Markov Decision Processes in Artificial Intelligence

Groupe PDMIA

April 27, 2009

Contents

Chapter 1

Approximate Dynamic Programming

In any complex or large scale sequential decision making problem, there is a crucial need to use function approximation to represent the relevant functions such as the value function or the policy.

The Dynamic Programming (DP) and Reinforcement Learning (RL) methods introduced in the chapters **??** and **??** make the implicit assumption that the value function can be perfectly represented (i.e. kept in memory), for example by using a look-up table (with a finite number of entries) assigning a value to all possible states (assumed to be finite) of the system. Those methods are called "exact" because they provide an exact computation of the optimal solution of the considered problem (or at least, enable the computations to converge to this optimal solution). However, such methods often apply to toy problems only, since in most interesting applications, the number of possible states is so large (and possibly infinite if we consider continuous spaces) that a perfect representation of the function at all states is impossible. It becomes necessary to approximate the function by using a moderate number of coefficients (which can be stored in a computer), and therefore extend the range of DP and RL to methods using such "approximate" representations. These "approximate" methods combine DP and RL methods with function approximation tools.

EXAMPLE.– Let us come back to the example of the car where maintenance operations should be optimized (see volume 1, section **??**). We have seen that the number of possible states is very large, although the state may be sometimes factorized, (see chapter **??** of volume 2). However, even if we consider a single element of the car,

---

Chapter written by Rémi MUNOS.

the brakes for example, the number of possible states may be described by a continuous variable (thickness of break pads). The state of (this element of) the car may thus vary continuously in a given interval and all previously seen methods would not apply here since they assume that the number of possible states is finite. The tools introduced in this chapter enable to take into account this problematic of continuous state space. This point is illustrated explicitly in the optimal replacement problem, Paragraph 1.2.3).

In this chapter we study the use of function approximation (taking inspiration from statistical learning and approximation theory) for approximating the value function, and we generalize DP and RL methods to this approximate resolution setting. Performance bounds resulting from the use of approximate representations will be expressed in terms of the capacity and approximation power of the considered function spaces.

1.1. Introduction

The idea of using function approximation in DP comes back to the early days of this domain. For example, Samuel [SAM 67] used linear approximation of the value function for the game of checkers, Bellman and Dreyfus [BEL 59] used polynomials in order to increase the speed of DP. A first theoretical analysis is undertaken in [REE 77]. More recently, RL and DP combined with function approximation enabled to solve successfully several large scale applications; for example, the program *TD-Gammon* [TES 95] using a neural network, produced a world champion program in backgammon. Other application domains include operational research and task scheduling [ZHA 95], e.g. the control of a set of lifts [CRI 96], factory maintenance [MAH 97], dynamic channel allocation [SIN 97], seats allocation on planes [GOS 04]. Some specific applications are described more precisely in the latter parts of each of the two volumes.

The purpose of this chapter is to present briefly the new problems that appear when one considers *approximate solution* to PD and RL problems and to provide bounds on the performance loss resulting of these approximations.

For simplicity of presentation, we consider here only the case of a *discounted reward* problem in *infinite-time horizon*. Thus, for a given policy π (mapping an action to each possible state), the value function V^π is defined as the expectation of the sum of future discounted rewards:

$$V^\pi(s) \stackrel{\text{def}}{=} \mathbb{E}\Big[\sum_{t\geq 0} \gamma^t r(s_t, \pi(s_t)) | s_0 = s, \pi\Big]. \tag{1.1}$$

(where $\gamma \in [0, 1)$ is the discount factor)

Other criteria (e.g. finite-time horizon or undiscounted problems) are subject to similar conceptual analysis, but may show differences in the mathematical formalism.

For these extensions, we refer the reader to the work of synthesis of Bertsekas and Tsitsiklis [BER 96].

The approach followed in this chapter is to build a approximation of the optimal value function, hoping that the performance of a greedy policy with respect to (w.r.t.) such approximation will provide near-optimal performance. This hope is justified by the fact that if V is a good approximation of the optimal value function V^*, then the performance V^π of the policy π which is greedy w.r.t. V is close to the performance V^* of an optimal policy π^*. Recall that we say that a policy $\pi : S \mapsto A$ is greedy w.r.t. a function V if, at any state $s \in S$, the action $\pi(s)$ is an action that maximizes the immediate reward plus the discounted expectation of V at the next state, i.e.:

$$\pi(s) \in \arg\max_{a \in A} \big[r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \big].$$

Indeed we have the following result [BER 96, p. 262]:

PROPOSITION 1.1.– *Let V be a real-valued function defined on S and π be a policy greedy w.r.t. V. Then, the performance loss resulting from using policy π instead of the optimal policy (i.e. difference between the optimal value function V^* and the value function V^π) is bounded as:*

$$||V^* - V^\pi||_\infty \leq \frac{2\gamma}{1-\gamma} ||V - V^*||_\infty, \tag{1.2}$$

where $||\cdot||_\infty$ denotes the supremum norm, written L^∞ (i.e. $||f||_\infty \stackrel{\text{def}}{=} \max_{s \in S} |f(s)|$), and γ is the discount factor.

Let us notice that in general, the value function V^π is different from the function V. This bound gives an argument justifying our approach of searching to construct a good approximation of the optimal value function (i.e. small $||V - V^*||$) in order to deduce a policy whose performance is close to the optimum (i.e. small $||V^* - V^\pi||$). Since the proof is elementary we include it now.

PROOF.– We remind the definition of the Bellman operators L and L_π (defined in Chapter **??**, Paragraph **??**): For any real-valued function W defined over S:

$$\begin{aligned} LW(s) &\stackrel{\text{def}}{=} \max_{a \in A} \big[r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) W(s') \big], \\ L_\pi W(s) &\stackrel{\text{def}}{=} r(s,\pi(s)) + \gamma \sum_{s' \in S} p(s'|s,\pi(s)) W(s'). \end{aligned}$$

L_π also writes (in vector notation) $L_\pi W = r_\pi + \gamma P_\pi W$, where P_π is the transition matrix for policy π (i.e. whose elements (s,s') are $p(s'|s,\pi(s))$), and r_π the vector with components $r(s,\pi(s))$.

We have the property that L and L_π are contraction operators in L^∞ norm, with a contraction factor γ. This means that, for any couple of functions W_1 and W_2 defined over S, we have $||LW_1 - LW_2||_\infty \leq \gamma||W_1 - W_2||_\infty$ and similarly for the operator L_π (the proof is elementary).

From the fact that V^* and V^π are fixed points of the operators L and L_π respectively (which means that $V^* = LV^*$ and $V^\pi = L_\pi V^\pi$), by using triangle inequality, it comes that :

$$\begin{aligned} ||V^* - V^\pi||_\infty &\leq ||LV^* - L_\pi V||_\infty + ||L_\pi V - L_\pi V^\pi||_\infty \\ &= ||LV^* - LV||_\infty + \gamma||V - V^\pi||_\infty \\ &\leq \gamma||V^* - V||_\infty + \gamma(||V - V^*||_\infty + ||V^* - V^\pi||_\infty), \end{aligned}$$

were we use at the second line the fact that the policy π is greedy with respect to V, i.e. $LV = L_\pi V$. We deduce the bound on the performance loss $V^* - V^\pi$ resulting from using policy π instead of an optimal policy π^* :

$$||V^* - V^\pi||_\infty \leq \frac{2\gamma}{1-\gamma}||V^* - V||_\infty.$$

□

In order to build a good approximation of the optimal value function, we need to generalize the previously seen DP and RL algorithms. We now start by presenting the approximate value iteration algorithm. Then, in Sections 1.3 and 1.4 we consider the approximate policy iteration algorithm and the Bellman residual minimization algorithm. Finally, in Section 1.5, we explain the limits of the usual L^∞-norm analysis of DP and explore extensions to L^p-norm analysis (for $p \geq 1$) and establish preliminary links with the field of Statistical Learning. In particular, we provide finite-time (i.e. non-asymptotic) performance bounds in terms of the capacity and richness of the considered function spaces used in the approximations.

1.2. Approximate Value Iteration (AVI)

Again, we consider solving a Markov Decision Problem (MDP) [PUT 94] using a discounted criterion under an infinite horizon setting. The set of states is assumed to be large (but finite for the moment, in order to keep notations simple).

From what has been said in Chapter **??**, Volume 1, the *value iteration* algorithm consists in calculating the optimal value function V^* by successive evaluations V_n computed by the iteration $V_{n+1} = LV_n$, where L is the Bellman operator. Thanks to the contraction property (in L^∞-norm) of L, the iterates V_n converge to V^* (the fixed-point of L) when $n \to \infty$ (since we have $||V_{n+1} - V^*||_\infty = ||TV_n - TV^*||_\infty \leq \gamma||V_n - V^*||_\infty \leq \gamma^n||V_1 - V^*||_\infty$).

When the number of states is so large that an exact representation of the function V_n is impossible to memorized, we need to consider approximate representations; which results in the so-called *approximate value iteration* (AVI) algorithm. AVI is very popular and has been implemented from the early works on DP [BEL 59, SAM 59] and more recently in the context of RL [BER 96, SUT 98] with many variants, such as the *fitted Q-iteration* [ERN 05].

Let us write $\mathcal{F}$ the space of considered approximation functions. For example, $\mathcal{F}$ may be the span of a finite set of generative functions (called *features*): Any function in $\mathcal{F}$ is thus defined as a linear combination of the features weighted by some real coefficients. This is the so-called *linear approximation*.

AVI algorithm builds a sequence of functions $V_n \in \mathcal{F}$ calculated according to the iterations:

$$V_{n+1} = \mathcal{A}LV_n, \tag{1.3}$$

where L is the Bellman operator and $\mathcal{A}$ an *approximation operator* from functions in $\mathcal{F}$. For example, in the case of linear approximation, $\mathcal{A}$ is the projection operator onto $\mathcal{F}$, which means that : $\mathcal{A}f \in \mathcal{F}$ is the function in $\mathcal{F}$ with the minimal distance to f: $||\mathcal{A}f - f|| = \inf_{g\in\mathcal{F}} ||g - f||$ (for some norm $||\cdot||$).

Thus, AVI consists in a sequence of iterations where at each round, a new representation $V_{n+1} \in \mathcal{F}$ is obtained by projecting onto $\mathcal{F}$ the Bellman image of the previous approximation V_n. The iteration process (1.3) is illustrated in Figure 1.1.

When the approximation operation is performed based on data (samples) (for example when considering a projection minimizing an empirical distance), then we call this *supervised learning* or *regression* (see for example [HAS 01]); This case is illustrates in the next paragraph.

