

From $Q(\lambda)$ to Average Q-learning: Efficient Implementation of an Asymptotic Approximation

Frédéric Garcia* and Florent Serre

INRA - Unité de Biométrie et Intelligence Artificielle.
BP 27, Auzeville. 31326 Castanet Tolosan cedex, France.

*fgarcia@toulouse.inra.fr

Abstract

$Q(\lambda)$ is a reinforcement learning algorithm that combines Q-learning and TD(λ). Online implementations of $Q(\lambda)$ that use eligibility traces have been shown to speed basic Q-learning. In this paper we present an asymptotic analysis of Watkins' $Q(\lambda)$ with accumulative eligibility traces. We first introduce an asymptotic approximation of $Q(\lambda)$ that appears to be a gain matrix variant of basic Q-learning. Using the ODE method, we then determine an optimal gain matrix for Q-learning that maximizes its rate of convergence toward the optimal value function Q^* . The similarity between this optimal gain and the asymptotic gain of $Q(\lambda)$ explains the relative efficiency of the latter for $\lambda > 0$. Furthermore, by minimizing the difference between these two gains, optimal values for the λ parameter and the decreasing learning rates can be determined. This optimal λ strongly depends on the exploration policy during learning. A robust approximation of these learning parameters leads to the definition of a new efficient algorithm called AQ-learning (Average Q-learning), that shows a close resemblance to Schwartz' R-learning. Our results have been demonstrated through numerical simulations.

1 Introduction

TD(λ) and Q-learning are undoubtedly two of the most important methods that have been proposed in Reinforcement Learning. Q-learning is a simple direct algorithm for learning optimal policies, and the TD(λ) rule is used in many existing reinforcement learning and stochastic dynamic programming algorithms for efficiently evaluating a given policy.

The Q-learning algorithm basically implements the TD(0) update rule. In order to improve the convergence rate of Q-learning, it has been proposed to integrate the TD(λ), $\lambda > 0$, with the Q-learning update rule. That resulted in the family of $Q(\lambda)$ algorithms [Watkins, 1989; Peng and Williams, 1994]. It is commonly believed that $Q(\lambda)$, $\lambda > 0$, improves the Q-learning behaviour, but no definite theoretical analysis of $Q(\lambda)$ has come in support of this conjecture: there is no proof of convergence for $Q(\lambda)$, and as for TD(λ), the issue

of the rational choice of λ remains [Singh and Dayan, 1998; Kearns and Singh, 2000].

The use of $Q(\lambda)$ is hampered by another important difficulty. Eligibility trace implementations of $Q(\lambda)$ based on look-up tables require many updates of Q-value and trace components after each transition. The time complexity of this approach may be very high for large state and action spaces. In spite of several recent attempts [Cichosz, 1995; Wiering and Schmidhuber, 1997], the multi-component update rule of $Q(\lambda)$ remains problematic.

In this paper we propose a new algorithm that eliminates the above-mentioned shortcoming of $Q(\lambda)$. Our approach relies on an asymptotic approximation of Watkins' $Q(\lambda)$ (section 2 and 3), that appears to be a gain matrix variant of basic Q-learning. Then an optimal convergence rate analysis of Q-learning based on the ODE method leads us to determine an optimal gain matrix for Q-learning (section 4). The similar structure of these two gains explains the relative efficiency of $Q(\lambda)$. Furthermore, by minimizing the difference between these two gains, we are able to determine near-optimal λ and learning rates that allow an efficient implementation of a new reinforcement learning algorithm called AQ-learning (average Q-learning), which shows a close resemblance to Schwartz' R-learning [Schwartz, 1993] (section 5). We present in section 6 some experimental results that confirm the soundness of the asymptotic analysis and the near-optimal behaviour of AQ-learning.

2 The Watkins' $Q(\lambda)$ algorithm

Like the TD(λ) and Q-learning reinforcement learning algorithms, $Q(\lambda)$ can be described within the framework of Markov Decision Processes (MDP) [Puterman, 1994]. Assume a stationary infinite-horizon MDP model that is defined by a state space S of size n_S and an action space A of size n_A , by a Markovian dynamics on S characterized by the transition probabilities $p(s'|s, a)$ of moving from s to s' after applying the action a , and by the instantaneous reward functions $r(s, a, s') \in \mathbb{R}$ associated to each transition (s, a, s') .

A policy is a function $\pi : S \rightarrow A$ that assigns an action $a = \pi(s)$ to any possible state s . Given an initial state s_0 , following a policy π defines a random trajectory $(s_0, a_0, r_0) \rightarrow (s_1, a_1, r_1) \rightarrow \dots$ with $a_i = \pi(s_i)$, $r_i = r(s_i, \pi(s_i), s_{i+1})$, according to the probabilities $p(s_{i+1} | s_i, a_i)$.

