Proceedings of the European Control Conference 2007
Kos, Greece, July 2-5, 2007

Convergence of Q-learning with linear function approximation

Francisco S. Melo
Institute for Systems and Robotics,
Instituto Superior Técnico,
Lisboa, PORTUGAL
fmelo@isr.ist.utl.pt

M. Isabel Ribeiro
Institute for Systems and Robotics,
Instituto Superior Técnico,
Lisboa, PORTUGAL
mir@isr.ist.utl.pt

Abstract— **In this paper, we analyze the convergence properties of Q-learning using linear function approximation. This algorithm can be seen as an extension to stochastic control settings of TD-learning using linear function approximation, as described in [1]. We derive a set of conditions that implies the convergence of this approximation method with probability 1, when a fixed learning policy is used. We provide an interpretation of the obtained approximation as a fixed point of a Bellman-like operator. We then discuss the relation of our result with several related works as well as its general applicability.**

I. INTRODUCTION

Reinforcement learning addresses the problem of an agent faced with a sequential decision problem and using evaluative feedback as a performance measure [2]. Reinforcement learning methods compute a mapping from the set of states of the agent to the set of possible actions. Such mapping is called a *policy* and it is customary to define a utility-function, or *value-function*, estimating the practical utility of each particular policy. Value-based methods such as TD-learning [3], Q-learning [4] or SARSA [5] have been exhaustively covered in the literature and, under mild assumptions, have been proven to converge to the desired solution [6]–[8].

However, many such algorithms require explicit representation of the state-space, and is often the case that the state-space is unsuited for explicit representation. Instead, the decision-maker should be able to *generalize* its action-pattern from the collected experience. Gerald Tesauro's backgammon player [9], [10], which was able to learn master-level play, boosted the interest in the topic of generalization. In his work, Tesauro combines temporal-difference learning with neural networks, and the impressive results of his learning agent have established the applicability of function approximation in reinforcement learning problems. Other works on generalization include [11]–[14].

In this paper, we describe *Q-learning with linear function approximation*. This algorithm can be seen as an extension to control problems of temporal-difference learning using linear function approximation as described in [1]. Convergence of Q-learning with function approximation has been a long standing question in reinforcement learning. Our result provides a contribution towards the solution of this problem. We identify a set of conditions that ensure convergence of this method with probability 1. We also provide an interpretation for the obtained approximation as the fixed-point of a Bellman-like operator. Finally, we discuss our convergence results in face of those reported using so different approaches as soft-state aggregation [13] or interpolation-based Q-learning [14].

In the next section, we describe the framework addressed throughout this paper. We then present the algorithm and the main result of the paper, concerning its convergence. We conclude the paper with some discussion. Details of the proof are found in the appendix.

II. THE MARKOV DECISION PROCESS FRAMEWORK

Let $\mathcal{X}$ be a compact subspace of $\mathbb{R}^p$ and $\{X_t\}$ a $\mathcal{X}$-valued controlled Markov chain. The transition probabilities for the chain are given by a probability kernel

$$\mathbb{P}\left[X_{t+1} \in U \mid X_t = x, A_t = a\right] = \mathsf{P}_a(x, U),$$

where $U \subset \mathcal{X}$. The $\mathcal{A}$-valued process $\{A_t\}$ represents the control process; A_t is the control action at time instant t and $\mathcal{A}$ is the finite set of possible actions. The agent aims at choosing the control process $\{A_t\}$ so as to maximize the infinite-horizon total discounted reward, *i.e.*,

$$V(\{A_t\}, x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(X_t, A_t) \mid X_0 = x\right],$$

where $0 \leq \gamma < 1$ is a discount-factor and $R(x, a)$ represents a random "reward" received for taking action $a \in \mathcal{A}$ in state $x \in \mathcal{X}$.

We assume throughout this paper that there is a deterministic function $r : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \longrightarrow \mathbb{R}$ assigning a reward $r(x, a, y)$ every time a transition from x to y occurs after taking action a and that

$$\mathbb{E}\left[R(x, a)\right] = \int_{\mathcal{X}} r(x, a, y)\mathsf{P}_a(x, dy).$$

This simplifies the notation without introducing a great loss in generality. We further assume that there is a constant $\mathcal{R} \in \mathbb{R}$ such that $|r(x, a, y)| < \mathcal{R}$ for all $x, y \in \mathcal{X}$ and all $a \in \mathcal{A}$.[1] We refer to the 5-tuple $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ as a *Markov decision process* (MDP).

Given the MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, the *optimal value function* V^* is defined for each state $x \in \mathcal{X}$ as

$$V^*(x) = \max_{\{A_t\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(X_t, A_t) \mid X_0 = x\right]$$

[1]This assumption is tantamount to the standard requirement that the rewards $R(x, a)$ have uniformly bounded variance.

and verifies

$$V^*(x) = \max_{a\in\mathcal{A}} \int_{\mathcal{X}} \big[r(x,a,y) + \gamma V^*(y)\big] \mathsf{P}_a(x,dy),$$

which is a form of the Bellman optimality equation. From the optimal value function, the optimal Q-values $Q^*(x,a)$ are defined for each state-action pair $(x,a) \in \mathcal{X}\times\mathcal{A}$ as

$$Q^*(x,a) = \int_{\mathcal{X}} \big[r(x,a,y) + \gamma V^*(y)\big] \mathsf{P}_a(x,dy). \tag{1}$$

From the optimal Q-function Q^* it is possible to define a mapping $\delta^* : \mathcal{X} \longrightarrow \mathcal{A}$ as

$$\delta^*(x) = \arg\max_{a\in\mathcal{A}} Q^*(x,a),$$

for all $x \in \mathcal{X}$ and the control process $\{A_t\}$ defined by $A_t = \delta^*(X_t)$ is optimal in the sense that $V(\{A_t\}, x) = V^*(x)$, for all $x \in \mathcal{X}$. The mapping δ^* is an *optimal policy* for the MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$.

