l1-penalized projected Bellman residual

Matthieu Geist, Bruno Scherrer

ℓ_1-penalized projected Bellman residual

Matthieu Geist[1] and Bruno Scherrer[2]

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

Abstract. We consider the task of feature selection for value function approximation in reinforcement learning. A promising approach consists in combining the Least-Squares Temporal Difference (LSTD) algorithm with ℓ_1-regularization, which has proven to be effective in the supervised learning community. This has been done recently whit the LARS-TD algorithm, which replaces the projection operator of LSTD with an ℓ_1-penalized projection and solves the corresponding fixed-point problem. However, this approach is not guaranteed to be correct in the general off-policy setting. We take a different route by adding an ℓ_1-penalty term to the projected Bellman residual, which requires weaker assumptions while offering a comparable performance. However, this comes at the cost of a higher computational complexity if only a part of the regularization path is computed. Nevertheless, our approach ends up to a supervised learning problem, which let envision easy extensions to other penalties.

1 Introduction

A core problem of reinforcement learning (RL) [19] is to assess the quality of some control policy (for example within a policy iteration context), quantified by an associated value function. In the less constrained setting (large state space, unknown transition model), there is a need for estimating this function from sampled trajectories. Often, a parametric representation of the value function is adopted, and many algorithms have been proposed to learn the underlying parameters [2, 21]. This implies to choose *a priori* the underlying architecture, such as basis functions for a parametric representation or the neural topology for a multi-layered perceptron. This problem-dependent task is more difficult in RL that in the more classical supervised setting because the value function is never directly observed, but defined as the fixed-point of an associated Bellman operator.

A general direction to alleviate this problem is the study of non-parametric approaches for value function approximation. This implies many different methods, such as feature construction [10, 15] or Kernel-based approaches [7, 22]. Another approach consists in defining beforehand a (very) large number of features and then choosing automatically those which are relevant for the problem at hand. This is generally known as feature selection. In the supervised learning setting, this general idea is notably instantiated by ℓ_1-regularization [24, 5], which has been recently extended to value function approximation using different approaches [13, 12, 11, 16].

In this paper, we propose an alternative ℓ_1-regularization of the Least-Squares Temporal Difference (LSTD) algorithm [4]. One searches for an approximation of the value function V (being a fixed-point of the Bellman operator T) belonging to some (linear) hypothesis space $\mathcal{H}$, onto which one projects any function using the related projection operator Π. LSTD provides $\hat{V} \in \mathcal{H}$, the fixed-point of the composed operator ΠT. The sole generalization of LSTD to ℓ_1-regularization has been proposed in [12] ([11] solves the same problem, [13] regularizes a -biased- Bellman Residual and [16] considers linear programming). They add an ℓ_1-penalty term to the projection operator and solve the consequent fixed-point problem, the corresponding algorithm being called LARS-TD in reference to the homotopy path algorithm LARS (Least Angle Regression) [6] which inspired it. However, their approach does not correspond to any convex optimization problem and is improper if some conditions are not met.

In this paper, we propose to take a different route to combine LSTD with ℓ_1-regularization. Instead of searching for a fixed-point of the Bellman operator combined with the ℓ_1-regularized projection operator, we add an ℓ_1 penalty term to the minimization of a projected Bellman residual, introducing the ℓ_1-PBR (Projected Bellman Residual) algorithm. Compared to [12], the proposed approach corresponds to a convex optimization problem. Consequently, it is correct under much weaker assumptions, at the cost of a generally higher computational cost. Section 2 reviews some useful preliminaries, notably the LSTD and the LARS-TD algorithms. Section 3 presents the proposed approach and discusses some of its properties, in light of the state of the art. Section 4 illustrates our claims and intuitions on simple problems and Section 5 opens perspectives.

2 Preliminaries

A Markovian decision process (MDP) is a tuple $\{S, A, P, R, \gamma\}$ where S is the finite[3] state space, A the finite action space, $P : s, a \in S \times A \to p(.|s, a) \in \mathcal{P}(S)$ the family of Markovian transition probabilities, $R : s \in S \to r = R(s) \in \mathbb{R}$ the bounded reward function and γ the discount factor weighting long-term rewards. According to these definitions, the system stochastically steps from state to state conditionally on the actions the agent performs. Let i be the discrete time step. To each transition (s_i, a_i, s'_i) is associated an immediate reward r_i. The action selection process is driven by a policy $\pi : s \in S \to \pi(s) \in A$. The quality of a policy is quantified by the value function V^π, defined as the expected discounted cumulative reward starting in a state s and then following the policy π: $V^\pi(s) = E[\sum_{i=0}^{\infty} \gamma^i r_i | s_0 = s, \pi]$. Thanks to the Markovian property, the value function of a policy π is the unique fixed-point of the Bellman operator

$$T^\pi : V \in \mathbb{R}^S \to T^\pi V \in \mathbb{R}^S : T^\pi V(s) = E_{s'|s,\pi(s)}[R(s) + \gamma V(s')]. \quad (1)$$

Let $P^\pi = (p(s'|s, \pi(s)))_{1 \leq s,s' \leq |S|}$ be the associated transition matrix, the value function is therefore the solution of the linear system $V^\pi = R + \gamma P^\pi V^\pi$.

