Q-Learning with Linear Function Approximation*

Francisco S. Melo** and M. Isabel Ribeiro

Institute for Systems and Robotics,
Instituto Superior Técnico,
Lisboa, Portugal
{fmelo,mir}@isr.ist.utl.pt

Abstract. In this paper, we analyze the convergence of Q-learning with linear function approximation. We identify a set of conditions that implies the convergence of this method with probability 1, when a fixed learning policy is used. We discuss the differences and similarities between our results and those obtained in several related works. We also discuss the applicability of this method when a changing policy is used. Finally, we describe the applicability of this approximate method in partially observable scenarios.

1 Introduction

Value-based methods such as TD-learning [1], Q-learning [2], SARSA [3] and others [4,5,6] have been exhaustively covered in the reinforcement learning (RL) literature and, under mild assumptions, have been proven to converge to the desired solution [7].

However, many such algorithms require explicit representation of the state-space, and it is often the case that the latter is unsuited for explicit representation. A common way to overcome this difficulty is to combine a suitable approximation architecture with one's favorite learning method [8,9]. Encouraging results were reported, perhaps the most spectacular of which by Tesauro's Gammon player [10]. Several other works provided formal analysis of convergence when RL algorithms are combined with function approximation. We refer the early works by Singh et al. [11], Gordon [12] and Van Roy [13]. A few other works further extended the applicability/performance of these methods, *e.g.*, [6, 14, 15, 16].

In this paper, we analyze the convergence of Q-learning with linear function approximation. Our approach is closely related to interpolation-based Q-learning [15] and the learning algorithm by Borkar [17]. We identify conditions that ensure convergence of our method with probability 1 (w.p.1). We interpret the obtained approximation and discuss the corresponding error bounds. We conclude the

* This work was partially supported by Programa Operacional Sociedade do Conhecimento (POS_C) that includes FEDER funds.

** The author acknowledges the PhD grant SFRH/BD/3074/2000.

N. Bshouty and C. Gentile (Eds.): COLT 2007, LNAI 4539, pp. 308–322, 2007.

paper by addressing the applicability of our methods to partially observable scenarios. [1]

2 The Framework of Markov Decision Process

A *Markov decision process* (MDP) is a tuple $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ where $\mathcal{X}$ is compact subspace of $\mathbb{R}^p$ representing the state-space and $\mathcal{A}$ is a finite set of possible actions. $\mathsf{P}_a(x, U)$ is a probability kernel determining the probability of moving from state $x \in \mathcal{X}$ to a measurable set $U \subset \mathcal{X}$ by choosing action $a \in \mathcal{A}$. The function $r : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \longrightarrow \mathbb{R}$ is a deterministic function assigning a numerical reward $r(x, a, y)$ every time a transition from x to y occurs after taking action a. The use of this function r greatly 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 $a \in \mathcal{A}$.[2] The constant $0 < \gamma < 1$ is a discount-factor.

The purpose of the agent is to maximize the expected total sum of discounted rewards, $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(X_t, A_t)\right]$, where $R(x, a)$ represents the random "reward" received for taking action $a \in \mathcal{A}$ in state $x \in \mathcal{X}$.[3] The *optimal value function* V^* is defined for each state $x \in \mathcal{X}$ as

$$V^*(x) = \max_{\{A_t\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(X_t, A_t) \mid X_0 = x\right] \tag{1}$$

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. The optimal Q-values $Q^*(x, a)$ are defined for each state-action pair $(x, a) \in \mathcal{X} \times \mathcal{A}$ as

$$Q^*(x, a) = \int_{\mathcal{X}} \Big[r(x, a, y) + \gamma V^*(y)\Big] \mathsf{P}_a(x, dy). \tag{2}$$

From Q^*, the *optimal policy* is defined as a mapping $\pi^* : \mathcal{X} \longrightarrow \mathcal{A}$ verifying

$$\pi^*(x) = \arg\max_{a \in \mathcal{A}} Q^*(x, a), \qquad \text{for all } x \in \mathcal{X}.$$

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

More generally, we define a *policy* π_t as a mapping $\pi_t : \mathcal{X} \times \mathcal{A} \longrightarrow [0, 1]$ that generates a control process $\{A_t\}$ verifying

$$\mathbb{P}\left[A_t = a \mid X_t = x\right] = \pi_t(x, a),$$

[1] Due to space limitations, we do not include the proof of the results in here and simply provide the general idea behind the proof. The details can be found in [18].

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

[3] Notice that $R(x, a)$ is random in its dependence of the next state.

for all t. Since $\pi_t(x,\cdot)$ is a probability distribution over $\mathcal{A}$, it must satisfy $\sum_{a\in\mathcal{A}} \pi_t(x,a) = 1$, for all $x \in \mathcal{X}$. A *stationary policy* is a policy π that does not depend on t. A *deterministic policy* is a policy assigning probability 1 to a single action in each state. We denote such policy as a function $\pi_t : \mathcal{X} \longrightarrow \mathcal{A}$.

