HAL open science

Recursive Least-Squares Learning with Eligibility Traces

Bruno Scherrer, Matthieu Geist

To cite this version:

Bruno Scherrer, Matthieu Geist. Recursive Least-Squares Learning with Eligibility Traces. European Wrokshop on Reinforcement Learning (EWRL 11), Sep 2011, Athens, Greece. hal-00644511

HAL Id: hal-00644511

https://inria.hal.science/hal-00644511v1

Submitted on 24 Nov 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Recursive Least-Squares Learning with Eligibility Traces

Bruno Scherrer[1] and Matthieu Geist[2]

[1] INRIA, MAIA Project-Team, Nancy, France
[2] Supélec, IMS Research Group, Metz, France

Abstract. In the framework of Markov Decision Processes, we consider the problem of learning a linear approximation of the value function of some fixed policy from one trajectory possibly generated by some other policy. We describe a systematic approach for adapting *on-policy* learning least squares algorithms of the literature (LSTD [5], LSPE [15], FPKF [7] and GPTD [8]/KTD [10]) to *off-policy* learning *with eligibility traces*. This leads to two known algorithms, LSTD(λ)/LSPE(λ) [21] and suggests new extensions of FPKF and GPTD/KTD. We describe their recursive implementation, discuss their convergence properties, and illustrate their behavior experimentally. Overall, our study suggests that the state-of-art LSTD(λ) [21] remains the best least-squares algorithm.

1 Introduction

We consider the problem of learning a linear approximation of the value function of some fixed policy in a Markov Decision Process (MDP) framework, in the most general situation where learning must be done from a single trajectory possibly generated by some other policy, *a.k.a. off-policy* learning. Given samples, well-known methods for estimating a value function are temporal difference (TD) learning and Monte Carlo [19]. TD learning with eligibility traces [19], known as TD(λ), provide a nice bridge between both approaches, and by controlling the bias/variance trade-off [12], their use can significantly speed up learning. When the value function is approximated through a linear architecture, the depth λ of the eligibility traces is also known to control the quality of approximation [20]. Overall, the use of these traces often plays an important practical role.

In the *on-policy* case (where the policy to evaluate is the same as the one that generated data), there has been a significant amount of research on linear Least-Squares (LS) approaches, which are more sample-efficient than TD/Monte-Carlo. Notable such works include LSTD(λ) [5], LSPE(λ) [15], FPKF [7] and GPTD [8]/KTD [10]. Works on off-policy linear learning are sparser: [16] proposed a variation of TD(λ) that could combine off-policy learning with linear approximation and eligibility traces. Recently, [21] proposed and analysed off-policy versions of LSTD(λ) and LSPE(λ). The first motivation of this article is to argue that it is conceptually simple to extend *all* the LS algorithms we have just mentionned so that they can be applied to the off-policy setting *and* use eligibility traces. If this allows to rederive the off-policy versions of LSTD(λ) and LSPE(λ) [21], it also leads to new candidate algorithms, for which we will derive recursive

formulations. The second motivation of this work is to describe the subtle differences between these intimately-related algorithms on the analytical side, and to provide some comparative insights on their empirical behavior (a topic that has to our knowledge not been considered in the literature, even in the particular *on-policy* and *no-trace* situation).

The rest of the paper is organized as follows. Sec. 2 introduces the background of Markov Decision Processes and describes the state-of-the-art algorithms for on-policy learning with recursive LS methods. Sec. 3 shows how to adapt these methods so that they can both deal with the off-policy case and use eligibility traces. The resulting algorithms are formalized, the formula for their recursive implementation is derived, and we discuss their convergence properties. Sec. 4 illustrates empirically the behavior of these algorithms and Sec. 5 concludes.

2 Background and state-of-the-art on-policy algorithms

A Markov Decision Process (MDP) is a tuple $\{S, A, P, R, \gamma\}$ in which S is a finite state space identified with $\{1, 2, \ldots, N\}$, A a finite action space, $P \in \mathcal{P}(S)^{S\times A}$ the set of transition probabilities, $R \in \mathbb{R}^{S\times A}$ the reward function and γ the discount factor. A mapping $\pi \in \mathcal{P}(A)^S$ is called a policy. For any policy π, let P^π be the corresponding stochastic transition matrix, and R^π the vector of mean reward when following π, *i.e.* of components $E_{a|\pi,s}[R(s,a)]$. The value $V^\pi(s)$ of state s for a policy π is the expected discounted cumulative reward starting in state s and then following the policy π: $V^\pi(s) = E_\pi[\sum_{i=0}^\infty \gamma^i r_i | s_0 = s]$ where E_π denotes the expectation induced by policy π. The value function satisfies the (linear) Bellman equation: $\forall s, V^\pi(s) = E_{s',a|s,\pi}[R(s,a) + \gamma V^\pi(s')]$. It can be rewritten as the fixed-point of the Bellman evaluation operator: $V^\pi = TV^\pi$ where for all V, $\;TV = R^\pi + \gamma P^\pi V$.

In this article, we are interested in learning an approximation of this value function V^π under some constraints. First, we assume our approximation to be linearly parameterized: $\hat{V}_\theta(s) = \theta^T\phi(s)$ with $\theta \in \mathbb{R}^p$ being the parameter vector and $\phi(s)$ the feature vector. Also, we want to estimate the value function V^π (or equivalently associated parameters) from a single finite trajectory generated using a possibly different behaviorial policy π_0. Let μ_0 be the stationary distribution of the stochastic matrix $P_0 = P^{\pi_0}$ of the *behavior policy* π_0 (we assume it exists and is unique). Let D_0 be the diagonal matrix of which the elements are $(\mu_0(i))_{1\le i\le N}$. Let Φ be the matrix of feature vectors: $\Phi = [\phi(1)\ldots\phi(N)]^T$. The projection Π_0 onto the hypothesis space spanned by Φ with respect to the μ_0-quadratic norm, which will be central for the understanding of the algorithms, has the following closed-form: $\Pi_0 = \Phi(\Phi^T D_0 \Phi)^{-1}\Phi^T D_0$.