[3] The finite state space assumption is made for simplicity, but this work can easily be extended to continuous state spaces.

In the general setting addressed here, two problems arise. First, the model (that is R and P^π) is unknown, and one should estimate the value function from sampled transitions. Second, the state space is too large to allow an exact representation and one has to rely on some approximation scheme. Here, we search for a value function $\hat{V}^\pi$ being a linear combination of p basis functions $\phi_i(s)$ chosen beforehand, the parameter vector being noted θ:

$$\hat{V}^\pi(s) = \sum_{i=1}^{p} \theta_i \phi_i(s) = \theta^T \phi(s), \theta \in \mathbb{R}^p, \phi(s) = \big(\phi_1(s) \ \dots \ \phi_p(s)\big)^T. \quad (2)$$

Let us note $\Phi \in \mathbb{R}^{|S| \times p}$ the feature matrix whose rows contain the feature vectors $\phi(s)^T$ for any state $s \in S$. This defines an hypothesis space $\mathcal{H} = \{\Phi\theta | \theta \in \mathbb{R}^p\}$ into which we should search for a good approximation $\hat{V}^\pi$ of V^π.

2.1 LSTD

The LSTD algorithm [4] minimizes the distance between the value function $\hat{V}$ and the back-projection onto $\mathcal{H}$ of its image under the Bellman operator (this image having no reason to belong to $\mathcal{H}$):

$$\hat{V}^\pi = \underset{V \in \mathcal{H}}{\operatorname{argmin}} \|V - \Pi T^\pi V\|_D^2, \quad \Pi T^\pi V = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \|T^\pi V - h\|_D^2, \quad (3)$$

with $D \in \mathbb{R}^{|S| \times |S|}$ being a diagonal matrix whose components are some state distribution. With a linear parameterization, $\hat{V}^\pi$ is actually the fixed-point of the composed ΠT^π operator: $\hat{V}^\pi = \Pi T^\pi \hat{V}^\pi$.

However, the model is unknown (and hence T^π), so LSTD actually solves a samples-based fixed-point problem. Assume that we have a set of n transitions $\{(s_i, a_i, r_i, s'_i)\}_{1 \leq i \leq n}$, not necessarily sampled along one trajectory. Let us introduce the sampled based feature and reward matrices:

$$\tilde{\Phi} = \begin{pmatrix} \phi(s_1)^T \\ \vdots \\ \phi(s_n)^T \end{pmatrix} \in \mathbb{R}^{n \times p}, \quad \tilde{\Phi}' = \begin{pmatrix} \phi(s'_1)^T \\ \vdots \\ \phi(s'_n)^T \end{pmatrix} \in \mathbb{R}^{n \times p}, \quad \tilde{R} = \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix} \in \mathbb{R}^n. \quad (4)$$

The LSTD estimate θ^* is thus given by the following nested optimization problems, the first equation depicting the projection and the second the minimization:

$$\begin{cases} \omega_\theta = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \|\tilde{R} + \gamma \tilde{\Phi}'\theta - \tilde{\Phi}\omega\|^2 \\ \theta^* = \operatorname{argmin}_{\theta \in \mathbb{R}^p} \|\tilde{\Phi}\omega_\theta - \tilde{\Phi}\theta\|^2 \end{cases}. \quad (5)$$

The parameterization being linear, this can be easily solved:

$$\theta^* = \tilde{A}^{-1}\tilde{b}, \quad \tilde{A} = \tilde{\Phi}^T \Delta\tilde{\Phi}, \quad \Delta\tilde{\Phi} = \tilde{\Phi} - \gamma\tilde{\Phi}', \quad \tilde{b} = \tilde{\Phi}^T \tilde{R}. \quad (6)$$

Asymptotically, $\Phi\theta^*$ converges to the fixed-point of ΠT^π.

2.2 LARS-TD

In supervised learning, ℓ_1-regularization [24, 5] consists in adding a penalty on the minimized objective function, this penalty being proportional to the ℓ_1-norm of the parameter vector, $\|\theta\|_1 = \sum_{j=1}^p |\theta_j|$. As ℓ_2-regularization, this prevents

overfitting, but the use of the ℓ_1-norm also produces sparse solutions (components of θ being exactly set to zero). Therefore, adding such a penalty is often understood as performing feature selection.