Given any function $q : \mathcal{X} \times \mathcal{A} \longrightarrow \mathbb{R}$, we can define the operator

$$(\mathbf{H}q)(x,a) = \int_{\mathcal{X}} \big[r(x,a,y) + \gamma \max_{u\in\mathcal{A}} q(y,u)\big] \mathsf{P}_a(x,dy). \tag{3}$$

The function Q^* introduced above is a fixed-point of the operator $\mathbf{H}$. This operator is a contraction in the sup-norm and, theoretically, a fixed-point iteration could be used to determine Q^*. On the other hand, if P or r (or both) are not known, the *Q-learning algorithm* can be used, defined by the update rule

$$Q_{k+1}(x,a) = (1-\alpha_k)Q_k(x,a) + \alpha_k\big[R(x,a) + \gamma \max_{u\in\mathcal{A}} Q_k(X(x,a),u)\big], \tag{4}$$

where $Q_k(x,a)$ is the kth estimate of $Q^*(x,a)$, $X(x,a)$ is a $\mathcal{X}$-valued random variable obtained according to the probabilities defined by P and $\{\alpha_k\}$ is a step-size sequence. 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. The estimates Q_k converge with probability 1 (w.p.1) to Q^* as long as

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

The Q-learning algorithm was first proposed by Watkins in 1989 [2] and its convergence w.p.1 later established by several authors [19, 7].

3 *Q*-Learning with Linear Function Approximation

In this section, we establish the convergence properties of Q-learning when using linear function approximation. We identify the conditions ensuring convergence w.p.1 and derive error bounds for the obtained approximation. The results derived herein are deeply related with other approaches described in the literature, *e.g.*, [15, 17].

3.1 Combining *Q*-Learning with Linear Function Approximation

We previously suggested that a fixed-point iteration could theoretically be used to determine Q^*. This implicitly requires that the successive estimates for Q^* can be represented compactly and stored in a computer with finite memory. To solve for Q^* we can use the fixed-point iteration proposed in Section 2 or the Q-learning algorithm, if P and r are not known.

However, if $\mathcal{X}$ is an infinite set, it is no longer possible to straightforwardly apply any of the aforementioned methods. For example, the updates in (4) explicitly consider the Q-values for each individual state-action pair and there will

be infinitely many such pairs if $\mathcal{X}$ is not finite. Therefore, some compact representation of either $\mathcal{X}$ or Q^* is necessary to tackle the infinite nature of $\mathcal{X}$. In our approach, we focus on compact representations for Q^*.

In our pursuit to approximate Q^*, we consider a family of functions $\mathcal{Q} = \{Q_\theta\}$ parameterized by a finite-dimensional parameter vector $\theta \in \mathbb{R}^M$. We replace the iterative procedure to find Q^* by a suitable "equivalent" procedure to find a parameter θ^* so as to best approximate Q^* by a function in $\mathcal{Q}$. We thus move from a search in an infinite dimensional function space to a search in a finite dimensional space ($\mathbb{R}^M$). This has an immediate implication: unless if $Q^* \in \mathcal{Q}$, we will not be able to determine Q^* exactly. Instead, we will determine the fixed point of a combined operator $\mathcal{P}\mathbf{H}$, where $\mathcal{P}$ is some mapping that "projects" a function defined in $\mathcal{X} \times \mathcal{A}$ to a point in $\mathcal{Q}$.

In this paper 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 $\alpha \in \mathbb{R}$. Thus, $\mathcal{Q}$ is the linear span of some set of linearly independent functions $\xi_i : \mathcal{X} \times \mathcal{A} \longrightarrow \mathbb{R}$, and each $q \in \mathcal{Q}$ can be written as a linear combination of such ξ_i. Therefore, if $\Xi = \{\xi_1, \ldots, \xi_M\}$ is a set of linearly independent functions, we interchangeably use Q_θ and $Q(\theta)$ to denote the function

$$Q_\theta(x, a) = \sum_{i=1}^{M} \xi_i(x, a)\theta(i) = \xi^\top(x, a)\theta, \tag{5}$$

where $\theta(i)$ is the ith component of the the vector $\theta \in \mathbb{R}^M$ and $\xi_i(x, a)$ is the ith component of the vector $\xi(x, a) \in \mathbb{R}^M$.

We throughout take $\Xi = \{\xi_i, i = 1, \ldots, M\}$ as a set of M bounded, linearly independent functions verifying

$$\sum_i |\xi_i(x, a)| \leq 1 \tag{6}$$

for all $(x, a) \in \mathcal{X} \times \mathcal{A}$ and eventually introduce further restrictions on the set Ξ as needed.

3.2 Linear Approximation Using Sample-Based Projection

We now consider a sample-based approximation model that, while imposing somewhat strict conditions on the set of functions Ξ, will allow us to derive useful error bounds for the obtained approximation Q_{θ^*}. For that, we assume that the functions in Ξ verify

$$\|\xi_i\|_\infty = 1. \tag{7}$$

Clearly, if (6) and (7) simultaneously hold, linear independence of the functions in Ξ arises as an immediate consequence. [4] We take the family $\mathcal{Q}$ as the linear span of Ξ.

For each function $\xi_i \in \Xi$ we take a point (x_i, a_i) in $\mathcal{X} \times \mathcal{A}$ such that $|\xi_i(x_i, a_i)| = 1$ and denote by I the set obtained by gathering M of such points, one for each $\xi_i \in$

[4] For each function $\xi_i \in \Xi$ there is a point (x, a) such that $|\xi_i(x, a)| = 1$, as $\|\xi_i\|_\infty = 1$. Then, $\xi_j(x, a) = 0$ for all $j \neq i$ and the functions in Ξ are linearly independent.

