Mach Learn
DOI 10.1007/s10994-009-5128-4

Hybrid least-squares algorithms for approximate policy evaluation

Jeff Johns · Marek Petrik · Sridhar Mahadevan

Received: 12 June 2009 / Revised: 12 June 2009 / Accepted: 16 June 2009

Abstract The goal of approximate policy evaluation is to "best" represent a target value function according to a specific criterion. Different algorithms offer different choices of the optimization criterion. Two popular least-squares algorithms for performing this task are the *Bellman residual* method, which minimizes the Bellman residual, and the *fixed point* method, which minimizes the *projection* of the Bellman residual. When used within policy iteration, the fixed point algorithm tends to ultimately find better performing policies whereas the Bellman residual algorithm exhibits more stable behavior between rounds of policy iteration. We propose two *hybrid least-squares algorithms* to try to combine the advantages of these algorithms. We provide an analytical and geometric interpretation of hybrid algorithms and demonstrate their utility on a simple problem. Experimental results on both small and large domains suggest hybrid algorithms may find solutions that lead to better policies when performing policy iteration.

Keywords Reinforcement learning · Markov decision processes

1 Introduction

Solving Markov decision processes (MDPs) (Puterman 1994) with large or infinite state spaces requires some form of function approximation. Algorithms that use value functions, such as approximate value iteration and approximate policy iteration, encounter the problem of how best to represent a target function. Given a policy π, the *approximate policy evaluation* problem is to compute an approximate value function $\hat{V}$ that represents that policy's

Editors: Aleksander Kołcz, Dunja Mladenić, Wray Buntine, Marko Grobelnik, and John Shawe-Taylor.

J. Johns (✉) · M. Petrik · S. Mahadevan
Department of Computer Science, University of Massachusetts Amherst, Amherst, MA 01003, USA
e-mail: johns@cs.umass.edu

M. Petrik
e-mail: petrik@cs.umass.edu

S. Mahadevan
e-mail: mahadeva@cs.umass.edu

Published online: 23 July 2009

value function V^π. Temporal difference (TD) methods (Sutton 1988) and Bellman residual methods (Schweitzer and Seidmann 1985) offer different solutions to this problem. These methods, which behave very differently in practice, are similar in that they both stem from functions of the Bellman equation. Baird (1995) proposed a combination of the two methods using a single parameter that at one extreme defaults to the temporal difference method and at the other extreme defaults to the Bellman residual method. Intermediate values combine the objective functions of the two methods. Baird termed this a residual algorithm, which for clarity we refer to as a hybrid algorithm.

The original hybrid algorithm (Baird 1995) was incremental in nature as updates to the value function occur either after each observed transition or after each epoch. Incremental algorithms have the disadvantage that they make inefficient use of data and require appropriately setting a learning rate. Least-squares algorithms, which have access to a batch of samples and can make multiple passes over the data, remedy these issues. The fixed point algorithm, also referred to as the least-squares temporal difference (LSTD) algorithm, was proposed by Bradtke and Barto (1996) and generalized by Boyan (1999) to learn a value function for a fixed policy. Least-squares techniques have also been applied to the Bellman residual method. These two algorithms can be used to learn action-value functions for control problems (Lagoudakis and Parr 2003). When used within policy iteration, the fixed point algorithm tends to find better policies while the Bellman residual algorithm exhibits more stable behavior between rounds of policy iteration.[1] We propose two ways to implement hybrid algorithms using the more efficient least-squares approach.

Hybrid algorithms can be understood both analytically and geometrically. They produce approximate value functions that are projections of the target function V^π under different weightings. We provide equations for the projections. We also describe hybrid algorithms from a geometric perspective associated with the Bellman equation. This perspective offers some further intuition on the stability of the algorithms. Finally, we compare the various algorithms empirically on small domains and on Tetris, a large problem with more than 10^{60} states.

2 Background and terminology

We consider Markov decision processes (Puterman 1994) with a finite state space S of N states, a finite action space A, a state transition function $P(s, a, s')$ yielding the probability of moving from state $s \in S$ to state $s' \in S$ given action $a \in A$, and a reward function $R(s, a, s')$ giving the expected reward under the transition from $s \in S$ to $s' \in S$ given $a \in A$. A policy π is simply a mapping from states to actions. Let P^π be a $N \times N$ matrix with $P^\pi(i, j) = P(i, \pi(i), j)$ and let R^π be a length N vector with $R^\pi(i) = \sum_j P^\pi(i, j) R(i, \pi(i), j)$. The value function V^π is a length N vector that solves the Bellman equation

$$V^\pi = R^\pi + \gamma P^\pi V^\pi \tag{1}$$

where $\gamma \in [0, 1)$ is a discount factor.

[1] By stability, we do not mean the likelihood of a value function diverging to infinity (as was the original issue with incremental TD methods when used with function approximation (Baird 1995)). Rather, we refer to how rapidly the functions fluctuate between rounds of policy iteration.

Policy evaluation involves computing V^π for an arbitrary policy π. The value function V^π can be computed directly $V^\pi = (I - \gamma P^\pi)^{-1} R^\pi$ or iteratively

$$V_{k+1} = T^\pi(V_k) = R^\pi + \gamma P^\pi V_k \tag{2}$$

where $T^\pi : \mathbb{R}^N \to \mathbb{R}^N$ is the Bellman operator. The iterative method converges to V^π.

When an exact representation of V^π is infeasible, the value function must be approximated. We consider *linear* function approximation where a value function $\hat{V}$ is expressed as a linear combination of basis functions. This is written $\hat{V} = \Phi w$ where $w \in \mathbb{R}^K$ is an adjustable parameter vector and $\Phi = [\Phi_1 | \ldots | \Phi_K] \in \mathbb{R}^{N \times K}$ with each column Φ_i being a basis function. We assume without loss of generality that the basis functions are linearly independent. It is also typical for the number of free parameters to be much smaller than the number of states ($K \ll N$).