In the rest of this section, we review existing on-policy least-squares based temporal difference learning algorithms. In this case, the behavior and target policies are the same so we omit the subscript 0 for the policy (π) and the projection (Π). We assume that a trajectory $(s_1, a_1, r_1, s_2, \ldots, s_j, a_j, r_j, s_{j+1}, \ldots s_{i+1})$ sampled according to the policy π is available. Let us introduce the sampled Bellman operator $\hat{T}_j$, defined as: $\hat{T}_j : V \in \mathbb{R}^S \to \hat{T}_j V = r_j + \gamma V(s_{j+1}) \in \mathbb{R}$ so that

$\hat{T}_j V$ is an unbiased estimate of $TV(s_j)$. If values were observable, estimating the projected parameter vector θ would reduce to project the value function onto the hypothesis space using the empirical projection operator. This would be the classical least-squares approach. Since values are not observed — only transitions (rewards and next states) are —, we will rely on *temporal differences* (terms of the form $\hat{T}_j V - V(s_j)$) to estimate the value function.

The Least-Squares Temporal Differences (LSTD) algorithm of [6] aims at finding the fixed point of the operator being the composition of the projection onto the hypothesis space and of the Bellman operator. Otherwise speaking, it searches for the fixed point $\hat{V}_\theta = \Pi T \hat{V}_\theta$, Π being the just introduced projection operator. Using the available trajectory, LSTD solves the following fixed-point problem: $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} (\hat{T}_j \hat{V}_{\theta_i} - \hat{V}_\omega(s_j))^2$. The Least-Squares Policy Evaluation (LSPE) algorithm of [15] searches for the same fixed point, but in an iterative way instead of directly (informally, $\hat{V}_{\theta_i} \simeq \Pi T \hat{V}_{\theta_{i-1}}$). The corresponding optimization problem is: $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} (\hat{T}_j \hat{V}_{\theta_{i-1}} - \hat{V}_\omega(s_j))^2$. The Fixed-Point Kalman Filter (FPKF) algorithm of [7] is a least-squares variation of the classical temporal difference learning algorithm [19]. Value function approximation is treated as a supervised learning problem, and unobserved values are bootstrapped: the unobserved value $V^\pi(s_j)$ is replaced by the estimate $\hat{T}_j \hat{V}_{\theta_{j-1}}$. This is equivalent to solving the following optimization problem: $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} (\hat{T}_j \hat{V}_{\theta_{j-1}} - \hat{V}_\omega(s_j))^2$. Finally, the Bellman Residual Minimization (BRM) algorithm aims at minimizing the distance between the value function and its image through the Bellman operator, $\|V - TV\|^2$. When the sampled operator is used, this leads to biased estimates (*e.g.* see [1]). The corresponding optimization problem is: $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} (\hat{T}_j \hat{V}_\omega - \hat{V}_\omega(s_j))^2$. This cost function has originally been proposed by [2] who minimized it using a stochastic gradient approach. It has been considered by [14] with a least-squares approach, however with a double sampling scheme to remove the bias. The parametric Gaussian Process Temporal Differences (GPTD) algorithm of [8] and the linear Kalman Temporal Differences (KTD) algorithm of [10] can be shown to minimize this cost using a least-squares approach (so with bias).

All these algorithms can be summarized as follows:

$$\theta_i = \operatorname*{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} \left(\hat{T}_j \hat{V}_\xi - \hat{V}_\omega(s_j) \right)^2 . \tag{1}$$

One of the presented approach is obtained by instantiating $\xi = \theta_i$, θ_{i-1}, θ_{j-1} or ω and solving the corresponding optimization problem. If more algorithms can be summarized under this generic equation (see [11]), the current paper will restrict its focus on linear least-squares based approaches.

3 Extension to eligibility traces and off-policy learning

This section contains the core of our contribution: we are going to describe a systematic approach in order to adapt the previously mentionned algorithms so that they can deal with eligibility traces and off-policy learning. The actual

formalization of the algorithms, along with the derivation of their recursive implementation, will then follow.

Let $0 \leq \lambda \leq 1$ be the eligibility factor. Using eligibility traces amounts to looking for the fixed point of the following variation of the Bellman operator [4]: $\forall V \in \mathbb{R}^S, \quad T^\lambda V = (1-\lambda)\sum_{i=0}^{\infty} \lambda^i T^{i+1} V$, that makes a geometric average with parameter λ of the powers of the original Bellman operator T. Clearly, any fixed point of T is a fixed point of T^λ and vice-versa. An equivalent *temporal difference* based definition of T^λ is (see *e.g.* [15]): $\forall s$, $T^\lambda V(s) = V(s) + E_\pi[\sum_{j=i}^{\infty}(\gamma\lambda)^{j-i}\Big(r_j + \gamma V(s_{j+1}) - V(s_j)\Big)\Big| s_i = s]$.

As learning is done over a finite trajectory, it is natural to introduce the following truncated operator, which considers samples until time n: $\forall s$, $T_n^\lambda V(s) = V(s) + E_\pi[\sum_{j=i}^{n}(\gamma\lambda)^{j-i}\Big(r_j + \gamma V(s_{j+1}) - V(s_j)\Big)\Big| s_i = s]$. To use it practically, we still need to remove the dependency to the model (*i.e.* the expectation) and to take into account the fact that we want to consider off-policy learning. Assume from now on that we have a trajectory $(s_1, a_1, r_1, s_2, \ldots, s_n, a_n, r_n, s_{n+1})$ sampled according to the behaviour policy π_0. As behaviorial and target policies are different, estimates of T_n^λ need to be corrected through importance sampling [17]. For all s, a, let us introduce the following weight: $\rho(s,a) = \frac{\pi(a|s)}{\pi_0(a|s)}$. In our trajectory context, write $\rho_i^j = \prod_{k=i}^{j} \rho_k$ with $\rho_j = \rho(s_j, a_j)$. Now, consider the off-policy, sampled and truncated $\hat{T}_{i,n}^\lambda : \mathbb{R}^S \to \mathbb{R}$ operator as: $\hat{T}_{i,n}^\lambda V = V(s_i) + \sum_{j=i}^{n}(\gamma\lambda)^{j-i}\Big(\rho_i^j \hat{T}_j V - \rho_i^{j-1} V(s_j)\Big)$. It can be seen that $\hat{T}_{i,n}^\lambda V$ is an unbiased estimate of $T_n^\lambda V(s_i)$ (see [16,21] for details).

