On the Theory of Reinforcement Learning Methods, Convergence Analysis and Sample Complexity

Proefschrift

ter verkrijging van de graad van doctor aan de Radboud Universiteit Nijmegen
op gezag van de rector magnificus prof. mr. S.C.J.J. Kortmann, volgens
besluit van het college van decanen in het openbaar te verdedigen op dinsdag
25 oktober 2012 om 13.00 uur precies

door

Mohammad Gheshlaghi Azar

geboren op 1 mei 1981
te Tehran, Iran

Promotor: Prof. dr. H. J. Kappen

Copromotor: Dr. R. Munos (INRIA, Lille)

Manuscriptcommissie:

Prof. dr. T. Heskes
Prof. dr. P. Auer (Leoben Universiteit, Leoben)
Dr. N. Vlassis (Luxembourg Universiteit, Luxembourg)

ISBN 978-90-8891472-0
Printed by Uitgeverij BOXPress, Weerdskampweg 15, 5222 BA
's-Hertogenbosch

Contents

CHAPTER 1

Introduction

In this chapter, we introduce some of the basic concepts of reinforcement learning and Markov decision processes, on which we rely in the rest of this thesis. We also take a brief look on the important problems that we encounter in this thesis and describe our proposed solutions for those problems.

Consider the problem of navigating an autonomous mobile robot in an unknown world. The world may involve several static and dynamic objects including other robots or humans, each of them may have some influence on the behavior of robot. The robot may also influence the world through her control *actions*. However, she does not have access to any map or a model of her world for planning, nor she has any idea that to what extent her actions may change the world. On the other hand, the robot is equipped with a set of sensors, which provides her with some information regarding the state of the world. This information may involve the location of mobile robot in the world, e.g., through a GPS device, her distance from the nearby objects or some visual information such as digital images, video, etc. In addition, upon taking a control action, the world may provide the robot with a reward or a punishment (negative reward), depending on her performance. For instance, the robot may be rewarded for finding a treasure or locating the place of an accident, and she may be punished

for hitting an obstacle or consuming too much energy, i.e., taking large control actions. The robot may use these rewards and punishments for optimizing her performance. For example, consider the case that upon hitting an obstacle the robot receives a large negative reward. She can make use of this bad experience to avoid the obstacle in the next encounter. For that she may need to build some kind of map or model of the world using the sensory data. Based on this model, she can find a sequence of control actions, which minimizes the chance of hitting the obstacle.

The above example presents some of the features of the kind of problems we deal with in reinforcement learning (RL) (Sutton and Barto, 1998). In RL, similar to the example of the mobile robot, the agent often has no substantive prior knowledge of the outside world. However, she can *discover* the world through trial and error by interacting with her surrounding environment: the agent may try different control policies, i.e., a sequence of control actions, observe the changes in the state of the world and accumulate the rewards. Based on these information, she aims at finding the *optimal policy*, i.e., the policy which optimizes the long-term performance. The long-term performance of the agent, also known as the value function, is usually quantified as the expected value of the (weighted)-sum of future rewards upon taking a control policy from each state. The problem of finding the optimal policy or the optimal value function from a set of observations, control actions and rewards is one of the main challenges in the field of RL.

In this thesis, we consider the problem of reinforcement learning in the infinite-horizon discounted reward Markov decision processes (MDPs) (Puterman, 1994b). In particular, we focus on finite state and action problems, where it is possible to store the (action)-value function for all state-(action) pairs. Nevertheless, our algorithms and theoretical results can be extended to large-scale (possibly continuous state-action) problems by the means of function approximation and Monte-Carlo sampling. In Chapter 2, we consider the possibility of such an extension for dynamic policy programming (DPP) algorithm. Throughout this thesis, we assume that the state of the world is fully observable by the agent, i.e., we do not consider the partial observable problems where the observation of state is incomplete or corrupted by the noise. In addition, we usually assume that we have access to a stochastic oracle (generative model) which can generate state-transition samples for every state-action pair on the request of the RL agent. This is a common assumption in the theory of RL which simplifies the analysis of RL algorithms (Farahmand et al., 2008b; Even-Dar and Mansour, 2003; Kearns and Singh, 1999). Nonetheless, in Chapter 3 we relax this assumption by extending our results to a more realistic setting, where the

samples are streams of state-action pairs generated by following a control policy.

Many RL methods rely on the idea of dynamic programming (DP) for estimating the optimal policy or the optimal value function (Bertsekas, 2007a). Dynamic programming aims at finding the optimal value function by iterating some initial value function through the Bellman recursion until it converges to the the optimal value function. For the discounted MDPs, one can prove that DP converges to the optimal value function exponentially fast (Bertsekas, 2007b). However, in large-scale problems the exact DP may become computationally infeasible since the computational cost of exact DP may scale quadratically with the size of state space. Also, DP relies on an explicit knowledge of the state-transition model and reward function which is often not available in RL. A common approach to deal with these problems is through approximate dynamic programming (ADP) which combines DP with the function approximation techniques and the Monte-Carlo simulation. ADP algorithms such as approximate policy iteration (API) and approximate value iteration (AVI) have been successfully applied to many real-world problems. Furthermore, the asymptotic and finite-time behavior of ADP algorithms has been throughly investigated in previous studies (Farahmand et al., 2010; Thiery and Scherrer, 2010; Munos and Szepesvári, 2008; Antos et al., 2007; Munos, 2005, 2003; Bertsekas and Tsitsiklis, 1996).

In Chapter 2, we introduce a new convergent policy iteration algorithm, called dynamic policy programming (DPP). DPP is different from the standard policy iteration in the sense that it does not rely on the Bellman equation to estimate the value function. Instead, DPP iterates the parametrized policy from the DPP update rule. We prove that the exact variant of DPP rapidly converges to the optimal policy. We also investigate the behavior of DPP in the presence of approximation, where we prove theoretical guarantees for approximate DPP in the form of asymptotic and finite-iteration performance-loss bounds. Our theoretical results indicate that the performance loss of DPP, at each iteration, depends on the average of the approximation errors of all previous iterations. This is in contrast to the previous bounds of AVI and API, which are expressed in terms of the supremum of the errors of previous iterations. The bound with dependency on the average error can be useful, when we rely on sampling to estimate the optimal policy. This is due to the fact that the average of some random variables, in this case sampling errors, is often smaller in size than their supremum.

RL algorithms may be considered as model based or model free. In model-based RL, we first estimate the state-transition model and the reward function using a batch of state-transition and reward samples. We then plug this em-

pirical model in to the Bellman equation and estimate the value function and the optimal policy through DP methods such as value iteration and policy iteration (Wiering and van Otterlo, 2012a). In model-free methods we directly aim at estimating the optimal value function or the optimal policy without resorting to a model of the MDP. Examples of model-free methods are Q-learning (Watkins and Dayan, 1992), actor-critic (Barto et al., 1983) and policy gradient methods (Sutton et al., 1999). Model-free methods, in general, have better memory requirement than Model-based methods since they do not need to store the model data. On the other hand, the existing model-free RL algorithms such as Q-learning have been proven to be less efficient than their model-based counterparts in terms of the number of samples required to find a near-optimal policy (Even-Dar and Mansour, 2003; Kakade, 2004).

In Chapter 3, we introduce a new model-free RL algorithm called speedy Q-learning (SQL) to address the problem of slow convergence in the standard Q-learning. Like Q-learning, SQL relies on stochastic approximation (Kushner and Yin, 2003) for estimating the optimal action-value function in an incremental fashion. However, SQL differs form Q-learning by making use of two previous estimates of the optimal action value-function. This allows SQL to closely follow the Bellman operator which leads to a fast rate of convergence of $1/\sqrt{k}$, where k is the number of iterations. Furthermore, one can show that SQL is superior to model-based methods such as Q-value iteration in terms of the number of computations required to achieve a near-optimal estimate of the optimal value function.

Finally, an important theoretical problem in the field of reinforcement learning is to analyze the finite-time performance of RL algorithms. One may quantify the finite-time performance of an RL algorithm in terms of the number of samples required by the algorithm to find a near optimal policy, also known as the sample complexity of the algorithm (Kakade, 2004). There exist some previous results which prove bounds on the sample complexity of different model-based and model-free methods. In particular, the sample complexity of RL methods under the *probably approximately correct* (PAC) model has been extensively studied in several previous works (Munos and Szepesvári, 2008; Even-Dar et al., 2006; Kakade, 2004; Even-Dar and Mansour, 2003; Kearns and Singh, 1999). The best PAC-style upper-bound on the sample complexity of reinforcement learning matches the lower-bound of RL (Even-Dar et al., 2006) in terms of *almost* all the parameters of interest. However, in the case of discounted MDP with a discount factor γ, there still exists a gap of β^2 between the best previously-known upper-bound and lower bound, where $\beta = 1/(1-\gamma)$ is the *effective* horizon of the discounted MDP.

In Chapter 4, we focus on closing the above-mentioned gap between the upper bound and the lower bound of model-based RL algorithms. We prove new upper bounds on the sample complexity of two well-known RL algorithms, the model-based value iteration (QVI) and the model-based policy iteration (PI) which improve on the best previous result by a factor of β. We also prove a new general lower bound on the sample complexity of every RL algorithm which matches the upper bound up to a multiplicative factor. For the upper bound we rely on the Bernstein's inequality which bounds the difference between the empirical mean and the true mean of a random variable in terms of the variance of the random variable. This combined with the fact that the variance of sum of discounted rewards satisfies a Bellman-like equation leads to a tight upper bound on the sample complexity of QVI and PI. For the lower-bound, we find a new *hard* class of MDPs in which every RL algorithm requires a large number of samples, as much as required by the upper bound up to a multiplicative constant, to achieve a near-optimal estimate of the optimal value function on all entries of the class.

CHAPTER 2

Dynamic Policy Programming

In this chapter, 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.[a]

[a]This chapter is based on (Azar and Kappen, 2012) and (Azar et al., 2011a) (see the publications by the author).

2.1 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, 2007b; Lagoudakis and Parr, 2003; Perkins and Precup, 2002; 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[b]

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2} \limsup_{k\to\infty} \|\varepsilon_k\| , \tag{2.1}$$

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 (2.1) 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

[b]For AVI the approximation error ε_k is defined as the error associated with the approximation of the Bellman optimality operator. In the case of API, ε_k is the policy evaluation error (see Farahmand et al., 2010; Bertsekas and Tsitsiklis, 1996, chap. 6, for more details).

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 $\bar{\varepsilon}_k = 1/(k+1)\sum_{j=0}^{k}\varepsilon_j$ 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 $\bar{\varepsilon}_k$ 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 chapter, 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 $\|\bar{\varepsilon}_k\|$. 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 $O(1/((1-\gamma)^6\varepsilon^2))$ 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 2.6 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 chapter is organized as follows. In Section 2.2, we present the notation which is used in this chapter. We introduce DPP and we investigate its convergence properties in Section 2.3. In Section 2.4, 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 2.5, presents numerical experiments on several problem domains including the optimal replacement problem (Munos and Szepesvári, 2008) and a stochastic grid world. In Section 2.6 we briefly review some related work. Finally, in Section 2.7, we summarize our results and discuss some of the implications of our work.

2.2 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.[c] 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.

[c] For further reading see Szepesvári (2010).

2.2.1 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 $P(\cdot|x, a)$. $\mathcal{R}$ 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 2.1 (MDP Regularity)**.** *We assume that $\mathcal{X}$ and $\mathcal{A}$ are finite sets with the cardinalities $|\mathcal{X}|$ and $|\mathcal{A}|$, respectively. 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 2.1. *To keep the representation succinct, we make use of the shorthand 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:

$$V^*(x) = \sup_{\pi(\cdot|x)} \sum_{\substack{y\in\mathcal{X}\\ a\in\mathcal{A}}} \pi(a|x)\left[r(x,a) + P(y|x,a)V^*(y)\right]$$
$$= \max_{a\in\mathcal{A}} \left[r(x,a) + \sum_{y\in\mathcal{X}} P(y|x,a)V^*(y) \right] \quad , \qquad \forall x \in \mathcal{X}. \tag{2.2}$$

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, 2007b, chap. 1). In other words, for any two real-valued action-value functions Q and Q' and every policy π, we have

$$\|\mathcal{T}Q - \mathcal{T}Q'\| \leq \gamma \|Q - Q'\|, \qquad \|\mathcal{T}^\pi Q - \mathcal{T}^\pi Q'\| \leq \gamma \|Q - Q'\|. \tag{2.3}$$

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

$$\mathcal{T}^\pi Q(x,a) = r(x,a) + \gamma(P^\pi Q)(x,a), \qquad \mathcal{T}Q(x,a) = r(x,a) + \gamma(P\mathcal{M}Q)(x,a). \tag{2.4}$$

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.

2.3 Dynamic Policy Programming

In this section, we introduce and analyze the DPP algorithm. We first present the *dynamic policy programming* (DPP) algorithm in Subsection 2.3.1 (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 2.3.2.

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 2.1):

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

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:

$$\pi_k(a|x) = \frac{\exp(\eta\Psi_k(x,a))}{\sum_{b\in\mathcal{A}} \exp(\eta\Psi_k(x,b))}, \qquad \forall (x,a) \in \mathcal{Z}. \tag{2.5}$$

Based on (2.5) one can re-express the DPP operator on the action preferences Ψ_k as

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}. \tag{2.6}$$

Algorithm 2.1 (DPP) Dynamic Policy Programming

Require: Action preferences $\Psi_0(\cdot,\cdot)$, γ and η
  for $k = 0, 1, 2, \ldots, K-1$ **do** ▷ main loop
    for each $(x,a) \in \mathcal{Z}$ **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 for
    for each $(x,a) \in \mathcal{Z}$ **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 for
  end for
  for each$(x,a) \in \mathcal{Z}$ **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 for
  return π_K

2.3.2 Performance Guarantee

In this subsection, we investigate the finite-iteration and asymptotic behavior of Algorithm 2.1. We begin by proving a finite-iteration performance guarantee for DPP:

Theorem 2.1 (The ℓ_∞-norm performance loss bound of DPP)**.** *Let Assumption 2.1 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 $k \geq 0$:*

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

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 $\mathcal{M}$ are replaced with the greedy policy and the max-operator $\mathcal{M}$, respectively. Therefore, for $\eta = +\infty$ the DPP recursion is re-expressed as

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

We must point out that the choice of $\eta < +\infty$ may be still useful in the presence of function approximation, where the greedy update rule can be unstable due to the non-differentiability of the max operator. In fact, our numerical results in Subsection 2.5.2 suggests that the performance of DPP in the presence of function approximation is optimized for some finite value of η rather than $\eta = +\infty$ (see Subsection 2.5.2 for more details).

As an immediate consequence of Theorem 2.1, we obtain the following result:

Corollary 2.1. *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 2.2. *Let Assumption 2.1 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 2.1). Also, the fact that in Corollary 2.2 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.

2.4 Dynamic Policy Programming with Approximation

Algorithm 2.1 (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.

2.4.1 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:

$$\varepsilon_k(x,a) \triangleq \Psi_{k+1}(x,a) - \mathcal{O}\Psi_k(x,a), \qquad \forall (x,a) \in \mathcal{Z}. \tag{2.7}$$

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:

$$\begin{aligned}\Psi_{k+1}(x,a) &= \mathcal{O}\Psi_k(x,a) + \varepsilon_k(x,a) \\ &= \Psi_k(x,a) + r(x,a) + \gamma P\mathcal{M}_\eta\Psi_k(x,a) - \mathcal{M}_\eta\Psi_k(x,a) + \varepsilon_k(x,a) \\ &= \Psi_k(x,a) + \mathcal{T}^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x,a) + \varepsilon_k(x,a),\end{aligned} \tag{2.8}$$

where π_k is given by (2.5).

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 2.2 (Finite-iteration performance loss bound of approximate DPP)**.** *Let Assumption 2.1 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* (2.7) *and the accumulated error E_k as*

$$E_k(x,a) \triangleq \sum_{j=0}^{k} \varepsilon_j(x,a), \qquad \forall (x,a) \in \mathcal{Z}. \tag{2.9}$$

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

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

Proof See Appendix C. ■

Taking the upper-limit yields corollary 2.3.

Corollary 2.3 (Asymptotic performance-loss bound of approximate DPP)**.** *Define $\bar{\varepsilon} \triangleq \limsup_{k\to\infty} \|E_k\| / (k+1)$. Then, the following inequality holds:*

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2} \bar{\varepsilon}. \tag{2.10}$$

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 $\varepsilon_{\max} = \limsup_{k\to\infty} \|\varepsilon_k\|$. The difference is that in (2.10) 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 $\{\varepsilon_0, \varepsilon_1, \varepsilon_2, \dots\}$ are some i.i.d. zero-mean random variables bounded by $0 < U < \infty$. Corollary 2.3 combined with the law of large numbers then leads to the following asymptotic bound for approximate DPP:

$$\limsup_{k\to\infty} \|Q^* - Q^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)^2}\bar{\varepsilon} = 0, \qquad \text{w.p. (with probability) } 1, \tag{2.11}$$

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 2.2. *The i.i.d. assumption may be replaced by some weaker and more realistic assumption that only requires the error sequence* $\{\varepsilon_0, \varepsilon_1, \dots, \varepsilon_k\}$ *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 2.3).*

2.4.2 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 (2.6). 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 $P(\cdot|x,a)$:

$$\Psi_{k+1}(x,a) \triangleq \Psi_k(x,a) + \mathcal{T}_k^{\pi_k}\Psi_k(x,a) - \pi_k\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}. \quad (2.12)$$

Based on (2.12), 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 (2.7), 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:

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

The DPP-RL update rule can then be considered as a special case of the more general approximate DPP update rule of Equation (2.8).

Equation (2.12) is just an approximation of the DPP update rule (2.6). Therefore, the convergence result of Corollary 2.1 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 (2.6) are asymptotically *averaged out* by DPP-RL, as postulated by Corollary 2.3. To prove this result we need the following lemma, which bounds the estimation error ε_k.

Lemma 2.1 (Boundedness of ε_k)**.** *Let Assumption 2.1 hold and assume that the initial action-preference function Ψ_0 is uniformly bounded by $V_{\max}$, then we have, for all $k \geq 0$,*

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

Proof See Appendix D. ■

Lemma 2.1 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 $\varepsilon_k = \mathcal{T}^{\pi_k}\Psi_k - \mathcal{T}_k^{\pi_k}\Psi_k$ 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 2.3 (Asymptotic convergence of DPP-RL)**.** *Let Assumption 2.1 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 2.2:

Theorem 2.4 (Finite-time high-probability loss-bound of DPP-RL)**.** *Let Assumption 2.1 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*

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

Proof See Appendix D.2. ■

Theorem 2.2 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 2.4 w.r.t. η which leads to the following corollary:

Corollary 2.4. *Let Assumption 2.1 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 $\mathcal{M}$:

$$\Psi_{k+1}(x,a) := \Psi_k(x,a) + \mathcal{T}_k\Psi_k(x,a) - \mathcal{M}\Psi_k(x), \qquad \forall (x,a) \in \mathcal{Z}, \quad (2.13)$$

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 2.2.

Algorithm 2.2 (DPP-RL) Reinforcement learning with DPP

Require: 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
  for each $(x,a) \in \mathcal{Z}$ **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 for
  for each $x \in \mathcal{X}$ **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 for
end for
return π_K

Furthermore, the following PAC bound which determines the number of steps k required to achieve the error $\varepsilon > 0$ in estimating the optimal policy, w.p. $1-\delta$, is an immediate consequence of Theorem 2.4.

Corollary 2.5 (Finite-time PAC bound of DPP-RL)**.** *Let Assumption 2.1 hold. Then, for any* $\varepsilon > 0$*, after*

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

steps of Algorithm 2.2, the uniform approximation error $\|Q^* - Q^{\pi_k}\| \leq \varepsilon$*, w. p.* $1-\delta$*.*

2.4.3 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 $\mathcal{F}_\varphi = \{\varphi_1, \ldots, \varphi_m\}$, where each $\varphi_i : \mathcal{Z} \to \mathbb{R}$ 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 $\{\varphi_i\}_{i=1:m}$ 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:

$$J_k(\theta; \Psi) \triangleq \left\| \theta^\mathsf{T}\Phi - \mathcal{O}\Psi_k \right\|_{2,\mu}^2, \tag{2.14}$$

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

$$\theta_{k+1} = \arg\min J_k(\theta; \Psi) = \left[\mathbb{E}\left(\Phi\Phi^\mathsf{T}\right)\right]^{-1}\mathbb{E}(\Phi\mathcal{O}\Psi_k), \tag{2.15}$$

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, 2007b, 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

$$\tilde{\theta}_{k+1} = \left[\sum_{n=1}^{N}\Phi(X_n, A_n)\Phi(X_n, A_n)^\mathsf{T} + \alpha N\mathbf{I}\right]^{-1}\sum_{n=1}^{N}\mathcal{O}_n\Psi_k\Phi(X_n, A_n), \tag{2.16}$$

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 (2.7).

Algorithm 2.3 presents the *sampling-based approximate dynamic policy programming* (SADPP) in which we rely on (2.16) to approximate DPP operator at each iteration.

Algorithm 2.3 (SADPP) Sampling-based approximate dynamic policy programming

Require: $\tilde{\theta}_0$, 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 $P(\cdot|\cdot)$
  for each $n = 1, 2, 3, \ldots, N$ **do**
    for each $a \in \mathcal{A}$ **do** ▷ compute Ψ_k for every action of states X_n, Y_n
      $\Psi_k(X_n, a) = \tilde{\theta}_k^\intercal \Phi(X_n, a)$
      $\Psi_k(Y_n, a) = \tilde{\theta}_k^\intercal \Phi(Y_n, a)$
    end for
    $\mathcal{M}_\eta \Psi_k(X_n) = \sum_{a \in \mathcal{A}} \frac{\exp(\eta \Psi_k(X_n,a)) \Psi_k(X_n,a)}{\sum_{b \in \mathcal{A}} \exp \eta \Psi_k(X_n,b)}$
    $\mathcal{M}_\eta \Psi_k(Y_n) = \sum_{a \in \mathcal{A}} \frac{\exp(\eta \Psi_k(Y_n,a)) \Psi_k(Y_n,a)}{\sum_{b \in \mathcal{A}} \exp \eta \Psi_k(Y_n,b)}$ ▷ soft-max $\mathcal{M}_\eta \Psi_k$ for X_n and Y_n
  $\mathcal{O}_n \Psi_k = \Psi_k(X_n, A_n) - r(X_n, A_n) - \gamma(\mathcal{M}_\eta \Psi_k)(Y_n) + (\mathcal{M}_\eta \Psi_k)(X_n)$ ▷ empirical DPP operator
  end for
  $\tilde{\theta}_{k+1} = \left[\sum_{n=1}^N \Phi(X_n, A_n)\Phi(X_n, A_n)^\intercal + \alpha N \mathbf{I}\right]^{-1} \sum_{n=1}^N \mathcal{O}_n \Psi_k \Phi(X_n, A_n)$ ▷ SADPP update rule
end for
return $\tilde{\theta}_K$

2.5 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 2.2) 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

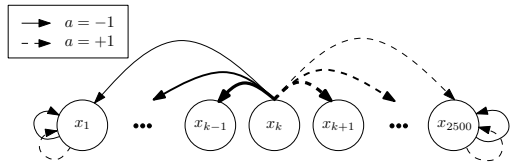

Figure 2.1: Linear MDP: Illustration of the linear MDP problem. Nodes indicate states. States x_1 and x_{2500} are the two absorbing states and state x_k is an example of interior state. Arrows indicate possible transitions of **these three nodes only**. From x_k any other node is reachable with transition probability (arrow thickness) proportional to the inverse of the distance to x_k (see the text for details).

investigate the finite-time performance of SADPP (Algorithm 2.3) 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.[d]

2.5.1 DPP-RL

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

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 2.1). 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

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

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 2.2).

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 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

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

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

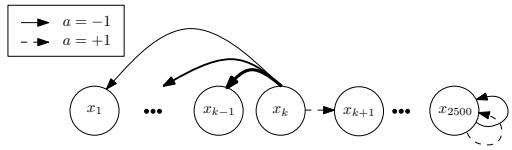

Figure 2.2: Combination lock: illustration of the combination lock MDP problem. Nodes indicate states. State x_{2500} is the goal (absorbing) state and state x_k is an example of interior state. Arrows indicate possible transitions of **these two nodes only**. From x_k any previous state is reachable 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).

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 $1/\|c_x - c_y\|_2$.

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.

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

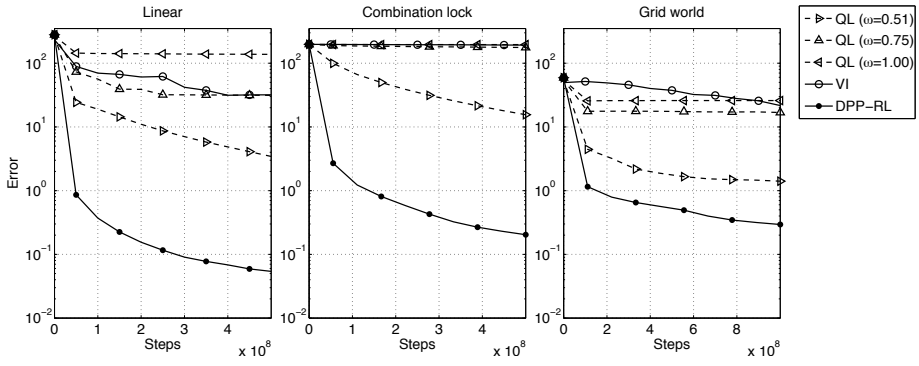

Figure 2.3: Comparison between DPP-RL, QL and VI in terms of **number of steps**, defined as the number of iterations times the number of computations per iteration of the particular algorithm. Each plot shows the averaged error of the induced policies over 50 different runs (see the text for details).

accumulated reward, which is usually used when the RL algorithm learns from a stream of samples.

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 2.3 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 im-

provement 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 2.3, 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 result.

Table 2.1 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 2.3 and 2.2, DPP-RL manages to average out the simulation noise caused by sampling and converges to a near optimal solution, which is very robust.

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.

2.5.2 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.

Table 2.1: Comparison between DPP-RL, QL and VI given a fixed computational and sampling budget. Table 2.1 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)

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. (2007) 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).

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:

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

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:

$$a^*(x)=\begin{cases}0 & \text{if } x\in[0,\bar{x}]\\ 1 & \text{if } x>\bar{x}\end{cases}. \tag{2.17}$$

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

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

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 control in the optimal replacement problem (see Equation 2.17). We discretize the state space in $K=100$ and compute the error as follows:

$$\text{Error}=\frac{1}{K}\sum_{k=1}^{K}|a^*(x_k)-\widehat{a}(x_k)|, \tag{2.18}$$

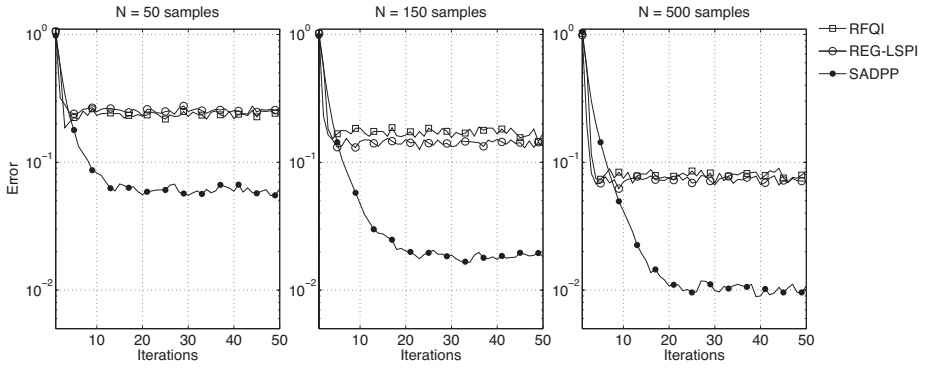

Figure 2.4: Numerical results for the optimal replacement problem. Each plot shows the error of RFQI, REG-LSPI and SADPP for certain number of samples N. Error is defined as in Equation (2.18) and averaged over 200 repetitions (see the text for details).

where $\widehat{a}$ 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 $\widehat{a}$ for SADPP by choosing the most probable action from the induced soft-max policy, and then use this to compute (2.18). 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); Farias and Roy (2000).

We are interested in the behavior of the error as a function of the iteration

Table 2.2: 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)

number for different number of samples N per iteration. Figure 2.4 and Table 2.2 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 2.2) 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 2.4. We can also conclude that, for our choice of settings, SADPP outperforms RFQI and REG-LSPI.

2.6 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 2.1, 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; Szepesvari 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 2.4 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 2.5.1 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 2.4.2, 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., 2011b, sec. 3.3.1).[e] 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.

