2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

An Application of Convex Optimization Concepts to Approximate Dynamic Programming

Edilson F. Arruda, Marcelo D. Fragoso and João Bosco R. do Val

Abstract— **This paper deals with approximate value iteration (AVI) algorithms applied to discounted dynamic (DP) programming problems. The so-called Bellman residual is shown to be convex in the Banach space of candidate solutions to the DP problem. This fact motivates the introduction of an AVI algorithm with local search that seeks an approximate solution in a lower dimensional space called approximation architecture. The optimality of a point in the approximation architecture is characterized by means of convex optimization concepts and necessary and sufficient conditions to global optimality are derived. To illustrate the method, two examples are presented which were previously explored in the literature.**

I. INTRODUCTION

Dynamic Programming (DP), e.g. [5] is a very elegant and powerful tool devised to solve (stochastic) sequential decision and planning problems. However, the solution to large scale DP problems can become prohibitively demanding, see [12]. Approximate Dynamic Programming (ADP) algorithms, e.g. [7], [13], [14], offer a plausible alternative of finding approximate solutions to DP problems that would otherwise be intractable. This paper explores a very popular ADP technique of incorporating a function approximation scheme into the problem and searching an approximate solution in a lower dimensional space. This approach has proved successful in real-world applications, e.g. [15]. Nevertheless, convergence is not one of its intrinsic properties. In fact, authors have striven to produce robust and convergent ADP algorithms, e.g. [2], [9].

Convergent ADP algorithms often rely on specific properties of the approximation architecture and/or the *projection/fitting operator*, i.e. the operator that converts elements in the Banach space of value function candidates into elements (*approximate solutions*) in the approximation space. The non-expansion based algorithm in [9] applies to non-expansive projection mappings. Residual and gradient descent algorithms, see [2] and [3], respectively, were derived for problems with a differential mapping from value function to approximation parameters. These algorithms seek to decrease the so-called Bellman residual at each iteration. Although convergence is guaranteed, not much can be inferred about the accumulation point. A convergent algorithm for general approximation architectures under a class of expansive projection mappings was introduced in [1]. It was shown to converge to the projection of a local solution to the DP problem in the approximation space under suitable conditions.

This work introduces an algorithm similar in nature to the residual algorithm and addresses its properties. In contrast to the latter algorithm, where the objective is to guarantee convergence and the nature of the accumulation point is to some extent immaterial, the proposed algorithm seeks to minimize the Bellman residual in the approximation space. This is motivated by the fact that the Bellman residual is in fact an estimate of the distance between any value function candidate and the *true* value function. Moreover, since the value function is not known a priori, an evaluation based on a given error criterion with respect to it is impractical.

This paper also studies properties of the Bellman residual. An appropriate function of the Bellman residual is shown to be convex in the Banach space of real-valued functions. Given that convexity implies strictly quasi-convexity, this makes it possible to introduce a derivative-free local search (*line search*) procedure (e.g. Fibonnacci search or Golden Section, see [4][Chapter 8]) in the proposed algorithm, in order to identify descent directions for the desired function. Hence, in contrast to residual algorithms, the proposed procedure does not require the existence of a differential mapping from real-valued functions (i.e. value function candidates) to approximation parameters. Furthermore, convex programming theory can be applied to derive sufficient conditions for the optimality of any given element in the approximation space. For more details regarding the optimization of convex functions, see [4].

II. PRELIMINARIES

Let S be the state space of the dynamic programming (DP) problem *P*. Let $\mathcal{B}$ be the space of non-negative real valued functions $V : S \to \mathbb{R}$. A standard approach to solving *P* is to define a contraction mapping T in the supremum norm that maps $\mathcal{B}$ into itself, e.g. [12]. The unique fixed point of mapping T, which is denoted by V^* and coincides with the solution to *P*, can be computed recursively by the value iteration algorithm

$$\begin{aligned} &V_0 \in \mathcal{B} \\ &V_{k+1} = TV_k. \end{aligned} \tag{1}$$

This algorithm can be deemed as an unconstrained subgradient algorithm that takes at any iteration k a *descent direction* $d_k = TV_k - V_k$. Both the convergence and uniqueness of the fixed point of Algorithm (1) follow from the contraction property, which states that

E. F. Arruda and M. D. Fragoso are with the Department of Systems And Control, National Laboratory for Scientific Computation, Petrópolis RJ, Brazil efarruda@lncc.br, frag@lncc.br

J. B. R. do Val is with the Department of Telematics, School of Electrical and Computer Engineering, State University of Campinas - UNICAMP, Campinas SP, Brazil jbosco@dt.fee.unicamp.br

$$||TV - V'|| \leq \alpha ||V - V'||, \ \forall V, V' \in \mathcal{B}, \tag{2}$$

where $|| \cdot ||$ denotes the supremum norm.