In order to combine LSTD with ℓ_1-regularization, It has been proposed to add an ℓ_1-penalty term to the projection equation [12]. This corresponds to the following optimization problem:

$$\begin{cases} \omega_\theta = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \|\tilde{R} + \gamma \tilde{\Phi}' \theta - \tilde{\Phi} \omega\|^2 + \lambda \|\omega\|_1 \\ \theta^* = \operatorname{argmin}_{\theta \in \mathbb{R}^p} \|\tilde{\Phi} \omega_\theta - \tilde{\Phi} \theta\|^2 \end{cases}, \tag{7}$$

where λ is the regularization parameter. Equivalently, this can be seen as solving the following fixed-point problem:

$$\theta^* = \operatorname*{argmin}_{\theta \in \mathbb{R}^p} \|\tilde{R} + \gamma \tilde{\Phi}' \theta^* - \tilde{\Phi} \theta\|^2 + \lambda \|\theta\|_1. \tag{8}$$

This optimization problem cannot be formulated as a convex one [12]. However, based on subdifferential calculus, equivalent optimality conditions can be derived, which can be used to provide a LARS-like homotopy path algorithm. Actually, under some conditions, the estimate θ^* as a function of λ is piecewise linear. In the supervised setting, LARS [6] is an algorithm which compute efficiently the whole regularization path, that is the solutions $\theta^*(\lambda)$ for any $\lambda \geq 0$, by identifying the breaking points of the regularization path. In [12], a similar algorithm solving optimization problem (7) is proposed. A finite sample analysis of LARS-TD has been provided recently [8], in the on-policy case.

For LARS-TD to be correct (that is admitting a continuous and unique regularization path [11]), it is sufficient for $\tilde{A}$ to be a P-matrix[4] [12]. In the on-policy case (that is the state distribution is the MDP stationary distribution induced by the policy π), given enough samples, $\tilde{A}$ is positive definite and hence a P-matrix. However, if the state distribution is different from the stationary distribution, which is typically the case in an off-policy setting, no such guarantee can be given. This is a potential weakness of this approach, as policy evaluation often occurs in some off-policy policy iteration context.

3 ℓ_1-penalized projected Bellman residual

Starting from the same classical formulation of LSTD in Equation (5), we take a different route to add regularization, to be compared to the LARS-TD approach depicted in Equation (7):

$$\begin{cases} \omega_\theta = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \|\tilde{R} + \gamma \tilde{\Phi}' \theta - \tilde{\Phi} \omega\|^2 \\ \theta^* = \operatorname{argmin}_{\theta \in \mathbb{R}^p} \|\tilde{\Phi} \omega_\theta - \tilde{\Phi} \theta\|^2 + \lambda \|\theta\|_1 \end{cases}. \tag{9}$$

Instead of adding the ℓ_1-penalty term to the projection equation, as in LARS-TD, we propose to add it to the minimization equation.

In order to investigate the conceptual difference between both approaches, we will now consider their asymptotic behavior. Let Π_λ be the ℓ_1-penalized projection operator: $\Pi_\lambda V = \operatorname{argmin}_{h \in \mathcal{H}} \|V - h\|_D^2 + \lambda \|h\|_1$. The LARS-TD algorithm

[4] A square matrix is a P-matrix if all its principle minors are strictly positive. This is a strict superset of the class of (non-symmetric) definite positive matrices.

searches for a fixed-point of the composed operator $\Pi_\lambda T^\pi$, which appears clearly from its analysis [8]: $\hat{V} = \Pi_\lambda T^\pi \hat{V}$. On the other hand, the approach proposed in this paper adds an ℓ_1-penalty term to the minimization of the (classical) projection of the Bellman residual:

$$\hat{V} = \underset{\Phi\theta \in \mathcal{H}}{\operatorname{argmin}} \|\Pi(\Phi\theta - T^\pi(\Phi\theta))\|_D^2 + \lambda\|\theta\|_1 \tag{10}$$

Because of this, we name it ℓ_1-PBR (Projected Bellman Residual). Both approaches make sense, both with their own pros and cons. Before discussing this and studying the properties of ℓ_1-PBR, we provide a practical algorithm.

3.1 Practical algorithm

The proposed ℓ_1-PBR turns out to be much simpler to solve than LARS-TD. We assume that the matrix $\tilde{\Phi}^T\tilde{\Phi} \in \mathbb{R}^{p\times p}$ is invertible[5]. The projection equation can be solved analytically:

$$\Phi\omega_\theta = \hat{\Pi}(\tilde{R} + \gamma\tilde{\Phi}'\theta), \quad \hat{\Pi} = \tilde{\Phi}(\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T \tag{11}$$

Therefore, optimization problem (9) can be written in the following equivalent form:

$$\theta^* = \underset{\theta\in\mathbb{R}^p}{\operatorname{argmin}} \|\tilde{y} - \tilde{\Psi}\theta\|^2 + \lambda\|\theta\|_1, \quad \tilde{y} = \hat{\Pi}\tilde{R}, \quad \tilde{\Psi} = \tilde{\Phi} - \gamma\hat{\Pi}\tilde{\Phi}' \tag{12}$$

The interesting thing here is that $\tilde{y}$ and $\tilde{\Psi}$ being defined, we obtain a purely supervised learning problem. First, this allows solving it by applying directly the LARS algorithm, or any other approache such as LCP. Second, this would still hold for any other penalization term. Therefore, extensions of the proposed ℓ_1-PBR to adaptive lasso [25] or elastic-net [26] (among others) are straightforward.