More generally, we define a (stochastic) *policy* as a mapping $\delta : \mathcal{X}\times\mathcal{A} \longrightarrow [0,1]$ that generates a control process $\{A_t\}$ verifying $\mathbb{P}[A_t = a \mid X_t = x] = \delta(x,a)$, for all t. Simply stated, $\delta(x,a)$ defines the probability of choosing each action a in state x. Clearly, since $\delta(x,\cdot)$ is a probability distribution over $\mathcal{A}$, it must satisfy $\sum_{a\in\mathcal{A}} \delta(x,a) = 1$, for all $x \in \mathcal{X}$. In the particular case where, for every $x \in \mathcal{A}$, $\delta(x,a) = 1$ for some $a \in \mathcal{A}$, we say that δ is a *deterministic policy* and simply denote by $\delta(x)$ the action chosen at state x with probability 1. In this case, a control process is generated by policy δ if $A_t = \delta(X_t)$ for all t. We write $V^\delta(x)$ instead of $V(\{A_t\}, x)$ if the control process $\{A_t\}$ is generated by a policy δ.

The optimal control process can be obtained from the optimal policy δ^*, which can in turn be obtained from Q^*. Therefore, the optimal control problem is solved once the function Q^* is known for all pairs $(x,a) \in \mathcal{X}\times\mathcal{A}$.

The original Q-learning algorithm [4] uses a stochastic iterative update to determine the optimal Q-values. The update-rule for Q-learning is

$$\begin{aligned} Q_{k+1}(x,a) = Q_k(x,a) + \alpha_k\big[R(x,a) + \\ + \gamma \max_{b\in\mathcal{A}} Q_k(X(x,a),b) - Q_k(x,a)\big], \end{aligned} \tag{2}$$

where $Q_k(x,a)$ is the k^{th} estimate of $Q^*(x,a)$, $X(x,a)$ is a $\mathcal{X}$-valued random variable obtained according to the probabilities defined by P_a and α_k are step-sizes verifying $\sum_k \alpha_k = \infty$ and $\sum_k \alpha_k^2 < \infty$. Notice that $R(x,a)$ and $X(x,a)$ can be obtained through some simulation device, not requiring the knowledge of either P or r.

Q-leaning was originally proposed to deal with MDPs with finite state-spaces. When $\mathcal{X}$ is an infinite set, it is not possible to straightforwardly apply the update rule (2), since it explicitly updates the Q-value for each individual state-action pair and there are infinitely many such pairs.

In this paper we consider a parameterized family $\mathcal{Q}$ of functions $Q_\theta : \mathcal{X}\times\mathcal{A} \longrightarrow \mathbb{R}$, where θ is a parameter in $\mathbb{R}^M$. We want to determine the point θ^* in parameter space such that Q_{θ^*} is the best approximation of Q^* in $\mathcal{Q}$, in a sense yet to be made clear. By defining a suitable recursion for θ, we reduce the determination of the infinite-dimensional function Q^* to the determination of a finite-dimensional vector θ^*.

III. MAIN RESULT

In this section, we establish the convergence properties of Q-learning when using linear function approximation. As will be seen, the result derived herein can be seen as a generalization of other results from the literature (such as those reported on methods based on discretization [15], soft-discretization [13] or interpolation [14]).

We consider a family of functions $\mathcal{Q} = \{Q_\theta\}$ parameterized by a finite-dimensional parameter vector $\theta \in \mathbb{R}^M$. We admit the family $\mathcal{Q}$ to be linear in that if $q_1, q_2 \in \mathcal{Q}$, then so does $\alpha q_1 + q_2$ for any $\alpha \in \mathbb{R}$. $\mathcal{Q}$ is therefore the linear span of a set of M linearly independent functions $\xi_i : \mathcal{X}\times\mathcal{A} \longrightarrow \mathbb{R}$, and each $q \in \mathcal{Q}$ can be written as

$$q(x,a) = \sum_{i=1}^{M} \xi_i(x,a)\theta(i),$$

for all pairs $(x,a) \in \mathcal{X}\times\mathcal{A}$, where $\theta(i)$ is the ith component of the vector $\theta \in \mathbb{R}^M$. Given now a set $\{\xi_i, i = 1,\ldots,M\}$ of linearly independent functions and a vector $\theta \in \mathbb{R}^M$, we denote interchangeably by Q_θ and $Q(\theta)$ the function

$$Q_\theta(x,a) = \sum_{i=1}^{M} \xi_i(x,a)\theta(i) = \xi^\top(x,a)\theta, \tag{3}$$

where $\xi(x,a)$ is a vector in $\mathbb{R}^M$ with ith component given by $\xi_i(x,a)$ and the superscript $^\top$ represents the transpose operator.

Let δ be a stochastic stationary policy and suppose that $\{x_t\}$, $\{a_t\}$ and $\{r_t\}$ are sampled trajectories obtained from the MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ using policy δ. In the original Q-learning algorithm, the Q-values are updated according to (2), using the *temporal difference* at time t,

$$\Delta_t = r_t + \gamma \max_{b\in\mathcal{A}} Q_t(x_{t+1}, b) - Q_t(x_t, a_t).$$

The temporal difference Δ_t works as a 1-step estimation error with respect to the optimal function Q^*; the update rule in Q-learning "moves" the estimates Q_t closer to the desired function Q^*, minimizing the expected value of Δ_t. Applying the same underlying idea, we obtain the following update rule for the vector θ_t:

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha_t \nabla_\theta Q_\theta(x_t, a_t)\big(r_t + \\ &\quad + \gamma \max_{b\in\mathcal{A}} Q_{\theta_t}(x_{t+1}, b) - Q_{\theta_t}(x_t, a_t)\big) = \\ &= \theta_t + \alpha_t \xi(x_t, a_t)\big(r_t + \\ &\quad + \gamma \max_{b\in\mathcal{A}} Q_{\theta_t}(x_{t+1}, b) - Q_{\theta_t}(x_t, a_t)\big). \end{aligned} \tag{4}$$

Notice that (4) updates θ_t using the temporal difference Δ_t as the error. The gradient $\nabla_\theta Q_\theta$ provides the "direction" in which this update is performed. Roughly speaking, (4) can be seen as a gradient descent update using $\alpha_t\Delta_t$ as the step-size, and (4) iteratively determines the parameter θ corresponding to the function in $\mathcal{Q}$ minimizing the expected value of Δ_t.[2]

We show that, under some regularity assumptions on the Markov chain $(\mathcal{X}, \mathsf{P}_\delta)$ obtained using the policy δ, the trajectories of the algorithm closely follow those of an associated ODE with a globally asymptotically stable equilibrium point θ^*. Therefore, the sequence $\{\theta_t\}$ defined recursively by (4) will converge w.p.1 to the equilibrium point θ^* of the ODE. We also show that this equilibrium point is the fixed-point of a Bellman-like operator,

$$\theta^* = \mathcal{P}\mathbf{H}Q(\theta^*), \tag{5}$$

where $\mathcal{P}$ represents a projection and $\mathbf{H}$ is the Bellman operator

$$(\mathbf{H}q)(x,a) = \int_{\mathcal{X}} \left[r(x,a,y) + \gamma \max_{b\in\mathcal{A}} q(y,b)\right] \mathsf{P}_a(x,dy). \tag{6}$$

We now state our main convergence result. Given a MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, let δ be a stationary stochastic policy and $(\mathcal{X}, \mathsf{P}_\delta)$ the corresponding Markov chain with invariant probability measure μ_X. Denote by $\mathbb{E}_{\mu_\delta}[\cdot]$ the expectation w.r.t. the probability measure μ_δ defined for every set $Z \times U \subset \mathcal{X} \times \mathcal{A}$ as

$$\mu_\delta(Z \times U) = \int_Z \sum_{a\in U} \delta(x,a)\mu_X(x).$$

Theorem 3.1: Let $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ be a Markov decision process and assume the Markov chain $(\mathcal{X}, \mathsf{P}_\delta)$ to be geometrically ergodic with invariant probability measure μ_X. Suppose that $\delta(x,a) > 0$ for all $a \in \mathcal{A}$ and μ_X-almost all $x \in \mathcal{X}$.

Let $\Xi = \{\xi_i, i = 1, \ldots, M\}$ be a set of M bounded, linearly independent functions defined on $\mathcal{X} \times \mathcal{A}$ and taking values in $\mathbb{R}$. In particular, admit that $\sum_{i=1}^N |\xi_i(x,a)| \leq 1$ for all $(x,a) \in \mathcal{X} \times \mathcal{A}$.

Then, the following hold.

1) **Convergence**
For any initial condition $\theta_0 \in \mathbb{R}^M$, the algorithm

$$\begin{aligned}\theta_{t+1} = \theta_t + \alpha_t \xi(x_t,a_t)\big(r_t + \\ + \gamma \max_{b\in\mathcal{A}} Q_{\theta_t}(x_{t+1},b) - Q_{\theta_t}(x_t,a_t)\big)\end{aligned} \tag{7}$$

converges w.p.1 as long as the step-size sequence $\{\alpha_t\}$ verifies

$$\sum_t \alpha_t = \infty \qquad \sum_t \alpha_t^2 \leq \infty.$$

2) **Limit of convergence**
Under these conditions, the limit θ^* of (7) verifies

$$Q_{\theta^*}(x,a) = (\mathcal{P}_\mathcal{Q}\mathbf{H}Q_{\theta^*})(x,a), \tag{8}$$

where $\mathcal{P}_\mathcal{Q}$ denotes the orthogonal projection operator on $\mathcal{Q}$ defined by

$$(\mathcal{P}_\mathcal{Q}Q)(x,a) = \xi^\top(x,a)\Sigma^{-1}\mathbb{E}_{\mu_\delta}\left[\xi(z,u)Q(z,u)\right].$$

and the matrix Σ is given by

$$\Sigma = \mathbb{E}_{\mu_\delta}\left[\xi(x,a)\xi^\top(x,a)\right].$$

Proof: See Appendix I. ■

To convey a deeper insight on the conditions of the Theorem, we remark that the geometric ergodicity assumption and the requirement that $\delta(x,a) > 0$ for all $a \in \mathcal{A}$ and μ_X-almost all $x \in \mathcal{X}$ can be interpreted as a continuous counterpart to the usual condition that all state-action pairs are visited infinitely often. In fact, geometric ergodicity implies that all the regions of the state space with positive μ_X measure are "sufficiently" visited [17], and the condition that $\delta(x,a) > 0$ ensures that, at each state, every action is "sufficiently" tried.

On the other hand, geometric ergodicity guarantees that the Markov chain $(\mathcal{X}, \mathsf{P}_\delta)$ obtained using the learning policy δ converges exponentially fast to stationarity, and thus the analysis of convergence of the algorithm can be done in terms of a "stationary version" of it.[3]

We also notice that the requirements on the basis functions ξ_i simply guarantee (in a rather conservative way) that no two functions ξ_i lead to "colliding updates", as in the known counter-example in [16].