Very recently, Azar et al. (2011b) 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 (2005b) 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 require Bellman-like iterations. Likewise, our formalism is superficially similar to PoWER (Kober and Peters,

[e]Note that Even-Dar and Mansour (2003) make use of a slightly different performance measure than the one we use in this chapter: 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 chapter 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.

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.

2.7 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 2.4). At the same time, we admit that the existence of an additional parameter η favors SADPP since SADPP performs best for a finite-value of η . 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., 2010b; 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., 2007; Munos, 2005). This would require extending the error propagation result of Theorem 2.2 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 (2.6), 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.

CHAPTER 3

Speedy Q-Learning

We consider the problem of model-free reinforcement learning (RL) in the Markovian decision processes (MDP) under the probably approximately correct (PAC) model. We introduce a new variant of Q-learning, called speedy Q-learning (SQL), to address the problem of the slow convergence in the standard Q-learning algorithm, and prove PAC bounds on the performance of this algorithm. The bounds indicate that for any MDP with n state-action pairs and discount factor $\gamma \in [0, 1)$, a total number of $O\big(n \log(n)/((1-\gamma)^4 \varepsilon^2)\big)$ steps suffices for SQL to converge to an ε-optimal action-value function with high probability. Our results have better dependencies on ε and $1-\gamma$ (the same dependency on n), and thus, are tighter than the best available results for Q-learning. The SQL algorithm also improves on existing results for the batch Q-value iteration, in terms of the computational budget required to achieve a near optimal solution.[a]

[a]This chapter is based on (Azar et al., 2011b) and (Azar et al., 2012a).

3.1 Introduction

Finding an optimal policy for a Markovian decision process (MDP) is a classical problem in the fields of operations research and decision theory. When an explicit model of an MDP (i.e., transition probability and reward functions) is known, one can rely on dynamic programming (DP) (Bellman, 1957) algorithms such as value iteration or policy iteration (see, e.g., Bertsekas, 2007a; Puterman, 1994a) to compute an optimal policy of the MDP. Value iteration algorithm computes the optimal action-value function Q^* by successive iterations of the Bellman operator $\mathcal{T}$ (will be defined in Section 3.2). One can show that in the discounted infinite-horizon setting the convergence of value iteration is exponentially fast, since the Bellman operator $\mathcal{T}$ is a contraction mapping (Bertsekas, 2007b) on the action-value function Q. However, value iteration rely on an explicit knowledge of the MDP. In many real world problems the transition probabilities are not initially known, but one may observe transition samples using Monte-Carlo sampling, either as a single trajectory obtained by following an exploration policy (a rollout), or by simulating independent transition samples in the state(-action) space using a generative model (simulator) of the dynamical system. The field of reinforcement learning (RL) is concerned with the problem of finding an optimal policy or the optimal value function from the observed reward and transition samples (Sutton and Barto, 1998; Szepesvári, 2010).

One may characterize RL methods as model-based or model-free. In model-based RL, we first learn a model of the MDP and then use it to approximate value functions using DP techniques. In contrary, model-free methods compute an approximation of a value function by making use of a sample-based estimate of the Bellman operator without resorting to learning an explicit model of the dynamical system. Q-learning (QL) is a well-known model-free RL algorithm that incrementally finds an estimate of the optimal action-value function (Watkins, 1989). The QL algorithm can be seen as a combination of the value iteration algorithm and stochastic approximation, where at each time step k a new estimate of the optimal action-value function for all state-action pairs (x, a) is calculated using the following update rule:[b]

$$\begin{aligned} Q_{k+1}(x,a) &= (1-\alpha_k)Q_k(x,a) + \alpha_k \mathcal{T}_k Q_k(x,a) \\ &= (1-\alpha_k)Q_k(x,a) + \alpha_k \left(\mathcal{T}Q_k(x,a) - \varepsilon_k(x,a)\right), \end{aligned}$$

[b] In this section, for the sake of simplicity in the notation, we assume that the action-values of all state-action pairs are updated in parallel. Note that, in general, this assumption is not required for the proof of convergence of Q-learning.

where $\varepsilon_k(x,a) = \mathcal{T}_k Q_k(x,a) - \mathcal{T}Q_k(x,a)$, $\mathcal{T}_k Q_k(x,a)$ is the empirical estimation of Bellman operator and α_k is the learning step. One may show, using an induction argument, that for the choice of linear learning step, i.e., $\alpha_k = \frac{1}{k+1}$, Q_{k+1} can be seen the average of the estimates of Bellman operator throughout the learning process:

$$Q_{k+1}(x,a) = \frac{1}{k+1}\sum_{j=0}^{k}(\mathcal{T}Q_j(x,a) - \varepsilon_j(x,a)).$$

It is not then difficult to prove, using a law of large number argument, that the term $1/(k+1)\sum_{j=0}^{k}\varepsilon_j$, is asymptotically averaged out, and thus, for $k \to \infty$ the update rule of QL becomes equivalent to $Q_{k+1} = 1/(k+1)\sum_{j=0}^{k}\mathcal{T}Q_j$. The problem with this result is that the rate of convergence of the recursion $Q_{k+1} = 1/(k+1)\sum_{j=0}^{k}\mathcal{T}Q_j$ to Q^* is significantly slower than the original Bellman recursion $Q_{k+1} = \mathcal{T}Q_k$. In fact, one can prove that the asymptotic rate of convergence of QL with linear learning step is of order $\tilde{O}(1/k^{1-\gamma})$ (Szepesvári, 1997),[c] which in the case of γ close to 1 makes its convergence extremely slower than the standard value iteration algorithm, which enjoys a fast convergence rate of order $\tilde{O}(\gamma^k)$. This slow rate of convergence, i.e., high sample complexity, may explain why the practitioners often prefer the batch RL methods, such as approximate value iteration (AVI) (Bertsekas, 2007b), to QL despite the fact that QL has better memory requirements than the batch RL methods.

In this chapter, we focus on RL problems that are formulated as finite state-action discounted infinite-horizon MDPs and propose a new algorithm, called *speedy Q-learning* (SQL), to address the problem of slow convergence of Q-learning. The main idea is to modify the update rule of Q-learning such that, at each iteration k, the new estimate of action-value function Q_{k+1} closely follows the Bellman operator $\mathcal{T}Q_k$. This guarantees that the rate of convergence of SQL, unlike QL, is close to the fast rate of convergence of the value iteration algorithm. At each time step k, SQL uses two successive estimates of the bellman operator $\mathcal{T}Q_k$ and $\mathcal{T}Q_{k-1}$ to update the action-value function

$$Q_{k+1}(x,a) = (1-\alpha_k)Q_k(x,a) + \alpha_k\big[k\mathcal{T}Q_k(x,a) - (k-1)\mathcal{T}Q_{k-1}(x,a) - \varepsilon_k(x,a)\big], \tag{3.1}$$

which makes its space complexity twice as QL. However, this allows SQL to achieve a significantly faster rate of convergence than QL, since it reduces

[c] The notation $g = \tilde{O}(f)$ implies that there are constants c_1 and c_2 such that $g \leq c_1 f \log^{c_2}(f)$.

the dependency on the previous Bellman operators from the average $1/(k+1)\sum_{j=0}^{k}\mathcal{T}Q_j$ (in the case of QL) to only $\mathcal{T}Q_k + O\big(1/(k+1)\big)$, with the choice of $\alpha_k = 1/(k+1)$:

$$\begin{aligned} Q_{k+1}(x,a) &= (1-\alpha_k)Q_k(x,a) \\ &\quad + \alpha_k\big[k\mathcal{T}Q_k(x,a) - (k-1)\mathcal{T}Q_{k-1}(x,a) - \varepsilon_k(x,a)\big] \\ &= \frac{1}{k+1}\sum_{j=0}^{k}\big(j\mathcal{T}Q_j(x,a) - (j-1)\mathcal{T}Q_{j-1}(x,a) - \varepsilon_j(x,a)\big) \\ &= \mathcal{T}Q_k(x,a) + \frac{1}{k+1}(\mathcal{T}Q_{-1}(x,a) - \mathcal{T}Q_k(x,a)) \\ &\quad - \frac{1}{k+1}\sum_{j=0}^{k}\varepsilon_j(x,a), \end{aligned}$$

where in the second line we rely on an induction argument. This shows that similar to QL, the iterates of SQL are expressed in terms of the average estimation error, and thus, the SQL update rule asymptotically averages out the sampling errors. However, SQL has the advantage that at each time step k the iterate Q_{k+1} closely follows (up to a factor of $O(1/(k+1))$ the latest Bellman iterate $\mathcal{T}Q_k$ instead of the average $1/(k+1)\sum_{j=0}^{k}\mathcal{T}Q_j$ in the case of QL. As a result, unlike QL, it does not suffer from the slow convergence due to slow down in the value iteration process (see Section 3.3.3 for a detailed comparison of QL and SQL's convergence rates).

The idea of using previous estimates of the action-values has already been employed in order to improve the performance of QL. A popular algorithm of this kind is Q(λ) (Watkins, 1989; Peng and Williams, 1996), which incorporates the concept of eligibility traces in QL, and has been empirically shown to have a better performance than QL, i.e., Q(0), for suitable values of λ. Another recent work in this direction is *Double Q-learning* (van Hasselt, 2010), which uses two estimators for the action-value function in order to alleviate the over-estimation of action-values in QL. This over-estimation is caused by a positive bias introduced by using the maximum action-value as an approximation for the maximum expected action-value.

The rest of the chapter is organized as follows. After introducing the notation used in the chapter in Section 3.2, we present our *Speedy Q-learning* algorithm in Section 3.3. We first describe the synchronous and asynchronous versions of the algorithm in Section 3.3.1, then state our main theoretical result, i.e.,

high-probability bounds on the performance of SQL in Section 3.3.2, and finally compare our bound with the previous results on QL and Q-value iteration in Section 3.3.3. In Section 3.4, we numerically evaluate the performance of SQL on different problems. Section 3.5 contains the detailed proofs of the results of Section 3.3.2. Finally, we conclude the chapter and discuss some future directions in Section 3.6.

3.2 Preliminaries

In this section, we introduce some concepts, definitions, and notation from the Markov decision processes (MDPs) theory and stochastic processes that are used throughout the chapter. We start by the definition of supremum norm (ℓ_∞-norm). For a real-valued function $g : \mathcal{Y} \mapsto \mathbb{R}$, where $\mathcal{Y}$ is a finite set, the supremum norm of g is defined as $\|g\| \triangleq \max_{y\in\mathcal{Y}} |g(y)|$.

We consider the standard reinforcement learning (RL) framework (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998) in which a learning agent interacts with a stochastic environment and this interaction is modeled as a discrete-time discounted MDP. A discounted MDP is a quintuple $(\mathcal{X}, \mathcal{A}, P, \mathcal{R}, \gamma)$, where $\mathcal{X}$ and $\mathcal{A}$ are the set of states and actions, P is the state transition distribution, $\mathcal{R}$ is the reward function, and $\gamma \in (0, 1)$ is a discount factor. We denote the *effective* horizon of MDP by β defined as $\beta = 1/(1-\gamma)$. We also denote by $P(\cdot|x, a)$ and $r(x, a)$ the probability distribution over the next state and the immediate reward of taking action a at state x, respectively.[d] To keep the representation succinct, we use $\mathcal{Z}$ for the joint state-action space $\mathcal{X} \times \mathcal{A}$.

Assumption 3.1 (MDP regularity)**.** *We assume that the joint state-action set $\mathcal{Z}$ is finite with cardinality n, and the immediate rewards $r(x, a)$ are in the interval $[0, 1]$.*[e]

A policy π determines the distribution of the control action given the past observations. A policy is called *stationary* if the distribution depends only on the last state x and is *deterministic* if it assigns a unique action to each state $x \in \mathcal{X}$. The *value* and the *action-value functions* of a policy π, denoted respectively by $V^\pi : \mathcal{X} \mapsto \mathbb{R}$ and $Q^\pi : \mathcal{Z} \mapsto \mathbb{R}$, are defined as the expected

[d] For the sake of simplicity in notation, here we assume that the reward $r(x, a)$ is a deterministic function of state-action pairs (x, a). It is straightforward to extend our results to the case of stochastic rewards under some mild assumptions, e.g., boundedness of the absolute value of the rewards.

[e] Our results also hold if the rewards are taken from some interval $[r_{\min}, r_{\max}]$ instead of $[0, 1]$, in which case the bounds scale with the factor $r_{\max} - r_{\min}$.

sum of discounted rewards that are encountered when the policy π is executed. Given a MDP, the goal is to find a policy that attains the best possible values, $V^*(x) \triangleq \sup_\pi V^\pi(x),\ \forall x \in \mathcal{X}$. Function V^* is called the *optimal value function*. Similarly the *optimal action-value function* is defined as $Q^*(x,a) = \sup_\pi Q^\pi(x,a),\ \forall (x,a) \in \mathcal{Z}$. The optimal action-value function Q^* is the unique fixed-point of the *Bellman optimality operator* $\mathcal{T}$ defined as

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

where $\mathcal{M}$ is the max operator over action-value functions and is defined as $(\mathcal{M}Q)(y) = \max_{a \in \mathcal{A}} Q(y,a),\ \forall y \in \mathcal{X}$.[f] We now define the cover time of MDP under the policy π as follows:

Definition 3.1 (Cover Time)**.** *Let π be a policy over a finite stat-action MDP and $t \geq 0$ be an integer. Define $\tau_\pi(x,t)$ to be the number of time-steps between t and the first future time that all state-action pairs $z \in \mathcal{Z}$ are visited (the MDP is covered) starting from state $x \in \mathcal{X}$ at time-step t and following π. The state-action space $\mathcal{Z}$ is* covered *by the policy π if all the state-action pairs are visited at least once under the policy π.*

The following assumption that bounds the expected cover time of the MDP guarantees that asymptotically all the state-action pairs are visited infinitely many times under the policy π.

Assumption 3.2 (Boundedness of the expected cover time)**.** *Let $0 < L < \infty$ and t be an integer. We assume that under the policy π, for all $x \in \mathcal{X}$ and $t > 0$, we have*

$$\mathbb{E}\big(\tau_\pi(x,t)\big) \leq L.$$

3.3 Speedy Q-Learning

In this section, we introduce a new RL algorithm, called *speedy Q-Learning* (SQL), derive performance bounds for its synchronous and asynchronous variants, and compare these bounds with similar results on standard Q-learning (QL).

[f] It is important to note that $\mathcal{T}$ is a γ-contraction mapping w.r.t. to the ℓ_∞-norm, i.e., for any pair of action-value functions Q and Q', we have $\|\mathcal{T}Q - \mathcal{T}Q'\| \leq \gamma \|Q - Q'\|$ (Bertsekas, 2007b, Chap. 1).

3.3.1 Synchronous and Asynchronous SQL Algorithms

In this subsection, we introduce two variants of the SQL algorithm, synchronous SQL and asynchronous SQL. In the asynchronous version, at each time step, the action-value of only one state-action pair, the current observed state-action, is updated, while the action-values of the rest of the state-action pairs remain unchanged. For the convergence of this instance of the algorithm, it is required that all the states and actions are visited infinitely many times, which makes the analysis slightly more complicated. On the other hand, having access to a simulator that can generate samples anywhere in the state-action space, the algorithm may be formulated in a synchronous fashion, in which we first generate a next state $y \sim P(\cdot|x,a)$ for each state-action pair (x,a), and then update the action-values of all the state-action pairs using these samples. The pseudo-code of the synchronous and asynchronous versions of SQL are shown in Algorithms 3.1 and 3.2, respectively. It is possible to show that asynchronous SQL is reduced to synchronous SQL when the cover time $\tau_\pi(x,t) = n$ for all $x \in \mathcal{X}$ and $t \geq 0$. In this case, the action-values of all state-action pairs are updated in a row. In other words, Algorithm 3.1 may be seen as a special case of Algorithm 3.2. Therefore, in the sequel we only describe the more general asynchronous SQL algorithm.

Algorithm 3.1 Synchronous Speedy Q-learning

Require: Initial action-values Q_0, discount factor γ, and number of steps T

$Q_{-1} := Q_0$ ▷ Initalization
$t := k := 0$
repeat ▷ Main loop
  $\alpha_k := \dfrac{1}{k+1}$
  for each $(x,a) \in \mathcal{Z}$ **do** ▷ Update the action-value function for all $(x,a) \in \mathcal{Z}$
    Generate the next state sample $y_k \sim P(\cdot|x,a)$
    $\mathcal{T}_k Q_{k-1}(x,a) := r(x,a) + \gamma \mathcal{M} Q_{k-1}(y_k)$
    $\mathcal{T}_k Q_k(x,a) := r(x,a) + \gamma \mathcal{M} Q_k(y_k)$
    $Q_{k+1}(x,a) := (1-\alpha_k) Q_k(x,a) + \alpha_k \big(k \mathcal{T}_k Q_k(x,a) - (k-1) \mathcal{T}_k Q_{k-1}(x,a)\big)$ ▷ SQL update rule
    $t := t+1$
  end for
  $k := k+1$
until $t \geq T$
return Q_k

As it can be seen from the update rule of Algorithm 3.2, at each time step, the algorithm keeps track of the action-value functions of the two most recent

Algorithm 3.2 Asynchronous Speedy Q-learning

Require: Initial action-values Q_0, policy π, discount factor γ, number of step T, and initial state X_0

$t := k := 0$ ▷ Initialization
$\alpha_0 = 1$
for each $(x, a) \in \mathcal{Z}$ **do** $Q_{-1}(x, a) := Q_0(x, a)$ $N_0(x, a) := 0$
end for
repeat ▷ Main loop
  Draw the action $A_t \sim \pi(\cdot|X_t)$
  Generate the next state sample $y_k \sim P(\cdot|X_t, A_t)$
  $\eta_N := \dfrac{1}{N_k(X_t, A_t) + 1}$
  $\mathcal{T}_k Q_{k-1}(X_t, A_t) := (1 - \eta_N)\mathcal{T}_k Q_{k-1}(X_t, A_t) + \eta_N \big(r(X_t, A_t) + \gamma \mathcal{M} Q_{k-1}(y_k)\big)$
  $\mathcal{T}_k Q_k(X_t, A_t) := (1 - \eta_N)\mathcal{T}_k Q_k(X_t, A_t) + \eta_N \big(r(X_t, A_t) + \gamma \mathcal{M} Q_k(y_k)\big)$
  $Q_{k+1}(X_t, A_t) := (1 - \alpha_k)Q_k(X_t, A_t) + \alpha_k \big(k\mathcal{T}_k Q_k(X_t, A_t) - (k - 1)\mathcal{T}_k Q_{k-1}(X_t, A_t)\big)$ ▷ SQL update rule
  $N_k(X_t, A_t) := N_k(X_t, A_t) + 1$
  $X_{t+1} = y_k$
  if $\min_{(x,a)\in\mathcal{Z}} N_k(x, a) > 0$ **then** ▷ Check if $\forall (x, a) \in \mathcal{Z}$ have been visited at round k
    $k := k + 1$
    $\alpha_k := \dfrac{1}{k + 1}$
    for each $(x, a) \in \mathcal{Z}$ **do** $N_k(x, a) := 0$
    end for
  end if $t := t + 1$
until $t \geq T$
return Q_k

iterations Q_k and Q_{k-1}, and its main update rule is of the following form at time step t and iteration k:

$$\begin{aligned} Q_{k+1}(X_t, A_t) = (1 - \alpha_k)Q_k(X_t, A_t) \\ + \alpha_k \big(k\mathcal{T}_k Q_k(X_t, A_t) - (k - 1)\mathcal{T}_k Q_{k-1}(X_t, A_t)\big), \end{aligned} \tag{3.2}$$

where $\mathcal{T}_k Q(X_t, A_t) = 1/|\mathcal{Y}_k| \sum_{y\in\mathcal{Y}_k} \big[r(X_t, A_t) + \gamma \mathcal{M} Q(y)\big]$ is the empirical Bellman optimality operator using the set of next state samples $\mathcal{Y}_k$, where $\mathcal{Y}_k$ is a short-hand notation for $\mathcal{Y}_{k,t}(x, a)$, the set of all samples generated up to time step t in round k by taking action a in state x. At each time step t, Algorithm 3.2 works as follows: **(i)** it simulates the MDP for one-step at state X_t, i.e., it first draws the action $A_t \in \mathcal{A}$ from the distribution $\pi(\cdot|X_t)$ and then makes a transition to a new state $y_k \sim P(\cdot|X_t, A_t)$, **(ii)** it updates the two sample estimates $\mathcal{T}_k Q_{k-1}(X_t, A_t)$ and $\mathcal{T}_k Q_k(X_t, A_t)$ of the Bellman optimality operator

applied to the estimates Q_{k-1} and Q_k of the action-value function at the previous and current rounds $k-1$ and k, for the state-action pair (X_t, A_t) using the next state y_k, **(iii)** it updates the action-value function of (X_t, A_t), generates $Q_{k+1}(X_t, A_t)$, using the update rule of Eq. 3.2, **(iv)** it checks if all $(x, a) \in \mathcal{Z}$ have been visited at least once at iteration k, and if this condition is satisfied, we move to the next round $k+1$, and finally, **(v)** we replace X_{t+1} with y_k and repeat the whole process until $t \geq T$. Moreover, we let α_k decays linearly with the number of iterations k, i.e., $\alpha_k = 1/(k+1)$. Note that the update rule $\mathcal{T}_k Q_k(X_t, A_t) := (1-\eta_N)\mathcal{T}_k Q_k(X_t, A_t) + \eta_N(r(X_t, A_t) + \gamma \mathcal{M} Q_k(y_k))$ is used to incrementally generate an unbiased estimate of $\mathcal{T} Q_k$.

3.3.2 Main Theoretical Results

The main theoretical results of this chapter are expressed as high-probability bounds for the performance of both synchronous and asynchronous versions of the SQL algorithms. [g]

Theorem 3.1 (Performance Bound of Synchronous SQL)**.** *Let Assumption 4.1 hold and Q_T be the estimate of Q^* generated by Algorithm 3.1 after T steps. Then, with probability at least $1-\delta$, we have*

$$\|Q^* - Q_T\| \leq \beta^2 \left[\frac{\gamma n}{T} + \sqrt{\frac{2n \log \frac{2n}{\delta}}{T}} \right].$$

Theorem 3.2 (Performance Bound of Asynchronous SQL)**.** *Let Assumption 4.1 and Assumption 3.2 hold and Q_T be the estimate of Q^* generated by Algorithm 3.2 after T steps. Then, with probability at least $1-\delta$, we have*

$$\|Q^* - Q_T\| \leq \beta^2 \left[\frac{\gamma e L \log \frac{2}{\delta}}{T} + \sqrt{\frac{2eL \log \frac{2}{\delta} \log \frac{4n}{\delta}}{T}} \right].$$

These results, combined with the Borel-Cantelli lemma (Feller, 1968), guarantee that Q_T converges almost surely to Q^* with the rate $\sqrt{1/T}$ for both Algorithms 3.1 and 3.2. Moreover, the PAC bounds of Corollaries 3.1 and 3.2, which quantify the number of steps T required to reach the error $\varepsilon > 0$ in estimating the optimal action-value function w.p. $1-\delta$, are immediate consequences of Theorems 3.1 and 3.2, respectively.

[g] We report the detailed proofs in Section 3.5.

Corollary 3.1 (Finite-time PAC Bound of Synchronous SQL)**.** *Under Assumption 4.1, after*

$$T = \lceil \frac{4\ n\ \beta^4\ \log \frac{2n}{\delta}}{\varepsilon^2} \rceil$$

steps (transitions), the uniform approximation error of Algorithm 3.1 is small, i.e., $\|Q^* - Q_T\| \leq \varepsilon$*, with probability at least* $1 - \delta$*.*[h]

Corollary 3.2 (Finite-time PAC Bound of Asynchronous SQL)**.** *Under Assumption 4.1 and Assumption 3.2, after*

$$T = \lceil \frac{4\ e\ L\ \beta^4\ \log \frac{2}{\delta}\ \log \frac{4n}{\delta}}{\varepsilon^2} \rceil$$

steps (transitions), the uniform approximation error of Algorithm 3.2 is small, i.e., $\|Q^* - Q_T\| \leq \varepsilon$*, with probability at least* $1 - \delta$*.*

3.3.3 Relation to the Existing Results

In this section, we first compare our results for the SQL algorithm with the existing results on the convergence of the standard Q-learning. The comparison indicates that SQL accelerates the convergence of QL, especially for large values of β and small values of α. We then compare SQL with batch Q-value iteration (QVI) in terms of the sample and computational complexities, i.e., the number of samples and the number of time units [i] required to achieve an ε-optimal solution with high probability, as well as space complexity, i.e., the memory required at each step of the algorithm.

A Comparison with the Convergence Rate of the Standard Q-Learning

There are not many studies in the literature concerning the convergence rate of incremental model-free RL algorithms such as QL. Szepesvári (1997) provided the asymptotic convergence rate for QL under the assumption that all the states have the same next state distribution. This result shows that the asymptotic convergence rate of QL with a linearly decaying learning step has exponential dependency on β, i.e. $T = \tilde{O}(1/\varepsilon^{\beta})$.

Even-Dar and Mansour (2003) investigated the finite-time behavior of synchronous QL for different time scales. Their main result indicates that by using

[h] For every real number u, $\lceil u \rceil$ is defined as the smallest integer number not less than u.

[i] In the sequel, we consider the CPU time required to compute a single-sample estimate of the Bellman optimality operator as the time unit.

the polynomial learning step $\alpha_k = 1/(k+1)^{\omega}$, $0.5 < \omega < 1$, synchronous QL achieves ε-optimal performance w.p. at least $1-\delta$ after

$$T = O\left(n\left[\left(\frac{\beta^4 \log\frac{n\beta}{\delta\varepsilon}}{\varepsilon^2}\right)^{\frac{1}{w}} + \left(\beta \log\frac{\beta}{\varepsilon}\right)^{\frac{1}{1-\omega}}\right]\right), \tag{3.3}$$

steps, where the time-scale parameter ω may be tuned to achieve the best performance. When $\gamma \approx 1$, the horizon $\beta = 1/(1-\gamma)$ becomes the dominant term in the bound of Eq. 3.3, and thus, the bound is optimized by finding an ω that minimizes the dependency on β. This leads to the optimized bound of order $\tilde{O}\left(\beta^5/\varepsilon^{2.5}\right)$ with the choice of $\omega = 0.8$. On the other hand, SQL is guaranteed to achieve the same precision with only $O\left(\beta^4/\varepsilon^2\right)$ steps. The difference between these two bounds is substantial for large β^2/ε.

Even-Dar and Mansour (2003) also proved bounds for the asynchronous variant of Q-learning in the case that the cover time of MDP can be uniformly bounded from above by some finite constant. The extension of their results to the more realistic case that the expected value of the cover-time is bounded by some $L > 0$ (Assumption Assumption 3.2) leads to the following PAC bound:

Proposition 3.1 (Even-Dar and Mansour, 2003)**.** *Under Assumption 4.1 and Assumption 3.2, for all* $\omega \in (0.5, 1)$*, after*

$$T = O\left(\left[\frac{\left(L\log\frac{1}{\delta}\right)^{1+3\omega}\beta^4\log\frac{n\beta}{\delta\varepsilon}}{\varepsilon^2}\right]^{\frac{1}{w}} + \left[L\beta\log\frac{1}{\delta}\log\frac{\beta}{\varepsilon}\right]^{\frac{1}{1-\omega}}\right)$$

steps (transitions), the approximation error of asynchronous QL $\|Q^* - Q_T\| \leq \varepsilon$*, w.p. at least* $1-\delta$*.*

The dependence on L in this algorithm is of order $O(L^{3+\frac{1}{\omega}} + L^{\frac{1}{1-\omega}})$, which with the choice of $\omega \approx 0.77$ leads to the optimized dependency of order $O(L^{4.34})$, whereas asynchronous SQL achieves the same accuracy after just $O(L)$ steps. This result indicates that for MDPs with large expected cover-time, i.e., slow-mixing MDPs, asynchronous SQL may converge substantially faster to a near-optimal solution than its QL counterpart.

SQL vs. Q-Value Iteration

Finite sample bounds for both model-based and model-free (Phased Q-learning) QVI have been derived in (Kearns and Singh, 1999; Even-Dar et al., 2002;

Kakade, 2004, chap. 9.1). These algorithms can be considered as the batch version of Q-learning. They show that to quantify ε-optimal action-value functions with high probability, we need $\tilde{O}\big(n\beta^5/\varepsilon^2\big)$ and $\tilde{O}\big(n\beta^4/\varepsilon^2\big)$ samples in model-free and model-based QVI, respectively.[j] A comparison between their results and the main result of this chapter suggests that the sample complexity of SQL, which is of order $\tilde{O}\big(n\beta^4/\varepsilon^2\big)$, is better than model-free QVI in terms of β. Although the sample complexities of SQL and model-based QVI are of the same order, SQL has a significantly better computational and space complexity than model-based QVI: SQL needs only $2n$ memory space, while the space complexity of model-based QVI is $\min\big(\tilde{O}(n\beta^4/\varepsilon^2), n(|\mathcal{X}|+1)\big)$(Kearns and Singh, 1999). SQL also improves the computational complexity by a factor of $\tilde{O}(\beta)$ compared to both model-free and model-based QVI.[k] Table 3.1 summarizes the comparisons between SQL and the RL methods discussed in this section.