Approximate policy evaluation involves computing an approximate value function $\hat{V} = \Phi w$ that represents V^π. The following four techniques address this problem.[2]

1. *Optimal Approximate Solution* (*OPT*)
   If the target value function V^π were known, then it is easy to find an approximation $\hat{V}$ simply by projecting V^π onto the space spanned by the basis functions. This directly minimizes $\|\hat{V} - V^\pi\|_\rho$, where the errors for each state are weighted according to distribution ρ. Thus, the solution is $\hat{V} = \Phi w = \Pi_\rho V^\pi$ where $\Pi_\rho = \Phi(\Phi^T D_\rho \Phi)^{-1} \Phi^T D_\rho$ is a projection matrix and D_ρ is a diagonal matrix $D_\rho(i,i) = \rho(i)$. The difficulty of this method is in computing V^π, which can in principle be done using Monte Carlo methods.

2. *Bellman Residual Minimization* (*BR*)
   This technique computes a solution by minimizing the magnitude of the Bellman residual, $\|T^\pi(\hat{V}) - \hat{V}\|_\rho$, where the errors for each state are weighted according to distribution ρ:

$$\begin{aligned} &\min_w \|T^\pi(\hat{V}) - \hat{V}\|_\rho \\ &\quad = \min_w \|R^\pi + \gamma P^\pi \Phi w - \Phi w\|_\rho. \end{aligned} \tag{3}$$

   The least-squares solution is to minimize $\|A_{\mathrm{BR}} w - b_{\mathrm{BR}}\|_\rho$ where:

$$\begin{aligned} A_{\mathrm{BR}} &= \Phi^T (I - \gamma P^\pi)^T D_\rho (I - \gamma P^\pi) \Phi, \\ b_{\mathrm{BR}} &= \Phi^T (I - \gamma P^\pi)^T D_\rho R^\pi. \end{aligned}$$

   This technique, proposed by Schweitzer and Seidmann (1985), has also been referred to as the residual-gradient method (Baird 1995; Schoknecht 2003), the quadratic residual method (Munos 2003), and as the Bellman residual method (Lagoudakis and Parr 2003; Antos et al. 2008). When the model is unknown or is too large, the matrix A_{BR} and vector b_{BR} must be estimated from samples. To achieve an *unbiased* estimate, two samples from each state are required (Sutton and Barto 1998). Given double samples $\langle s, \pi(s), r', s' \rangle$ and $\langle s, \pi(s), r'', s'' \rangle$, the updates are

$$\begin{aligned} \hat{A}_{\mathrm{BR}} &= \hat{A}_{\mathrm{BR}} + \rho(s)(\phi(s) - \gamma\phi(s'))(\phi(s) - \gamma\phi(s''))^T, \\ \hat{b}_{\mathrm{BR}} &= \hat{b}_{\mathrm{BR}} + \rho(s)(\phi(s) - \gamma\phi(s'))r' \end{aligned}$$

[2] The first three techniques were similarly described by Munos (2003).

where $\phi(s)$ is a column vector of length K associated with the sth row of Φ (i.e. $\phi(s) = \Phi(s,:)^T$). If only a single sample is available, then replacing $\phi(s'')$ with $\phi(s')$ in the equation above for $\hat{A}_{\text{BR}}$ results in a biased estimate of A_{BR}. This occurs because the term $\gamma^2 \Phi^T (P^\pi)^T D_\rho P^\pi \Phi$ in A_{BR} cannot be estimated from just a single transition. Two common heuristics for dealing with this issue are to hypothesize a second sample using a nearest neighbor of s' and to not update $\hat{A}_{\text{BR}}$ until a state s has been visited at least twice. Recently, Antos et al. (2008) proposed a technique for avoiding double samples by adding an auxiliary function which itself must be optimized.

3. *Fixed Point Solution* (*FP*)
This technique computes a solution by forcing $\hat{V}$ to be a fixed point of the Bellman operator. Since the Bellman operator can back up values out of the space spanned by the basis functions, it must be followed by a projection onto the column space of Φ (written $[\Phi]$) to ensure $\hat{V}$ is a fixed point. Thus, the solution is to minimize $\|\Pi_\rho T^\pi(\hat{V}) - \hat{V}\|_\rho$:

$$\begin{aligned} &\min_w \|\Pi_\rho T^\pi(\hat{V}) - \hat{V}\|_\rho \\ &\quad = \min_w \|\Pi_\rho (T^\pi(\hat{V}) - \hat{V})\|_\rho \\ &\quad = \min_w \|\Pi_\rho (R^\pi + \gamma P^\pi \Phi w - \Phi w)\|_\rho. \end{aligned} \tag{4}$$

In the second step above, note that $\hat{V} = \Pi_\rho \hat{V}$. The least-squares solution to this problem is to find w such that $A_{\text{FP}} w = b_{\text{FP}}$ where:

$$\begin{aligned} A_{\text{FP}} &= \Phi^T D_\rho (I - \gamma P^\pi)\Phi, \\ b_{\text{FP}} &= \Phi^T D_\rho R^\pi. \end{aligned}$$

We refer to this technique as the FP solution to be consistent with previous work (Lagoudakis and Parr 2003), but it has also been referred to as the temporal difference method (Schoknecht 2003; Munos 2003). Unbiased estimates of the matrix A_{FP} and vector b_{FP} can be obtained from a single sample $\langle s, \pi(s), r', s' \rangle$ by the following updates:

$$\begin{aligned} \hat{A}_{\text{FP}} &= \hat{A}_{\text{FP}} + \rho(s)\phi(s)(\phi(s) - \gamma\phi(s'))^T, \\ \hat{b}_{\text{FP}} &= \hat{b}_{\text{FP}} + \rho(s)\phi(s)r'. \end{aligned}$$

4. *Hybrid Minimization* (*H*)
The BR solution minimizes the Bellman residual (3) and the FP solution minimizes the projected Bellman residual (4). Baird (1995) proposed *residual algorithms* as a way to combine these techniques. The term "residual" algorithm was used to emphasize that it was different from a "residual-gradient" algorithm (his terminology for BR). To avoid any confusion, we refer to residual algorithms as *hybrid algorithms*. This name also emphasizes the fact that it is a combination of BR and FP. Baird's original version was an incremental algorithm. An update to the weight vector was computed by linearly combining the updates due to the BR and FP: $\Delta w_{\text{H}} = \beta \Delta w_{\text{BR}} + (1 - \beta)\Delta w_{\text{FP}}$. We present two ways to formulate the hybrid technique using least-squares.