1.2.1. *Sample-based implementation and supervised learning*

For illustration, a sample-based implementation of AVI could be defined as follows: at step n, we select K states $(s_k)_{1\leq k\leq K}$ independently sampled from a given distribution μ over the state space S. We compute the Bellman image of V_n at those states, thus defining the values $\{v_k \stackrel{\text{def}}{=} LV_n(s_k)\}$. Then we make a call to a supervised learning algorithm provided with the learning data $\{(s_k, v_k)\}_{1\leq k\leq K}$ (input, desired output), which returns a function $V_{n+1} \in \mathcal{F}$ minimizing an empirical error, such as:

$$V_{n+1} = \arg\min_{f\in\mathcal{F}} \frac{1}{K} \sum_{1\leq k\leq K} \big(f(s_k) - v_k\big)^2. \tag{1.4}$$

This minimization problem is defined in terms of the quadratic norm L^2, like this is the case in least-squares methods, locally linear regression, neural networks, and

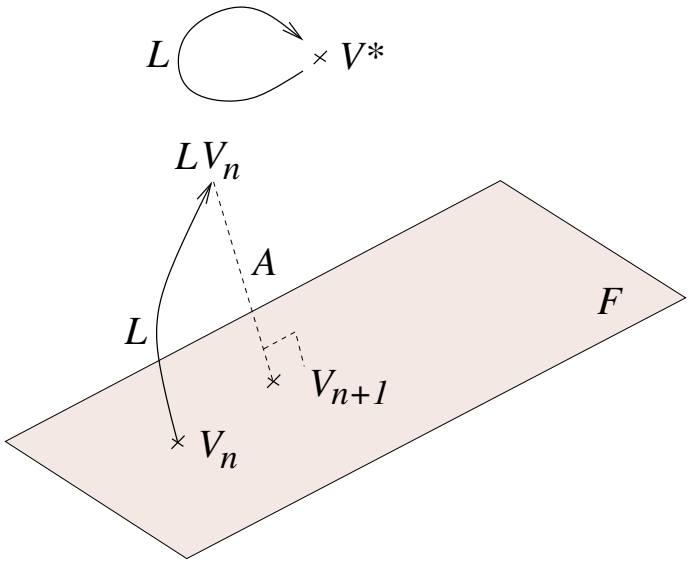

Figure 1.1. *Schematic representation of a AVI iteration: the approximation space $\mathcal{F}$ is a vector space of finite dimension. $V_n \in \mathcal{F}$ is the approximation at time n. The Bellman operator L is applied to V_n (in general, LV_n does not belong to $\mathcal{F}$, i.e. it is not representable in this approximation architecture), then the projection $\mathcal{A}$ onto $\mathcal{F}$ defines the next approximation $V_{n+1} = \mathcal{A}LV_n$. The optimal value function V^* (fixed-point of L) is also shown.*

many other supervised learning algorithms. Of course, there exists other minimization problems using different norms, such as the L^1-norm (absolute value) or its variants (such as the ϵ-insensitive L^1 norm used in the Support Vectors Machines [VAP 98]) as well as regularized version [**?**] which are often used. This regression problem is a specific case of supervised learning (or statistical learning). We will not go further in the details of the important issues (overfitting, bias-variance tradeoff) of this domain and refer the interested reader to usual references, such as [HAS 01].

Let us simply mention that linear approximation consists in performing a projection onto a vector space spanned by a finite family of features, which includes decomposition with splines, Fourier basis, radial basis functions, wavelets. Sometimes, a better regression is obtained when the family of generative functions upon which the projection is performed is chosen according to the smoothness and regularities of the function we wish to approximate. Such cases, referred to as *non-linear approximation*, may be particularly efficient when the target function possesses local regularities (for example, in adaptive wavelet bases [MAL 97] such functions may be represented sparsely, i.e. with a small number of non-zero coefficients). Greedy algorithms, like *matching pursuit* and other variants [DAV 97] select the best basis functions among a redundant dictionnary. Approximation theory studies the approximation error in terms

of the regularities of the target function [DEV 98]. In statistical learning, examples of other non-linear approximation tools that are very popular are *neural networks*, *locally weighted regression* [ATK 97], *Support Vector Machines* and *kernel methods* in *Reproducing Kernel Hilbert Spaces* [VAP 97, VAP 98].

1.2.2. *Analysis of the AVI algorithm*

Consider the AVI algorithm defined by the iteration (1.3) and define:

$$\varepsilon_n \stackrel{\text{def}}{=} LV_n - V_{n+1} \tag{1.5}$$

the *approximation error* at round n. In general, AVI does not converge to the optimal value function V^* (in opposition to the value iteration algorithm) since V^* usually does not belong to the representation space $\mathcal{F}$. Actually, even if $V^* \in \mathcal{F}$, we would have no guarantee that the iterates V_n converge to V^*. In reality, AVI may oscillate or even diverge, as illustrated in very simple examples described in [BAI 95, TSI 96a] and [BER 96, p. 334]. However, this algorithm is very popular because it has shown good results in real applications.

In order to understand the reason for this variety of observed behaviors depending on the applications, we wish to analyze the AVI algorithm and establish performance bounds. A first result provides a bound on the performance loss (w.r.t. optimal performance) resulting from the use of a policy π_n greedy w.r.t. V_n, as a function of the approximation errors ε_n.

PROPOSITION 1.2 [BER 96].– *Write π_n a greedy policy w.r.t. the approximate V_n, and V^{π_n} the value function corresponding to that policy, we have :*

$$\limsup_{n\to\infty} ||V^* - V^{\pi_n}||_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \limsup_{n\to\infty} ||\varepsilon_n||_\infty. \tag{1.6}$$

PROOF.– From (1.2) applied to V_n, we deduce:

$$||V^* - V^{\pi_n}||_\infty \leq \frac{2\gamma}{1-\gamma} ||V^* - V_n||_\infty. \tag{1.7}$$

Moreover:

$$\begin{aligned} ||V^* - V_{n+1}||_\infty &\leq ||LV^* - LV_n||_\infty + ||LV_n - V_{n+1}||_\infty \\ &\leq \gamma ||V^* - V_n||_\infty + ||\varepsilon_n||_\infty. \end{aligned}$$

Now, taking the limit sup, it comes:

$$\limsup_{n\to\infty} ||V^* - V_n||_\infty \leq \frac{1}{1-\gamma} \limsup_{n\to\infty} ||\varepsilon_n||_\infty,$$

which, combined to (1.7), leads to (1.6). □

Notice that this bound makes use of the L^∞-norm of the approximation errors ε_n, which means that it depends on the worst possible error $\varepsilon_n(s)$ over all the domain (when s sweeps S). This uniform error is general difficult to control, especially for large state space problems. In addition, it is not very useful from a practical point of view since most function approximation techniques and supervised learning algorithms solve, as illustrated in Paragraph 1.2.1, an empirical minimization problem in L^2 or L^1 norm. We will see in Section 1.5 a possible extension of Proposition 1.2 to L^p-norms (for $p \geq 1$). However, let us mention existing works [GOR 95, GUE 01] on function approximation using L^∞-norm such as the *averagers* in the field of DP.

1.2.3. *Numerical illustration*

Here we illustrate the behavior of the AVI algorithm for an optimal replacement problem, excerpted from [RUS 96]. We also show on this example that the previous results generalize naturally to the case of continuous state spaces.

A one-dimensional variable $s_t \in S \stackrel{\text{def}}{=} [0, s_{\max}]$ measures the accumulated use of a product (for example the odometer may measure the global state of a car). $s_t = 0$ denotes a brand new product. At each discrete time t (for example each year), there are two possible decisions: either keep ($a_t = \text{K}$), or replace ($a_t = \text{R}$) the product, in which case, an additional cost $C_{remplace}$ (for selling the current product and buying a new one) is suffered.

We assume transitions follow an exponential law with parameter β (with a truncated tail): if the next state y is larger than a given maximal value $s_{\max}$ (for example a critical state for the car) then a new state is immediately drawn and a penalty cost $C_{death} > C_{remplace}$ is received. Thus, by writting $p(\cdot|s,a)$ the transition density function of the next state given that the current state is s and the chosen action is $a \in \{\text{K}, \text{R}\}$ (which means that for any subset $B \subset S$ the probability of being in B at the next state is $\int_B p(ds'|s,a)$), we define:

$$p(s'|s,\text{R}) \stackrel{\text{def}}{=} \begin{cases} q(s') & \text{if } s' \in [0, s_{\max}], \\ 0 & \text{otherwise;} \end{cases}$$

$$p(s'|s,\text{K}) \stackrel{\text{def}}{=} \begin{cases} q(s'-s) & \text{if } s' \in [s, s_{\max}], \\ q(s'-s+s_{\max}) & \text{if } s' \in [0, s), \\ 0 & \text{otherwise;} \end{cases}$$

with $q(s) \stackrel{\text{def}}{=} \beta e^{-\beta s}/(1 - e^{-\beta s_{\max}})$ (the truncated exponential density).

The immediate cost function (opposite of a reward) $c(s)$ is the sum of a slowly increasing monotonous function (which may correspond to maintenance costs for the car) and a punctually discontinuous cost function (insurance fees for example). The immediate cost function and the optimal value function (numerically computed by using a very fine grid) are shown on Figure 1.2 for the numerical values: $\gamma = 0.6$, $\beta = 0.6$, $C_{remplace} = 50$, $C_{mort} = 70$ and $s_{\max} = 10$.