Replacing $\hat{T}_j$ by $\hat{T}_{j,i}^\lambda$ in the optimization problem of Eq. (1), is a generic way to extend any parametric value function approximators to the *off-policy* setting and the use of *eligibility traces*: $\theta_i = \operatorname{argmin}_{\omega\in\mathbb{R}^p} \sum_{j=1}^{i}(\hat{T}_{j,i}^\lambda \hat{V}_\xi - \hat{V}_\omega(s_j))^2$. In the rest of this section, by instantiating ξ to θ_i, θ_{i-1}, θ_{j-1} or ω, we derive the already existing algorithms off-policy LSTD(λ)/LSPE(λ) [21], and we extend two existing algorithms to eligibility traces and to off-policy learning, that we will naturally call FPKF(λ) and BRM(λ). With $\lambda = 0$, this exactly corresponds to the algorithms we described in the previous section. When $\lambda = 1$, it can be seen that $\hat{T}_{i,n}^\lambda V = \hat{T}_{i,n}^1 V = \sum_{j=i}^{n} \gamma^{j-i}\rho_i^j r_j + \gamma^{n-i+1}\rho_i^n V(s_{n+1})$; thus, if $\gamma^{n-i+1}\rho_i^n$ tends to 0 when n tends to infinity[3] so that the influence of ξ in the definition of $\hat{T}_{i,n}^\lambda V_\xi$ vanishes, all algorithms should asymptotically behave the same.

Recall that a linear parameterization is chosen here, $\hat{V}_\xi(s_i) = \xi^T\phi(s_i)$. We adopt the following notations: $\phi_i = \phi(s_i)$, $\Delta\phi_i = \phi_i - \gamma\rho_i\phi_{i+1}$ and $\tilde{\rho}_j^{k-1} = (\gamma\lambda)^{k-j}\rho_j^{k-1}$. The generic cost function to be solved is therefore:

$$\theta_i = \operatorname*{argmin}_{\omega\in\mathbb{R}^p} \sum_{j=1}^{i}(\phi_j^T\xi + \sum_{k=j}^{i}\tilde{\rho}_j^{k-1}(\rho_k r_k - \Delta\phi_k^T\xi) - \phi_j^T\omega)^2. \tag{2}$$

[3] This is not always the case, see [21] and the discussion in Sec. 3.4.

3.1 Off-policy LSTD(λ)

The off-policy LSTD(λ) algorithm corresponds to instantiating Problem (2) with $\xi = \theta_i$. This can be solved by zeroing the gradient respectively to ω: $0 = \sum_{j=1}^{i}(\sum_{k=1}^{j} \phi_k \tilde{\rho}_k^{j-1})(\rho_j r_j - \Delta\phi_j^T \theta_i)$. Introducing the (corrected) eligibility vector z_j:

$$z_j = \sum_{k=1}^{j} \phi_k \tilde{\rho}_k^{j-1} = \sum_{k=1}^{j} \phi_k (\gamma\lambda)^{j-k} \prod_{m=k}^{j-1} \rho_m = \gamma\lambda\rho_{j-1} z_{j-1} + \phi_j, \tag{3}$$

one obtains the following batch estimate:

$$\theta_i = (\sum_{j=1}^{i} z_j \Delta\phi_j^T)^{-1} \sum_{j=1}^{i} z_j \rho_j r_j = (A_i)^{-1} b_i. \tag{4}$$

A recursive implementation of this algorithm (where $M_i = (A_i)^{-1}$ is updated on-the-fly) has been proposed and analyzed recently by [21] and is described in Alg. 1. The author proves that if the *behavior* policy π_0 induces an irreducible Markov chain and chooses with positive probability any action that may be chosen by the *target* policy π, and if the compound (linear) operator $\Pi_0 T^\lambda$ has a unique fixed point[4], then off-policy LSTD(λ) converges to it almost surely. Formally, it converges to the solution θ^* of the so-called *projected fixed-point* equation:

$$V_{\theta^*} = \Pi_0 T^\lambda V_{\theta^*}. \tag{5}$$

Using the expression of the projection Π_0 and the form of the Bellman operator T^λ it can be seen that θ^* satisfies (see [21] for details) $\theta^* = A^{-1}b$ where

$$A = \Phi^T D_0 (I - \gamma P)(I - \lambda\gamma P)^{-1}\Phi \quad \text{and} \quad b = \Phi^T D_0 (I - \lambda\gamma P)^{-1} R. \tag{6}$$

The core of the analysis of [21] consists in showing that $\frac{1}{i}A_i$ and $\frac{1}{i}b_i$ defined in Eq. (4) respectively converge to A and b almost surely. Through Eq. (4), this implies the convergence of θ_i to θ^*.

