Journal of Machine Learning Research ??? (???) ??? Submitted ???; Published ???

Dynamic Policy Programming

Mohammad Gheshlaghi Azar M.AZAR@SCIENCE.RU.NL
Vicenç Gómez V.GOMEZ@SCIENCE.RU.NL
Hilbert J. Kappen B.KAPPEN@SCIENCE.RU.NL
Department of Biophysics
Radboud University Nijmegen
6525 EZ Nijmegen, The Netherlands

Editor:

Abstract

In this paper, we propose a novel policy iteration method, called dynamic policy programming (DPP), to estimate the optimal policy in the infinite-horizon Markov decision processes. DPP is an incremental algorithm that forces a gradual change in policy update. This allows to prove finite-iteration and asymptotic ℓ_∞-norm performance-loss bounds in the presence of approximation/estimation error which depend on the average accumulated error as opposed to the standard bounds which are expressed in terms of the supremum of the errors. The dependency on the average error is important in problems with limited number of samples per iteration, for which the average of the errors can be significantly smaller in size than the supremum of the errors. Based on these theoretical results, we prove that a sampling-based variant of DPP (DPP-RL) asymptotically converges to the optimal policy. Finally, we illustrate numerically the applicability of these results on some benchmark problems and compare the performance of the approximate variants of DPP with some existing reinforcement learning (RL) methods.

Keywords: Approximate dynamic programming, reinforcement learning, Markov decision processes, Monte-Carlo methods, function approximation.

1. Introduction

Many problems in robotics, operations research and process control can be represented as a control problem that can be solved by finding the optimal policy using *dynamic programming* (DP). DP is based on estimating some measures of the value of state-action $Q^*(x, a)$ through the Bellman equation. For high-dimensional discrete systems or for continuous systems, computing the value function by DP is intractable. The common approach to make the computation tractable is to approximate the value function using function-approximation and Monte-Carlo sampling (Szepesvári, 2010; Bertsekas and Tsitsiklis, 1996). Examples of such approximate dynamic programming (ADP) methods are approximate policy iteration (API) and approximate value iteration (AVI) (Bertsekas, 2007; Lagoudakis and Parr, 2003; Perkins and Precup, 2002; de Farias and Roy, 2000). In addition to these approaches, there are methods which do not rely exclusively on an approximate value function. These methods include, for instance, actor-critic methods (Barto et al., 1983), which explicitly consider

two interacting processes, policy gradient methods (Baxter and Bartlett, 2001; Sutton et al., 1999), and dual dynamic programming (Wang et al., 2007a,b).

ADP methods have been successfully applied to many real world problems, and theoretical results have been derived in the form of finite iteration and asymptotic performance guarantee of the induced policy. In particular, the formal analysis of these algorithms is usually characterized in terms of bounds on the difference between the optimal and the estimated value function induced by the algorithm (performance loss) (Farahmand et al., 2010; Thiery and Scherrer, 2010; Munos, 2005; Bertsekas and Tsitsiklis, 1996). For instance, in the case of AVI and API, the asymptotic ℓ_∞-norm performance-loss bounds in the presence of approximation error ϵ_k can be expressed as

$$\limsup_{k\to\infty}\|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}\limsup_{k\to\infty}\|\epsilon_k\|\,, \tag{1}$$

where γ denotes the discount factor, $\|\cdot\|$ is the ℓ_∞-norm w.r.t. the state-action pair (x,a) and π_k is the control policy at iteration k.

The bound of Equation (1) is expressed in terms of the supremum of the approximation errors. Intuitively, the dependency on the supremum error means that to have a small overall performance loss the approximation errors of all iterations should be small in size, i.e., a large approximation error in only one iteration can derail the whole learning process. This can cause a major problem when the approximation error ϵ_k arises from sampling. In many problems of interest, the sampling error can be large and hard to control, since only a limited number of samples can be used at each iteration. Also, even in those cases where we have access to large number of samples, it may be difficult, if not impossible, to control the size of errors for all iterations. This is due to the fact that the sampling errors are random objects and, regardless of the number of samples used at each iteration, there is always a fair chance that in some few *out-lier* iterations the sampling errors take large values in their interval of definition. In all those cases, a bound which depends on the average accumulated error $\bar{\epsilon}_k = 1/(k+1)\sum_{j=0}^{k}\epsilon_j$ instead of the supremum error is preferable. The rationale behind this idea is that the average of the sum of random variables, under some mild assumptions, can be significantly smaller in size than the supremum of the random variables. Also, the average error $\bar{\epsilon}_k$ is less sensitive to the outliers than the supremum error. Therefore, a bound which depends on the average error can be tighter than the one with dependency on the supremum error. To the best of authors' knowledge, there exists no previous work that provides such a bound.

In this paper, we propose a new incremental policy-iteration algorithm called dynamic policy programming (DPP). DPP addresses the above problem by proving the first asymptotic and finite-iteration performance loss bounds with dependency on $\|\bar{\epsilon}_k\|$. This implies the previously mentioned advantages in terms of performance guarantees. The intuition is that DPP, by forcing an incremental change between two consecutive policies, accumulates the approximation errors of all the previous iterations, rather than just minimizing the approximation error of the current iteration. We also introduce a new RL algorithm based on the DPP update rule, called DPP-RL, and prove that it converges to the optimal policy with the convergence rate of order $1/\sqrt{k}$. This rate of convergence leads to a PAC ("probably approximately correct") sample-complexity bound of order $O(1/((1-\gamma)^6\epsilon^2))$ to find an ϵ-optimal policy with high probability, which is superior to the best existing

result of standard Q-learning (Even-Dar and Mansour, 2003). See Section 6 for a detailed comparison with incremental RL algorithms such as Q-learning and SARSA.

DPP shares some similarities with the well-known actor-critic (AC) method of Barto et al. (1983), since both methods make use of an approximation of the optimal policy by means of action preferences and soft-max policy. However, DPP uses a different update rule which is only expressed in terms of the action preferences and does not rely on the estimate of the value function to criticize the control policy.

The contribution of this work is mainly theoretical, and focused on the problem of estimating the optimal policy in an infinite-horizon MDP. Our setting differs from the standard RL in the following: we rely on a *generative model* from which samples can be drawn. This means that the agent has full control on the sample queries that can be made about for any arbitrary state. Such an assumption is commonly made in theoretical studies of RL algorithms (Farahmand et al., 2008a; Munos and Szepesvári, 2008; Kearns and Singh, 1999) because it simplifies the analysis of learning and exploration to a great extent. We compare DPP empirically with other methods that make use of this assumption. The reader should notice that this premise does not mean that the agent needs explicit knowledge of the model dynamics to perform the required updates, nor does it need to learn one.

This article is organized as follows. In Section 2, we present the notation which is used in this paper. We introduce DPP and we investigate its convergence properties in Section 3. In Section 4, we demonstrate the compatibility of our method with the approximation techniques by generalizing DPP bounds to the case of function approximation and Monte-Carlo sampling. We also introduce a new convergent RL algorithm, called DPP-RL, which relies on a sampling-based variant of DPP to estimate the optimal policy. Section 5, presents numerical experiments on several problem domains including the optimal replacement problem (Munos and Szepesvári, 2008) and a stochastic grid world. In Section 6 we briefly review some related work. Finally, in Section 7, we summarize our results and discuss some of the implications of our work.

2. Preliminaries

In this section, we introduce some concepts and definitions from the theory of Markov decision processes (MDPs) and reinforcement learning (RL) as well as some standard notation.[1] We begin by the definition of the ℓ_2-norm (Euclidean norm) and the ℓ_∞-norm (supremum norm). Assume that $\mathcal{Y}$ is a finite set. Given the probability measure μ over $\mathcal{Y}$, for a real-valued function $g : \mathcal{Y} \to \mathbb{R}$, we shall denote the ℓ_2-norm and the weighted $\ell_{2,\mu}$-norm of g by $\|g\|_2^2 \triangleq \sum_{y\in\mathcal{Y}} g(y)^2$ and $\|g\|_{2,\mu}^2 \triangleq \sum_{y\in\mathcal{Y}} \mu(y)g(y)^2$, respectively. Also, the ℓ_∞-norm of g is defined by $\|g\| \triangleq \max_{y\in\mathcal{Y}} |g(y)|$ and $\log(\cdot)$ denotes the natural logarithm.

2.1 Markov Decision Processes

A discounted MDP is a quintuple $(\mathcal{X}, \mathcal{A}, P, \mathcal{R}, \gamma)$, where $\mathcal{X}$ and $\mathcal{A}$ are, respectively, the state space and the action space. P shall denote the state transition distribution and $\mathcal{R}$ denotes the reward kernel. $\gamma \in [0, 1)$ denotes the discount factor. The transition P is a probability kernel over the next state upon taking action a from state x, which we shall denote by

1. For further reading see Szepesvári (2010).

$P(\cdot|x,a)$. $\mathcal{R}$ is a set of real-valued numbers. A reward $r(x,a) \in \mathcal{R}$ is associated with each state x and action a.

Assumption 1 (MDP Regularity) *We assume $\mathcal{X}$ and $\mathcal{A} = \{a_1, a_2, \ldots, a_L\}$ are finite sets. Also, the absolute value of the immediate reward $r(x,a)$ is bounded from above by $R_{\max} > 0$ for all $(x,a) \in \mathcal{Z}$.*

Remark 1 *To keep the representation succinct, we make use of the short-hand notation $\mathcal{Z}$ for the joint state-action space $\mathcal{X} \times \mathcal{A}$. We also denote $R_{\max}/(1-\gamma)$ by $V_{\max}$.*

A Markovian policy kernel determines the distribution of the control action given the current state. The policy is called stationary if the distribution of the control action is independent of time. Given the current state x, we shall denote the Markovian stationary policy, or in short only policy, by $\pi(\cdot|x)$. A policy is called deterministic if for any state $x \in \mathcal{X}$ there exists some action a such that $\pi(a|x) = 1$. Given the policy π its corresponding value function $V^\pi : \mathcal{X} \to \mathbb{R}$ denotes the expected total discounted reward in each state x, when the action is chosen by policy π which we denote by $V^\pi(x)$. Often it is convenient to associate value functions not with states but with state-action pairs. Therefore, we introduce $Q^\pi : \mathcal{Z} \to \mathbb{R}$ as the expected total discounted reward upon choosing action a from state x and then following policy π, which we shall denote by $Q^\pi(x,a)$. We define the *Bellman operator* $\mathcal{T}^\pi$ on the action-value functions by:

$$\mathcal{T}^\pi Q(x,a) \triangleq r(x,a) + \gamma \sum_{(y,b)\in\mathcal{Z}} P(y|x,a)\pi(b|y)Q(y,b), \qquad \forall (x,a) \in \mathcal{Z}.$$

We notice that Q^π is the fixed point of $\mathcal{T}^\pi$.

The goal is to find a policy π^* that attains the *optimal value function*, $V^*(x) \triangleq \sup_\pi V^\pi(x)$, at all states $x \in \mathcal{X}$. The optimal value function satisfies the Bellman equation:

$$\begin{aligned} V^*(x) &= \sup_{\pi(\cdot|x)} \sum_{\substack{y\in\mathcal{X}\\ a\in\mathcal{A}}} \pi(a|x)\left[r(x,a) + P(y|x,a)V^*(y)\right] \\ &= \max_{a\in\mathcal{A}} \left[r(x,a) + \sum_{y\in\mathcal{X}} P(y|x,a)V^*(y) \right] \end{aligned} \quad , \qquad \forall x \in \mathcal{X}. \qquad (2)$$

Likewise, the *optimal action-value function* Q^* is defined by $Q^*(x,a) = \sup_\pi Q^\pi(x,a)$ for all $(x,a) \in \mathcal{Z}$. We shall define the *Bellman optimality operator* $\mathcal{T}$ on the action-value functions as

$$\mathcal{T}Q(x,a) \triangleq r(x,a) + \gamma \sum_{y\in\mathcal{X}} P(y|x,a) \max_{b\in\mathcal{A}} Q(y,b), \qquad \forall (x,a) \in \mathcal{Z}.$$

Q^* is the fixed point of $\mathcal{T}$.

Both $\mathcal{T}$ and $\mathcal{T}^\pi$ are contraction mappings, w.r.t. the supremum norm, with the factor γ (Bertsekas, 2007, chap. 1). In other words, for any two real-valued action-value functions Q and Q' and every policy π, we have:

$$\left\|\mathcal{T}Q - \mathcal{T}Q'\right\| \le \gamma \left\|Q - Q'\right\|, \qquad \left\|\mathcal{T}^\pi Q - \mathcal{T}^\pi Q'\right\| \le \gamma \left\|Q - Q'\right\|. \qquad (3)$$

The policy distribution π defines a right-linear operator $P^\pi \cdot$ as

$$(P^\pi Q)(x,a) \triangleq \sum_{(y,b)\in\mathcal{Z}} \pi(b|y)P(y|x,a)Q(y,b), \qquad \forall (x,a)\in\mathcal{Z}.$$

Further, we define two other right-linear operators $\pi\cdot$ and $P\cdot$ as

$$(\pi Q)(x) \triangleq \sum_{a\in\mathcal{A}} \pi(a|x)Q(x,a), \quad \forall x\in\mathcal{X},$$
$$(PV)(x,a) \triangleq \sum_{y\in\mathcal{X}} P(y|x,a)V(y), \quad \forall (x,a)\in\mathcal{Z}.$$

We define the max operator $\mathcal{M}$ on the action value functions as $(\mathcal{M}Q)(x) \triangleq \max_{a\in\mathcal{A}} Q(x,a)$, for all $x\in\mathcal{X}$. Based on the new definitions one can rephrase the Bellman operator and the Bellman optimality operator as

$$\mathcal{T}^\pi Q(x,a) = r(x,a) + \gamma(P^\pi Q)(x,a), \qquad \mathcal{T}Q(x,a) = r(x,a) + \gamma(P\mathcal{M}Q)(x,a). \tag{4}$$

In the sequel, we repress the state(-action) dependencies in our notation wherever these dependencies are clear, e.g., $\Psi(x,a)$ becomes Ψ, $Q(x,a)$ becomes Q. Also, for simplicity of the notation, we remove some parenthesis, e.g., writing $\mathcal{M}Q$ for $(\mathcal{M}Q)$ and $P^\pi Q$ for $(P^\pi Q)$, when there is no possible confusion.

3. Dynamic Policy Programming

In this section, we introduce and analyze the DPP algorithm. We first present the *dynamic policy programming* (DPP) algorithm in Subsection 3.1 (see Appendix A for some intuition on how DPP can be related to the Bellman equation). we then investigate the finite-iteration and the asymptotic behavior of DPP and prove its convergence in Subsection 3.2.