The optimization problem associated to a discounted Markov Decision Problem is to find a policy π^* that maximizes for any initial state s_0 the expected sum of the discounted rewards along a trajectory $V^\pi(s_0) = E[\sum_{i=0}^{\infty} \gamma^i r(s_i, \pi(s_i), s_{i+1})]$, where $0 \leq \gamma < 1$ is the discount factor.

The value function V^π of a policy π can be directly learnt from observed trajectories by using the TD(λ) algorithm, without maintaining an estimation of the transition probabilities $p(s'|s, \pi(s))$ [Sutton, 1988]. It is also possible to learn directly an estimation of the optimal value function of the problem with the Q-learning algorithm [Watkins, 1989]. Q-learning regularly updates an estimation Q_n of the optimal Q-value function denoted by Q^* :

$$\forall s, a \quad Q^*(s, a) = \sum_{s' \in S} p(s'|s, a) \left(r(s, a, s') + \gamma V^{\pi^*}(s') \right),$$

which is characterized by the optimality equations: $\forall s, a$

$$Q^*(s, a) = \sum_{s' \in S} p(s'|s, a) \left(r(s, a, s') + \gamma \max_{a' \in A} Q^*(s', a') \right).$$

An optimal policy can then be directly derived from Q^* by $\pi^*(s) = \operatorname{argmax}_{a \in A} Q^*(s, a)$, $\forall s$.

The principle of Q(λ) is to combine Q-learning and the temporal difference error used in the TD(λ) method, in order to accelerate the convergence of Q-learning. For the discounted value function, the potentially infinite length of the trajectories leads us to consider the Q(λ) methods based on eligibility traces. After each observed transition (s_n, a_n, s_{n+1}, r_n) , these algorithms update the estimated value function Q_n by

$$\forall s, a \quad Q_{n+1}(s, a) \leftarrow Q_n(s, a) + \alpha_n(s, a) z_n(s, a) d_n \quad (1)$$

where $d_n = r_n + \gamma \max_{a' \in A} Q_n(s_{n+1}, a') - Q_n(s_n, a_n)$ is the current temporal difference, $\alpha_n(s, a)$ is the stepsize and $z_n(s, a)$ is the eligibility trace.

Usually, stepsizes decay regularly toward 0, or are very small constants. The dynamics of the eligibility trace is more complex. Like in TD(λ), the value $z_n(s, a)$ is increased for the pair (s_n, a_n) , and decays exponentially according to the parameter $\lambda \in [0, 1]$ for all pairs $(s, a) \neq (s_n, a_n)$. Following this general principle, several eligibility trace dynamics for Q(λ) have been proposed [Watkins, 1989; Peng and Williams, 1994; Singh and Sutton, 1996]. The simplest one, due to Watkins [Watkins, 1989], is directly derived from the accumulative eligibility trace of TD(λ), and has the supplementary effect of resetting to 0 the whole trace vector when the current action a_n is not a *greedy action*, that is $a_n \neq a_n^* \triangleq \operatorname{argmax}_a Q_n(s_n, a)$. The complete definition of this trace is given by:

$$\begin{aligned} z_n(s_n, a_n) &\leftarrow \begin{cases} \gamma \lambda z_{n-1}(s_n, a_n) + 1 & \text{if } a_n = a_n^* \\ 1 & \text{if } a_n \neq a_n^* \end{cases} \\ z_n(s, a) &\leftarrow \begin{cases} \gamma \lambda z_{n-1}(s, a) & \text{if } a_n = a_n^* \\ 0 & \text{if } a_n \neq a_n^* \end{cases} \end{aligned} \quad (2)$$

$\forall (s, a) \neq (s_n, a_n)$, for $n \geq 0$, with $\gamma < 1$, $\lambda \in [0, 1]$, and $\forall (s, a) z_{-1}(s, a) = 0$.

Q(λ) with accumulative eligibility trace is a direct generalization of Q-learning; Q(0) exactly corresponds to Q-learning, where $Q_n(s_n, a_n)$ is the only component updated after each transition. Unlike the convergence of Q-learning that has been proved [Bertsekas and Tsitsiklis, 1996; Borkar and Meyn, 2000], the convergence of Q(λ) and its overcoming of Q-learning have only been shown experimentally [Peng and Williams, 1994; Wiering and Schmidhuber, 1997]. In practice, the main drawback of Q(λ) is relative to the update costs of Q_n and z_n that are proportional to the size $n_S n_A$ of $S \times A$ for $\lambda > 0$. For large state-action space problems, the computation time of Q(λ), $\lambda > 0$, can be very high.