Ξ. Let $\mathcal{B}$ is the set of all (essentially) bounded functions defined on $\mathcal{X} \times \mathcal{A}$ and taking values on $\mathbb{R}$ and define the mapping $\wp : \mathcal{B} \longrightarrow \mathbb{R}^M$ as

$$(\wp f)(i) = f(x_i, a_i), \tag{8}$$

where $(\wp f)(i)$ denotes the ith component of $\wp f$ and (x_i, a_i) is the point in I corresponding to ξ_i. $\wp f$ is properly defined for every $f \in \mathcal{B}$ and verifies

$$\|\wp f\|_\infty \leq \|f\|_\infty$$
$$\wp[\alpha f_1 + f_2] = \alpha \wp f_1 + \wp f_2.$$

Our variant of Q-learning iteratively determines the point $\theta^* \in \mathbb{R}^M$ verifying the fixed-point recursion

$$\theta^* = \wp \mathbf{H} Q(\theta^*), \tag{9}$$

where $\mathbf{H}$ is the operator defined in (3). Since $\mathbf{H}$ is a contraction in the sup-norm and $\sum_i |\xi_i(x, a)| \leq 1$, the fixed point in (9) is properly and uniquely defined.

To derive the expression of the algorithm, we remark that (9) can be explicitly written as

$$\theta^*(i) = \int_{\mathcal{X}} \delta_{(x_i,a_i)}(x, a) \int_{\mathcal{X}} \big[r(x, a, y) + \gamma \max_u \xi^\top(y, u)\theta^*\big] \mathsf{P}_a(x, dy) d\mu(x, a),$$

where μ is some probability measure on $\mathcal{X} \times \mathcal{A}$ and $\delta_{(x_i,a_i)}$ is the Dirac delta centered around (x_i, a_i). Let g_ε be a smooth Dirac approximation, such that

$$\int g_\varepsilon(x, a; y, u) d\mu(y, u) = 1$$
$$\lim_{\varepsilon \longrightarrow 0} \int g_\varepsilon(x, a; y, u) f(y, u) d\mu(y, u) = f(x, a).$$

Let π be a stochastic stationary policy and suppose that $\{x_t\}$, $\{a_t\}$ and $\{r_t\}$ are sampled trajectories from the MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ using policy π. Then, given any initial estimate θ_0, we generate a sequence $\{\theta_t\}$ according to the update rule

$$\theta_{t+1}(i) = \theta_t(i) + \alpha_t g_{\varepsilon_t}(x_i, a_i; x_t, a_t)\big[r_t + \gamma \max_{u \in \mathcal{A}} \xi^\top(x_{t+1}, u)\theta_t - \xi^\top(x_t, a_t)\theta_t\big],$$

where $\{\varepsilon_t\}$ is a sequence verifying

$$\varepsilon_{t+1} = (1 - \beta_t)\varepsilon_t.$$

More generally, we can have

$$\varepsilon_{t+1} = \varepsilon_t + \beta_t h(\varepsilon_t),$$

where h is chosen so that the ODE $\dot{x_t} = h(x_t)$ has a globally asymptotically stable equilibrium in the origin.

Under some regularity assumptions on the Markov chain $(\mathcal{X}, \mathsf{P}_\pi)$ obtained using the policy π and on the step-sizes α_t and β_t, the trajectories of the algorithm closely follow those of an associated ODE with a globally asymptotically

stable equilibrium point θ^*. Therefore, the sequence $\{\theta_t\}$ will converge w.p.1 to the equilibrium point θ^* of the ODE.

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

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

Also, define $\hat{\alpha}_t(i)$ as

$$\hat{\alpha}_t(i) = \alpha_t g_{\varepsilon_t}(x_i, a_i; x_t, a_t).$$

Theorem 1. *Let $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ be a Markov decision process and assume the Markov chain $(\mathcal{X}, \mathsf{P}_\pi)$ to be geometrically ergodic with invariant probability measure μ_X. Suppose that $\pi(x, a) > 0$ 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 functions defined on $\mathcal{X} \times \mathcal{A}$ and taking values in $\mathbb{R}$. In particular, admit the functions in Ξ to verify $\|\xi_i\|_\infty = 1$ and $\sum_i |\xi_i(x, a)| \leq 1$.

Then, the following hold:

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

$$\theta_{t+1}(i) = \theta_t(i) + \alpha_t g_{\varepsilon_t}(x_i, a_i; x_t, a_t)\big[r_t + \gamma \max_{u \in \mathcal{A}} \xi^\top(x_{t+1}, u)\theta_t - \xi^\top(x_t, a_t)\theta_t\big], \tag{10a}$$

$$\varepsilon_{t+1} = (1 - \beta_t)\varepsilon_t. \tag{10b}$$

converges w.p.1 as long as the step-size sequences $\{\alpha_t\}, \{\beta_t\}$ are such that

$$\sum_t \alpha_t = \infty \qquad \sum_t \alpha_t^2 < \infty; \tag{11a}$$

$$\sum_t \beta_t = \infty \qquad \sum_t \beta_t^2 < \infty \tag{11b}$$

$\beta_t = \mathrm{o}(\alpha_t)$ and $\{\alpha_t\}$ is built so that $\min_i \sum_t \hat{\alpha}_t(i) = \infty$.

2. ***Limit of convergence:*** *Under these conditions, the limit function $Q(\theta^*)$ of (10) verifies*

$$Q_{\theta^*}(x, a) = (\mathcal{P}\mathbf{H}Q_{\theta^*})(x, a), \tag{12}$$

where $\mathcal{P} : \mathcal{B} \to \mathcal{Q}$ denotes the operator given by

$$(\mathcal{P}Q)(x, a) = \xi^\top(x, a) \wp Q.$$

3. ***Error bounds:*** *Under these conditions, the limit function Q_{θ^*} verifies the bound*

$$\|Q(\theta^*) - Q^*\|_\infty \leq \frac{1}{1 - \gamma} \|\mathcal{P}Q^* - Q^*\|_\infty. \tag{13}$$

Proof. See [18].