3 Hybrid least-squares algorithms

The hybrid approach accounts for both the Bellman residual (which is minimized by the BR in (3)) *and* the projection of the Bellman residual onto $[\Phi]$ (which is minimized by the

FP in (4). Our first algorithm H_1 linearly combines these two errors from which we derive a least-squares equation. The second algorithm H_2 is a more direct combination of the BR and FP least-squares statistics. Both H_1 and H_2 when used with an exact representation (i.e. $\Phi = I_N$) produce the target value function V^π. When using an approximate representation, H_1 and H_2 produce different results and have different storage and computational requirements.

3.1 Motivation

There are three factors that motivate hybrid algorithms. First, as pointed out by Baird (1995), hybrid algorithms are a general class of algorithms that include BR and FP as special cases at opposite ends of a spectrum. Fully understanding this spectrum is worthwhile in its own right. Also, as least-squares techniques have been applied to BR and FP (Bradtke and Barto 1996; Boyan 1999; Lagoudakis and Parr 2003) to make them more data efficient than their incremental counterparts, it makes sense to design a least-squares version of hybrid algorithms. These least-squares algorithms have an intuitive geometric perspective. BR and FP minimize the length of different sides of a triangle defined by the Bellman equation (Lagoudakis and Parr 2003). Hybrid algorithms naturally complete this perspective.

The second factor motivating hybrid least-squares algorithms is the empirical behavior of BR and FP when used within approximate policy iteration. The FP algorithm tends to produce better policies than the BR algorithm (Lagoudakis and Parr 2003). However, this increase in performance comes at the expense of stability (Baird 1995; Munos 2003). Li (2008) analyzed incremental versions of FP and BR under a particular learning model and concluded that BR can achieve smaller residuals while FP can make more accurate predictions. Hybrid algorithms have the potential to achieve both stability and improved performance. To illustrate this on a concrete example, consider the six state MDP shown in Fig. 1 with discount factor $\gamma = 0.99$. The optimal policy is to move right in the first three states and left in the last three states ($\pi^* = \mathit{RRRLLL}$). Let the initial policy be $\pi_0 = \mathit{LLLLLL}$ and assume there are three basis functions corresponding to the first three eigenvectors of the graph Laplacian (Mahadevan 2005). These basis functions are symmetric and are rich enough to represent an approximate value function whose corresponding greedy policy is π^*. The basis functions are shown in Fig. 2. The distribution ρ can be set to either the invariant distribution of P^{π_0} or the uniform distribution (which is appropriate when performing policy iteration (Koller and Parr 2000)); the results hold for both distributions. The approximate value functions $\hat{V}_{\mathrm{BR}}^{\pi_0}$ and $\hat{V}_{\mathrm{FP}}^{\pi_0}$ were computed according to the least-squares solutions described in Sect. 2. Then the model was used to determine a policy π_1 that is greedy with respect to $\hat{V}$. The BR produces a policy $\pi_1 = \mathit{LLLLLL}$ while the FP produces a policy $\pi_1 = \mathit{RRRRRR}$. Thus, after one round of policy iteration, BR converges on the initial policy and FP completely flips the policy. Moreover, since the model and basis functions are symmetric, FP oscillates forever between LLLLLL and RRRRRR. This example demonstrates FP's potential instability. We will revisit this example later to show that hybrid least-squares algorithms find solutions between these two extremes.

The third motivating factor is the bias associated with $\hat{A}_{\mathrm{BR}}$ when only a single sample is available from each state. Denote the sampled Bellman update for a fixed policy π as $\mathcal{T}$. Note this is a random variable, which depends on the samples, such that the expected value $\mathbf{E}[\mathcal{T}] = T^\pi$. Antos et al. (2008) showed that the expected value of the (single) sample-based Bellman residual is equal to the true Bellman residual plus the variance of the sampled Bellman update. This takes the form:

$$\mathbf{E}\left[\|\mathcal{T}\hat{V} - \hat{V}\|_\rho^2\right] = \|T^\pi \hat{V} - \hat{V}\|_\rho^2 + \rho^T \mathbf{Var}[\mathcal{T}\hat{V}]$$

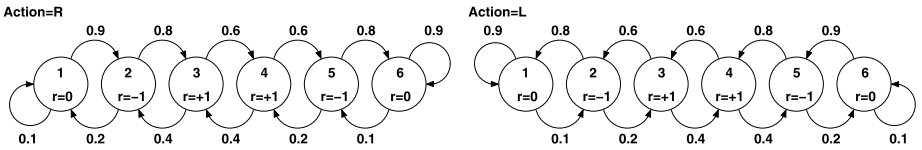


Fig. 1 Reward and transition functions for a six state MDP with two possible actions

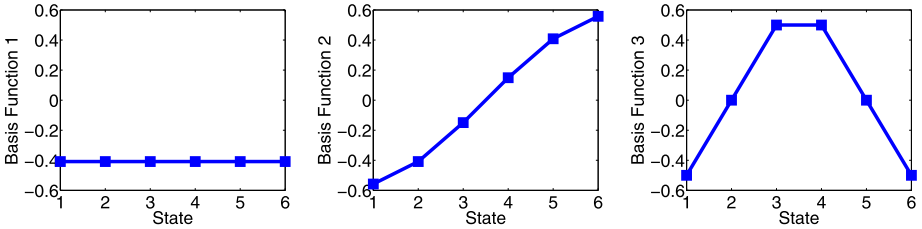


Fig. 2 First three Laplacian eigenvectors associated with the MDP in Fig. 1

where the variance $\mathbf{Var}[\cdot]$ is point-wise and ρ is a distribution vector. On the other hand, $\hat{A}_{\mathrm{FP}}$ is unbiased when only a single sample is available from each state. Hybrid algorithms, by naturally combining BR and FP, offer some control over the impact of the bias.