3 Asymptotic approximation of Q(λ)

In this section we intend to define an asymptotic approximation of Watkins' Q(λ) when $n \rightarrow \infty$. The following assumptions are made. At each iteration the action a_n is chosen with a semi-uniform exploration policy: a greedy action a_n^* is selected with a probability $0 < \tau < 1$, or any other action in A is chosen randomly. Asymptotically, when Q_n has converged toward Q^* , this process defines a Markov chain (s_n, a_n) on $S \times A$. We assume that this Markov chain is regular (recurrent, aperiodic, irreducible) and therefore has a unique invariant distribution $\mu > 0$.

3.1 Asymptotic accumulating trace

Let us consider a trajectory $(s_0, a_0, r_0) \rightarrow (s_1, a_1, r_1) \rightarrow \dots (s_n, a_n, r_n)$ obtained by Q(λ), such that Q_n converges toward Q^* . Generalizing our work developed for TD(λ) [Garcia and Serre, 2000], the average asymptotic behaviour of the accumulative trace $z_n(s, a)$ on this trajectory can be calculated:

Proposition 1

$$\forall s \in S, a \in A \quad \lim_{n \rightarrow \infty} E[z_n(s, a)] = \frac{\mu(s, a)}{1 - \gamma \lambda \tau}.$$

For $\tau \rightarrow 1$ the formula is similar to the one developed for TD(λ). For $\tau \rightarrow 0$, the trace is regularly set to 0 or 1, and its average value is equal to the invariant distribution μ . In the general case, we can see that the role of the trace $z_n(s, a)$ in the update rule (1) is to increase credit of state-action pairs that occur more frequently, particularly when the factor $1 - \gamma \lambda \tau$ is small.

Proposition 1 gives the average asymptotic value of $z_n(s, a)$ when $n \rightarrow \infty$. From the update rule (2), we can derive an approximation of the asymptotic expectation of the trace at time n conditioned on the current state-action pair (s_n, a_n) :

$$\bar{z}_n(s, a) = \begin{cases} \chi \mu(s, a) + 1 & \text{if } (s, a) = (s_n, a_n), a_n = \pi^*(s_n) \\ \chi \mu(s, a) & \text{if } (s, a) \neq (s_n, a_n), a_n = \pi^*(s_n) \\ 1 & \text{if } (s, a) = (s_n, a_n), a_n \neq \pi^*(s_n) \\ 0 & \text{if } (s, a) \neq (s_n, a_n), a_n \neq \pi^*(s_n) \end{cases}$$

$$\forall s, a, \text{ with } \chi = \frac{\gamma \lambda}{1 - \gamma \lambda \tau}.$$

3.2 Asymptotic $Q(\lambda)$

We consider the asymptotic approximation $AQ(\lambda)$ of $Q(\lambda)$ obtained by substituting in the $Q(\lambda)$ update rule (1) the trace factor $z_n(s, a)$ by its approximation $\bar{z}_n(s, a)$.

Let us denote by A_n the $n_S n_A \times n_S n_A$ diagonal matrix with diagonal entries $\alpha_n(s, a)$, and by Γ^z the $n_S n_A \times n_S n_A$ eligibility matrix

$$\Gamma^z = I + \chi \begin{pmatrix} & n_S & & n_S(n_A-1) \\ \vdots & \mu & \cdots & \mu & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \end{pmatrix}$$

where the state-action pairs of $S \times A$ are ordered as follows: $(s_1, \pi^*(s_1)), \dots, (s_{n_S}, \pi^*(s_{n_S})), \dots, (s_i, a_j | a_j \neq \pi^*(s_i)), \dots$ (the column headings indicate the block dimensions).

Let $D_n \in \mathbb{R}^{n_S n_A}$ be the temporal difference vector :

$$D_n^T = (0, \dots, 0, D_n(s_n, a_n) = d_n, 0, \dots, 0)$$

In vector notation, the asymptotic approximation $AQ(\lambda)$ of $Q(\lambda)$ is defined by

$$Q_{n+1} \leftarrow Q_n + A_n \Gamma^z D_n. \quad (3)$$

4 Optimal gain matrix for Q-learning

One can see that (3) is a gain matrix variant of the basic Q-learning algorithm:

$$Q_{n+1} \leftarrow Q_n + \frac{1}{n} D_n.$$

Using a gain matrix in order to guide and accelerate the convergence of a stochastic adaptive algorithm is a well-known approach in stochastic approximation theory [Benveniste *et al.*, 1990; Kushner and Yin, 1997]. In this section, we calculate the optimal gain matrix for Q-learning that maximizes its rate of convergence.