We consider the implementation of the sampled-based AVI algorithm described in Section 1.2.1. The states $\{s_k \stackrel{\text{def}}{=} ks_{\max}/K\}_{0 \leq k < K}$ (with $K = 200$) are uniformly sampled over the domain S. The approximation function space $\mathcal{F}$ is defined by the vector space of dimension $M = 20$ generated by a cosine family:

$$\mathcal{F} \stackrel{\text{def}}{=} \left\{ V_\alpha(s) \stackrel{\text{def}}{=} \sum_{m=1}^{M} \alpha_m \cos(m\pi \frac{s}{s_{\max}}) \right\}_{\alpha \in \mathbb{R}^M}.$$

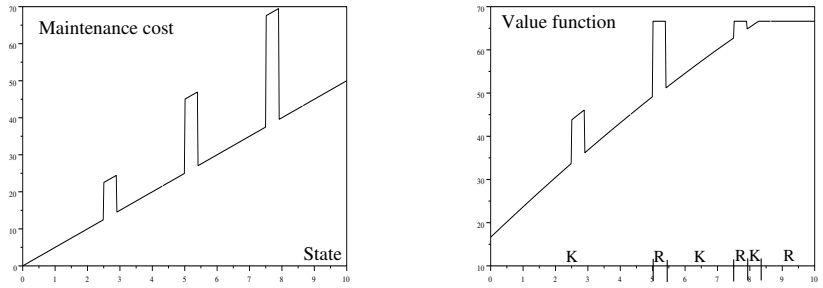

Figure 1.2. *Immediate cost function c and optimal value function V^*. The letters R and K on the right figure indicate the optimal policy (greedy w.r.t. V^*) as a function of the state.*

Thus, at each step n, a new approximation $V_{n+1} \in \mathcal{F}$ is obtained by solving the least-squares fitting problem:

$$V_{n+1} = \arg\min_{f \in \mathcal{F}} \frac{1}{K} \sum_{k=1}^{K} \left[f(s_k) - LV_n(s_k) \right]^2.$$

We start with an initial value function $V_0 = 0$. Figure 1.3 represents the first iteration: the values $\{LV_0(s_k)\}_{1 \leq k \leq K}$ are shown by the crosses on the left figure and the corresponding best fit $V_1 \in \mathcal{F}$ (best approximation in the space $\mathcal{F}$). Figure 1.4 illustrates analogously the second iteration. Figure 1.5 shows the approximate value function $V_{20} \in \mathcal{F}$ obtained after 20 iterations. In this simulation, the AVI algorithm performs well and a good approximation of the optimal value function V^* is obtained in $\mathcal{F}$.

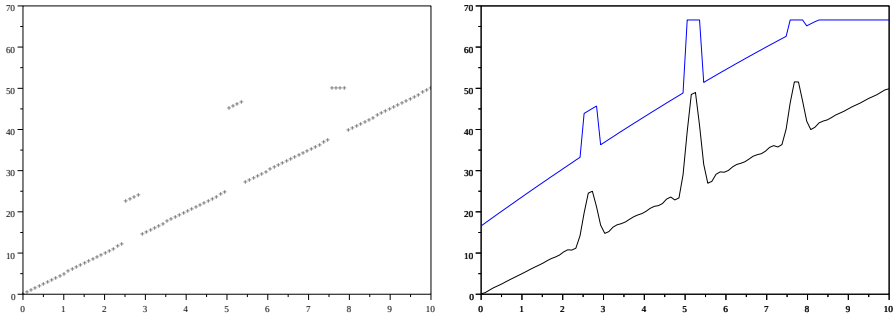

Figure 1.3. *Values $\{LV_0(s_k)\}_{1\leq k\leq K}$ (left figure), best fit $V_1 \in \mathcal{F}$ of LV_0, as well as the optimal value function (right figure)*

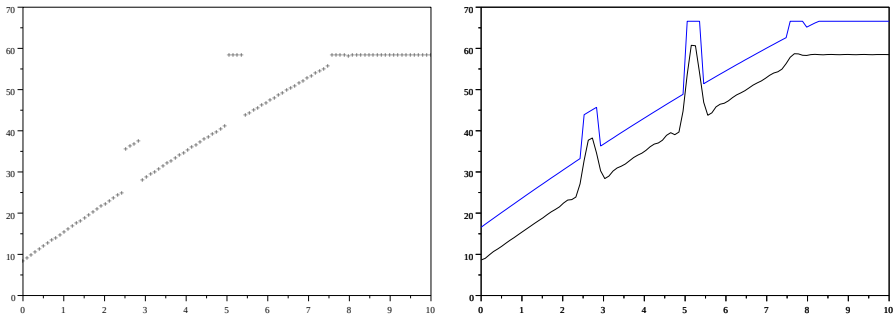

Figure 1.4. *Values $\{LV_1(s_k)\}_{1\leq k\leq K}$ (left figure), best fit $V_2 \in \mathcal{F}$ of LV_1, as well as the optimal value function (right figure)*

1.3. Approximate Policy Iteration (API)

We now consider the *Approximate Policy Iteration* algorithm (API) [BER 96] which generalizes the policy iteration algorithm described in Chapter **??** to the use of function approximation. The algorithm is defined by the iteration of the two steps:

– *Approximate policy evaluation step*: for a policy π_n, an approximation V_n of the value function V^{π_n} is generated;

– *Policy improvement step*: A new policy π_{n+1} is generated as a greedy policy w.r.t. V_n, i.e. such that for all $s \in S$,

$$\pi_{n+1}(s) \in \arg\max_{a\in A} \left[r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a)V_n(s)\right]. \tag{1.8}$$

Paragraph 1.3.1 provides a bound on the performance loss $||V^{\pi_n} - V^*||_\infty$ resulting from the use of the policies generated by the API algorithm, instead of the optimal one,

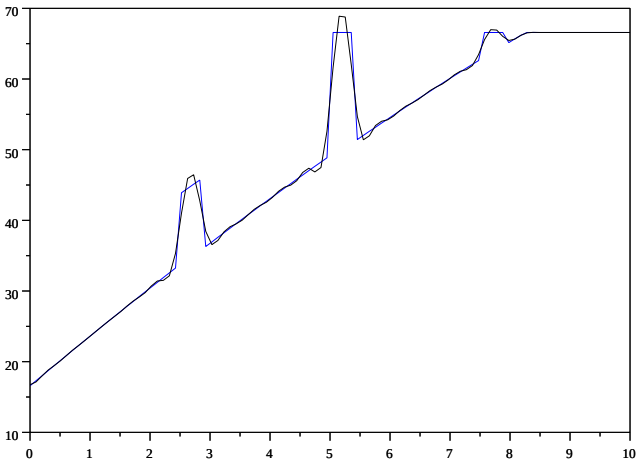

Figure 1.5. *Approximation $V_{20} \in \mathcal{F}$ obtained at the 20^{th} iteration and the optimal value function*

in terms of the value function approximation errors $||V_n - V^{\pi_n}||_\infty$. Paragraph 1.3.2 describes the approximate policy evaluation step and paragraph 1.3.3 details the case of *linear approximation* providing an extension of TD(λ) algorithm as well as least-squares methods, and finally an implementation using Q-value function that does not need the *prior* knowledge of the transion probabilities is presented.

1.3.1. *Analysis in L^∞-norm of the API algorithm*

We write π_n the policy derived by API algorithm at step n. Let V_n be an approximation of the value function V^{π_n} and π_{n+1} a policy greedy w.r.t. V_n (defined by (1.8)). The next result (stated in [BER 96, p. 276]) gives a bound on the loss $V^* - V^{\pi_n}$ resulting from the use of the policy π_n instead of the optimal policy, as a function of the *approximation errors* $V_n - V^{\pi_n}$ in L^∞-norm.

PROPOSITION 1.3.– *We have:*

$$\limsup_{n\to\infty} ||V^* - V^{\pi_n}||_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \limsup_{n\to\infty} ||V_n - V^{\pi_n}||_\infty. \tag{1.9}$$

PROOF.– Define $e_n \stackrel{\text{def}}{=} V_n - V^{\pi_n}$ the *approximation error* of V^{π_n} by V_n, $g_n \stackrel{\text{def}}{=} V^{\pi_{n+1}} - V^{\pi_n}$ the *performance gain from one iteration to the next* and $l_n \stackrel{\text{def}}{=} V^* - V^{\pi_n}$ the *performance loss* due to the use of policy π_n instead of π^* (the later being the quantity that we wish to bound).

If the approximation error is small, then the performance of the next policy cannot be much worst than the current one. Indeed, using vector notation, we have, componentwise:

$$\begin{aligned} g_n &= V^{\pi_{n+1}} - V^{\pi_n} \\ &= L_{\pi_{n+1}} V^{\pi_{n+1}} - L_{\pi_{n+1}} V^{\pi_n} + L_{\pi_{n+1}} V^{\pi_n} - L_{\pi_{n+1}} V_n \\ &\quad + L_{\pi_{n+1}} V_n - L_{\pi_n} V_n + L_{\pi_n} V_n - L_{\pi_n} V^{\pi_n} \\ &\geq \gamma P_{\pi_{n+1}} g_n - \gamma (P_{\pi_{n+1}} - P_{\pi_n})\, e_n \end{aligned}$$

(where we used the definition of the Bellman operators and the fact that π_{n+1} is greedy w.r.t. V_n, thus $L_{\pi_{n+1}} V_n = L V_n \geq L_{\pi_n} V_n$, so $(I - \gamma P_{\pi_{n+1}}) g_n \geq -\gamma (P_{\pi_{n+1}} - P_{\pi_n}) e_n$. Moreover, the matrix $P_{\pi_{n+1}}$ is a stochastic matrix thus its eigenvalues have a modulus less or equal to one. And since $\gamma < 1$, the matrix $I - \gamma P_{\pi_{n+1}}$ does not have null eigenvalue and thus is invertible. Since its inverse $(I - \gamma P_{\pi_{n+1}})^{-1}$, which also writes $\sum_{t \geq 0} (P_{\pi_{n+1}})^t$, only contains positive elements, we deduce that:

$$g_n \geq -\gamma (I - \gamma P_{\pi_{n+1}})^{-1} (P_{\pi_{n+1}} - P_{\pi_n})\, e_n. \tag{1.10}$$

Now we can bound the loss l_{n+1} at the next iteration in terms of the current loss l_n: since $L_{\pi^*} V_n \leq L_{\pi_{n+1}} V_n$, we have:

$$\begin{aligned} l_{n+1} &= V^* - V^{\pi_{n+1}} \\ &= L_{\pi^*} V^* - L_{\pi^*} V^{\pi_n} + L_{\pi^*} V^{\pi_n} - L_{\pi^*} V_n + L_{\pi^*} V_n - L_{\pi_{n+1}} V_n \\ &\quad + L_{\pi_{n+1}} V_n - L_{\pi_{n+1}} V^{\pi_n} + L_{\pi_{n+1}} V^{\pi_n} - L_{\pi_{n+1}} V^{\pi_{n+1}} \\ &\leq \gamma [P_{\pi^*} l_n - P_{\pi_{n+1}} g_n + (P_{\pi_{n+1}} - P_{\pi^*}) e_n]. \end{aligned}$$

From which we deduce, using (1.10), that:

$$\begin{aligned} l_{n+1} &\leq \gamma P_{\pi^*} l_n + \gamma [\gamma P_{\pi_{n+1}} (I - \gamma P_{\pi_{n+1}})^{-1} (P_{\pi_{n+1}} - P_{\pi_n}) + P_{\pi_{n+1}} - P_{\pi^*}] e_n \\ &\leq \gamma P_{\pi^*} l_n + \gamma [P_{\pi_{n+1}} (I - \gamma P_{\pi_{n+1}})^{-1} (I - \gamma P_{\pi_n}) - P_{\pi^*}] e_n. \end{aligned}$$