3.3 Discussion

Before concluding this section, we briefly discuss the conditions of Theorem 1 and compare our results with several related works in the literature.

Convergence Conditions: In Theorem 1 we identified several conditions that guarantee convergence w.p.1 of the algorithm defined by the update rule in (10). These conditions can be classified in two main groups: *conditions on the problem* and *conditions on the algorithm*.

The fundamental condition on the model is that of *geometric ergodicity of the Markov chain* $(\mathcal{X}, \mathsf{P}_\pi)$. Geometric ergodicity ensures that the chain converges exponentially fast to stationarity and, as such, its steady-state behavior is properly captured by the sample trajectories used in the updates. This allows the analysis of convergence to be conducted in terms of a stationary "version" of it: we compare the trajectories of the algorithm with those a "mean" ODE, which is globally asymptotically stable with equilibrium point θ^*.

Moreover, geometric ergodicity also ensures that all "interesting" regions of the state-space are visited infinitely often [20]. The condition that $\pi(x, a) > 0$ for all $a \in \mathcal{A}$ and μ_X-almost every $x \in \mathcal{X}$ ensures that, in these "interesting" regions of the state-space, every action is tried infinitely often. Therefore, geometric ergodicity and the requirement that $\pi(x, a) > 0$ 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.

The conditions on the algorithm are those concerning the basis functions used and those concerning the step-size sequences ($\{\alpha_t\}$ and $\{\beta_t\}$). With respect to the former, we require that the functions are linearly independent. This is a simple way of guaranteeing (in a rather conservative way) that no two functions ξ_i lead to "colliding updates" as happens in the known counter-example presented by [21]. Furthermore, by requiring that $\sum |\xi_i(x, a)| \leq 1$ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$, we ensure that $\|Q(\theta)\|_\infty \leq \|\theta\|_\infty$, thus making $\mathbf{H}Q(\theta)$ a contraction in θ (in the sup-norm). This fact is important, for example, to ensure the existence of θ^*.

To clarify the conditions on the step-size sequences, we start by remarking that, if ε is held fixed, the algorithm will converge to a *neighborhood of the desired point in parameter space*. We could then proceed as follows. As soon as the estimates were "sufficiently close" to this neighborhood, we could decrease ε and wait for the estimates to, once again, approach a new, smaller neighborhood of the desired point. We would then decrease ε once again, etc.

This "gross" version of our algorithm illustrates the fact that ε cannot go to zero arbitrarily fast. In particular, it is necessary to ensure that each component of the estimate vector θ_t is "sufficiently" updated as ε is decreased. This clearly depends on the smooth Dirac approximation chosen. The relation between the two referred entities (g_ε and the rate of convergence of ε_t) is stated in (11).

Such condition on the step-sizes $\{\alpha_t\}$ can be ensured in different ways (for example, defining α_t from the ε-cuts of g_ε as in [15]). As one final note, we remark that the use of "broader" Dirac approximations will probably allow faster convergence of ε_t while "narrower" Dirac approximations will probably lead to slower convergence of ε_t.

Finally, one last remark to state that since the space $\mathcal{B}$ of essentially bounded functions with the sup-norm is a Banach space (with no orthogonal projection defined), we defined a projection operator $\mathcal{P}$ that is non-expansive in the sup-norm, thus making the combined operator $\mathcal{P}\mathbf{H}$ a contraction in this norm.

Related Work: The early works by Gordon [12] and Tsitsiklis and Van Roy [13] provide convergence analysis for several variations of dynamic programming using function approximation. There is also a brief discussion on how stochastic variations of these algorithms can be used. Closely related is the soft-state aggregation approach [11]. This approach uses a "soft"-partition of the state-space (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. The method uses standard Q-learning updates to determine the average Q-values for each region.

In a different work, Tsitsiklis and Van Roy [16] provide a detailed analysis of temporal difference methods for policy evaluation. Given a stationary policy π whose value function V^π is to be estimated, a parameterized linear family $\mathcal{V}$ of functions is used to approximate V^π. The authors establish the the convergence of this method w.p.1 and provide an interpretation of the obtained limit point as a fixed point of a composite operator $\mathcal{P}\mathbf{T}^{(\lambda)}$, where $\mathcal{P}$ is the orthogonal projection into $\mathcal{V}$ and $\mathbf{T}^{(\lambda)}$ is the TD operator. The authors also derive error bounds on the obtained approximation. Several authors later extended these results, *e.g.*, [6, 14, 22].

Szepesvári and Smart [15] proposed a version of Q-learning that approximates the optimal Q-values at a given set of sample points $\{(x_i, a_i), i = 1, \ldots, N\}$ and then uses interpolation to estimate Q^* at any query point. This method, dubbed *interpolation-based Q-learning* (IBQL) uses the update rule

$$\theta_{t+1}(i) = \theta_t(i) + \alpha_t(i) g_\varepsilon(x_i, a_i; x_t, a_t)\big(r_t + \max_{u \in \mathcal{A}} Q_{\theta_t}(x_{t+1}, u) - \theta_t(i)\big). \qquad (14)$$

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 $\mathcal{P}\hat{\mathbf{H}}$, where $\mathcal{P}$ is a projection-like operator and $\hat{\mathbf{H}}$ can be interpreted as a modified Bellman operator.