The ordinary differential equation (ODE) method provides some analytic tools for analysing the convergence of the general stochastic algorithm

$$\theta_{n+1} \leftarrow \theta_n + \frac{1}{n} H(\theta_n, X_n)$$

where θ_n is the current estimation of the target θ^* , and X_n is the observed input random vector at time n . Classic reinforcement learning algorithms like Q-learning or TD(λ) have already been analysed with the ODE method [Bertsekas and Tsitsiklis, 1996; Borkar and Meyn, 2000], in order to prove their convergence.

Another potential application of the ODE method concerns the optimization of the rate of convergence of stochastic algorithms. One can show that under some general stability conditions [Benveniste *et al.*, 1990] the new algorithm

$$\theta_{n+1} \leftarrow \theta_n + \frac{1}{n} \Gamma^* H(\theta_n, X_n)$$

where Γ^* is the gain matrix defined by

$$\Gamma^* = -h_\theta(\theta^*)^{-1},$$

and h_θ is the Jacobian matrix of the function

$$h(\theta) = \lim_{n \rightarrow \infty} E_\theta[H(\theta, X_n)]$$

still converges toward θ^* , with a minimal asymptotic variance $\lim_{n \rightarrow \infty} nE[\|\theta_n - \theta^*\|^2]$.

This approach, that has already been applied to Q-learning in finite horizon [Garcia and Ndiaye, 1998] and TD(λ) [Garcia and Serre, 2000], is developed here for Q-learning. We have $\theta_n = Q_n \in \mathbb{R}^{n_S n_A}$, $\theta^* = Q^*$, $X_n = (s_n, a_n, s_{n+1})$ and $H(\theta_n, X_n) = D_n$. The Jacobian matrix and its inverse can be calculated:

$$h_Q(Q^*) = \begin{pmatrix} M^* (\gamma P^* - I) & 0 \\ \frac{M^*}{M \gamma \bar{P}} & -\frac{1}{M} \end{pmatrix},$$

$$\Gamma^* = \begin{pmatrix} (I - \gamma P^*)^{-1} M^{*-1} & 0 \\ \gamma \bar{P} (I - \gamma P^*)^{-1} M^{*-1} & \bar{M}^{-1} \end{pmatrix},$$

where M^* is the $n_S \times n_S$ diagonal matrix with diagonal coefficients $\mu(s, \pi^*(s))$, $\bar{M}$ is the $n_S(n_A - 1) \times n_S(n_A - 1)$ diagonal matrix with diagonal coefficients $\mu(s, a)$ for $a \neq \pi^*(s)$, P^* is the $n_S \times n_S$ transition matrix of the Markov chain on S defined by π^* , and $\bar{P}$ the $n_S(n_A - 1) \times n_S$ rectangular matrix of coefficients $\bar{P}_{(s,a),s'} = p(s'|s, a)$ for $a \neq \pi^*(s)$.

By developing this expression, we show

Proposition 2

The optimal gain matrix for Q-learning is

$$\Gamma^* = M^{-1} + \gamma \begin{pmatrix} P_\gamma^* M^{*-1} & 0 \end{pmatrix}, \quad (4)$$

where

$$M = \begin{pmatrix} M^* & 0 \\ 0 & \bar{M} \end{pmatrix}$$

is the diagonal matrix with diagonal coefficients $\mu(s, a)$, and P_γ^* is the $n_S(n_A - 1) \times n_S$ rectangular matrix of coefficients

$$P_{\gamma(s,a),s'}^* = \sum_{n=0}^{\infty} \gamma^n p_n^*(s' | s, a),$$

where $p_n^*(s' | s, a)$ is the probability of moving from s to s' by applying the first action a and then by following the optimal policy π^* on the next n transitions.

This optimal gain Γ^* leads to an “ideal” algorithm, called Optimal-Q-learning:

$$Q_{n+1} \leftarrow Q_n + \frac{1}{n} \Gamma^* D_n. \quad (5)$$

This algorithm theoretically maximizes the rate of convergence of Q-learning, by minimizing its asymptotic variance $\lim_{n \rightarrow \infty} nE[\|Q_n - Q^*\|^2]$. One can see that the structure of Γ^* implies that when a greedy action is chosen in s_n , all the $Q_n(s, a)$ components have to be updated, and when an exploratory action has been selected, only $Q_n(s_n, a_n)$ is modified. Note that Γ^* is independent of the reward functions $r()$, and depends on the parameter τ that characterizes the exploration policy, through the μ invariant distribution.