IV. RELATED WORK

In [1], Tsitsiklis and Van Roy provide a detailed analysis of temporal difference methods. In the referred work, the authors establish important results regarding the convergence/divergence of such methods when linear function approximation is used: the trajectories of TD-learning closely follow those of an associated globally asymptotically stable ODE and thus converge to the unique equilibrium point of such ODE. They also provide error bounds for the obtained approximation and an interpretation of the limit function as the fixed point of a composite operator $\mathcal{P}_\mathcal{V}\mathbf{T}^{(\lambda)}$, where $\mathcal{P}_\mathcal{V}$ is an orthogonal projection into a linear function space $\mathcal{V}$ and $\mathbf{T}^{(\lambda)}$ is a TD operator. The authors remark, however, that if off-policy updates are used (such as those in Q-learning) their convergence results no longer hold. More detailed analysis of this method as well as some variations can be found in [6], [12], [18].

Least-squares temporal difference learning methods (LSTD) depart from the same idea and propose an alternative method to estimate the same approximation used in TD-learning with function approximation [19]. As established by Tsitsiklis and Van Roy [1], TD-learning with linear function approximation converges in the limit to a parameter vector θ^* that verifies a linear system of the type

$$\mathbf{A}\theta^* + \mathbf{b} = 0,$$

where the matrices $\mathbf{A}$ and $\mathbf{b}$ arise from a "stationary version" of the algorithm. LSTD estimates directly the matrices $\mathbf{A}$ and $\mathbf{b}$ from the sample trajectories of the underlying Markov chain, thus converging to the same limiting approximation

[2]We remark that this interpretation lacks rigor and aims only at clarifying the underlying working of the algorithm. We refer to [16], where reinforcement learning is addressed using gradient methods.

[3]In particular, exponential convergence to stationarity ensures that the analysis of the trajectories of the sequence $\{\theta_t\}$ can be performed in terms of the trajectories of an associated ODE.

and with some extra argued advantages [19]. Although not exactly in the line of the algorithm described herein, it is important to mention LSTD as closely related with TD-learning with function approximation.

Szepesvári and Smart [14] extend the results in [1] to control settings. In this paper, the authors propose a modified version of Q-learning that approximates the optimal Q-values of a given set of sample points and then uses interpolation to approximate Q^* at any query point. The update rule for interpolation-based Q-learning uses a *spreading function* similar to the one used in multi-state Q-learning [20]. As in [1], the authors establish convergence w.p.1 of the algorithm and provide an interpretation of the limit point as the fixed-point of a composite operator $\mathcal{PH}$, where $\mathcal{P}$ is a projection-like operator and $\mathcal{H}$ can be interpreted as a modified TD-learning operator. Interpolated Q-learning approximates the value of the optimal Q-function at a previously chosen set of sample points.

Closely related with the method in [14] is the soft-state aggregation algorithm by [13]. In this work, the authors propose the use of a "soft"-partition of the state-space: the state-space is split into "soft" regions (each state x belongs to region i with a probability $p_i(x)$) and an "average" Q-value $Q(i, a)$ is defined for each region-action pair. Each of these regions is then treated as a "hyper-state" and the method uses standard Q-learning updates to determine the average Q-values for each region. The function Q^* is then approximated for a state-action pair (x, a) as $Q^*(x, a) \approx \sum_i p_i(x)Q(i, a)$.

Both interpolated Q-learning and Q-learning with soft-state aggregation provide important positive convergence results for Q-learning using function approximation, arising as encouraging counterparts to the divergent counterexamples reported in several works [1], [16].

Finally, Atkeson et al. [21] describe the application of local weighted regression methods to estimate the model (namely the kernel P and the reward function r) in a continuous-state reinforcement learning problem and this fundamental idea is generalized in [22]. In the latter work, the authors establish convergence in probability and derive the limiting distribution of the obtained approximation. The authors also address the bias-variance tradeoff in reinforcement learning problems.

In the next section we conclude the paper by further discussing our results in face of some of the aforementioned works.

V. AN ILLUSTRATIVE EXAMPLE

Consider the indoor environment depicted in Figure 1.

A mobile robot is intended to navigate to the goal region, signaled with the dashed lines. The environment is a 1×1 square, and the state of the robot at each time instant is a pair $(\mathbf{x}, \mathbf{y})$ of coordinates.[4] The coordinates of the goal corner are $(1, 1)$.

[4] We use boldface symbols $\mathbf{x}$ and $\mathbf{y}$ to denote the physical coordinates of the robot to distinguish from the symbols x and y used to denote generic elements of the state-space $\mathcal{X}$.

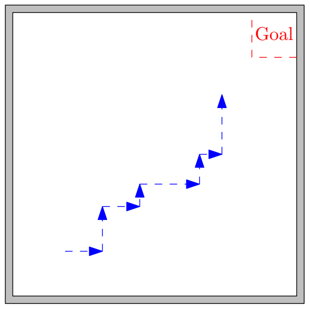


Fig. 1. Simple indoor environment.

The navigation problem can be described by a MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, where

- $\mathcal{X} = [0; 1] \times [0; 1]$;
- $\mathcal{A} = \{N, S, E, W\}$;
- Each action in $\mathcal{A}$ moves the robot in the corresponding direction a random distance between 0 and 0.3;
- The reward function r assigns a reward of $+10$ for every transition triplet (x, a, y) such that $\|y - (1; 1)\| < 0.1$ and 0 otherwise.

When the robot reaches the goal region, its position is randomly reset to any point in the room, independently of the agent's action.