Methods	SQL	Q-learning	Model-based QVI	Model-free QVI
SC	$\tilde{O}\left(\frac{n\beta^4}{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{n\beta^5}{\varepsilon^{2.5}}\right)$	$\tilde{O}\left(\frac{n\beta^4}{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{n\beta^5}{\varepsilon^2}\right)$
CC	$\tilde{O}\left(\frac{n\beta^4}{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{n\beta^5}{\varepsilon^{2.5}}\right)$	$\tilde{O}\left(\frac{n\beta^5}{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{n\beta^5}{\varepsilon^2}\right)$
SPC	$\Theta(n)$	$\Theta(n)$	$\min\left(\tilde{O}\left(\frac{n\beta^4}{\varepsilon^2}\right), O(n(\lvert\mathcal{X}\rvert))\right)$	$\Theta(n)$

Table 3.1: Comparison between SQL, Q-learning, model-based, and model-free Q-value iteration (QVI) in terms of sample complexity (SC), computational complexity (CC), and space complexity (SPC).

3.4 Experiments

In this section, we empirically evaluate the performance of the synchronous SQL (Algorithm 3.1) on the discrete state-action problems, which we considered in

[j] For the sake of simplicity, here we ignore the logarithmic dependencies of the bounds.

[k] Since SQL performs only one Q-value update per sample, its sample and computational complexities of are of the same order. The same argument also applies to the standard Q-learning. On the other hand, in the case of model-based QVI, the algorithm needs to iterate the action-value function of all the state-action pairs at least $\tilde{O}(\beta)$ times. This leads to a computational complexity of order $\tilde{O}(n\beta^5/\varepsilon^2)$ given that only $\tilde{O}(n\beta^4/\varepsilon^2)$ entries of the estimated transition matrix are non-zero.

Section 2.5.1. We also examine the convergence of these algorithms and compare it with Q-learning and model-based Q-value iteration (QVI) (Kearns and Singh, 1999). The source code of all the algorithms is available at
`http://www.mbfys.ru.nl/~mazar/Research_Top.html`.

3.4.1 Experimental Setup and Results

We now describe our experimental setting. The convergence behavior of SQL is compared to two other algorithms: the Q-learning algorithm of Even-Dar and Mansour (2003) (QL) and the model-based Q-value iteration (QVI) of Kearns and Singh (1999). QVI is a batch RL algorithm that first estimates the model using the whole data set and then performs value iteration on the learned model.

All the algorithms are evaluated in terms of ℓ_∞-norm performance loss of the action-value function $\|Q^* - Q_T\|$ at time-step T. We choose this performance measure in order to be consistent with the performance measure used in Section 3.3.2. The optimal action-value function Q^* is computed with high accuracy using value iteration. We consider QL with polynomial learning step $\alpha_k = 1/(k+1)^\omega$ where $\omega \in \{0.51, 0.6, 0.8\}$ and the linear learning step $\alpha_k = 1/(k+1)$. Note that ω needs to be larger than 0.5, otherwise QL may diverge (see Even-Dar and Mansour, 2003, for the proof).

To have a fair comparison of the three algorithms, since each algorithm requires different number of computations per iteration, we fix the total computational budget of the algorithms to the same value for each benchmark. The computation time is constrained to 30 seconds for the linear MDP and combination lock problems. For the grid world, which has twice as many actions as the other benchmarks, the maximum running time is fixed to 60 seconds. We also fix the total number of samples, per state-action, to 10^5 samples for all problems and algorithms. Smaller number of samples leads to a dramatic decrease in the quality of the solutions of all the three algorithms. Algorithms were implemented as MEX files (in C++) and ran on a Intel core i5 processor with 8 GB of memory. CPU time was computed using the system function `times()` that provides process-specific CPU time. Randomization was implemented using `gsl_rng_uniform()` function of the GSL library, which is superior to the standard `rand()`.[1] Sampling time, which is the same for all the algorithms, were not included in the CPU time. At the beginning of every run **(i)** the action-value functions are randomly initialized in the interval $[-\beta, \beta]$, and **(ii)**

[1] http://www.gnu.org/s/gsl.

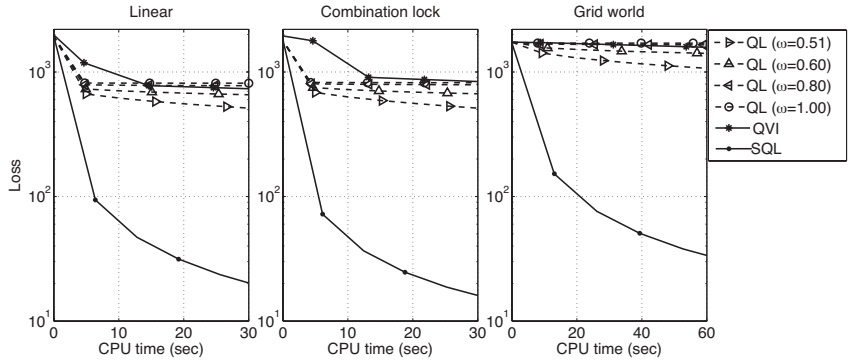

Figure 3.1: A comparison between SQL, QL, and QVI. Each plot compares the performance loss of the policies induced by the algorithms at one of the three problems considered in this section. All the results are averaged over 50 different runs.

a new set of samples is generated from $P(\cdot|x,a)$ for all $(x,a) \in \mathcal{Z}$. The corresponding results are computed after a small fixed amount of iterations. All the results are averaged over 50 different runs.

Figure 3.1 shows the performance-loss in terms of the elapsed CPU time for the three problems and algorithms with the choice of $\beta = 1000$. We observe that SQL outperforms QL and QVI in all the three problems. It achieves a reasonable performance very rapidly, just in a few seconds. The minimum and maximum errors are attained for the combination lock and grid world problems, respectively. We also observe that the difference between the final outcome of SQL and QL (the second best method) is significant, about 30 times, in all domains.

Figures 3.2 and 3.3 show the means and standard deviations of the final performance-loss as a function of the horizon β. We observe that for large values of β, i.e. $\beta \geq 100$, SQL outperforms other methods by more than an order of magnitude in terms of both mean and standard deviation of performance loss. SQL performs slightly worse than QVI for $\beta \leq 10$. However, the loss of QVI scales worse than SQL with β, e.g., for $\beta = 1000$, SQL has about two order of magnitude advantage over QVI. QL performs better for larger values of ω when the horizon β is small, whereas for large values of β smaller ω's are more preferable.

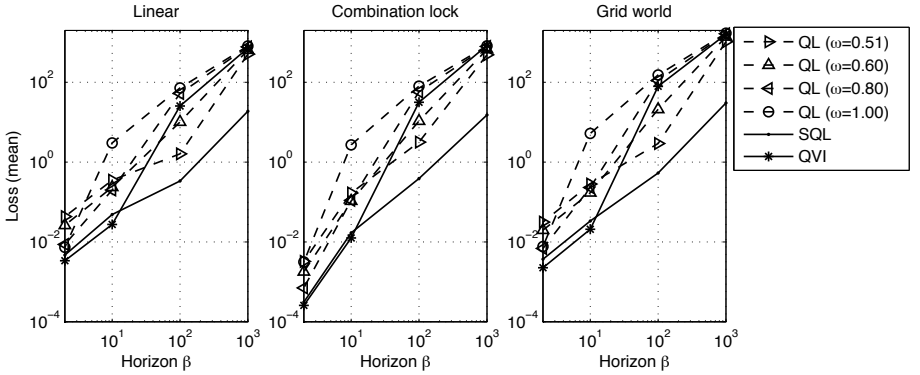

Figure 3.2: A comparison between SQL, QL, and QVI given a fixed computational and sampling budget. The plot shows the means of the final performance of the algorithms in terms of the horizon β. All the results are averaged over 50 different runs.

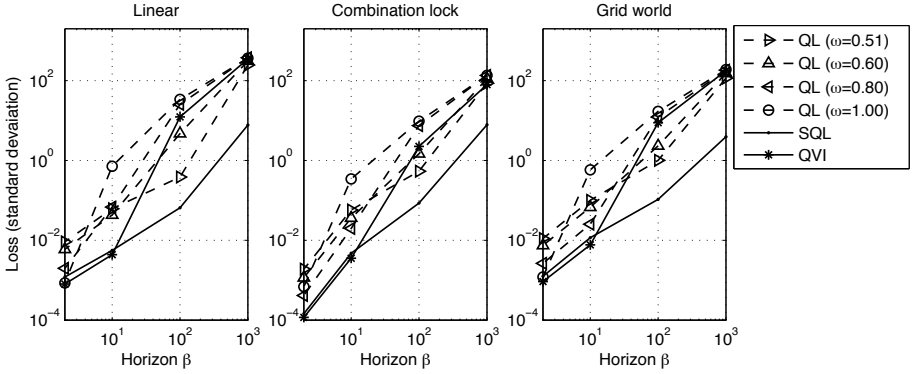

Figure 3.3: A comparison between SQL, QL, and QVI given a fixed computational and sampling budget. The plot shows the the standard deviations of the final performance of the algorithms in terms of the horizon β. All the results are averaged over 50 different runs.

These results are consistent with the performance bound of Theorem 3.1 and indicate that the SQL algorithm manages to average out the simulation noise caused by sampling, and converges rapidly to a near optimal solution, which is

robust in comparison to the other algorithms. Moreover, we may conclude that SQL significantly improves the computational complexity of learning w.r.t. the standard QL and QVI in the three problems studied in this section.

3.5 Analysis

In this section, we give some intuition about the convergence of asynchronous variant of SQL and provide the full proof of the finite-time analysis reported in Theorems 3.1 and 3.2. We start by introducing some notation.

Let $\mathcal{Y}^k$ be the set of all samples drawn at round k of the SQL algorithms and $\mathcal{F}_k$ be the filtration generated by the sequence $\{\mathcal{Y}^0, \mathcal{Y}^1, \ldots, \mathcal{Y}^k\}$. Note that for all $(x,a) \in \mathcal{Z}$, the update rule of Equation 3.2 may be rewritten in the following more compact form

$$Q_{k+1}(x,a) = (1-\alpha_k)Q_k(x,a) + \alpha_k \mathcal{D}_k[Q_k, Q_{k-1}](x,a),$$

where $\mathcal{D}_k[Q_k, Q_{k-1}](x,a) \triangleq \frac{1}{\alpha_k}\big[(1-\alpha_k)\mathcal{T}_k Q_k(x,a) - (1-2\alpha_k)\mathcal{T}_k Q_{k-1}(x,a)\big]$ and $\alpha_k = 1/(k+1)$. We now define the operator $\mathcal{D}[Q_k, Q_{k-1}]$ as the expected value of the empirical operator $\mathcal{D}_k$ conditioned on the filtration $\mathcal{F}_{k-1}$, i.e.,

$$\begin{aligned}\mathcal{D}[Q_k, Q_{k-1}](x,a) &\triangleq \mathbb{E}\big(\mathcal{D}_k[Q_k, Q_{k-1}](x,a) \big| \mathcal{F}_{k-1}\big) \\ &= \frac{1-\alpha_k}{\alpha_k}\mathcal{T}Q_k(x,a) - \frac{1-2\alpha_k}{\alpha_k}\mathcal{T}Q_{k-1}(x,a),\end{aligned}$$

where the last equality follows from the fact that in both Algorithms 3.1 and 3.2, $\mathcal{T}_k Q_k(x,a)$ and $\mathcal{T}_k Q_{k-1}(x,a)$ are unbiased empirical estimates of the Bellman optimality operators $\mathcal{T}Q_k(x,a)$ and $\mathcal{T}Q_{k-1}(x,a)$, respectively. Thus, the update rule of SQL can be rewritten as

$$Q_{k+1}(x,a) = (1-\alpha_k)Q_k(x,a) + \alpha_k\big(\mathcal{D}[Q_k, Q_{k-1}](x,a) - \varepsilon_k(x,a)\big), \qquad (3.4)$$

where the estimation error ε_k is defined as the difference between the operator $\mathcal{D}[Q_k, Q_{k-1}]$ and its sample estimate $\mathcal{D}_k[Q_k, Q_{k-1}]$, i.e.,

$$\varepsilon_k(x,a) \triangleq \mathcal{D}[Q_k, Q_{k-1}](x,a) - \mathcal{D}_k[Q_k, Q_{k-1}](x,a), \qquad \forall (x,a) \in \mathcal{Z}.$$

We have the property that $\mathbb{E}\big[\varepsilon_k(x,a)|\mathcal{F}_{k-1}\big] = 0$, which means that for all $(x,a) \in \mathcal{Z}$, the sequence of estimation errors $\big\{\varepsilon_1(x,a), \varepsilon_2(x,a), \ldots, \varepsilon_k(x,a)\big\}$ is

a martingale difference sequence w.r.t. the filtration $\mathcal{F}_k$. Finally, we define the martingale $E_k(x,a)$ to be the sum of the estimation errors, i.e.,

$$E_k(x,a) \triangleq \sum_{j=0}^{k} \varepsilon_j(x,a), \qquad \forall (x,a) \in \mathcal{Z}. \tag{3.5}$$

The following steps lead to the proof of Theorems 3.1 and 3.2: **(i)** Lemma 3.1 shows the stability of SQL, i.e., the sequence of Q_k's stays bounded in SQL. **(ii)** Lemma 3.2 states the key property that each iterate Q_{k+1} in SQL is close to the Bellman operator applied to the previous iterate Q_k, i.e., $\mathcal{T}Q_k$. More precisely, in this lemma we show that Q_{k+1} is equal to $\mathcal{T}Q_k$ plus an estimation error term of order E_k/k. **(iii)** Lemma 3.3 provides a performance bound on $\|Q^* - Q_k\|$ in terms of a discounted sum of the cumulative estimation errors $\{E_j\}_{j=0}^{k-1}$. Lemma 3.1 to 3.3 hold for both Algorithms 3.1 and 3.2. **(iv)** Given these results, we prove Theorem 3.1 using a maximal Azuma's inequality stated in Lemma 3.2. **(v)** Finally, we extend this proof to asynchronous SQL, and prove Therorem 3.2, using the result of Lemma 3.5.

For simplicity of the notation, we often remove the dependence on (x,a), e.g., writing Q for $Q(x,a)$ and E_k for $E_k(x,a)$. Also note that for all $k \geq 0$, the following relations hold between α_k and α_{k+1} in Algorithms 3.1 and 3.2:

$$\alpha_{k+1} = \frac{\alpha_k}{\alpha_k + 1} \qquad \text{and} \qquad \alpha_k = \frac{\alpha_{k+1}}{1 - \alpha_{k+1}}.$$

Lemma 3.1 (Stability of SQL)**.** *Let Assumption 4.1 hold and assume that the initial action-value function $Q_0 = Q_{-1}$ is uniformly bounded by $V_{\max} = \beta$, then we have*

$$\|Q_k\| \leq V_{\max}, \quad \|\varepsilon_k\| \leq V_{\max}, \quad \textit{and} \quad \|\mathcal{D}_k[Q_k, Q_{k-1}]\| \leq V_{\max} \quad \forall k \geq 0.$$

Proof

We first prove that $\|\mathcal{D}_k[Q_k, Q_{k-1}]\| \leq V_{\max}$ by induction. For $k = 0$ we have

$$\|\mathcal{D}_0[Q_0, Q_{-1}]\| = \|\mathcal{T}_0 Q_{-1}\| \leq \|r\| + \gamma \|\mathcal{M} Q_{-1}\| \leq R_{\max} + \gamma V_{\max} = V_{\max}.$$

Now let us assume that for any $k \geq 0$, $\|\mathcal{D}_k[Q_k, Q_{k-1}]\| \leq V_{\max}$. Then we obtain

$$
\begin{aligned}
\|\mathcal{D}_{k+1}[Q_{k+1},Q_k]\| &= \left\| \tfrac{1-\alpha_{k+1}}{\alpha_{k+1}} \mathcal{T}_{k+1} Q_{k+1} - \tfrac{1-2\alpha_{k+1}}{\alpha_{k+1}} \mathcal{T}_{k+1} Q_k \right\| \\
&\le \left\| \left(\tfrac{1-\alpha_{k+1}}{\alpha_{k+1}} - \tfrac{1-2\alpha_{k+1}}{\alpha_{k+1}} \right) r \right\| \\
&\quad + \gamma \left\| \tfrac{1-\alpha_{k+1}}{\alpha_{k+1}} \mathcal{M} Q_{k+1} - \tfrac{1-2\alpha_{k+1}}{\alpha_{k+1}} \mathcal{M} Q_k \right\| \\
&\le \|r\| + \gamma \left\| \tfrac{1-\alpha_{k+1}}{\alpha_{k+1}} \mathcal{M}\big((1-\alpha_k)Q_k + \alpha_k \mathcal{D}_k[Q_k,Q_{k-1}]\big) \right. \\
&\quad \left. - \tfrac{1-2\alpha_{k+1}}{\alpha_{k+1}} \mathcal{M} Q_k \right\| \\
&= \|r\| + \gamma \left\| \mathcal{M}\left(\tfrac{1-\alpha_k}{\alpha_k} Q_k + \mathcal{D}_k[Q_k,Q_{k-1}] \right) - \tfrac{1-\alpha_k}{\alpha_k} \mathcal{M} Q_k \right\| \\
&\le \|r\| + \gamma \left\| \mathcal{M}\left(\tfrac{1-\alpha_k}{\alpha_k} Q_k + \mathcal{D}_k[Q_k,Q_{k-1}] - \tfrac{1-\alpha_k}{\alpha_k} Q_k \right) \right\| \\
&\le \|r\| + \gamma \, \|\mathcal{D}_k[Q_k,Q_{k-1}]\| \le R_{\max} + \gamma V_{\max} = V_{\max},
\end{aligned}
$$

and thus by induction, we deduce that for all $k \ge 0$, $\|\mathcal{D}_k[Q_k,Q_{k-1}]\| \le V_{\max}$.

The bound on ε_k follows from

$$
\|\varepsilon_k\| = \|\mathbb{E}\big(\mathcal{D}_k[Q_k,Q_{k-1}]|\mathcal{F}_{k-1}\big) - \mathcal{D}_k[Q_k,Q_{k-1}]\| \le V_{\max},
$$

and the bound $\|Q_k\| \le V_{\max}$ is deduced by the fact that

$$
Q_k = \frac{1}{k} \sum\nolimits_{j=0}^{k-1} \mathcal{D}_j[Q_j,Q_{j-1}].
$$

■

The next lemma shows that Q_k is close to $\mathcal{T}Q_{k-1}$, up to a $O(\frac{1}{k})$ term minus the cumulative estimation error $\frac{1}{k}E_{k-1}$.

Lemma 3.2. *Under Assumption 4.1, for any $k \ge 1$ we have*

$$
Q_k = \mathcal{T}Q_{k-1} + \frac{1}{k}\left(\mathcal{T}Q_0 - \mathcal{T}Q_{k-1} - E_{k-1}\right). \tag{3.6}
$$

Proof

We prove this result by induction. The result holds for $k = 1$, where Equation 3.6 reduces to Equation 3.4. We now show that if Equation 3.6 holds for $k \ge 1$ then it also holds for $k+1$. Assume that Equation 3.6 holds for k, then from Equation 3.4 we have

$$
\begin{aligned}
Q_{k+1} &= (1-\alpha_k)Q_k + \alpha_k\left[\frac{1-\alpha_k}{\alpha_k}\mathcal{T}Q_k - \frac{1-2\alpha_k}{\alpha_k}\mathcal{T}Q_{k-1} - \varepsilon_k\right] \\
&= (1-\alpha_k)\left[\mathcal{T}Q_{k-1} + \alpha_{k-1}\left(\mathcal{T}Q_0 - \mathcal{T}Q_{k-1} - E_{k-1}\right)\right] \\
&\quad + \alpha_k\left[\frac{1-\alpha_k}{\alpha_k}\mathcal{T}Q_k - \frac{1-2\alpha_k}{\alpha_k}\mathcal{T}Q_{k-1} - \varepsilon_k\right] \\
&= (1-\alpha_k)\left[\mathcal{T}Q_{k-1} + \frac{\alpha_k}{1-\alpha_k}\left(\mathcal{T}Q_0 - \mathcal{T}Q_{k-1} - E_{k-1}\right)\right] \\
&\quad + (1-\alpha_k)\mathcal{T}Q_k - (1-2\alpha_k)\mathcal{T}Q_{k-1} - \alpha_k\varepsilon_k \\
&= (1-\alpha_k)\mathcal{T}Q_k + \alpha_k\left(\mathcal{T}Q_0 - E_{k-1} - \varepsilon_k\right) = \mathcal{T}Q_k + \alpha_k\left(\mathcal{T}Q_0 - \mathcal{T}Q_k - E_k\right) \\
&= \mathcal{T}Q_k + \frac{1}{k+1}\left(\mathcal{T}Q_0 - \mathcal{T}Q_k - E_k\right).
\end{aligned}
$$

Thus Equation 3.6 holds for $k+1$, and as a result, holds for all $k \geq 1$. ■

Now we bound the difference between Q^* and Q_k in terms of the discounted sum of the cumulative estimation errors $\{E_0, E_1, \ldots, E_{k-1}\}$.

Lemma 3.3 (Error propagation in SQL)**.** *Let Assumption 4.1 hold and assume that the initial action-value function $Q_0 = Q_{-1}$ is uniformly bounded by $V_{\max} = \beta$, then for all $k \geq 1$, we have*

$$
\begin{aligned}
\|Q^* - Q_k\| &\leq \frac{1}{k}\left[\gamma\beta^2 + \sum_{j=1}^{k}\gamma^{k-j}\|E_{j-1}\|\right] && (3.7)\\
&\leq \frac{\beta}{k}\left[\gamma\beta + \max_{j=1:k}\|E_{j-1}\|\right]. && (3.8)
\end{aligned}
$$

Proof

For any sequence of cumulative errors $\{E_0, E_1, \ldots, E_{k-1}\}$, we have $\sum_{j=1}^{k}\gamma^{k-j}\|E_{j-1}\| \leq \beta\max_{j=1:k}\|E_{j-1}\|$. Thus, we only need to prove (3.7), and (3.8) will automatically follow. We again prove this lemma by induction. The result holds for $k = 1$ since we have

$$
\begin{aligned}
\|Q^* - Q_1\| &= \|\mathcal{T}Q^* - \mathcal{T}Q_0 - \varepsilon_0\| \leq \gamma\|Q^* - Q_0\| + \|\varepsilon_0\| \\
&\leq 2\gamma V_{\max} + \|\varepsilon_0\| \leq \gamma\beta^2 + \|E_0\|.
\end{aligned}
$$

Note that the first equality follows from Lemma 3.2. We now show that if the bound holds for k, then it should also hold for $k+1$. If (3.7) holds for k, then using Lemma 3.2 we have

$$\begin{aligned}
\left\|Q^* - Q_{k+1}\right\| &= \left\|Q^* - \mathcal{T}Q_k - \frac{1}{k+1}\left(\mathcal{T}Q_0 - \mathcal{T}Q_k - E_k\right)\right\| \\
&\leq \left\|\alpha_k(\mathcal{T}Q^* - \mathcal{T}Q_0) + (1-\alpha_k)(\mathcal{T}Q^* - \mathcal{T}Q_k)\right\| + \alpha_k \left\|E_k\right\| \\
&\leq \alpha_k \left\|\mathcal{T}Q^* - \mathcal{T}Q_0\right\| + (1-\alpha_k)\left\|\mathcal{T}Q^* - \mathcal{T}Q_k\right\| + \alpha_k \left\|E_k\right\| \\
&\leq \gamma\alpha_k V_{\max} + \gamma(1-\alpha_k)\left\|Q^* - Q_k\right\| + \alpha_k \left\|E_k\right\| .
\end{aligned}$$

Since we assumed that (3.7) holds for k, we may write

$$\begin{aligned}
\left\|Q^* - Q_{k+1}\right\| &\leq \alpha_k\gamma V_{\max} + \gamma(1-\alpha_k)\alpha_{k-1}\left[\gamma\beta^2 + \sum_{j=1}^{k}\gamma^{k-j}\left\|E_{j-1}\right\|\right] \\
&\quad + \alpha_k\left\|E_k\right\| \\
&= \gamma\beta\alpha_k + \gamma^2\beta^2\alpha_k + \gamma\alpha_k\sum_{j=1}^{k}\gamma^{k-j}\left\|E_{j-1}\right\| + \alpha_k\left\|E_k\right\| \\
&= \frac{1}{k+1}\left[\gamma\beta^2 + \sum_{j=1}^{k+1}\gamma^{k+1-j}\left\|E_{j-1}\right\|\right] .
\end{aligned}$$

Thus, Equation 3.7 holds for $k+1$, and as a result by induction, it holds for all $k \geq 1$.

■

Before stating the next lemma, we report the maximal Azuma-Hoeffding's inequality (see e.g., Cesa-Bianchi and Lugosi 2006), which is used in the proof of this lemma.

Proposition 3.2 (Maximal Azuma-Hoeffding's Inequality)**.** *Let* $\mathcal{V} = \{V_1, V_2, \ldots, V_k\}$ *be a martingale difference sequence w.r.t. a sequence of random variables* $\{X_1, X_2, \ldots, X_k\}$*, i.e.,* $\mathbb{E}(V_{j+1}|X_1, \ldots, X_j) = 0$ *for all* $0 < j \leq k$*, such that* $\mathcal{V}$ *is uniformly bounded by* $L > 0$*. If we define* $S_k = \sum_{i=1}^{k} V_i$*, then for any* $\varepsilon > 0$*, we have*

$$\mathbb{P}\left(\max_{j=1:k} S_j > \varepsilon\right) \leq \exp\left(\frac{-\varepsilon^2}{2kL^2}\right).$$

We now state Lemma 3.4 in which we prove a high probability bound on the estimation error term in Lemma 3.3, i.e., $\max_{j=1:k} \|E_{j-1}\|$.

Lemma 3.4. *Let Assumption 4.1 hold and assume that the initial action-value function $Q_0 = Q_{-1}$ is uniformly bounded by $V_{\max} = \beta$. Then for all $k \geq 1$, with probability at least $1 - \delta$, we have*

$$\max_{j=1:k} \|E_{j-1}\| \leq \beta\sqrt{2k \log \frac{2n}{\delta}}. \tag{3.9}$$

Proof

We begin by providing a high probability bound on $\max_{1\leq j\leq k} |E_{j-1}(x,a)|$ for a given state-action pair (x,a). Note that

$$\begin{aligned}
&\mathbb{P}\left(\max_{j=1:k} |E_{j-1}(x,a)| > \varepsilon\right) \\
=&\mathbb{P}\left(\max\left[\max_{j=1:k} E_{j-1}(x,a)\ ,\ \max_{j=1:k}\big(-E_{j-1}(x,a)\big)\right] > \varepsilon\right) \\
=&\mathbb{P}\left(\left\{\max_{j=1:k} E_{j-1}(x,a) > \varepsilon\right\} \bigcup \left\{\max_{j=1:k}\big(-E_{j-1}(x,a)\big) > \varepsilon\right\}\right) \\
\leq&\mathbb{P}\left(\max_{j=1:k} E_{j-1}(x,a) > \varepsilon\right) + \mathbb{P}\left(\max_{j=1:k}\big(-E_{j-1}(x,a)\big) > \varepsilon\right),
\end{aligned} \tag{3.10}$$

We can now bound both terms in (3.10) using the maximal Azuma-Hoeffding's inequality stated in Proposition 3.2. As mentioned earlier, the sequence of random variables $\{\varepsilon_0(x,a), \varepsilon_1(x,a), \ldots, \varepsilon_k(x,a)\}$ is martingale difference w.r.t. the filtration $\mathcal{F}_k$ generated by random samples $\{y_0, y_1, \ldots, y_k\}$
(x,a) for all (x,a), i.e., $\mathbb{E}[\varepsilon_k(x,a)|\mathcal{F}_{k-1}] = 0$. So, we have

$$\begin{aligned}
\mathbb{P}\left(\max_{j=1:k} E_{j-1}(x,a) > \varepsilon\right) &\leq \exp\left(\frac{-\varepsilon^2}{2kV_{\max}^2}\right), \\
\mathbb{P}\left(\max_{j=1:k}\big(-E_{j-1}(x,a)\big) > \varepsilon\right) &\leq \exp\left(\frac{-\varepsilon^2}{2kV_{\max}^2}\right).
\end{aligned} \tag{3.11}$$

Combining (3.11) with (3.10), we deduce

$$\mathbb{P}\left(\max_{j=1:k}|E_{j-1}(x,a)|>\varepsilon\right)\leq 2\exp\left(\frac{-\varepsilon^2}{2kV_{\max}^2}\right),$$

and then by a union bound over the state-action space, we obtain

$$\mathbb{P}\left(\max_{j=1:k}\|E_{j-1}\|>\varepsilon\right)\leq 2n\exp\left(\frac{-\varepsilon^2}{2kV_{\max}^2}\right). \tag{3.12}$$

Equation 3.12 may be rewritten for any $\delta>0$ as

$$\mathbb{P}\left(\max_{j=1:k}\|E_{j-1}\|\leq V_{\max}\sqrt{2k\log\frac{2n}{\delta}}\right)\geq 1-\delta,$$

which concludes the proof. ■

Proof of Theorem 3.1

The result of the theorem follows by plugging Equation 3.9 into Equation 3.8, and taking into account that if $n(k-1)<T\leq nk$ then Algorithm 3.1 stops after k iterations and returns Q_k. ■

For the proof of Theorem 3.2, we rely on the following lemma which bounds the number of steps required to visit all state-action pairs k times with high probability.

