

de Farias, D. P. and Van Roy, B. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

de Farias, D. P. and Van Roy, B. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478, 2004.

d'Epenoux, F. A probabilistic production and inventory problem. *Management Science*, 10(1):98–108, 1963.

Dietterich, T. G., Taleghan, M. A., and Crowley, M. PAC optimal planning for invasive species management: Improved exploration for reinforcement learning from simulator-defined MDPs. In *AAAI*, 2013.

Du, S. S., Chen, J., Li, L., Xiao, L., and Zhou, D. Stochastic variance reduction methods for policy evaluation. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1049–1058, 2017.

Elkan, C. Reinforcement learning with a bilinear Q function. In *Recent Advances in Reinforcement Learning - 9th European Workshop (EWRL)*, number 7188 in Lecture Notes in Computer Science, pp. 78–88, 2011.

Fox, R., Pakman, A., and Tishby, N. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence (UAI-16)*, 2016.

Jaksch, T., Ortner, R., and Auer, P. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600, 2010.

Kivinen, J. and Warmuth, M. K. Exponentiated gradient versus gradient descent for linear predictors. *Inform. and Comput.*, 132(1):1–63, 1997. ISSN 0890-5401. doi: 10.1006/inco.1996.2612.

Lagoudakis, M. G. and Parr, R. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

Lakshminarayanan, C., Bhatnagar, S., and Szepesvári, C. A linearly relaxed approximate linear program for Markov decision processes. *IEEE Transactions on Automatic Control*, 63(4):1185–1191, 2017.

Macua, S. V., Chen, J., Zazo, S., and Sayed, A. H. Distributed policy evaluation under multiple behavior strategies. *IEEE Transactions on Automatic Control*, 60(5): 1260–1274, 2015.

Maei, H. R., Szepesvri, C., Bhatnagar, S., and Sutton, R. S. Toward off-policy learning control with function approximation. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 719–726, 2010.

Mahadevan, S., Liu, B., Thomas, P. S., Dabney, W., Giguere, S., Jacek, N., Gemp, I., and Liu, J. Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces, 2014. CoRR abs/1405.6757.

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 (ICML)*, pp. 664–671, 2008.

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

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

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P. Trust region policy optimization. In *Proceedings of the Thirty-Second International Conference on Machine Learning (ICML-15)*, pp. 1889–1897, 2015.

Schweitzer, P. J. and Seidmann, A. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110(2):568–582, 1985.

Strehl, A. L., Li, L., and Littman, M. L. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10:2413–2444, 2009.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 1998.

Sutton, R. S., Szepesvári, C., and Maei, H. R. A convergent $o(n)$ algorithm for off-policy temporal-difference learning with linear function approximation. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pp. 1609–1616, 2009.

Taleghan, M. A., Dietterich, T. G., Crowley, M., Hall, K., and Albers, H. J. PAC optimal mdp planning with application to invasive species management. *Journal of Machine Learning Research*, 16:3877–3903, 2015.

Taylor, G. and Parr, R. Value function approximation in noisy environments using locally smoothed regularized approximate linear programs. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, pp. 835–842, 2012.

Tsitsiklis, J. N. and Van Roy, B. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42:674–690, 1997.

Wang, M. Primal-dual π learning: Sample complexity and sublinear run time for ergodic markov decision problems. *CoRR*, abs/1710.06100, 2017a.

Wang, M. Randomized linear programming solves the discounted Markov decision problem in nearly-linear running time. *arXiv preprint arXiv:1704.01869*, 2017b.

Wang, M. and Chen, Y. An online primal-dual method for discounted Markov decision processes. In *IEEE Conference of Decisions and Control*, 2016.

Wang, T., Lizotte, D. J., Bowling, M. H., and Schuurmans, D. Stable dual dynamic programming. In *Advances in Neural Information Processing Systems 20 (NIPS-07)*, pp. 1569–1576, 2007.

Yang, M., Li, Y., and Schuurmans, D. Dual temporal difference learning. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 631–638, 2009.