3.1 Algorithm

DPP is a policy iteration algorithm which represents the policy π_k in terms of some action preference numbers Ψ_k (Sutton and Barto, 1998, chap. 2.8). Starting at Ψ_0, DPP iterates the action preferences of all state-action pairs $(x,a)\in\mathcal{Z}$ through the following Bellman-like recursion (the pseudo code of DPP is presented in Algorithm 1):

$$\Psi_{k+1}(x,a) = \mathcal{O}\Psi_k(x,a) \triangleq \Psi_k(x,a) - (\mathcal{M}_\eta\Psi_k)(x) + r(x,a) + \gamma(P\mathcal{M}_\eta\Psi_k)(x,a),$$

where $\mathcal{O}$ and $\mathcal{M}_\eta$ denote the DPP and the softmax operators, respectively. The softmax operator $\mathcal{M}_\eta$ is defined on every $f:\mathcal{Z}\to\mathbb{R}$ as

$$(\mathcal{M}_\eta f)(x) \triangleq \frac{\sum_{a\in\mathcal{A}} \exp(\eta f(x,a)) f(x,a)}{\sum_{b\in\mathcal{A}} \exp(\eta f(x,b))},$$

where $\eta > 0$ is the inverse temperature.

The control policy π_k is then computed as a function of Ψ_k at each iteration k:

$$\pi_k(a|x) = \frac{\exp(\eta \Psi_k(x,a))}{\sum_{b \in \mathcal{A}} \exp(\eta \Psi_k(x,b))}, \qquad \forall (x,a) \in \mathcal{Z}. \tag{5}$$

Based on (5) one can re-express the DPP operator on the action preferences Ψ_k as

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) + \mathcal{T}^{\pi_k} \Psi_k(x,a) - \pi_k \Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}. \tag{6}$$

Algorithm 1: (DPP) Dynamic Policy Programming

Input: Action preferences $\Psi_0(\cdot,\cdot)$, γ and η

for $k = 0, 1, 2, \ldots, K-1$ **do** `// main loop`

  foreach $(x,a) \in \mathcal{Z}$ **do** `// compute the control policy`

    $\pi_k(a|x) := \dfrac{\exp(\eta \Psi_k(x,a))}{\sum_{b \in \mathcal{A}} \exp(\eta \Psi_k(x,b))}$;

  end

  foreach $(x,a) \in \mathcal{Z}$ **do** `// compute the new action-preferences`

    $\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}^{\pi_k} \Psi_k(x,a) - \pi_k \Psi_k(x)$; `// DPP update rule`

  end

end

foreach $(x,a) \in \mathcal{Z}$ **do** `// compute the last policy`

  $\pi_K(a|x) := \dfrac{\exp(\eta \Psi_K(x,a))}{\sum_{b \in \mathcal{A}} \exp(\eta \Psi_K(x,b))}$;

end

return π_K;

3.2 Performance Guarantee

In this subsection, we investigate the finite-iteration and asymptotic behavior of Algorithm 1. We begin by proving a finite-iteration performance guarantee for DPP:

Theorem 2 (The ℓ_∞-norm performance loss bound of DPP) *Let Assumption 1 hold. Also, assume that Ψ_0 is uniformly bounded by $V_{\max}$ for all $(x,a) \in \mathcal{Z}$, then the following inequality holds for the policy induced by DPP at iteration $k \geq 0$:*

$$\|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma \left(4V_{\max} + \frac{\log(L)}{\eta}\right)}{(1-\gamma)^2 (k+1)}.$$

Proof See Appendix B.1. ■

Note that the DPP algorithm converges to the optimal policy for every $\eta > 0$ and choosing a different η only changes the rate of convergence. The best rate of convergence is achieved by setting $\eta = \infty$, for which the softmax policy and the softmax operator $\mathcal{M}_\eta$ are replaced with the greedy policy and the max-operator $\mathcal{M}$, respectively. Therefore, for $\eta = +\infty$ the DPP recursion is re-expressed as

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) - (\mathcal{M}\Psi_k)(x) + r(x,a) + \gamma(P\mathcal{M}\Psi_k)(x,a).$$

As an immediate consequence of Theorem 2, we obtain the following result:

Corollary 3 *The following relation holds in limit:*

$$\lim_{k\to+\infty} Q^{\pi_k}(x,a) = Q^*(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

In words, the policy induced by DPP asymptotically converges to the optimal policy π^*. The following corollary shows that there exists a unique limit for the action preferences in infinity if the optimal policy π^* is unique.

Corollary 4 *Let Assumption 1 hold and k be a non-negative integer. Assume that the optimal policy π^* is unique and let $\Psi_k(x,a)$, for all $(x,a) \in \mathcal{Z}$, be the action preference after k iteration of DPP. Then, we have:*

$$\lim_{k\to+\infty} \Psi_k(x,a) = \begin{cases} V^*(x) & a = a^*(x) \\ -\infty & \text{otherwise} \end{cases}, \qquad \forall x \in \mathcal{X}.$$

Proof See Appendix B.2. ■

Notice that the assumption on the *uniqueness* of the optimal policy π^* is not required for the main result of this section (Theorem 2). Also, the fact that in Corollary 4 the action preferences of sub-optimal actions tend to $-\infty$ is the natural consequence of the convergence of π_k to the optimal policy π^*, which forces the probability of the sub-optimal actions to be 0.

4. Dynamic Policy Programming with Approximation

Algorithm 1 (DPP) only applies to small problems with a few states and actions. Also, to compute the optimal policy by DPP an explicit knowledge of model is required. In many real world problems, this information is not available. Instead it may be possible to simulate the state transition by Monte-Carlo sampling and then *estimate* the optimal policy using these samples. In this section, we first prove some general bounds on the performance of DPP in the presence of approximation/estimation error and compare these bounds with those of AVI and API. We then present new approximate algorithms for implementing DPP with Monte-Carlo sampling (DPP-RL) and linear function approximation (SADPP). For both DPP-RL and SADPP we assume that we have access to the generative model of MDP, i.e., an oracle can generate the next sample y from $P(\cdot|x,a)$ for every state-action pair $(x,a) \in \mathcal{Z}$ on the request of the learner.

4.1 The ℓ_∞-Norm Performance-Loss Bounds for Approximate DPP

Let us consider a sequence of action preferences $\{\Psi_0, \Psi_1, \Psi_2, \dots\}$ such that, at round k, the action preferences Ψ_{k+1} is the result of approximately applying the DPP operator by the means of function approximation or Monte-Carlo simulation, i.e., for all $(x,a) \in \mathcal{Z}$: $\Psi_{k+1}(x,a) \approx \mathcal{O}\Psi_k(x,a)$. The error ϵ_k is defined as the difference of $\mathcal{O}\Psi_k$ and its approximation:

$$\epsilon_k(x,a) \triangleq \Psi_{k+1}(x,a) - \mathcal{O}\Psi_k(x,a), \qquad \forall (x,a) \in \mathcal{Z}. \tag{7}$$

Note that this definition of ϵ_k is rather general and does not specify the approximation technique used to compute Ψ_{k+1}. In the following subsections, we provide specific update rules to approximate Ψ_{k+1} for both DPP-RL and SADPP algorithms which also makes the definition of ϵ_k more specific.

The approximate DPP update rule then takes the following forms:

$$\begin{aligned}\Psi_{k+1}(x,a) &= \mathcal{O}\Psi_k(x,a) + \epsilon_k(x,a) \\ &= \Psi_k(x,a) + r(x,a) + \gamma P \mathcal{M}_\eta \Psi_k(x,a) - \mathcal{M}_\eta \Psi_k(x,a) + \epsilon_k(x,a) \\ &= \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k \Psi_k(x,a) + \epsilon_k(x,a),\end{aligned} \tag{8}$$

where π_k is given by (5).

We begin by the finite-iteration analysis of approximate DPP. The following theorem establishes an upper-bound on the performance loss of DPP in the presence of approximation error. The proof is based on generalization of the bound that we established for DPP by taking into account the error ϵ_k:

Theorem 5 (Finite-iteration performance loss bound of approximate DPP) *Let Assumption 1 hold. Assume that k is a non-negative integer and Ψ_0 is bounded by $V_{\max}$. Further, define ϵ_k for all k by (7) and the accumulated error E_k as*

$$E_k(x,a) \triangleq \sum_{j=0}^{k} \epsilon_j(x,a), \qquad \forall (x,a) \in \mathcal{Z}. \tag{9}$$

Then the following inequality holds for the policy induced by approximate DPP at round k:

$$\|Q^* - Q^{\pi_k}\| \le \frac{1}{(1-\gamma)(k+1)} \left[\frac{2\gamma \left(4V_{\max} + \frac{\log(L)}{\eta} \right)}{(1-\gamma)} + \sum_{j=0}^{k} \gamma^{k-j} \|E_j\| \right].$$

Proof See Appendix C. ■

Taking the upper-limit yields corollary 6.

Corollary 6 (Asymptotic performance-loss bound of approximate DPP) *Define $\bar{\varepsilon} \triangleq \limsup_{k\to\infty} \|E_k\| / (k+1)$. Then, the following inequality holds:*

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \le \frac{2\gamma}{(1-\gamma)^2} \bar{\varepsilon}. \tag{10}$$

The asymptotic bound is similar to the existing results of AVI and API (Thiery and Scherrer, 2010; Bertsekas and Tsitsiklis, 1996, chap. 6):

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}\varepsilon_{\max},$$

where $\varepsilon_{\max} = \limsup_{k\to\infty} \|\epsilon_k\|$. The difference is that in (10) the supremum norm of error $\varepsilon_{\max}$ is replaced by the supremum norm of the average error $\bar{\varepsilon}$. In other words, unlike AVI and API, the size of error at each iteration is not a critical factor for the performance of DPP and as long as the size of average error remains close to 0, DPP is guaranteed to achieve a near-optimal performance even when the individual errors ϵ_k are large

As an *example*: Consider a case in which, for both DPP and AVI/API, the sequence of errors $\{\epsilon_0, \epsilon_1, \epsilon_2, \dots\}$ are some i.i.d. zero-mean random variables bounded by $0 < U < \infty$. Corollary 6 combined with the law of large numbers then leads to the following asymptotic bound for approximate DPP:

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}\bar{\varepsilon} = 0, \qquad \text{w.p. (with probability) } 1, \tag{11}$$

whilst for API and AVI we have:

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}U.$$

In words, approximate DPP manages to cancel i.i.d. noise and asymptotically converges to the optimal policy whereas there is no guarantee, in this case, for the convergence of API and AVI to the optimal solution. This example suggests that DPP, in general, may average out some of the simulation noise caused by Monte-Carlo sampling and eventually achieve a better performance than AVI and API in the presence of sampling error.

Remark 7 *The i.i.d. assumption may be replaced by some weaker and more realistic assumption that only only requires the error sequence* $\{\epsilon_0, \epsilon_1, \dots, \epsilon_k\}$ *to be a sequence of martingale differences, i.e., the errors do not need to be independent as long as the expected value of* ϵ_k*, conditioned on the past observations, is* 0*. We prove, in the next subsection, that DPP-RL satisfies this assumption and, therefore, asymptotically converges to the optimal policy (see Theorem 9).*

4.2 Reinforcement Learning with Dynamic Policy Programming

To compute the optimal policy by DPP one needs an explicit knowledge of model. In many problems, we do not have access to this information but instead we can generate samples by simulating the model. The optimal policy can then be *learned* using these samples. In this section, we introduce a new RL algorithm, called DPP-RL, which relies on a sampling-based variant of DPP to update the policy. The update rule of DPP-RL is very similar to (6). The only difference is that we replace the Bellman operator $\mathcal{T}^\pi\Psi(x,a)$ with its sample estimate $\mathcal{T}_k^\pi\Psi(x,a) \triangleq r(x,a) + \gamma(\pi\Psi)(y_k)$, where the next sample y_k is drawn from $P(\cdot|x,a)$:

$$\Psi_{k+1}(x,a) \triangleq \Psi_k(x,a) + \mathcal{T}_k^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}. \tag{12}$$

Based on (12), we estimate the optimal policy by iterating some initial Ψ_0 through the DPP-RL update rule, where at each iteration we draw y_k for every $(x,a) \in \mathcal{Z}$. From Equation (7), the estimation error of the k^{th} iterate of DPP-RL is then defined as the difference between the Bellman operator $\mathcal{T}^{\pi_k}\Psi_k(x,a)$ and its sample estimate:

$$\epsilon_k(x,a) = \mathcal{T}_k^{\pi_k}\Psi_k(x,a) - \mathcal{T}^{\pi_k}\Psi_k(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

The DPP-RL update rule can then be considered as a special case of the more general approximate DPP update rule of Equation (8).

Equation (12) is just an approximation of the DPP update rule (6). Therefore, the convergence result of Corollary 3 does not hold for DPP-RL. However, the new algorithm still converges to the optimal policy since one can show that the errors associated with approximating (6) are asymptotically *averaged out* by DPP-RL, as postulated by Corollary 6. To prove this result we need the following lemma, which bounds the estimation error ϵ_k.

Lemma 8 (Boundedness of ϵ_k) *Let Assumption 1 hold and assume that the initial action-preference function Ψ_0 is uniformly bounded by $V_{\max}$, then we have, for all $k \geq 0$,*

$$\left\|\mathcal{T}_k^{\pi_k}\Psi_k\right\| \leq \frac{2\gamma \log L}{\eta(1-\gamma)} + V_{\max}, \qquad \|\epsilon_k\| \leq \frac{4\gamma \log L}{\eta(1-\gamma)} + 2V_{\max}.$$

Proof See Appendix D. ■

Lemma 8 is an interesting result, which shows that, despite the fact that Ψ_k tends to $-\infty$ for the sub-optimal actions, the error ϵ_k is uniformly bounded by some finite constant. Note that $\epsilon_k = \mathcal{T}^{\pi_k}\Psi_k - \mathcal{T}_k^{\pi_k}\Psi_k$ can be expressed in terms of the soft-max $\mathcal{M}_\eta\Psi_k$, which unlike Ψ_k, is always bounded by a finite constant, for every $\eta > 0$.

The following theorem establishes the asymptotic convergence of DPP-RL to the optimal policy.

Theorem 9 (Asymptotic convergence of DPP-RL) *Let Assumption 1 hold. Assume that the initial action-value function Ψ_0 is uniformly bounded by $V_{\max}$ and π_k is the policy induced by Ψ_k after k iteration of DPP-RL. Then, w.p. 1, the following holds:*

$$\lim_{k\to\infty} Q^{\pi_k}(x,a) = Q^*(x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

Proof See Appendix D.1. ■

We also prove the following result on the converge rate of DPP-RL to the optimal policy by making use of the result of Theorem 5:

Theorem 10 (Finite-time high-probability loss-bound of DPP-RL) *Let Assumption 1 hold and k be a positive integer and $0 < \delta < 1$. Then, at iteration k of DPP-RL with probability at least $1-\delta$, we have*

$$\|Q^* - Q^{\pi_k}\| \leq \frac{4(\gamma \log(L)/\eta + 2R_{\max})}{(1-\gamma)^3}\left[\frac{1}{k+1} + \sqrt{\frac{2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{k+1}}\right].$$

Proof See Appendix D.2. ■

Theorem 5 implies that, regardless of the value of η and γ, DPP-RL always converges with the rate of $1/\sqrt{k}$.