We emphasize the similarity between the update rules in (14) and (10). The fundamental difference between these two methods lies on the fact that IBQL only makes use of the estimated Q-function to predict the value of the next state, as seen in (14). Therefore, the updates of IBQL rely on a vector $\hat{d}_t$ of modified temporal differences with ith component given by

$$\begin{aligned}\hat{d}_t(i) &= r_t + \gamma \max_{u \in \mathcal{A}} Q_{\theta_t}(x_{t+1}, u) - \theta_t(i) = \\ &= r_t + \gamma \max_{u \in \mathcal{A}} Q_{\theta_t}(x_{t+1}, u) - Q_{\theta_t}(x_i, a_i).\end{aligned}$$

Notice that each $\hat{d}_t(i)$ is not a temporal-difference in the strict sense, since it does not provide a one-step estimation "error". This means that the information provided by $\hat{d}_t(i)$ may lead to "misleading" updates. Although not affecting the

convergence of IBQL in the long-run, IBQL may exhibit slower convergence because of this. On the other hand, if IBQL is used with a vanishing ε, the effect of these misleading updates will vanish as $t \to \infty$. In the experimental results portrayed in [15], a vanishing ε was used. Nevertheless, IBQL exhibited initially slower convergence than of other methods, probably because of this reported effect.

We also remark that, in [15], the convergence result requires the underlying Markov chain to be positive Harris and aperiodic. These conditions are actually weaker than the geometric ergodicity required by our result. However, in many practical situations, the former conditions will actually imply the latter.[5] This means that the conditions on the problem required in Theorem 1 are essentially similar to those in [15] placing the results of both papers in a common line of work and, basically, leading to concordant conclusions.

Finally, we also refer the close relation between the method in Subsection 3.2 and the algorithm described in [17]. In the aforementioned work, Borkar provides a convergence analysis of what we may refer to as *functional Q-learning*. This functional Q-learning can be seen as an extension of classical Q-learning to functional spaces, and arises from the approach proposed by Baker [23] to stochastic approximation in function spaces. The update equation for this method is fundamentally similar to (10). The main difference is that, while we consider only a fixed, finite set of points $I = \{(x_1, a_1), \ldots, (x_M, a_M)\}$, the algorithm in [17] maintains a *complete representation of* Q^*, each component of which is updated at each iteration. Clearly, maintaining such a representation of Q^* is computationally infeasible and the algorithm should instead maintain a complete record of the history of past events $\mathcal{H} = \{(x_0, a_0, r_0), \ldots, (x_t, a_t, r_t), \ldots\}$, used to estimate Q^* at a generic point (x, a).

4 Partially Observable Markov Decision Processes

Recall that, in a Markov decision process $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, an agent acts at each time instant based on the current state of the environment and so as to maximize its expected total discounted reward. However, if the current state is unknown and the agent has available only a noisy observation of it, the elegant theory and effective algorithms developed for Markov decision processes are in general not applicable, even in the simpler case of finite $\mathcal{X}$.

Partially observable Markov decision processes (POMDPs) present a complex challenge due to the remarkable complications arising from the "simple" consideration of partial state observability. Exact solution methods for POMDPs generally consist on dynamic-programming based iterative procedures and have been found computationally too expensive for systems with more than a few dozen states [24, 25]. This has led many researchers to focus on developing approximate methods using a variety of approaches. We refer to [26, 27] for good surveys on POMDP exact and approximate methods.

[5] An aperiodic, positive Harris chain is geometrically ergodic as long as supp μ_X has non-empty interior.

Some approximate solution methods rely on value-based reinforcement learning algorithms such as Q-learning. Examples include the Linear-Q algorithm [28], the SPOVA-RL algorithm [29] or the Fast-RL algorithm [30]. A thorough analysis of several such methods can also be found in [26].

In this section we discuss how our results from the previous section can be applied to POMDPs. We identify a set of conditions on POMDPs that ensure the applicability of the method in Section 3. As a side-note, we remark that the Linear-Q algorithm referred above can be cast as a simple variation of the method described in Section 3. Our analysis in this section can easily be adapted to provide a formal proof of the convergence of this algorithm.

4.1 Partial Observability and Internal State

Let $(\mathcal{X}, \mathsf{P})$ be a finite state-space Markov chain. Let $\mathcal{Z}$ be a finite set of possible observations and suppose that, at each time instant, the the state X_t of the chain is unaccessible. Instead, a random measurement Z_t is "observed" which depends on the state X_t according to an observation probability given by

$$\mathbb{P}\left[Z_t = z \mid X_t = x\right] = \mathsf{O}(x, z), \tag{15}$$

A partially observable Markov chain is a 4-tuple $(\mathcal{X}, \mathcal{Z}, \mathsf{P}, \mathsf{O})$, where $\mathcal{X}$ and $\mathcal{Z}$ are, respectively, the state and observation spaces (both considered finite) and P and O are the transition and observation probability matrices.

Let b_t be a discrete probability measure on $\mathcal{X}$ conveying the probability distribution of the state X_t over the set $\mathcal{X}$ at time instant t. Since $\mathcal{X}$ is assumed finite, b_t is a vector with xth component

$$b_t(x) = \mathbb{P}\left[X_t = x \mid \mathcal{F}_t\right], \tag{16}$$

where $\mathcal{F}_t$ is the history up to time t. Suppose that at time instant t the chain is in state $x \in \mathcal{X}$ with probability $b_t(x)$ and a transition occurs, with an observation $Z_{t+1} = z$ made at instant $t + 1$. Then it holds that

$$b_{t+1}(y) = \frac{\sum_{x \in \mathcal{X}} b_t(x)\mathsf{P}(x, y)\mathsf{O}(y, z)}{\sum_{x,w \in \mathcal{X}} b_t(x)\mathsf{P}(x, w)\mathsf{O}(w, x)}. \tag{17}$$

It is clear from (17) that b_{t+1} *is Markovian in its dependence of the past history.* Therefore, we define from b_t a sequence $\{B_t\}$ of random variables, each taking the value $B_t = b_t$ at time instant t. Since each b_t is a probability vector with, say, $n + 1$ components, B_t lies in the n-dimensional probability simplex $\mathbb{S}^n$.