Lemma 3.5. *Under Assumption 3.2, from any initial state x_0 and for any integer $k>0$, after running Algorithm 3.2 for $T=ekL\log\frac{1}{\delta}$ steps, the state-action space $\mathcal{Z}$ is covered at least k times under the policy π with probability at least $1-\delta$.*

Proof

For any state $x_0\in\mathcal{X}$ and time $t>0$, we define a random variable $\mathcal{Q}_k$ as the number of steps required to cover the MDP k times starting from x_0 at time t. Using Markov inequality (Feller, 1968), for any x_0 and t, we can bound $\mathcal{Q}_k$ with high probability as

$$\mathbb{P}(\mathcal{Q}_k>ekL)\leq\frac{\mathbb{E}\left(\mathcal{Q}_k\right)}{ekL}\leq\frac{k\sup_{t>0}\max_{x\in\mathcal{X}}\mathbb{E}\big(\tau_\pi(x,t)\big)}{ekL}\leq\frac{kL}{ekL}=\frac{1}{e}.$$

This mean that after a run of length ekL, the probability that the entire state-action space is not covered at least k times is less than $\frac{1}{e}$. The fact

that the bound holds for any initial state and time implies that after $m > 0$ intervals of length ekL, the chance of not covering the MDP k times is less than $\frac{1}{e^m}$, i.e., $\mathbb{P}(\mathcal{Q}_k > mekL) \leq \frac{1}{e^m}$. With the choice of $m = \log \frac{1}{\delta}$, we obtain $\mathbb{P}(\mathcal{Q}_k > ekL \log \frac{1}{\delta}) \leq \delta$. The bound then can be rewritten as

$$\mathbb{P}\left(\mathcal{Q}_k \leq ekL \log \frac{1}{\delta}\right) \geq 1 - \delta,$$

which concludes the proof. ■

Proof of Theorem 3.2

Plugging the results of Lemmas 3.5 and 3.4 into Equation 3.8 (each holds with probability at least $1 - \delta'$, with $\delta' = \delta/2$) concludes the proof of the theorem. ■

3.6 Conclusions and Future Work

In this chapter, we presented a new reinforcement learning (RL) algorithm, called speedy Q-learning (SQL). We analyzed the finite-time behavior of this algorithm as well as its asymptotic convergence to the optimal action-value function. Our results are in the form of high probability bounds on the performance loss of SQL, which suggest that the algorithm converges to the optimal action-value function in a faster rate than the standard Q-learning. The numerical experiments in Section 3.4 confirm our theoretical results showing that for large value of $\beta = 1/(1 - \gamma)$, SQL outperforms the standard Q-learning by a wide margin. Overall, SQL is a simple, efficient, and theoretically well-founded RL algorithm that improves on the existing similar methods such as Q-learning and sample-based value iteration.

In this work, we are only interested in the estimation of the optimal action-value function and not the problem of exploration. Therefore, we did not compare our results to PAC-MDP methods (Strehl et al., 2009; Szita and Szepesvári, 2010) and upper-confidence bound based algorithms (Bartlett and Tewari, 2009; Jaksch et al., 2010b), in which the choice of the exploration policy has an influence on the behavior of the learning algorithm. However, we believe that it would be possible to gain w.r.t. the state of the art in PAC-MDPs by combining the asynchronous version of SQL with a smart exploration strategy. This is mainly due to the fact that the bound for SQL has been proved to be tighter

than the RL algorithms used for estimating the value function in PAC-MDP methods (especially in the model-free case). Also, SQL has a better computational requirement in comparison to the standard RL methods. We leave this as a subject for future work.

Another possible direction for future work is to scale up SQL to large, possibly continuous, state and action spaces, where function approximation is needed. We believe that it would be possible to extend the current SQL analysis to the continuous case along the same line as in the fitted value iteration analysis by Antos et al. (2007) and Munos and Szepesvári (2008).

CHAPTER 4

Minimax Bounds on the Sample Complexity of RL

We consider the problem of learning the optimal action-value function in discounted-reward Markov decision processes (MDPs). We prove new PAC bounds on the sample-complexity of two well-known model-based reinforcement learning (RL) algorithms in the presence of a generative model of the MDP: value iteration and policy iteration. The first result indicates that for an MDP with N state-action pairs and the discount factor $\gamma \in [0, 1)$ only $O\big(N \log(N/\delta)/\big((1-\gamma)^3\varepsilon^2\big)\big)$ state-transition samples are required to find an ε-optimal estimation of the action-value function with the probability (w.p.) $1-\delta$. Further, we prove that, for small values of ε, an order of $O\big(N \log(N/\delta)/\big((1-\gamma)^3\varepsilon^2\big)\big)$ samples is required to find an ε-optimal policy w.p. $1-\delta$. We also prove a matching lower bound of $\Theta\big(N \log(N/\delta)/\big((1-\gamma)^3\varepsilon^2\big)\big)$ on the sample complexity of estimating the optimal action-value function. To the best of our knowledge, this is the first minimax result on the sample complexity of RL: the upper bound matches the lower bound in terms of N, ε, δ and $1/(1-\gamma)$ up to a constant factor. Also, both our lower bound and upper bound improve on the state-of-the-art in terms of their dependence on $1/(1-\gamma)$.[a]

[a] This chapter is based on (Azar et al., 2012c) and (Azar et al., 2012b).

4.1 Introduction

An important problem in the field of reinforcement learning (RL) is to estimate the optimal policy (or the optimal value function) from the observed rewards and the transition samples (Sutton and Barto, 1998; Szepesvári, 2010). To estimate the optimal policy one may use model-free or model-based approaches. In model-based RL, we first learn a model of the MDP using a batch of state-transition samples and then use this model to estimate the optimal policy or the optimal action-value function using the Bellman recursion, whereas model-free methods directly aim at estimating the optimal value function without resorting to learning an explicit model of the dynamical system. The fact that the model-based RL methods decouple the model-estimation problem from the value (policy) iteration problem may be useful in problems with a limited budget of sampling. This is because the model-based RL algorithms, after learning the model, can perform many Bellman recursion steps without any need to make new transition samples, whilst the model-free RL algorithms usually need to generate fresh samples at each step of value (policy) iteration process.

The focus of this chapter is on model-based RL algorithms for finite state-action problems, where we have access to a generative model (simulator) of the MDP. Especially, we derive tight sample-complexity upper bounds for two well-known model-based RL algorithms, the model-based value iteration and the model-based policy iteration (Wiering and van Otterlo, 2012a), It has been shown (Kearns and Singh, 1999; Kakade, 2004, chap. 9.1) that an action-value based variant of model-based value iteration algorithm, Q-value iteration (QVI), finds an ε-optimal estimate of the action-value function with high probability (w.h.p.) using only $\widetilde{O}(N/\big((1-\gamma)^4\varepsilon^2\big))$ samples, where N and γ denote the size of state-action space and the discount factor, respectively.[b] One can also prove, using the result of Singh and Yee (1994), that QVI w.h.p. finds an ε-optimal policy using an order of $\widetilde{O}(N/\big((1-\gamma)^6\varepsilon^2\big))$ samples. An upper-bound of a same order can be proven for model-based PI. These results match the best upper-bound currently known (Azar et al., 2011b) for the sample complexity of RL. However, there exist gaps with polynomial dependency on $1/(1-\gamma)$ between these upper bounds and the state-of-the-art lower bound, which is of order $\widetilde{\Omega}\big(N/((1-\gamma)^2\varepsilon^2)\big)$ (Azar et al., 2011a; Even-Dar et al., 2006).[c] It has not been clear, so far, whether the upper bounds or the lower bound can be

[b] The notation $g = \widetilde{O}(f)$ implies that there are constants c_1 and c_2 such that $g \leq c_1 f \log^{c_2}(f)$.

[c] The notation $g = \widetilde{\Omega}(f)$ implies that there are constants c_1 and c_2 such that $g \geq c_1 f \log^{c_2}(f)$.

improved or both.

In this chapter, we prove new bounds on the performance of QVI and PI which indicate that for both algorithms with the probability (w.p) $1-\delta$ an order of $O\big(N\log(N/\delta)/\big((1-\gamma)^3\varepsilon^2\big)\big)$ samples suffice to achieve an ε-optimal estimate of action-value function as well as to find an ε-optimal policy. The new upper bound improves on the previous result of AVI and API by an order of $1/(1-\gamma)$. We also present a new minimax lower bound of $\Theta\big(N\log(N/\delta)/\big((1-\gamma)^3\varepsilon^2\big)\big)$, which also improves on the best existing lower bound of RL by an order of $1/(1-\gamma)$. The new results, which close the above-mentioned gap between the lower bound and the upper bound, guarantee that no learning method, given the generative model of the MDP, can be significantly more efficient than QVI and PI in terms of the sample complexity of estimating the optimal action-value function or the optimal policy.

The main idea to improve the upper bound of the above-mentioned RL algorithms is to express the performance loss Q^*-Q_k, where Q_k is the estimate of the action-value function after k iteration of QVI or PI, in terms of Σ^{π^*}, the variance of the sum of discounted rewards under the optimal policy π^*, as opposed to the maximum $V_{\max}=R_{\max}/(1-\gamma)$ as was used before. For this we make use of the Bernstein's concentration inequality (Cesa-Bianchi and Lugosi, 2006, appendix, pg. 361), which is expressed in terms of the variance of the random variables. We also rely on the fact that the variance of the sum of discounted rewards, like the expected value of the sum (value function), satisfies a Bellman-like equation, in which the variance of the value function plays the role of the instant reward in the standard Bellman equation (Munos and Moore, 1999; Sobel, 1982). These results allow us to prove a high-probability bound of order $\widetilde{O}(\sqrt{\Sigma^{\pi^*}/(n(1-\gamma))})$ on the performance loss of both algorithms, where n is the number of samples per state-action. This leads to a tight PAC upper-bound of $\widetilde{O}(N/(\varepsilon^2(1-\gamma)^3))$ on the sample complexity of these methods.

In the case of lower bound, we introduce a new class of "hard" MDPs, which adds some structure to the bandit-like class of MDP used previously by Azar et al. (2011a); Even-Dar et al. (2006): in the new model, there exist states with high probability of transition to themselves. This adds to the difficulty of estimating the value function, since even a small modeling error may cause a large error in the estimate of the optimal value function, especially when the discount factor γ is close to 1.

The rest of the chapter is organized as follows. After introducing the notations used in the chapter in Section 4.2, we describe the *model-based Q-value iteration* (QVI) algorithm and the *model-based policy iteration* (PI) in Subsec-

tion 4.2.1. We then state our main theoretical results, which are in the form of PAC sample complexity bounds in Section 4.3. Section 4.4 contains the detailed proofs of the results of Sections 4.3, i.e., sample complexity bound of QVI and a matching lower bound for RL. Finally, we conclude the chapter and propose some directions for the future work in Section 4.5.

4.2 Background

In this section, we review some standard concepts and definitions from the theory of Markov decision processes (MDPs). We then present two model-based RL algorithms which make use of generative model for sampling: the model-based Q-value iteration and the model-based policy iteration (Wiering and van Otterlo, 2012a; Kearns and Singh, 1999).

We consider the standard reinforcement learning (RL) framework (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998), where an RL agent interacts with a stochastic environment and this interaction is modeled as a discrete-time discounted MDP. A discounted MDP is a quintuple $(\mathcal{X}, \mathcal{A}, P, \mathcal{R}, \gamma)$, where $\mathcal{X}$ and $\mathcal{A}$ are the set of states and actions, P is the state transition distribution, $\mathcal{R}$ is the reward function, and $\gamma \in [0, 1)$ is a discount factor.[d] We denote by $P(\cdot|x, a)$ and $r(x, a)$ the probability distribution over the next state and the immediate reward of taking action a at state x, respectively.

To keep the representation succinct, in the sequel, we use the notation $\mathcal{Z}$ for the joint state-action space $\mathcal{X} \times \mathcal{A}$. We also make use of the shorthand notations z and β for the state-action pair (x, a) and $1/(1-\gamma)$, respectively.

Assumption 4.1 (MDP Regularity)**.** *We assume $\mathcal{Z}$ and, subsequently, $\mathcal{X}$ and $\mathcal{A}$ are finite sets with cardinalities N, $|\mathcal{X}|$ and $|\mathcal{A}|$, respectively. We also assume that the immediate reward $r(x, a)$ is taken from the interval $[0, 1]$.*[e]

A mapping $\pi : \mathcal{X} \to \mathcal{A}$ is called a stationary and deterministic Markovian policy, or just a policy in short. Following a policy π in an MDP means that at each time step t the control action $A_t \in \mathcal{A}$ is given by $A_t = \pi(X_t)$,

[d] For simplicity, here we assume that the reward $r(x, a)$ is a deterministic function of state-action pairs (x, a). Nevertheless, It is straightforward to extend our results to the case of stochastic rewards under some mild assumption, e.g., boundedness of the absolute value of the rewards.

[e] Our results also hold if the rewards are taken from some interval $[r_{\min}, r_{\max}]$ instead of $[0, 1]$, in which case the bounds scale with the factor $r_{\max} - r_{\min}$.

where $X_t \in \mathcal{X}$. The *value* and the *action-value functions* of a policy π, denoted respectively by $V^\pi : \mathcal{X} \to \mathbb{R}$ and $Q^\pi : \mathcal{Z} \to \mathbb{R}$, are defined as the expected sum of discounted rewards that are encountered when the policy π is executed. Given an MDP, the goal is to find a policy that attains the best possible values, $V^*(x) \triangleq \sup_\pi V^\pi(x)$, $\forall x \in \mathcal{X}$. The function V^* is called the *optimal value function*. Similarly the *optimal action-value function* is defined as $Q^*(x,a) = \sup_\pi Q^\pi(x,a)$. We say that a policy π^* is optimal if it attains the optimal $V^*(x)$ for all $x \in \mathcal{X}$. The policy π defines the state transition kernel P_π as $P_\pi(y|x) \triangleq P(y|x,\pi(x))$ for all $x \in \mathcal{X}$. The right-linear operators $P^\pi \cdot$, $P \cdot$ and $P_\pi \cdot$ are also defined as $(P^\pi Q)(z) \triangleq \sum_{y \in \mathcal{X}} P(y|z) Q(y, \pi(y))$, $(PV)(z) \triangleq \sum_{y \in \mathcal{X}} P(y|z) V(y)$ for all $z \in \mathcal{Z}$ and $(P_\pi V)(x) \triangleq \sum_{y \in \mathcal{X}} P_\pi(y|x) V(y)$ for all $x \in \mathcal{X}$, respectively. The optimal action-value function Q^* is the unique fixed-point of the *Bellman optimality operator* defined as

$$(\mathcal{T}Q)(z) \triangleq r(z) + \gamma(P^{\pi^*}Q)(z), \qquad \forall z \in \mathcal{Z}.$$

Also, for the policy π, the action-value function Q^π is the unique fixed-point of the *Bellman operator* $\mathcal{T}^\pi$ which is defined as $(\mathcal{T}^\pi Q)(z) \triangleq r(z) + \gamma(P^\pi Q)(z)$ for all $z \in \mathcal{Z}$. One can also define the Bellman optimality operator and the Bellman operator on the value function as $(\mathcal{T}V)(x) \triangleq r(x, \pi^*(x)) + \gamma(P_{\pi^*}V)(x)$ and $(\mathcal{T}^\pi V)(x) \triangleq r(x,\pi(x)) + \gamma(P_\pi V)(x)$ for all $x \in \mathcal{X}$, respectively.

It is important to note that $\mathcal{T}$ and $\mathcal{T}^\pi$ are γ-contractions, i.e., for any pair of value functions V and V' and any policy π, we have $\|\mathcal{T}V - \mathcal{T}V'\| \leq \gamma\|V - V'\|$ and $\|\mathcal{T}^\pi V - \mathcal{T}^\pi V'\| \leq \gamma\|V - V'\|$ (Bertsekas, 2007b, Chap. 1). $\|\cdot\|$ shall denote the supremum (ℓ_∞) norm, defined as $\|g\| \triangleq \max_{y \in \mathcal{Y}} |g(y)|$, where $\mathcal{Y}$ is a finite set and $g : \mathcal{Y} \to \mathbb{R}$ is a real-valued function. We also define the ℓ_1-norm on the function g as $\|g\|_1 = \sum_{y \in \mathcal{Y}} |g(y)|$.

For ease of exposition, in the sequel, we remove the dependence on z and x, e.g., writing Q for $Q(z)$ and V for $V(x)$, when there is no possible confusion.

4.2.1 Algorithms

We begin by describing the procedure which is used by both PI and QVI to make an empirical estimate of the state-transition distributions.

The model estimator makes n transition samples for each state-action pair $z \in \mathcal{Z}$ for which it makes n calls to the generative model, i.e., the total number of calls to the generative model is $T = nN$. It then builds an empirical model of the transition probabilities as $\widehat{P}(y|z) \triangleq m(y,z)/n$, where $m(y,z)$ denotes the

number of times that the state $y \in \mathcal{X}$ has been reached from the state-action pair $z \in \mathcal{Z}$ (see Algorithm 4.3). Based on the empirical model $\widehat{P}$ the operator $\widehat{\mathcal{T}}$ is defined on the action-value function Q, for all $z \in \mathcal{Z}$, by $\widehat{\mathcal{T}}Q(z) = r(z) + \gamma(\widehat{P}V)(z)$, with $V(x) = \max_{a\in\mathcal{A}}(Q(x,a))$ for all $x \in \mathcal{X}$. Also, the empirical operator $\widehat{\mathcal{T}}^{\pi}$ is defined on the action-value function Q, for every policy π and all $z \in \mathcal{Z}$, by $\widehat{\mathcal{T}}^{\pi}Q(z) = r(z)+\gamma\widehat{P}^{\pi}Q(z)$. Likewise, one can also define the empirical Bellman operator $\widehat{\mathcal{T}}$ and $\widehat{\mathcal{T}}^{\pi}$ for the value function V. The fixed points of the operator $\widehat{\mathcal{T}}$ in $\mathcal{Z}$ and $\mathcal{X}$ domains are denoted by $\widehat{Q}^*$ and $\widehat{V}^*$, respectively. Also, the fixed points of the operator $\widehat{\mathcal{T}}^{\pi}$ in $\mathcal{Z}$ and $\mathcal{X}$ domains are denoted by $\widehat{Q}^{\pi}$ and $\widehat{V}^{\pi}$, respectively. The empirical optimal policy $\widehat{\pi}^*$ is the policy which attains $\widehat{V}^*$ under the model $\widehat{P}$.

Having the empirical model $\widehat{P}$ estimated, QVI and PI rely on standard value iteration and policy iteration schemes to estimate the optimal action-value function: QVI iterates some action-value function Q_j, with the initial value of Q_0, through the empirical Bellman optimality operator $\widehat{\mathcal{T}}$ until Q_j admits some convergence criteria. PI, in contrast, relies on iterating some policy π_j with the initial value π_0: at each iteration $j > 0$, the algorithm solves the dynamic programming problem for a fixed policy π_j using the empirical model $\widehat{P}$. The next policy π_{j+1} is then determined as the greedy policy w.r.t. the action-value function $\widehat{Q}^{\pi_j}$, that is, $\pi_{j+1}(x) = \arg\max_{a\in\mathcal{A}} \widehat{Q}^{\pi_j}(x,a)$ for all $x \in \mathcal{X}$. Note that Q_k, as defined by PI and QVI are deferent, but nevertheless we use a same notation for both action-functions since we will show in the next section that they enjoy the same performance guarantees. The pseudo codes of both algorithms are provided in Algorithm 4.1 and Algorithm 4.2.

Algorithm 4.1 Model-based Q-value Iteration (QVI)

Require: reward function r, discount factor γ, initial action-value function Q_0, samples per state-action n, number of iterations k

```
P̂ = EstimateModel(n)                          ▷ Estimate the model (defined in Algorithm 4.3)
for j := 0, 1, ..., k − 1 do
    for each x ∈ X do
        π_j(x) = arg max_{a∈A} Q_j(x, a)       ▷ greedy policy w.r.t. the latest estimation of Q*
        for each a ∈ A do
            T̂Q_j(x, a) = r(x, a) + γ(P̂^{π_j} Q_j)(x, a)        ▷ empirical Bellman operator
            Q_{j+1}(x, a) = T̂Q_j(x, a)                          ▷ Iterate the action-value function Q_j
        end for
    end for
end for
return Q_k
```

Algorithm 4.2 Model-based Policy Iteration (PI)

Require: reward function r, discount factor γ, initial policy π_0, samples per state-action n, number of iterations k

$\widehat{P}$ =EstimateModel(n) ▷ Estimate the model (defined in Algorithm 4.3)
Q_0=SolveDP($\widehat{P}, \pi_0$)
for $j := 0, 1, \ldots, k-1$ **do**
  for each $x \in \mathcal{X}$ **do**
    $\pi_j(x) = \arg\max_{a \in \mathcal{A}} Q_j(x, a)$ ▷ greedy policy w.r.t. the latest estimation of Q^*
  end for
  $\widehat{Q}^{\pi_j}$=SolveDP($\widehat{P}, \pi_j$) ▷ Find the fixed point of the Bellman operator for the policy π_j
  $Q_{j+1} = \widehat{Q}^{\pi_j}$ ▷ Iterate the action-value function Q_j
end for
return Q_k

function SolveDP(P, π)
  $Q = (I - \gamma P^{\pi})^{-1} r$
  return Q
end function

Algorithm 4.3 Function: EstimateModel

Require: The generative model (simulator) P
function EstimateModel(n) ▷ Estimating the transition model using n samples
  $\forall (y, z) \in \mathcal{X} \times \mathcal{Z} : m(y, z) = 0$ ▷ initialization
  for each $z \in \mathcal{Z}$ **do**
    for $i := 1, 2, \ldots, n$ **do**
      $y \sim P(\cdot|z)$ ▷ Generate a state-transition sample
      $m(y, z) := m(y, z) + 1$ ▷ Count the transition samples
    end for
    $\forall y \in \mathcal{X} : \widehat{P}(y|z) = \frac{m(y,z)}{n}$ ▷ Normalize by n
  end for
  return $\widehat{P}$ ▷ Return the empirical model
end function

4.3 Main Results

Our main results are in the form of PAC (probably approximately correct) sample complexity bounds on the total number of samples required to attain a near-optimal estimate of the action-value function:

Theorem 4.1 (PAC-bound on $Q^* - Q_k$)**.** *Let Assumption 4.1 hold and T be a positive integer. Then, there exist some constants c, c_0, d and d_0 such that for all $\varepsilon \in (0,1)$ and $\delta \in (0,1)$, a total sampling budget of*

$$T = \lceil \frac{c\beta^3 N}{\varepsilon^2} \log \frac{c_0 N}{\delta} \rceil,$$

suffices for the uniform approximation error $\|Q^* - Q_k\| \leq \varepsilon$, w.p. at least $1-\delta$, after $k = \lceil d \log(d_0 \beta/\varepsilon)/\log(1/\gamma) \rceil$ iteration of QVI or PI algorithm.[f]

We also prove a similar bound on the sample-complexity of finding a near-optimal policy for small values of ε:

Theorem 4.2 (PAC-bound on $Q^* - Q^{\pi_k}$)**.** *Let Assumption 4.1 hold and T be a positive integer. Define π_k as the greedy policy w.r.t. Q_k at iteration k of PI or QVI. Then, there exist some constants c', c'_0, c'_1, d' and d'_0 such that for all $\varepsilon \in (0, c'_1 \sqrt{\beta/(\gamma|\mathcal{X})|})$ and $\delta \in (0,1)$, a total sampling budget of*

$$T = \lceil \frac{c'\beta^3 N}{\varepsilon^2} \log \frac{c'_0 N}{\delta} \rceil,$$

suffices for the uniform approximation error $\|V^* - V^{\pi_k}\| \leq \|Q^* - Q^{\pi_k}\| \leq \varepsilon$, w.p. at least $1-\delta$, after $k = d' \lceil \log(d'_0 \beta/\varepsilon)/\log(1/\gamma) \rceil$ iteration of QVI or PI algorithm.

The following general result provides a tight lower bound on the number of transitions T for every RL algorithm to find a near optimal solution w.p. $1-\delta$, under the assumption that the algorithm is (ε, δ, T)-correct:

Definition 4.1 ((ε,δ)-correct algorithm)**.** *Let $Q^{\mathfrak{A}} : \mathcal{Z} \to \mathbb{R}$ be the output of some RL Algorithm $\mathfrak{A}$. We say that $\mathfrak{A}$ is (ε,δ)-correct on the class of MDPs $\mathbb{M} = \{M_1, M_2, \ldots, M_m\}$ if $\left\|Q^* - Q^{\mathfrak{A}}\right\| \leq \varepsilon$ with probability at least $1-\delta$ for all $M \in \mathbb{M}$.*[g]

Theorem 4.3 (Lower bound on the sample complexity of RL)**.** *Let Assumption 4.1 hold and T be a positive integer. There exist some constants ε_0, δ_0, c_1, c_2, and a class of MDPs $\mathbb{M}$, such that for all $\varepsilon \in (0, \varepsilon_0)$, $\delta \in (0, \delta_0/N)$, and every (ε,δ)-correct RL Algorithm $\mathfrak{A}$ on the class of MDPs $\mathbb{M}$ the total number of state-transition samples (sampling budget) needs to be at least*

[f] For every real number u, $\lceil u \rceil$ is defined as the smallest integer number not less than u.

[g] Algorithm $\mathfrak{A}$, unlike QVI and PI, does not require a same number of transition samples for every state-action pair and can generate samples arbitrarily.

$$T = \lceil \frac{\beta^3 N}{c_1 \varepsilon^2} \log \frac{N}{c_2 \delta} \rceil.$$

4.4 Analysis

In this section, we first provide the full proof of the finite-time PAC bound of QVI and PI, reported in Theorem 4.1 and Theorem 4.2, in Subsection 4.4.1. We then prove Theorem 4.3, a new RL lower bound, in Subsection 4.4.2.

4.4.1 Proofs of Theorem 4.1 and Theorem 4.2 - the Upper Bounds

We begin by introducing some new notation. For any policy π, we define $\Sigma^\pi(z) \triangleq \mathbb{E}[|\sum_{t\geq 0}\gamma^t r(Z_t) - Q^\pi(z)|^2 | Z_0 = z]$ as the variance of the sum of discounted rewards starting from $z \in \mathcal{Z}$ under the policy π. We also make use of the following definition of the variance of a function: for any real-valued function $f : \mathcal{Y} \to \mathbb{R}$, where $\mathcal{Y}$ is a finite set, we define $\mathbb{V}_{y\sim\rho}(f(y)) \triangleq \mathbb{E}_{y\sim\rho}|f(y) - \mathbb{E}_{y\sim\rho}(f(y))|^2$ as the variance of f under the probability distribution ρ, where ρ is a probability distribution on $\mathcal{Y}$. Weshall denote σ_{V^π} and σ_{V^*} as the discounted variance of the value function V^π and V^* defined as $\sigma_{V^\pi}(z) \triangleq \gamma^2 \mathbb{V}_{y\sim P(\cdot|z)}[V^\pi(y)]$ and $\sigma_{V^*}(z) \triangleq \gamma^2 \mathbb{V}_{y\sim P(\cdot|z)}[V^*(y)]$, for all $z \in \mathcal{Z}$, respectively. For each of these variances we define the corresponding empirical variance $\widehat{\sigma}_{V^\pi}(z) \triangleq \gamma^2 \mathbb{V}_{y\sim \widehat{P}(\cdot|z)}[V^\pi(y)]$ and $\widehat{\sigma}_{V^*}(z) \triangleq \gamma^2 \mathbb{V}_{y\sim \widehat{P}(\cdot|z)}[V^*(y)]$, respectively, for all $z \in \mathcal{Z}$ under the model $\widehat{P}$. We also notice that for any policy π and for all $z \in \mathcal{Z}$, σ_{V^π} can be written as

$$\sigma_{V^\pi}(z) = \gamma^2 P[|V^\pi - PV^\pi|^2](z) = \gamma^2 P^\pi[|Q^\pi - P^\pi Q^\pi|^2](z).$$

We now prove our first result which shows that Q_k, for both QVI and PI, is very close to $\widehat{Q}^*$ up to an order of $O(\gamma^k)$. Therefore, to prove bound on $\|Q^* - Q_k\|$, one only needs to bound $\|Q^* - \widehat{Q}^*\|$ in high probability.