We applied Q-learning with function approximation to this MDP using three different sets of basis functions. The first set of basis functions was obtained by dividing the state-space into a uniform grid of 0.1×0.1 rectangles. The second set was obtained by scaling the four linear surfaces $\sigma_i, i = 1, \ldots, 4$, defined at each point $(\mathbf{x}, \mathbf{y})$ as

$$\sigma_1((\mathbf{x}, \mathbf{y})) = \mathbf{x}; \qquad \sigma_2((\mathbf{x}, \mathbf{y})) = 1 - \mathbf{x};$$
$$\sigma_3((\mathbf{x}, \mathbf{y})) = \mathbf{y}; \qquad \sigma_4((\mathbf{x}, \mathbf{y})) = 1 - \mathbf{y}.$$

Finally, the last set of functions considered 4 normalized gaussian kernels centered around each of the 4 corners of the rectangle $[0; 1] \times [0; 1]$. All basis functions further considered an action-dependent component, allowing the robot to learn different values for each function.[5]

TABLE I

COMPARATIVE RESULTS FOR Q-LEARNING WITH FUNCTION APPROXIMATION FOR THE DIFFERENT SETS OF BASIS FUNCTIONS. WE PRESENT THE AVERAGE TOTAL DISCOUNTED REWARD AND STANDARD DEVIATION OBTAINED OVER $2'000$ MONTE-CARLO RUNS.

Method	**Q-learning**
Grid-based	8.474 ± 1.728
Plane-based	8.221 ± 1.751
Kernel-based	8.554 ± 1.687

For each set of basis functions, the agent was allowed to explore and learn during $20'000$ time steps, and the obtained policy was then evaluated for 100 time steps. To test the

[5] In other words, each state-dependent function $f(x)$ led to $|\mathcal{A}|$ different basis functions ξ_{i_a}, where $\xi_{i_a}(x, u) = f(x)\mathbb{I}_a(u)$ for every $(x, u) \in \mathcal{X} \times \mathcal{A}$.

obtained policy we ran $2'000$ Monte Carlo independent trials. The results are presented in Table I, corresponding to the mean and standard deviation of the obtained discounted reward with the learnt policy. Notice that all approximations lead to a similar performance. This means that all sets of basis functions provided an equally accurate approximation. Figure 2 depicts the policies learnt using each approximation. We present in Figure 3 a detail of the policy obtained using Q-learning with a plane-based approximation. This detail helps to clarify the different patterns observed in the policy representations of Figure 2. Notice that all policy representations correspond to similar behavior, which is moving to the upper right corner (as expected).

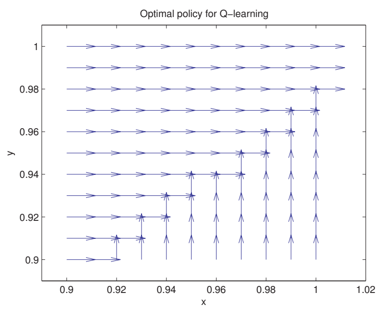


Fig. 3. Detail of Figure 2 a).

To further illustrate the similarity of the learned functions for all sets of basis functions, we present in Figure 4 a representation of the obtained value functions for each of the different sets of basis functions.

VI. CONCLUDING REMARKS

From all the works described in Section IV, the approach in [1] is the closest to the one presented here. In [1], the authors make use of the operator $\mathbf{T}^{(\lambda)}$ that is a contraction in the 2-norm. In this norm, the orthogonal projection $\mathcal{P}_\mathcal{V}$ is naturally defined and no additional conditions are required to ensure that $\mathcal{P}_\mathcal{V}$ is non-expansive.

In this paper, we extend this approach to control settings. To this end, we are interested in approximating the fixed point of the operator $\mathbf{H}$ defined in (6). This operator is a contraction in the maximum norm and no orthogonal projection is naturally defined in a space with such a norm. However, by requiring that

$$\sum_{i=1}^{N} |\xi_i(x,a)| \leq 1 \tag{9}$$

for all (x,a), we guarantee that the operator $\mathbf{H}$ is also a contraction in θ, which implies the exponential stability of the associated ODE.

Another interesting remark is that (9) implies that the functions $\{\xi_i, i = 1, \ldots, M\}$ can be used to define a soft-partition of $\mathcal{X}$. This means that the method presented here can be cast as a modified version of Q-learning with soft-state aggregation. Or, in yet another point-of-view, convergence of Q-learning with soft-state aggregation arises as a consequence of our convergence results.

It is important to emphasize that the condition (9) is not too restrictive. In fact, given a set of linearly independent functions $\Xi = \{\xi_i, i = 1, \ldots, M\}$, we can easily ensure such bound by scaling the functions in Ξ. We further emphasize that the condition (9) is trivially verified in many common approximation strategies. For example, approximation strategies based on state-space discretization [15] or convex interpolation approximators [14] verify the bound in (9), as do soft-discretization architectures [13].

We finish with four concluding remarks.

First of all, the conditions of Theorem 3.1 on the Markov chain are similar to those required in [1] and in [14]. This places those works as well as this paper in a common line of work and, basically, leading to concordant conclusions.

We also notice that, although the conditions of Theorem 3.1 are posed as sufficient (and not necessary), the non-verification of some of the conditions may lead to divergence. As a simple illustration, in the example of divergence presented in [16], the functions used in the linear approximation scheme arenot linearly independent and do not verify the bound in (9).

Another observation is related with the rate of convergence of the algorithm presented herein. Our proof of convergence uses standard results from stochastic approximation algorithms and several known results exist describing the rates of convergence for this class of methods. These results [23] lead to the interesting conclusion that the rate of convergence of our algorithm (which works on infinite state-spaces) is essentially similar to that of standard Q-learning as described in [24] (which works on finite state-spaces).

Finally, the method proposed herein is off-policy, *i.e.*, learns the value of the optimal policy while executing a different policy during learning. Also, unlike the approach in [1], we make no use of eligibility traces, which could improve the overall performance of the algorithm (the use of eligibility traces guarantees tighter error bounds). Nevertheless, there is no formal reason why the algorithm in this paper cannot be modified so as to accommodate eligibility traces. Furthermore, the convergence result presented can be of use in the analysis of an on-policy version of the algorithm, by using results on the stability of perturbed ODEs.