3.2 Algorithm H_1

We combine the BR minimization problem (3) and the FP minimization problem (4) with a parameter $\beta \in [0, 1]$. Simply combining these two problems results in a minimization over two separate norms; however, these can be combined into a single norm as we prove below:

$$
\begin{aligned}
&\min_w \big[\beta \|T^\pi(\hat{V}) - \hat{V}\|_\rho^2 + (1-\beta)\|\Pi_\rho(T^\pi(\hat{V}) - \hat{V})\|_\rho^2\big] \\
&\quad = \min_w \big[\beta \|(I - \Pi_\rho + \Pi_\rho)(T^\pi(\hat{V}) - \hat{V})\|_\rho^2 + (1-\beta)\|\Pi_\rho(T^\pi(\hat{V}) - \hat{V})\|_\rho^2\big] \\
&\quad = \min_w \big[\beta \|(I - \Pi_\rho)(T^\pi(\hat{V}) - \hat{V})\|_\rho^2 + \|\Pi_\rho(T^\pi(\hat{V}) - \hat{V})\|_\rho^2\big] \\
&\quad = \min_w \|\sqrt{\beta}(I - \Pi_\rho)(T^\pi(\hat{V}) - \hat{V}) + \Pi_\rho(T^\pi(\hat{V}) - \hat{V})\|_\rho^2 \\
&\quad = \min_w \|(\sqrt{\beta} I + (1 - \sqrt{\beta})\Pi_\rho)(T^\pi(\hat{V}) - \hat{V})\|_\rho^2 \\
&\quad = \min_w \|(\sqrt{\beta} I + (1 - \sqrt{\beta})\Pi_\rho)(R^\pi + \gamma P^\pi \Phi w - \Phi w)\|_\rho^2. \qquad (5)
\end{aligned}
$$

The chain of steps relies on the Pythagorean theorem (used in both the third and fourth lines) and the fact that $[I - \Pi_\rho]$ and $[\Pi_\rho]$ are orthogonal subspaces. Note that the squared length $\|\cdot\|^2$ was used in the derivation for ease of use with the Pythagorean theorem. A least-squares equation of the form $\|A_{\mathrm{H}_1} w - b_{\mathrm{H}_1}\|_\rho$ can be derived from the minimization problem in (5). To simplify the derivation, let $F = \sqrt{\beta} I + (1 - \sqrt{\beta})\Pi_\rho$ and let $G = (I - \gamma P^\pi)$:

$$
\begin{aligned}
FG\Phi w &= FR^\pi, \\
(FG\Phi)^T D_\rho FG\Phi w &= (FG\Phi)^T D_\rho FR^\pi,
\end{aligned}
$$

$$\Phi^T G^T F^T D_\rho F G \Phi w = \Phi^T G^T F^T D_\rho F R^\pi.$$

It is easy to show that $F^T D_\rho F = D_\rho(\beta I + (1-\beta)\Pi_\rho)$. The final result is therefore:

$$A_{\mathrm{H}_1} = \Phi^T (I - \gamma P^\pi)^T D_\rho (\beta I + (1-\beta)\Pi_\rho)(I - \gamma P^\pi)\Phi,$$
$$b_{\mathrm{H}_1} = \Phi^T (I - \gamma P^\pi)^T D_\rho (\beta I + (1-\beta)\Pi_\rho) R^\pi.$$

To estimate A_{H_1} and b_{H_1} from samples, it is necessary to store three $K \times K$ matrices and two length K vectors. This can be seen by rewriting the equations:

$$A_{\mathrm{H}_1} = \beta A_{\mathrm{BR}} + (1-\beta) A_{\mathrm{FP}}^T C^{-1} A_{\mathrm{FP}}$$
$$b_{\mathrm{H}_1} = \beta b_{\mathrm{BR}} + (1-\beta) A_{\mathrm{FP}}^T C^{-1} b_{\mathrm{FP}}$$

where $C = \Phi^T D_\rho \Phi$. Thus, A_{H_1} can be estimated by incrementally updating $\hat{A}_{\mathrm{BR}}$, $\hat{A}_{\mathrm{FP}}$, $\hat{b}_{\mathrm{BR}}$, $\hat{b}_{\mathrm{FP}}$, as well as the matrix $\hat{C}$ via $\hat{C} = \hat{C} + \rho(s)\phi(s)\phi(s)^T$ given sample $\langle s, \pi(s), r', s' \rangle$. If only a single sample is available from each state, then $\hat{A}_{\mathrm{H}_1}$ will be a biased estimate of A_{H_1} because of the bias in $\hat{A}_{\mathrm{BR}}$. However, as mentioned in Sect. 3.1, hybrid algorithms can reduce the impact of the bias. This is achieved simply by setting β to a value less than one. This occurs because only one of the two norms that make up H_1 is biased (i.e. $\mathbf{E}[\beta \|\mathcal{T}\hat{V} - \hat{V}\|_\rho^2]$ is biased but $\mathbf{E}[(1-\beta)\|\Pi_\rho(\mathcal{T}\hat{V} - \hat{V})\|_\rho^2]$ is unbiased).

Both the BR and FP least-squares problems only need to store one $K \times K$ matrix and one length K vector, whereas H_1 requires three matrices and two vectors. Moreover, the matrix C must be inverted when computing A_{H_1}. These issues motivated our second implementation.