Algorithm 1: LSTD(λ)

Initialization;
Initialize vector θ_0 and matrix M_0 ;
Set $z_0 = 0$;
for $i = 1, 2, \ldots$ **do**
  Observe ϕ_i, r_i, ϕ_{i+1} ;
  Update traces ;
  $z_i = \gamma\lambda\rho_{i-1} z_{i-1} + \phi_i$;
  Update parameters ;
  $K_i = \frac{M_{i-1} z_i}{1 + \Delta\phi_i^T M_{i-1} z_i}$;
  $\theta_i = \theta_{i-1} + K_i(\rho_i r_i - \Delta\phi_i^T \theta_{i-1})$;
  $M_i = M_{i-1} - K_i (M_{i-1}^T \Delta\phi_i)^T$;

Algorithm 2: LSPE(λ)

Initialization;
Initialize vector θ_0 and matrix N_0 ;
Set $z_0 = 0$, $A_0 = 0$ and $b_0 = 0$;
for $i = 1, 2, \ldots$ **do**
  Observe ϕ_i, r_i, ϕ_{i+1};
  Update traces ;
  $z_i = \gamma\lambda\rho_{i-1} z_{i-1} + \phi_i$;
  Update parameters ;
  $N_i = N_{i-1} - \frac{N_{i-1}\phi_i \phi_i^T N_{i-1}}{1 + \phi_i^T N_{i-1} \phi_i}$;
  $A_i = A_{i-1} + z_i \Delta\phi_i^T$;
  $b_i = b_{i-1} + \rho_i z_i r_i$;
  $\theta_i = \theta_{i-1} + N_i(b_i - A_i \theta_{i-1})$;

3.2 Off-policy LSPE(λ)

The off-policy LSPE(λ) algorithm corresponds to the instantiation $\xi = \theta_{i-1}$ in Problem (2). This can be solved by zeroing the gradient respectively to ω: $\theta_i =$

[4] It is not always the case, see [20] or Sec. 4 for a counter-example.

$\theta_{i-1} + (\sum_{j=1}^{i} \phi_j \phi_j^T)^{-1} \sum_{j=1}^{i} z_j(\rho_j r_j - \Delta\phi_j^T \theta_{i-1})$, where we used the eligibility vector z_j defined Eq. (3). Write

$$N_i = (\sum_{j=1}^{i} \phi_j \phi_j^T)^{-1} = N_{i-1} - \frac{N_{i-1}\phi_i \phi_i^T N_{i-1}}{1 + \phi_i^T N_{i-1} \phi_i} \tag{7}$$

where the second equality follows from the Sherman-Morrison formula. Using A_i and b_i as defined in the LSTD description in Eq. (4), one gets: $\theta_i = \theta_{i-1} + N_i(b_i - A_i\theta_{i-1})$. The overall computation is provided in Alg. 2. This algorithm, (briefly) mentionned by [21], generalizes the LSPE(λ) algorithm of [15] to off-policy learning. With respect to LSTD(λ), which computes $\theta_i = (A_i)^{-1} b_i$ (*cf.* Eq. (4)) at each iteration, LSPE(λ) is fundamentally recursive. Along with the almost sure convergence of $\frac{1}{i}A_i$ and $\frac{1}{i}b_i$ to A and b (defined in Eq. (6)), it can be shown that iN_i converges to $N = (\Phi^T D_0 \Phi)^{-1}$ (see for instance [15]) so that, asymptotically, LSPE(λ) behaves as: $\theta_i = \theta_{i-1} + N(b - A\theta_{i-1}) = Nb + (I - NA)\theta_{i-1}$, which is equivalent to (*e.g.* see [15]):

$$V_{\theta_i} = \Phi\theta_i = \Phi N b + \Phi(I - NA)\theta_{i-1} = \Pi_0 T^\lambda V_{\theta_{i-1}}. \tag{8}$$

The behavior of this sequence depends on the spectral radius of $\Pi_0 T^\lambda$. Thus, the analyses of [21] and [15] (for the convergence of N_i) imply the following convergence result: under the assumptions required for the convergence of off-policy LSTD(λ), and the additional assumption that the operator $\Pi_0 T^\lambda$ has spectral radius smaller than 1 (so that it is contracting), LSPE(λ) also converges almost surely to the fixed point of the compound $\Pi_0 T^\lambda$ operator.

There are two sufficient conditions that can (independently) ensure such a desired contraction property. The first one is when one considers on-policy learning (see *e.g.* [15], where the authors studied the on-policy case and use this property in the proof). When the behavior policy π_0 is different from the target policy π, a sufficient condition for contraction is that λ be close enough to 1; indeed, when λ tends to 1, the spectral radius of T^λ tends to zero and can potentially balance an expansion of the projection Π_0. In the off-policy case, when γ is sufficiently big, a small value of λ can make $\Pi_0 T^\lambda$ expansive (see [20] for an example in the case $\lambda = 0$) and off-policy LSPE(λ) will then diverge. Eventually, Equations (5) and (8) show that when $\lambda = 1$, both LSTD(λ) and LSPE(λ) asymptotically coincide (as $T^1 V$ does not depend on V).

3.3 Off-policy FPKF(λ)

The off-policy FPKF(λ) algorithm corresponds to the instantiation $\xi = \theta_{j-1}$ in Problem (2). This can be solved by zeroing the gradient respectively to ω: $\theta_i = N_i(\sum_{j=1}^{i} \phi_j \phi_j^T \theta_{j-1} + \sum_{j=1}^{i} \sum_{k=1}^{j} \phi_k \tilde{\rho}_k^{j-1} (\rho_j r_j - \Delta\phi_j^T \theta_{k-1}))$ where N_i is the matrix introduced for LSPE(λ) in Eq. (7). With respect to the previously described algorithms, the difficulty here is that on the right side there is a dependence with all the previous terms θ_{k-1} for $1 \leq k \leq i$. Using the symmetry of the dot product $\Delta\phi_j^T \theta_{k-1} = \theta_{k-1}^T \Delta\phi_j$, it is possible to write a recursive algorithm by introducing the trace matrix Z_j that integrates the subsequent values of θ_k as follows: $Z_j = \sum_{k=1}^{j} \tilde{\rho}_k^{j-1} \phi_k \theta_{k-1}^T = Z_{j-1} + \gamma\lambda\rho_{j-1}\phi_j\theta_{j-1}^T$. With this notation we

obtain: $\theta_i = N_i(\sum_{j=1}^{i} \phi_j \phi_j^T \theta_{j-1} + \sum_{j=1}^{i}(z_j \rho_j r_j - Z_j \Delta\phi_j)$. Using Eq. (7) and a few algebraic manipulations, we end up with: $\theta_i = \theta_{i-1} + N_i(z_i \rho_i r_i - Z_i \Delta\phi_i)$. This provides Alg. 3.