The pseudo-code of ℓ_1-PBR is provided in Alg. 1 ($\tilde{\Psi}_\mathcal{A}$ denotes the columns of $\tilde{\Psi}$ corresponding to the indices in the current active set $\mathcal{A}$, and similarly for a vector). As it is a direct use of LARS, we refer the reader to [6] for full details and provide only the general idea. It can be shown (see Sec. 3.2) that the regularization path is piecewise linear. Otherwise speaking, there exists $\{\lambda_0 = 0, \ldots, \lambda_k\}$ such that for any $\lambda \in]\lambda_i, \lambda_{i+1}[$, $\nabla_\lambda\theta^*$ is a constant vector. Let us call this regularization values the breaking points. As for λ large enough, the trivial solution is $\theta^* = 0$, $\lambda_k < \infty$. The LARS algorithm starts by identifying the breaking point λ_k at which $\theta^* = 0$. Then, it sequentially discovers other breaking points until $\lambda_0 = 0$ or until a specified regularization factor is reached (or possibly until a specified number of features have been added to the active set). For each interval $]\lambda_i, \lambda_{i+1}[$, it computes the constant vector $\nabla_\lambda\theta^* = \Delta\theta$, which allows inferring the solution for any point of the path. These breaking points correspond to activation or deactivation of a basis function (that is a parameter becomes nonzero or zero), this aspect being inherent to the fact that the ℓ_1-norm is not differentiable at zero. Computing the candidate breaking points in the algorithm corresponds to detecting when one of the equivalent optimality conditions is violated.

[5] If this is not the case, it is sufficient to add an arbitrary small amount of ℓ_2-regularization to the projection equation.

Algorithm 1: ℓ_1-PBR

Initialization;
Compute $\hat{\Pi} = \tilde{\Phi}(\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T$, $\tilde{y} = \hat{\Pi}\tilde{R}$, $\quad \tilde{\Psi} = \tilde{\Phi} - \gamma\hat{\Pi}\tilde{\Phi}'$;
Set $\theta = \mathbf{0}$ and initialize the correlation vector $c = \tilde{\Psi}^T\tilde{R}$;
Let $\{\bar{\lambda}, i\} = \max_j(|c_j|)$ and initialize the active set $\mathcal{A} = \{i\}$;
while $\bar{\lambda} > \lambda$ **do**
  Compute update direction: $\Delta\theta_{\mathcal{A}} = (\tilde{\Psi}_{\mathcal{A}}^T\tilde{\Psi}_{\mathcal{A}})^{-1}\operatorname{sgn}(c_{\mathcal{A}})$;
  Find step size to add element: $\{\delta_1, i_1\} = \min^+{}_{j\notin\mathcal{A}}(\frac{c_j-\bar{\lambda}}{d_j-1}, \frac{c_j+\bar{\lambda}}{d_j+1})$ with $d = \tilde{\Psi}^T\tilde{\Psi}_{\mathcal{A}}\Delta\theta_{\mathcal{A}}$;
  Find step size to remove element: $\{\delta_2, i_2\} = \min^+{}_{j\in\mathcal{A}}(-\frac{\theta_j}{\Delta\theta_j})$;
  Compute $\delta = \min(\delta_1, \delta_2, \bar{\lambda} - \lambda)$. Update $\theta \leftarrow \theta_{\mathcal{A}} - \delta_{\mathcal{A}}\Delta\theta_{\mathcal{A}}$, $\bar{\lambda} \leftarrow \bar{\lambda} - \delta$ and $c \leftarrow c - \delta d$;
  Add i_1 to $(\delta_1 < \delta_2)$ or remove i_2 from $(\delta_2 < \delta_1)$ the active set $\mathcal{A}$;

Compared to LARS-TD, the disadvantage of ℓ_1-PBR is its higher time and memory complexities. Both algorithms share the same complexities per iteration of the LARS-like homotopy path algorithm. However, ℓ_1-PBR additionally requires projecting the reward and some features onto the hypothesis space $\mathcal{H}$ (that is computing $\tilde{y}$ and $\tilde{\Psi}$ in Equation (12)). This adds the complexity of a full least-squares. Computing the full regularization path with LARS-TD presents also the same complexity as a full least-squares; in this case, both approaches requires the same order of computations and memory. However, if only a part of the regularization path is computed, the complexity of LARS-TD decreases to solving a least-squares with as many parameters as there are active features for the smallest value of λ, whereas the complexity of ℓ_1-PBR keeps the same order.

3.2 Correctness of ℓ_1-PBR

The LARS-TD algorithm requires the matrix $\tilde{A} = \tilde{\Phi}^T\Delta\tilde{\Phi}$ to be a P-matrix in order to find a solution. The next straightforward property shows that ℓ_1-PBR requires much weaker conditions.