Summarizing, for any partially observable Markov chain $(\mathcal{X}, \mathcal{Z}, \mathsf{P}, \mathsf{O})$ there is an equivalent fully-observable Markov chain $(\mathbb{S}^n, \hat{\mathsf{P}})$, where the kernel $\hat{\mathsf{P}}$ is given, for any $b \in \mathbb{S}^n$ and any measurable set $U \subset \mathbb{S}^n$, by

$$\hat{\mathsf{P}}(b, U) = \sum_z \sum_{x,y} b(x)\mathsf{P}(x, y)\mathsf{O}(y, z)\mathbb{I}_U(B(b, z)),$$

where $B(b,z)$ is the vector obtained from b using (17) with observation z and $\mathbb{I}_U$ is the indicator function for the set U. Notice that the xth coordinate of vector B_t describes the *belief* that the underlying state of the chain is $X_t = x$, and it is common to refer to the b vectors as *belief-states*.

Notice that, by considering the chain $(\mathbb{S}^n, \hat{\mathsf{P}})$ of beliefs instead of the partially observable chain $(\mathcal{X}, \mathcal{Z}, \mathsf{P}, \mathsf{O})$ we move from a finite, partially observable Markov chain with state-space $\mathcal{X}$ to an infinite, fully observable Markov chain with state-space $\mathbb{S}^n$. We now identify conditions on P and/or O that ensure the chain $(\mathbb{S}^n, \hat{\mathsf{P}})$ to be uniformly ergodic.

Theorem 2. *Let $(\mathcal{X}, \mathcal{Z}, \mathsf{P}, \mathsf{O})$ be a partially observable Markov chain, where the chain $(\mathcal{X}, \mathsf{P})$ is irreducible and aperiodic. Suppose that there is an observation $z \in \mathcal{Z}$ and a state $x^* \in \mathcal{X}$ such that, for all $y \in \mathcal{X}$, $\mathsf{O}(y,z) = \delta(x^*, y)$. Then, the Markov chain $(\mathbb{S}^n, \hat{\mathsf{P}})$ is uniformly ergodic.*

Proof. See [18].

4.2 POMDPs and Associated MDPs

A tuple $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$ is a *partially Observable Markov Decision Process* (POMDP), where $\mathcal{X}, \mathcal{A}, \mathsf{P}, r$ and γ are as defined in Section 2, $\mathcal{Z}$ is the observation-space and O represents the (action-dependent) observation probabilities. We consider $\mathcal{X}$, $\mathcal{A}$ and $\mathcal{Z}$ to be finite sets.

Using a development entirely similar to the one presented in the previous subsection, given a POMDP $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$ we can derive a fully observable MDP $(\mathbb{S}^n, \mathcal{A}, \hat{\mathsf{P}}, \hat{r}, \gamma)$, where, for each $a \in \mathcal{A}$, $\hat{\mathsf{P}}$ and $\hat{r}$ are defined as

$$\hat{\mathsf{P}}_a(b, U) = \sum_z \sum_{x,y} b(x)\mathsf{P}_a(x,y)\mathsf{O}_a(y,z)\mathbb{I}_U(B(b,a,z));$$

$$\hat{r}(b,a,b') = \sum_{x,y} b(x)\mathsf{P}_a(x,y)r(x,a,y),$$

where $B(b,a,z)$ is the updated probability vector given action a and observation z with yth component given by

$$B(b,a,z)_y = \frac{\sum_{x\in\mathcal{X}} b_t(x)\mathsf{P}_a(x,y)\mathsf{O}_a(y,z)}{\sum_{x,w\in\mathcal{X}} b_t(x)\mathsf{P}_a(x,w)\mathsf{O}_a(w,x)}.$$

Notice that the reward $\hat{r}(b,a,b')$ corresponds to the expected immediate reward for being in each state x with probability $b(x)$ and taking action a. As expected, it does not depend on b'.[6]

This new MDP is an infinite state-space counterpart to the partially observable Markov decision process $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$ and we are interested in applying the methods from the previous section to this continuous-state MDP.

[6] Notice that the rewards do not depend on the observations and the belief b' is a function of the current belief, action and observation, so it is natural that $\hat{r}$ is independent of b'.

Notice that, even if the complete POMDP model is known, the use of a simulation-based solution may still be preferable to the computationally heavier, exact methods. On the other hand, it may happen that the reward r is unknown and, therefore, recurring to simulation-based methods is the only alternative available. Finally, we emphasize that, in order to use the methods from the previous section, the MDP $(\mathbb{S}^n, \mathcal{A}, \hat{\mathsf{P}}, \hat{r}, \gamma)$ needs to be fully observable, *i.e.*, the beliefs b_t must be computable at every time step t. This means that the agent must know the model parameters P and O.