We can optimize the bound of Theorem 10 w.r.t. η which leads to the following corollary:

Corollary 11 *Let Assumption 1 hold and k be a positive integer, also set the inverse temperature $\eta = +\infty$, Then, at iteration k of DPP-RL with probability at least $1-\delta$, we have*

$$\|Q^* - Q^{\pi_k}\| \leq \frac{8R_{\max}}{(1-\gamma)^3}\left[\frac{1}{k+1} + \sqrt{\frac{2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{k+1}}\right].$$

This result implies that, in order to achieve the best rate of convergence, one can set the value of η to $+\infty$, i.e., to replace the soft-max $\mathcal{M}_\eta$ with the max operator $\mathcal{M}$:

$$\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}_k\Psi_k(x,a) - \mathcal{M}\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}, \tag{13}$$

where $\mathcal{T}_k\Psi(x,a) \triangleq r(x,a) + \gamma(\mathcal{M}\Psi)(y_k)$ for all $(x,a) \in \mathcal{Z}$. The pseudo-code of DPP-RL algorithm, which sets $\eta = +\infty$, is shown in Algorithm 2.

Algorithm 2: (DPP-RL) Reinforcement learning with DPP

Input: Initial action preferences $\Psi_0(\cdot,\cdot)$, discount factor γ and number of steps T

for $k = 1, 2, 3, \ldots, K-1$ **do** `// main loop`
  foreach $(x,a) \in \mathcal{Z}$ **do** `// update` $\Psi_k(\cdot,\cdot)$ `for all state-action pairs`
    $y_k \sim P(\cdot|x,a)$; `// generate the next sample`
    $\mathcal{T}_k\Psi_k(x,a) := r(x,a) + \gamma\mathcal{M}\Psi_k(y_k)$; `// empirical Bellman operator`
    $\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}_k\Psi_k(x,a) - \mathcal{M}\Psi_k(x)$; `// DPP update rule`
  end
  foreach $x \in \mathcal{X}$ **do** `// compute the control policy`
    $a_{\max} := \arg\max_{a\in\mathcal{A}} \Psi_{k+1}(x,a)$;
    $\pi(\cdot|x) := 0$;
    $\pi_{k+1}(a_{\max}|x) := 1$;
  end
end
return π_K

Furthermore, the following PAC bound which determines the number of steps k required to achieve the error $\epsilon > 0$ in estimating the optimal policy, w.p. $1-\delta$, is an immediate consequence of Theorem 10.

Corollary 12 (Finite-time PAC bound of DPP-RL) *Let Assumption 1 hold. Then, for any $\epsilon > 0$, after*

$$k = \frac{256R_{\max}^2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{(1-\gamma)^6\epsilon^2}.$$

steps of Algorithm 2, the uniform approximation error $\|Q^* - Q^{\pi_k}\| \leq \epsilon$, w. p. $1-\delta$.

4.3 Approximate Dynamic Policy Programming with Linear Function Approximation

In this subsection, we consider DPP with *linear function approximation* (LFA) and *least-squares regression*. LFA is commonly used in many RL algorithms (Szepesvári, 2010, sec. 3.2). Given a set of basis functions $\mathcal{F}_\phi = \{\phi_1, \ldots, \phi_m\}$, where each $\phi_i : \mathcal{Z} \to \mathbb{R}$ is a bounded real valued function, the sequence of action preferences $\{\Psi_0, \Psi_1, \Psi_2 \cdots\}$ are defined as a linear combination of these basis functions: $\Psi_k = \theta_k^\mathsf{T}\Phi$, where Φ is a $m \times 1$ column vector with the entries $\{\phi_i\}_{i=1:m}$ and $\theta_k \in \mathbb{R}^m$ is a $m \times 1$ vector of parameters.

The action preference function Ψ_{k+1} is an approximation of the DPP operator $\mathcal{O}\Psi_k$. In the case of LFA the common approach to approximate DPP operator is to find a vector θ_{k+1} that projects $\mathcal{O}\Psi_k$ on the column space spanned by Φ by minimizing the loss function:

$$J_k(\theta; \Psi) \triangleq \left\| \theta^\mathsf{T}\Phi - \mathcal{O}\Psi_k \right\|_{2,\mu}^2, \tag{14}$$

where μ is a probability measure on $\mathcal{Z}$. The best solution, that minimize J, is called the least-squares solution:

$$\theta_{k+1} = \arg\min_{\theta \in \mathbb{R}^m} J_k(\theta; \Psi) = \left[\mathbb{E}\left(\Phi\Phi^\mathsf{T}\right)\right]^{-1}\mathbb{E}(\Phi\mathcal{O}\Psi_k), \tag{15}$$

where the expectation is taken w.r.t. $(x, a) \sim \mu$. In principle, to compute the least squares solution equation one needs to compute $\mathcal{O}\Psi_k$ for all states and actions. For large scale problems this becomes infeasible. Instead, one can make a sample estimate of the least-squares solution by minimizing the empirical loss $\tilde{J}_k(\theta; \Psi)$ (Bertsekas, 2007, chap. 6.3):

$$\tilde{J}_k(\theta; \Psi) \triangleq \frac{1}{N}\sum_{n=1}^{N}(\theta^\mathsf{T}\Phi(X_n, A_n) - \mathcal{O}_n\Psi_k)^2 + \alpha\theta^\mathsf{T}\theta,$$

where $\{(X_n, A_n)\}_{n=1:N}$ is a set of N i.i.d. samples drawn from the distribution μ. Also, $\mathcal{O}_n\Psi_k$ denotes a single sample estimate of $\mathcal{O}\Psi_k(X_n, A_n)$ defined by $\mathcal{O}_n\Psi_k \triangleq \Psi_k(X_n, A_n) + r(X_n, A_n) + \gamma\mathcal{M}_\eta\Psi_k(Y_n) - \mathcal{M}_\eta\Psi_k(X_n)$, where $Y_n \sim P(\cdot|X_n, A_n)$. Further, to avoid over-fitting due to the small number of samples, one adds a quadratic regularization term to the loss function. The empirical least-squares solution which minimizes $\tilde{J}_k(\theta; \Psi)$ is given by

$$\tilde{\theta}_{k+1} = \left[\sum_{n=1}^{N}\Phi(X_n, A_n)\Phi(X_n, A_n)^\mathsf{T} + \alpha N\mathbf{I}\right]^{-1}\sum_{n=1}^{N}\mathcal{O}_n\Psi_k\Phi(X_n, A_n), \tag{16}$$

and $\Psi_k(x, a) = \tilde{\theta}_{k+1}\Phi(x, a)$. This defines a sequence of action preferences $\{\Psi_0, \Psi_1, \Psi_2, \cdots\}$ and the sequence of approximation error through Equation (7).

Algorithm 3 presents the *sampling-based approximate dynamic policy programming* (SADPP) in which we rely on (16) to approximate DPP operator at each iteration.

Algorithm 3: (SADPP) Sampling-based approximate dynamic policy programming

Input: $\tilde{\theta}_0$, η, γ, α, K and N

for $k = 0, 1, 2, \ldots, K-1$ **do** `// main loop`

  $\{(X_n, A_n)\}_{n=1:N} \sim \mu(\cdot, \cdot)$; `// generate` n `i.i.d. samples from` $\mu(\cdot, \cdot)$

  $\{Y_n\}_{n=1:N} \sim P(\cdot|\{(X_n, A_n)\}_{n=1:N})$; `// generate next states from` $P(\cdot|\cdot)$

  foreach $n = 1, 2, 3, \ldots, N$ **do**

    foreach $a \in \mathcal{A}$ **do** `// compute` Ψ_k `for every action of states` X_n, Y_n

      $\Psi_k(X_n, a) = \tilde{\theta}_k^\mathsf{T} \Phi(X_n, a)$;

      $\Psi_k(Y_n, a) = \tilde{\theta}_k^\mathsf{T} \Phi(Y_n, a)$;

    end

    $\mathcal{M}_\eta \Psi_k(X_n) = \sum_{a \in \mathcal{A}} \frac{\exp(\eta \Psi_k(X_n, a)) \Psi_k(X_n, a)}{\sum_{b \in \mathcal{A}} \exp \eta \Psi_k(X_n, b)}$;

    $\mathcal{M}_\eta \Psi_k(Y_n) = \sum_{a \in \mathcal{A}} \frac{\exp(\eta \Psi_k(Y_n, a)) \Psi_k(Y_n, a)}{\sum_{b \in \mathcal{A}} \exp \eta \Psi_k(Y_n, b)}$; `// soft-max` $\mathcal{M}_\eta \Psi_k$ `for` X_n `and` Y_n

    `// empirical DPP operator`

    $\mathcal{O}_n \Psi_k = \Psi_k(X_n, A_n) - r(X_n, A_n) - \gamma (\mathcal{M}_\eta \Psi_k)(Y_n) + (\mathcal{M}_\eta \Psi_k)(X_n)$;

  end

  `// SADPP update rule`

  $\tilde{\theta}_{k+1} = \left[\sum_{n=1}^N \Phi(X_n, A_n) \Phi(X_n, A_n)^\mathsf{T} + \alpha N \mathbf{I} \right]^{-1} \sum_{n=1}^N \mathcal{O}_n \Psi_k \Phi(X_n, A_n)$;

end

return $\tilde{\theta}_K$

5. Numerical Results

In this section, we illustrate empirically the theoretical performance guarantee introduced in the previous sections for both variants of DPP: the exact case (DPP-RL) and the approximate case (SADPP). In addition, we compare with existing algorithms for which similar theoretical results have been derived.

We first examine the convergence properties of DPP-RL (Algorithm 2) on several discrete state-action problems with large state spaces. We compare it with a synchronous variant of Q-learning (Even-Dar and Mansour, 2003) (QL) and a model-based Q-value iteration (VI) (Kearns and Singh, 1999). Next, we investigate the finite-time performance of SADPP (Algorithm 3) in the presence of function approximation and a limited sampling budget per iteration. In this case, we compare SADPP with regularized least-squares fitted Q-iteration (RFQI) (Farahmand et al., 2008a) and regularized least-squares policy iteration (REG-LSPI) (Farahmand et al., 2008b), two algorithms that, like SADPP, control the complexity of the solution using regularization.[2]

5.1 DPP-RL

To illustrate the performance of DPP-RL, we consider the following MDPs:

2. The source code of all tested algorithms is available in: `http://www.mbfys.ru.nl/~mazar/Research_Top.html`.

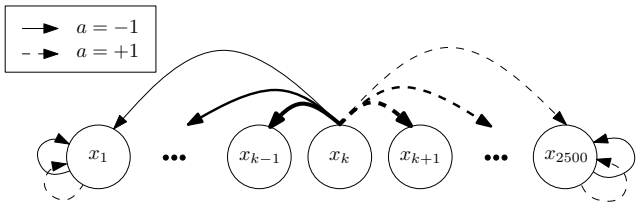

Figure 1: Linear MDP: Illustration of the linear MDP problem. Nodes indicate states. States x_1 and x_{2500} are the two absorbing states and state x_k is an example of interior state. Arrows indicate possible transitions of **these three nodes only**. From x_k any other node is reachable with transition probability (arrow thickness) proportional to the inverse of the distance to x_k (see the text for details).

Linear MDP: this problem consists of states $x_k \in \mathcal{X}, k = \{1, 2, \ldots, 2500\}$ arranged in a one-dimensional chain (see Figure 1). There are two possible actions $\mathcal{A} = \{-1, +1\}$ (left/right) and every state is accessible from any other state except for the two ends of the chain, which are absorbing states. A state $x_k \in \mathcal{X}$ is called absorbing if $P(x_k|x_k, a) = 1$ for all $a \in \mathcal{A}$ and $P(x_l|x_k, a) = 0, \forall l \neq k$. The state space is of size $|\mathcal{X}| = 2500$ and the joint action state space is of size $|\mathcal{Z}| = 5000$. Note that naive storing of the model requires $\mathcal{O}(10^7)$ memory.

Transition probability from an interior state x_k to any other state x_l is inversely proportional to the distance in the direction of the selected action. Formally, consider the following quantity $n(x_l, a, x_k)$ assigned to all non-absorbing states x_k and to every $(x_l, a) \in \mathcal{Z}$:

$$n(x_l, a, x_k) = \begin{cases} \dfrac{1}{|l-k|} & \text{for } (l-k)a > 0 \\ 0 & \text{otherwise} \end{cases}.$$

We can write the transition probabilities as:

$$P(x_l|x_k, a) = \frac{n(x_l, a, x_k)}{\sum\limits_{x_m \in \mathcal{X}} n(x_m, a, x_k)}.$$

Transitions to an absorbing state have associated reward 1 and to any interior state has associated reward -1.

The optimal policy corresponding to this problem is to reach the closest absorbing state as soon as possible.

Combination lock: the combination lock problem considered here is a stochastic variant of the reset state space models introduced in Koenig and Simmons (1993), where more than one reset state is possible (see Figure 2).

In our case we consider, as before, a set of states $x_k \in \mathcal{X}, k \in \{1, 2, \ldots, 2500\}$ arranged in a one-dimensional chain and two possible actions $\mathcal{A} = \{-1, +1\}$. In this

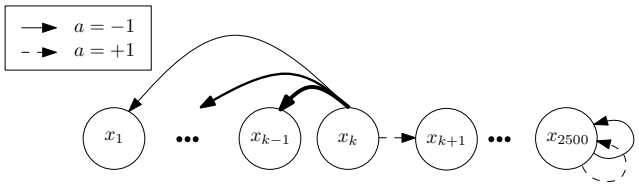

Figure 2: Combination lock: illustration of the combination lock MDP problem. Nodes indicate states. State x_{2500} is the goal (absorbing) state and state x_k is an example of interior state. Arrows indicate possible transitions of **these two nodes only**. From x_k any previous state is reachable with transition probability (arrow thickness) proportional to the inverse of the distance to x_k. Among the future states only x_{k+1} is reachable (arrow dashed).

problem, however, there is only one absorbing state (corresponding to the state *lock-opened*) with associated reward of 1. This state is reached if the all-ones sequence $\{+1, +1, \ldots, +1\}$ is entered correctly. Otherwise, if at some state x_k, $k < 2500$, action -1 is taken, the lock automatically resets to some previous state x_l, $l < k$ randomly (in the original problem, the reset state is always the initial state x_1).

For every intermediate state, the rewards of actions -1 and $+1$ are set to 0 and -0.01, respectively. The transition probability upon taking the wrong action -1 from state x_k to state x_l is $P(x_l|x_k, -1)$, as before, inversely proportional to the distance of the states. That is

$$n(x_k, x_l) = \begin{cases} \dfrac{1}{k-l} & \text{for } l < k \\ 0 & \text{otherwise} \end{cases}, \qquad P(x_l|x_k, -1) = \frac{n(x_k, x_l)}{\sum\limits_{x_m \in \mathcal{X}} n(x_k, x_m)}.$$

Note that this problem is more difficult than the linear MDP since the goal state is only reachable from one state, x_{2499}.