3.3 Algorithm H_2

Rather than starting from the BR and FP minimization problems, we consider a direct combination of the BR and FP least-squares statistics:

$$\begin{aligned} A_{\mathrm{H}_2} &= \beta A_{\mathrm{BR}} + (1-\beta) A_{\mathrm{FP}} \\ &= \Phi^T (I - \beta\gamma P^\pi)^T D_\rho (I - \gamma P^\pi)\Phi, \\ b_{\mathrm{H}_2} &= \beta b_{\mathrm{BR}} + (1-\beta) b_{\mathrm{FP}} \\ &= \Phi^T (I - \beta\gamma P^\pi)^T D_\rho R^\pi. \end{aligned}$$

By definition, this technique returns the BR solution when $\beta = 1$ and the FP solution when $\beta = 0$. Only one $K \times K$ matrix and one length K vector are required for H_2. The incremental update given double samples $\langle s, \pi(s), r', s' \rangle$ and $\langle s, \pi(s), r'', s'' \rangle$ has the form:

$$\hat{A}_{\mathrm{H}_2} = \hat{A}_{\mathrm{H}_2} + \rho(s)(\phi(s) - \beta\gamma\phi(s'))(\phi(s) - \gamma\phi(s''))^T,$$
$$\hat{b}_{\mathrm{H}_2} = \hat{b}_{\mathrm{H}_2} + \rho(s)(\phi(s) - \beta\gamma\phi(s'))r'.$$

It is worthwhile noting that these updates are nearly identical to those for $\hat{A}_{\mathrm{BR}}$ and $\hat{b}_{\mathrm{BR}}$ except for the extra parameter β.

3.4 Difference between H_1 and H_2

Aside from the different data structures used by H_1 and H_2, it is useful to elucidate any other differences. To make this comparison more obvious, the least-squares equations can be rewritten as follows:

$$A_{\mathrm{FP}} = A_{\mathrm{BR}} - \gamma^2 \Phi^T (P^\pi)^T D_\rho P^\pi \Phi + \gamma \Phi^T (P^\pi)^T D_\rho \Phi,$$
$$A_{\mathrm{H}_1} = A_{\mathrm{BR}} - (1-\beta)\gamma^2 \Phi^T (P^\pi)^T D_\rho P^\pi \Phi + (1-\beta)\gamma^2 \Phi^T (P^\pi)^T D_\rho \Pi_\rho P^\pi \Phi,$$
$$A_{\mathrm{H}_2} = A_{\mathrm{BR}} - (1-\beta)\gamma^2 \Phi^T (P^\pi)^T D_\rho P^\pi \Phi + (1-\beta)\gamma \Phi^T (P^\pi)^T D_\rho \Phi.$$

Matrices A_{BR} and A_{H_1} are symmetric by definition whereas A_{FP} and A_{H_2} (except when $\beta = 1$) are not symmetric. Consider the extreme values of β. Both A_{H_1} and A_{H_2} are clearly the same as A_{BR} when $\beta = 1$. When $\beta = 0$, it is obvious that A_{H_2} and A_{FP} are identical. It is less obvious that H_1 produces the same solution w to the least-squares minimization as H_2 and FP when $\beta = 0$, but this can in fact be shown. The interesting case occurs when $0 < \beta < 1$ because the H_1 and H_2 solutions differ. Notice the only difference between A_{H_1} and A_{H_2} is in the final term shown above. The final term in A_{H_2} is $\gamma \Phi^T (P^\pi)^T D_\rho \Phi$, while A_{H_1} includes the same term times its transpose. This occurs during the least-squares derivation of A_{H_1}.

As shown in (5), H_1 can be written as a minimization over the sum of two norms: the norm of the Bellman residual and the norm of the projected Bellman residual. H_2, by virtue of directly combining the FP and BR least-squares solutions, does not have the same analytical form. Although H_2 seems less justified than H_1, it outperformed H_1 experimentally.

3.5 Other possible algorithms

The two proposed hybrid algorithms *implicitly* constrain the Bellman residual by the choice of the parameter β. This constraint could be made explicit. The problem would be to minimize the projection of the Bellman residual subject to an inequality constraint on the Bellman residual (either on its magnitude or component-wise):

$$\begin{aligned} &\min_w \|A_{\mathrm{FP}} w - b_{\mathrm{FP}}\|_\rho \\ &\text{subject to:} \quad \|A_{\mathrm{BR}} w - b_{\mathrm{BR}}\|_\rho \le \delta \\ &\text{or:} \quad \pm(A_{\mathrm{BR}} w - b_{\mathrm{BR}}) \le \Delta. \end{aligned}$$

The parameters $\delta \in \mathbb{R}^+$ and $\Delta \in \mathbb{R}^{K+}$ must be set appropriately based on the minimal value of the Bellman residual magnitude attained with the BR. We point out the possibility of explicitly controlling the Bellman residual to be thorough. However, since this increases the computational complexity, we limit our discussion to the two simple least-squares algorithms H_1 and H_2.

4 Analysis

4.1 Projections of the target function

The first three approximate policy evaluation techniques were shown to be images of the target function V^π under different projection operations (Schoknecht 2003). More specifically, each method $X = \{\mathrm{OPT}, \mathrm{BR}, \mathrm{FP}\}$ produces an approximate value function with the

following form: $\hat{V} = \Phi w = \Phi A_X^{-1} b_X = \Phi(\Phi^T D_X \Phi)^{-1} \Phi^T D_X V^\pi$. The matrix D_X controls the weighting of the projection and takes on the following values (Schoknecht 2003):

$$
\begin{aligned}
D_{\text{OPT}} &= D_\rho, \\
D_{\text{BR}} &= (I - \gamma P^\pi)^T D_\rho (I - \gamma P^\pi), \\
D_{\text{FP}} &= D_\rho (I - \gamma P^\pi).
\end{aligned}
$$

The hybrid methods have a similar characterization:

$$
\begin{aligned}
D_{\text{H}_1} &= (I - \gamma P^\pi)^T D_\rho (\beta I + (1-\beta)\Pi_\rho)(I - \gamma P^\pi), \\
D_{\text{H}_2} &= (I - \beta\gamma P^\pi)^T D_\rho (I - \gamma P^\pi).
\end{aligned}
$$

4.2 Geometry of the Bellman equation

Each approximate policy evaluation algorithm uses the Bellman equation in different ways to compute a value function. There is an intuitive geometric perspective to the algorithms when using linear function approximation. We expand on the original presentation of this perspective (Lagoudakis and Parr 2003).