Theorem 1. *If $\tilde{A} = \tilde{\Phi}^T\Delta\tilde{\Phi}$ and $\tilde{M} = \tilde{\Phi}^T\tilde{\Phi}$ are invertible, then the ℓ_1-PBR algorithm finds a unique solution for any $\lambda \geq 0$, and the associated regularization path is piecewise linear.*

Proof. As Equation (12) defines a supervised optimization problem, checking that $\tilde{\Psi}^T\tilde{\Psi}$ is symmetric positive definite is sufficient for the result (this matrix being symmetric positive by construction). The matrix $\tilde{M}$ being invertible, the empirical projection operator $\hat{\Pi}$ is well defined. Moreover, $\hat{\Pi}$ being a projection, $\hat{\Pi}\tilde{\Phi} = \tilde{\Phi}$, $\hat{\Pi}^T = \hat{\Pi}$ and $\hat{\Pi}^2 = \hat{\Pi}$. Therefore, we have that $\tilde{\Psi}^T\tilde{\Psi} = \tilde{A}^T\tilde{M}^{-1}\tilde{A}$. The considered optimization problem is then strictly convex, hence the existence

and uniqueness of its solution. Piecewise linearity of the regularization path is a straightforward consequence of Prop. 1 of [17].

These conditions are much weaker than the ones of LARS-TD. Moreover, even if they are not satisfied (for example if there are more basis functions than samples, that is $p > n$), one can add a small ℓ_2-penalty term to each equation of (9). This would correspond to replacing the projection by an ℓ_2-penalized projection and the ℓ_1-penalty term by an elastic net one. In [12], it is argued that the matrix $\tilde{A}$ can be ensured to be a P-matrix by adding an ℓ_2-penalty term. This should be true only for a high enough associated regularization parameter, whereas any strictly positive parameter is sufficient in our case. Moreover, for LARS-TD, a badly chosen regularization parameter can lead to instabilities (if an eigenvalue of $\tilde{A}$ is too close to zero).

3.3 Discussion

Using an ℓ_1-penalty term for value function approximation has been considered before in [16] in an approximate linear programming context or in [13] where it is used to minimize a Bellman residual[6]. Our work is closer to LARS-TD [12], briefly presented in Section 2.2, and both approaches are compared next. In [11], it is proposed to solve the same fixed-point optimization problem (7) using a Linear Complementary Problem (LCP) approach instead of a LARS-like algorithm. This has several advantages. Notably, it allows using warm starts (initializing the algorithm with starting points from similar problems), which is useful in a policy iteration context. Notice that this LCP approach can be easily adapted to the proposed ℓ_1-PBR algorithm. Recall also that ℓ_1-PBR can be easily adapted to many penalty terms, as it ends up to a supervised learning problem (see Equation (12)). This is less clear for other approaches.

As explained in Section 2.2, LARS-TD requires $\tilde{A}$ to be a P-matrix in order to be correct (the LCP approach requires the same assumption [11]). This condition is satisfied in the on-policy case, given enough transitions. However, there is no such result in the more general off-policy case, which is of particular interest in a policy iteration context. An advantage of ℓ_1-PBR is that it relies on much weaker conditions, as shown in Proposition 1. Therefore, the proposed approach can be used safely in an off-policy context, which is a clear advantage over LARS-TD. Regarding this point, an interesting analogy for the difference between LARS-TD and ℓ_1-BRM is the difference between the classical TD algorithm [19] and the recent TDC (TD with gradient Correction) [20].

TD is an online stochastic gradient descent algorithm which aims at solving the fixed-point problem $\hat{V} = \Pi T^\pi \hat{V}$ using a bootstrapping approach. One of its weaknesses is that it can be unstable in an off-policy setting. The TDC algorithm has been introduced in [20] in order to alleviate this problem. TDC is also an online stochastic gradient descent algorithm, but it minimizes the

[6] As noted in [11], they claim to adapt LSTD while actually regularizing a Bellman residual minimization, which is well known to produce biased estimates [1].

projected Bellman residual $\|\hat{V} - \Pi T^\pi \hat{V}\|^2$. When both approaches converge, they do so to the same solution. However, contrary to TD, TDC is provably convergent in an off-policy context, the required conditions being similar to those of Proposition 1 (A and M should not be singular). This is exactly the difference between LARS-TD and ℓ_1-PBR: LARS-TD penalizes the projection defining the fixed-point problem of interest, whereas ℓ_1-PBR penalizes the projected Bellman residual.

However, this weaker usability conditions have a counterpart: ℓ_1-PBR has generally higher time and memory complexities than LARS-TD, as explained in Section 3.1. Nevertheless, off-policy learning usually suggests batch learning, so this increased cost might not be such a problem. Also, if ℓ_1-PBR is used in a policy iteration context, the computation of the projection can be factorized over iterations, as it does not depend on transiting states.

The proposed algorithm can also be linked to an (unbiased) ℓ_1-penalized Bellman residual minimization (BRM). Let us consider the asymptotic form of the optimization problem solved by ℓ_1-PBR, depicted in Equation (10). Using the Pythagorean theorem, it can be rewritten as follows:

$$\theta^* = \operatorname*{argmin}_{\theta \in \mathbb{R}^p} \|\Phi\theta - T^\pi \Phi\theta\|_D^2 - \|\Pi T^\pi \Phi\theta - T^\pi \Phi\theta\|_D^2 + \lambda\|\theta\|_1 \tag{13}$$

Assume that the hypothesis space is rich enough to represent $T^\pi V$, for any $V \in \mathcal{H}$. Then, the term $\|\Pi T^\pi \Phi\theta - T^\pi \Phi\theta\|_D^2$ vanishes and ℓ_1-PBR ends up to add an ℓ_1-penalty term to the Bellman Residual Minimization (BRM) cost function. Surely, in practice this term will not vanish, because of the finite number of samples and of a not rich enough hypothesis space. Nevertheless, we expect it to be small given a large enough $\mathcal{H}$, so ℓ_1-PBR should behave similarly to some hypothetic ℓ_1-BRM algorithm, unbiased because computed using the transition model. BRM is known to be more stable and more predictable than LSTD [14, 18]. However, it generally leads to a biased estimate, unless a double sampling approach is used or the model is known [1], a problem we do not have. We illustrate this intuition in the next section.