Writting $f_n \stackrel{\text{def}}{=} \gamma [P_{\pi_{n+1}} (I - \gamma P_{\pi_{n+1}})^{-1} (I - \gamma P_{\pi_n}) - P_{\pi^*}] e_n$, this last inequality may rewrite:

$$l_{n+1} \leq \gamma P_{\pi^*} l_n + f_n.$$

Taking the limit superior, it comes:

$$(I - \gamma P_{\pi^*}) \limsup_{n \to \infty} l_n \leq \limsup_{n \to \infty} f_n,$$

and using the same argument as before, we deduce:

$$\limsup_{n\to\infty} l_n \leq (I-\gamma P_{\pi^*})^{-1} \limsup_{n\to\infty} f_n. \tag{1.11}$$

This inequality having only positive terms, is preserved in norm:

$$\begin{aligned}\limsup_{n\to\infty} ||l_n||_\infty &\leq \frac{\gamma}{1-\gamma} \limsup_{n\to\infty} ||P_{\pi_{n+1}}(I-\gamma P_{\pi_{n+1}})^{-1}(I+\gamma P_{\pi_n}) + P_{\pi^*}||_\infty ||e_n||_\infty \\ &\leq \frac{\gamma}{1-\gamma}(\frac{1+\gamma}{1-\gamma}+1) \limsup_{n\to\infty} ||e_n||_\infty = \frac{2\gamma}{(1-\gamma)^2} \limsup_{n\to\infty} ||e_n||_\infty\end{aligned}$$

(where we used the fact that all stochastic matrices P have a norm $||P||_\infty = 1$).

□

1.3.2. *Approximate policy evaluation*

We now study the approximation policy evaluation step of API, i.e. for a given policy π we wish to build a good approximation of the value function V^π.

Let us remind that the value function V^π is defined as the expectation of the sum of discounted rewards, when one follows the policy π, see (1.1), and that V^π solves the Bellman equation: $V^\pi = L_\pi V^\pi$, where L_π is the Bellman operator defined by $L_\pi W \stackrel{\text{def}}{=} r_\pi + \gamma P_\pi W$, for any vector W.

Several methods enable to find an approximation of V^π:

– *iterative methods*, analogously to the AVI algorithm, where the operator L_π combined with an approximation operator $\mathcal{A}$ is iterated according to: $V_{n+1} = \mathcal{A}L_\pi V_n$. A result similar to that in Proposition 1.2 may easily be deduced;

– *linear system methods*, since the Bellman operator L_π is affine, V^π is the solution of a linear system: $(I-\gamma P_\pi)V^\pi = r_\pi$, and usual methods for solving approximately linear systems may apply. The main drawback of those methods is their computational complexity when the number of states is large;

– *Monte-Carlo (MC) methods*. From the definition of the value function (1.1), an unbiased estimate of $V^\pi(s)$ is obtained by simulating $M \geq 1$ trajectories starting from the initial state s, using policy π, and taking the empirical average of the sum of discounted rewards obtained along those M trajectories. The variance of this estimate is $O(1/M)$. If we repeat this process from initial states $\{s_k\}_{1\leq k\leq K}$ sampled from a distribution μ over S, and derive estimates $\{v_k\}$ of $\{V^\pi(s_k)\}$, then a good approximation in $\mathcal{F}$ may be found by solving the least-squares fitting problem:

$$\arg\min_{f\in\mathcal{F}} \frac{1}{K}\sum_{k=1}^{K}(f(s_k)-v_k)^2.$$

Again, this is a projection of V^π onto $\mathcal{F}$ (using the empirical L^2-norm defined by the samples). There exists several variance reduction techniques that enable to improve the accuracy of the estimate. For example, from a first rough approximation of the value function, one may use a MC method estimating the residual which enables to make a correction in order to improve the current estimate, and this process may be iterated (recursive Monte-Carlo methods) [MUN 06];

– *Temporal Difference (TD) algorithms*. TD algorithms [SUT 88] are based on stochastic approximation algorithms [KUS 97] for finding the fixed-point of contractant operators. The generalization of those methods to approximate functions is not obvious in general (no convergence proof for $\lambda < 1$) but such approaches have been applied with great success, for example in the game of backgammon [TES 95]. In the case of linear approximation (the method will be described in the next section), there exists theoretical guarantees on the resulting performance [TSI 96b];

– *Least-squares methods*. In the case of linear approximation, one may also consider very efficient methods based on least-squares fitting. This setting is consider in the next paragraph.

1.3.3. *Linear approximation and least-squares methods*

In this section we consider the specific case of linear approximation. This case has been often studied in combination with TD(λ) algorithms [TSI 96b] (see Section **??**) or least-squares methods (*Least Squares Temporal Differences* LSTD(0) [BRA 96], LSTD(λ) [BOY 99]) and applied, with success, to control problems [LAG 03].

The value function is approximated by a linear combination of K basis functions called *features* $(\phi_k)_{1 \leq k \leq K}$, which means that the approximation space $\mathcal{F}$ is a vector space spanned by those features:

$$\mathcal{F} \stackrel{\text{def}}{=} \{V_\alpha(s) \stackrel{\text{def}}{=} \sum_{k=1}^{K} \alpha_k \phi_k(s), \alpha \in \mathbb{R}^K\}.$$

Our goal is to find a parameter $\alpha \in \mathbb{R}^K$ such that V_α is close to V^π. Let us start with the direct extension of TD(λ) algorithm previously seen (Chapter **??**).

1.3.3.1. *TD(λ)*

TD(λ) algorithm is defined like in Chapter **??**. We use an eligibility trace $z \in \mathbb{R}^K$ with same dimension (K) as the parameter α, initialized to zero. From an initial state, we generate a trajectory $(s_0, s_1, s_2, \dots)$ by using policy π. At each time t, we compute the temporal difference for the current approximation V_α:

$$d_t \stackrel{\text{def}}{=} r(s_t, \pi(s_t)) + \gamma V_\alpha(s_{t+1}) - V_\alpha(s_t)$$

and update both the parameter α and the trace z:

$$\begin{aligned} \alpha_{t+1} &= \alpha_t + \eta_t d_t z_t, \\ z_{t+1} &= \lambda\gamma z_t + \phi(s_{t+1}), \end{aligned}$$

where the η_t is a decreasing sequence and $\phi : S \to \mathbb{R}^K$ the vector of components ϕ_k.

This algorithm builds a sequence of approximations V_{α_t} that converges, under some ergodicity assumption of the corresponding Markov chain to a function whose approximation error (distance to V^π) is bounded in terms of the minimal approximation error using functions in $\mathcal{F}$.

PROPOSITION 1.4 [TSI 96B].– *Assume that the steps (η_t) satisfy $\sum_{t\geq 0}\eta_t = \infty$ and $\sum_{t\geq 0}\eta_t^2 < \infty$, there exists a distribution μ over S such that $\forall s, s' \in S$, $\lim_{t\to\infty} P(s_t = s'|s_0 = s) = \mu(s')$ and that the vectors $(\phi_k)_{1\leq k\leq K}$ are linearly independent. Then α_t converges. Write α^* its limit. We have:*

$$||V_{\alpha^*} - V^\pi||_\mu \leq \frac{1-\lambda\gamma}{1-\gamma} \inf_\alpha ||V_\alpha - V^\pi||_\mu, \tag{1.12}$$

where $||\cdot||_\mu$ means the L^2-norm weighted by the distribution μ, i.e. $||f||_\mu \stackrel{\text{def}}{=} \left[\sum_{s\in S} f(s)^2\mu(s)\right]^{1/2}$.

When $\lambda = 1$, we obtain the result that the Monte-Carlo estimate gives the best approximation of V^π in $\mathcal{F}$, i.e. the projection of V^π onto $\mathcal{F}$. Now, if $\lambda < 1$, the approximation quality deteriorates (introduction of a bias), but the variance of the estimate is smaller, thus approximating its value up to some given accuracy may be easier.

From its stochastic approximation aspect, TD(λ) is very costly in terms of experimental data, in the sense that it requires the observation of many transitions $s_t, a_t \to s_{t+1}$ (and several times the same transitions) in order to see the parameter α converging. This problem has motivated the introduction of least-squares methods which are much more parsimonious in terms of data.

1.3.3.2. *Least Squares methods*

Least-squares methods (*Least Squares Temporal Differences* [BRA 96, BOY 99, LAG 03]) are based on the property that since the approximate function V_α is linear in α and the operator L_π is affine, then the Bellman mapping: $\alpha \to R_\alpha \stackrel{\text{def}}{=} L_\pi V_\alpha - V_\alpha$ is also affine. Since the Bellman residual of the desired value function V^π is zero, i.e. $L_\pi V^\pi - V^\pi = 0$, it seems reasonable to search for a parameter α such that the Bellman residual R_α be as close as possible to 0. Two approaches are generally considered (see [SCH 03, MUN 03]):

– *the quadratic residual (QR) solution*: the parameter α_{QR} minimizes the norm of the Bellman residual R_α (see illustration Figure 1.6):

$$\alpha_{\text{QR}} = \arg \min_{\alpha \in \mathbb{R}^K} ||R_\alpha||_\mu,$$

for some norm $|| \cdot ||_\mu$;

– *the temporal difference (TD) solution*: the parameter α_{TD} is such that the residual $R_{\alpha_{\text{TD}}}$ is orthogonal to all features ϕ_k, thus to $\mathcal{F}$ (see Figure 1.6). Thus $V_{\alpha_{\text{TD}}}$ is the fixed-point of the combined Bellman operator L_π followed by the orthogonal projection $\mathcal{A}$ onto $\mathcal{F}$ (according to the L^2-norm weighted by the distribution μ). It is the same solution as that obtained by TD(λ) for $\lambda = 0$ (which justifies its name).