When the state space is prohibitively large, an alternative is to substitute Algorithm (1) for an approximate value iteration (AVI) algorithm that seeks an approximate solution in a parametric *approximation space* $A \subset \mathcal{B}$. The approximate algorithm applies mapping T to a subset of S and *projects* the samples thus obtained into the approximation space by means of a projection operator $\mathcal{P}_A : \mathcal{B} \rightarrow A$. Such an algorithm and can be defined by the recursion below

$$\begin{array}{l} \mathcal{V}_0 \in A \\ \mathcal{V}_{k+1} = \mathcal{V}_k + \mathcal{P}_A(T\mathcal{V}_k) - \mathcal{V}_k. \end{array} \tag{3}$$

The goal is to find the solution to the auxiliary problem

$$\min_{\mathcal{V} \in A} f(\mathcal{V}) := \min_{\mathcal{V} \in A} d(\mathcal{V}, V^*), \tag{4}$$

where $d : A \rightarrow \mathbb{R}$ is some distance function defined a priori. Hence, the approximate algorithm seeks the best available approximation (in A) to V^*, according to some error criterion defined a priori. However, considering the V^* is not known a priori, the evaluation of f in (4) is impractical, and therefore the optimality of any element in A cannot be verified unless the original DP problem is solved to optimality.

In contrast to pure DP algorithms, which converge to the optimal solution and monotonically decrease the so-called 'Bellman' residual ($||TV - V||, \ V \in \mathcal{B}$, where $|| \cdot ||$ denotes the supremum norm), AVI algorithms can present a divergent behavior, e.g. [8]. Moreover, the Bellman residual is no longer guaranteed to monotonically decrease. In order to overcome this difficulty, residual gradient algorithms, introduced in [2], alter the update in (4) to make sure that the residual is decreased at each update, based on the derivative of the residual with respect to the approximation parameters. However, the properties of the accumulation point of such an algorithm remain poorly understood and not much can be inferred about the quality of the approximate solution obtained.

This work introduces an alternative AVI algorithm that seeks to minimize a semi-norm function of the Bellman residual, which can be deemed as an upper bound to f (4), and is indicative of the quality of any given value function approximation. This algorithm comprises a derivative-free local search procedure that makes sure that this norm function is decreased at each iteration, thereby eliminating the need for a differential mapping from real-valued functions to approximation parameters.

III. Formulation and Definitions

Let P be a discounted DP problem and $\mathcal{B}$ be the Banach space of value function candidates for this problem. Let S denote the state space of the discounted DP problem and $R(V) := TV - V, \ V \in \mathcal{B}$ be the Bellman residual associated to V. It is known that the solution to P is unique, e.g. [6] and that a necessary and sufficient condition for the optimality of $V^* \in \mathcal{B}$ is $R(V^*) = 0$. Therefore, problem P can be equivalently formulated as

$$\min g(V) = \min ||R(V)||_s = \min ||TV - V||_s, \quad V \in \mathcal{B}. \tag{5}$$

where $|| \cdot ||_s$ denotes the span semi-norm, given by

$$||V||_s = \max_{x \in S} V(x) - \min_{x \in S} V(x).$$

The following equations hold.

$$\begin{aligned} ||V^* - V|| &\leq ||TV^* - TV|| + ||TV - V|| \\ &\leq \alpha ||V^* - V|| + ||TV - V||, \end{aligned}$$

where the first equation results from the triangular inequality, noting that $V^* = TV^*$; and the last inequality follows from (2). The above expressions imply

$$||V^* - V|| \leq \frac{||TV - V||}{1 - \alpha}. \tag{6}$$

Hence, considering the definitions of the supremum norm and the span semi-norm, at any point $V \in \mathcal{B}$ it holds that

$$||V^* - V|| \leq \frac{g(V)}{1 - \alpha} + C, \tag{7}$$

where α is the linear convergence rate of the VI algorithm, and

$$C = \min_{x \in S} \frac{TV(x) - V(x)}{1 - \alpha}$$

Therefore $g(V)$ is, up to a constant, indicative of the proximity between V and the exact solution V^* to the DP problem.

In this work, we address a constrained version of the DP problem, which can formulated as

$$\begin{array}{l} \min g(\mathcal{V}) = ||R(\mathcal{V})||_s = ||T\mathcal{V} - \mathcal{V}||_s, \quad \mathcal{V} \in \mathcal{B}. \\ \text{subject to } \mathcal{V} \in A \end{array} \tag{8}$$

The concepts below, extracted from [4], will be useful in the definition of local optimality that follows.