Grid world: this MDP consists of a grid of 50×50 states. A set of four actions {RIGHT, UP, DOWN, LEFT} is assigned to every state $x \in \mathcal{X}$. Although the state space of the grid world is of the same size than the previous two problems, $|\mathcal{X}| = 2500$, the joint action state space is larger, $|\mathcal{Z}| = 10^4$.

The location of each state x of the grid is determined by the coordinates $c_x = (h_x, v_x)$, where h_x and v_x are some integers between 1 and 50. There are 196 absorbing *wall states* surrounding the grid and another one at the center of grid, for which a reward -1 is assigned. The reward for the walls is

$$r(x, a) = -\frac{1}{\|c_x\|_2}, \qquad \forall a \in \mathcal{A}.$$

Also, we assign reward 0 to all of the remaining (non-absorbing) states.

This means that both the top-left absorbing state and the central state have the least possible reward (-1), and that the remaining absorbing states have reward which

increases proportionally to the distance to the state in the bottom-right corner (but are always negative).

The transition probabilities are defined in the following way: taking action a from any non-absorbing state x results in a one-step transition in the direction of action a with probability 0.6, and a random move to a state $y \neq x$ with probability inversely proportional to their Euclidean distance $1/\left\|c_x - c_y\right\|_2$.

This problem is interesting because of the presence of the absorbing walls, which prevent the agent to escape and because of the high level of noise: from a non-absorbing state, many states are reachable with significant probability.

The resulting optimal policy is to *survive* in the grid as long as possible by avoiding both the absorbing walls and the center of the grid. Note that because of the difference between the cost of walls, the optimal control prefers the states near the bottom-right corner of the grid, thus avoiding absorbing states with higher cost.

5.1.1 Experimental Setup and Results

For consistency with the theoretical results, we evaluate the performance of all algorithms in terms of ℓ_∞-norm error of the action-value function $\|Q^* - Q^{\pi_k}\|$ obtained by policy π_k induced at iteration k. The discount factor γ is fixed to 0.995 and the optimal action-value function Q^* is computed with high accuracy through value iteration.

We compare DPP-RL with two other algorithms:

Q-learning (QL) : we consider a synchronous variant of Q-learning for which convergence results have been derived in Even-Dar and Mansour (2003). Since QL is sensitive to the learning step, we consider QL with polynomial learning step $\alpha_k = 1/(k+1)^\omega$ where $\omega \in \{0.51, 0.75, 1.0\}$. It is known that ω needs to be larger than 0.5, otherwise QL can asymptotically diverge (see Even-Dar and Mansour, 2003, for the proof).

Model-based Q-value iteration (VI) : The VI algorithm (Kearns and Singh, 1999) first estimates a model using all the data samples and then performs value iteration on the learned model. Therefore, unlike QL and DPP, VI is a model-based algorithm and requires to store the model.

Comparison between VI and both DPP-RL and QL is specially problematic: first, the number of computations per iteration is different. Whereas DPP-RL and QL require $|\mathcal{Z}|$ computations per iteration, VI requires $|\mathcal{Z}||\mathcal{X}|$. Second, VI requires to estimate the model initially (using a given number of samples) and then iterates until convergence. This latter aspect is also different from DPP-RL and QL, which use one sample per iteration. Therefore, the number of samples determines the number of iterations for DPP-RL and QL, but not for VI. What is determined for VI is the number of samples dedicated to estimate the model.

For consistency with the theoretical results, we use as error measure, the distance between the optimal action-value function and the value function of the policy induced by the algorithms. instead of the more popular average accumulated reward, which is usually used when the RL algorithm learns from a stream of samples.

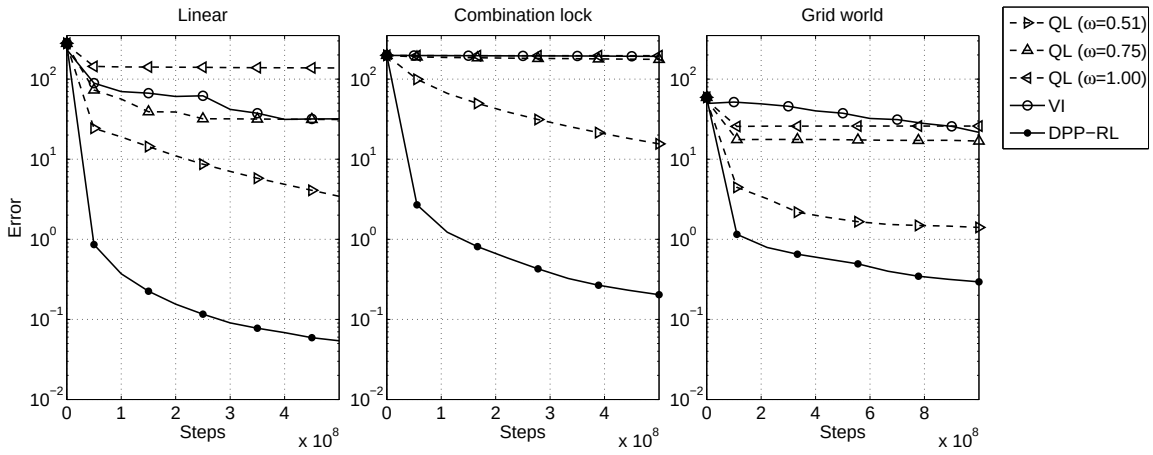

Figure 3: Comparison between DPP-RL, QL and VI in terms of **number of steps**, defined as the number of iterations times the number of computations per iteration of the particular algorithm. Each plot shows the averaged error of the induced policies over 50 different runs (see the text for details).

Simulations are performed using the following procedure: at the beginning of each run **(i)** the action-value function and the action preferences are randomly initialized in the interval $[-V_{\max}, V_{\max}]$, and **(ii)** a set of 10^5 samples is generated from $P(\cdot|x,a)$ for all $(x,a) \in \mathcal{Z}$. As mentioned before, this fixes the maximum number of iterations for DPP-RL and QL to 10^5, but not for VI. We run VI until convergence. We repeat this procedure 50 times and compute the average error in the end. Using significantly less number of samples leads to a dramatic decrease of the quality of the solutions using all approaches and no qualitative differences in the comparison.

To compare the methods using equivalent logical units independently of the particular implementation, we rescale their number of iterations by the number of steps required in one iteration. For the case of VI, the *step* units are the number of iterations times $|\mathcal{Z}||\mathcal{X}|$ and for DPP-RL and QL, the number of iterations times $|\mathcal{Z}|$.

Figure 3 shows the error as a function of the number of steps. First, in agreement with the theoretical results, we observe that the DPP-error decays very fast in the beginning and keeps decreasing at a smaller rate afterwards. We also observe that DPP-RL performs significantly better than QL. The improvement is about two orders of magnitude in both the linear MDP and the combination lock problems and more than four times better in the Grid world. QL shows the best performance for $\omega = 0.51$ and the quality degrades as a function of ω.

Although the performance of VI looks poor for the number of steps shown in Figure 3, we observe that VI reaches an average error of 0.019 after convergence ($\approx 2 \cdot 10^{10}$ steps) for the linear MDP and the combination lock and an error of 0.10 after $\approx 4 \cdot 10^{10}$ steps for the grid problem. This means for a fixed number of samples, the asymptotic solution of VI is better than the one of DPP-RL, at the cost of much larger number of steps.

To illustrate the performance of the methods using a limited CPU time budget, we also compare the average and standard deviations of the errors in terms of elapsed CPU time by running the algorithms until a maximum allowed time is reached. We choose 30 seconds in the case of linear MDP and combination lock and 60 seconds for the grid world, which has twice as many actions as the other benchmarks. To minimize the implementation dependent variability, we coded all three algorithms in C++ and ran them on the same processor. CPU time was acquired using the system function `times()` which provides process-specific CPU time. Sampling time was identical for all methods and not included in the analysis.

Table 1 shows the final average errors (standard deviations between parenthesis) in the CPU time comparison. As before, we observe that DPP-RL converges very fast achieving near optimal performance after a few seconds. The small variance of estimation of DPP-RL suggests that, as derived in Theorems 9 and 5, DPP-RL manages to average out the simulation noise caused by sampling and converges to a near optimal solution, which is very robust.

Table 1: Comparison between DPP-RL, QL and VI given a fixed computational and sampling budget. Table 1 shows error means and standard deviations (between parenthesis) at the end of the simulations for three different algorithms (columns) and three different benchmarks (rows).

Benchmark		Linear MDP		Combination lock		Grid world	
Run Time		30 sec.		30 sec.		60 sec.	
DPP-RL		**0.05**	(**0.02**)	**0.20**	(**0.09**)	**0.32**	(**0.03**)
VI		16.60	(11.60)	69.33	(15.38)	5.67	(1.73)
QL	$\omega = 0.51$	4.08	(3.21)	18.18	(4.36)	1.46	(0.12)
	$\omega = 0.75$	31.41	(12.77)	176.13	(25.68)	17.21	(7.31)
	$\omega = 1.00$	138.01	(146.28)	195.74	(5.73)	25.92	(20.13)

Overall, these results complement the theory presented in previous sections. We can conclude that for the chosen benchmarks DPP-RL converges significantly faster than VI and QL. However, for a fixed number of samples, VI obtains a better solution than DPP-RL requiring significantly more computation.

5.2 SADPP

In this subsection, we illustrate the performance of the SADPP algorithm in the presence of function approximation and limited sampling budget per iteration. The purpose of this subsection is to analyze numerically the sample complexity, that is, the number of samples required to achieve a near optimal performance with low variance.

We compare SADPP with ℓ_2-regularized versions of the following two algorithms:

Regularized fitted Q-iteration (RFQI) (Farahmand et al., 2008a). RFQI performs value iteration to approximate the optimal action value function. See also Antos et al. (2008) and Ernst et al. (2005).

Regularized Least Squares Policy Iteration (REG-LSPI) (Farahmand et al., 2008b). It can be regarded as a Monte-Carlo sampling implementation of approximate value iteration (AVI) with action-state representation (see also Lagoudakis and Parr, 2003).

The benchmark we consider is a variant of the *optimal replacement problem* presented in Munos and Szepesvári (2008).

5.2.1 Optimal replacement problem

This problem is an infinite-horizon, discounted MDP. The state measures the accumulated use of a certain product and is represented as a continuous, one-dimensional variable. At each time-step t, either the product is kept $a(t) = 0$ or replaced $a(t) = 1$. Whenever the product is replaced by a new one, the state variable is reset to zero $x(t) = 0$, at an additional cost C. The new state is chosen according to an exponential distribution, with possible values starting from zero or from the current state value, depending on the latest action:

$$p(y|x, a = 0) = \begin{cases} \beta e^{\beta(y-x)} & \text{if } y \geq x \\ 0 & \text{if } y < 0 \end{cases} \qquad p(y|x, a = 1) = \begin{cases} \beta e^{\beta} y & \text{if } y \geq 0 \\ 0 & \text{if } y < 0 \end{cases}.$$

The reward function is a monotonically decreasing function of the state x if the product is kept $r(x, 0) = -c(x)$ and constant if the product is replaced $r(x, 1) = -C - c(0)$, where $c(x) = 4x$.

The optimal action is to keep as long as the accumulated use is below a threshold or to replace otherwise:

$$a^*(x) = \begin{cases} 0 & \text{if } x \in [0, \bar{x}] \\ 1 & \text{if } x > \bar{x} \end{cases}. \tag{17}$$

Following Munos and Szepesvári (2008), $\bar{x}$ can be obtained exactly via the Bellman equation and is the unique solution to

$$C = \int_0^{\bar{x}} \frac{c'(y)}{1-\gamma} \left(1 - \gamma e^{-\beta(1-\gamma)y}\right) dy.$$

5.2.2 Experimental setup and results

For all algorithms we map the state-action space using twenty radial basis functions (ten for the continuous one-dimensional state variable x, spanning the state space $\mathscr{X}$, and two for the two possible actions). Other parameter values where chosen to be the same as in Munos and Szepesvári (2008), that is, $\gamma = 0.6, \beta = 0.5, C = 30$, which results in $\bar{x} \simeq 4.8665$. We also fix an upper bound for the states, $x_{\max} = 10$ and modify the problem definition such that if the next state y happens to be outside of the domain $[0, x_{\max}]$ then the product is replaced immediately, and a new state is drawn as if action $a = 1$ were chosen in the previous time step.

We measure the performance loss of the algorithms in terms of the difference between the optimal action a^* and the action selected by the algorithms. We use this performance measure since it is easy to compute as we know the analytical solution of the optimal

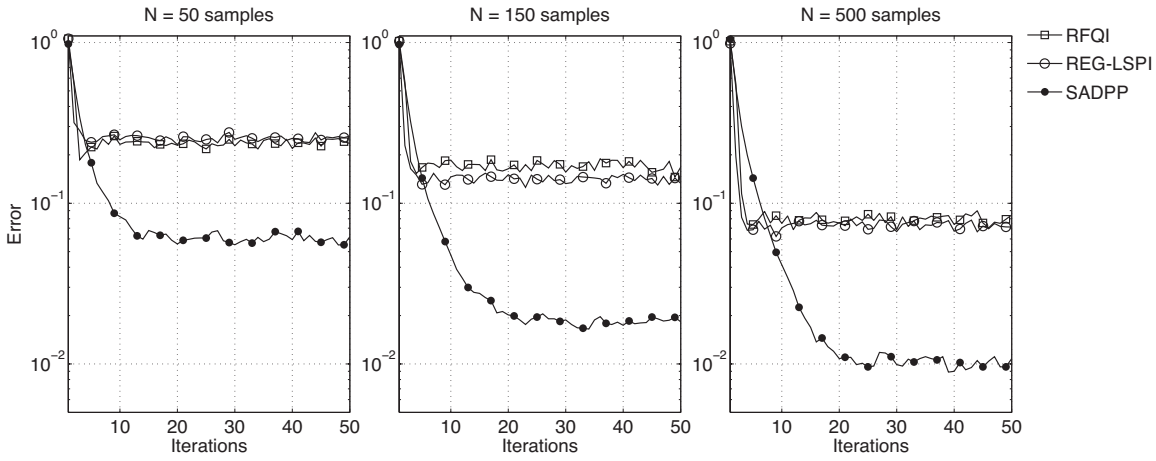

Figure 4: Numerical results for the optimal replacement problem. Each plot shows the error of RFQI, REG-LSPI and SADPP for certain number of samples N. Error is defined as in Equation (18) and averaged over 200 repetitions (see the text for details).

control in the optimal replacement problem (see Equation 17). We discretize the state space in $K = 100$ and compute the error as follows:

$$\text{Error} = \frac{1}{K} \sum_{k=1}^{K} |a^*(x_k) - \hat{a}(x_k)|, \tag{18}$$

where $\hat{a}$ is the action selected by the algorithm. Note that, unlike RFQI and REG-LSPI, SADPP induces a stochastic policy, that is, a distribution over actions. We select $\hat{a}$ for SADPP by choosing the most probable action from the induced soft-max policy, and then use this to compute (18). RFQI and REG-LSPI select the action with highest action-value function.