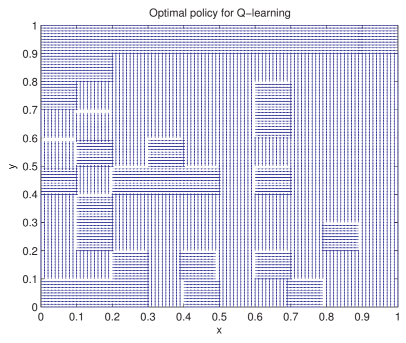
APPENDIX I
PROOF OF THEOREM 3.1

We separately establish each of the three statements in Theorem 3.1. To prove the Statement 1, we write (7) in the form

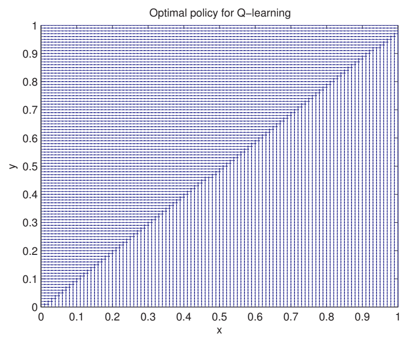
$$\theta_{t+1} = \theta_t + \alpha_{t+1} H(\theta_t, Y_{t+1}),$$

and use Theorem 17 of [25]: we show that the associated ODE has a globally asymptotically stable equilibrium point, which implies the existence of a Lyapunov function $\mathcal{U}$ verifying the requirements of the referred theorem. That establishes convergence of (7) w.p.1.

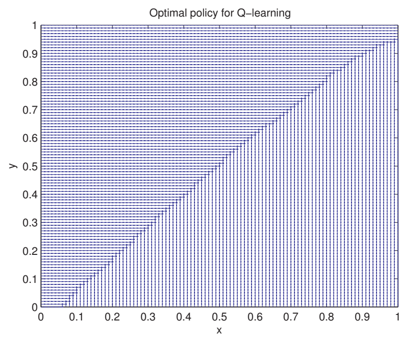
To prove Statement 2 of Theorem 3.1, we provide an interpretation of the equilibrium point of the ODE associated with algorithm (7) as the fixed point of a composite operator.



a) Grid-based approximation;

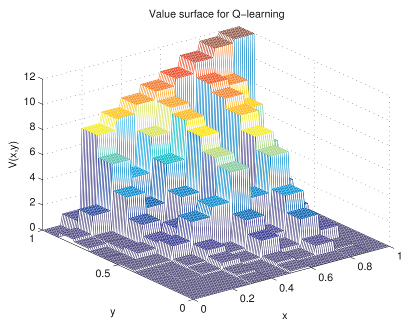


b) Linear-based approximation;

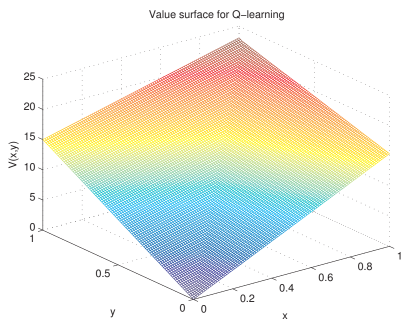


c) Kernel-based approximation.

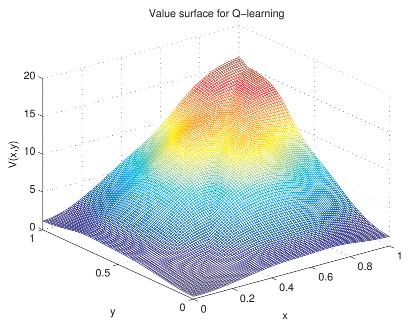
Fig. 2. Policy learnt by Q-learning with function approximation.



a) Grid-based approximation;



b) Linear-based approximation;



c) Kernel-based approximation.

Fig. 4. Representation of the optimal value-function learnt by Q-learning with function approximation.

A. Convergence of the iterates

We start by writing (7) as

$$\theta_{t+1} = \theta_t + \alpha_{t+1} H(\theta_t, Y_{t+1}),$$

where $Y_{t+1} = (X_t, A_t, X_{t+1})$. Since the chain $\{X_t\}$ is geometrically ergodic and $\delta(x,a) > 0$ for μ_X-almost all $x \in \mathcal{X}$, it follows that so is the chain $\{Y_t\}$. Let $\hat{\mu}$ be the corresponding invariant probability measure. On the other hand, since $\mathcal{X}$ is compact and $\mathcal{A}$ is finite, $\{Y_t\}$ also lies in a compact set. Finally, since the functions ξ_i are clearly bounded and so are the rewards $r(x,a,y)$, H verifies the bound

$$\|H(\theta, Y)\|_\infty \leq C(1 + \|\theta\|_\infty)$$

for some $C > 0$. Finally, the geometric ergodicity of $\{Y_t\}$ and the fact that δ does not depend on θ ensure that the requirements of Theorem 17 of [25] can be applied, and convergence of $\{\theta_t\}$ w.p.1 is established as long as the ODE

$$\dot{\theta}_t = h(\theta_t), \tag{10}$$

with

$$h(\theta) = \mathbb{E}_{\hat{\mu}}\Big[\xi(x,a)\big(r(x,a,z) + \\ + \gamma \max_{b \in \mathcal{A}} \xi^\top(z,b)\theta - \xi^\top(x,a)\theta\big)\Big], \tag{11}$$

has a globally asymptotically stable equilibrium θ^*.