In the new MDP $(\mathbb{S}^n, \mathcal{A}, \hat{\mathsf{P}}, \hat{r}, \gamma)$, it is straightforward to define the optimal value function $V^* : \mathbb{S}^n \longrightarrow \mathbb{R}$, verifying

$$V^*(b) = \max_{a \in \mathcal{A}} \mathbb{E}\left[\hat{r}(b, a, b') + \gamma V^{\delta}(b')\right],$$

and the optimal Q-function, verifying

$$Q^*(b, a) = \mathbb{E}\left[r(b, a, b') + \gamma \max_{u \in \mathcal{A}} Q^*(b', u)\right].$$

More intuitive and well-known expressions for these functions can readily be obtained by replacing $\hat{\mathsf{P}}$ and $\hat{r}$ by the corresponding definitions, yielding

$$V^*(b) = \max_{a \in \mathcal{A}} \sum_{x,y \in \mathcal{X}} b(x)\mathsf{P}_a(x, y)\left[r(x, a, y) + \gamma \sum_{z \in \mathcal{Z}} \mathsf{O}_a(y, z)V^*(b_z)\right];$$

$$Q^*(b, a) = \sum_{x,y \in \mathcal{X}} b(x)\mathsf{P}_a(x, y)\left[r(x, a, y) + \gamma \sum_{z \in \mathcal{Z}} \mathsf{O}_a(y, z) \max_{b \in \mathcal{A}} Q^*(b_z, b)\right].$$

To apply the method from Section 3 to the MDP $M = (\mathbb{S}^n, \mathcal{A}, \hat{\mathsf{P}}, \hat{r}, \gamma)$ with guaranteed convergence, we need to check if M verifies all conditions *on the problem* required in Theorem 1. This condition is concerned with the geometric ergodicity of the chain obtained with the learning policy. Combining Theorem 1 with Theorem (2), it is immediate that the Q-learning algorithm with linear function approximation analyzed in Section 3 can be applied to POMDPs with guaranteed convergence, as long as the underlying MDP is ergodic and there is a *distinguishable* state $x^* \in \mathcal{X}$. We note that ergodicity of the underlying MDP is a standard assumption in classical RL methods and, therefore, partial observability simply requires the single additional condition of a distinguishable state.

5 Conclusions and Future Work

In this paper we have analyzed the convergence of Q-learning with linear function approximation. Given a linear family $\mathcal{Q}$ of functions, we defined an update rule that "relies" on a projection operator $\mathcal{P}$ defined in the space of (essentially) bounded functions. For the algorithm thus obtained we identified the conditions

under which convergence w.p.1 is guaranteed. We also showed the limit function to verify the fixed-point recursion

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

and discussed the relation between the method and results in this paper and those in related works such as [15, 17]. Finally, we showed that partially observable Markov decision processes can be addressed by RL algorithms using function approximation as long as the typical convergence conditions are verified for the underlying Markov decision process and there is, at least, one observable state.

Several important remarks are in order. First of all, the error bound in Theorem 1 is given as a function of the quantity $\|\mathcal{P}Q^* - Q^*\|$. Notice that the function $\mathcal{P}Q^*$ can be interpreted as the "best" representation of Q^* in $\mathcal{Q}$. The error bound in Theorem 1 means that the obtained approximation is, at most, "almost as good" as $\mathcal{P}Q^*$. It also means that, this approximation may be of little use, if the space $\mathcal{Q}$ poorly represents the desired function: the closest function in $\mathcal{Q}$ will still be a poor approximation, and there are no guarantees on its practical usefulness (in terms of the corresponding greedy policy). Notice nevertheless that, if $Q^* \in \mathcal{Q}$, the method will deliver the optimal function Q^*. Therefore, when using function approximation, the space $\mathcal{Q}$ should be chosen so as to include all available information regarding the true function to be estimated. The problem of how to choose the basis functions is currently the target of intense research in the RL community. Some work has been done in this area [31, 32, 33], but a lot more can be done.

A second remark concerns the usefulness of the algorithm in Section 3 if a fixed policy must be used during learning (instead of a policy that depends on the estimates Q_t). Although the result described in the paper considers a fixed learning policy, it is possible to extend this result to encompass the use of a policy π_θ that depends continuously on θ. In particular, if the following condition holds for every $(x,a) \in \mathcal{X} \times \mathcal{A}$

$$|\pi_\theta(x,a) - \pi_{\theta'}(x,a)| \leq C \|\theta - \theta'\|,$$

with $C > 0$, it is possible to extend the conclusions of Theorem 1 to algorithms using θ-dependent policies. Further work can explore results on the stability of perturbed ODEs to extend the fundamental ideas in this paper to address the convergence of on-policy learning algorithm (*e.g.*, SARSA).

Also, the methods proposed make no use of eligibility traces. It seems likely that the results in this paper can be modified so as to accommodate eligibility traces and thus improve their overall performance.

Thirdly, we comment on the results presented in Section 3. In this section, we described the use of the algorithm in Section 3 to POMDPs by considering equivalent, fully observable MDPs. Recall that tracking the state of an associated MDP consists in tracking the belief-state b_t of the original POMDP. As already stated, this implies that the agent must know the parameters P and O of the POMDP. This is less general than the approach adopted in many RL methods, where no model of the system is assumed. However, in several practical applications (*e.g.*, robotic applications) this is a reasonable assumption.

Finally, notice that the overall conditions required to ensure convergence of the methods in Section 3 in partially observable scenarios are similar to the requirements for convergence in fully observable scenarios. Convergence in partially observable scenarios simply requires one extra condition: that at least one state is identifiable. If we consider that, in many situations, *the reinforcement function provides additional information on the underlying state of the system*, the existence of a distinguishable state may be a less stringent condition than it appears at first sight. Nevertheless, it is likely that results on the ergodic behavior of the posterior probabilities of hidden Markov models may be adapted so as to alleviate this condition.