Simulations are performed using the same following procedure for all three algorithms: at the beginning of each run, the vector $\tilde{\theta}_0$ is initialized in the interval $[-1, 1]$. We then let the algorithm run for 10^3 iterations for 200 different runs. A new independent set of samples is generated at each iteration.

For each of the algorithms and each N, we optimize their parameters for the best asymptotic performance. Note that SADPP, in addition to the regularizer parameter α, has an extra degree of freedom η. Empirically, we observe that the optimal performance of SADPP is attained for finite η. This differs from DPP-RL, for which the convergence rate is optimized for $\eta = \infty$. This difference may be related to the observation that replacing the non-differentiable max-operator ($\eta = +\infty$) with a differentiable soft-max operator ($\eta < +\infty$) can improve the convergence behavior of the algorithm, as shown in Perkins and Precup (2002); de Farias and Roy (2000).

Table 2: Comparison between SADPP, RFQI and REG-LSPI for the optimal replacement problem. Table shows error means and standard deviations (between parenthesis) at the end of the simulations (after 10^3 iterations) for the three different algorithms (columns) and three different number of samples (rows).

Num. samples	50		150		500	
SADPP	**0.07**	(**0.06**)	**0.02**	(**0.01**)	**0.01**	(**0.01**)
RFQI	0.24	(0.19)	0.17	(0.12)	0.08	(0.07)
REG-LSPI	0.26	(0.16)	0.13	(0.10)	0.07	(0.06)

We are interested in the behavior of the error as a function of the iteration number for different number of samples N per iteration. Figure 4 and Table 2 show the performance results of the three different algorithms for $N \in \{50, 150, 500\}$ for the first 50 iterations and the total 10^3 iterations respectively. We observe that after an initial transient, all algorithms reach a nearly optimal solution after 50 iterations.

First, we note that SADPP asymptotically outperforms RFQI and REG-LSPI on average in all cases. Interestingly, there is no significant difference between the performance of RFQI and REG-LSPI. The performance of all algorithms improve for larger N. We emphasize that SADPP using only 50 samples shows comparable results to both RFQI and REG-LSPI using ten times more samples.

A comparison of the variances after the transient (see Table 2) shows that the sample complexity of SADPP is significantly smaller than RFQI and REG-LSPI. The variance of SADPP using again only 50 samples is comparable to the one provided by the other two methods using $N = 500$ samples.

Globally, we can conclude that SADPP has positive effects in reducing the effect of simulation noise, as postulated in Section 4. We can also conclude that, for our choice of settings, SADPP outperforms RFQI and REG-LSPI.

6. Related Work

In this section, we review some previous RL methods and compare them with DPP.

Policy-gradient actor-critic methods As we explained earlier in Section 1, actor-critic method is a popular incremental RL algorithm (Sutton and Barto, 1998; Barto et al., 1983, chap. 6.6), which makes use of a separate structure to store the value function (critic) and the control policy (actor). An important extension of AC, the *policy-gradient actor critic* (PGAC), extends the idea of AC to problems of practical scale (Sutton et al., 1999; Peters and Schaal, 2008). In PGAC, the actor updates the parameterized policy in the direction of the (natural) gradient of performance, provided by the critic. The gradient update ensures that PGAC asymptotically converges to a local maximum, given that an unbiased estimate of the gradient is provided by the critic (Maei et al., 2010; Bhatnagar et al., 2009; Konda and Tsitsiklis, 2003; Kakade, 2001). The parameter η in DPP is reminiscent of the learning step β in PGAC methods, since it influences the rate of change of the policy and in this sense

may play a similar role as the learning step β in PGAC (Konda and Tsitsiklis, 2003; Peters and Schaal, 2008). However, it is known that in the presence of sampling error, asymptotic convergence to a local maxima is only attained when β asymptotically decays to zero (Konda and Tsitsiklis, 2003; Baxter and Bartlett, 2001), whereas the parameter η in DPP, and DPP-RL, can be an arbitrary constant.

Q-learning DPP is not the only method which relies on an incremental update rule to control the sampling error. There are other incremental RL methods which aim to address the same problem (see, e.g., Maei et al., 2010; Singh et al., 2000; Watkins and Dayan, 1992).

One of the most well-known algorithms of this kind is Q-learning (QL) (Watkins and Dayan, 1992), which controls the sampling error by introducing a decaying learning step to the update rule of value iteration. QL has been shown to converge to the optimal value function in tabular case (Bertsekas and Tsitsiklis, 1996; Jaakkola et al., 1994). Also, there are some studies in the literature concerning the asymptotic convergence of Q-learning in the presence of function approximation (Melo et al., 2008; Szepesvári and Smart, 2004). However, the convergence rate of QL is very sensitive to the choice of learning step, and a bad choice of the learning step may lead to a slow rate of convergence (Even-Dar and Mansour, 2003). For instance, the convergence rate of QL with a linearly decaying learning step is of order $(1/k)^{1-\gamma}$, which makes the Q-learning algorithm extremely slow for $\gamma \approx 1$ (Szepesvári, 1997). This is in contrast to our previously mentioned result on the convergence of DPP-RL in Theorem 10 which guarantees that, regardless of the value of η and γ, DPP-RL always converges to the optimal policy with a rate of order $1/\sqrt{k}$. The numerical results of Subsection 5.1 confirms the superiority of DPP-RL to QL in terms of the rate of convergence.

One can also compare the finite-time behavior of DPP-RL and QL in terms of the PAC sample complexity of these methods. We have proven a sample-complexity PAC bound of order $O(1/(1-\gamma)^6)$ for DPP-RL in Subsection 4.2, whereas the best existing PAC bound for standard QL, to find an ϵ-optimal policy, is of order $O(1/(1-\gamma)^7)$ (Even-Dar and Mansour, 2003; Azar et al., 2011, sec. 3.3.1).[3] This theoretical result suggests that DPP-RL is superior to QL in terms of sample complexity of the estimating the optimal policy, especially, when γ is close to 1.

There is an on-policy version of Q-learning algorithm called SARSA (see, e.g., Singh et al., 2000) which also guarantees the asymptotic convergence to the optimal value function. However little is known about the rate of convergence and the finite-time behavior of this algorithm.

3. Note that Even-Dar and Mansour (2003) make use of a slightly different performance measure than the one we use in this paper: The optimized result of Even-Dar and Mansour (2003), which is of order $O(1/(1-\gamma)^5)$, is a bound on the sample complexity of estimating Q^* with ϵ precision, whereas in this paper we consider the sample complexity of finding an ϵ-optimal policy. However, the latter can be easily derived for QL from the inequality $\|Q^* - Q^{\pi_k}\| \leq 1/(1-\gamma)\|Q^* - Q_k\|$, where π_k is the greedy policy w.r.t. Q_k and Q_k is the estimate of action-value function at iteration k. This inequality combined with the result of Even-Dar and Mansour (2003) implies a sample complexity bound of order $O(1/(1-\gamma)^7)$ for QL.

Very recently, Azar et al. (2011) propose a new variant of Q-learning algorithm, called speedy Q-learning (SQL), which makes use of a different update rule than standard Q-learning of Watkins and Dayan (1992). Like DPP-RL, SQL convergences to the optimal policy with the rate of convergence of order $1/\sqrt{k}$. However, DPP-RL is superior to SQL in terms of memory space requirement, since SQL needs twice as much space as DPP-RL does.

Relative-entropy methods The DPP algorithm is originally motivated (see Appendix A) by the work of Kappen (2005) and Todorov (2006), who formulate a stochastic optimal control problem to find a conditional probability distribution $p(y|x)$ given an uncontrolled dynamics $\bar{p}(y|x)$. The control cost is the relative entropy between $p(y|x)$ and $\bar{p}(y|x)\exp(r(x))$. The difference is that in their work a restricted class of control problems is considered for which the optimal solution p can be computed directly in terms of $\bar{p}$ without requiring Bellman-like iterations. Instead, the present approach is more general, but does requires Bellman-like iterations. Likewise, our formalism is superficially similar to PoWER (Kober and Peters, 2008) and SAEM (Vlassis and Toussaint, 2009), which rely on EM algorithm to maximize a lower bound for the expected return in an iterative fashion. This lower-bound also can be written as a KL-divergence between two distributions. Also, the natural policy gradient method can be seen as a relative entropy method, in which the second-order Taylor expansion of the relative-entropy between the distribution of the states is considered as the metric for policy improvement (Bagnell and Schneider, 2003). Another relevant study is *relative entropy policy search* (REPS) (Daniel et al., 2012; Peters et al., 2010) which relies on the idea of minimizing the relative entropy to control the size of policy update. However there are some differences between REPs and DPP. **(i)** In REPS the inverse temperature η needs to be optimized while DPP converges to the optimal solution for any inverse temperature η, and **(ii)** here we provide a convergence analysis of DPP, while there is no convergence analysis in REPS.

7. Discussion and Future Work

We have presented a new approach, dynamic policy programming (DPP), to compute the optimal policy in infinite-horizon discounted-reward MDPs. We have theoretically proven the convergence of DPP to the optimal policy for the tabular case. We have also provided performance-loss bounds for DPP in the presence of approximation. The bounds have been expressed in terms of supremum norm of average accumulated error as opposed to the standard bounds which are expressed in terms of supremum norm of the errors. We have then introduced a new incremental RL algorithm, called DPP-RL, which relies on a sample estimate instance of the DPP update rule to estimate the optimal policy. We have proven that DPP-RL converges to the optimal policy with the rate of $1/\sqrt{k}$.

We have also compared numerically the finite-time behavior of DPP-RL with similar RL methods. Experimental results have shown a better performance of DPP-RL when compared to QL and VI in terms of convergence rate. In these problems, for equal number of samples, VI converged to a better solution than DPP-RL, at the cost of many more steps. When compared to VI, DPP-RL does not need to store the model dynamics, resulting in

significant less memory requirements for large-scale MDPs. This statement is general and holds when comparing DPP-RL to any model-based method.

We have proposed SADPP as a variant of DPP which makes use of linear function approximation and regularization. SADPP has been shown to perform better than two other regularized methods, RFQI and LSPI. We think that this is mainly due to the reduction of the effect of simulation noise (Section 4). At the same time, we admit that the existence of an additional parameter η favors SADPP. Therefore, it is interesting to consider soft-max variants of RFQI and LSPI which also make use of the inverse temperature η. In these cases, η should be initialized at a finite value and would gradually grow to $+\infty$.

The empirical comparison with those methods that do not make use of generative model assumption is outside of the scope of the current work and is left for future research. These methods include, for instance, PGAC methods that use sequences of samples to learn the value function of the current policy (Peters and Schaal, 2008; Konda and Tsitsiklis, 2003; Sutton et al., 1999), or upper-confidence bounds methods which address the exploration-exploitation dilemma (Jaksch et al., 2010; Szita and Szepesvári, 2010; Bartlett and Tewari, 2009; Strehl et al., 2009).

Another interesting line of future research is to devise finite-sample PAC bounds for SADPP in the spirit of previous theoretical results available for fitted value iteration and fitted Q-iteration (Munos and Szepesvári, 2008; Antos et al., 2008; Munos, 2005). This would require extending the error propagation result of Theorem 5 to an ℓ_2-norm analysis and combining it with the standard regression bounds.

Finally, an important extension of our results would be to apply DPP to large-scale action problems. This would require an efficient way to approximate $\mathcal{M}_\eta \Psi_k(x)$ in the update rule of Equation (6), since computing the exact summations becomes expensive. One idea is to sample estimate $\mathcal{M}_\eta \Psi_k(x)$ using Monte-Carlo simulation (MacKay, 2003, chap. 29), since $\mathcal{M}_\eta \Psi_k(x)$ is the expected value of $\Psi_k(x,a)$ under the soft-max policy π_k.

Appendix A. From Bellman Equation to DPP Recursion

In this appendix we give an informal *derivation* of the DPP equation. This is only for helping the reader to understand the origin of the DPP equation and it is in no way meant as a justification of DPP. The theoretical analysis and the proof of convergence of DPP is provided in Subsection 3.2.

Let $\bar{\pi}$ be a stochastic policy, i.e, $\bar{\pi}(a|x) > 0$ for all $(x,a) \in \mathcal{Z}$. Consider the relative entropy between the policy π and some baseline policy $\bar{\pi}$:

$$g^\pi_{\bar{\pi}}(x) \triangleq \mathrm{KL}\left(\pi(\cdot|x) \| \bar{\pi}(\cdot|x)\right) = \sum_{a \in \mathcal{A}} \pi(a|x) \log\left[\frac{\pi(a|x)}{\bar{\pi}(a|x)}\right], \qquad \forall x \in \mathcal{X}.$$

Note that $g^\pi_{\bar{\pi}}(x)$ is a positive function of x which is also bounded from above due to the assumption that $\bar{\pi}$ is a stochastic policy. We define a new value function $V^\pi_{\bar{\pi}}$, for all $x \in \mathcal{X}$, which incorporates g as a penalty term for deviating from the base policy $\bar{\pi}$ and the reward under the policy π:

$$V_{\bar{\pi}}^{\pi}(x) \triangleq \lim_{n\to\infty} \mathbb{E}\left[\sum_{k=0}^{n} \gamma^k \left(r_{t+k} - \frac{1}{\eta} g_{\bar{\pi}}^{\pi}(x_{t+k})\right) \middle| x_t = x\right],$$

where η is a positive constant and r_{t+k} is the reward at time $t+k$. Also, the expected value is taken w.r.t. the state transition probability distribution P and the policy π. The optimal value function $V_{\bar{\pi}}^{*}(x) \triangleq \sup_{\pi} V_{\bar{\pi}}^{\pi}(x)$ then exists and is bounded by some finite constant $c > 0$. Also, the value function $V_{\bar{\pi}}^{*}(x)$ satisfies the following Bellman equation for all $x \in \mathcal{X}$:

$$V_{\bar{\pi}}^{*}(x) = \sup_{\pi(\cdot|x)} \sum_{a\in\mathcal{A}} \pi(a|x) \left[r(x,a) - \frac{1}{\eta} \log \frac{\pi(a|x)}{\bar{\pi}(a|x)} + \gamma (P V_{\bar{\pi}}^{*})(x,a)\right]. \tag{19}$$

Equation (19) is a modified version of (2) where, in addition to maximizing the expected reward, the optimal policy $\bar{\pi}^*$ also minimizes the distance with the baseline policy $\bar{\pi}$. The maximization in (19) can be performed in closed form. Following Todorov (2006), we state Proposition 1 (closely related results to proposition 1 can be found in the recent works of Still and Precup, 2011; Peters et al., 2010):