Proposition 1.1: The ODE

$$\dot{\theta}_t = h(\theta_t), \tag{12}$$

where h is defined in (11), has a globally asymptotically stable equilibrium θ^* verifying the recursive relation

$$\theta^* = \mathcal{P}\mathbf{H}Q(\theta^*).$$

REMARK: The operator $\mathbf{H}$ is defined in (6) and the projection operator $\mathcal{P}$ is defined for any real function q defined on $\mathcal{X} \times \mathcal{A}$ as

$$\mathcal{P}q = \Sigma^{-1}\mathbb{E}_{\hat{\mu}}\left[\xi(x,a)q(x,a)\right].$$

Proof: We start by rewriting h as

$$h(\theta) = h_1(\theta) + h_2(\theta),$$

with

$$h_1(\theta) = \mathbb{E}_{\hat{\mu}}\left[\xi(x,a)\big(r(x,a,z) + \gamma \max_{b \in \mathcal{A}} \xi^\top(z,b)\theta\big)\right]$$

and

$$h_2(\theta) = \mathbb{E}_{\hat{\mu}}\left[\xi(x,a)\xi^\top(x,a)\theta\right].$$

Some simple calculations lead to the conclusion that

$$\|h_1(\theta_1) - h_1(\theta_2)\|_\infty \leq \gamma \|\theta_1 - \theta_2\|_\infty \tag{13}$$

and

$$\|h_2(\theta_1) - h_2(\theta_2)\|_\infty \leq \|\theta_1 - \theta_2\|_\infty. \tag{14}$$

On the other hand, any equilibrium point θ^* of (12) must verify $h(\theta^*) = 0$ or, which is the same,

$$h(\theta) = h(\theta) - h(\theta^*).$$

Explicit calculations now yield

$$\begin{aligned}\frac{d}{dt}\|\theta_t - \theta^*\|_p = \|\theta_t - \theta^*\|_p^{1-p} \sum_i (\theta_t(i) - \theta^*(i))^{p-1} \cdot \\ \cdot (h_1(\theta)_i - h_1(\theta^*)_i) - \\ - \|\theta_t - \theta^*\|_p^{1-p} \sum_i (\theta_t(i) - \theta^*(i))^{p-1} \cdot \\ \cdot (h_2(\theta)_i - h_2(\theta^*)_i),\end{aligned}$$

where we denoted by $h_1(\theta)_i$ the i^{th} component of $h_1(\theta)$ and similarly for h_2. Applying Hölder's inequality to the summations leads to

$$\frac{d}{dt}\|\theta_t - \theta^*\|_p \leq \|h_1(\theta) - h_1(\theta^*)\|_p - \|h_2(\theta) - h_2(\theta^*)\|_p.$$

Taking the limit as $p \to \infty$ and using (13) and (14) leads to

$$\frac{d}{dt}\|\theta_t - \theta^*\|_\infty \leq \gamma \|\theta_t - \theta^*\|_\infty - \|\theta_t - \theta^*\|_\infty$$

which, in turn, leads to

$$\frac{d}{dt}\|\theta_t - \theta^*\|_\infty \leq (\gamma - 1)\|\theta_t - \theta^*\|_\infty. \tag{15}$$

Let $\lambda = 1 - \gamma > 0$. Upon integration, (15) becomes

$$\|\theta_t - \theta^*\|_\infty \leq e^{-\lambda t}\|\theta_0 - \theta^*\|_\infty,$$

which establishes the existence of a globally asymptotically stable equilibrium point for (12).

It is now clear that $h(\theta^*) = 0$ is equivalent to

$$\mathbb{E}_{\hat{\mu}}\left[\xi(x,a)\big(r(x,a,z) + \gamma \max_{b \in \mathcal{A}} \xi^\top(z,b)\theta\big)\right] = \\ = \mathbb{E}_{\hat{\mu}}\left[\xi(x,a)\xi^\top(x,a)\theta\right]$$

which in turn leads to

$$\theta^* = \Sigma^{-1}\mathbb{E}_{\hat{\mu}}\left[\xi(x,a)\big(r(x,a,z) + \gamma \max_{b \in \mathcal{A}} \xi^\top(z,b)\theta\big)\right]$$

and the proof is complete. ■

The fact that the equilibrium point θ^* is globally asymptotically stable is sufficient to ensure the existence of a function $\mathcal{U}$ verifying the conditions of Theorem 17 in [25],[6] which in turn establishes the convergence of the sequence $\{\theta_t\}$ w.p.1.

B. Limit of convergence

We have established that the sequence $\{\theta_t\}$ generated by (7) converges w.p.1. The limit point θ^* is the globally asymptotically stable equilibrium of the ODE (12), verifying the following recursive relation:

$$\theta^* = \mathcal{P}\mathbf{H}Q(\theta^*)$$

or, more explicitly,

$$\theta^* = \Sigma^{-1}\mathbb{E}_Y\left[\xi(x,a)(\mathbf{H}Q_{\theta^*})(x,a)\right], \tag{16}$$

where $Q(\theta)$ is the function

$$Q_\theta(x,a) = \xi^\top(x,a)\theta.$$

[6]This guarantee arises from standard converse Lyapunov theorems.

The purpose of (7) is to approximate the optimal Q-function Q^* by an element of the linear space $\mathcal{Q}$ spanned by the basis functions $\xi_i \in \Xi$. We would like to determine the element $q^* \in \mathcal{Q}$ that, in some sense, "best approximates" Q^*. One possibility is to consider the orthogonal projection of Q^* on $\mathcal{Q}$. Denote the orthogonal projection operator on $\mathcal{Q}$ by $\mathcal{P}_{\mathcal{Q}}$.