The Bellman equation with linear function approximation has three components: $\hat{V}$, $T^\pi(\hat{V})$, and $\Pi_\rho T^\pi(\hat{V})$. These components geometrically form a triangle where $\hat{V}$ and $\Pi_\rho T^\pi(\hat{V})$ reside in the space spanned by Φ while $T^\pi(\hat{V})$ is, in general, outside this space. This is illustrated in the leftmost triangle of Fig. 3. The three-dimensional space in the figure is the space of exact value functions while the two-dimensional plane represents the space of approximate value functions in $[\Phi]$. The angle between subspace $[\Phi]$ and the vector $T^\pi(\hat{V}) - \hat{V}$ is denoted θ. The BR and FP solutions minimize the length of different sides of the triangle. The second triangle in Fig. 3 shows the BR solution, which minimizes the length of $T^\pi(\hat{V}) - \hat{V}$. The third (degenerative) triangle shows the FP solution, which minimizes the length of $\Pi_\rho T^\pi(\hat{V}) - \hat{V}$. This length is 0 which means $\theta_{\text{FP}} = 90°$. The fourth triangle shows the H solution, which minimizes a combination of the lengths of the two sides. In general, θ_{H} lies between θ_{BR} and 90°. The hybrid solution allows for controlling the shape of this triangle. We purposefully drew the triangles in Fig. 3 suggestively to not only accentuate their angles, but also to emphasize that the length of the Bellman residual

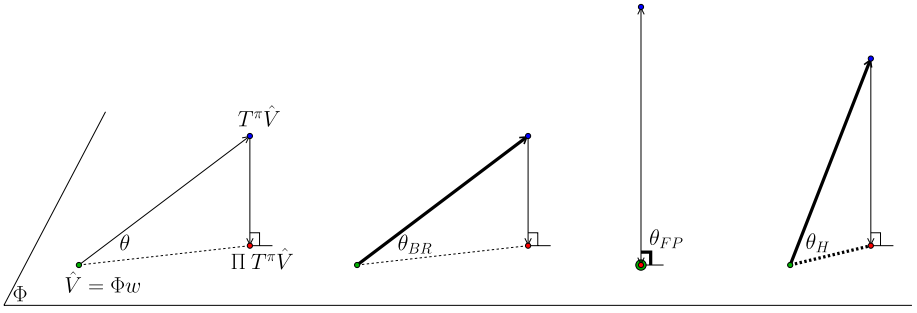


Fig. 3 The triangle on the left shows the general form of the Bellman equation. The other three triangles correspond to the different approximate policy evaluation algorithms where the *bold lines* indicate what is being optimized

$(T^{\pi}(\hat{V}) - \hat{V})$ can become large at times for FP. By including the norm of the Bellman residual in their objective functions, hybrid algorithms can protect against such large residual vectors. They also have the flexibility of finding solutions that are almost fixed points but have more desirable properties (smaller Bellman residuals).

5 Experiments

We first present a policy evaluation experiment on the six state chain MDP (Fig. 1) described in Sect. 3.1. This simple problem makes evident the difference between the BR and FP solutions. The next two experiments compare the various approximate policy evaluation algorithms within the broader context of policy iteration. We emphasize there is a linear dependence on β only for policy evaluation. Policy iteration results are not necessarily linear in β given the nonlinearity of policy improvement.

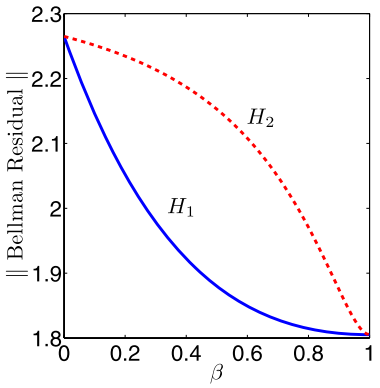
5.1 Chain MDP

Reconsider the chain MDP from Fig. 1 and the basis functions from Fig. 2. The optimal policy for this MDP with a discount factor $\gamma = 0.99$ is $\pi^* = \textit{RRRLLL}$. Starting from initial policy $\pi_0 = \textit{LLLLLL}$, BR results in an approximate value function $\hat{V}_{\mathrm{BR}}^{\pi_0}$ whose greedy policy is also $\pi_1 = \textit{LLLLLL}$. In other words, after one round of policy iteration, BR converges on the initial policy. FP produces an approximate value function $\hat{V}_{\mathrm{FP}}^{\pi_0}$ whose greedy policy is $\pi_1 = \textit{RRRRRR}$. Hybrid algorithms find solutions between these two extremes. We ranged the value of β from 0 to 1 and computed $\hat{V}_{\mathrm{H}_1}^{\pi_0}$ and $\hat{V}_{\mathrm{H}_2}^{\pi_0}$ using the equations in Sect. 3 (the transition matrix and reward function were not sampled but rather were used explicitly). We also recorded the norm of the Bellman residual, the norm of the projected Bellman residual, the angle between the Bellman residual and the space spanned by the basis functions Φ, and the greedy policies associated with the approximate value functions. The results are shown in Fig. 4 using a uniform distribution ρ; however, the results are very similar when setting ρ to be the invariant distribution of P^{π_0}. Note the trade-off between the Bellman residual and the projected Bellman residual for different values of β in Figs. 4(a) and 4(b). In Fig. 4(b), the curve associated with H_2 is beneath that of H_1. This indicates algorithm H_2 places more weight on minimizing the projected Bellman residual compared to algorithm H_1. Also, note that the greedy policies in Figs. 4(d) and 4(e) run the full gamut from $\textit{RRRRRR}$ at $\beta = 0$ to $\textit{LLLLLL}$ at $\beta = 1$.

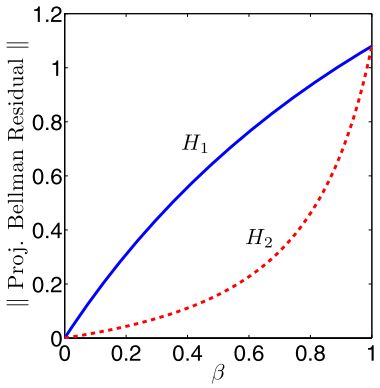
5.2 Grid MDP

We compared all methods on a 10×10 grid MDP. The MDP has 100 states, 4 actions that have probability 0.9 of success (an unsuccessful action resulted in a transition in one of the other three directions), a 0.95 discount factor, and a reward of $+1$ in one corner and $+2$ in the diagonal corner. Fifteen Laplacian eigenvectors (Mahadevan 2005) were used as basis functions.