Proposition 1 *Let η be a positive constant, then for all $x \in \mathcal{X}$ the optimal value function $V_{\bar{\pi}}^{*}(x)$ and for all $(x,a) \in \mathcal{Z}$ the optimal policy $\bar{\pi}^*(a|x)$, respectively, satisfy:*

$$V_{\bar{\pi}}^{*}(x) = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\big[\eta(r(x,a) + \gamma (P V_{\bar{\pi}}^{*})(x,a))\big], \tag{20}$$

$$\bar{\pi}^*(a|x) = \frac{\bar{\pi}(a|x) \exp\big[\eta(r(x,a) + \gamma (P V_{\bar{\pi}}^{*})(x,a))\big]}{\exp\left(\eta V_{\bar{\pi}}^{*}(x)\right)}. \tag{21}$$

Proof

We must optimize π subject to the constraints $\sum_{a\in\mathcal{A}} \pi(a|x) = 1$ and $0 < \pi(a|x) < 1$. We define the Lagrangian function $\mathcal{L}\left(x;\lambda_x\right) : \mathcal{X} \to \Re$ by adding the term $\lambda_x\big[\sum_{a\in\mathcal{A}} \pi(a|x) - 1\big]$ to the RHS of (19). Because $\bar{\pi}$ is strictly positive, minimizing $\mathcal{L}$ ensures that the solution is positive and the constraints $0 < \pi(a|x) \leq 1$ are automatically satisfied. Note that the KL-divergence is well-defined when both $\bar{\pi}$ and π are positive.

$$\mathcal{L}\left(x;\lambda_x\right) = \sum_{a\in\mathcal{A}} \pi(a|x) \left[r(x,a) + \gamma \left(P V_{\bar{\pi}}^{*}\right)(x,a)\right] - \frac{1}{\eta} \mathrm{KL}\left(\pi(\cdot|x) \| \bar{\pi}(\cdot|x)\right) - \lambda_x \left[\sum_{a\in\mathcal{A}} \pi(a|x) - 1\right].$$

The maximization in (19) can be expressed as maximizing the Lagrangian function $\mathcal{L}\left(x,\lambda_x\right)$. The necessary condition for the extremum with respect to $\pi(\cdot|x)$ is:

$$0 = \frac{\partial \mathcal{L}\left(x,\lambda_x\right)}{\partial \pi(a|x)} = r(x,a) + \gamma \left(P V_{\bar{\pi}}^{*}\right)(x,a) - \frac{1}{\eta} - \frac{1}{\eta} \log\left(\frac{\pi(a|x)}{\bar{\pi}(a|x)}\right) - \lambda_x,$$

which leads to:

$$\bar{\pi}^*(a|x) = \bar{\pi}(a|x) \exp\left(-\eta\lambda_x - 1\right) \exp\left[\eta(r(x,a) + \gamma \left(P V_{\bar{\pi}}^{*}\right)(x,a))\right], \qquad \forall x \in \mathcal{X}. \tag{22}$$

The Lagrange multipliers can then be solved from the constraints:

$$1 = \sum_{a\in\mathcal{A}} \bar{\pi}^*(a|x) = \exp\left(-\eta\lambda_x - 1\right) \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a))\right],$$

$$\lambda_x = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a))\right] - \frac{1}{\eta}. \tag{23}$$

By plugging (23) into (22) we deduce

$$\bar{\pi}^*(a|x) = \frac{\bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a))\right]}{\sum\limits_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma\left(PV^*_{\bar{\pi}}\right)(x,a))\right]}, \qquad \forall (x,a) \in \mathcal{Z}. \tag{24}$$

The results then follows by substituting (24) in (19).

■

The optimal policy $\bar{\pi}^*$ is a function of the base policy, the optimal value function $V^*_{\bar{\pi}}$ and the state transition probability P. One can first obtain the optimal value function $V^*_{\bar{\pi}}$ through the following fixed-point iteration:

$$V^{k+1}_{\bar{\pi}}(x) = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\big[\eta(r(x,a) + \gamma(PV^k_{\bar{\pi}})(x,a))\big], \tag{25}$$

and then compute $\bar{\pi}^*$ using (21). $\bar{\pi}^*$ maximizes the value function $V^{\pi}_{\bar{\pi}}$. However, we are not, in principle, interested in quantifying $\bar{\pi}^*$, but in solving the original MDP problem and computing π^*. The idea to further improve the policy towards π^* is to replace the base-line policy with the just newly computed policy of (21). The new policy can be regarded as a *new base-line policy*, and the process can be repeated again. This leads to a double-loop algorithm to find the optimal policy π^*, where the outer-loop and the inner-loop would consist of a policy update, Equation (21), and a value function update, Equation (25), respectively.

We then follow the following steps to derive the final DPP algorithm: **(i)** We introduce some extra smoothness to the policy update rule by replacing the double-loop algorithm by direct optimization of both value function and policy simultaneously using the following fixed point iterations:

$$V^{k+1}_{\bar{\pi}}(x) = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}_k(a|x) \exp\big[\eta(r(x,a) + \gamma(PV^k_{\bar{\pi}})(x,a))\big], \tag{26}$$

$$\bar{\pi}_{k+1}(a|x) = \frac{\bar{\pi}_k(a|x) \exp\big[\eta(r(x,a) + \gamma(PV^k_{\bar{\pi}})(x,a))\big]}{\exp\left(\eta V^{k+1}_{\bar{\pi}}(x)\right)}. \tag{27}$$

Further, **(ii)** we define the action preference function Ψ_k (Sutton and Barto, 1998, chap. 6.6), for all $(x,a) \in \mathcal{Z}$ and $k \geq 0$, as follows:

$$\Psi_{k+1}(x,a) \triangleq \frac{1}{\eta} \log \bar{\pi}_k(a|x) + r(x,a) + \gamma(PV^k_{\bar{\pi}})(x,a). \tag{28}$$

By comparing (28) with (27) and (26), we deduce:

$$\bar{\pi}_k(a|x) = \frac{\exp(\eta\Psi_k(x,a))}{\sum_{a'\in\mathcal{A}} \exp(\eta\Psi_k(x,a'))}, \tag{29}$$

$$V^k_{\bar{\pi}}(x) = \frac{1}{\eta}\log\sum_{a\in\mathcal{A}} \exp(\eta\Psi_k(x,a))). \tag{30}$$

Finally, **(iii)** by plugging (29) and (30) into (28) we derive:

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) - \mathcal{L}_\eta\Psi_k(x) + r(x,a) + \gamma(P\mathcal{L}_\eta\Psi_k)(x,a), \tag{31}$$

with $\mathcal{L}_\eta$ operator being defined by $\mathcal{L}_\eta\Psi(x) \triangleq 1/\eta \log\sum_{a\in\mathcal{A}}\exp(\eta\Psi(x,a))$. (31) is one form of the DPP equations. There is a more efficient and analytically more tractable version of the DPP equation, where we replace $\mathcal{L}_\eta$ by the Boltzmann soft-max $\mathcal{M}_\eta$ defined by $\mathcal{M}_\eta\Psi(x) \triangleq \sum_{a\in\mathcal{A}}\big[\exp(\eta\Psi(x,a))\Psi(x,a)/\sum_{a'\in\mathcal{A}}\exp(\eta\Psi(x,a'))\big]$.[4] In principle, we can provide formal analysis for both versions. However, the proof is somewhat simpler for the $\mathcal{M}_\eta$ case, which we make use of it in the rest of this paper. By replacing $\mathcal{L}_\eta$ with $\mathcal{M}_\eta$ we deduce the DPP recursion:

$$\begin{aligned}\Psi_{k+1}(x,a) = \mathcal{O}\Psi_k(x,a) &\triangleq \Psi_k(x,a) + r(x,a) + \gamma P\mathcal{M}_\eta\Psi_k(x,a) - \mathcal{M}_\eta\Psi_k(x)\\ &= \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x)\end{aligned}, \quad \forall(x,a)\in\mathcal{Z},$$

where $\mathcal{O}$ is an operator defined on the action preferences Ψ_k and π_k is the soft-max policy associated with Ψ_k:

$$\pi_k(a|x) \triangleq \frac{\exp(\eta\Psi_k(x,a))}{\sum_{a'\in\mathcal{A}}\exp(\eta\Psi_k(x,a'))}.$$

Appendix B. The Proof of Convergence of DPP- Theorem 2 and Theorem 4

In this section, we provide a formal analysis of the convergence behavior of DPP.

B.1 Poof of Theorem 2

In this subsection we establish a rate of convergence for the value function of the policy induced by DPP. The main result is in the form of following finite-iteration performance-loss

4. Replacing $\mathcal{L}_\eta$ with $\mathcal{M}_\eta$ is motivated by the following relation between these two operators:

$$|\mathcal{L}_\eta\Psi(x) - \mathcal{M}_\eta\Psi(x)| = 1/\eta H_\pi(x) \le \frac{\log(L)}{\eta}, \qquad \forall x\in\mathcal{X}, \tag{32}$$

with $H_\pi(x)$ is the entropy of the policy distribution π obtained by plugging Ψ into (A). In words, $\mathcal{M}_\eta\Psi(x)$ is close to $\mathcal{L}_\eta\Psi(x)$ up to the constant $\log(L)/\eta$. Also, both $\mathcal{L}_\eta\Psi(x)$ and $\mathcal{M}_\eta\Psi(x)$ converge to $\mathcal{M}\Psi(x)$ when η goes to $+\infty$. For the proof of (32) and further readings see MacKay (2003, chap. 31).

bound, for all $k \geq 0$:

$$\|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma\left(4V_{\max} + \frac{\log(L)}{\eta}\right)}{(1-\gamma)^2(k+1)}. \tag{33}$$

Here, Q^{π_k} is the action-values under the policy π_k and π_k is the policy induced by DPP at step k.

To derive (33) one needs to relate Q^{π_k} to the optimal Q^*. Unfortunately, finding a direct relation between Q^{π_k} and Q^* is not an easy task. Instead, we relate Q^{π_k} to Q^* via an auxiliary action-value function Q_k, which we define below. In the remainder of this Section we take the following steps: **(i)** we express Ψ_k in terms of Q_k in Lemma 13. **(ii)** we obtain an upper bound on the normed error $\|Q^* - Q_k\|$ in Lemma 14. Finally, **(iii)** we use these two results to derive a bound on the normed error $\|Q^* - Q^{\pi_k}\|$. For the sake of readability, we skip the formal proofs of the Lemmas in this section since we prove a more general case in Section C.

Now let us define the auxiliary action-value function Q_k. The sequence of auxiliary action-value functions $\{Q_0, Q_1, Q_2, \dots\}$ is obtained by iterating the initial $Q_0 = \Psi_0$ from the following recursion:

$$Q_k = \frac{k-1}{k}\mathcal{T}^{\pi_{k-1}}Q_{k-1} + \frac{1}{k}\mathcal{T}^{\pi_{k-1}}Q_0, \tag{34}$$

where π_k is the policy induced by the k^{th} iterate of DPP.

Lemma 13 relates Ψ_k with Q_k:

Lemma 13 *Let k be a positive integer. Then, we have:*

$$\Psi_k = kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0). \tag{35}$$

We then relate Q_k and Q^*:

Lemma 14 *Let Assumption 1 hold and L denotes the cardinality of $\mathcal{A}$ and k be a positive integer, also assume that $\|\Psi_0\| \leq V_{\max}$ then the following inequality holds:*

$$\|Q^* - Q_k\| \leq \frac{\gamma\left(4V_{\max} + \frac{\log(L)}{\eta}\right)}{(1-\gamma)k}. \tag{36}$$

Lemma 14 provides an upper bound on the normed-error $\|Q_k - Q^*\|$. We make use of Lemma 14 to prove the main result of this Subsection:

$$\begin{aligned}
\|Q^* - Q^{\pi_k}\| &= \|Q^* - Q_{k+1} + Q_{k+1} - \mathcal{T}^{\pi_k}Q^* + \mathcal{T}^{\pi_k}Q^* - Q^{\pi_k}\| \\
&\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k}Q^*\| + \|\mathcal{T}^{\pi_k}Q^* - \mathcal{T}^{\pi_k}Q^{\pi_k}\| \\
&\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k}Q^*\| + \gamma\|Q^* - Q^{\pi_k}\|.
\end{aligned}$$

By collecting terms we obtain:

$$\begin{aligned}
\|Q^* - Q^{\pi_k}\| &\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\|\right) \\
&= \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \left\|\frac{k}{k+1}\mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1}\mathcal{T}^{\pi_k} Q_0 - \mathcal{T}^{\pi_k} Q^*\right\|\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{k}{k+1}\|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_k\| + \frac{1}{k+1}\|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_0\|\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{\gamma}{k+1}\|Q^* - Q_0\|\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{2\gamma V_{\max}}{k+1}\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{\gamma(4V_{\max} + \log(L)/\eta)}{k+1}\right).
\end{aligned}$$

This combined with Lemma 14 completes the proof.

B.2 Proof of Theorem 4

First, we note that Q_k converges to Q^* (Lemma 14) and π_k converges to π^* by Corollary 3. Therefore, there exists a limit for Ψ_k since Ψ_k is expressed in terms of Q_k, Q_0 and π_{k-1} (Lemma 13).

Now, we compute the limit of Ψ_k. Q_k converges to Q^* with a linear rate from Lemma 14. Also, we have $V^* = \pi^* Q^*$ by definition of V^* and Q^*. Then, by taking the limit of (35) we deduce:

$$\begin{aligned}
\lim_{k\to\infty} \Psi_k(x,a) &= \lim_{k\to\infty} \left[kQ^*(x,a) + Q_0(x,a) - (k-1)V^*(x) - (\pi^* Q_0)(x)\right] \\
&= \lim_{k\to\infty} k(Q^*(x,a) - V^*(x)) \\
&\quad + Q_0(x,a) - (\pi^* Q_0)(x) + V^*(x).
\end{aligned}$$

We then deduce, for all $(x,a) \in \mathcal{Z}$,

$$\lim_{k\to\infty} \Psi_k(x,a) = \begin{cases} Q_0(x,a) - (\pi^* Q_0)(x) + V^*(x) & a = a^*(x) \\ -\infty & a \neq a^*(x) \end{cases},$$

where $a^*(x) = \max_{a\in\mathcal{A}}(Q^*(x,a))$. This combined with the assumption that the optimal policy is unique completes the proof.

Appendix C. Proof of Theorem 5

This section provides a formal theoretical analysis of the performance of dynamic policy programming in the presence of approximation.

Consider a sequence of the action preferences $\{\Psi_0, \Psi_1, \Psi_2, \dots\}$ as the iterates of (8). Our goal is to establish an ℓ_∞-norm performance loss bound of the policy induced by approximate DPP. The main result is that at iteration $k \geq 0$ of approximate DPP, we have:

$$\|Q^* - Q^{\pi_k}\| \leq \frac{1}{(1-\gamma)(k+1)} \left[\frac{2\gamma \left(4V_{\max} + \frac{\log(L)}{\eta} \right)}{(1-\gamma)} + \sum_{j=1}^{k+1} \gamma^{k-j+1} \|E_{j-1}\| \right], \tag{37}$$

where $E_k = \sum_{j=0}^{k} \epsilon_k$ is the cumulative approximation error up to step k. Here, Q^{π_k} denotes the action-value function of the policy π_k and π_k is the soft-max policy associated with Ψ_k.