4 Illustration

Two simple problems are considered here. The first one is a two-state MDP which shows that ℓ_1-PBR finds solutions when LARS-TD does not and illustrates the improved stability of the proposed approach. The second problem is the Boyan chain [3]. It is used to illustrate our intuition about the relation between ℓ_1-PBR and ℓ_1-BRM, depicted in Section 3.3, and to compare prediction abilities of LARS-TD and of our approach.

4.1 The two-state MDP

The first problem is a simple two-state MDP [2, 12, 18]. The transition matrix is $P = \left(\begin{smallmatrix} 0 & 1 \\ 0 & 1 \end{smallmatrix}\right)$ and the reward vector $R = \begin{pmatrix} 0 & -1 \end{pmatrix}^T$. The optimal value function is

therefore $v^* = \frac{-1}{1-\gamma}\begin{pmatrix}\gamma & 1\end{pmatrix}^T$. Let us consider the one-feature linear approximation $\Phi = \begin{pmatrix}1 & 2\end{pmatrix}^T$ with uniform distribution $D = \begin{pmatrix}\frac{1}{2} & 0 \\ 0 & \frac{1}{2}\end{pmatrix}$. Let γ be the discount factor. Consequently, we have that $A = \Phi^T D(\Phi - \gamma P\Phi) = \frac{5}{2}(1 - \frac{6}{5}\gamma)$ and $b = \Phi^T DR = -1$. The value $\gamma = \frac{5}{6}$ is singular. Below it, A is a P-matrix, but above it is not the case (obviously, in this case $A < 0$).

The solutions to problems (7) and (9), respectively noted $\theta_\lambda^{\text{lars}}$ and $\theta_\lambda^{\text{pbr}}$, can be easily computed analytically in this case. If $\gamma < \frac{5}{6}$, both approaches have an unique regularization path. For LARS-TD, we have $\theta_\lambda^{\text{lars}} = 0$ if $\lambda > 1$ and $\theta_\lambda^{\text{lars}} = -\frac{2}{5(1-\frac{6}{5}\gamma)}(1-\lambda)$ else. For ℓ_1-PBR, we have $\theta_\lambda^{\text{pbr}} = 0$ if $\lambda > |1 - \frac{6}{5}\gamma|$ and $\theta_\lambda^{\text{pbr}} = -\frac{2}{5(1-\frac{6}{5}\gamma)}(1 - \frac{\lambda \operatorname{sgn}(\theta_\lambda^{\text{pbr}})}{1-\frac{6}{5}\gamma})$ else. If $\gamma > \frac{5}{6}$, the ℓ_1-PBR solution still holds, but LARS-TD no longer admits a unique solution, A being not a P-matrix. The solutions of LARS-TD are the following: $\theta_\lambda^{\text{lars}} = 0$ if $\lambda > 1$, $\theta_\lambda^{\text{lars}} = -\frac{2}{5(1-\frac{6}{5}\gamma)}(1-\lambda)$ if $\lambda > 1$, and $\theta_\lambda^{\text{lars}} = -\frac{2}{5(1-\frac{6}{5}\gamma)}(1+\lambda)$ for any $\lambda \geq 0$.

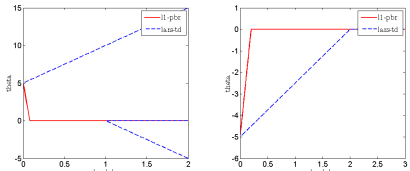

Fig. 1. Two-state MDP, regularization paths (left panel: off-policy; right panel: on-policy).

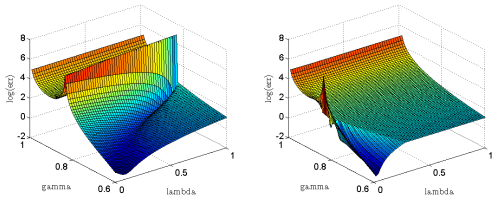

Fig. 2. Two-state MDP, error surface (left: LARS-TD; right: ℓ_1-PBR).

Figure 1 shows the regularization paths of LARS-TD and ℓ_1-PBR for $\gamma = 0.9$, in the just depicted off-policy case (left panel) as well as in the on-policy case (right panel). For $\lambda = 0$, both approaches coincide, as they provide the LSTD solution. In the off-policy case, LARS-TD has up to three solutions, and the regularization path is not continuous, which was already noticed in [12]. ℓ_1-PBR has not this problem, this illustrates Proposition 1. In the on-policy case, both approaches work, providing different regularization paths.