Finding the projection of Q^* on $\mathcal{Q}$ translates into finding the function $f \in \mathcal{Q}$ verifying

$$f(x, a) = (\mathcal{P}_{\mathcal{Q}} Q^*)(x, a) = (\mathcal{P}_{\mathcal{Q}} \mathbf{H} Q^*)(x, a).$$

However, f thus defined is not a fixed point of any of the involved operators, and there is not an obvious procedure to write a stochastic approximation algorithm to find f. Therefore, we consider the function g verifying

$$g(x, a) = (\mathcal{P}_{\mathcal{Q}} \mathbf{H} g)(x, a). \tag{17}$$

The function g is a fixed point of the operator $\mathcal{P}_{\mathcal{Q}}\mathbf{H}$ and can easily be implemented using stochastic approximation. It turns out that the projection $\mathcal{P}_{\mathcal{Q}}$ can be expressed as

$$(\mathcal{P}_{\mathcal{Q}} f)(x, a) = \xi^\top(x, a)\Sigma^{-1}\mathbb{E}_Y\left[\xi(z, u)f(z, u)\right]$$

for any function f. But this means that, given the limit point θ^* of algorithm (7), the corresponding function Q_{θ^*} verifies

$$\begin{aligned} Q_{\theta^*}(x, a) &= \xi^\top(x, a)(\mathcal{P}\mathbf{H}Q_{\theta^*}) = \\ &= \xi^\top(x, a)\Sigma^{-1}\mathbb{E}_Y\left[\xi(z, u)\mathbf{H}Q_{\theta^*}(z, u)\right] = \\ &= (\mathcal{P}_{\mathcal{Q}}\mathbf{H}Q_\theta)(x, a) \end{aligned}$$

where the second equality comes from (16). Finally, this implies that Q_{θ^*} verifies the fixed point equation in (17), which establishes statement (8) of Theorem 3.1.

This concludes the proof of Theorem 3.1.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the helpful discussions with João Xavier, from ISR/IST and the comments from the anonymous reviewers. This work was partially supported by the POS_C financing program, that includes FEDER funds. The first author acknowledges the PhD grant SFRH/BD/3074/2000.

REFERENCES

[1] J. N. Tsitsiklis and B. Van Roy, "An analysis of temporal-difference learning with function approximation," *IEEE Transactions on Automatic Control*, vol. 42, no. 5, pp. 674–690, May 1996.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 3rd ed., ser. Adaptive Computation and Machine Learning Series. Cambridge, Massachusetts: MIT Press, 1998.

[3] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, pp. 9–44, 1988.

[4] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, University of Cambridge, May 1989.

[5] G. A. Rummery and M. Niranjan, "On-line Q-learning using connectionist systems," Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR 166, 1994.

[6] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, ser. Optimization and Neural Computation Series. Belmont, Massachusetts: Athena Scientific, 1996.

[7] T. Jaakkola, M. I. Jordan, and S. P. Singh, "On the convergence of stochastic iterative dynamic programming algorithms," *Neural Computation*, vol. 6, no. 6, pp. 1185–1201, 1994.

[8] S. P. Singh, T. Jaakkola, M. Littman, and C. Szepesvari, "Convergence results for single-step on-policy reinforcement-learning algorithms," *Machine Learning*, vol. 38, no. 3, pp. 287–310, 2000.

[9] G. Tesauro, "TD-Gammon, a self-teaching backgammon program, achieves master-level play," *Neural Computation*, vol. 6, no. 2, pp. 215–219, 1994.

[10] ——, "Temporal difference learning and TD-Gammon," *Communications of the ACM*, vol. 38, no. 3, pp. 58–68, 1995.

[11] G. J. Gordon, "Stable function approximation in dynamic programming," School of Computer Science, Carnegie Mellon University, Tech. Rep. CMU-CS-95-103, 1995.

[12] B. Van Roy, "Learning and value function approximation in complex decision processes," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1998.

[13] S. P. Singh, T. Jaakkola, and M. I. Jordan, "Reinforcement learning with soft state aggregation," in *Advances in Neural Information Processing Systems*, 1994, vol. 7, pp. 361–368.

[14] C. Szepesvári and W. D. Smart, "Interpolation-based Q-learning," in *Proceedings of the 21st International Conference on Machine learning (ICML'04)*. New York, USA: ACM Press, July 2004, pp. 100–107.

[15] J. Rust, "Using randomization to break the curse of dimensionality," *Econometrica*, vol. 65, no. 3, pp. 487–516, 1997.

[16] L. C. Baird, "Residual algorithms: Reinforcement learning with function approximation," in *Proceedings of the 12th International Conference on Machine Learning (ICML'95)*. San Francisco, CA: Morgan Kaufman Publishers, 1995, pp. 30–37.

[17] S. P. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability*, ser. Communications and Control Engineering Series. New York: Springer-Verlag, 1993.

[18] D. P. Bertsekas, V. S. Borkar, and A. Nedić, *Improved temporal difference methods with linear function approximation*. Wiley Publishers, 2004, ch. 9, pp. 235–260.

[19] J. A. Boyan, "Technical update: Least-squares temporal difference learning," *Machine Learning*, vol. 49, pp. 233–246, 2002.

[20] C. Ribeiro and C. Szepesvári, "Q-learning combined with spreading: Convergence and results," in *Proceedings of the ISRF-IEE International Conference: Intelligent and Cognitive Systems (Neural Networks Symposium)*, 1996, pp. 32–36.

[21] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning for control," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 75–113, 1997.

[22] D. Ormoneit and Śaunak Sen, "Kernel-based reinforcement learning," *Machine Learning*, vol. 49, pp. 161–178, 2002.

[23] M. Pelletier, "On the almost sure asymptotic behaviour of stochastic algorithms," *Stochastic Processes and their Applications*, vol. 78, pp. 217–244, 1998.

[24] C. Szepesvári, "The asymptotic convergence rates for Q-learning," in *Proceedings of Neural Information Processing Systems (NIPS'97)*, vol. 10, 1997, pp. 1064–1070.

[25] A. Benveniste, M. Métivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximations*, ser. Applications of Mathematics. Berlin: Springer-Verlag, 1990, vol. 22.