It generalizes the FPKF algorithm of [7] that was originally only introduced without traces and in the on-policy case. As LSPE(λ), this algorithm is fundamentally recursive. However, its overall behavior is quite different. As we discussed for LSPE(λ), iN_i asymptotically tends to $N = (\Phi^T D_0 \Phi)^{-1}$ and FPKF(λ) iterates eventually resemble: $\theta_i = \theta_{i-1} + \frac{1}{i} N(z_i \rho_i r_i - Z_i \Delta\phi_i)$. The term in brackets is a random component (that depends on the last transition) and $\frac{1}{i}$ acts as a learning coefficient that asymptotically tends to 0. In other words, FPKF(λ) has a *stochastic approximation* flavour. In particular, one can see FPKF(0) as a stochastic approximation of LSPE(0)[5]. When $\lambda > 0$, the situation is less clear (all the more that, as previously mentionned, we expect LSTD/LSPE/FPKF to asymptotically behave the same when λ tends to 1).

Due to its much more involved form (notably the matrix trace Z_j integrating the values of all the values θ_k from the start), we have not been able to obtain a formal analysis of FPKF(λ), even in the on-policy case. To our knowledge, there is no *full proof of convergence* for stochastic approximation algorithms with eligibility traces in the off-policy case[6], and a related result for FPKF(λ) thus seems difficult. Nevertheless, it is reasonable to conjecture that off-policy FPKF(λ) has the same asymptotic behavior as LSPE(λ).

Algorithm 3: FPKF(λ)

Initialization;
Initialize vector θ_0 and matrix N_0 ;
Set $z_0 = 0$ and $Z_0 = 0$;
for $i = 1, 2, \ldots$ **do**
  Observe ϕ_i, r_i, ϕ_{i+1};
  Update traces ;
  $z_i = \gamma\lambda\rho_{i-1} z_{i-1} + \phi_i$;
  $Z_i = \gamma\lambda\rho_{i-1} Z_{i-1} + \phi_i \theta_{i-1}^T$;
  Update parameters ;
  $N_i = N_{i-1} - \frac{N_{i-1}\phi_i \phi_i^T N_{i-1}}{1+\phi_i^T N_{i-1} \phi_i}$;
  $\theta_i = \theta_{i-1} + N_i(z_i \rho_i r_i - Z_i \Delta\phi_i)$;

Algorithm 4: BRM(λ)

Initialization;
Initialize vector θ_0 and matrix C_0 ;
Set $y_0 = 0$, $\Delta_0 = 0$ and $z_0 = 0$;
for $i = 1, 2, \ldots$ **do**
  Observe ϕ_i, r_i, ϕ_{i+1};
  Pre-update traces ;
  $y_i = (\gamma\lambda\rho_{i-1})^2 y_{i-1} + 1$;
  Compute ;
  $U_i = \left(\sqrt{y_i}\Delta\phi_i + \frac{\gamma\lambda\rho_{i-1}}{\sqrt{y_i}}\Delta_{i-1} \;\; \frac{\gamma\lambda\rho_{i-1}}{\sqrt{y_i}}\Delta_{i-1}\right)$;
  $V_i = \left(\sqrt{y_i}\Delta\phi_i + \frac{\gamma\lambda\rho_{i-1}}{\sqrt{y_i}}\Delta_{i-1} \;\; -\frac{\gamma\lambda\rho_{i-1}}{\sqrt{y_i}}\Delta_{i-1}\right)^T$;
  $W_i = \left(\sqrt{y_i}\rho r_i + \frac{\gamma\lambda\rho_{i-1}}{\sqrt{y_i}} z_{i-1} \;\; -\frac{\gamma\lambda\rho_{i-1}}{\sqrt{y_i}} z_{i-1}\right)^T$;
  Update parameters ;
  $\theta_i = \theta_{i-1} + C_{i-1}U_i(I_2 + V_i C_{i-1} U_i)^{-1}(W_i - V_i \theta_{i-1})$;
  $C_i = C_{i-1} - C_{i-1}U_i(I_2 + V_i C_{i-1} U_i)^{-1} V_i C_{i-1}$;
  Post-update traces ;
  $\Delta_i = (\gamma\lambda\rho_{i-1})\Delta_{i-1} + \Delta\phi_i y_i$;
  $z_i = (\gamma\lambda\rho_{i-1}) z_{i-1} + r_i \rho_i y_i$;

3.4 Off-policy BRM(λ)

The off-policy BRM(λ) algorithm corresponds to the instantiation $\xi = \omega$ in Problem (2): $\theta_i = \text{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i}(z_{j\to i} - \psi_{j\to i}^T \omega)^2$ where $\psi_{j\to i} = \sum_{k=j}^{i} \tilde{\rho}_j^{k-1} \Delta\phi_k$

[5] To see this, one can compare the asymptotic behavior of both algorithms. FPKF(0) does $\theta_i = \theta_{i-1} + \frac{1}{i} N(\rho_i \phi_i r_i - \phi_i \Delta\phi_i^T \theta_{i-1})$. One then notices that $\rho_i \phi_i r_i$ and $\phi_i \Delta\phi_i^T$ are samples of A and b to which A_i and b_i converge through LSPE(0).

[6] An analysis of TD(λ), with a simplifying assumption that forces the algorithm to stay bounded, is given in [21]. An analysis of a related algorithm, GQ(λ), is provided in [13], with an assumption on the second moment of the traces, which does not hold in general (see Propostion 2 in [21]). A full analysis thus remains to be done.

and $z_{j\to i} = \sum_{k=j}^{i} \tilde{\rho}_j^{k-1} \rho_k r_k$. Thus $\theta_i = (\tilde{A}_i)^{-1}\tilde{b}_i$ where $\tilde{A}_i = \sum_{j=1}^{i} \psi_{j\to i}\psi_{j\to i}^T$ and $\tilde{b}_i = \sum_{j=1}^{i} \psi_{j\to i} z_{j\to i}$. The recursive implementation of θ_i is somewhat tedious, but space restriction precludes its description. The resulting algorithm, which is based on 3 traces — 2 reals (y_i and z_i) and 1 vector (Δ_i) — and involves the (straightforward) inversion of a 2×2 matrix, is described in Alg. 4.