5 Average Q-learning

Optimal-Q-learning cannot be directly implemented since it would require the costly online estimation of the P_γ^* coefficients. However, as shown next, it is possible to determine some λ and α_n parameters that minimize the distance between the gain matrix of $AQ(\lambda)$ and Γ^* , and that allow an efficient implementation of the corresponding $AQ(\lambda)$ algorithm.

5.1 Optimization of $AQ(\lambda)$

To be consistent with the definition of Γ^* that assumes a step-size equal to $\frac{1}{n}$, we consider the case $A_n = \frac{1}{n}A$, where

$$A = \begin{pmatrix} n_S & n_S(n_A-1) \\ A^* & 0 \\ 0 & \bar{A} \end{pmatrix}$$

is the $n_S n_A \times n_S n_A$ diagonal matrix with constant coefficients $\alpha(s, a)$. We propose to optimize the choice of the α and λ parameters by minimizing the quadratic norm $\|A\Gamma^z - \Gamma^*\|$ of the difference between the gain of $AQ(\lambda)$ and the optimal gain of Q-learning.

We have

$$A\Gamma^z = \begin{pmatrix} A^*(I + \chi M^* U^*) & 0 \\ \chi A M \bar{U} & \bar{A} \end{pmatrix},$$

where the $n_S \times n_S$ matrix U^* and the $n_S(n_A-1) \times n_S$ matrix $\bar{U}$ have coefficients equal to 1.

Making equal the lower right blocks leads to the choice $\bar{A} = \bar{M}^{-1}$. A numerical analysis of the upper left blocks, similar to the one developed in [Garcia and Serre, 2000] for TD(λ), results in the choice $A^* = M^{*-1}$, and thus $A = M^{-1}$. Then

$$A\Gamma^z = M^{-1} + \chi \begin{pmatrix} n_S & n_S(n_A-1) \\ 1 & \dots & 1 & 0 \end{pmatrix}.$$

This equality exhibits some similarity with equation (4). As we can note, the structures of the two gains are identical, which might explain the efficiency of $Q(\lambda)$ for $\lambda > 0$.

Minimizing

$$f(\chi) = \sum_{s,a,s'} \left(\chi - \gamma \frac{P_{\gamma(s,a),s'}^*}{\mu(s', \pi^*(s'))} \right)^2$$

yields

$$\chi_{opt} = \gamma \frac{1}{n_S^2 n_A} \sum_{s,a,s'} \frac{P_{\gamma(s,a),s'}^*}{\mu(s', \pi^*(s'))}.$$

Like for Γ^* , this χ_{opt} value cannot be exactly implemented in practice in $AQ(\lambda)$. However, $\chi_{opt} \approx \frac{1}{\tau} \frac{\gamma}{1-\gamma}$ for uniform MDPs where all the transition probabilities are nearly equal to $\frac{1}{n_S}$ (then $P_\gamma^*(s' | s, a) \approx \frac{1}{n_S} \frac{1}{1-\gamma}$, $\mu(s', \pi^*(s')) \approx \frac{\tau}{n_S}$ and so $\chi_{opt} \approx \frac{1}{\tau} \frac{\gamma}{1-\gamma}$). It appears experimentally that for large random MDP, this approximation is still valid.

The value $\chi_{opt} \approx \frac{1}{\tau} \frac{\gamma}{1-\gamma}$ leads to $\lambda_{opt} = \frac{1}{\gamma} \frac{\chi_{opt}}{1+\tau\chi_{opt}} \approx \frac{1}{\tau}$. This value greater than 1 could be surprising. In fact convergence of eligibility trace methods requires that $\gamma\lambda < 1$ [Bertsekas and Tsitsiklis, 1996], and so $\lambda = \frac{1}{\tau}$ could be implemented in $Q(\lambda)$ assuming $\tau > \gamma$. In case $\tau \leq \gamma$, $\frac{1}{\tau}$ is no more an admissible value for λ , and the distance is minimized for $\lambda = \frac{1}{\gamma}$ on the boundary.

We are thus able to define a new optimized update rule from the asymptotic approximation (3) where $\chi = \frac{1}{\tau} \frac{\gamma}{1-\gamma}$ and $\forall n, s, a, \alpha_n(s, a) = \frac{1}{n} \frac{1}{\mu(s, a)}$:

$$Q_{n+1} \leftarrow Q_n + \frac{1}{n} \Gamma D_n, \quad (6)$$

$$\text{where } \Gamma = M^{-1} + \frac{1}{\tau} \frac{\gamma}{1-\gamma} \begin{pmatrix} n_S & n_S(n_A-1) \\ 1 & \dots & 1 & 0 \end{pmatrix}.$$

5.2 Efficient implementation of AQ -learning

A direct implementation of the update rule (6) would have a complexity similar to $Q(\lambda)$, proportional to $n_S \times n_A$. Fortunately, an equivalent expression can be derived. It only requires a small number of updates per iteration, independent of the number of states and actions.