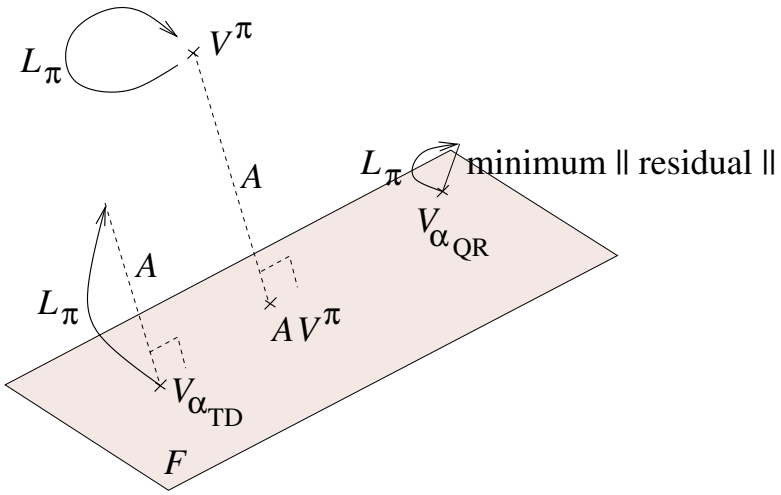

Figure 1.6. *Approximation of the value function V^π in $\mathcal{F}$. The best possible approximation $\mathcal{A}V^\pi$ is the projection of V^π onto $\mathcal{F}$. The quadratic residual solution $V_{\alpha_{QR}}$ minimizes (in $\mathcal{F}$) the norm $||L_\pi V_\alpha - V_\alpha||$. The temporal difference solution $V_{\alpha_{TD}}$ is such that $\mathcal{A}L_\pi V_{\alpha_{TD}} = V_{\alpha_{TD}}$.*

In both cases, the parameter α is obtained by solving a linear system of size K (the number of parameters). Those methods are called *projection methods* [JUD 98] because we search for a parameter α such that the residual is orthogonal to a set of test functions (the features ϕ_k in the TD case, the derivatives $\partial_{\alpha_k} R_\alpha$ in the QR case). Let us consider the corresponding linear systems.

Quadratic residual solution: since the mapping $\alpha \rightarrow R_\alpha$ is affine, the mapping $\alpha \rightarrow ||R_\alpha||^2_\mu$ (for a L^2 norm weighted by μ) is quadratic. Its minimum (obtained by setting its gradient to zero) is thus the solution to the linear system: $A\alpha = b$, with the

squared matrix A and the vector b (of size K) being defined as:

$$\left\{ \begin{array}{rcll} A_{ij} & \stackrel{\text{def}}{=} & \langle \phi_i - \gamma P_\pi \phi_i, \phi_j - \gamma P_\pi \phi_j \rangle_\mu, & \text{for } 1 \leq i,j \leq K \\ b_i & \stackrel{\text{def}}{=} & \langle \phi_i - \gamma P_\pi \phi_i, r_\pi \rangle_\mu, & \text{for } 1 \leq i \leq K \end{array} \right. \tag{1.13}$$

where the inner product $\langle u, v \rangle_\mu$ of two functions u and v (defined on S) is defined by $\langle u, v \rangle_\mu \stackrel{\text{def}}{=} \sum_{s \in S} u(s)v(s)\mu(s)$.

This system always possesses a (unique) solution when $\mu > 0$ and the features ϕ_k are linearly independent (since in that case, the matrix A is positive definite). Let us notice that this problem may be considered as a linear regression problem with another set of basis functions $\{\psi_i \stackrel{\text{def}}{=} \phi_i - \gamma P_\pi \phi_i\}_{i=1..K}$ where it comes down to finding α that minimizes $||\alpha \cdot \psi - r_\pi||_\mu$. Usual supervised learning methods may thus be considered.

When μ is the stationnary distribution associated to the policy π (this means that we have $\mu P_\pi = \mu$, i.e. $\mu(s) = \sum_{s'} p(s|s', \pi(s'))\mu(s')$ for all $s \in S$), one may deduce a bound on the approximation error $V^\pi - V_{\alpha_{\text{QR}}}$ in terms of the minimized residual or in terms of the distance between V^π and $\mathcal{F}$ (in L^2-norm with weight μ):

PROPOSITION 1.5.– *We have:*

$$\begin{aligned} ||V^\pi - V_{\alpha_{QR}}||_\mu &\leq \frac{1}{1-\gamma} ||R_{\alpha_{QR}}||_\mu = \frac{1}{1-\gamma} \inf_{V_\alpha \in \mathcal{F}} ||L_\pi V_\alpha - V_\alpha||_\mu \quad (1.14) \\ &\leq \frac{1+\gamma}{1-\gamma} \inf_{V_\alpha \in \mathcal{F}} ||V^\pi - V_\alpha||_\mu. \quad (1.15) \end{aligned}$$

PROOF.– Since μ is the stationnary distribution associated to π, we have the property that $||P_\pi||_\mu = 1$ (see for example [TSI 96b]). In addition, for all α, we have:

$$R_\alpha = L_\pi V_\alpha - V_\alpha = (I - \gamma P_\pi)(V^\pi - V_\alpha), \tag{1.16}$$

thus by considering α_{QR}, we deduce that $V^\pi - V_{\alpha_{\text{QR}}} = (I - \gamma P_\pi)^{-1} R_{\alpha_{\text{QR}}}$. Thus in norm:

$$\begin{aligned} ||V^\pi - V_{\alpha_{\text{QR}}}||_\mu &\leq ||(I - \gamma P_\pi)^{-1}||_\mu ||R_{\alpha_{\text{QR}}}||_\mu \\ &\leq \sum_{t \geq 0} \gamma^t ||P_\pi||_\mu^t ||R_{\alpha_{\text{QR}}}||_\mu \leq \frac{1}{1-\gamma} ||R_{\alpha_{\text{QR}}}||_\mu, \end{aligned}$$

which proves (1.14). Moreover, taking the norm of (1.16), it comes $||R_\alpha||_\mu \leq (1+\gamma)||V^\pi - V_\alpha||_\mu$ and (1.15) follows. □

Temporal Difference solution: The TD solution, i.e. the fixed-point of the combined Bellman operator L_π followed by the projection $\mathcal{A}$ onto $\mathcal{F}$ (using norm $||\cdot||_\mu$), is obtained by solving the linear system $A\alpha = b$ with the matrix A and the vector b:

$$\left\{ \begin{array}{rcll} A_{ij} & \stackrel{\text{def}}{=} & \langle \phi_i, \phi_j - \gamma P_\pi \phi_j \rangle_\mu, & \text{for } 1 \leq i,j \leq K, \\ b_i & \stackrel{\text{def}}{=} & \langle \phi_i, r_\pi \rangle_\mu, & \text{for } 1 \leq i \leq K. \end{array} \right. \tag{1.17}$$

We should be careful here, because the invertibility of the matrix A depends on the considered distribution μ. It is invertible when the features (ϕ_i) are linearly independent and when μ is the stationnary distribution associated to the policy π [MUN 03]. In that case, the obtained solution is the same as that obtained by TD(0) algorithm [SCH 03]. A consequence of Proposition 1.4 is the bound on the approximation error in terms of the distance between V^π and $\mathcal{F}$ (in L^2-norm weighted by μ):

$$||V^\pi - V_{\alpha_{\text{TD}}}||_\mu \leq \frac{1}{1-\gamma} \inf_{V_\alpha \in \mathcal{F}} ||V^\pi - V_\alpha||_\mu.$$

The generalization of least squares methods to TD(λ) systems with $\lambda > 0$ (i.e. providing the TD(λ) solution) may be found in [BOY 99].

Let us notice that the size of the linear system is K, the number of coefficients (or number of features), which in general is much smaller than the number of states (the latter possibly being infinite). However, in order to use this method, one needs to be able to compute the result of the transition operator P_π applied to the features ϕ_k, as well as the inner products (weighted sum over all states for which $\mu > 0$). This problem is all the more troublesome when one considers the RL setting when the transition probabilities are unknown from the learning agent. In addition, when this evaluation method is used in an API algorithm, the lack of knowledge of those probabilities makes the policy improvement step (1.8) problematic. Those problems are solved by introducing, like in Chapter **??**, an approximation of the state-action value function Q. We now explain this implementation.

1.3.3.3. *Linear approximation of the state-action value function*

Here we do not assume anymore that the transition probabilities $p(s'|s,a)$ are known from the learning agent. Rather, we now assume that the agent has access to *generative model* [KAK 03] which enables to sample a successor state $s' \sim p(\cdot|s,a)$ in any state s action a, and thus to generate trajectories when following a given policy.

Here we consider an approximation of the state-action value function (or Q-value function) Q^π [WAT 89, SUT 98] instead of the value function V^π. We remind that Q^π is defined, for any couple (s,a), by the immediate reward when action a is chosen in

state s plus the expected sum of discounted rewards when we use policy π afterwards, i.e. by using the definition of V^π:

$$Q^\pi(s,a) \stackrel{\text{def}}{=} r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^\pi(s').$$

The representations in terms of Q-value function or V-value function are equivalent: Q^π may be expressed using V^π as shown above, and symmetrically V^π is defined from Q^π according to: $V^\pi(s) = Q^\pi(s, \pi(s))$. However, the benefit of the Q-value function is that the greedy policy is very easily deduced: for any $s \in S$, the greedy policy w.r.t. V^π in s is $\arg\max_{a \in A} Q^\pi(s,a)$.

In a way similar to what has been done in the previous paragraph, we consider the linear approximation space:

$$\mathcal{F} \stackrel{\text{def}}{=} \left\{ Q_\alpha(s,a) \stackrel{\text{def}}{=} \sum_{k=1}^{K} \alpha_k \phi_k(x,a), \alpha \in \mathbb{R}^K \right\},$$

where the features ϕ_k are now defined over the product space $S \times A$.

API algorithm using Q-value function representation is defined as follows: at round n, the approximate policy evaluation step computes an approximation Q_n of Q^{π_n}; the policy improvement step defines the next policy π_{n+1} as:

$$\pi_{n+1}(s) \stackrel{\text{def}}{=} \arg\max_{a \in A} Q_n(s,a).$$

The two kinds of least-squares methods for policy evaluation apply immediately. In addition, the matrix A and vector b of the linear systems (1.13) and (1.17) may be estimated from observed data, as explained in [LAG 03]: a data base D is built from a set of transitions. At each transition (state s, action a) to state $s' \sim p(\cdot|s,a)$ with reward r, we add to D, the data (s,a,s',r). Those data about transitions and rewards are built incrementally [BOY 99] or from the observation of trajectories induced by different policies [LAG 03], or also from data coming from prior knowledge about the state dynamics.