Figure 2 shows the error (defined here as $\|v^* - \Phi\theta_\lambda\|_D$) as a function of the discount factor γ and of the regularization factor λ, in the off-policy case. We restrict ourselves to $\lambda \in [0, 1]$, such that LARS-TD has a unique solution for any value of λ. The left panel show the error surface of LARS-TD and the right panel the one of ℓ_1-PBR. For γ small enough, A is a P-matrix and the error is usually slightly lower for LARS-TD than for ℓ_1-PBR. However, when γ is close to the singular value, LARS-TD presents a high error for any value of λ whereas ℓ_1-PBR is more stable (high errors only occurs for small values of λ, close to the singular discount factor). Consequently, on this (somehow pathological) simple example, LARS-TD may have a slightly better prediction ability, but at the cost of a larger zone of instabilities.

4.2 The Boyan chain

The Boyan chain is a 13-state Markov chain where state s^0 is an absorbing state, s^1 transits to s^0 with probability 1 and a reward of -2, and s^i transits to either s^{i-1} or s^{i-2}, $2 \leq i \leq 12$, each with probability 0.5 and reward -3. The feature vectors $\phi(s)$ for states s^{12}, s^8, s^4 and s^0 are respectively $[1,0,0,0]^T$, $[0,1,0,0]^T$, $[0,0,1,0]^T$ and $[0,0,0,1]^T$, and they are obtained by linear interpolation for other states. The optimal value function is exactly linear in these features, and the corresponding optimal parameter vector is $\theta^* = [-24,-16,-8,0]^T$. In addition to these 4 relevant features, we added 9 irrelevant features, containing Gaussian random noise for each state (adding more than 9 features would prevent computing the whole regularization path: if $p > |S|$, A and M are necessarily singular).

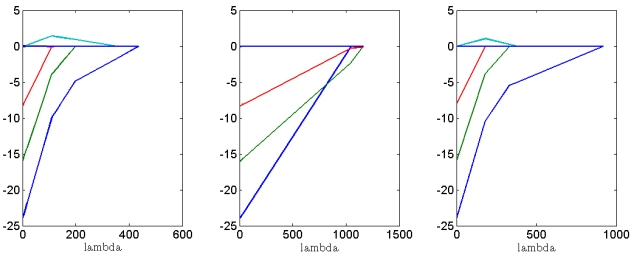

Fig. 3. Boyan chain, regularization paths (left: ℓ_1-PBR; middle: LARS-TD; right: ℓ_1-BRM).

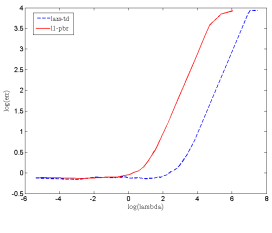

Fig. 4. Boyan chain, error curves.

First, Figure 4 illustrates the regularization paths for ℓ_1-PBR (left panel), LARS-TD (middle panel) and ℓ_1-BRM (Bellman Residual Minimization, right panel). This last algorithm minimizes the classical (unbiased) BRM cost function penalized with an ℓ_1-norm. More formally, the considered optimization problem is:

$$\theta^* = \underset{\theta \in \mathbb{R}^p}{\operatorname{argmin}} \|\tilde{R} - (\tilde{\Phi} - \gamma\tilde{\Xi})\theta\|^2 + \lambda\|\theta\|_1 \text{ with } \tilde{\Xi} = \left[E_{s'|s_1}[\phi(s')] \ \ldots \ E_{s'|s_n}[\phi(s')]\right]^T. \tag{14}$$

This can be easily solved using the LARS algorithm, by treating it as a supervised learning approach with observations $\tilde{R}$ and predictors $\tilde{\Phi} - \gamma\tilde{\Xi}$.

The regularization paths have been computed using samples collected from 50 trajectories. One can see on Figure 4 that ℓ_1-PBR and LARS-TD have quite different regularization paths, whereas those of ℓ_1-PBR and ℓ_1-BRM are really close. Irrelevant features have small weights along the whole regularization path for all approaches (most of them cannot even be seen on the figure), and all algorithms converge to the LSTD solution. This tends to confirm the intuition discussed in Section 3.3: with a rich enough hypothesis space, ℓ_1-PBR is close to unbiased ℓ_1-BRM (which is not practical in general, as it requires knowing the transition model for computing the $\tilde{\Xi}$ features).

As regularization paths are quite different for LARS-TD and ℓ_1-PBR, it is interesting to compare their prediction abilities. Figure 4 shows the prediction error (more formally $\|v^* - \Phi\theta\|$) as a function of the regularization parameter for both algorithms (notice the logarithmic scale for both axes). This figure is an average of 1000 independent learning runs using samples generated from 50 trajectories. Error curves are similar, whereas not for the same range of regularization values. Therefore, both approaches offer similar performance on this example.