GPTD and KTD, which are close to BRM, have also been extended with some trace mechanism; however, GPTD(λ) [8], KTD(λ) [9] and the just described BRM(λ) are different algorithms. Briefly, GPTD(λ) mimics the on-policy LSTD(λ) algorithms and KTD(λ) uses a different Bellman operator[7]. As BRM(λ) builds a linear systems of which it updates the solution recursively, it resembles LSTD(λ). However, the system it builds is different. The following theorem characterizes the behavior of BRM(λ) and its potential limit.

Theorem 1. *Assume that the stochastic matrix P_0 of the* behavior *policy is irreducible and has stationary distribution μ_0. Further assume that*

$$\text{there exists a coefficient } \beta < 1 \text{ such that } \forall (s,a), \quad \lambda\gamma\rho(s,a) \leq \beta, \tag{9}$$

then $\frac{1}{i}\tilde{A}_i$ and $\frac{1}{i}\tilde{b}_i$ respectively converge almost surely to $\tilde{A} = \Phi^T[D - \gamma DP - \gamma P^T D + \gamma^2 D' + S(I - \gamma P) + (I - \gamma P^T)S^T]\Phi$ and $\tilde{b} = \Phi^T[(I - \gamma P^T)Q^T D + S]R^\pi$ where we wrote: $D = \mathrm{diag}((I - (\lambda\gamma)^2\tilde{P}^T)^{-1}\mu_0)$, $D' = \mathrm{diag}(\tilde{P}^T(I - (\lambda\gamma)^2\tilde{P}^T)^{-1}\mu_0)$, $Q = (I - \lambda\gamma P)^{-1}$, $S = \lambda\gamma(DP - \gamma D')Q$, and where $\tilde{P}$ is the matrix of coordinates $\tilde{p}_{ss'} = \sum_a \pi(s,a)\rho(s,a)T(s,a,s')$.

As a consequence the BRM(λ) algorithm converges with probability 1 to $\tilde{A}^{-1}\tilde{b}$. The assumption given by Eq. (9) trivially holds in the on-policy case (in which $\rho(s,a) = 1$ for all (s,a)) and in the off-policy case when $\lambda\gamma$ is small with respect to the mismatch between policies $\rho(s,a)$. The matrix $\tilde{P}$, which is in general not a stochastic matrix, can have a spectral radius bigger than 1; Eq. (9) ensures that $(\lambda\gamma)^2\tilde{P}$ has spectral radius smaller than β so that D and D' are well defined. Finally, note that there is probably no hope to remove the assumption of Eq. (9) since by making $\lambda\gamma$ big enough, one may force the spectral radius of $(\lambda\gamma)^2\tilde{P}$ to be as close to 1 as one may want, which would make $\tilde{A}$ and $\tilde{b}$ diverge.

The proof of this Theorem follows the general lines of that of Proposition 4 in [3]. Due to space constraints, we only provide its sketch: Eq. (9) implies that the traces can be truncated at some depth l, of which the influence on the potential limit of the algorithm vanishes when l tends to ∞. For all l, the l-truncated version of the algorithm can easily be analyzed through the ergodic theorem for Markov chains. Making l tend to ∞ allows to tie the convergence of the original arguments to that of the truncated version. Eventually, the formula for the limit of the truncated algorithm is (tediously) computed and one derives the limit.

The fundamental idea behind the Bellman Residual approach is to address the computation of the fixed point of T^λ differently from the previous methods. Instead of computing the projected fixed point as in Eq. (5), one considers the overdetermined system: $\Phi\theta \simeq T^\lambda\Phi\theta \quad \Leftrightarrow \quad \Phi\theta \simeq (I - \lambda\gamma P)^{-1}(R +$

[7] Actually, the corresponding loss is $(\hat{T}^0_{j,i}\hat{V}(\omega) - \hat{V}_\omega(s_j) + \gamma\lambda(\hat{T}^1_{j+1,i}\hat{V}(\omega) - \hat{V}_\omega(s_{j+1})))^2$. With $\lambda = 0$ it gives $\hat{T}^0_{j,i}$ and with $\lambda = 1$ it provides $\hat{T}^1_{j,i}$

$(1-\lambda)\gamma P\Phi\theta) \quad \Leftrightarrow \quad \Phi\theta \simeq QR + (1-\lambda)\gamma PQ\Phi\theta \quad \Leftrightarrow \quad \Psi\theta \simeq QR$ with $\Psi = \Phi - (1-\lambda)\gamma PQ\Phi$ and solves it in a least-squares sense, that is by computing $\theta^* = \bar{A}^{-1}\bar{b}$ with $\bar{A} = \Psi^T\Psi$ and $\bar{b} = \Psi^T QR$. One of the motivation for this approach is that, contrary to the matrix A of LSTD/LSPE/FPKF, $\bar{A}$ is inversible for all values of λ, and one can always guarantee a finite error bound with respect to the best projection [18]. If the goal of BRM(λ) is to compute $\bar{A}$ and $\bar{b}$ from samples, what it actually computes ($\tilde{A}$ and $\tilde{b}$) will in general be biased because it is based on a single trajectory[8]. Such a bias adds an uncontrolled variance term to $\bar{A}$ and $\bar{b}$ (*e.g.* see [1]) of which an interesting consequence is that $\tilde{A}$ remains non singular[9]. More precisely, there are two sources of bias in the estimation: one results from the non Monte-carlo evaluation (the fact that $\lambda < 1$) and the other from the use of the correlated importance sampling factors (as soon as one considers off-policy learning). The interested reader may check that in the on-policy case, and when λ tends to 1, $\tilde{A}$ and $\tilde{b}$ coincide with $\bar{A}$ and $\bar{b}$. However, in the strictly off-policy case, taking $\lambda = 1$ does not prevent the bias due to the correlated importance sampling factors. If we have argued that LSTD/LSPE/FPKF asymptotically coincide when $\lambda = 1$, we see here that BRM may generally differ in an off-policy situation.