From this data base D, at each round n of the API algorithm, in order to approximately evaluate the policy π_n, we select in D the data $\{(s_m, a_m, s'_m, r_m)\}_{1 \le m \le M}$ such that the chosen action corresponds to the policy under evaluation (i.e. $a_m = \pi_n(s_m)$). From this selected set of data, we define an unbiased estimate of A and b for the TD system (1.17): for $1 \le i, j \le K$:

$$\left\{ \begin{array}{rcl} \widehat{A}_{ij} & \stackrel{\text{def}}{=} & \frac{1}{M} \sum_{m=1}^{M} \phi_i(s_m,a_m)[\phi_j(s_m,a_m) - \gamma \phi_j(s'_m, \pi_n(s'_m))], \\ \widehat{b}_i & \stackrel{\text{def}}{=} & \frac{1}{M} \sum_{m=1}^{M} \phi_i(s_m,a_m) r_m. \end{array} \right.$$

Indeed, since the next states s' are generated according to $p(\cdot|s,a)$, we have the property that $\phi_j(s', \pi_n(s'))$ is an unbiased estimate of the operator P_{π_n} applied to ϕ_j in s, i.e. $\sum_{s' \in S} p(s'|s,a)\phi_j(s', \pi_n(s'))$. Moreover, from the law of large numbers, when the number of sampled data M is large, the empirical average over those M samples concentrates around the corresponding expectation, i.e. the inner products in (1.17). Thus $\widehat{A}$ and $\widehat{b}$ are unbiased and consistent estimates of A and b with a variance of order $O(1/M)$. When the system (1.17) is invertible, the solution $\widehat{\alpha}$ of the approximated system $\widehat{A}\alpha = \widehat{b}$ tends to the solution α_{TD} of the temporal difference system (1.17) when $M \to \infty$.

In a similar way, we could think that:

$$\frac{1}{M}\sum_{m=1}^{M}\left[\phi_i(s_m,a_m) - \gamma\phi_i(s'_m, \pi_n(s'_m))\right]\left[\phi_j(s_m,a_m) - \gamma\phi_j(s'_m, \pi_n(s'_m))\right] \quad (1.18)$$

would provide an unbiased estimate of the element A_{ij} for the quadratic residual system (1.13). However this is not true (see [SUT 98, p. 220] or [LAG 03, MUN 03]). This estimate would actually lead to a parametrization that tends to reduce the variance of the Q-value function of the successor states. The problem comes from the fact that the random variables $\phi_i(s', \pi_n(s'))$ and $\phi_j(s', \pi_n(s'))$ are correlated. Several ways to recover an unbiased estimate of A and b are:

– for each couple (s_m, a_m), use two independent samples s'_m and s''_m drawn from $p(\cdot|s_m, a_m)$ by using the generative model, in order to decorrelate $\phi_i(s'_m, \pi_n(s'_m))$ and $\phi_j(s''_m, \pi_n(s''_m))$. Then, an unbiased estimate of A and b for the quadratic residual system (1.13) is:

$$\begin{cases} \widehat{A}_{ij} & \stackrel{\text{def}}{=} & \frac{1}{M}\sum_{m=1}^{M}\left[\phi_i(s_m,a_m) - \gamma\phi_i(s'_m, \pi_n(s'_m))\right] \\ & & \left[\phi_j(s_m,a_m) - \gamma\phi_j(s''_m, \pi_n(s''_m))\right], \\ \widehat{b}_i & \stackrel{\text{def}}{=} & \frac{1}{M}\sum_{m=1}^{M}\left[\phi_i(s_m,a_m) - \gamma\phi_i(s'_m, \pi_n(s'_m))\right] r_m; \end{cases}$$

– If one only possesses a single sample s'_m per couple (s_m, a_m) (for instance because the data have been generated by following trajectories) we can consider the nearest-neighbor $s_{m'}$ of s_m such that the data $(s_{m'}, a_m, s'_{m'}, r_{m'})$ is in D. This introduces a bias (due to the distance between the samples s_m and $s_{m'}$) which depends on the smoothness of the transition probabilities and the density of points. The corresponding estimate of A and b for the system (1.13) is deduced analogously;

– A third approach, analyzed in [ANT 08], consists in modifying the criterion to be minimized by subtracting from the residual ($||L_\pi Q_n - Q_n||^2_\mu$) a term ($||\mathcal{A}L_\pi Q_n - L_\pi Q_n||^2_\mu$ where $\mathcal{A}L_\pi Q_n$ is the projection onto $\mathcal{F}$ of $L_\pi Q_n$) whose estimate (from data) has the same bias as that of the residual (1.18), killing consequently the bias of the resulting estimate. We do not further describe this approach but simply mention the fact that in the specific case of linear approximation, the corresponding solution is the same as that of the temporal difference system, but this method extends naturally to non-linear approximation. See [**?**] for such an implementation using penalization.

The algorithms that have been presented in this section are efficient in terms of samples, since the observed transition and reward data $s, a \rightarrow s', r$ are memorized and enable to find directly the parameter α by solving a linear system. Incremental versions are of course possible.

1.4. Direct minimization of the Bellman residual

In addition to the usual value and policy iteration methods previously seen, we mention here a method which aims at directly minimizing the norm of the Bellman residual (for the operator L). The idea is very simple: since the optimal value function V^* is the fixed-point of the operator L, i.e. the norm of its residual $||LV^* - V^*||$ is zero, we may wish to find the minimum in a function space $\mathcal{F}$ of the norm of the residual:

$$\inf_{V_\alpha \in \mathcal{F}} ||LV_\alpha - V_\alpha||, \tag{1.19}$$

where $||\cdot||$ is a given norm.

The following results tells us that if the residual is well minimized (in L^∞-norm), then the performance of the corresponding greedy policy is close to the optimum.

PROPOSITION 1.6 [WIL 93].– *Let V be a function defined over S and π a greedy policy w.r.t. V. The performance loss resulting from using the policy π instead of an optimal policy is bounded in terms of the Bellman residual of V as:*

$$||V^* - V^\pi||_\infty \leq \frac{2}{1-\gamma} ||LV - V||_\infty. \tag{1.20}$$

PROOF.– Since $L_\pi V = LV \geq L_{\pi^*} V$, we have:

$$\begin{aligned} V^* - V^\pi &= L_{\pi^*} V^* - L_{\pi^*} V + L_{\pi^*} V - LV + L_\pi V - L_\pi V^\pi \\ &\leq \gamma P_{\pi^*}(V^* - V^\pi + V^\pi - V) + \gamma P_\pi (V - V^\pi). \end{aligned}$$

Thus:

$$(I - \gamma P_{\pi^*})(V^* - V^\pi) \leq \gamma (P_{\pi^*} - P_\pi)(V^\pi - V),$$

and from the property $V^\pi - V = (I - \gamma P_\pi)^{-1}(LV - V)$, it comes:

$$\begin{aligned} V^* - V^\pi &\leq \gamma (I - \gamma P_{\pi^*})^{-1} (P_{\pi^*} - P_\pi)(I - \gamma P_\pi)^{-1}(LV - V) \\ &= \left[(I - \gamma P_{\pi^*})^{-1} - (I - \gamma P_\pi)^{-1}\right](LV - V), \end{aligned} \tag{1.21}$$

thus in norm L^∞:

$$\begin{aligned} ||V^* - V^\pi||_\infty &\leq \left[||(I-\gamma P_{\pi^*})^{-1}||_\infty + ||(I-\gamma P_\pi)^{-1}||_\infty\right]||LV - V||_\infty \\ &\leq \frac{2}{1-\gamma}||LV - V||_\infty. \end{aligned}$$

Thus we have a performance guarantee for policies greedy w.r.t. functions V_α efficiently minimizing the norm of the residual. However the minimization problem (1.19) may be hard to solve, even in the case of a linear parametrization, since the operator L is not affine (in opposition to the operator L_π) because of the maximum operator over actions. There does not exist simple methods for finding the global minimum (in opposition to the previous case using the operator L_π and a linear approximation where the norm of the residual to be minimized $\alpha \rightarrow ||L_\pi V_\alpha - V_\alpha||_\mu$ was a quadratic mapping in α). However local optimization techniques exist (such as gradient methods, where the direction opposite to the gradient $\nabla_\alpha ||LV_\alpha - V_\alpha||_\mu^2$ is followed), for example neural networks are commonly used in practice to minimize the L^2-norm of the residual, although there are no global convergence guarantee.

1.5. Towards an analysis of dynamic programming in L^p-norm

We have provided several performance bounds ((1.6) and (1.9) respectively for the AVI and API algorithms) in terms of the approximation errors *in L^∞-norm*. However, as illustrated in paragraph 1.2.1, usual supervised learning algorithms solve an optimization problem using an empirical L^p-norm (with $p = 1$ or 2). Hence, bounds on the approximation errors (such as (1.12) or (1.15)) are of L^p kind whereas those on error propagation in dynamic programming are of L^∞ kind.

The fundamental problem of the analysis of DP with function approximation lies in the different tools that are used to analyze DP and approximation theory:

– usual DP analysis makes use of the norm L^∞, which is very natural since the Bellman operators L and L_π are contraction operators in this norm. The value iteration, policy iteration, and their RL variants are all based on this property;

– function approximation make almost exclusively use of L^p-norms: for example least squares methods, neural networks, Support Vector Machines and other kernel methods, etc.

The different norms explain the difficulty to analyze DP combined with function approximation. For example, if one considers the AVI algorithm defined by (1.3), the Bellman operator L is a contraction in L^∞-norm, the approximation operator $\mathcal{A}$ is a non-expansion in L^2-norm (in the case of an orthogonal projection onto $\mathcal{F}$), but

one cannot say anything about the combined operator $\mathcal{A}L$. The L^∞ analysis of this algorithm, illustrated by the result (1.6), provides a performance bound in terms of the uniform approximation error $||\varepsilon_n||_\infty$, which is very difficult to control, especially in large scale problems. This result is hardly useful in practice. Moreover, most of the approximation operators and supervised learning algorithms solve a minimization problem using an empirical L^1 or L^2 norm, for example (1.4). The uniform error $||\varepsilon_n||_\infty$ is difficult to bound in terms of the error actually minimized by the empirical L^p problem (1.4).

An L^p analysis of the AVI algorithm which would take into account the approximation errors in L^p-norm would enable to evaluate the performance in terms of the empirical errors and a capacity term (such as usual Vapnik-Chervonenkis dimension or covering numbers [POL 84, VAP 98]) of the considered function space $\mathcal{F}$. Indeed, following usual results in statistical learning, the approximation error in L^p-norm (the so-called *generalization error*) may be upper bounded by the empirical error (or *learning error*) actually minimized plus a capacity term of $\mathcal{F}$. In addition, since most of the approximation operators and supervised learning algorithms provide good fits by minimizing a L^p-norm, it appears essential to analyze the performance of dynamic programming using this same norm.

First steps along this direction are reported in [MUN 07, MUN 08] and briefly described in the two next section.

1.5.1. *Intuition of an L^p analysis in dynamic programming*

First let us remind the definition of L^p-norm (for $p \geq 1$) weighted by a distribution μ: $||f||_{p,\mu} \stackrel{\text{def}}{=} \left[\sum_{s\in S} \mu(s)|f(s)|^p\right]^{1/p}$. When $p = 2$, we use the simplified notation $||f||_\mu$.