Lemma 4.1. *Let Assumption 4.1 hold and $Q_0(z)$ be in the interval $[0, \beta]$ for all $z \in \mathcal{Z}$. Then, for both QVI and PI, we have*

$$\|Q_k - \widehat{Q}^*\| \leq \gamma^k \beta.$$

Proof

First, we prove the result for QVI. For all $k \geq 0$, we have

$$\|Q_k - \widehat{Q}^*\| = \|\widehat{\mathcal{T}} Q_{k-1} - \widehat{\mathcal{T}}\widehat{Q}^*\| \leq \gamma \|Q_{k-1} - \widehat{Q}^*\|.$$

Thus by an immediate recursion

$$\|Q_k - \widehat{Q}^*\| \leq \gamma^k \|Q_0 - \widehat{Q}^*\| \leq \gamma^k \beta.$$

In the case of PI, we notice that $Q_k = \widehat{Q}^{\pi_{k-1}} \geq \widehat{Q}^{\pi_{k-2}} = Q_{k-1}$, which implies that

$$\begin{aligned} 0 \leq \widehat{Q}^* - Q_k &= \gamma \widehat{P}^{\widehat{\pi}^*} \widehat{Q}^* - \gamma \widehat{P}^{\pi_{k-1}} \widehat{Q}^{\pi_{k-1}} \leq \gamma (\widehat{P}^{\widehat{\pi}^*} \widehat{Q}^* - \widehat{P}^{\pi_{k-1}} \widehat{Q}^{\pi_{k-2}}) \\ &= \gamma (\widehat{P}^{\widehat{\pi}^*} \widehat{Q}^* - \widehat{P}^{\pi_{k-1}} Q_{k-1}) \leq \gamma \widehat{P}^{\widehat{\pi}^*} (\widehat{Q}^* - Q_{k-1}), \end{aligned}$$

where in the last line we rely on the fact that π_{k-1} is the greedy policy w.r.t. Q_{k-1}, which implies the component-wise inequality $\widehat{P}^{\pi_{k-1}} Q_{k-1} \geq \widehat{P}^{\widehat{\pi}^*} Q_{k-1}$.

The result then follows by taking the ℓ_∞-norm on both sides of the inequality and then recursively expand the resulted bound.

■

One can easily prove the following lemma, which bounds the difference between $\widehat{Q}^*$ and $\widehat{Q}^{\pi_k}$, based on the result of Lemma 4.1 and the main result of Singh and Yee (1994). Lemma 4.2 is required for the proof of Theorem 4.2.

Lemma 4.2. *Let Assumption 4.1 hold and π_k be the greedy policy induced by the k^{th} iterate of QVI and PI. Also, let $Q_0(z)$ takes value in the interval $[0, \beta]$ for all $z \in \mathcal{Z}$. Then we have*

$$\|\widehat{Q}^{\pi_k} - \widehat{Q}^*\| \leq 2\gamma^{k+1}\beta^2, \qquad \textit{and} \qquad \|\widehat{V}^{\pi_k} - \widehat{V}^*\| \leq 2\gamma^{k+1}\beta^2.$$

Proof

Based on the main theorem of Singh and Yee (1994) we have, for both QVI and PI:

$$\begin{aligned} \|\widehat{V}^{\pi_k} - \widehat{V}^*\| \leq \|\widehat{Q}^{\pi_k} - \widehat{Q}^*\| &\leq 2\gamma\beta \|Q_k - \widehat{Q}^*\| \\ &\leq 2\gamma^{k+1}\beta^2, \end{aligned}$$

where in the last line we make use of the result of Lemma 4.1.

■

We notice that the tight bound on $\|\widehat{Q}^{\pi_k} - \widehat{Q}^*\|$ for PI is of order $\gamma^{k+1}\beta$ since $\widehat{Q}^{\pi_k} = Q_{k+1}$. However, for ease of exposition we make use of the bound of Lemma 4.2 for both QVI and PI.

In the rest of this subsection, we focus on proving a high probability bound on $\|Q^* - \widehat{Q}^*\|$. One can prove a crude bound of $\widetilde{O}(\beta^2/\sqrt{n})$ on $\|Q^* - \widehat{Q}^*\|$ by first proving that $\|Q^* - \widehat{Q}^*\| \leq \beta\|(P - \widehat{P})V^*\|$ and then using the Hoeffding's tail inequality (Cesa-Bianchi and Lugosi, 2006, appendix, pg. 359) to bound the random variable $\|(P - \widehat{P})V^*\|$ in high probability. Here, we follow a different and more subtle approach to bound $\|Q^* - \widehat{Q}^*\|$, which leads to our desired result of $\widetilde{O}(\beta^{1.5}/\sqrt{n})$: **(i)** We prove in Lemma 4.3 component-wise upper and lower bounds on the error $Q^* - \widehat{Q}^*$ which are expressed in terms of $(I-\gamma\widehat{P}^{\pi^*})^{-1}\big[P-\widehat{P}\big]V^*$ and $(I-\gamma\widehat{P}^{\widehat{\pi}^*})^{-1}\big[P-\widehat{P}\big]V^*$, respectively. **(ii)** We make use of of Bernstein's inequality to bound $\big[P-\widehat{P}\big]V^*$ in terms of the squared root of the variance of V^* in high probability. **(iii)** We prove the key result of this subsection (Lemma 4.7) which shows that the variance of the sum of discounted rewards satisfies a Bellman-like recursion, in which the instant reward $r(z)$ is replaced by $\sigma_{V^\pi}(z)$. Based on this result we prove an upper-bound of order $O(\beta^{1.5})$ on $(I-\gamma P^\pi)^{-1}\sqrt{\sigma_{V^\pi}}$ for every policy π, which combined with the previous steps leads to an upper bound of $\widetilde{O}(\beta^{1.5}/\sqrt{n})$ on $\|Q^*-\widehat{Q}^*\|$. A similar approach leads to a bound of $\widetilde{O}(\beta^{1.5}/\sqrt{n})$ on $\|Q^*-Q^{\pi_k}\|$ under the assumption that there exist constants $c_1 > 0$ and $c_2 > 0$ such that $n > c_1\gamma^2\beta^2|\mathcal{X}|\log(c_2 N/\delta)$).

The following component-wise results bound $Q^*-\widehat{Q}^*$ from above and below:

Lemma 4.3 (Component-wise bounds on $Q^* - \widehat{Q}^*$)**.**

$$Q^* - \widehat{Q}^* \leq \gamma(I - \gamma\widehat{P}^{\pi^*})^{-1}\big[P - \widehat{P}\big]V^*, \tag{4.1}$$

$$Q^* - \widehat{Q}^* \geq \gamma(I - \gamma\widehat{P}^{\widehat{\pi}^*})^{-1}\big[P - \widehat{P}\big]V^*. \tag{4.2}$$

Proof

We have that $\widehat{Q}^* \geq \widehat{Q}^{\pi^*}$. Thus:

$$\begin{aligned} Q^* - \widehat{Q}^* &\leq Q^* - \widehat{Q}^{\pi^*} = (I - \gamma P^{\pi^*})^{-1}r - (I - \gamma\widehat{P}^{\pi^*})^{-1}r \\ &= (I - \gamma\widehat{P}^{\pi^*})^{-1}\big[(I - \gamma\widehat{P}^{\pi^*}) - (I - \gamma P^{\pi^*})\big](I - \gamma P^{\pi^*})^{-1}r \\ &= \gamma(I - \gamma\widehat{P}^{\pi^*})^{-1}\big[P^{\pi^*} - \widehat{P}^{\pi^*}\big]Q^* = \gamma(I - \gamma\widehat{P}^{\pi^*})^{-1}\big[P - \widehat{P}\big]V^*. \end{aligned}$$

In the case of Ineq. (4.2) we have

$$\begin{aligned} Q^* - \widehat{Q}^* &= (I - \gamma P^{\pi^*})^{-1} r - (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} r \\ &= (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} \big[(I - \gamma \widehat{P}^{\widehat{\pi}^*}) - (I - \gamma P^{\pi^*}) \big] (I - \gamma P^{\pi^*})^{-1} r \\ &= \gamma (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} \big[P^{\pi^*} - \widehat{P}^{\widehat{\pi}^*} \big] Q^* \\ &\geq \gamma (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} \big[P^{\pi^*} - \widehat{P}^{\pi^*} \big] Q^* = \gamma (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} \big[P - \widehat{P} \big] V^*, \end{aligned}$$

in which we make use of the following component-wise inequality:[h]

$$\begin{aligned} (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} \widehat{P}^{\pi^*} Q^* &= \sum_{i \geq 0} \left(\gamma \widehat{P}^{\pi^*} \right)^i \widehat{P}^{\pi^*} Q^* \\ &\geq \sum_{i \geq 0} \left(\gamma \widehat{P}^{\widehat{\pi}^*} \right)^i \widehat{P}^{\widehat{\pi}^*} Q^* = (I - \gamma \widehat{P}^{\widehat{\pi}^*})^{-1} \widehat{P}^{\widehat{\pi}^*} Q^*. \end{aligned}$$

■

We now concentrate on bounding the RHS (right hand sides) of (4.1) and (4.2) in high probability, for that we need the following technical lemmas (Lemma 4.4 and Lemma 4.5).

Lemma 4.4. *Let Assumption 4.1 hold. Then, for any* $0 < \delta < 1$ *w.p at least* $1 - \delta$

$$\|V^* - \widehat{V}^{\pi^*}\| \leq c_v, \qquad \textit{and} \qquad \|V^* - \widehat{V}^*\| \leq c_v,$$

where $c_v \triangleq \gamma \beta^2 \sqrt{2 \log(2N/\delta)/n}$.

Proof

We begin by proving bound on $\|V^* - \widehat{V}^{\pi^*}\|$:

$$\begin{aligned} \|V^* - \widehat{V}^{\pi^*}\| = \|\mathcal{T}^{\pi^*} V^* - \widehat{\mathcal{T}}^{\pi^*} \widehat{V}^{\pi^*}\| &\leq \|\mathcal{T}^{\pi^*} V^* - \widehat{\mathcal{T}}^{\pi^*} V^*\| + \|\widehat{\mathcal{T}}^{\pi^*} V^* - \widehat{\mathcal{T}}^{\pi^*} \widehat{V}^{\pi^*}\| \\ &\leq \gamma \|P_{\pi^*} V^* - \widehat{P}_{\pi^*} V^*\| + \gamma \|V^* - \widehat{V}^{\pi^*}\|. \end{aligned}$$

[h] For any policy π and $k \geq 1$: $(P^\pi)^k(\cdot) \triangleq \underbrace{P^\pi \cdots P^\pi}_{k}(\cdot)$.

By solving this inequality w.r.t. $\|V^* - \widehat{V}^{\pi^*}\|$ we deduce

$$\|V^* - \widehat{V}^{\pi^*}\| \leq \gamma\beta\|(P_{\pi^*} - \widehat{P}_{\pi^*})V^*\| \leq \gamma\beta\|(P - \widehat{P})V^*\|. \tag{4.3}$$

Now we focus on bounding $\|V^* - \widehat{V}^*\|$:

$$\begin{aligned}
\|V^* - \widehat{V}^*\| \leq \|Q^* - \widehat{Q}^*\| &= \|\mathcal{T}Q^* - \widehat{\mathcal{T}}\widehat{Q}^*\| \\
&\leq \|\mathcal{T}Q^* - \widehat{\mathcal{T}}^{\pi^*}Q^*\| + \|\widehat{\mathcal{T}}^{\pi^*}Q^* - \widehat{\mathcal{T}}\widehat{Q}^*\| \\
&= \gamma\|P^{\pi^*}Q^* - \widehat{P}^{\pi^*}Q^*\| + \gamma\|\widehat{P}^{\pi^*}Q^* - \widehat{P}^{\widehat{\pi}^*}\widehat{Q}^*\| \\
&= \gamma\|(P - \widehat{P})V^*\| + \gamma\|\widehat{P}(V^* - \widehat{V}^*)\| \\
&\leq \gamma\|(P - \widehat{P})V^*\| + \gamma\|V^* - \widehat{V}^*\|.
\end{aligned} \tag{4.4}$$

By solving this inequality w.r.t. $\|V^* - \widehat{V}^*\|$ we deduce:

$$\|V^* - \widehat{V}^*\| \leq \gamma\beta\|(P - \widehat{P})V^*\|. \tag{4.5}$$

We then make use of Hoeffding's inequality (Cesa-Bianchi and Lugosi, 2006, Appendix A, pg. 359) to bound $|(P - \widehat{P})V^*(z)|$ for all $z \in \mathcal{Z}$ in high probability:

$$\mathbb{P}(|((P - \widehat{P})V^*)(z)| \geq \varepsilon) \leq 2\exp\left(\tfrac{-n\varepsilon^2}{2\beta^2}\right).$$

By applying the union bound we deduce

$$\mathbb{P}(\|(P - \widehat{P})V^*\| \geq \varepsilon) \leq 2|\mathcal{Z}|\exp\left(\tfrac{-n\varepsilon^2}{2\beta^2}\right). \tag{4.6}$$

We then define the probability of failure δ as

$$\delta \triangleq 2N\exp\left(\tfrac{-n\varepsilon^2}{2\beta^2}\right). \tag{4.7}$$

By plugging (4.7) into (4.6) we deduce

$$\mathbb{P}\left[\|(P - \widehat{P})V^*\| < \beta\sqrt{2\log\left(2N/\delta\right)/n}\right] \geq 1 - \delta. \tag{4.8}$$

The results then follow by plugging (4.8) into (4.5) and (4.4).

■

We now state Lemma 4.5 which relates σ_{V^*} to $\widehat{\sigma}_{\widehat{Q}^{\pi^*}}$ and $\widehat{\sigma}_{\widehat{Q}^*}$. Later, we make use of this result in the proof of Lemma 4.6.

Lemma 4.5. *Let Assumption 4.1 hold and* $0 < \delta < 1$. *Then, w.p. at least* $1 - \delta$*:*

$$\sigma_{V^*} \leq \widehat{\sigma}_{\widehat{V}^{\pi^*}} + b_v \mathbf{1}, \tag{4.9}$$
$$\sigma_{V^*} \leq \widehat{\sigma}_{\widehat{V}^*} + b_v \mathbf{1}, \tag{4.10}$$

where b_v *is defined as*

$$b_v \triangleq \sqrt{\frac{18\gamma^4\beta^4 \log\frac{3N}{\delta}}{n}} + \frac{4\gamma^2\beta^4 \log\frac{6N}{\delta}}{n},$$

and $\mathbf{1}$ *is a function which assigns* 1 *to all* $z \in \mathcal{Z}$.

Proof

Here, we only prove (4.9). One can prove (4.10) following similar lines.

$$\begin{aligned}\sigma_{V^*}(z) &= \sigma_{V^*}(z) - \gamma^2 \mathbb{V}_{Y\sim\widehat{P}(\cdot|z)}(V^*(Y)) + \gamma^2 \mathbb{V}_{Y\sim\widehat{P}(\cdot|z)}(V^*(Y)) \\ &\leq \gamma^2\big((P-\widehat{P})V^{*2}\big)(z) - \gamma^2[(PV^*)^2(z) - (\widehat{P}V^*)^2(z)] \\ &\quad + \gamma^2 \mathbb{V}_{Y\sim\widehat{P}(\cdot|z)}(V^*(Y) - \widehat{V}^{\pi^*}(Y)) + \gamma^2 \mathbb{V}_{Y\sim\widehat{P}(\cdot|z)}(\widehat{V}^{\pi*}(Y)).\end{aligned}$$

It is not difficult to show that $\mathbb{V}_{Y\sim\widehat{P}(\cdot|z)}(V^*(Y) - \widehat{V}^{\pi^*}(Y)) \leq \|V^* - \widehat{V}^{\pi^*}\|^2$, which implies that

$$\begin{aligned}\sigma_{V^*}(z) &\leq \gamma^2[P-\widehat{P}]V^{*2}(z) - \gamma^2[(P-\widehat{P})V^*][(P+\widehat{P})V^*](z) \\ &\quad + \gamma^2\|V^* - \widehat{V}^{\pi^*}\|^2 + \widehat{\sigma}_{\widehat{V}^{\pi^*}}(z).\end{aligned}$$

The following inequality then holds w.p. at least $1 - \delta$:

$$\sigma_{V^*}(z) \leq \widehat{\sigma}_{\widehat{V}^{\pi^*}}(z) + \gamma^2 \left[3\beta^2\sqrt{2\frac{\log\frac{3}{\delta}}{n}} + \frac{2\beta^4 \log\frac{6N}{\delta}}{n}\right], \tag{4.11}$$

in which we make use of Hoeffding's inequality as well as Lemma 4.4 and a union bound to prove the bound on σ_{V^*} in high probability. This combined with a union bound on all state-action pairs in Eq.(4.11) completes the proof. ■

The following result proves a bound on $\gamma(P - \widehat{P})V^*$, for which we make use of the Bernstein's inequality (Cesa-Bianchi and Lugosi, 2006, appendix, pg. 361) as well as Lemma 4.5.

Lemma 4.6. *Let Assumption 4.1 hold and* $0 < \delta < 1$. *Define* $c_{pv} \triangleq 2\log(2N/\delta)$ *and* b_{pv} *as*

$$b_{pv} \triangleq \left(\frac{5(\gamma\beta)^{4/3}\log\frac{6N}{\delta}}{n}\right)^{3/4} + \frac{3\beta^2\log\frac{12N}{\delta}}{n}.$$

Then w.p. at least $1 - \delta$ *we have*

$$\gamma(P - \widehat{P})V^* \leq \sqrt{\frac{c_{pv}\widehat{\sigma}_{\widehat{V}^{\pi^*}}}{n}} + b_{pv}\mathbf{1}, \tag{4.12}$$

$$\gamma(P - \widehat{P})V^* \geq -\sqrt{\frac{c_{pv}\widehat{\sigma}_{\widehat{V}^*}}{n}} - b_{pv}\mathbf{1}. \tag{4.13}$$

Proof

For all $z \in \mathcal{Z}$ and all $0 < \delta < 1$, Bernstein's inequality implies that w.p. at least $1 - \delta$:

$$(P - \widehat{P})V^*(z) \leq \sqrt{\frac{2\sigma_{V^*}(z)\log\frac{1}{\delta}}{\gamma^2 n}} + \frac{2\beta\log\frac{1}{\delta}}{3n},$$

$$(P - \widehat{P})V^*(z) \geq -\sqrt{\frac{2\sigma_{V^*}(z)\log\frac{1}{\delta}}{\gamma^2 n}} - \frac{2\beta\log\frac{1}{\delta}}{3n}.$$

We deduce (using a union bound)

$$\gamma(P - \widehat{P})V^* \leq \sqrt{c'_{pv}\frac{\sigma_{V^*}}{n}} + b'_{pv}\mathbf{1}, \tag{4.14}$$

$$\gamma(P - \widehat{P})V^* \geq -\sqrt{c'_{pv}\frac{\sigma_{V^*}}{n}} - b'_{pv}\mathbf{1}, \tag{4.15}$$

where $c'_{pv} \triangleq 2\log(N/\delta)$ and $b'_{pv} \triangleq 2\gamma\beta\log(N/\delta)/3n$. The result then follows by plugging (4.9) and (4.10) into (4.14) and (4.15), respectively, and then taking a union bound. ■

We now state the key lemma of this section which shows that for any policy π the variance Σ^π satisfies the following Bellman-like recursion. Later, we use this result, in Lemma 4.8, to bound $(I-\gamma P^\pi)^{-1}\sigma_{V^\pi}$.

Lemma 4.7. *Σ^π satisfies the Bellman equation*

$$\Sigma^\pi = \sigma_{V^\pi} + \gamma^2 P^\pi \Sigma^\pi. \tag{4.16}$$

Proof

For all $z \in \mathcal{Z}$ we have[i]

$$\begin{aligned}
\Sigma^\pi(z) &= \mathbb{E}\left[\left|\sum_{t\geq 0}\gamma^t r(Z_t) - Q^\pi(z)\right|^2\right] \\
&= \mathbb{E}_{Z_1\sim P^\pi(\cdot|z)}\mathbb{E}\left[\left|\sum_{t\geq 1}\gamma^t r(Z_t) - \gamma Q^\pi(Z_1) - (Q^\pi(z) - r(z) - \gamma Q^\pi(Z_1))\right|^2\right] \\
&= \gamma^2 \mathbb{E}_{Z_1\sim P^\pi(\cdot|z)}\mathbb{E}\left[\left|\sum_{t\geq 1}\gamma^{t-1} r(Z_t) - Q^\pi(Z_1)\right|^2\right] \\
&\quad - 2\mathbb{E}_{Z_1\sim P^\pi(\cdot|z)}\left[(Q^\pi(z) - r(z) - \gamma Q^\pi(Z_1))\mathbb{E}\left(\sum_{t\geq 1}\gamma^t r(Z_t) - \gamma Q^\pi(Z_1)\middle| Z_1\right)\right] \\
&\quad + \mathbb{E}_{Z_1\sim P^\pi(\cdot|z)}(|Q^\pi(z) - r(z) - \gamma Q^\pi(Z_1)|^2) \\
&= \gamma^2 \mathbb{E}_{Z_1\sim P^\pi(\cdot|z)}\mathbb{E}\left[\left|\sum_{t\geq 1}\gamma^{t-1} r(Z_t) - Q^\pi(Z_1)\right|^2\right] + \gamma^2 \mathbb{V}_{Y_1\sim P(\cdot|z)}(Q^\pi(Y_1, \pi(Y_1))) \\
&= \gamma^2 [P^\pi \Sigma^\pi](z) + \sigma_{V^\pi}(z),
\end{aligned}$$

in which we rely on $\mathbb{E}(\sum_{t\geq 1}\gamma^t r(Z_t) - \gamma Q^\pi(Z_1)|Z_1) = 0$.

■

Based on Lemma 4.7, one can prove the following result on the discounted variance.

Lemma 4.8.

$$\|(I-\gamma^2 P^\pi)^{-1}\sigma_{V^\pi}\| = \|\Sigma^\pi\| \leq \beta^2, \tag{4.17}$$

$$\|(I-\gamma P^\pi)^{-1}\sqrt{\sigma_{V^\pi}}\| \leq 2\log(2)\|\sqrt{\beta\Sigma^\pi}\| \leq 2\log(2)\beta^{1.5}. \tag{4.18}$$

[i] For ease of exposition, we slightly abuse the notation here by treating P^π as a probability distribution over state-action pairs.

Proof

The first inequality follows from Lemma 4.7 by solving (4.16) in terms of Σ^π and taking the sup-norm over both sides of the resulted equation. In the case of Eq. (4.18) we have

$$\begin{aligned}
\|(I-\gamma P^\pi)^{-1}\sqrt{\sigma_{V^\pi}}\| &= \left\|\sum_{k\geq 0}(\gamma P^\pi)^k\sqrt{\sigma_{V^\pi}}\right\| \\
&= \left\|\sum_{l\geq 0}(\gamma P^\pi)^{tl}\sum_{j=0}^{t-1}(\gamma P^\pi)^j\sqrt{\sigma_{V^\pi}}\right\| \\
&\leq \sum_{l\geq 0}(\gamma^t)^l\left\|\sum_{j=0}^{t-1}(\gamma P^\pi)^j\sqrt{\sigma_{V^\pi}}\right\| \\
&= \frac{1}{1-\gamma^t}\left\|\sum_{j=0}^{t-1}(\gamma P^\pi)^j\sqrt{\sigma_{V^\pi}}\right\|,
\end{aligned} \tag{4.19}$$

in which we write $k = tl + j$ with t any positive integer.[j] We now prove a bound on $\left\|\sum_{j=0}^{t-1}(\gamma P^\pi)^j\sqrt{\sigma_{V^\pi}}\right\|$ by making use of Jensen's inequality, Cauchy-Schwarz inequality and Eq. (4.17):

$$\begin{aligned}
\left\|\sum_{j=0}^{t-1}(\gamma P^\pi)^j\sqrt{\sigma_{V^\pi}}\right\| &\leq \left\|\sum_{j=0}^{t-1}\gamma^j\sqrt{(P^\pi)^j\sigma_{V^\pi}}\right\| \leq \sqrt{t}\left\|\sqrt{\sum_{j=0}^{t-1}(\gamma^2 P^\pi)^j\sigma_{V^\pi}}\right\| \\
&\leq \sqrt{t}\left\|\sqrt{(I-\gamma^2 P^\pi)^{-1}\sigma_{V^\pi}}\right\| = \|\sqrt{t\Sigma^\pi}\|.
\end{aligned} \tag{4.20}$$

The result then follows by plugging (4.20) into (4.19) and optimizing the bound in terms of t to achieve the best dependency on β.

■

Now, we make use of Lemma 4.8 and Lemma 4.6 to bound $\|Q^* - \widehat{Q}^*\|$ in high probability.

Lemma 4.9. *Let Assumption 4.1 hold. Then, for any* $0 < \delta < 1$*:*

[j] For any real-valued function f, $\sqrt{f}$ is defined as a component wise squared-root operator on f.

$$\|Q^* - \widehat{Q}^*\| \leq \varepsilon',$$

w.p. at least $1-\delta$*, where* ε' *is defined as*

$$\varepsilon' \triangleq \sqrt{\frac{4\beta^3 \log \frac{4N}{\delta}}{n}} + \left(\frac{5(\gamma\beta^2)^{4/3} \log \frac{12N}{\delta}}{n}\right)^{3/4} + \frac{3\beta^3 \log \frac{24N}{\delta}}{n}. \tag{4.21}$$

Proof

By incorporating the result of Lemma 4.6 and Lemma 4.8 into Lemma 4.3 and taking in to account that $(I - \gamma \widehat{P}^{\pi^*})^{-1}\mathbf{1} = \beta\mathbf{1}$, we deduce [k]

$$\begin{aligned} Q^* - \widehat{Q}^* &\leq b\mathbf{1}, \\ Q^* - \widehat{Q}^* &\geq -b\mathbf{1}, \end{aligned} \tag{4.22}$$

w.p. at least $1-\delta$. The scalar b is given by

$$b \triangleq \sqrt{\frac{4\beta^3 \log \frac{2N}{\delta}}{n}} + \left(\frac{5(\gamma\beta^2)^{4/3} \log \frac{6N}{\delta}}{n}\right)^{3/4} + \frac{3\beta^3 \log \frac{12N}{\delta}}{n}. \tag{4.23}$$

The result then follows by combining these two bounds using a union bound and taking the ℓ_∞ norm.

■

Proof of Theorem 4.1

We define the total error $\varepsilon = \varepsilon' + \gamma^k\beta$ which bounds $\|Q^* - Q_k\| \leq \|Q^* - \widehat{Q}^*\| + \|\widehat{Q}^* - Q_k\|$ in high probability (ε' is defined in Lemma 4.9). The results then follows by solving this bound w.r.t. n and k and then quantifying the total number of samples $T = nN$.

■

We now draw our attention to the proof of Theorem 4.2, for which we need the following component-wise bound on $Q^* - Q^{\pi_k}$.

[k] Note that Lemma 4.8 implies $(I - \gamma P^\pi)^{-1}\sqrt{\sigma_{V^\pi}} \leq 2\log(2)\beta^{1.5}\mathbf{1}$ for any policy π.

Lemma 4.10. *Let Assumption 4.1 hold. Then w.p. at least* $1-\delta$

$$Q^* - Q^{\pi_k} \leq \widehat{Q}^{\pi_k} - Q^{\pi_k} + (b + 2\gamma^{k+1}\beta^2)\mathbf{1},$$

where b *is defined by* (4.23).