As in the proof of Theorem 2, we relate Q^* with Q^{π_k} via an auxiliary action-value function Q_k. In the rest of this section, we first express Ψ_k in terms of Q_k in Lemma 15. Then, we obtain an upper bound on the normed error $\|Q^* - Q_k\|$ in Lemma 19. Finally, we use these two results to derive (37).

Now, let us define the auxiliary action-value function Q_k. The sequence of auxiliary action-value function $\{Q_0, Q_1, Q_2, \dots\}$ is resulted by iterating the initial action-value function $Q_0 = \Psi_0$ from the following recursion:

$$Q_k = \frac{k-1}{k} \mathcal{T}^{\pi_{k-1}} Q_{k-1} + \frac{1}{k} (\mathcal{T}^{\pi_{k-1}} Q_0 + E_{k-1}), \tag{38}$$

where (38) may be considered as an approximate version of (34). Lemma 15 relates Ψ_k with Q_k:

Lemma 15 *Let k be a positive integer and π_k denotes the policy induced by the approximate DPP at iteration k. Then we have*

$$\Psi_k = kQ_k + Q_0 - \pi_{k-1} \left((k-1)Q_{k-1} + Q_0 \right). \tag{39}$$

Proof We rely on induction for the proof of this theorem. The result holds for $k = 1$ since one can easily show that (39) reduces to (8). We then show that if (39) holds for k then it also holds for $k+1$. From (8) we have:

$$\begin{aligned}
\Psi_{k+1} &= \Psi_k + \mathcal{T}^{\pi_k} \Psi_k - \pi_k \Psi_k + \epsilon_k \\
&= kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0) + \mathcal{T}^{\pi_k}(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) \\
&\quad - \pi_k(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) + \epsilon_k \\
&= kQ_k + Q_0 + \mathcal{T}^{\pi_k}(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) - \pi_k(kQ_k + Q_0) \\
&\quad + E_k - E_{k-1} \\
&= kQ_k + Q_0 + r + \gamma P^{\pi_k}(kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0)) - \pi_k(kQ_k + Q_0) \\
&\quad + E_k - E_{k-1} \\
&= kQ_k + Q_0 + k(r + \gamma P^{\pi_k} Q_k) + r + \gamma P^{\pi_k} Q_0 - (k-1)(r + \gamma P^{\pi_{k-1}} Q_{k-1}) \\
&\quad - (r + \gamma P^{\pi_{k-1}} Q_0) + \pi_{k-1}((k-1)Q_{k-1} + Q_0)) - \pi_k(kQ_k + Q_0) + E_k - E_{k-1} \\
&= kQ_k - (k-1)\mathcal{T}^{\pi_{k-1}} Q_{k-1} - \mathcal{T}^{\pi_{k-1}} Q_0 - E_{k-1} + k\mathcal{T}^{\pi_k} Q_k + \mathcal{T}^{\pi_k} Q_0 + E_k \\
&\quad + Q_0 - \pi_k(kQ_k + Q_0) \\
&= (k+1)Q_{k+1} + Q_0 - \pi_k(kQ_k + Q_0),
\end{aligned}$$

in which we rely on

$$\pi_k \pi_{k-1}(\cdot) = \pi_{k-1}(\cdot), \qquad \mathcal{T}^{\pi_k}\pi_{k-1}(\cdot) = \mathcal{T}^{\pi_{k-1}}(\cdot),$$

and Equation (38).

Thus (39) holds for $k+1$, and is thus true for all $k \geq 1$. ■

Based on Lemma 15, one can express the policy induced by DPP, π_k, in terms of Q_k and Q_0:

Lemma 16 *For all $(x, a) \in \mathcal{Z}$:*

$$\pi_k(a|x) = \frac{\exp\left(\eta\left(kQ_k(x,a) + Q_0(x,a)\right)\right)}{\sum_{b\in\mathcal{A}} \exp\left(\eta\left(kQ_k(x,b) + Q_0(x,b)\right)\right)}.$$

Proof

$$\begin{aligned}\pi_k(a|x) &= \frac{\exp\left(\eta\left(kQ_k(x,a) + Q_0(x,a) - \pi_{k-1}((k-1)Q_{k-1} + Q_0)(x)\right)\right)}{Z(x)} \\ &= \frac{\exp\left(\eta\left(kQ_k(x,a) + Q_0(x,a)\right)\right)}{Z'(x)},\end{aligned}$$

where $Z(x)$ and $Z'(x) = Z(x)\exp\left(\eta\pi_{k-1}((k-1)Q_{k-1} + Q_0)(x)\right)$ are the normalization factors. ■

In an analogy to Lemma 14 we establish a bound on $\|Q^* - Q_k\|$ for which we make use of the following technical results:

Lemma 17 *Let $\eta > 0$ and $\mathcal{Y}$ be a finite set with cardinality L. Also assume that $\mathcal{F}$ denotes the space of real-valued functions on $\mathcal{Y}$. Then the following inequality holds for all $f \in \mathcal{F}$:*

$$\max_{y\in\mathcal{Y}} f(y) - \sum_{y\in\mathcal{Y}} \frac{\exp(\eta f(y)) f(y)}{\sum\limits_{y'\in\mathcal{Y}} \exp(\eta f(y'))} \leq \frac{\log(L)}{\eta}.$$

Proof For any $f \in \mathcal{F}$ we have:

$$\max_{y\in\mathcal{Y}} f(y) - \sum_{y\in\mathcal{Y}} \frac{\exp(\eta f(y)) f(y)}{\sum\limits_{y'\in\mathcal{Y}} \exp(\eta f(y'))} = \sum_{y\in\mathcal{Y}} \frac{\exp(-\eta g(y)) g(y)}{\sum\limits_{y'\in\mathcal{Y}} \exp(-\eta g(y'))},$$

with $g(y) = \max_{y\in\mathcal{Y}} f(y) - f(y)$. According to MacKay (2003, chap. 31):

$$\sum_{y\in\mathcal{Y}} \frac{\exp(-\eta g(y)) g(y)}{\sum\limits_{y'\in\mathcal{Y}} \exp(-\eta g(y'))} = -\frac{1}{\eta}\log\sum_{y\in\mathcal{Y}} \exp(-\eta g(y)) + \frac{1}{\eta} H_p,$$

where H_p is the entropy of probability distribution p defined by:

$$p(y) = \frac{\exp(-\eta g(y))}{\sum\limits_{y'\in\mathcal{Y}} \exp(-\eta g(y'))}.$$

Define $\mathcal{Y}^f_{\max} \subset \mathcal{Y}$ as the set of all entries of $\mathcal{Y}$ which maximizes $f \in \mathcal{F}$. The following steps complete the proof.

$$\begin{aligned}\sum_{y\in\mathcal{Y}} \frac{\exp(-\eta g(y))g(y)}{\sum_{y'\in\mathcal{Y}}\exp(-\eta g(y'))} &= -\frac{1}{\eta}\log\sum_{y\in\mathcal{Y}}\exp(-\eta g(y)) + \frac{1}{\eta}H_p \\ &\le -\frac{1}{\eta}\log\left[1+\sum_{y\notin\mathcal{Y}^f_{\max}}\exp(-\eta g(y)))\right] + \frac{1}{\eta}H_p \\ &\le \frac{1}{\eta}H_p \le \frac{\log(L)}{\eta},\end{aligned}$$

in which we make use of $-\frac{1}{\eta}\log\left[1+\sum_{y\notin\mathcal{Y}^f_{\max}}\exp(-\eta g(y)))\right] \le 0$. ■

Lemma 18 *Let $\eta > 0$ and k be a positive integer. Assume $\|Q_0\| \le V_{\max}$, then the following holds:*

$$\|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k}Q_k - \mathcal{T}^{\pi_k}Q_0\| \le \gamma\left(2V_{\max} + \frac{\log(L)}{\eta}\right).$$

Proof We have, by definition of operator $\mathcal{T}$:

$$\begin{aligned}\|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k}Q_k - \mathcal{T}^{\pi_k}Q_0\| &\le \gamma\,\|kP\mathcal{M}Q_k + P\mathcal{M}Q_0 - kP^{\pi_k}Q_k - P^{\pi_k}Q_0\| \\ &= \gamma\,\|P(\mathcal{M}kQ_k + \mathcal{M}Q_0 - \pi_k(kQ_k+Q_0))\| \\ &\le \gamma\,\|\mathcal{M}kQ_k + \mathcal{M}Q_0 - \pi_k(kQ_k+Q_0)\| \\ &\le \gamma\,\|2\mathcal{M}Q_0 + \mathcal{M}(kQ_k+Q_0) - \pi_k(kQ_k+Q_0)\| \\ &\le \gamma\,(2\,\|Q_0\| + \|\mathcal{M}(kQ_k+Q_0) - \mathcal{M}_\eta(kQ_k+Q_0)\|),\end{aligned} \tag{40}$$

where in the last line we make use of Lemma 16. The result then follows by comparing (40) with Lemma 17.

■

Now, we prove a bound on $\|Q^* - Q_k\|$:

Lemma 19 *Let Assumption 1 hold. Define Q_k by (38). Let L denotes the cardinality of $\mathcal{A}$ and k be a non-negative integer, also, assume that $\|\Psi_0\| \le V_{\max}$, then the following inequality holds:*

$$\|Q^* - Q_k\| \le \frac{\gamma\left(4V_{\max} + \frac{\log(L)}{\eta}\right)}{(1-\gamma)k} + \frac{1}{k}\sum_{j=1}^{k}\gamma^{k-j}\|E_{j-1}\|.$$

Proof We rely on induction for the proof of this Lemma. Obviously the result holds for $k = 0$. Then we need to show that if (36) holds for k it also holds for $k+1$:

$$
\begin{aligned}
\|Q^* - Q_{k+1}\| &= \left\| \mathcal{T}Q^* - \left(\frac{k}{k+1} \mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1} (\mathcal{T}^{\pi_k} Q_0 + E_k) \right) \right\| \\
&= \left\| \frac{1}{k+1} (\mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_0) + \frac{k}{k+1} (\mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_k) - \frac{1}{k+1} E_k \right\| \\
&= \frac{1}{k+1} \|\mathcal{T}Q^* - \mathcal{T}Q_0 + \mathcal{T}Q_0 - \mathcal{T}^{\pi_k} Q_0 + k(\mathcal{T}Q^* - \mathcal{T}Q_k + \mathcal{T}Q_k - \mathcal{T}^{\pi_k} Q_k)\| \\
&\quad + \frac{1}{k+1} \|E_k\| \\
&\leq \frac{1}{k+1} [\|\mathcal{T}Q^* - \mathcal{T}Q_0\| + \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\|] \\
&\quad + \frac{k}{k+1} \|\mathcal{T}Q^* - \mathcal{T}Q_k\| + \frac{1}{k+1} \|E_k\| \\
&\leq \frac{1}{k+1} [\gamma \|Q^* - Q_0\| + \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\|] \\
&\quad + \frac{\gamma k}{k+1} \|Q^* - Q_k\| + \frac{1}{k+1} \|E_k\| .
\end{aligned}
\tag{41}
$$

Now based on Lemma 18 and by plugging (36) into (41) we have:

$$
\begin{aligned}
\|Q^* - Q_{k+1}\| &\leq \frac{\gamma}{k+1} \left[4V_{\max} + \frac{\log L}{\eta} \right] + \frac{\gamma k}{k+1} \left[\frac{\gamma \left(4V_{\max} + \frac{\log(L)}{\eta} \right)}{k(1-\gamma)} + \frac{1}{k} \sum_{j=1}^{k} \gamma^{k-j} \|E_{j-1}\| \right] \\
&\quad + \frac{1}{k+1} \|E_k\| \\
&= \frac{\gamma \left(4V_{\max} + \frac{\log(L)}{\eta} \right)}{(1-\gamma)(k+1)} + \frac{1}{k+1} \sum_{j=1}^{k+1} \gamma^{k-j+1} \|E_{j-1}\|.
\end{aligned}
$$

The result then follows, for all $k \geq 0$, by induction. ■

Lemma 19 provides an upper-bound on the normed-error $\|Q^* - Q_k\|$. We make use of this result to derive a bound on the performance loss $\|Q^* - Q^{\pi_k}\|$:

$$
\begin{aligned}
\|Q^* - Q^{\pi_k}\| &= \|Q^* - Q_{k+1} + Q_{k+1} - \mathcal{T}^{\pi_k} Q^* + \mathcal{T}^{\pi_k} Q^* - Q^{\pi_k}\| \\
&\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\| + \|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q^{\pi_k}\| \\
&\leq \|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\| + \gamma \|Q^* - Q^{\pi_k}\| .
\end{aligned}
$$

By collecting terms we obtain:

$$\begin{aligned}
\|Q^* - Q^{\pi_k}\| &\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \|Q_{k+1} - \mathcal{T}^{\pi_k} Q^*\|\right) \\
&= \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \left\|\frac{k}{k+1}\mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1}(\mathcal{T}^{\pi_k} Q_0 + E_k) - \mathcal{T}^{\pi_k} Q^*\right\|\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{k}{k+1}\|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_k\| + \frac{1}{k+1}\|\mathcal{T}^{\pi_k} Q^* - \mathcal{T}^{\pi_k} Q_0\|\right) \\
&\quad + \frac{1}{(1-\gamma)(k+1)}\|E_k\| \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{1}{k+1}\|E_k\| + \frac{\gamma}{k+1}\|Q^* - Q_0\|\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{1}{k+1}\|E_k\| + \frac{2\gamma V_{\max}}{k+1}\right) \\
&\leq \frac{1}{1-\gamma}\left(\|Q^* - Q_{k+1}\| + \frac{\gamma k}{k+1}\|Q^* - Q_k\| + \frac{1}{k+1}\|E_k\| + \frac{\gamma(4V_{\max} + \log(L)/\eta)}{k+1}\right).
\end{aligned}$$

This combined with the result of Lemma 19 completes the proof.