Definition 1: Given a point $V \in \mathcal{B}$ and $\epsilon > 0$, the set $N_\epsilon(V) = \{V' : ||V' - V|| \leq \epsilon\}$ is called an ϵ-neighborhood of V.

Definition 2: $\overline{\mathcal{V}} \in A$ is a local minimum to (8) if there exists an ϵ-neighborhood $N_\epsilon(\mathcal{V})$ around $\overline{\mathcal{V}}$ such that $R(\overline{\mathcal{V}}) \leq R(\mathcal{V})$ for all $\mathcal{V} \in N_\epsilon(\overline{\mathcal{V}}) \cap A$, for some $\epsilon > 0$.

IV. Optimality Characterization

The linear convergence rate of the VI algorithm to the optimal solution of problem (5), e.g. [6], suggests that this problem is a well defined one. Indeed, we prove in this section that function $g : \mathcal{B} \rightarrow \mathbb{R}$, $g = ||R\mathcal{V}||_s$ is a convex function. Therefore, convex optimization theory yields that any local optimum to (5) is also a global optimum and the solution to this problem can be equivalently searched by means of any subgradient algorithm. Moreover, for convex approximation architectures, the same result yields that the solution to (8) coincides with any local minimum of that problem.

This section proceeds with a proof that function $g : \mathcal{B} \to \mathbb{R}$ is convex. It finishes with a characterization of global optimality for Problem (8), which makes use of this result.

Let $d \in \mathcal{B}$ be an arbitrary direction, $x \in S$ and $V_1 \in \mathcal{B}$ and $\alpha \in (0, 1)$ be the convergence rate of the VI algorithm. Then, for any $d \in \mathcal{B}$,

$$T(V_1 + d)(x) = TV_1(x) + \alpha E[d(x_1)|x_0 = x].$$

With these elements, the following lemma can be stated.

Lemma 1: Function $g : \mathcal{B} \to \mathbb{R}$ is a convex function.

Proof: From the definition of R, one can easily see that

$$R(V_1 + \delta)(x) = RV_1 + \alpha E[\delta(x_1)|x_0 = x] - \delta(x),\ \forall x \in S$$

It follows from the above equality that

$$\begin{aligned} R(V_1 + (1-\lambda)\delta)(x) =& RV_1(x) + \\ & (1-\lambda)\Big[\alpha E\big[\delta(x_1)|x_0 = x\big] - \delta(x)\Big] \\ R(V_1 + (1-\lambda)\delta)(x) =& RV_1(x) + \\ & (1-\lambda)\big[R(V_1+\delta)(x) - RV_1(x)\big] \\ R(V_1 + (1-\lambda)\delta)(x) =& \lambda RV_1(x) + \\ & (1-\lambda)R(V_1+\delta)(x),\ \forall x \in S. \end{aligned} \tag{9}$$

Hence, it follows that

$$\begin{aligned} R(\lambda V_1 + (1-\lambda)V_2)(x) &= R(V_1 + (1-\lambda)(V_2 - V_1)))(x) \\ &= \lambda RV_1(x) + (1-\lambda)RV_2(x), \end{aligned} \tag{10}$$

where the last equality follows from (9), with $\delta = (\mathcal{V}_2 - \mathcal{V}_1)$.

By the triangular inequality,

$$||R(\lambda V_1 + (1-\lambda)V_2)||_s \leq \lambda ||RV_1||_s + (1-\lambda)||RV_2||_s$$

And that concludes the proof. ■

It follows from the lemma above that any local minimum to (5) is also a global minimum. The same applies to problem (8) when the approximation space A is convex.

Let $F_{\mathcal{V}} = \{d : (\mathcal{V} + d) \in A\},\ \mathcal{V} \in A$ and let

$$F := \{\ d \in F_{\mathcal{V}} : g(\mathcal{V} + \lambda d) \leq g(\mathcal{V}),\ \forall \lambda \in (0, \delta),\ \delta > 0\ \}$$

denote the set of improving or descent directions of function g at $\mathcal{V}$.

Proposition 1 (Necessary and Sufficient Condition): $\overline{\mathcal{V}} \in A$ is a global minimum to (8) if and only if $F = \emptyset$.