The underlying intuition of an L^p analysis of DP is simple and comes from componentwise bounds. Indeed, let f and g two positive functions defined over S, such that $f \leq Pg$, with P being a stochastic matrix. Of course, this implies that $||f||_\infty \leq ||g||_\infty$ (since $||P||_\infty = 1$), but in addition, if ν and μ are distributions over S such that $\nu P \leq C\mu$, (one should interpret the notation νP as the matrix product of the row vector ν by the matrix P) for some constant $C \geq 1$, then we may also deduce that $||f||_{p,\nu} \leq C^{1/p}||g||_{p,\mu}$.

Indeed, we have:

$$\begin{aligned} ||f||_{p,\nu}^p = \sum_{s\in S} \nu(s)|f(s)|^p &\leq \sum_{s\in S} \nu(s)\big|\sum_{s'\in S} P(s'|s)g(s')\big|^p \\ &\leq \sum_{s\in S} \nu(s) \sum_{s'\in S} P(s'|s)|g(s')|^p \\ &\leq C\sum_{s'\in S} \mu(s')|g(s')|^p = C||g||_{p,\mu}^p, \end{aligned}$$

where we used Jensen's inequality (i.e. convexity of $x \rightarrow |x|^p$) at the second line.

For instance, the componentwise bound (1.21) enables to deduce the bound (1.20) in terms of the L^∞-norm of the Bellman residual. From this same bound (1.21), we may also deduce a bound in terms of the L^p-norm of the Bellman residual:

$$||V^* - V^\pi||_{p,\nu} \leq \frac{2}{1-\gamma} C(\nu,\mu)^{1/p} ||LV - V||_{p,\mu}, \tag{1.22}$$

where ν and μ are two distributions over S and $C(\nu,\mu)$ is a constant that measures the concentrability (relative to μ) of the discounted future state distribution (given that the initial state is sampled from ν) of the MDP (seer [MUN 07, MUN 08] for a precise definition and the link with Lyapunov exponents in dynamical systems). This bound is tighter that the L^∞ bound since when $p \rightarrow \infty$ we recover the bound (1.20).

Similar results may be derived for the AVI and API algorithms. For illustration, for the AVI algorithm, one may prove the componentwise bound:

$$\begin{aligned} \limsup_{n\rightarrow\infty} V^* - V^{\pi_n} \leq \limsup_{n\rightarrow\infty} &(I - \gamma P_{\pi_n})^{-1} \\ &\Big(\sum_{k=0}^{n-1} \gamma^{n-k}\big[(P_{\pi^*})^{n-k} + P_{\pi_n} P_{\pi_{n-1}} \dots P_{\pi_{k+2}} P_{\pi_{k+1}}\big]|\varepsilon_k|\Big), \end{aligned}$$

with $\varepsilon_k = LV_k - V_{k+1}$ (approximation error at round k). By taking the L^∞-norm, this bound gives (1.6). But one may also deduce the L^p-norm bound:

$$\limsup_{n\rightarrow\infty} ||V^* - V^{\pi_n}||_{p,\nu} \leq \frac{2\gamma}{(1-\gamma)^2} C(\nu,\mu)^{1/p} \limsup_{n\rightarrow\infty} ||\varepsilon_n||_{p,\mu}. \tag{1.23}$$

Similarly for the API algorithm, the componentwise bound (1.11) enables to deduce the L^∞ result (1.9) as previously shown, but also to derive the following L^p bound:

$$\limsup_{n\rightarrow\infty} ||V^* - V^{\pi_n}||_{p,\nu} \leq \frac{2\gamma}{(1-\gamma)^2} C(\nu,\mu)^{1/p} \limsup_{n\rightarrow\infty} ||V_n - V^{\pi_n}||_{p,\mu}. \tag{1.24}$$

This L^p analysis in DP enables to establish a link with statistical learning and deduce PAC bounds (*Probably Approximately Correct*) [VAL 84, BOU 92] for RL algorithm.

1.5.2. *PAC bounds for RL algorithms*

We now provide a PAC bound (detailed in [MUN 08]) for a RL algorithm based on AVI. We consider a large state space, for example continuous. At each iteration, the approximation operator consists in performing an empirical regression based on a finite number N of sampled states, where in each state, the Bellman operator is estimated by using M transition samples obtained from the generative model.

More precisely, we repeat K approximate policy iteration steps (1.3). At round $1 \leq k < K$, $V_k \in \mathcal{F}$ denotes the current value function approximation, and a new approximation V_{k+1} is obtained as follows. We sample N states $\{s_n\}_{1\leq n\leq N} \in S$ independently from a distribution μ over S. For each state s_n and each action $a \in A$, we generate M successor states $\{s'_{n,a,m} \sim p(\cdot|s_n, a)\}_{1\leq m\leq M}$ and we formulate an empirical estimate of the Bellman operator applied to V_k in s_n :

$$v_n \stackrel{\text{def}}{=} \max_{a\in A} \big[r(s_n, a) + \gamma \frac{1}{M} \sum_{m=1}^{M} V_k(s'_{n,a,m})\big].$$

Finally V_{n+1} is defined as the solution to this L^p fitting problem:

$$V_{n+1} \stackrel{\text{def}}{=} \arg\min_{f\in\mathcal{F}} \frac{1}{N} \sum_{n=1}^{N} \big|f(s_n) - v_n\big|^p.$$

At round K, we write π_K a greedy policy w.r.t. V_K and we wish to evaluate its performance (compared to the optimal performance) in terms of the number of iterations K, the number of sampled states N, the number of successors M, the capacity of the function space $\mathcal{F}$, the smoothness of the MDP (concentrability constant $C(\nu, \mu)$ in the bound (1.23)), and the Bellman residual $d(L\mathcal{F}, \mathcal{F})$ of the function space $\mathcal{F}$. We have the following result.

PROPOSITION 1.7 [MUN 08].– *For any $\delta > 0$, with probability at least $1 - \delta$, we have:*

$$\begin{aligned} ||V^* - V^{\pi_K}||_{p,\nu} \quad \leq \quad & \frac{2\gamma}{(1-\gamma)^2} C(\nu, \mu)^{1/p} \; d(L\mathcal{F}, \mathcal{F}) + O(\gamma^K) \qquad (1.25) \\ & + O\left\{ \left(\frac{V_{\mathcal{F}^+} \log(1/\delta)}{N} \right)^{1/2p} + \left(\frac{\log(1/\delta)}{M} \right)^{1/2} \right\}, \end{aligned}$$

where $d(L\mathcal{F}, \mathcal{F}) \stackrel{\text{def}}{=} \sup_{g \in \mathcal{F}} \inf_{f \in \mathcal{F}} ||Lg - f||_{p,\mu}$ *is the Bellman residual of the space* $\mathcal{F}$*, and* $V_{\mathcal{F}^+}$ *a capacity measure (the pseudo-dimension) of* $\mathcal{F}$ *[HAU 95].*

The four terms of this bound respectively mean:

– the Bellman residual $d(L\mathcal{F}, \mathcal{F})$ generalizes the notion of Bellman residual to a class of functions $\mathcal{F}$. It measures how well the function space $\mathcal{F}$ enables to approximate functions Lg, $g \in \mathcal{F}$ (images by the Bellman operator of functions in $\mathcal{F}$). This term is analogous to the notion of distance between the target function and $\mathcal{F}$ in the case of regression. However here there is no target function to approximate since our goal is to approximate the fixed-point of the Bellman operator with functions in $\mathcal{F}$. When the MDP is smooth (for example if the transition probabilities $p(s'|\cdot, a)$ and the reward function $r(\cdot, a)$ are Lipschitzian), we can show that the term $d(L\mathcal{F}, \mathcal{F})$ decreases when the function space $\mathcal{F}$ increases (since then the Bellman operator possesses a regularizing effect and the space $L\mathcal{F}$ is a subspace of Lipschitz functions with Lipschitz coefficient independent of $\mathcal{F}$, see [MUN 08]);

– the term coming from the finite number K of iteration, tends to 0 exponentially fast;

– two terms of order $O((V_{\mathcal{F}^+}/N)^{1/2p}) + O(1/\sqrt{M})$ bound the estimation error in terms of the number of samples N and M.

This result states that, if we use enough samples (N, M, K), the performance of this algorithm can be arbitrarily close (up to a constant) to the Bellman residual of the space $\mathcal{F}$, which itself can be made arbitrarily small by considering a sufficiently rich function approximation class. This kind of bound is analogous to the bounds obtained in supervised learing [GYÖ 02] and allows to analyze the stability and convergence rate of AVI based on samples, especially:

– it enables to understand the *bias-variance trade-off* in approximate dynamic programming. Indeed, the performance bound (1.25) contains a bias term, the Bellman residual $d(L\mathcal{F}, \mathcal{F})$, that decreases when $\mathcal{F}$ gets richer, and a variance term, due to the capacity $V_{\mathcal{F}^+}$ of $\mathcal{F}$, that increases with the richness of $\mathcal{F}$, but which may be reduced by using a larger number of samples N (in order to avoid overfitting);

– it enables to understand the well-known counter-examples (of divergence of the algorithm) mentioned in previous works (in particular, the Bellman residual $d(L\mathcal{F}, \mathcal{F})$ of the function spaces used in [BAI 95, TSI 96a] is infinite) and to be able to predict the behavior of this kind of algorithms in terms of the characteristics of the MDP, the capacity and richness of the function space $\mathcal{F}$ and the number of samples.

1.6. Conclusions

The results mentioned in the previous paragraph are a direct consequence of statistical learning results combined with the L^p-norm analysis of DP briefly introduced.

Many extensions are possible, such as RL methods based on API (like the LSPI of [LAG 03]) even when samples are obtained by the observation of a unique trajectory [ANT 08]. In parallel to these theoretical works, we should mention the important diversity and quantity of works about the use of approximate representations of the value function for the purpose of decision making. Kernel methods [SCH 01] have been applied to DP and RL [ORM 02, RAS 04], as well as decision trees [WAN 99, ERN 05], neural networks [BER 96, COU 02], Bayesian approaches [DEA 98], factored representations [GUE 01, KOL 00, DEG 06] (see Chapter **??**) only to cite a few.

This chapter focused on approximate representation of the *value function*, following the dynamic programming principle initiated by Bellman [BEL 57]. Let us finally mention that there exists a completely different approach, somehow dual to this one, which consists in searching directly an approximation of the optimal policy by considering a class of parameterized policies. Those methods, now called *direct policy search* originate from Pontryagin's principle [PON 62] which sets necessary conditions for optimality by considering sensitivity analysis of the performance measure with respect to the policy parameters. This approach is the object of another chapter.