4 Illustration of the algorithms

In this section, we briefly illustrate the behavior of all the algorithms we have described so far. In a first set of experiments, we consider random Markov chains involving 3 states and 2 actions and projections onto random spaces[10] of dimension 2. The discount factor is $\gamma = 0.99$. For each experiment, we have run all algorithms (plus TD(λ) with stepsize $\alpha_t = \frac{1}{t+1}$) 50 times with initial matrix (M_0, N_0, C_0) equal to[11] $100I$, with $\theta_0 = 0$ and during $100,000$ iterations. For each of these 50 runs, the different algorithms share the same samples, that are generated by a random uniform policy π_0 (*i.e.* that chooses each action with probability 0.5). We consider two situations: *on-policy*, where the policy to evaluate is $\pi = \pi_0$, and *off-policy*, where the policy to evaluate is random (*i.e.* , it picks the actions with probabilities p and $1-p$, where p is chosen uniformly at random). In the curves we are about to describe, we display on the abscissa the iteration number and on the ordinate the median value of the distance (quadratic, weighted by the stationary distribution of P) between the computed value $\Phi\theta$ and the real value $V = (I - \gamma P)^{-1}R$ (*i.e.* the lower the better).

[8] It is possible to remove the bias when $\lambda = 0$ by using double samples. However, in the case where $\lambda > 0$, the possibility to remove the bias seems much more difficult.

[9] $\bar{A}$ is by construction positive definite, and $\tilde{A}$ equals $\bar{A}$ plus a positive term (the variance term), and is thus also positive definite.

[10] For each action, rewards are uniform random vectors on $(0,1)^3$, transition matrices are random uniform matrices on $(0,1)^{3\times 3}$ normalized so that the probabilities sum to 1. Random projections are induced by random uniform matrices Φ of size 3×2.

[11] This matrix acts as an L_2 regularization and is used to avoid numerical instabilities at the beginning of the algorithms. The bigger the value, the smaller the influence.

For each of the two situations (*on-* and *off-policy*), we present data in two ways. To appreciate the influence of λ, we display the curves on one graph per algorithm with different values of λ (Fig. 1 and 2). To compare the algorithms for solving the Bellman equation $V = T^\lambda V$, we show on one graph per value of λ the error for the different algorithms (Fig. 3 and 4). **In the on-policy setting,** LSTD and LSPE have similar performance and convergence speed for all values of λ. They tend to converge much faster than FPKF, which is slightly faster than TD. BRM is usually in between LSTD/LSPE and FPKF/TD, though for small values of λ, the bias seems significative. When λ increases, the performance of FPKF and BRM improves. At the limit when $\lambda = 1$, all algorithms (except TD) coincide (confirming the intuition for $\lambda = 1$, the influence of the choice ξ vanishes in Eq. (2)). **In the off-policy setting,** LSTD and LSPE still share the same behavior. The drawbacks of the other algorithms are amplified with respect to the on-line situation. As λ increases, the performance of FPKF catches that of LSTD/LSPE. However, the performance of BRM seems to worsen while λ is increased from 0 to 0.99 and eventually approaches that of the other algorithms when $\lambda = 1$ (though it remains different, *cf.* the discussion in the previous section). **Globally,** the use of eligibility traces allows to significantly improve the performance of FPKF(λ) over FPKF [7] in both on- and off-policy cases, and that of BRM(λ) over BRM/GPTD/KTD of [8,10] in the on-policy case. The performance of BRM(λ) in the off-policy case is a bit disappointing, probably because of its inherent bias, which deserves further investigation. However, LSTD(λ)/LSPE(λ) appear to be in general the best algorithms.

Eventually, we consider 2 experiments involving an MDP and a projection due to [20], in order to illustrate possible numerical issues when solving the projected fixed-point Eq. (5). In the first experiment one sets (λ, γ) such that $\Pi_0 T^\lambda$ is expansive; as expected one sees (Fig. 5) that LSPE and FPKF both diverge. In the latter experiment, one sets (λ, γ) so that the spectral radius of $\Pi_0 T^\lambda$ is 1 (so that A is singular), and in this case LSTD also diverges (Fig. 6). In both situations, BRM is the only one not to diverge[12].

5 Conclusion

We considered LS algorithms for value estimation in an MDP context. Starting from the on-policy case with no trace, we recalled that several algorithms (LSTD, LSPE, FPKF and BRM/GPTD/KTD) optimize similar cost functions. Substituting the original Bellman operator by an operator that deals with traces and off-policy samples leads to the state-of-the-art off-policy trace-based versions of LSTD and LSPE, and suggests natural extensions of FPKF and BRM. We described recursive implementations of these algorithms, discussed their convergence properties, and illustrated their behavior empirically. Overall, our study

[12] Note that this adverserial setting is meant to illustrate the fact that for the problem considered, some values (λ, γ) may be problematic for LSTD/LSPE/FPKF. In practice, λ can be chosen big enough so that these algorithms will be stable.

suggests that even if the use of eligibility traces generally improves the efficiency of the algorithms, LSTD(λ) and LSPE(λ) remain in general better than FPKF(λ) (that is much slower) and BRM(λ) (that may suffer from high bias). Furthermore, since LSPE(λ) requires more conditions for stability, LSTD(λ) probably remains the best choice in practice.