Proposition 3

The update rule (6) is equivalent to

$$\begin{cases} W_{n+1}(s_n, a_n) \leftarrow W_n(s_n, a_n) + \frac{1}{n} \frac{1}{\mu(s_n, a_n)} d_n, \\ \rho_{n+1} \leftarrow \rho_n + \frac{\gamma}{\tau} \frac{1}{n} d_n \text{ if } a_n = \pi^*(s_n), \\ d_n = r_n - \rho_n + \gamma \max_a W_n(s_{n+1}, a) - W_n(s_n, a_n), \end{cases}$$

and $\forall s, a \quad Q_n(s, a) = W_n(s, a) + \frac{1}{1-\gamma} \rho_n$.

Proof: From (6) we have $Q_n = \sum_{p < n} \frac{1}{p} M^{-1} D_p + \frac{1}{\tau} \frac{\gamma}{1-\gamma} \sum_{p < n} \frac{1}{p} d_p e \times \mathbb{1}_{a_p = \pi^*(s_p)}$, where e denotes the vector with coefficients equal to 1, and $\mathbb{1}_{x=y}$ equals 0 or 1 according as $x \neq y$ or $x = y$. We then define the new relative value function W_n and the scaling factor ρ_n by

$$\begin{cases} W_n = \sum_{p < n} \frac{1}{p} M^{-1} D_p, \\ \rho_n = \sum_{p < n} \frac{\gamma}{\tau} \frac{1}{p} d_p \times \mathbb{1}_{a_p = \pi^*(s_p)}. \end{cases}$$

The d_n error can be expressed as a function of ρ_n and W_n :

$$\begin{aligned} d_n &= r_n + \gamma \max_a Q_n(s_{n+1}, a) - Q_n(s_n, a_n) \\ &= r_n - \rho_n + \gamma \max_a W_n(s_{n+1}, a) - W_n(s_n, a_n). \quad \square \end{aligned}$$

Note that the greedy action a_n^* is defined by $a_n^* = \operatorname{argmax}_a Q_n(s_n, a) = \operatorname{argmax}_a W_n(s_n, a)$. The update

rule of W_n can be implemented by replacing the stepsize $\frac{1}{\mu(s_n, a_n)n}$ with its approximation $\frac{1}{N_n(s_n)p(a_n)}$, where $N_n(s_n)$ is the number of times the state s_n has been visited at time n , and $p(a_n)$ equals τ or $\frac{1-\tau}{n_A-1}$ whether $a_n = a_n^*$ or $a_n \neq a_n^*$. Similarly, the condition $a_n = \pi^*(s_n)$ for updating ρ_n might be replaced by $a_n = a_n^*$. This leads to the following algorithm AQ-learning, or Average Q-learning:

AQ-learning Algorithm

```

 $W = \rho = N = 0;$ 
Choose  $s_0$  in  $S$ ;
for  $n \leftarrow 0$  until  $N_{tot} - 1$  do
  Choose  $a_n$  in  $A$  ( $\tau$ -exploration policy);
   $N(s_n) \leftarrow N(s_n) + 1;$ 
   $(s_{n+1}, r_n) \leftarrow \text{Simulate}(s_n, a_n);$ 
   $d \leftarrow r_n - \rho + \gamma \max_a W(s_{n+1}, a) - W(s_n, a_n);$ 
   $W(s_n, a_n) \leftarrow W(s_n, a_n) + \frac{1}{N(s_n)p(a_n)}d;$ 
  if  $a_n = a_n^*$  then
     $\rho \leftarrow \rho + \frac{\gamma}{\tau} \frac{1}{n}d;$ 
for  $s \in S, a \in A$  do
   $Q(s, a) \leftarrow W(s, a) + \frac{1}{1-\gamma}\rho;$ 
return  $Q, W, \rho;$ 

```

In that algorithm, N_{tot} is the length of the learning trajectory. AQ-learning is particularly interesting, since it only requires 3 updates per iteration, namely $W_n(s_n, a_n)$, ρ_n and $N_n(s_n, a_n)$. AQ-learning should therefore improve the performance of Watkins' Q(λ) for two distinct reasons. First, AQ-learning is an optimized approximation of Q(λ) that minimizes the asymptotic variance around Q^* . Second, like Q-learning, AQ-learning is an asynchronous algorithm that only updates one component of the value function (here W_n) per iteration. It is clear that for large state-action space problems, this feature is of crucial importance regarding the time complexity of the algorithm, as it is shown in simulations in the next section.