Proof: Suppose, in view of contradiction, that $F \neq \emptyset$. Then, there exists a direction $d \in F_{\overline{\mathcal{V}}} : ||R(\overline{\mathcal{V}} + \lambda d)||_s \leq ||R(\overline{\mathcal{V}})||_s,\ \forall \lambda \in (0, \delta)$ for some $\delta > 0$. Therefore, $\overline{\mathcal{V}}$ does not meet Definition 2 and is not a local minimum (8).

Now suppose $\overline{\mathcal{V}}$ is a local minimum to (8). Then, by Definition 2, $F = \emptyset$. Thus, it has been shown that $F = \emptyset$ is a necessary and sufficient condition for local optimality.

By the convexity of A, and Lemma 1, any local minimum to (8) is also a global minimum. And that concludes the proof. ■

A straightforward corollary to Proposition 1 is stated below.

Corollary 1 (Necessary Condition): If $\overline{\mathcal{V}}$ is a global minimum to (8), then $\overline{\mathcal{V}}$ is also a global minimum in the subset

$$A_{\bar{\mathcal{V}},\delta} = \{\bar{\mathcal{V}} \in A : \bar{\mathcal{V}} = \bar{\mathcal{V}} + \lambda\delta,\ \lambda \in \mathbb{R}\},$$

$\delta = \mathcal{P}_A(TV) - \mathcal{V}$.

Proposition 1 is very powerful, but can be difficult to verify. When the approximation space is non-differentiable with respect to the parameters a verification step can imply sweeping every feasible direction around a solution candidate. Corollary 1 provides a weaker condition that can be verified by the simple application of the prescribed projection mapping and applications of the VI mapping T. At a minimum, one obtains the optimal solution in the convex subset $A_{\bar{\mathcal{V}},\ \delta}$ of the approximation architecture, defined above.

V. The Proposed Algorithm

In order to exploit the results in Section IV and improve the convergence properties of Algorithm (3), a new algorithm is proposed that incorporates a unidimensional search procedure whenever necessary. At any iteration, in case the projection mapping does not provide a descent direction, i.e. if the current iterate satisfies Corollary 1, unidimensional search is applied sequentially in a prescribed set of feasible directions until a descent direction is identified or the set is completely swept. The idea is to find a residual decreasing direction in the prescribed set and minimize the residual in that direction. The algorithm terminates when no further improvement is attained. A pseudo-code of the proposed algorithm is presented below.

Any standard unidimensional search procedure can be applied in Step 4 of Algorithm 1. However, for general approximation architectures, a derivative-free procedure, such as Golden Section or Fibonacci search is advised. Observe that the eventual function evaluations in the local search step can make use of Equation (10). Therefore, an explicit evaluation of $g(V)$ is only necessary in the end points of the segment being evaluated. Thus, at most $n+1$ applications of mapping T are required during each local search cycle (step 4) of the proposed algorithm.

VI. Numerical Example

In this section, we replicate the chain-walk problem from [11]. The problem consists of a chain of N states, labeled 1 to N. At each state there are two actions available, “go left” (L) and “go right” (R). The actions succeed with probability 0.9 and fail with probability 0.1; when an action fails, the complementary action is the one actually executed; the two boundaries of the chain are dead-ends. The “reward” function is nil at the boundaries and unitary otherwise and the discount factor is 0.9. The objective is to maximize the cumulative discounted reward in an infinite horizon. Figure 1 depicts a chain walk example with $N = 4$.

The chain-walk example is attractive because it is simple enough to allow a comparison between the exact and the approximate solution. Furthermore, the simple 4-state chain

Algorithm 1 Approximate Value Iteration with Unidimensional Search

```
Step 1: (Initialization)
        • Choose $\mathcal{V}_0 \in A$
        • Choose $n$ and $tol$
        • $k \leftarrow 0$
        • $sp \leftarrow \infty$
        • $\epsilon_0 = ||T\mathcal{V}_0 - \mathcal{V}_0||_s$
Step 2: (Projection)
        • $\mathcal{V}' = \mathcal{P}_A(T\mathcal{V}_k)$
        • $\epsilon_{k+1} = ||T\mathcal{V}' - \mathcal{V}'||_s$
Step 3: (Descent Direction Verification)
        • If $(\epsilon_{k+1} < \epsilon_k - tol)$
          – $\mathcal{V}_{k+1} = \mathcal{V}'$
          – $k \leftarrow k + 1$
          – Return to Step 2
        • Else
          – $\delta \leftarrow \mathcal{V}' - V$
        • End If
Step 4: (Unidimensional Search)
        • $d_1 \leftarrow \delta$
        • Select Feasible Directions $\{d_2, \ldots, d_n\},\ n < \infty$
        • $F_d = \{d_1, \ldots, d_n\}$
        • $j \leftarrow 1,\ \bar{\lambda} \leftarrow 0$
        • While $(\lambda = 0)$ AND $j \leq n$
          – $\bar{\lambda} \leftarrow \arg\min_{\lambda \in [-1,1]} ||R(\mathcal{V}_k + \lambda d_j)||_s$
        • End While
        • $\mathcal{V}' = \mathcal{V}_k + \bar{\lambda} d_j$
        • $\mathcal{V}_{k+1} \leftarrow \mathcal{V}'$
Step 4:(Convergence Test)
        • If $||\mathcal{V}_{k+1} - \mathcal{V}_k|| \leq tol$
          – STOP
        • Else
          – $k \leftarrow k + 1$, return to step 2
        • End If
```