Appendix D. The Proof of Convergence of DPP-RL - Theorem 9 and Theorem 10

We begin the analysis by introducing some new notation. Let us define $\mathcal{F}_k$ as the filtration generated by the sequence of all random variables $\{y_1, y_2, y_3, \ldots, y_k\}$ drawn from the distribution $P(\cdot|x,a)$ for all $(x,a) \in \mathcal{Z}$. We know, by the definition of ϵ_k, that $\mathbb{E}(\epsilon_k(x,a)|\mathcal{F}_{k-1}) = 0$, which means that for all $(x,a) \in \mathcal{Z}$ the sequence of estimation errors $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_k\}$ is a martingale difference sequence w.r.t. the filtration $\mathcal{F}_k$. Now, we provide the proof of Lemma 8, on which we rely for the analysis of both Theorem 9 and Theorem 10:

Proof of Lemma 8 We first prove that $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq \frac{2\gamma \log L}{\eta(1-\gamma)} + V_{\max}$ by induction. Let us assume that the bound $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq \frac{2\gamma \log L}{\eta(1-\gamma)} + V_{\max}$ holds. Thus

$$\begin{aligned}
\|\mathcal{T}_{k+1}^{\pi_k}\Psi_{k+1}\| &\leq \|r\| + \gamma \|P^{\pi_k}\Psi_{k+1}\| \leq \|r\| + \gamma \|\mathcal{M}_\eta \Psi_{k+1}\| \\
&= \|r\| + \gamma \left\|\mathcal{M}_\eta \left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta \Psi_k\right)\right\| \\
&\leq \|r\| + \gamma \left\|\mathcal{M}_\eta \left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta \Psi_k\right) - \mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta \Psi_k\right)\right\| \\
&\quad + \gamma \left\|\mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta \Psi_k\right)\right\| \\
&\leq \|r\| + \frac{\gamma \log L}{\eta} + \gamma \left\|\mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta \Psi_k\right)\right\| \\
&= \|r\| + \frac{\gamma \log L}{\eta} + \gamma \left\|\mathcal{M}\left(\Psi_k + \mathcal{T}_k^{\pi_k}\Psi_k - \mathcal{M}_\eta \Psi_k + \mathcal{M}\Psi_k - \mathcal{M}\Psi_k\right)\right\| \\
&\leq \|r\| + \frac{\gamma \log L}{\eta} + \gamma \|\mathcal{M}(\mathcal{M}\Psi_k - \mathcal{M}_\eta \Psi_k)\| + \gamma \|\mathcal{M}\left(\Psi_k - \mathcal{M}\Psi_k\right)\| \\
&\quad + \gamma \left\|\mathcal{M}\mathcal{T}_k^{\pi_k}\Psi_k\right\| \\
&\leq \|r\| + \frac{2\gamma \log L}{\eta} + \gamma \left\|\mathcal{T}_k^{\pi_k}\Psi_k\right\| \leq \|r\| + \frac{2\gamma \log L}{\eta} + \frac{2\gamma^2 \log(L)}{\eta(1-\gamma)} + \gamma V_{\max} \\
&\leq \frac{2\gamma \log L}{\eta(1-\gamma)} + R_{\max} + \gamma V_{\max} = \frac{2\gamma \log L}{\eta(1-\gamma)} + V_{\max},
\end{aligned}$$

where we make use of Lemma 17 to bound the difference between the max operator $\mathcal{M}(\cdot)$ and the soft-max operator $\mathcal{M}_\eta(\cdot)$. Now, by induction, we deduce that for all $k \geq 0$, $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq 2\gamma \log L/(\eta(1-\gamma)) + V_{\max}$. The bound on ϵ_k is an immediate consequence of this result. ■

D.1 Proof of Theorem 9

In this subsection, we provide the proof of Theorem 9 which guarantees that DPP-RL asymptotically converges to the optimal policy w.p. 1.

We make use of the result of Lemma 8 and Corollary 6 to prove the theorem. We begin by recalling the result of Corollary 6:

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2} \lim_{k\to\infty} \frac{1}{k+1} \|E_k\| .$$

Therefore, to prove the convergence of DPP-RL, one only needs to prove that $1/(k+1)\|E_k\|$ asymptotically converges to 0 w.p. 1. For this we rely on the strong law of large numbers for martingale differences (Hoffmann-Jørgensen and Pisier, 1976), which states that the average of a sequence of martingale differences asymptotically converges, almost surely, to 0 if the second moments of all entries of the sequence are bounded by some $0 \leq U \leq \infty$. This is the case for the sequence of martingales $\{\epsilon_1, \epsilon_2, \dots\}$ since we already have proven the boundedness of $\|\epsilon_k\|$ in Lemma 8. Thus, we deduce:

$$\lim_{k\to\infty} \frac{1}{k+1} |E_k(x,a)| = 0, \qquad \text{w.p. 1.}$$

Thus:

$$\lim_{k\to\infty} \frac{1}{k+1} \|E_k\| = 0, \qquad \text{w.p. } 1. \tag{42}$$

The result then follows by combining (42) with Corollary 6.

D.2 Proof of Theorem 10

In this subsection, we prove Theorem 10, for which we rely on a maximal Azuma's inequality (see, e.g., Cesa-Bianchi and Lugosi, 2006, appendix, pg. 359):

Lemma 20 (Azuma, 1967) *Let $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_K\}$ be a martingale difference sequence w.r.t. a sequence of random variables $\{X_1, X_2, \ldots, X_K\}$, i.e., $\mathbb{E}(Y_{k+1}|X_1, \ldots X_k) = 0$ for all $0 < k \leq K$. Also, let $\mathcal{Y}$ be uniformly bounded by $U > 0$. Define $S_k = \sum_{i=1}^{k} Y_i$. Then, for any $\epsilon > 0$, we have*

$$\Pr\left(\max_{1\leq k\leq K} S_k > \epsilon\right) \leq \exp\left(\frac{-\epsilon^2}{2KU^2}\right).$$

We recall the result of Theorem 5 at iteration k:

$$\|Q^* - Q^{\pi_k}\| \leq \frac{\gamma\left(4V_{\max} + \frac{\log(L)}{\eta}\right)}{(1-\gamma)^2(k+1)} + \frac{1}{(1-\gamma)(k+1)} \sum_{j=0}^{k} \gamma^{k-j} \|E_j\|.$$

Note that the main difference between this bound and the result of Theorem 10 is just in the second term. So, to prove Theorem 10 we need to show that the following inequality holds, with probability at least $1-\delta$:

$$\frac{1}{k+1} \sum_{j=0}^{k} \gamma^{k-j} \|E_j\| \leq \frac{4(\gamma \log(L)/\eta + 2R_{\max})}{(1-\gamma)^2} \sqrt{\frac{2\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}{k+1}}. \tag{43}$$

We first notice that:

$$\frac{1}{k+1} \sum_{j=0}^{k} \gamma^{k-j} \|E_j\| \leq \frac{1}{k+1} \sum_{k=0}^{j} \gamma^{k-j} \max_{0\leq j\leq k} \|E_j\| \leq \frac{\max_{0\leq j\leq k} \|E_j\|}{(1-\gamma)(k+1)}. \tag{44}$$

Therefore, in order to prove (43) it is sufficient to bound $\max_{0\leq j\leq k} \|E_j\| = \max_{(x,a)\in\mathcal{Z}} \max_{0\leq j\leq k} |E_{k-1}(x,a)|$ in high probability.

We begin by proving high probability bound on $\max_{0\leq j\leq k} |E_j(x,a)|$ for a given (x,a). We first notice that

$$\begin{aligned}
\Pr\left(\max_{0\leq j\leq k} |E_j(x,a)| > \epsilon\right) &= \Pr\left(\max\left[\max_{0\leq j\leq k}(E_j(x,a)), \max_{0\leq j\leq k}(-E_j(x,a))\right] > \epsilon\right) \\
&= \Pr\left(\left\{\max_{0\leq j\leq k}(E_j(x,a)) > \epsilon\right\} \bigcup \left\{\max_{0\leq j\leq k}(-E_j(x,a)) > \epsilon\right\}\right) \\
&\leq \Pr\left(\max_{0\leq j\leq k}(E_j(x,a)) > \epsilon\right) + \Pr\left(\max_{0\leq j\leq k}(-E_j(x,a)) > \epsilon\right),
\end{aligned} \tag{45}$$

The sequence of random variables $\{\epsilon_0(x,a), \epsilon_1(x,a), \cdots, \epsilon_k(x,a)\}$ is a martingale difference sequence w.r.t. the filtration $\mathcal{F}_k$ (generated by the random samples $\{y_0, y_1, \ldots, y_k\}(x,a)$ for all (x,a)), i.e., $\mathbb{E}[\epsilon_k(x,a)|\mathcal{F}_{k-1}] = 0$. It follows from Lemma 20 and Lemma 8 that for any $\epsilon > 0$ we have

$$\begin{aligned} \Pr\left(\max_{0\leq j\leq k}(E_j(x,a)) > \epsilon\right) &\leq \exp\left(\frac{-\epsilon^2}{2(k+1)(\frac{4\gamma\log L}{\eta(1-\gamma)} + 2V_{\max})^2}\right) \\ \Pr\left(\max_{0\leq j\leq k}(-E_j(x,a)) > \epsilon\right) &\leq \exp\left(\frac{-\epsilon^2}{2(k+1)(\frac{4\gamma\log L}{\eta(1-\gamma)} + 2V_{\max})^2}\right). \end{aligned} \tag{46}$$

By combining (46) with (45) we deduce that

$$\Pr\left(\max_{0\leq j\leq k}|E_j(x,a)| > \epsilon\right) \leq 2\exp\left(\frac{-\epsilon^2}{2(k+1)(\frac{4\gamma\log L}{\eta(1-\gamma)} + 2V_{\max})^2}\right),$$

and a union bound over the state-action space leads to

$$\Pr\left(\max_{0\leq j\leq k}\|E_j\| > \epsilon\right) \leq 2|\mathcal{X}||\mathcal{A}|\exp\left(\frac{-\epsilon^2}{2(k+1)(\frac{4\gamma\log L}{\eta(1-\gamma)} + 2V_{\max})^2}\right).$$

For any $0 < \delta < 1$, this bound can be re-expressed as

$$\Pr\left(\max_{0\leq j\leq k}\|E_j\| \leq \left(\frac{4\gamma\log L}{\eta(1-\gamma)} + 2V_{\max}\right)\sqrt{2(k+1)\log\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}}\right) \geq 1-\delta.$$

This combined with (44) proves (43) and Theorem 10.

References

A. Antos, R. Munos, and Cs. Szepesvári. Fitted Q-iteration in continuous action-space MDPs. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*. MIT Press, 2008.

M. Gheshlaghi Azar, R. Munos, M. Ghavamzadeh, and H. J. Kappen. Speedy Q-learning. In *Advances in Neural Information Processing Systems 24*, pages 2411–2419. 2011.

J. A. Bagnell and J. G. Schneider. Covariant policy search. In *International Joint Conference on Artificial Intelligence*, pages 1019–1024, 2003.

P. L. Bartlett and A. Tewari. REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 2009.

A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transaction on System, Man, and Cybernetics*, pages 834–846, 1983.

J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume II. Athena Scientific, third edition, 2007.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

S. Bhatnagar, R. S. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

C. Daniel, G. Neumann, and J. Peters. Hierarchical relative entropy policy search. In *International Conference on Artificial Intelligence and Statistics*, 2012.

D. P. de Farias and B. Van Roy. On the existence of fixed points for approximate value iteration and temporal-difference learning. *Journal of Optimization Theory and Applications*, 105(3):589–608, 2000.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, December 2005.

E. Even-Dar and Y. Mansour. Learning rates for Q-learning. *Journal of Machine Learning Research*, 5:1–25, 2003.

A. Farahmand, M. Ghavamzadeh, Cs. Szepesvári, and S. Mannor. Regularized fitted Q-iteration: Application to planning. In *European Workshop on Reinforcement Learning*, Lecture Notes in Computer Science. Springer, 2008a.

A. Farahmand, M. Ghavamzadeh, Cs. Szepesvári, and S. Mannor. Regularized policy iteration. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*. MIT Press, 2008b.

A. Farahmand, R. Munos, and Cs. Szepesvári. Error propagation for approximate policy and value iteration. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*. MIT Press, 2010.

J. Hoffmann-Jørgensen and G. Pisier. The law of large numbers and the central limit theorem in banach spaces. *The Annals of Probability*, 4(4):587–599, 1976.

T. Jaakkola, M. I. Jordan, and S. Singh. On the convergence of stochastic iterative dynamic programming. *Neural Computation*, 6(6):1185–1201, 1994.

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

S. Kakade. Natural policy gradient. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2001.

H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Statistical Mechanics*, 2005(11):P11011, 2005.

M. Kearns and S. Singh. Finite-sample convergence rates for Q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems 12*. MIT Press, 1999.

J. Kober and J. Peters. Policy search for motor primitives in robotics. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*. MIT Press, 2008.

S. Koenig and Re. G. Simmons. Complexity analysis of real-time reinforcement learning. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*. AAAI Press, 1993.

V. Konda and J. N. Tsitsiklis. On actor-critic algorithms. *SIAM Journal on Control and Optimization*, 42(4):1143–1166, 2003.

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

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

H. Maei, Cs. Szepesvári, S. Bhatnagar, and R. S. Sutton. Toward off-policy learning control with function approximation. In *Proceedings of the 27th Annual International Conference on Machine Learning*. Omnipress, 2010.

F. Melo, S. Meyn, and I. Ribeiro. An analysis of reinforcement learning with function approximation. In *Proceedings of 25 International Conference on Machine Learning*. ACM, 2008.

R. Munos. Error bounds for approximate value iteration. In *Proceedings of the 20th National Conference on Artificial Intelligence*, volume II. AAAI Press, 2005.

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

T. J. Perkins and D. Precup. A convergent form of approximate policy iteration. In *Advances in Neural Information Processing Systems 15*. MIT Press, 2002.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7–9):1180–1190, 2008.

J. Peters, K. Mülling, and Y. Altun. Relative entropy policy search. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*. AAAI Press, 2010.

S. Singh, T. Jaakkola, M.L. Littman, and Cs. Szepesvari. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3):287–308, 2000.

S. Still and D. Precup. An information- theoretic approach to curiosity-driven reinforcement learning. In *International Conference on Humanoid Robotics*, 2011.

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

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*. MIT Press, 1999.

Cs. Szepesvári. The asymptotic convergence-rate of Q-learning. In *Advances in Neural Information Processing Systems 10*. MIT Press, 1997.

Cs. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

Cs. Szepesvári and W. Smart. Interpolation-based Q-learning. In *Proceedings of 21st International Conference on Machine Learning*, ACM, 2004.

I. Szita and Cs. Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1031–1038. Omnipress, 2010.

C. Thiery and B. Scherrer. Least-squares lambda policy iteration: Bias-variance trade-off in control problems. In *Proceedings of the 27th Annual International Conference on Machine Learning*. Omnipress, 2010.

E. Todorov. Linearly-solvable Markov decision problems. In *Proceedings of the 20th Annual Conference on Neural Information Processing Systems*. MIT Press, 2006.

N. Vlassis and M. Toussaint. Model-free reinforcement learning as mixture learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009.

T. Wang, M. Bowling, and D. Schuurmans. Dual representations for dynamic programming and reinforcement learning. In *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, 2007a.

T. Wang, D. Lizotte, M. Bowling, and D. Schuurmans. Stable dual dynamic programming. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, 2007b.

C.J.C.H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 3(8):279–292, 1992.