5 Conclusion

In this paper, we have proposed and alternative to LARS-TD, which searches for the fixed-point of the ℓ_1-projection composed with the Bellman operator. Instead, we add an ℓ_1-penalty term to the minimization of the projected Bellman residual. Notice that the same algorithm has been proposed in parallel and independently in [9], which provides a complementary point of view. Our approach is somehow reminiscent of how TDC [20] has been introduced in order to alleviate the inherent drawback of the classical TD. The proposed approach is correct under weaker conditions and can therefore be used safely in an off-policy setting, contrary to LARS-TD (even if this seems not to be a problem according to the few experiments published [12, 11]). Preliminary experiments suggest that both approaches offer comparable performance. As it ends up to a supervised learning problem, ℓ_1-PBR can also be easily extended to other penalty terms. However, this comes at the cost of a higher computational cost. Even if not described in the paper, extension of ℓ_1-PBR to the state-action value function approximation is straightforward. In the future, we plan to perform a deeper theoretical study of the proposed approach (the analysis of [7] in the case of ℓ_2-penalized LSTD can be a lead) and to apply it to control problems (notably Tetris [23] should be an interesting application, as features are quite interpretable).

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. Machine Learning 71(1), 89–129 (2008)
2. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific
3. Boyan, J.A.: Technical Update: Least-Squares Temporal Difference Learning. Machine Learning 49(2-3), 233–246 (1999)
4. Bradtke, S.J., Barto, A.G.: Linear Least-Squares algorithms for temporal difference learning. Machine Learning 22(1-3), 33–57 (1996)
5. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic Decomposition by Basis Pursuit. SIAM Journal on Scientific Computing 20, 33–61 (1999)
6. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least Angle Regression. Annals of Statistics 32(2), 407–499 (2004)
7. Farahmand, A., Ghavamzadeh, M., Szepesvári, C., Mannor, S.: Regularized policy iteration. In: 22nd Annual Conference on Neural Information Processing Systems (NIPS 21). Vancouver, Canada (2008)

8. Ghavamzadeh, M., Lazaric, A., Munos, R., Hoffman, M.: Finite-Sample Analysis of Lasso-TD. In: International Conference on Machine Learning (2011)
9. Hoffman, M.W., Lazaric, A., Ghavamzadeh, M., , Munos, R.: Regularized least squares temporal difference learning with nested ℓ_2 and ℓ_1 penalization. In: European Workshop on Reinforcement Learning (2011)
10. Johns, J., Mahadevan, S.: Constructing basis functions from directed graphs for value function approximation. In: Proceedings of the 24th international conference on Machine learning. pp. 385–392. ICML '07, ACM, New York, NY, USA (2007)
11. Johns, J., Painter-Wakefield, C., Parr, R.: Linear Complementarity for Regularized Policy Evaluation and Improvement. In: Lafferty, J., Williams, C.K.I., Shawe-Taylor, J., Zemel, R., Culotta, A. (eds.) NIPS 23. pp. 1009–1017 (2010)
12. Kolter, J.Z., Ng, A.Y.: Regularization and Feature Selection in Least-Squares Temporal Difference Learning. In: proceedings of the 26th International Conference on Machine Learning (ICML 2009). Montreal Canada (2009)
13. Loth, M., Davy, M., Preux, P.: Sparse Temporal Difference Learning using LASSO. In: IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning. Hawaï, USA (2007)
14. Munos, R.: Error bounds for approximate policy iteration. In: International Conference on Machine Learning (2003)
15. Parr, R., Li, L., Taylor, G., Painter-Wakefield, C., Littman, M.L.: An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In: Proceedings of the 25th international conference on Machine learning. pp. 752–759. ICML '08, ACM, New York, NY, USA (2008)
16. Petrik, M., Taylor, G., Parr, R., Zilberstein, S.: Feature Selection Using Regularization in Approximate Linear Programs for Markov Decision Processes. In: Proceedings of ICML (2010)
17. Rosset, S., Zhu, J.: Piecewise linear regularized solution paths. The Annals of Statistics 35(3), 1012–1030 (2007)
18. Scherrer, B.: Should one compute the Temporal Difference fix point or minimize the Bellman Residual? The unified oblique projection view. In: 27th International Conference on Machine Learning - ICML 2010. Haïfa Israël (2010)
19. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning). The MIT Press (1998)
20. Sutton, R.S., Maei, H.R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., Wiewiora, E.: Fast gradient-descent methods for temporal-difference learning with linear function approximation. In: Proceedings of ICML. pp. 993–1000. ACM, New York, NY, USA (2009)
21. Szepesvári, C.: Algorithms for Reinforcement Learning. Morgan and Claypool (2010)
22. Taylor, G., Parr, R.: Kernelized value function approximation for reinforcement learning. In: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 1017–1024. ICML '09, ACM, New York, NY, USA (2009)
23. Thiery, C., Scherrer, B.: Building Controllers for Tetris. International Computer Games Association Journal 32, 3–11 (2009)
24. Tibshirani, R.: Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society. Series B (Methodological) 58(1), 267–288 (1996)
25. Zou, H.: The adaptive lasso and its oracle properties. Journal of the American Statistical Association 101(476), 1418–1429 (2006)
26. Zou, H., Zhang, H.H.: On the adaptive elastic-net with a diverging number of parameters. The Annals of Statistics 37(4), 1733–1751 (2009)