Amazingly it appears that AQ-learning is very close to the R-learning algorithm proposed by Schwartz [Schwartz, 1993]. R-learning is supposed to converge to gain optimal policies, that maximize the average-cost

$$\rho^\pi = \lim_{N \rightarrow \infty} E \left[\frac{1}{N} \sum_{i=0}^{N-1} r(s_i, \pi(s_i), s_{i+1}) \right].$$

AQ-learning and R-learning are identical when $\gamma \rightarrow 1$, with a correct choice of the R-learning learning rates. The fact that AQ-learning is a near-optimal gain matrix variant of Q-learning could explain why several experiments have demonstrated that R-learning could outperform Q-learning, despite the fact that these two algorithms have not been designed for the same optimality criterion.

We do not provide in this article a convergence proof for AQ-learning. We think, however, that it might be possible to establish such a proof either by checking the stability condi-

tion of the gain matrix $M^{-1}\Gamma^z$ for $\chi = \frac{1}{\tau} \frac{\gamma}{1-\gamma}$, or by exploiting the ODE approach of Borkar [Borkar, 1996] for two-time scale algorithms (it might be necessary in that case to modify the ρ_n update rule).

6 Simulations

We made numerical experiments in order to study the relative performance of Q-learning, Optimal-Q-learning, Watkins' Q(λ) and AQ-learning. These reinforcement learning algorithms were applied to randomly generated Markov Decision Problems, with all components $p(s'|s, a) > 0$, and with state rewards $r(s) \in [0, 1]$.

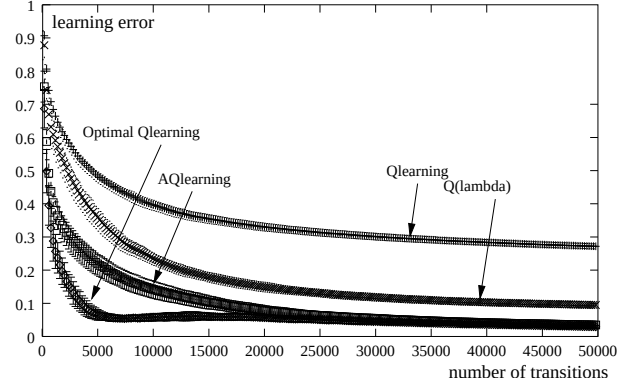


Figure 1: Learning error $\frac{\|V_n - V^*\|}{\|V^*\|}$

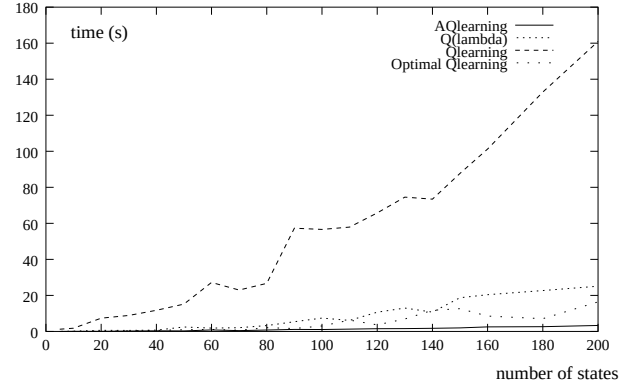


Figure 2: Computation time for $\frac{\|V_n - V^*\|}{\|V^*\|} < 0.05$

We present here two kinds of learning results that fairly describe the general behaviour of these algorithms. For both of them, the value functions Q_n , W_n and ρ_n were updated on a unique long trajectory on $S \times A$ with random initial state. All parameters were initially set to 0. The stepsizes $\alpha_n(s, a)$ were equal to $\frac{1}{N_n(s, a)}$. We chose a semi-uniform exploration strategy with $\tau > \gamma$. For each problem, V^* , Γ^* and λ_{opt} were exactly calculated from the data $p(\cdot)$ and $r(\cdot)$.

Figure 1 shows the relative learning error $\frac{\|V_n - V^*\|}{\|V^*\|}$ as a function of n (average value and standard deviation on 20

random trajectories). The size of the MDP was $n_S = 100$, $n_A = 5$, with $\gamma = 0.8$ and $\tau = 0.9$. This graph illustrates the different rates of convergence of Q-learning, AQ-learning, $Q(\lambda)$ for $\lambda = \lambda_{opt}$, and Optimal-Q-learning. These experimental results confirm that the faster rate of convergence is obtained with Optimal-Q-learning, and that AQ-learning converges nearly as fast as this ideal algorithm. AQ-learning even performs slightly better than $Q(\lambda)$ with the same learning rates and the λ_{opt} value. These three algorithms appear to be considerably faster than Q-learning.