Acknowledgements

The authors would like to acknowledge the helpful discussions with Prof. João Xavier and the many useful comments from the anonymous reviewers that helped to greatly improve the paper.

References

1. Sutton, R.: Learning to predict by the methods of temporal differences. Machine Learning 3, 9–44 (1988)
2. Watkins, C.: Learning from delayed rewards. PhD thesis, King's College, University of Cambridge (May 1989)
3. Rummery, G., Niranjan, M.: On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University Engineering Department (1994)
4. Sutton, R.: DYNA, an integrated architecture for learning, planning, and reacting. ACM SIGART Bulletin 2(4), 160–163 (1991)
5. Barto, A., Bradtke, S., Singh, S.: Learning to act using real-time dynamic programming. Technical Report UM-CS-1993-002, Department of Computer Science, University of Massachusetts at Amherst (1993)
6. Boyan, J.: Least-squares temporal difference learning. In: Proc. 16th Int. Conf. Machine Learning, 49–56 (1999)
7. Bertsekas, D., Tsitsiklis, J.: Neuro-Dynamic Programming. Athena Scientific (1996)
8. Sutton, R.: Generalization in reinforcement learning: Successful examples using sparse coarse coding. Advances in Neural Information Processing Systems 8, 1038–1044 (1996)
9. Boyan, J., Moore, A.: Generalization in reinforcement learning: Safely approximating the value function. Advances in Neural Information Processing Systems 7, 369–376 (1994)
10. Tesauro, G.: TD-Gammon, a self-teaching backgammon program, achieves master-level play. Neural Computation 6(2), 215–219 (1994)
11. Singh, S., Jaakkola, T., Jordan, M.: Reinforcement learning with soft state aggregation. Advances in Neural Information Processing Systems 7, 361–368 (1994)
12. Gordon, G.: Stable function approximation in dynamic programming. Technical Report CMU-CS-95-103, School of Computer Science, Carnegie Mellon University (1995)

13. Tsitsiklis, J., Van Roy, B.: Feature-based methods for large scale dynamic programming. Machine Learning 22, 59–94 (1996)
14. Precup, D., Sutton, R., Dasgupta, S.: Off-policy temporal-difference learning with function approximation. In: Proc. 18th Int. Conf. Machine Learning, 417–424 (2001)
15. Szepesvári, C., Smart, W.: Interpolation-based Q-learning. In: Proc. 21st Int. Conf. Machine learning, 100–107 (2004)
16. Tsitsiklis, J., Van Roy, B.: An analysis of temporal-difference learning with function approximation. IEEE Transactions on Automatic Control AC-42(5), 674–690 (1996)
17. Borkar, V.: A learning algorithm for discrete-time stochastic control. Probability in the Engineering and Informational Sciences 14, 243–258 (2000)
18. Melo, F., Ribeiro, M.I.: Q-learning with linear function approximation. Technical Report RT-602-07, Institute for Systems and Robotics (March 2007)
19. Watkins, C., Dayan, P.: Technical note: Q-learning. Machine Learning 8, 279–292 (1992)
20. Meyn, S., Tweedie, R.: Markov Chains and Stochastic Stability. Springer, Heidelberg (1993)
21. Baird, L.: Residual algorithms: Reinforcement learning with function approximation. In: Proc. 12th Int. Conf. Machine Learning, 30–37 (1995)
22. Bertsekas, D., Borkar, V., Nedić, A.: 9. In: Improved temporal difference methods with linear function approximation. Wiley Publishers, 235–260 (2004)
23. Baker, W.: Learning via stochastic approximation in function space. PhD Thesis (1997)
24. Lusena, C., Goldsmith, J., Mundhenk, M.: Nonapproximability results for partially observable Markov decision processes. J. Artificial Intelligence Research 14, 83–103 (2001)
25. Papadimitriou, C., Tsitsiklis, J.: The complexity of Markov chain decision processes. Mathematics of Operations Research 12(3), 441–450 (1987)
26. Cassandra, A.: Exact and approximate algorithms for partially observable Markov decision processes. PhD thesis, Brown University (May 1998)
27. Aberdeen, D.: A (revised) survey of approximate methods for solving partially observable Markov decision processes. Technical report, National ICT Australia, Canberra, Australia (2003)
28. Littman, M., Cassandra, A., Kaelbling, L.: Learning policies for partially observable environments: Scaling up. In: Proc. 12th Int. Conf. Machine Learning, 362–370 (1995)
29. Parr, R., Russell, S.: Approximating optimal policies for partially observable stochastic domains. In: Proc. Int. Joint Conf. Artificial Intelligence, 1088–1094 (1995)
30. He, Q., Shayman, M.: Solving POMDPs by on-policy linear approximate learning algorithm. In: Proc. Conf. Information Sciences and Systems (2000)
31. Glaubius, R., Smart, W.: Manifold representations for value-function approximation in reinforcement learning. Technical Report 05-19, Department of Computer Science and Engineering, Washington University in St. Louis (2005)
32. Keller, P., Mannor, S., Precup, D.: Automatic basis function construction for approximate dynamic programming and reinforcement learning. In: Proc. 23rd Int. Conf. Machine Learning, 449–456 (2006)
33. Menache, I., Mannor, S., Shimkin, N.: Basis function adaptation in temporal difference reinforcement learning. Annals of Operations Research 134(1), 215–238 (2005)