References

1. Antos, A., Szepesvári, C., Munos, R.: Learning Near-optimal Policies with Bellman-residual Minimization based Fitted Policy Iteration and a Single Sample Path. In: COLT (2006)
2. Baird, L.C.: Residual Algorithms: Reinforcement Learning with Function Approximation. In: ICML (1995)
3. Bertsekas, D.P., Yu, H.: Projected Equation Methods for Approximate Solution of Large Linear Systems. J. Comp. and Applied Mathematics 227(1), 27–50 (2009)
4. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific (1996)
5. Boyan, J.A.: Technical Update: Least-Squares Temporal Difference Learning. Machine Learning 49(2-3), 233–246 (1999)
6. Bradtke, S.J., Barto, A.G.: Linear Least-Squares algorithms for temporal difference learning. Machine Learning 22(1-3), 33–57 (1996)
7. Choi, D., Van Roy, B.: A Generalized Kalman Filter for Fixed Point Approximation and Efficient Temporal-Difference Learning. DEDS 16, 207–239 (2006)
8. Engel, Y.: Algorithms and Representations for Reinforcement Learning. Ph.D. thesis, Hebrew University (2005)
9. Geist, M., Pietquin, O.: Eligibility Traces through Colored Noises. In: ICUMT (2010)
10. Geist, M., Pietquin, O.: Kalman Temporal Differences. JAIR 39, 483–532 (2010)
11. Geist, M., Pietquin, O.: Parametric Value Function Approximation: a Unified View. In: ADPRL (2011)
12. Kearns, M., Singh, S.: Bias-Variance Error Bounds for Temporal Difference Updates. In: COLT (2000)
13. Maei, H.R., Sutton, R.S.: GQ(λ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. In: Conference on Artificial General Intelligence (2010)
14. Munos, R.: Error Bounds for Approximate Policy Iteration. In: ICML (2003)
15. Nedić, A., Bertsekas, D.P.: Least Squares Policy Evaluation Algorithms with Linear Function Approximation. DEDS 13, 79–110 (2003)
16. Precup, D., Sutton, R.S., Singh, S.P.: Eligibility Traces for Off-Policy Policy Evaluation. In: ICML (2000)
17. Ripley, B.D.: Stochastic Simulation. Wiley & Sons (1987)
18. Scherrer, B.: Should one compute the Temporal Difference fix point or minimize the Bellman Residual? The unified oblique projection view. In: ICML (2010)
19. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning). MIT Press, 3rd edn. (1998)
20. Tsitsiklis, J., Van Roy, B.: An analysis of temporal-difference learning with function approximation. IEEE Transactions on Automatic Control 42(5), 674–690 (1997)
21. Yu, H.: Convergence of Least-Squares Temporal Difference Methods under General Conditions. In: ICML (2010)

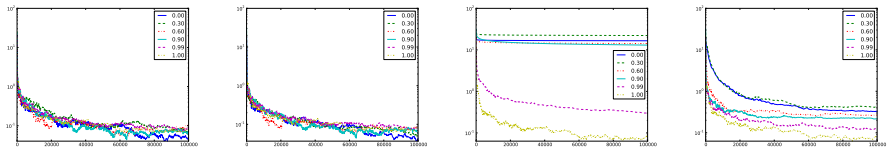

Fig. 1. Influence of λ, *on-policy* (LSTD, LSPE, FPKF and BRM).

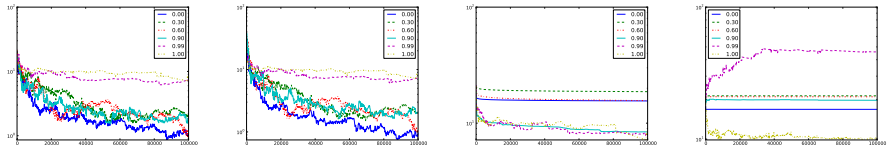

Fig. 2. Influence of λ, *off-policy* (LSTD, LSPE, FPKF and BRM).

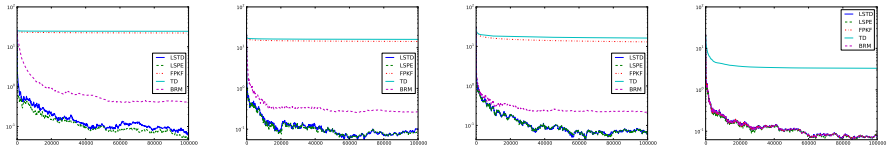

Fig. 3. Comparison of the algorithms, *on-policy* ($\lambda \in \{0.3, 0.6, 0.9, 1\}$).

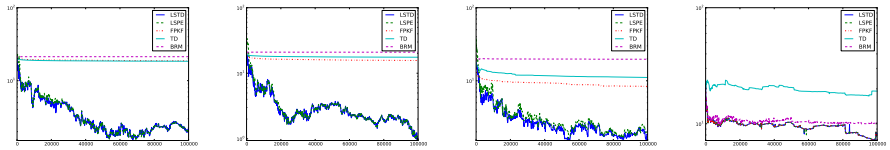

Fig. 4. Comparison of the algorithms, *off-policy* ($\lambda \in \{0.3, 0.6, 0.9, 1\}$).

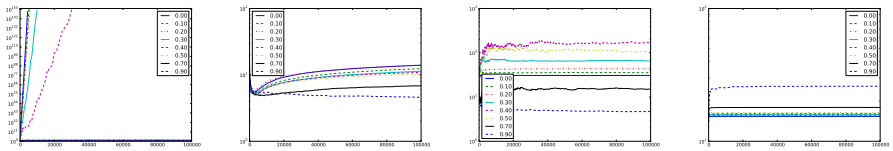

Fig. 5. Pathological situation where LSPE and FPKF diverge, while LSTD converges (LSPE, FPKF, LSTD and BRM).

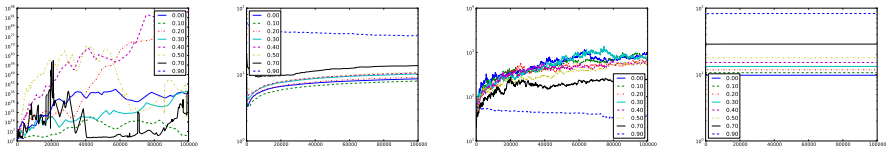

Fig. 6. Pathological situation where LSPE, FPKF and LSTD all diverge (LSPE, FPKF, LSTD and BRM).