Proof

We make use of Lemma 4.2 and Lemma 4.9 to prove the result:

$$\begin{aligned} Q^* - Q^{\pi_k} &= Q^* - \widehat{Q}^* + \widehat{Q}^* - \widehat{Q}^{\pi_k} + \widehat{Q}^{\pi_k} - Q^{\pi_k} \\ &\leq b\mathbf{1} + \widehat{Q}^* - \widehat{Q}^{\pi_k} + \widehat{Q}^{\pi_k} - Q^{\pi_k} && \text{by Eq. (4.22)} \\ &\leq (b + 2\gamma^{k+1}\beta^2)\mathbf{1} + \widehat{Q}^{\pi_k} - Q^{\pi_k} && \text{by Lemma 4.2.} \end{aligned}$$

■

Lemma 4.10 states that w.h.p. $Q^* - Q^{\pi_k}$ is close to $\widehat{Q}^{\pi_k} - Q^{\pi_k}$ for large values of k and n. Therefore, to prove the result of Theorem 4.2 we only need to bound $\widehat{Q}^{\pi_k} - Q^{\pi_k}$ in high probability:

Lemma 4.11 (Component-wise upper bound on $\widehat{Q}^{\pi_k} - Q^{\pi_k}$)**.**

$$\widehat{Q}^{\pi_k} - Q^{\pi_k} \leq \gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P - \widehat{P})V^* + \gamma\beta\|(P - \widehat{P})(V^* - V^{\pi_k})\|\mathbf{1}. \quad (4.24)$$

Proof

We prove this result using a similar argument as in the proof of Lemma 4.3:

$$\begin{aligned} \widehat{Q}^{\pi_k} - Q^{\pi_k} &= (I - \gamma\widehat{P}^{\pi_k})^{-1}r - (I - \gamma P^{\pi_k})^{-1}r \\ &= \gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P^{\pi_k} - \widehat{P}^{\pi_k})Q^{\pi_k} \\ &= \gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P - \widehat{P})V^{\pi_k} \\ &= \gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P - \widehat{P})V^* + \gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P - \widehat{P})(V^{\pi_k} - V^*) \\ &\leq \gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P - \widehat{P})V^* + \gamma\beta\|(P - \widehat{P})(V^* - V^{\pi_k})\|\mathbf{1}. \end{aligned}$$

■

Now we bound the terms in the RHS of Eq. (4.24) in high probability. We begin by bounding $\gamma(I - \gamma\widehat{P}^{\pi_k})^{-1}(P - \widehat{P})V^*$:

Lemma 4.12. *Let Assumption 4.1 hold. Then, w.p. at least* $1-\delta$ *we have*

$$\gamma(I-\gamma\widehat{P}^{\pi_k})^{-1}(P-\widehat{P})V^* \le \left(\sqrt{\frac{4\beta^3\log\frac{2N}{\delta}}{n}}+\left(\frac{5(\gamma\beta^2)^{4/3}\log\frac{6N}{\delta}}{n}\right)^{3/4}\right)\mathbf{1} + \left(\frac{3\beta^3\log\frac{12N}{\delta}}{n}+\sqrt{\frac{8\gamma^{2k+4}\beta^6\log\frac{2N}{\delta}}{n}}\right)\mathbf{1}.$$

Proof

From Lemma 4.6, w.p. at least $1-\delta$, we have

$$\begin{aligned}\gamma(P-\widehat{P})V^* &\le \sqrt{\frac{2\log\frac{2N}{\delta}\widehat{\sigma}_{\widehat{V}^*}}{n}}+b_{pv}\mathbf{1}\\ &\le \sqrt{\frac{2\log\frac{2N}{\delta}\left(\widehat{\sigma}_{\widehat{V}^{\pi_k}}+\gamma^2\|\widehat{V}^{\pi_k}-\widehat{V}^*\|^2\right)}{n}}+b_{pv}\mathbf{1}\\ &\le \sqrt{\frac{2\log\frac{2N}{\delta}\left(\widehat{\sigma}_{\widehat{V}^{\pi_k}}+\gamma^2\|\widehat{Q}^{\pi_k}-\widehat{Q}^*\|^2\right)}{n}}+b_{pv}\mathbf{1}\\ &\le \sqrt{\frac{2\log\frac{2N}{\delta}\widehat{\sigma}_{\widehat{V}^{\pi_k}}}{n}}+\left(b_{pv}+\sqrt{\frac{8\gamma^{2k+4}\beta^4\log\frac{2N}{\delta}}{n}}\right)\mathbf{1},\end{aligned} \tag{4.25}$$

where in the last line we rely on Lemma 4.2. The result then follows by combining (4.25) with the result of Lemma 4.8.

■

We now prove bound on $\|(P-\widehat{P})(V^*-\widehat{V}^{\pi_k})\|$ in high probability, for which we require the following technical result:

Lemma 4.13 (Weissman et. al. 2003)**.** *Let* ρ *be a probability distribution on the finite set* $\mathcal{X}$*. Let* $\{X_1, X_2, \cdots, X_n\}$ *be a set of i.i.d. samples distributed according to* ρ *and* $\widehat{\rho}$ *be the empirical estimation of* ρ *using this set of samples. Define* $\pi_\rho \triangleq \max_{X\subseteq\mathcal{X}}\min(\mathbb{P}_\rho(X), 1-\mathbb{P}_\rho(X))$*, where* $P_\rho(X)$ *is the probability of* X *under the distribution* ρ *and* $\varphi(p) \triangleq 1/(1-2p)\log((1-p)/p)$ *for all* $p\in[0,1/2)$ *with the convention* $\varphi(1/2)=2$*, then w.p. at least* $1-\delta$ *we have*

$$\|\rho - \widehat{\rho}\|_1 \leq \sqrt{\frac{2\log\frac{2^{|\mathcal{X}|}-2}{\delta}}{n\varphi(\pi_\rho)}} \leq \sqrt{\frac{2|\mathcal{X}|\log\frac{2}{\delta}}{n}}.$$

Lemma 4.14. *Let Assumption 4.1 hold. Then, w.p. at least* $1-\delta$ *we have*

$$\gamma\|(P-\widehat{P})(V^*-V^{\pi_k})\| \leq \sqrt{\frac{2\gamma^2|\mathcal{X}|\log\frac{2N}{\delta}}{n}}\|Q^*-Q^{\pi_k}\|.$$

Proof

From the Hölder's inequality for all $z \in \mathcal{Z}$ we have

$$\begin{aligned}\gamma|(P-\widehat{P})(V^*-V^{\pi_k})(z)| &\leq \gamma\|P(\cdot|z)-\widehat{P}(\cdot|z)\|_1\|V^*-V^{\pi_k}\| \\ &\leq \gamma\|P(\cdot|z)-\widehat{P}(\cdot|z)\|_1\|Q^*-Q^{\pi_k}\|.\end{aligned}$$

This combined with Lemma 4.13 implies that

$$\gamma|(P-\widehat{P})(V^*-V^{\pi_k})(z)| \leq \sqrt{\frac{2\gamma^2|\mathcal{X}|\log\frac{2}{\delta}}{n}}\|Q^*-Q^{\pi_k}\|.$$

The result then follows by taking union bound on all $z \in \mathcal{Z}$.

■

We now make use of the results of Lemma 4.14 and Lemma 4.12 to bound $\|Q^*-Q^{\pi_k}\|$ in high probability:

Lemma 4.15. *Let Assumption 4.1 hold. Assume that*

$$n \geq 8\gamma^2\beta^2|\mathcal{X}|\log\frac{4N}{\delta}. \tag{4.26}$$

Then, w.p. at least $1-\delta$ *we have*

$$\begin{aligned}\|Q^*-Q^{\pi_k}\| \leq 2\Bigg[\varepsilon' + 2\gamma^{k+1}\beta^2 + \sqrt{\frac{4\beta^3\log\frac{4N}{\delta}}{n}} + \left(\frac{5(\gamma\beta^2)^{4/3}\log\frac{12N}{\delta}}{n}\right)^{3/4} \\ + \frac{4\beta^3\log\frac{24N}{\delta}}{n} + \sqrt{\frac{8\gamma^{2k+4}\beta^6\log\frac{4N}{\delta}}{n}}\Bigg],\end{aligned}$$

where ε' *is defined by Eq.* (4.21).

Proof

By incorporating the result of Lemma 4.14 and Lemma 4.12 into Lemma 4.11 we deduce

$$\begin{aligned} \widehat{Q}^{\pi_k} - Q^{\pi_k} \leq & \sqrt{\frac{2\beta^2\gamma^2|\mathcal{X}|\log\frac{2N}{\delta}}{n}}\|Q^* - Q^{\pi_k}\|\mathbf{1} \\ & + \left(\sqrt{\frac{4\beta^3\log\frac{2N}{\delta}}{n}} + \left(\frac{5(\gamma\beta^2)^{4/3}\log\frac{6N}{\delta}}{n}\right)^{3/4}\right)\mathbf{1} \\ & + \left(\frac{3\beta^3\log\frac{12N}{\delta}}{n} + \sqrt{\frac{8\gamma^{2k+4}\beta^6\log\frac{2N}{\delta}}{n}}\right)\mathbf{1}, \end{aligned} \tag{4.27}$$

w.p. $1-\delta$. Eq. (4.27) combined with the result of Lemma 4.10 and a union bound implies that

$$\begin{aligned} Q^* - Q^{\pi_k} \leq & (\varepsilon' + 2\gamma^{k+1}\beta^2)\mathbf{1} + \sqrt{\frac{2\beta^2\gamma^2|\mathcal{X}|\log\frac{4N}{\delta}}{n}}\|Q^* - Q^{\pi_k}\|\mathbf{1} + \\ & + \left(\sqrt{\frac{4\beta^3\log\frac{2N}{\delta}}{n}} + \left(\frac{5(\gamma\beta^2)^{4/3}\log\frac{12N}{\delta}}{n}\right)^{3/4}\right)\mathbf{1} \\ & + \left(\frac{3\gamma\beta^3\log\frac{24N}{\delta}}{n} + \sqrt{\frac{8\gamma^{2k+4}\beta^6\log\frac{4N}{\delta}}{n}}\right)\mathbf{1}. \end{aligned}$$

By taking the ℓ_∞-norm and solving the resulted bound in terms of $\|Q^* - Q^{\pi_k}\|$ we deduce

$$\|Q^* - Q^{\pi_k}\| \leq \frac{1}{1 - \sqrt{\frac{2\beta^2\gamma^2|\mathcal{X}|\log\frac{4N}{\delta}}{n}}} \Bigg[\varepsilon' + 2\gamma^{k+1}\beta^2 + \sqrt{\frac{4\beta^3 \log\frac{4N}{\delta}}{n}} + \left(\frac{5(\gamma\beta^2)^{4/3}\log\frac{12N}{\delta}}{n}\right)^{3/4} + \frac{3\beta^3\log\frac{24N}{\delta}}{n} + \sqrt{\frac{8\gamma^{2k+4}\beta^6\log\frac{4N}{\delta}}{n}} \Bigg].$$

The choice of $n > 8\beta^2\gamma^2|\mathcal{X}|\log\frac{4N}{\delta}$ deduce the result.

■

Proof of Theorem 4.2

The result follows by solving the bound of Lemma 4.15 w.r.t. n and k, in that we also need to assume that $\varepsilon \leq c\sqrt{\frac{\beta}{\gamma|\mathcal{X}|}}$ for some $c > 0$ in order to reconcile the bound of Theorem 4.2 with Eq. (4.26).

■

4.4.2 Proof of Theorem 4.3 - The Lower-Bound

In this section, we provide the proof of Theorem 4.3. In our analysis, we rely on the likelihood-ratio method, which has been previously used to prove a lower bound for multi-armed bandits (Mannor and Tsitsiklis, 2004), and extend this approach to RL and MDPs.

We begin by defining a class of MDPs for which the proposed lower bound will be obtained (see Figure 4.1). We define the class of MDPs $\mathbb{M}$ as the set of all MDPs with the state-action space of cardinality $N = 3KL$, where K and L are positive integers. Also, we assume that for all $M \in \mathbb{M}$, the state space $\mathcal{X}$ consists of three smaller subsets $\mathcal{S}$, $\mathcal{Y}_1$ and $\mathcal{Y}_2$. The set $\mathcal{S}$ includes K states, each of those states corresponds with the set of actions $\mathcal{A} = \{a_1, a_2, \ldots, a_L\}$, whereas the states in $\mathcal{Y}^1$ and $\mathcal{Y}^2$ are single-action states. By taking the action $a \in \mathcal{A}$ from every state $x \in \mathcal{S}$, we move to the next state $y(z) \in \mathcal{Y}^1$ with the probability 1, where $z = (x, a)$. The transition probability from $\mathcal{Y}^1$ is characterized by the transition probability p_M from every $y(z) \in \mathcal{Y}^1$ to itself and with the probability

$1 - p_M$ to the corresponding $y(z) \in \mathcal{Y}^2$. We notice that every state $y \in \mathcal{Y}^2$ is only connected to one state in $\mathcal{Y}^1$ and $\mathcal{S}$, i.e., there is no overlapping path in the MDP. Further, for all $M \in \mathbb{M}$, $\mathcal{Y}^2$ consists of only absorbing states, i.e., for all $y \in \mathcal{Y}^2$, $P(y|y) = 1$. The instant reward r is set to 1 for every state in $\mathcal{Y}^1$ and 0 elsewhere. For this class of MDPs, the optimal action-value function Q^*_M can be solved in closed form from the Bellman equation. For all $M \in \mathbb{M}$

$$Q^*_M(z) \triangleq \gamma V^*(y(z)) = \frac{\gamma}{1 - \gamma p_M}, \qquad \forall z \in \mathcal{S} \times \mathcal{A}.$$

Now, let us consider two MDPs M_0 and M_1 in $\mathbb{M}$ with the transition probabilities

$$p_M = \begin{cases} p & M = M_0, \\ p + \alpha & M = M_1, \end{cases}$$

where α and p are some positive numbers such that $0 < p < p + \alpha \leq 1$, to be quantified later in this section. We denote the set $\{M_0, M_1\} \subset \mathbb{M}$ with $\mathbb{M}^*$.

In the rest of this section, we concentrate on proving a lower bound on $\|Q^*_M - Q^{\mathfrak{A}}_T\|$ for all $M \in \mathbb{M}^*$, where $Q^{\mathfrak{A}}_T$ is the output of Algorithm $\mathfrak{A}$ after observing T state-transition samples. It turns out that a lower-bound on the sample complexity of $\mathbb{M}^*$ also bounds the sample complexity of $\mathbb{M}$ from below. In the sequel, we make use of the notation $\mathbb{E}_m$ ad $\mathbb{P}_m$ for the expectation and the probability under the model $M_m : m \in \{0, 1\}$, respectively.

We follow the following steps in the proof: **(i)** we prove a lower bound on the sample-complexity of learning the action-value function for every state-action pair $z \in \mathcal{S} \times \mathcal{A}$ on the class of MDP $\mathbb{M}^*$ **(ii)** we then make use of the fact that the estimates of $Q^*(z)$ for different $z \in \mathcal{S} \times \mathcal{A}$ are independent of each others to combine the bounds for all $z \in \mathcal{S} \times \mathcal{A}$ and prove the tight result of Theorem 4.3.

We begin our analysis of the lower bound by proving a lower-bound on the probability of failure of any RL algorithm to estimate a near-optimal action-value function for every state-action pair $z \in \mathcal{S} \times \mathcal{A}$. In order to prove this result (Lemma 4.17) we need to introduce some new notation: we define $Q^{\mathfrak{A}}_t(z)$ as the output of Algorithm $\mathfrak{A}$ using $t > 0$ transition samples from the state $y(z) \in \mathcal{Y}^1$ for all $z \in \mathcal{S} \times \mathcal{A}$. We also define the event $\mathcal{E}_1(z) \triangleq \{|Q^*_{M_0}(z) - Q^{\mathfrak{A}}_t(z)| \leq \varepsilon\}$ for all $z \in \mathcal{S} \times \mathcal{A}$. We then define $k \triangleq r_1 + r_2 + \cdots + r_t$ as the sum of rewards of making t transitions from $y(z) \in \mathcal{Y}^1$. We also introduce the event $\mathcal{E}_2(z)$, for all $z \in \mathcal{S} \times \mathcal{A}$ as

$$\mathcal{E}_2(z) \triangleq \left\{ pt - k \leq \sqrt{2p(1-p)t \log \frac{c_2'}{2\theta}} \right\},$$

where $\theta \triangleq \exp\big(-c_1'\alpha^2 t/(p(1-p))\big)$, and c_1' and c_2' are positive constants (will be quantified later in this section). Further, we define $\mathcal{E}(z) \triangleq \mathcal{E}_1(z) \cap \mathcal{E}_2(z)$.

We also make use of the following technical lemma which bounds the probability of the event $\mathcal{E}_2(z)$ from below:

Lemma 4.16. *For all $p > \frac{1}{2}$ and every $z \in \mathcal{S} \times \mathcal{A}$, we have*

$$\mathbb{P}_0(\mathcal{E}_2(z)) > 1 - \frac{2\theta}{c_2'}.$$

Proof

We make use of the Chernoff-Hoeffding bound for Bernoulli's (Hagerup and Rüb, 1990) to prove the result: for $p > \frac{1}{2}$, define $\varepsilon = \sqrt{2p(1-p)t \log \frac{c_2'}{2\theta}}$, we then have

$$\begin{aligned}
\mathbb{P}_0(\mathcal{E}_2(z)) &> -\exp\left(-\frac{\mathrm{KL}(p+\varepsilon||p)}{t}\right) \\
&\geq 1 - \exp\left(-\frac{\varepsilon^2}{2tp(1-p)}\right) \\
&= 1 - \exp\left(-\frac{2tp(1-p)\log\frac{c_2'}{2\theta}}{2tp(1-p)}\right), \qquad \forall z \in \mathcal{S} \times \mathcal{A}, \\
&= 1 - \exp\left(-\log\frac{c_2'}{2\theta}\right) = 1 - \frac{2\theta}{c_2'}
\end{aligned}$$

where $\mathrm{KL}(p||q) \triangleq p\log(p/q) + (1-p)\log((1-p)/(1-q))$ denotes the Kullback-Leibler divergence between p and q.

■

We now state the key result of this section:

Lemma 4.17. *For every RL Algorithm $\mathfrak{A}$ and every $z \in \mathcal{S} \times \mathcal{A}$, there exists an MDP $M_m \in \mathbb{M}^*$ and constants $c_1' > 0$ and $c_2' > 0$ such that*

$$\mathbb{P}_m(|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)|) > \varepsilon) > \frac{\theta}{c'_2}, \tag{4.28}$$

by the choice of $\alpha = 2(1-\gamma p)^2\varepsilon/(\gamma^2)$.

Proof

To prove this result we make use of a contradiction argument, i.e., we assume that there exists an algorithm $\mathfrak{A}$ for which:

$$\begin{aligned} &\mathbb{P}_m((|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)|) > \varepsilon) \leq \frac{\theta}{c'_2} \\ &\textbf{or} \\ &\mathbb{P}_m((|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)|) \leq \varepsilon) \geq 1 - \frac{\theta}{c'_2}, \end{aligned} \tag{4.29}$$

for all $M_m \in \mathbb{M}^*$ and show that this assumption leads to a contradiction.

By the assumption that $\mathbb{P}_m(|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)|) > \varepsilon) \leq \theta/c'_2$ for all $M_m \in \mathbb{M}^*$, we have $\mathbb{P}_0(\mathcal{E}_1(z)) \geq 1 - \theta/c'_2 \geq 1 - 1/c'_2$. This combined with Lemma 4.16 and by the choice of $c'_2 = 6$ implies that, for all $z \in \mathcal{S} \times \mathcal{A}$, $\mathbb{P}_0(\mathcal{E}(z)) > 1/2$. Based on this result we now prove a bound from below on $\mathbb{P}_1(\mathcal{E}_1(z))$.

We define W as the history of all the outcomes of trying z for t times and the likelihood function $L_m(w)$ for all $M_m \in \mathbb{M}^*$ as

$$L_m(w) \triangleq \mathbb{P}_m(W = w),$$

for every possible history w and $M_m \in \mathbb{M}^*$. This function can be used to define a random variable $L_m(W)$, where W is the sample path of the random process (the sequence of observed transitions). The likelihood ratio of the event W between two MDPs M_1 and M_0 can then be written as

$$\begin{aligned} \frac{L_1(W)}{L_0(W)} &= \frac{(p+\alpha)^k(1-p-\alpha)^{t-k}}{p^k(1-p)^{t-k}} = \big(1+\frac{\alpha}{p}\big)^k\big(1-\frac{\alpha}{1-p}\big)^{t-k} \\ &= \big(1+\frac{\alpha}{p}\big)^k\big(1-\frac{\alpha}{1-p}\big)^{k\frac{1-p}{p}}\big(1-\frac{\alpha}{1-p}\big)^{t-\frac{k}{p}}. \end{aligned}$$

Now, by making use of $\log(1-u) \geq -u-u^2$ for $0 \leq u \leq 1/2$, and $\exp(-u) \geq 1-u$ for $0 \leq u \leq 1$, we have

$$\begin{aligned}\big(1-\frac{\alpha}{1-p}\big)^{(1-p)/p} &\geq \exp\left(\frac{1-p}{p}\big(-\frac{\alpha}{1-p}-(\frac{\alpha}{1-p})^2\big)\right)\\ &\geq \left(1-\frac{\alpha}{p}\right)\left(1-\frac{\alpha^2}{p(1-p)}\right),\end{aligned}$$

for $\alpha \leq (1-p)/2$. Thus

$$\begin{aligned}\frac{L_1(W)}{L_0(W)} &\geq \big(1-\frac{\alpha^2}{p^2}\big)^k\big(1-\frac{\alpha^2}{p(1-p)}\big)^k\big(1-\frac{\alpha}{1-p}\big)^{t-\frac{k}{p}}\\ &=\geq \big(1-\frac{\alpha^2}{p^2}\big)^t\big(1-\frac{\alpha^2}{p(1-p)}\big)^t\big(1-\frac{\alpha}{1-p}\big)^{t-\frac{k}{p}},\end{aligned}$$

since $k \leq t$.

Using $\log(1-u) \geq -2u$ for $0 \leq u \leq 1/2$, we have for $\alpha^2 \leq p(1-p)$,

$$\big(1-\frac{\alpha^2}{2p(1-p)}\big)^t \geq \exp\big(-2t\frac{\alpha^2}{p(1-p)}\big) \geq (2\theta/c_2')^{2/c_1'},$$

and for $\alpha^2 \leq p^2/2$, we have

$$\big(1-\frac{\alpha^2}{p^2}\big)^t \geq \exp\big(-t\frac{2\alpha^2}{p^2}\big) \geq (2\theta/c_2')^{2(1-p)/(pc_1')},$$

on $\mathcal{E}_2$. Further, we have $t-k/p \leq \sqrt{2\frac{1-p}{p}t\log(c_2/(2\theta))}$, thus for $\alpha \leq (1-p)/2$:

$$\begin{aligned}\big(1-\frac{\alpha}{1-p}\big)^{t-\frac{k}{p}} &\geq \big(1-\frac{\alpha}{1-p}\big)^{\sqrt{2\frac{1-p}{p}t\log(c_2'/2\theta)}}\\ &\geq \exp\left(-\sqrt{2\frac{\alpha^2}{p(1-p)}t\log(c_2'/(2\theta))}\right)\\ &\geq \exp\Big(-\sqrt{2/c1}\log(c_2'/\theta)\Big) = (2\theta/c_2')^{\sqrt{2/c_1'}}.\end{aligned}$$

We then deduce that

$$\frac{L_1(W)}{L_2(W)} \geq (2\theta/c_2')^{2/c_1'+2(1-p)/(pc_1')+\sqrt{2/c_1'}} \geq 2\theta/c_2',$$

for the choice of $c_1' = 8$. Thus

$$\frac{L_1(W)}{L_0(W)}\mathbb{1}_{\mathcal{E}} \geq 2\theta/c_2'\mathbb{1}_{\mathcal{E}},$$

where $\mathbb{1}_{\mathcal{E}}$ is the indicator function of the event $\mathcal{E}(z)$. Then by a change of measure we deduce

$$\begin{aligned}\mathbb{P}_1(\mathcal{E}_1(z)) \geq \mathbb{P}_1(\mathcal{E}(z)) = \mathbb{E}_1[\mathbb{1}_{\mathcal{E}}] = \mathbb{E}_0\left(\frac{L_1(W)}{L_0(W)}\mathbb{1}_{\mathcal{E}}\right) \\ \geq \mathbb{E}_0\left[2\theta/c_2'\mathbb{1}_{\mathcal{E}}\right] = 2\theta/c_2'\mathbb{P}_0(\mathcal{E}(z)) > \theta/c_2',\end{aligned} \tag{4.30}$$

where we make use of the fact that $\mathbb{P}_0(\mathcal{Q}(z)) > \frac{1}{2}$.

By the choice of $\alpha = 2(1-\gamma p)^2\varepsilon/(\gamma^2)$, we have $\alpha \leq (1-p)/2 \leq p(1-p) \leq p/\sqrt{2}$ whenever $\varepsilon \leq \frac{1-p}{4\gamma^2(1-\gamma p)^2}$. For this choice of α, we have that $Q^*_{M_1}(z) - Q^*_{M_0}(z) = \frac{\gamma}{1-\gamma(p+\alpha)} - \frac{\gamma}{1-\gamma p} > 2\varepsilon$, thus $Q^*_{M_0}(z) + \varepsilon < Q^*_{M_1}(z) - \varepsilon$. In words, the random event $\{|Q^*_{M_0}(z) - Q(z)| \leq \varepsilon\}$ does not overlap with the event $\{|Q^*_{M_1}(z) - Q(z)| \leq \varepsilon\}$.

Now let us return to the assumption of Eq. (4.29), which states that for all $M_m \in \mathbb{M}^*$, $\mathbb{P}_m(|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)|) \leq \varepsilon) \geq 1 - \theta/c_2'$ under Algorithm $\mathfrak{A}$. Based on Eq. (4.30), we have $\mathbb{P}_1(|Q^*_{M_0}(z) - Q^{\mathfrak{A}}_t(z)| \leq \varepsilon) > \theta/c_2'$. This combined with the fact that $\{|Q^*_{M_0}(z) - Q^{\mathfrak{A}}_t(z)|\}$ and $\{|Q^*_{M_1}(z) - Q^{\mathfrak{A}}_t(z)|\}$ do not overlap implies that $\mathbb{P}_1(|Q^*_{M_1}(z) - Q^{\mathfrak{A}}_t(z)|) \leq \varepsilon) \leq 1 - \theta/c_2'$, which violates the assumption of Eq. (4.29). Therefore, the lower bound of Eq. (4.28) shall hold.

■

Based on the result of Lemma 4.17 and by the choice of $p = \frac{4\gamma-1}{3\gamma}$ and $c_1 = 8100$, we have that for every $\varepsilon \in (0, 3]$ and for all $0.4 = \gamma_0 \leq \gamma < 1$ there exists an MDP $M_m \in \mathbb{M}^*$ such that

$$\mathbb{P}_m(|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)) > \varepsilon) > \frac{1}{c_2'}\exp\left(\frac{-c_1 t\varepsilon^2}{6\beta^3}\right),$$

This result implies that for any state-action pair $z \in \mathcal{S} \times \mathcal{A}$:

$$\mathbb{P}_m(|Q^*_{M_m}(z) - Q^{\mathfrak{A}}_t(z)| > \varepsilon) > \delta, \tag{4.31}$$

on M_0 or M_1 whenever the number of transition samples t is less than $\xi(\varepsilon, \delta) \triangleq \frac{6\beta^3}{c_1\varepsilon^2}\log\frac{1}{c_2'\delta}$.

Based on this result, we prove a lower bound on the number of samples T for which $\|Q^*_{M_m} - Q^{\mathfrak{A}}_T\| > \varepsilon$ on either M_0 or M_1:

Lemma 4.18. *For any $\delta' \in (0, 1/2)$ and any Algorithm $\mathfrak{A}$ using a total number of transition samples less than $T = \frac{N}{6}\xi\big(\varepsilon, \frac{12\delta'}{N}\big)$, there exists an MDP $M_m \in \mathbb{M}^*$ such that*

$$\mathbb{P}_m\big(\|Q^*_{M_m} - Q^{\mathfrak{A}}_T\| > \varepsilon\big) > \delta'. \tag{4.32}$$

Proof

First, we note that if the total number of observed transitions is less than $(KL/2)\xi(\varepsilon,\delta) = (N/6)\xi(\varepsilon,\delta)$, then there exists at least $KL/2 = N/6$ state-action pairs that are sampled at most $\xi(\varepsilon,\delta)$ times. Indeed, if this was not the case, then the total number of transitions would be strictly larger than $N/6\xi(\varepsilon,\delta)$, which implies a contradiction). Now let us denote those states as $z_{(1)}, \ldots, z_{(N/6)}$.