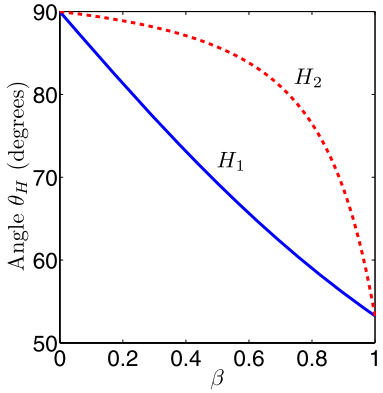
We ran 500 trials. Each trial began with a randomly initialized policy, then policy iteration was run using each policy evaluation method until the weight vector converged or 500 iterations were reached. The model was used during policy iteration to avoid any difficulty comparing the various methods due to sampling. The result of policy iteration is a final policy π_f. We evaluate these policies by computing V^{π_f} exactly and comparing it with V^*. The results, which are broken into the trials that converged and those that did not converge, are shown in Fig. 5.



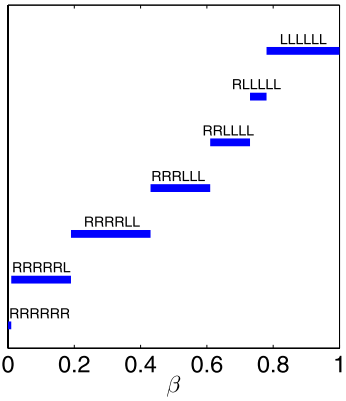
(a) Bellman residual vs. β.



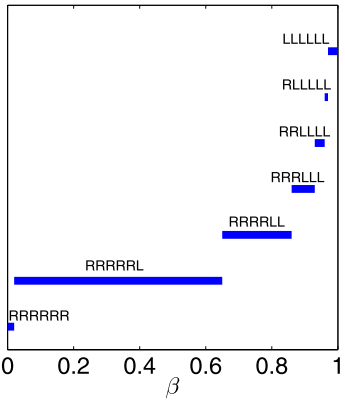
(b) Proj. Bellman residual vs. β.



(c) Angle θ_H vs. β.

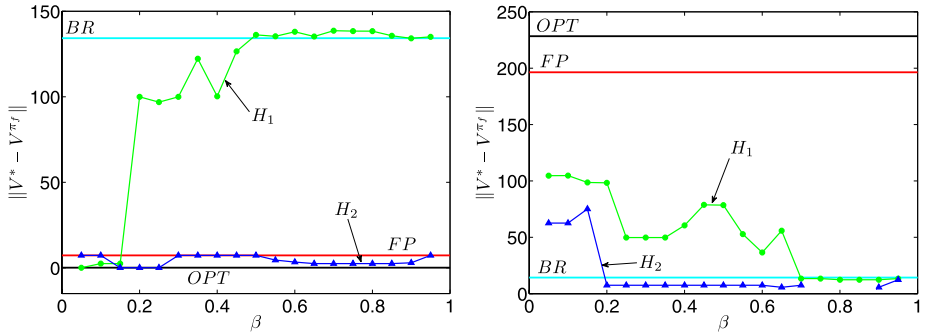


(d) Greedy policy w.r.t. $\hat{V}^{\pi_0}_{H_1}$.

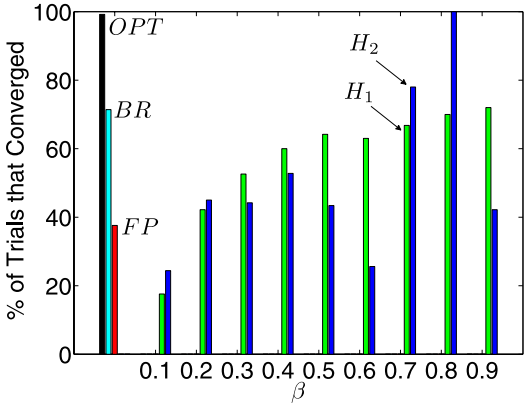


(e) Greedy policy w.r.t. $\hat{V}^{\pi_0}_{H_2}$.

Fig. 4 Results of approximate policy evaluation using H_1 and H_2 for the MDP in Fig. 1



(a) Converged trials. (b) Non-converged trials.



(c) % of trails that converged.

Fig. 5 Results of 500 policy iteration trials for the grid MDP. The results are divided into those trials that converged (**a**) versus those that did not converge (**b**). The median value of $\|V^* - V^{\pi_f}\|$ is plotted versus β, where π_f is the final policy attained when policy iteration terminates. The percentage of trials that converged is shown in (**c**)

The BR algorithm converged almost twice as often as the FP algorithm (71.4% compared to 37.6%). However, when BR did converge, it happened after only 8.5 rounds of policy iteration on average. That strongly contrasts with FP's average of 89.1 rounds of policy iteration until convergence. Since BR tends to make small changes to the value function between rounds of policy iteration, it is not surprising that this early convergence (starting from a random policy) leads to very suboptimal policies. It is interesting that BR discovered better policies when policy iteration did not converge. On the other hand, when FP converged it found excellent policies (small values of $\|V^* - V^{\pi_f}\|$).

The policies found by algorithm H_1 had a general linear trend between $\beta = 0$ (FP) and $\beta = 1$ (BR). The policy iteration convergence rate for H_1 had a similar effect. The convergence rate was not nearly as predictable for H_2. In fact, at $\beta = 0.8$, all 500 trials converged. The most interesting aspect of this experiment is the excellent performance of H_2. The method produced good policies regardless of convergence and across all β values.