Bibliography

[ANT 08] ANTOS A., SZEPESVÁRI C., MUNOS R., "Learning Near-Optimal Policies with Bellman-Residual Minimization Based Fitted Policy Iteration and a Single Sample Path", *Machine Learning Journal*, 2008, à paraître.

[ATK 97] ATKESON C. G., MOORE A. W., SCHAAL S. A., "Locally Weighted Learning", *AI Review*, vol. 11, 1997.

[BAI 95] BAIRD L. C., "Residual Algorithms: Reinforcement Learning with Function Approximation", *Proceedings of the 12th International Conference in Machine Learning (ICML'95)*, San Francisco, CA, Morgan Kaufman Publishers, 1995.

[BEL 57] BELLMAN R. E., *Dynamic Programming*, Princeton University Press, Princeton, N.J., 1957.

[BEL 59] BELLMAN R., DREYFUS S., "Functional Approximation and Dynamic Programming", *Math. Tables and other Aids Comp.*, vol. 13, p. 247–251, 1959.

[BER 96] BERTSEKAS D., TSITSIKLIS J., *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA, 1996.

[BOU 92] BOUCHERON S., *Théorie de l'Apprentissage: de l'approche formelle aux enjeux cognitifs*, Hermès, Paris, 1992.

[BOY 99] BOYAN J., "Least-Squares Temporal Difference Learning", *Proceedings of the 16th International Conference on Machine Learning (ICML'99)*, p. 49–56, 1999.

[BRA 96] BRADTKE S., BARTO A., "Linear Least-Squares Algorithms for Temporal Difference Learning", *Journal of Machine Learning Research*, vol. 22, p. 33–57, 1996.

[COU 02] COULOM R., Reinforcement Learning using Neural Networks, with Applications to Motor Control, PhD thesis, Institut National Polytechnique de Grenoble, 2002.

[CRI 96] CRITES R., BARTO A., "Improving Elevator Performance using Reinforcement Learning", *Advances in Neural Information Processing Systems 8 (NIPS'95)*, 1996.

[DAV 97] DAVIES G., MALLAT S., AVELLANEDA M., "Adaptive Greedy Approximations", *Journal of Constructive Approximation*, vol. 13, p. 57–98, 1997.

[DEA 98] DEARDEN R., FRIEDMAN N., RUSSELL S., "Bayesian Q-learning", *Proceedings of the National Conference on Artificial Intelligence (AAAI'98)*, 1998.

[DEG 06] DEGRIS T., SIGAUD O., WUILLEMIN P.-H., "Learning the Structure of Factored Markov Decision Processes in Reinforcement Learning Problems", *Proceedings of the International Conference on Machine Learning (ICML'06)*, 2006.

[DEV 98] DEVORE R., "Nonlinear Approximation", *Acta Numerica*, vol. 7, p. 51–150, 1998.

[ERN 05] ERNST D., GEURTS P., WEHENKEL L., "Tree-Based Batch Mode Reinforcement Learning", *Journal of Machine Learning Research*, vol. 6, p. 503–556, 2005.

[GOR 95] GORDON G., "Stable Function Approximation in Dynamic Programming", *Proceedings of the 12th International Conference on Machine Learning (ICML'95)*, San Francisco, CA, Morgan Kaufmann, p. 261–268, 1995.

[GOS 04] GOSAVI A., "A Reinforcement Learning Algorithm Based on Policy Iteration for Average Reward: Empirical Results with Yield Management and Convergence Analysis", *Machine Learning*, vol. 55, p. 5–29, 2004.

[GUE 01] GUESTRIN C., KOLLER D., PARR R., "Max-norm Projections for Factored MDPs", *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI'01)*, p. 673–680, 2001.

[GYÖ 02] GYÖRFI L., KOHLER M., KRZYŻAK A., WALK H., *A Distribution-Free Theory of Nonparametric Regression*, Springer-Verlag, 2002.

[HAS 01] HASTIE T., TIBSHIRANI R., FRIEDMAN J., *The Elements of Statistical Learning*, Springer Series in Statistics, 2001.

[HAU 95] HAUSSLER D., "Sphere Packing Numbers for Subsets of the Boolean n-Cube with Bounded Vapnik-Chervonenkis Dimension", *Journal of Combinatorial Theory Series A*, vol. 69, p. 217–232, 1995.

[JUD 98] JUDD K., *Numerical Methods in Economics*, MIT Press, Cambridge, MA, 1998.

[KAK 03] KAKADE S., On the Sample Complexity of Reinforcement Learning, PhD thesis, Gatsby Computationel Neuroscience Unit, University College London, 2003.

[KOL 00] KOLLER D., PARR R., "Policy Iteration for Factored MDPs", *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI'00)*, p. 326–334, 2000.

[KUS 97] KUSHNER H., YIN G., *Stochastic Approximation Algorithms and Applications*, Springer-Verlag, New York, 1997.

[LAG 03] LAGOUDAKIS M., PARR R., "Least-Squares Policy Iteration", *Journal of Machine Learning Research*, vol. 4, p. 1107–1149, 2003.

[MAH 97] MAHADEVAN S., MARCHALLECK N., DAS T., GOSAVI A., "Self-Improving Factory Simulation using Continuous-Time Average-Reward Reinforcement Learning", *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*, 1997.

[MAL 97] MALLAT S., *A Wavelet Tour of Signal Processing*, Academic Press, Londres, Royaume-Uni, 1997.

[MUN 03] MUNOS R., "Error Bounds for Approximate Policy Iteration", *Proceedings of the 19th International Conference on Machine Learning (ICML'03)*, 2003.

[MUN 06] MUNOS R., "Geometric Variance Reduction in Markov chains. Application to Value Function and Gradient Estimation", *Journal of Machine Learning Research*, vol. 7, p. 413–427, 2006.

[MUN 07] MUNOS R., "Performance Bounds in Lp norms for Approximate Value Iteration", *SIAM Journal on Control and Optimization*, vol. 46, 2007.

[MUN 08] MUNOS R., SZEPESVÁRI C., "Finite Time Bounds for Sampling Based Fitted Value Iteration", *Journal of Machine Learning Research*, 2008, à paraître.

[ORM 02] ORMONEIT D., SEN S., "Kernel-Based Reinforcement Learning", *Machine Learning*, vol. 49, p. 161–178, 2002.

[POL 84] POLLARD D., *Convergence of Stochastic Processes*, Springer Verlag, New York, 1984.

[PON 62] PONTRYAGIN L., BOLTYANSKII V., GAMKRILEDZE R., MISCHENKO E., *The Mathematical Theory of Optimal Processes*, Interscience, New York, 1962.

[PUT 94] PUTERMAN M., *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Inc., New York, Etats-Unis, 1994.

[RAS 04] RASMUSSEN C., KUSS M., "Gaussian Processes in Reinforcement Learning", *Advances in Neural Information Processing Systems 16 (NIPS'03)*, Cambridge, MA, MIT Press, p. 751–759, 2004.

[REE 77] REETZ D., "Approximate Solutions of a Discounted Markovian Decision Problem", *Bonner Mathematischer Schriften*, vol. 98: Dynamische Optimierungen, p. 77–92, 1977.

[RUS 96] RUST J., "Numerical Dynamic Programming in Economics", AMMAN H., KENDRICK D., RUST J., Eds., *Handbook of Computational Economics*, Elsevier, North Holland, 1996.

[SAM 59] SAMUEL A., "Some Studies in Machine Learning using the Game of Checkers", *IBM Journal of Research Development*, vol. 3, num. 3, p. 210–229, 1959.

[SAM 67] SAMUEL A., "Some Studies in Machine Learning using the Game of Checkers, II – Recent Progress", *IBM Journal on Research and Development*, vol. 11, num. 6, p. 601–617, 1967.

[SCH 01] SCHOLKOPF B., SMOLA A. J., *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*, MIT Press, Cambridge, MA, 2001.

[SCH 03] SCHOKNECHT R., "Optimality of Reinforcement Learning Algorithms with Linear Function Approximation", *Advances in Neural Information Processing Systems 15 (NIPS'02)*, 2003.

[SIN 97] SINGH S., BERTSEKAS D., "Reinforcement Learning for Dynamic Channel Allocation in Cellular Telephone Systems", *Advances in Neural Information Processing Systems 9 (NIPS'96)*, 1997.

[SUT 88] SUTTON R., "Learning to Predict by the Method of Temporal Differences", *Machine Learning*, vol. 3, num. 1, p. 9–44, 1988.

[SUT 98] SUTTON R. S., BARTO A. G., *Reinforcement Learning: An Introduction*, Bradford Book, MIT Press, Cambridge, MA, 1998.

[TES 95] TESAURO G., "Temporal Difference Learning and TD-Gammon", *Communication of the ACM*, vol. 38, p. 58–68, 1995.

[TSI 96a] TSITSIKLIS J. N., VAN ROY B., "Feature-Based Methods for Large Scale Dynamic Programming", *Machine Learning*, vol. 22, p. 59–94, 1996.

[TSI 96b] TSITSIKLIS J., ROY B. V., An Analysis of Temporal Difference Learning with Function Approximation, Report num. LIDS-P-2322, MIT, 1996.

[VAL 84] VALIANT L. G., "A Theory of the Learnable", *Communications of the ACM*, vol. 27, num. 11, p. 1134–1142, November 1984.

[VAP 97] VAPNIK V., GOLOWICH S. E., SMOLA A., "Support Vector Method for Function Approximation, Regression Estimation and Signal Processing", *Advances in Neural Information Processing Systems 9 (NIPS'96)*, p. 281–287, 1997.

[VAP 98] VAPNIK V., *Statistical Learning Theory*, John Wiley & Sons, New York, 1998.

[WAN 99] WANG X., DIETTERICH T., "Efficient Value Function Approximation Using Regression Trees", *Proceedings of the IJCAI Workshop on Statistical Machine Learning for Large-Scale Optimization*, 1999.

[WAT 89] WATKINS C., Learning from Delayed Rewards, PhD thesis, Cambridge University, Cambridge, Royaume-Uni, 1989.

[WIL 93] WILLIAMS R. J., BAIRD III L. C., Tight Performance Bounds on Greedy Policies Based on Imperfect Value Functions, Report num. NU-CCS-93-14, Northeastern University, College of Computer Science, Boston, MA, November 1993.

[ZHA 95] ZHANG W., DIETTERICH T., "A Reinforcement Learning Approach to Job-Shop Scheduling", *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, 1995.

Index