In order to prove that (4.32) holds for every RL algorithm, it is sufficient to prove it for the class of algorithms that return an estimate $Q^{\mathfrak{A}}_{T_z}(z)$, where T_z is the number of samples collected from z, for each state-action z based on the transition samples observed from z only.[1] This is due to the fact that the samples from z and z' are independent. Therefore, the samples collected from z' do not bring more information about $Q^*_M(z)$ than the information brought by the samples collected from z. Thus, by defining $\mathcal{Q}(z) \triangleq \{|Q^*_M(z) - Q^{\mathfrak{A}}_{T_z}(z)| > \varepsilon\}$ for all $M \in \mathbb{M}^*$we have that for such algorithms, the events $\mathcal{Q}(z)$ and $\mathcal{Q}(z')$ are conditionally independent given T_z and $T_{z'}$. Thus, there exists an MDP $M_m \in \mathbb{M}^*$ such that

$$\begin{aligned}
& \mathbb{P}_m\Big(\{\mathcal{Q}(z_{(i)})^c\}_{1\le i\le N/6} \cap \{T_{z(i)} \le \xi(\varepsilon,\delta)\}_{1\le i\le N/6}\Big) \\
= & \sum_{t_1=0}^{\xi(\varepsilon,\delta)} \cdots \sum_{t_{N/6}=0}^{\xi(\varepsilon,\delta)} \mathbb{P}_m\Big(\{T_{z(i)} = t_i\}_{1\le i\le N/6}\Big) \\
& \mathbb{P}_m\Big(\{\mathcal{Q}(z_{(i)})^c\}_{1\le i\le N/6} \cap \{T_{z(i)} = t_i\}_{1\le i\le N/6}\Big) \\
= & \sum_{t_1=0}^{\xi(\varepsilon,\delta)} \cdots \sum_{t_{N/6}=0}^{\xi(\varepsilon,\delta)} \mathbb{P}_m\Big(\{T_{z(i)} = t_i\}_{1\le i\le N/6}\Big) \prod_{1\le i\le N/6} \mathbb{P}_m\Big(\mathcal{Q}(z_{(i)})^c \cap T_{z(i)} = t_i\Big) \\
\le & \sum_{t_1=0}^{\xi(\varepsilon,\delta)} \cdots \sum_{t_{N/6}=0}^{\xi(\varepsilon,\delta)} \mathbb{P}_m\Big(\{T_{z(i)} = t_i\}_{1\le i\le N/6}\Big)(1-\delta)^{N/6},
\end{aligned}$$

[1] We let T_z to be random.

from Eq. (4.31), thus

$$\mathbb{P}_m\Big(\{\Omega(z_{(i)})^c\}_{1\leq i\leq N/6}\big|\{T_{z(i)}\leq\xi(\varepsilon,\delta)\}_{1\leq i\leq N/6}\Big)\leq(1-\delta)^{N/6}.$$

We finally deduce that if the total number of transition samples is less than $\frac{N}{6}\xi(\varepsilon,\delta)$, then

$$\begin{aligned}\mathbb{P}_m(\|Q^*_{M_m}-Q^{\mathfrak{A}}_T\|>\varepsilon\Big)&\geq\mathbb{P}_m\Big(\bigcup_{z\in\mathcal{S}\times\mathcal{A}}\Omega(z)\Big)\\&\geq 1-\mathbb{P}_m\Big(\{\Omega(z_{(i)})^c\}_{1\leq i\leq N/6}\big|\{T_{z(i)}\leq\xi(\varepsilon,\delta)\}_{1\leq i\leq N/6}\Big)\\&\geq 1-(1-\delta)^{N/6}\geq\frac{\delta N}{12},\end{aligned}$$

whenever $\frac{\delta N}{6}\leq 1$. Setting $\delta'=\frac{\delta N}{12}$, we obtain the desired result.

■

Lemma 4.18 implies that if the total number of samples T is less than $\beta^3 N/(c_1\varepsilon^2)\log(N/(c_2\delta))$, with the choice of $c_1=8100$ and $c_2=72$, then the probability of $\|Q^*_M-Q^{\mathfrak{A}}_T\|\leq\varepsilon$ is at maximum $1-\delta$ on either M_0 or M_1. This is equivalent to the argument that for every RL algorithm $\mathfrak{A}$ to be (ε,δ)-correct on the set $\mathbb{M}^*$, and subsequently on the class of MDPs $\mathbb{M}$, the total number of transitions T needs to satisfy the inequality $T\geq\beta^3 N/(c_1\varepsilon^2)\log(N/(c_2\delta))$, which concludes the proof of Theorem 4.3.

4.5 Conclusion and Future Works

In this chapter, we have presented the first minimax bound on the sample complexity of estimating the optimal action-value function in discounted reward MDPs. We have proven that both model-based Q-value iteration (QVI) and model-based policy iteration (PI), in the presence of the generative model of the MDP, are optimal in the sense that the dependency of their performances on $1/\varepsilon$, N, δ and $1/(1-\gamma)$ matches the lower bound of RL. Also, our results have significantly improved on the state-of-the-art in terms of dependency on $1/(1-\gamma)$.

Overall, we conclude that both QVI and PI are efficient RL algorithms in terms of the number of samples required to attain a near optimal solution as the upper bounds on the performance loss of both algorithms completely match the lower bound of RL up to a multiplicative factor.

In this chapter, we only consider the problem of estimating the optimal action-value function when a generative model of the MDP is available. This allows us to make an accurate estimate of the state-transition distribution for all state-action pairs and then estimate the optimal control policy based on this empirical model. This is in contrast to the online RL setup in which the choice of the exploration policy has an influence on the behavior of the learning algorithm and vise-versa. Therefore, we do not compare our results with those of online RL algorithms such as PAC-MDP (Szita and Szepesvári, 2010; Strehl et al., 2009), upper-confidence-bound reinforcement learning (UCRL) (Jaksch et al., 2010b) and REGAL of Bartlett and Tewari (2009). However, we believe that it would be possible to improve on the state-of-the-art in PAC-MDP, based on the results of this chapter. This is mainly due to the fact that most PAC-MDP algorithms rely on an extended variant of model-based Q-value iteration to estimate the action-value function. However, those results bound the estimation error in terms of $V_{\max}$ rather than the total variance of discounted reward which leads to a non-tight sample complexity bound. One can improve on those results, in terms of dependency on $1/(1-\gamma)$, using the improved analysis of this chapter which makes use of the sharp result of Bernstein's inequality to bound the estimation error in terms of the variance of sum of discounted rewards. It must be pointed out that, almost contemporaneously to our work, Lattimore and Hutter (2012) have independently proven a similar upper-bound of order $\widetilde{O}(N/(\varepsilon^2(1-\gamma)^3))$ for UCRL algorithm under the assumption that only two states are accessible form any state-action pair. Their work also includes a similar lower bound of $\widetilde{\Omega}(N/(\varepsilon^2(1-\gamma)^3))$ for any RL algorithm which matches, up to a logarithmic factor, the result of Theorem 4.3.

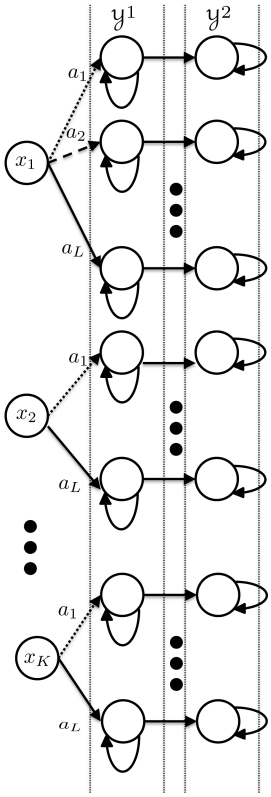

Figure 4.1: The class of MDPs considered in the proof of Theorem 4.3. Nodes represent states and arrows show transitions between the states (see the text for details).

APPENDIX A

From Bellman Equation to DPP Recursion

In this appendix we give an informal *derivation* of the DPP equation. This text 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 Section 2.3.2.

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 $\bar{\pi}$:

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

Note that $g_{\bar{\pi}}^{\pi}(x)$ 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 $V_{\bar{\pi}}^{\pi}$, 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 $x \in \mathcal{X}$:

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

Equation (A.1) is a modified version of (2.2) 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 (A.1) can be performed in closed form. Following Todorov (2006), we state Proposition A.1 (closely related results to proposition A.1 can be found in the recent works of Still and Precup, 2011; Peters et al., 2010):

Proposition A.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:*

$$V_{\bar{\pi}}^*(x) = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\big[\eta(r(x,a) + \gamma(P V_{\bar{\pi}}^*)(x,a))\big], \quad \text{(A.2)}$$

$$\bar{\pi}^*(a|x) = \frac{\bar{\pi}(a|x) \exp\big[\eta(r(x,a) + \gamma(P V_{\bar{\pi}}^*)(x,a))\big]}{\exp\left(\eta V_{\bar{\pi}}^*(x)\right)}. \quad \text{(A.3)}$$

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 $\mathcal{L}(x;\lambda_x) : \mathcal{X} \to \Re$ by adding the term $\lambda_x \big[\sum_{a\in\mathcal{A}} \pi(a|x) - 1\big]$ to the RHS of (A.1). 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.

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

The maximization in (A.1) can be expressed as maximizing the Lagrangian function $\mathcal{L}(x, \lambda_x)$. The necessary condition for the extremum with respect to $\pi(\cdot|x)$ is:

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

which leads to:

$$\bar{\pi}^*(a|x) = \bar{\pi}(a|x) \exp\left(-\eta\lambda_x - 1\right) \exp\left[\eta(r(x,a) + \gamma \left(PV^*_{\bar{\pi}}\right)(x,a))\right], \quad \forall x \in \mathcal{X}. \tag{A.4}$$

The Lagrange multipliers can then be solved from the constraints:

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

$$\lambda_x = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma \left(PV^*_{\bar{\pi}}\right)(x,a))\right] - \frac{1}{\eta}. \tag{A.5}$$

By plugging (A.5) into (A.4) we deduce

$$\bar{\pi}^*(a|x) = \frac{\bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma \left(PV^*_{\bar{\pi}}\right)(x,a))\right]}{\sum\limits_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma \left(PV^*_{\bar{\pi}}\right)(x,a))\right]}, \quad \forall (x,a) \in \mathcal{Z}. \tag{A.6}$$

The results then follows by substituting (A.6) in (A.1).

■

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:

$$V^{k+1}_{\bar{\pi}}(x) = \frac{1}{\eta} \log \sum_{a\in\mathcal{A}} \bar{\pi}(a|x) \exp\left[\eta(r(x,a) + \gamma(PV^k_{\bar{\pi}})(x,a))\right], \tag{A.7}$$

and then compute $\bar{\pi}^*$ using (A.3). $\bar{\pi}^*$ 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 (A.3). 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 (A.3), and a value function update, Equation (A.7), 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:

$$V_{\bar{\pi}}^{k+1}(x) = \frac{1}{\eta} \log \sum_{a \in \mathcal{A}} \bar{\pi}_k(a|x) \exp\big[\eta(r(x,a) + \gamma(PV_{\bar{\pi}}^k)(x,a))\big], \tag{A.8}$$

$$\bar{\pi}_{k+1}(a|x) = \frac{\bar{\pi}_k(a|x) \exp\big[\eta(r(x,a) + \gamma(PV_{\bar{\pi}}^k)(x,a))\big]}{\exp\big(\eta V_{\bar{\pi}}^{k+1}(x)\big)}. \tag{A.9}$$

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:

$$\Psi_{k+1}(x,a) \triangleq \frac{1}{\eta} \log \bar{\pi}_k(a|x) + r(x,a) + \gamma(PV_{\bar{\pi}}^k)(x,a). \tag{A.10}$$

By comparing (A.10) with (A.9) and (A.8), we deduce:

$$\bar{\pi}_k(a|x) = \frac{\exp(\eta \Psi_k(x,a))}{\sum_{a' \in \mathcal{A}} \exp(\eta \Psi_k(x,a'))}, \tag{A.11}$$

$$V_{\bar{\pi}}^k(x) = \frac{1}{\eta} \log \sum_{a \in \mathcal{A}} \exp(\eta \Psi_k(x,a))). \tag{A.12}$$

Finally, **(iii)** by plugging (A.11) and (A.12) into (A.10) we derive:

$$\Psi_{k+1}(x,a) = \Psi_k(x,a) - \mathcal{L}_\eta \Psi_k(x) + r(x,a) + \gamma(P\mathcal{L}_\eta \Psi_k)(x,a), \tag{A.13}$$

with $\mathcal{L}_\eta$ operator being defined by

$$\mathcal{L}_\eta \Psi(x) \triangleq 1/\eta \log \sum_{a \in \mathcal{A}} \exp(\eta \Psi(x,a)) \qquad \forall x \in \mathcal{X}.$$

Eq. (A.13) is one form of the DPP equations. There is an analytically more tractable version of the DPP equation, where we replace $\mathcal{L}_\eta$ by the Boltzmann soft-max $\mathcal{M}_\eta$ defined by[a]

$$\mathcal{M}_\eta\Psi(x) \triangleq \sum_{a\in\mathcal{A}} \big[\exp(\eta\Psi(x,a))\Psi(x,a)\big/ \sum_{a'\in\mathcal{A}} \exp(\eta\Psi(x,a'))\big], \quad \forall x \in \mathcal{X}.$$

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 appendix. By replacing $\mathcal{L}_\eta$ with $\mathcal{M}_\eta$ we deduce the DPP recursion:

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

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:

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

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

$$|\mathcal{L}_\eta\Psi(x) - \mathcal{M}_\eta\Psi(x)| = 1/\eta H_\pi(x) \le \frac{\log(|\mathcal{A}|)}{\eta}, \qquad \forall x\in\mathcal{X}, \qquad (A.14)$$

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 $\log(|\mathcal{A}|)/\eta$. 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 (A.14) and further readings see MacKay (2003, chap. 31).

APPENDIX B

The Convergence Proof of DPP

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

B.1 Proof of Theorem 2.1

In this section 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 bound, for all $k \geq 0$:

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

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 (B.1) 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 B.1. **(ii)** we obtain an upper bound on the

normed error $\|Q^* - Q_k\|$ in Lemma B.2. 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 appendix since we prove a more general case in Appendix 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:

$$Q_k = \frac{k-1}{k}\mathcal{T}^{\pi_{k-1}}Q_{k-1} + \frac{1}{k}\mathcal{T}^{\pi_{k-1}}Q_0, \tag{B.2}$$

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

Lemma B.1 relates Ψ_k with Q_k:

Lemma B.1. *Let k be a positive integer. Then, we have*

$$\Psi_k = kQ_k + Q_0 - \pi_{k-1}((k-1)Q_{k-1} + Q_0). \tag{B.3}$$

We then relate Q_k and Q^*:

Lemma B.2. *Let Assumption 2.1 hold k be a positive integer, also assume that $\|\Psi_0\| \leq V_{\max}$ then the following inequality holds:*

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

Lemma B.2 provides an upper bound on the normed-error $\|Q_k - Q^*\|$. We make use of Lemma B.2 to prove the main result of this Section:

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

By collecting terms we obtain

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

This combined with Lemma B.2 completes the proof.

B.2 Proof of Corollary 2.2

First, we note that Q_k converges to Q^* (Lemma B.2) and π_k converges to π^* by Corollary 2.1. 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 B.1).

Now, we compute the limit of Ψ_k. Q_k converges to Q^* with a linear rate from Lemma B.2. Also, we have $V^* = \pi^* Q^*$ by definition of V^* and Q^*. Then, by taking the limit of (B.3) 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 2.2

This appendix 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 (2.8). 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

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

where $E_k = \sum_{j=0}^{k} \varepsilon_k$ 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 2.1, we relate Q^* with Q^{π_k} via an auxiliary action-value function Q_k. In the rest of this appendix, we first express Ψ_k in terms of Q_k in Lemma C.1. Then, we obtain an upper bound on the normed error $\|Q^* - Q_k\|$ in Lemma C.5. Finally, we use these two results to derive (C.1).

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:

$$Q_k = \frac{k-1}{k}\mathcal{T}^{\pi_{k-1}}Q_{k-1} + \frac{1}{k}(\mathcal{T}^{\pi_{k-1}}Q_0 + E_{k-1}), \tag{C.2}$$

where (C.2) may be considered as an approximate version of (B.2). Lemma C.1 relates Ψ_k with Q_k:

Lemma C.1. *Let k be a positive integer and π_k denotes the policy induced by the approximate DPP at iteration k. Then we have*

$$\Psi_k = kQ_k + Q_0 - \pi_{k-1}\left((k-1)Q_{k-1} + Q_0\right). \tag{C.3}$$

Proof

We rely on induction for the proof of this theorem. The result holds for $k = 1$ since one can easily show that (C.3) reduces to (2.8). We then show that if (C.3) holds for k then it also holds for $k+1$. From (2.8) we have

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

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 (C.2).

Thus (C.3) holds for $k+1$, and is thus true for all $k \geq 1$.

■

Based on Lemma C.1, one can express the policy induced by DPP, π_k, in terms of Q_k and Q_0:

Lemma C.2. *For all $(x, a) \in \mathcal{Z}$:*

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

Proof

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

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 B.2 we establish a bound on $\|Q^* - Q_k\|$ for which we make use of the following technical results:

Lemma C.3. *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.

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

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

Lemma C.4. *Let $\eta > 0$ and k be a positive integer. Assume $\|Q_0\| \leq V_{\max}$, then the following holds:*

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

Proof

Define $A \triangleq \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\|$. We then obtain, by definition of operator $\mathcal{T}$:

$$
\begin{aligned}
A &\leq \gamma \| kP\mathcal{M}Q_k + P\mathcal{M}Q_0 - kP^{\pi_k}Q_k - P^{\pi_k}Q_0 \| \\
&= \gamma \left\| P(\mathcal{M}kQ_k + \mathcal{M}Q_0 - \pi_k(kQ_k + Q_0)) \right\| \\
&\leq \gamma \left\| \mathcal{M}kQ_k + \mathcal{M}Q_0 - \pi_k(kQ_k + Q_0) \right\| \\
&\leq \gamma \left\| 2\mathcal{M}Q_0 + \mathcal{M}(kQ_k + Q_0) - \pi_k(kQ_k + Q_0) \right\| \\
&\leq \gamma \left(2 \left\| Q_0 \right\| + \left\| \mathcal{M}(kQ_k + Q_0) - \mathcal{M}_\eta(kQ_k + Q_0) \right\| \right),
\end{aligned}
\tag{C.4}
$$

where in the last line we make use of Lemma C.2. The result then follows by comparing (C.4) with Lemma C.3.

■

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

Lemma C.5. *Let Assumption 2.1 hold. Define Q_k by* (C.2)*. Let k be a non-negative integer, also, assume that $\|\Psi_0\| \leq V_{\max}$, then the following inequality holds:*

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

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 (B.4) holds for k it also holds for $k+1$:

$$
\begin{aligned}
\|Q^* - Q_{k+1}\| &= \left\| \mathcal{T}Q^* - \left(\frac{k}{k+1} \mathcal{T}^{\pi_k} Q_k + \frac{1}{k+1} (\mathcal{T}^{\pi_k} Q_0 + E_k) \right) \right\| \\
&= \left\| \frac{1}{k+1} (\mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_0) + \frac{k}{k+1} (\mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_k) - \frac{1}{k+1} E_k \right\| \\
&= \frac{1}{k+1} \left\| \mathcal{T}Q^* - \mathcal{T}^{\pi_k} Q_0 + k(\mathcal{T}Q^* - \mathcal{T}Q_k + \mathcal{T}Q_k - \mathcal{T}^{\pi_k} Q_k) \right\| \\
&\quad + \frac{1}{k+1} \|E_k\| \\
&\leq \frac{1}{k+1} \left[\|\mathcal{T}Q^* - \mathcal{T}Q_0\| + \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\| \right] \\
&\quad + \frac{k}{k+1} \|\mathcal{T}Q^* - \mathcal{T}Q_k\| + \frac{1}{k+1} \|E_k\| \\
&\leq \frac{1}{k+1} \left[\gamma \|Q^* - Q_0\| + \|k\mathcal{T}Q_k + \mathcal{T}Q_0 - k\mathcal{T}^{\pi_k} Q_k - \mathcal{T}^{\pi_k} Q_0\| \right] \\
&\quad + \frac{\gamma k}{k+1} \|Q^* - Q_k\| + \frac{1}{k+1} \|E_k\| .
\end{aligned}
\tag{C.5}
$$

Now based on Lemma C.4 and by plugging (B.4) into (C.5) we have

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

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

Lemma C.5 proves 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

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

This combined with the result of Lemma C.5 completes the proof.

APPENDIX D

The Convergence Proof of DPP-RL

We begin this appendix 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 $\mathbb{E}(\varepsilon_k(x, a)|\mathcal{F}_{k-1}) = 0$, which means that for all $(x, a) \in \mathcal{Z}$ the sequence of estimation errors $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k\}$ is a martingale difference sequence w.r.t. the filtration $\mathcal{F}_k$. Now, we provide the proof of Lemma 2.1, on which we rely for the analysis of both Theorem 2.3 and Theorem 2.4:

Proof of Lemma 2.1

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

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

where we make use of LemmaSoftMaxDiv 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 $k \geq 0$, $\|\mathcal{T}_k^{\pi_k}\Psi_k\| \leq 2\gamma\log(|\mathcal{A}|)/(\eta(1-\gamma)) + V_{\max}$. The bound on ε_k is an immediate consequence of this result.

■

D.1 Proof of Theorem 2.3

In this section, we provide the proof of Theorem 2.3 which guarantees that DPP-RL asymptotically converges to the optimal policy w.p. 1.

We make use of the result of Lemma 2.1 and Corollary 2.3 to prove the theorem. We begin by recalling the result of Corollary 2.3:

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

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, 76), 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 $\{\varepsilon_1, \varepsilon_2, \dots\}$ since we already have proven the boundedness of $\|\varepsilon_k\|$ in Lemma 2.1. Thus, we deduce

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

Thus

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

The result then follows by combining (D.1) with Corollary 2.3.

D.2 Proof of Theorem 2.4

In this section, we prove Theorem 2.4, for which we rely on a maximal Azuma's inequality (see, e.g., Cesa-Bianchi and Lugosi, 2006, appendix, pg. 359):

Lemma D.1 (Azuma, 1967)**.** *Let $\mathcal{Y} = \{Y_1, Y_2, \dots, Y_K\}$ be a martingale difference sequence w.r.t. a sequence of random variables $\{X_1, X_2, \dots, X_K\}$, i.e., $\mathbb{E}(Y_{k+1}|X_1, \dots X_k) = 0$ 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 $\varepsilon > 0$, we have*

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

We recall the result of Theorem 2.2 at iteration k:

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

Note that the main difference between this bound and the result of Theorem 2.4 is just in the second term. So, to prove Theorem 2.4 we need to show that the following inequality holds, with probability at least $1-\delta$:

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

We first notice that

$$\frac{1}{k+1}\sum_{j=0}^{k}\gamma^{k-j}\left\|E_j\right\| \leq \frac{1}{k+1}\sum_{k=0}^{j}\gamma^{k-j}\max_{0\leq j\leq k}\left\|E_j\right\| \leq \frac{\max_{0\leq j\leq k}\left\|E_j\right\|}{(1-\gamma)(k+1)}. \tag{D.3}$$

Therefore, in order to prove (D.2) it is sufficient to bound $\max_{0\leq j\leq k}\left\|E_j\right\| = \max_{(x,a)\in\mathcal{Z}}$
$\max_{0\leq j\leq k}|E_{k-1}(x,a)|$ 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

$$\begin{aligned}
&\mathbb{P}\left[\max_{0\leq j\leq k}|E_j(x,a)|>\varepsilon\right] = \mathbb{P}\left[\max\left[\max_{0\leq j\leq k}(E_j(x,a)),\max_{0\leq j\leq k}(-E_j(x,a))\right]>\varepsilon\right]\\
=&\mathbb{P}\left[\left\{\max_{0\leq j\leq k}(E_j(x,a))>\varepsilon\right\}\bigcup\left\{\max_{0\leq j\leq k}(-E_j(x,a))>\varepsilon\right\}\right]\\
\leq&\mathbb{P}\left[\max_{0\leq j\leq k}(E_j(x,a))>\varepsilon\right]+\mathbb{P}\left(\max_{0\leq j\leq k}(-E_j(x,a))>\varepsilon\right),
\end{aligned} \tag{D.4}$$

The sequence of random variables $\{\varepsilon_0(x,a),\varepsilon_1(x,a),\cdots,\varepsilon_k(x,a)\}$ 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., $\mathbb{E}[\varepsilon_k(x,a)|\mathcal{F}_{k-1}]=0$. It follows from Lemma D.1 and Lemma 2.1 that for any $\varepsilon>0$ we have

$$\begin{aligned}
\mathbb{P}\left[\max_{0\leq j\leq k}(E_j(x,a))>\varepsilon\right] &\leq \exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right)\\
\mathbb{P}\left[\max_{0\leq j\leq k}(-E_j(x,a))>\varepsilon\right] &\leq \exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right).
\end{aligned} \tag{D.5}$$

By combining (D.5) with (D.4) we deduce that

$$\mathbb{P}\left[\max_{0\leq j\leq k}|E_j(x,a)|>\varepsilon\right]\leq 2\exp\left(\frac{-\varepsilon^2}{2(k+1)(\frac{4\gamma\log(|\mathcal{A}|)}{\eta(1-\gamma)}+2V_{\max})^2}\right),$$

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

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

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

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

This combined with (D.3) proves (D.2) and Theorem 2.4.

Bibliography

Antos, A., Munos, R., and Szepesvári, C. (2007). Fitted Q-iteration in continuous action-space MDPs. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*.

Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. J. (2011a). Reinforcement Learning with a Near Optimal Rate of Convergence. Technical report.

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

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

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

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

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

Bellman, R. (1957). *Dynamic Programming*. Princeton Univ. Press.

Bertsekas, D. P. (2007a). *Dynamic Programming and Optimal Control*, volume I. Athena Scientific, Belmount, Massachusetts, third edition.

Bertsekas, D. P. (2007b). *Dynamic Programming and Optimal Control*, volume II. Athena Scientific, Belmount, Massachusetts, third edition.

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

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

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

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

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

Even-Dar, E., Mannor, S., and Mansour, Y. (2002). PAC bounds for multi-armed bandit and Markov decision processes. In *15th Annual Conference on Computational Learning Theory*, pages 255–270.

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

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

Farahmand, A., Ghavamzadeh, M., Szepesvári, C., and Mannor, S. (2008a). Regularized fitted Q-iteration: Application to planning. In *EWRL*, pages 55–68.

Farahmand, A., Ghavamzadeh, M., Szepesvári, C., and Mannor, S. (2008b). Regularized policy iteration. In *NIPS*, pages 441–448.

Farahmand, A., Munos, R., and Szepesvari, C. (2010). Error propagation in approximate value and policy iteration. In *Neural Information Processing Systems*, Vancouver, British Columbia, Canada.

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

Feller, W. (1968). *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley.

Hagerup, L. and Rüb, C. (1990). A guided tour of chernoff bounds. *Information Processing Letters*, 33:305–308.

Hoffmann-Jørgensen, J. and Pisier, G. (76). The law of large numbers and the central limit theorem in banach spaces. *The Annals of Probability*, 4(4):587–599.

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

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

Kakade, S. (2001). Natural policy gradient. In *Advances in Neural Information Processing Systems 14*, pages 1531–1538, Vancouver, British Columbia, Canada.

Kakade, S. M. (2004). *On the Sample Complexity of Reinforcement Learning*. PhD thesis, Gatsby Computational Neuroscience Unit.

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

Kearns, M. and Singh, S. (1999). Finite-sample convergence rates for Q-learning and indirect algorithms. In *Advances in Neural Information Processing Systems 12*, pages 996–1002. MIT Press.

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

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

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

Kushner, H. J. and Yin, G. G. (2003). *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2nd edition.

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

Lattimore, T. and Hutter, M. (2012). Pac bounds for discounted mdps. *CoRR*, abs/1202.3890.

MacKay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, United Kingdom, first edition.

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

Mannor, S. and Tsitsiklis, J. N. (2004). The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5:623–648.

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

Munos, R. (2003). Error bounds for approximate policy iteration. In *ICML*, pages 560–567.

Munos, R. (2005). Error bounds for approximate value iteration. In *Proceedings of the 20th National Conference on Artificial Intelligence*, volume II, pages 1006–1011, Pittsburgh, Pennsylvania.

Munos, R. and Moore, A. (1999). Influence and variance of a Markov chain : Application to adaptive discretizations in optimal control. In *Proceedings of the 38th IEEE Conference on Decision and Control*.