Figure 2 illustrates for the same algorithms the different computation times required to achieve a relative learning error equal to 5%, as a function of the size n_S of the state space, for a constant number of actions $n_A = 5$, with $\gamma = 0.7$ and $\tau = 0.9$. We first observe on that graph that Q-learning is very slow, despite the fact that it only updates one component per iteration. But the main observation is that AQ-learning can be really more efficient than $Q(\lambda)$, especially when the number of states is important; for 200 states, AQ-learning is 10 times faster than $Q(\lambda)$.

7 Conclusions

We have shown that an asymptotic approximation of $Q(\lambda)$ could be efficiently implemented through two simple update rules on a relative value function $W_n(s, a)$ and on a scaling factor ρ_n . This new algorithm AQ-learning can be described as an average variant of Q-learning for the discounted criterion. Its update complexity is independent of the number of states and actions, and experimental results show that its convergence toward the optimal value function is always faster than for Q-learning or $Q(\lambda)$.

Our results have been obtained for the Watkins' $Q(\lambda)$ with accumulating eligibility trace. It would be interesting to apply the same approach to the case of replacing traces [Singh and Sutton, 1996], or to Peng and Williams' $Q(\lambda)$ [Peng and Williams, 1994] where the trace is never reset to 0, and for which an efficient implementation has been proposed by Wiering and Schmidhuber [Wiering and Schmidhuber, 1997].

Another direction for future work concerns the development of new reinforcement learning algorithms that approximate Optimal-Q-learning. Furthermore, we have seen that the Γ^* optimal gain for Q-learning depends on the exploration policy parameterized by τ . The question of finding the best value of τ remains open.

Finally, we need a better understanding of the relation between AQ-learning the Schwartz' R-learning algorithm [Schwartz, 1993].

References

- [Benveniste *et al.*, 1990] A. Benveniste, M. Metivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximation*. Springer-Verlag, Berlin, New York, 1990.
- [Bertsekas and Tsitsiklis, 1996] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont (MA), 1996.
- [Borkar and Meyn, 2000] V. S. Borkar and S. P. Meyn. The O.D.E. method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal of Control and Optimization*, 38:447–469, 2000.
- [Borkar, 1996] V. S. Borkar. Stochastic approximation with two-time scales. *System and Control Letters*, 29:291–294, 1996.
- [Cichosz, 1995] P. Cichosz. Truncating temporal differences: On the efficient implementation of TD(λ) for reinforcement learning. *Journal of Artificial Intelligence Research (JAIR)*, 2:287–318, 1995.
- [Garcia and Ndiaye, 1998] F. Garcia and S. Ndiaye. A Learning Rate Analysis of Reinforcement-Learning Algorithms in Finite-Horizon. In *International Conference on Machine Learning*, volume 15, Madison, USA, 1998.
- [Garcia and Serre, 2000] F. Garcia and F. Serre. Efficient asymptotic approximation in temporal difference learning. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, Berlin, 2000.
- [Kearns and Singh, 2000] M. Kearns and S.P. Singh. "bias-Variance" error bounds for temporal difference updates. In *Proceedings of the COLT 2000 conference*, 2000.
- [Kushner and Yin, 1997] H. J. Kushner and G. G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer, 1997.
- [Peng and Williams, 1994] J. Peng and R. J. Williams. Incremental Multi-Step Q-Learning. In *International Conference on Machine Learning*, volume 11, pages 226–232, 1994.
- [Puterman, 1994] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. Wiley-Interscience, New York, 1994.
- [Schwartz, 1993] A. Schwartz. A Reinforcement Learning Method for Maximizing Undiscounted Rewards. In *International Conference on Machine Learning*, volume 10, 1993.
- [Singh and Dayan, 1998] S. P. Singh and P. Dayan. Analytical Mean Squared Error Curves for Temporal Difference learning. *Machine Learning*, 1998.
- [Singh and Sutton, 1996] S. P. Singh and R. S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine Learning*, 1996.
- [Sutton, 1988] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.
- [Watkins, 1989] C. J. Watkins. *Learning from Delayed Rewards*. PhD thesis, Cambridge University, Cambridge, England, 1989.
- [Wiering and Schmidhuber, 1997] M. A. Wiering and J. Schmidhuber. Fast online $Q(\lambda)$. *Machine Learning*, 1997.