5.3 Tetris

We have presented hybrid least-squares algorithms for approximating value functions, but the same idea holds for approximating action-value functions. We omit the equations for

Table 1 Results of policy iteration for Tetris. An asterisk indicates policy iteration converged

Technique	Score	Technique	Score
BR	0	FP*	630
$H_1, \beta = 0.1$	15	$H_2, \beta = 0.1$*	800
$H_1, \beta = 0.2$	0	$H_2, \beta = 0.2$*	580
$H_1, \beta = 0.3$	80	$H_2, \beta = 0.3$*	645
$H_1, \beta = 0.4$	295	$H_2, \beta = 0.4$*	515
$H_1, \beta = 0.5$	60	$H_2, \beta = 0.5$*	455
$H_1, \beta = 0.6$	5	$H_2, \beta = 0.6$*	395
$H_1, \beta = 0.7$	5	$H_2, \beta = 0.7$*	370
$H_1, \beta = 0.8$	0	$H_2, \beta = 0.8$*	405
$H_1, \beta = 0.9$	0	$H_2, \beta = 0.9$*	330

lack of space, but they are very similar to those in Sect. 3. We tested all policy evaluation methods on the problem of learning an approximate action-value function for Tetris. Ten basis functions over state-action pairs (s, a) were used. The first four are for the current state s: maximum height, number of holes, sum of absolute height differences between adjacent columns, and the mean height. The next four basis functions are the change in the value of the first four features after taking action a from s. The last two are the change in the score and a constant 1. This feature set was proposed by Lagoudakis et al. (2002).

Forty episodes of data (∼30,000 samples) were generated using an initial policy greedy with respect to weight vector $w^{\pi_0} = [-1, -10, -1, -1, -2, -11, -2, -2, 50, 10]^T$. We ran policy iteration starting from w^{π_0} until the weight vector converged or 100 iterations were reached. Instead of generating double samples to form unbiased estimates of A and b, we used the model to compute the expectation over next-states and actions. For Tetris, each action results in seven equally likely next-states corresponding to the seven Tetris pieces. This method of using the model instead of samples was described by Lagoudakis and Parr (2003) (see **LSTD*Q*-Model**).

We tested the learned policies 50 times. Each time, we generated a random ordered set of pieces that *all* policies were forced to place to make the comparison more accurate. This is necessary because Tetris performance can be very sensitive to the exact order of pieces. The average score over the 50 trials is shown in Table 1. The initial policy w^{π_0} scored 310 on average. Policy iteration converged in less than 7 iterations for FP and H_2, whereas BR and H_1 did not converge. The performance split along this division. The final policy computed using BR rarely removed a line. This was also the case for policies learned using H_1 except when $\beta = 0.4$. On the other hand, the FP and H_2 policies performed at least as well as the initial policy and in some cases significantly better. The best policy was computed using H_2 with $\beta = 0.1$.

6 Conclusions

The fixed point (FP) and Bellman residual (BR) algorithms can be combined to form a hybrid approximate policy evaluation algorithm. We proposed two ways to implement hybrid algorithms using least-squares methods, thus improving efficiency over the original incremental algorithm (Baird 1995). The first implementation solved a least-squares problem minimizing the sum of two norms: the norm of the Bellman residual and the norm of the projected Bellman residual. The second implementation is a direct linear combination of the

least-squares statistics for FP and BR. We analyzed the algorithms in terms of projections of the target function and we showed that hybrid algorithms have an intuitive geometric interpretation.

Hybrid least-squares algorithms attempt to combine the stability of the BR solution with the improved performance of the FP solution. We presented an example on a chain MDP demonstrating this effect. Policy iteration experiments were conduced on a simple grid MDP so that the quality of the learned policies could be determined analytically. Experiments were also run on the challenging task of learning to play Tetris where learned policies were evaluated empirically. In both domains, the hybrid algorithm H_2 discovered policies that performed much better than BR and H_1 and as well as, and in some instances better than, FP. The H_2 algorithm has the same data structures and computational complexity as BR and FP. A surprising finding was H_2's robustness for a wide range of β values. One would expect that for β values close to 1, the difference between BR and H_2 would be minimal. An explanation of this effect would be useful. Other interesting areas for future work include a deeper understanding of the objective function being minimized by H_2 and a mechanism for automatically setting β. The latter idea could be challenging because the best value could depend on the specific domain, set of basis functions, and the policy being evaluated.

Acknowledgements We would like to thank the reviewers for providing valuable comments that improved the quality of the paper. Support for this research was provided in part by the National Science Foundation under grants IIS-0534999 and IIS-0803288.

References

Antos, A., Szepesvári, C., & Munos, R. (2008). Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, *71*(1), 89–129.

Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the 12th international conference on machine learning* (pp. 30–37).

Boyan, J. (1999). Least-squares temporal difference learning. In *Proceedings of the 16th international conference on machine learning* (pp. 49–56).

Bradtke, S., & Barto, A. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, *22*(1–3), 33–57.

Koller, D., & Parr, R. (2000). Policy iteration for factored MDPs. In *Proceedings of the 16th conference on uncertainty in artificial intelligence* (pp. 326–334). San Mateo: Morgan Kaufmann.

Lagoudakis, M., & Parr, R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, *4*, 1107–1149.

Lagoudakis, M., Parr, R., & Littman, M. (2002). Least-squares methods in reinforcement learning for control. In *Proceedings of the 2nd Hellenic conference on artificial intelligence* (pp. 249–260).

Li, L. (2008). A worst-case comparison between temporal difference and residual gradient with linear function approximation. In *Proceedings of the 25th international conference on machine learning* (pp. 560–567).

Mahadevan, S. (2005). Representation policy iteration. In *Proceedings of the 21st conference on uncertainty in artificial intelligence* (pp. 372–379).

Munos, R. (2003). Error bounds for approximate policy iteration. In *Proceedings of the 20th international conference on machine learning* (pp. 560–567).

Puterman, M. (1994). *Markov decision processes: discrete stochastic dynamic programming*. New York: Wiley.

Schoknecht, R. (2003). Optimality of reinforcement learning algorithms with linear function approximation. In *Advances in neural information processing systems* (Vol. 15, pp. 1555–1562).

Schweitzer, P., & Seidmann, A. (1985). Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, *110*, 568–582.

Sutton, R. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, *3*, 9–44.

Sutton, R., & Barto, A. (1998). *Reinforcement learning*. Cambridge: MIT Press.