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

Peng, J. and Williams, R. J. (1996). Incremental multi-step Q-learning. *Machine Learning*, 22(1-3):283–290.

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

Peters, J., Mülling, K., and Altun, Y. (2010). Relative entropy policy search. In *AAAI*.

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

Puterman, M. L. (1994a). *Markov Decision Processes, Discrete Stochastic Dynamic Programming*. A Wiley-Interscience Publication.

Puterman, M. L. (1994b). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1st edition.

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

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

Sobel, M. J. (1982). The variance of discounted markov decision processes. *Journal of Applied Probability*, 19:794–802.

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

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

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

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057–1063, Denver, Colorado, USA.

Szepesvári, C. (1997). The asymptotic convergence-rate of Q-learning. In *Advances in Neural Information Processing Systems 10, Denver, Colorado, USA, 1997*.

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

Szepesvari, C. and Smart, W. (2004). Interpolation-based Q-learning. In *Proceedings of 21st International Conference on Machine Learning*, volume 69 of *ACM International Conference Proceeding Series*, page 100, Banff, Alberta, Canada.

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

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

Todorov, E. (2006). Linearly-solvable markov decision problems. In *Proceedings of the 20th Annual Conference on Neural Information Processing Systems*, pages 1369–1376, Vancouver, British Columbia, Canada.

van Hasselt, H. (2010). Double Q-learning. In *Advances in Neural Information Processing Systems 23*, pages 2613–2621.

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

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

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

Watkins, C. (1989). *Learning from Delayed Rewards*. PhD thesis, Kings College, Cambridge, England.

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

Wiering, M. and van Otterlo, M. (2012a). *Reinforcement Learning: State-of-the-Art*, chapter 1, pages 3–39. Springer.

Publications by the Author

Azar, M. G., Gómez, V., and Kappen, H. J. (2011a). Dynamic Policy Programming with Function Approximation. *Journal of Machine Learning Research - Proceedings Track (AISTATS11)*, 15:119–127.

Azar, M. G. and Kappen, H. J. (2012). Dynamic Policy Programming. *Journal of Machine Learning Research (accepted)*.

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

Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. J. (2012a). Speedy Q-Learning: A Computationally Efficient Reinforcement Learning with a Near Optimal Rate of Convergence. *Journal of Machine Learning Research (submitted)*.

Azar, M. G., Munos, R., and Kappen, H. J. (2012b). Minimax PAC Bounds on the Sample Complexity of Reinforcement Learning with a Generative Model. *Machine Learning Journal (submitted)*.

Azar, M. G., Munos, R., and Kappen, H. J. (2012c). On the Sample Complexity of Reinforcement Learning with a Generative Model. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1263–1270.

Summary

The focus of this thesis is on the theory reinforcement learning (RL) in the infinite-horizon discounted-reward Markovian decision processes (MDPs). In RL we often deal with an agent which interacts with her surrounding environment. The goal of agent is to optimize her long-term performance by finding the optimal policy, i.e., a sequence of control actions which maximizes her long-term return. The long-term return of the agent is often expressed in terms of the value function which, in the case of discounted reward MDPs, is defined as the expected value of sum of discounted rewards given to the agent by the environment.

Many RL algorithms rely on dynamic programming (DP) to estimate the optimal policy through the Bellman iteration. In the case of discounted reward MDPs, one can prove that DP converges exponentially fast to the optimal solution. However, DP requires an explicit knowledge of transition model and reward function, which in many problems of interest is not available. In the absence of the model data, the common approach is to estimate of the optimal value function empirically using Monte-Carlo sampling. In recent years, the problem of estimating the optimal policy by sampling has drawn much attention and several RL methods have been developed to address this problem. Also, in some cases, finite-time and asymptotic performance guarantees have been established for these methods.

In this thesis, we concentrate on the problem of estimating the optimal policy and the optimal value function by sampling: we develop new RL algorithms and analyze their asymptotic and finite-time performances. In addition, we refine some of the existing theoretical results for the well-known RL algorithms such as model-based value iteration and policy iteration. Further, we prove some

general lower bounds on the number of samples required by any RL algorithm to achieve a near optimal solution (sample complexity bound).

Dynamic Policy Programming

In Chapter 2, we propose a new policy iteration algorithm called dynamic policy programming (DPP), to compute the optimal policy in discounted-reward MDPs. We prove that DPP asymptotically converges to the optimal policy and we establish the rate of convergence for DPP in the tabular case. We also prove performance-loss bounds for the approximate variants of DPP in terms of the supremum norm of the average approximation error. This is in contrast to the existing results for approximate dynamic programming, which are often in terms of the supremum of the errors. We also propose a new RL algorithm to estimate the optimal policy, called DPP-RL, which relies on a sample-estimate variant of the DPP update rule and prove its convergence.

Speedy Q-Learning

In Chapter 3, we introduce a new RL algorithm, called speedy Q-learning (SQL), to address the problem of slow convergence in the standard Q-learning. We examine the finite-time and the asymptotic behavior of SQL. We prove high probability bounds on the performance loss of SQL, suggesting that the algorithm converges faster to the optimal action-value function than standard Q-learning. The numerical experiments in Section 3.4 confirm our theoretical results showing that for $\gamma \approx 1$, where γ is the discount factor of the MDP, SQL performs better than the standard Q-learning.

Sample Complexity of Model-Based RL

In Chapter 4, we establish the first matching lower and upper bound on the sample complexity of estimating the optimal policy in discounted reward MDPs. In this chapter, we consider two well-known model-based RL algorithms, Q-value iteration (QVI) and policy iteration (PI). We show that upper-bounds on the sample complexity of these methods match the general lower bound of RL in terms of all the parameters of interest. We also compare our bounds with the previous results for QVI and PI. We show that our results improve on the state of the art by a factor of order $1/(1-\gamma)$.

Samenvatting

De focus van dit proefschrift is het toepassen van Reinforcement Learning (RL) op Markov beslissingsprocessen (Markov Decision Processes, MDPs) met discounted reward. In RL hebben we vaak te maken met een agent die interactie heeft met zijn omgeving. Het doel van de agent is om zijn lange termijn prestaties te optimaliseren door middel van het zoeken naar een optimaal policy, dat wil zeggen een opeenvolging van acties die uitkeringen op de lange termijn maximaliseren. De lange termijn uitkeringen van een agent worden doorgaans beschreven met behulp van een waarde functie die, in het geval van discounted reward, gedefinieerd is als de verwachtingswaarde van de som van verdisconteerde uitkeringen.

Veel RL algoritmes zijn gebaseerd op Dynamisch Programmeren (DP) en maken via de zogenaamde Bellman Iteraties een schatting van het optimale policy. In het geval van MDPs met discounted reward kan men aantonen dat deze schattingen exponentieel snel convergeren naar de optimale oplossing. DP algoritmes hebben echter expliciete kennis nodig van het transitie- en uitkeringen model, die voor veel interessante problemen niet voorhanden zijn. Bij gebrek aan kennis van het model maakt men veelal een empirische schatting van de waarde functie door middel van Monte Carlo simulaties. De laatste jaren is er veel aandacht geweest voor het schatten van het optimale policy aan de hand van simulaties en dit heeft een aantal RL methoden opgeleverd. In sommige gevallen zijn er garanties bepaald voor het presteren van deze methodes in asymptotische context of in een eindig tijdsbestek.

In dit proefschrift concentreren we ons op het schatten van het optimale policy en de optimale waarde functie door middel van simulaties. Tot dit doel ontwikkelen we nieuwe RL algoritmes en analyseren we hun prestaties

in zowel asymptotische context als in een eindig tijdsbestek. Daarnaast verbeteren we enkele theoretische resultaten van bekende RL algoritmen zoals Model Gebaseerde Waarde Iteraties en policy iteratie. Verder zullen we enkele algemene ondergrenzen bepalen voor het aantal simulaties dat nodig is voor een bijna optimale oplossing (simulatie complexiteit grens) die geldig zijn voor RL algoritmes in het algemeen.

Dynamisch Policy Programmeren

In Hoofdstuk 2 stellen we een nieuw policy iteratie algorithm voor, genaamd Dynamisch policy Programmeren (Dynamic Policy Programming, DPP), om het optimale policy te berekenen in MDPs met discounted reward. We bewijzen dat de uitkomsten van het DPP algoritme asymptotisch convergeren naar het optimale policy, en we bepalen de snelheid van convergeren in het discrete geval. We geven ook een grens op het verlies aan nauwkeurigheid dat DPP veroorzaakt bij het gebruik van een benaderend model. Dit verlies geven we in termen van de supremum-norm van de gemiddelde geaccumuleerde fout bij het gebruik van een benaderend model. Dit is in tegenstelling met de reeds bekende resultaten over waardeverlies bij DP, die simpelweg worden uitgedrukt in termen van het supremum van fouten. Ten slotte stellen we een RL algoritme voor, genaamd DPP-RL, welke een variante is op DPP die gebruikt maakt van een gesimuleerde schatting van de bijwerk-regel, en we bewijzen dat het algoritme convergeert.

Speedy Q-Learning

In Hoofdstuk 3 introduceren we een nieuw RL algoritme, genaamd Speedy Q-Learning (SQL), om de problemen van de relatief langzame convergentie van standaard Q-Learning te overkomen. We onderzoeken zowel het asymptotische gedrag van SQL als dat over eindige tijd. We bewijzen grote-waarschijnlijkheid grenzen voor het prestatie verlies van SQL, en concluderen dat SQL sneller convergeert dan standaard Q-Learning. De numerieke experimenten in Sectie 3.4 bevestigen onze theoretische bevindingen voor $\gamma \approx 1$, waar γ de discount factor van de MDP is.

Simulatie Complexiteit van Model Gebaseerd RL

In Hoofdstuk 4 geven we een (voorheen onbekende) aansluitende onder- en bovengrens van de simulatie complexiteit van de schatting van het optimale

policy in MDPs met discounted reward. In dit hoofdstuk beschouwen we problemen in de context van model gebaseerd RL. We beschrijven de bovengrenzen van de simulatie complexiteit van twee bekende model gebaseerd RL algoritmen, zijnde Q-Waarde Iteratie (Q-Value Iteration, QVI) en policy iteratie (Policy Iiteration, PI). Deze bovengrenzen sluiten aan bij de algemene ondergrenzen van RL algoritmen in termen van de parameters die er toe doen. Ook vergelijken we onze grenzen met de reeds bekende resultaten over QVI en PI, en laten we zien dat onze resultaten de huidige grenzen verbeteren met een factor van orde $1/(1-\gamma)$.

Acknowledgement

I would like to express my great appreciation to anyone who helped me with writing this thesis. I am particularly grateful to my research supervisor and promoter, Bert Kappen, for his patient guidance and valuable critiques of my research work. Through the duration of my study, Bert gave me the freedom to study and explore new ideas and approaches. I would like to express my very great appreciation to Rémi Munos, for his enthusiastic encouragements and invaluable assistance. Rémi had a profound influence on my research by his creative ideas and deep knowledge of machine learning.

I would also like to extend my thanks to my colleagues in SNN. In particular, I would like to thank Vicenç Gómez for assisting me with computer simulations and numerical results and Sep Thijssen for helping me with some mathematical details and translating the summary of thesis. Special thanks should be given to other colleagues in SNN, Alberto Llera, Annet Wanders, Wim Wiegerinck, Bart van den Broek, Patrick Lessmann, Willem Burgers, Joris Bierkens and Kevin Sharp. I appreciate Mohammad Ghavamzadeh, Csaba Szepesvari, Amir Massoud Farhamnd, Peter Auer, Alessandro Lazaric, Nikos Vlassis, Geoff Gordon, Jan Peters and Odalric-Ambrym Maillard for helpful discussions. I would like to convey thanks to PASCAL2 Network of Excellence Internal-Visit Programme and the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement nº 270327. Finally, I wish to thank my mother Ziba, my father Hassan and my sisters for their support and encouragement throughout my study.

Curriculum Vitae

Mohammad Gheshlaghi Azar was born on May 1st, 1981, in Tehran, Iran. He entered primary school in 1987 and graduated from high school with a major on math and physics at the age of 17 in 1998. At the same year, he was admitted for the bachelor program on electrical engineering at K.N.Toosi University of technology, Tehran, Iran. He graduated with a major in communication systems in 2003. He continued his education as a master student by studying control theory and artificial intelligence at University of Tehran from 2003 to 2006. In 2008, Mohammad joined the machine learning group of prof. Bert Kappen at Radboud University Nijmegen (RUN). During his time at RUN his primary focus was on the theory of reinforcement learning and stochastic optimal control. In 2011, Mohammad had a chance to visit INRIA, Lille, France, where he worked with Rémi Munos and Mohammad Ghavamzadeh on the theory of reinforcement learning. In August 2012, He finished his Ph.D. at RUN and joined the computer science department of Carnegie Mellon University as a post-doctoral fellow.

Donders Graduate School for Cognitive Neuroscience Series

1. van Aalderen-Smeets, S.I. (2007). *Neural dynamics of visual selection.* Maastricht University, Maastricht, the Netherlands.
2. Schoffelen, J.M. (2007). *Neuronal communication through coherence in the human motor system.* Radboud University Nijmegen, Nijmegen, the Netherlands.
3. de Lange, F.P. (2008). *Neural mechanisms of motor imagery.* Radboud University Nijmegen, Nijmegen, the Netherlands.
4. Grol, M.J. (2008). *Parieto-frontal circuitry in visuomotor control. Utrecht* University, Utrecht, the Netherlands.
5. Bauer, M. (2008). *Functional roles of rhythmic neuronal activity in the human visual and somatosensory system.* Radboud University Nijmegen, Nijmegen, the Netherlands.
6. Mazaheri, A. (2008). *The Influence of Ongoing Oscillatory Brain Activity on Evoked Responses and Behaviour*. Radboud University Nijmegen, Nijmegen, the Netherlands.
7. Hooijmans, C.R. (2008). *Impact of nutritional lipids and vascular factors in Alzheimer's Disease.* Radboud University Nijmegen, Nijmegen, the Netherlands.
8. Gaszner, B. (2008). *Plastic responses to stress by the rodent urocortinergic Edinger-Westphal nucleus.* Radboud University Nijmegen, Nijmegen, the Netherlands.
9. Willems, R.M. (2009). *Neural reflections of meaning in gesture, language and action.* Radboud University Nijmegen, Nijmegen, the Netherlands.
10. van Pelt, S. (2009). *Dynamic neural representations of human visuomotor space.* Radboud University Nijmegen, Nijmegen, the Netherlands.
11. Lommertzen, J. (2009). *Visuomotor coupling at different levels of complexity.* Radboud University Nijmegen, Nijmegen, the Netherlands.
12. Poljac, E. (2009). *Dynamics of cognitive control in task switching: Looking beyond the switch cost.* Radboud University Nijmegen, Nijmegen, the Netherlands.
13. Poser, B.A. (2009). *Techniques for BOLD and blood volume weighted fMRI.* Radboud University Nijmegen, Nijmegen, the Netherlands.
14. Baggio, G. (2009). *Semantics and the electrophysiology of meaning. Tense, aspect, event structure*. Radboud University Nijmegen, Nijmegen, the Netherlands.
15. van Wingen, G.A. (2009). *Biological determinants of amygdala functioning*. Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.
16. Bakker, M. (2009). *Supraspinal control of walking: lessons from motor imagery.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.
17. Aarts, E. (2009). *Resisting temptation: the role of the anterior cingulate cortex in adjusting cognitive control.* Radboud University Nijmegen, Nijmegen, the Netherlands.
18. Prinz, S. (2009). *Waterbath stunning of chickens – Effects of electrical parameters on the electroencephalogram and physical reflexes of broilers.* Radboud University Nijmegen, Nijmegen, the Netherlands.
19. Knippenberg, J.M.J. (2009). *The N150 of the Auditory Evoked Potential from the rat amygdala: In search for its functional significance.* Radboud University Nijmegen, Nijmegen, the Netherlands.
20. Dumont, G.J.H. (2009). *Cognitive and physiological effects of 3,4-methylenedioxymethamphetamine (MDMA or 'ecstasy') in combination with alcohol or cannabis in humans* Radboud University Nijmegen, Nijmegen, the Netherlands.
21. Pijnacker, J. (2010). *Defeasible inference in autism: a behavioral and electrophysiogical approach*. Radboud University Nijmegen, Nijmegen, the Netherlands.

22. de Vrijer, M. (2010). *Multisensory integration in spatial orientation.* Radboud University Nijmegen, Nijmegen, the Netherlands.
23. Vergeer, M. (2010). *Perceptual visibility and appearance: Effects of color and form.* Radboud University Nijmegen, Nijmegen, the Netherlands.
24. Levy, J. (2010). *In Cerebro Unveiling Unconscious Mechanisms during Reading.* Radboud University Nijmegen, Nijmegen, the Netherlands.
25. Treder, M. S. (2010). *Symmetry in (inter)action*. Radboud University Nijmegen, Nijmegen, the Netherlands.
26. Horlings C.G.C. (2010). *A Weak balance; balance and falls in patients with neuromuscular disorders.* Radboud University Nijmegen, Nijmegen, the Netherlands.
27. Snaphaan, L.J.A.E. (2010). *Epidemiology of post-stroke behavioural consequences*. Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.
28. Dado - Van Beek, H.E.A. (2010). *The regulation of cerebral perfusion in patients with Alzheimer's disease.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.
29. Derks, N.M. (2010). *The role of the non-preganglionic Edinger-Westphal nucleus in sex-dependent stress adaptation in rodents.* Radboud University Nijmegen, Nijmegen, the Netherlands.
30. Wyczesany, M. (2010). *Covariation of mood and brain activity. Integration of subjective self-report data with quantitative EEG measures*. Radboud University Nijmegen, Nijmegen, the Netherlands.
31. Beurze S.M. (2010). *Cortical mechanisms for reach planning.* Radboud University Nijmegen, Nijmegen, the Netherlands.
32. van Dijk, J.P. (2010). *On the Number of Motor Units.* Radboud University Nijmegen, Nijmegen, the Netherlands.
33. Lapatki, B.G. (2010). *The Facial Musculature - Characterization at a Motor Unit Level.* Radboud University Nijmegen, Nijmegen, the Netherlands.
34. Kok, P. (2010). *Word Order and Verb Inflection in Agrammatic Sentence Production.* Radboud University Nijmegen, Nijmegen, the Netherlands.
35. van Elk, M. (2010). *Action semantics: Functional and neural dynamics.* Radboud University Nijmegen, Nijmegen, the Netherlands.
36. Majdandzic, J. (2010). *Cerebral mechanisms of processing action goals in self and others.* Radboud University Nijmegen, Nijmegen, the Netherlands.
37. Snijders, T.M. (2010). *More than words - neural and genetic dynamics of syntactic unification.* Radboud University Nijmegen, Nijmegen, the Netherlands.
38. Grootens, K.P. (2010). *Cognitive dysfunction and effects of antipsychotics in schizophrenia and borderline personality disorder.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.
39. Nieuwenhuis, I.L.C. (2010). *Memory consolidation: A process of integration - Converging evidence from MEG, fMRI and behavior.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.
40. Menenti, L.M.E. (2010). *The right language: differential hemispheric contributions to language production and comprehension in context.* Radboud University Nijmegen, Nijmegen, the Netherlands.
41. van Dijk, H.P. (2010). *The state of the brain, how alpha oscillations shape behaviour and event related responses.* Radboud University Nijmegen, Nijmegen, the Netherlands.
42. Meulenbroek, O.V. (2010). *Neural correlates of episodic memory in healthy aging and Alzheimer's disease.* Radboud University Nijmegen, Nijmegen, the Netherlands.
43. Oude Nijhuis, L.B. (2010). *Modulation of human balance reactions.* Radboud

University Nijmegen, Nijmegen, the Netherlands.

44. Qin, S. (2010). *Adaptive memory: imaging medial temporal and prefrontal memory systems*. Radboud University Nijmegen, Nijmegen, the Netherlands.

45. Timmer, N.M. (2011). *The interaction of heparan sulfate proteoglycans with the amyloid protein.* Radboud University Nijmegen, Nijmegen, the Netherlands.

46. Crajé, C. (2011). *(A)typical motor planning and motor imagery.* Radboud University Nijmegen, Nijmegen, the Netherlands.

47. van Grootel, T.J. (2011). *On the role of eye and head position in spatial localisation behaviour.* Radboud University Nijmegen, Nijmegen, the Netherlands.

48. Lamers, M.J.M. (2011). *Levels of selective attention in action planning.* Radboud University Nijmegen, Nijmegen, the Netherlands.

49. Van der Werf, J. (2011). *Cortical oscillatory activity in human visuomotor integration*. Radboud University Nijmegen, Nijmegen, the Netherlands.

50. Scheeringa, R. (2011). *On the relation between oscillatory EEG activity and the BOLD signal.* Radboud University Nijmegen, Nijmegen, the Netherlands.

51. Bögels, S. (2011). *The role of prosody in language comprehension: when prosodic breaks and pitch accents come into play.* Radboud University Nijmegen, Nijmegen, the Netherlands.

52. Ossewaarde, L. (2011). *The mood cycle: hormonal influences on the female brain.* Radboud University Nijmegen, Nijmegen, the Netherlands.

53. Kuribara, M. (2011). *Environment-induced activation and growth of pituitary melanotrope cells of Xenopus laevis.* Radboud University Nijmegen, Nijmegen, the Netherlands.

54. Helmich, R.C.G. (2011). *Cerebral reorganization in Parkinson's disease.* Radboud University Nijmegen, Nijmegen, the Netherlands.

55. Boelen, D. (2011). *Order out of chaos? Assessment and treatment of executive disorders in brain-injured patients*. Radboud University Nijmegen, Nijmegen, the Netherlands.

56. Koopmans, P.J. (2011). *fMRI of cortical layers.* Radboud University Nijmegen, Nijmegen, the Netherlands.

57. van der Linden, M.H. (2011). *Experience-based cortical plasticity in object category representation.* Radboud University Nijmegen, Nijmegen, the Netherlands.

58. Kleine, B.U. (2011). *Motor unit discharges - Physiological and diagnostic studies in ALS.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

59. Paulus, M. (2011). *Development of action perception: Neurocognitive mechanisms underlying children's processing of others' actions.* Radboud University Nijmegen, Nijmegen, the Netherlands.

60. Tieleman, A.A. (2011). *Myotonic dystrophy type 2. A newly diagnosed disease in the Netherlands.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

61. van Leeuwen, T.M. (2011). *'How one can see what is not there': Neural mechanisms of grapheme-colour synaesthesia*. Radboud University Nijmegen, Nijmegen, the Netherlands.

62. van Tilborg, I.A.D.A. (2011). *Procedural learning in cognitively impaired patients and its application in clinical practice.* Radboud University Nijmegen, Nijmegen, the Netherlands.

63. Bruinsma, I.B. (2011). *Amyloidogenic proteins in Alzheimer's disease and Parkinson's disease: interaction with chaperones and inflammation.* Radboud University Nijmegen, Nijmegen, the Netherlands.

64. Voermans, N. (2011). *Neuromuscular features of Ehlers-Danlos syndrome and Marfan syndrome; expanding the phenotype of inherited connective tissue disorders and investigating the role of the extracellular matrix in muscle.* Radboud University

Nijmegen Medical Centre, Nijmegen, the Netherlands.

65. Reelick, M. (2011). *One step at a time. Disentangling the complexity of preventing falls in frail older persons.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

66. Buur, P.F. (2011). *Imaging in motion. Applications of multi-echo fMRI.* Radboud University Nijmegen, Nijmegen, the Netherlands.

67. Schaefer, R.S. (2011). *Measuring the mind's ear: EEG of music imagery.* Radboud University Nijmegen, Nijmegen, the Netherlands.

68. Xu, L. (2011). *The non-preganglionic Edinger-Westphal nucleus: an integration center for energy balance and stress adaptation.* Radboud University Nijmegen, Nijmegen, the Netherlands.

69. Schellekens, A.F.A. (2011). *Gene-environment interaction and intermediate phenotypes in alcohol dependence.* Radboud University Nijmegen, Nijmegen, the Netherlands.

70. van Marle, H.J.F. (2011). *The amygdala on alert: A neuroimaging investigation into amygdala function during acute stress and its aftermath.* Radboud University Nijmegen, Nijmegen, the Netherlands.

71. De Laat, K.F. (2011). *Motor performance in individuals with cerebral small vessel disease: an MRI study.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

72. Mädebach, A. (2011). *Lexical access in speaking: Studies on lexical selection and cascading activation.* Radboud University Nijmegen, Nijmegen, the Netherlands.

73. Poelmans, G.J.V. (2011). *Genes and protein networks for neurodevelopmental disorders.* Radboud University Nijmegen, Nijmegen, the Netherlands.

74. van Norden, A.G.W. (2011). *Cognitive function in elderly individuals with cerebral small vessel disease. An MRI study.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

75. Jansen, E.J.R. (2011). *New insights into V-ATPase functioning: the role of its accessory subunit Ac45 and a novel brain-specific Ac45 paralog.* Radboud University Nijmegen, Nijmegen, the Netherlands.

76. Haaxma, C.A. (2011). *New perspectives on preclinical and early stage Parkinson's disease.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

77. Haegens, S. (2012). *On the functional role of oscillatory neuronal activity in the somatosensory system.* Radboud University Nijmegen, Nijmegen, the Netherlands.

78. van Barneveld, D.C.P.B.M. (2012). *Integration of exteroceptive and interoceptive cues in spatial localization.* Radboud University Nijmegen, Nijmegen, the Netherlands.

79. Spies, P.E. (2012). *The reflection of Alzheimer disease in CSF.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

80. Helle, M. (2012). *Artery-specific perfusion measurements in the cerebral vasculature by magnetic resonance imaging.* Radboud University Nijmegen, Nijmegen, the Netherlands.

81. Egetemeir, J. (2012). *Neural correlates of real-life joint action.* Radboud University Nijmegen, Nijmegen, the Netherlands.

82. Janssen, L. (2012). *Planning and execution of (bi)manual grasping.* Radboud University Nijmegen, Nijmegen, the Netherlands.

83. Vermeer, S. (2012). *Clinical and genetic characterisation of Autosomal Recessive Cerebellar Ataxias.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

84. Vrins, S. (2012). *Shaping object boundaries: contextual effects in infants and adults.* Radboud University Nijmegen, Nijmegen, the Netherlands.

85. Weber, K.M. (2012). *The language learning brain: Evidence from second lan-*

guage and bilingual studies of syntactic processing. Radboud University Nijmegen, Nijmegen, the Netherlands.

86. Verhagen, L. (2012). *How to grasp a ripe tomato.* Utrecht University, Utrecht, the Netherlands.

87. Nonkes, L.J.P. (2012). *Serotonin transporter gene variance causes individual differences in rat behaviour: for better and for worse.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

88. Joosten-Weyn Banningh, L.W.A. (2012). *Learning to live with Mild Cognitive Impairment: development and evaluation of a psychological intervention for patients with Mild Cognitive Impairment and their significant others*. Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

89. Xiang, HD. (2012). *The language networks of the brain.* Radboud University Nijmegen, Nijmegen, the Netherlands

90. Snijders, A.H. (2012). *Tackling freezing of gait in Parkinson's disease.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

91. Rouwette, T.P.H. (2012). *Neuropathic Pain and the Brain - Differential involvement of corticotropin-releasing factor and urocortin 1 in acute and chronic pain processing*. Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

92. van de Meerendonk, N. (2012). *States of indecision in the brain: Electrophysiological and hemodynamic reflections of monitoring in visual language perception.* Radboud University Nijmegen, Nijmegen, the Netherlands.

93. Sterrenburg, A. (2012). *The stress response of forebrain and midbrain regions: neuropeptides, sex-specificity and epigenetics*. Radboud University Nijmegen, Nijmegen, The Netherlands.

94. Uithol, S. (2012). *Representing Action and Intention*. Radboud University Nijmegen, Nijmegen, The Netherlands.

95. van Dam, W.O. (2012). *On the specificity and flexibility of embodied lexical-semantic representations*. Radboud University Nijmegen, Nijmegen, The Netherlands.

96. Slats, D. (2012). *CSF biomarkers of Alzheimer's disease; serial sampling analysis and the study of circadian rhythmicity*. Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

97. Van Nuenen, B.F.L. *Cerebral reorganization in premotor parkinsonism.* Radboud University Nijmegen Medical Centre, Nijmegen, the Netherlands.

98. Van Schouwenburg, M.R. *Fronto-striatal mechanisms of attentional control*. Radboud University Nijmegen, Nijmegen, The Netherlands.

99. Azar, M.G. *On the theory of reinforcement learning: methods, convergence analysis and sample complexity*. Radboud University Nijmegen, Nijmegen, The Netherlands.