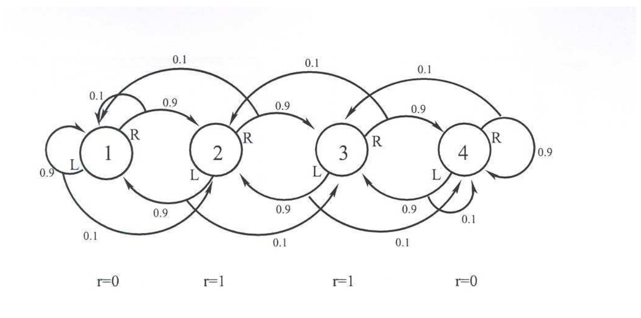

Fig. 1. Four-state Chain-walk Problem

in Figure 1 was reported to offer difficulties to a policy iteration based approximate dynamic programming algorithm [10].

Following [10] and [11], this paper uses a linear architecture to represent the value function as a combination of the following basis functions:

$$\phi(s) = \begin{pmatrix} 1 \\ s \\ s^2 \end{pmatrix},$$

where s is the state number. Let v^t indicate the transpose a vector v. The approximation architecture is the space of linear combinations of the basis functions $A = \{\phi w,\ w \in \mathbb{R}^{3\times 1}\}$, with

$$\phi = \begin{pmatrix} \phi(1)^t \\ \vdots \\ \phi(N)^t \end{pmatrix}.$$

The projection operator $\mathcal{P}_A$ is linear regression with least-squares minimization. Our experiments showed that, unlike its counterpart in [10], Algorithm 3 does not fail to converge to the value function for the example in Figure 1, which belongs the the approximation space. Therefore, the algorithm in Section V converges without a single application of the unidimensional search routine. This suggests that a VI-based approach can be more adequate for this problem than a policy-iteration based approach.

To verify the behavior of Algorithm 1 when the value function does not belong to the approximation domain, experiments with a 5-state and a 10-state chain-walk examples were performed. In these examples, approximate solutions with non-zero Bellman residual have to be pursued. In each experiment, $\mathcal{V}_0 = 0$ and the set F_d in Step 4 of Algorithm 1 is comprised of 5 (five) elements, which are vectors pointing from $\mathcal{V}' = \mathcal{P}_A(T\mathcal{V}_k)$ to the last 5 iterates of the algorithm, i.e. $F_d = [\mathcal{V}' - \mathcal{V}_{k-1}\ \ldots\ \mathcal{V}' - \mathcal{V}_{k-5}]^t$. In the first 5 iterations, F_d is completed with randomly generated feasible directions.

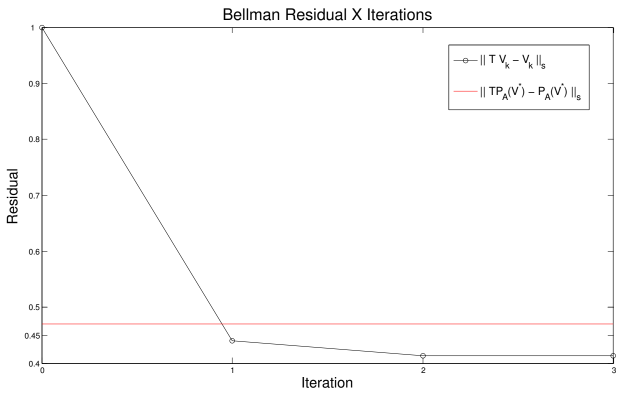

Fig. 2. Bellman Residual for the 5-state Chain-walk Problem

Figure 2 plots the Bellman residual against the iteration number for the 5-state chain-walk example. It shows that the algorithm takes 3 iterations to converge. Observe that the Bellman residual of the accumulation point is better than the value function projection $\mathcal{P}_A(V^*)$ in terms of Bellman residual. In Figure 3, the exact solution to the

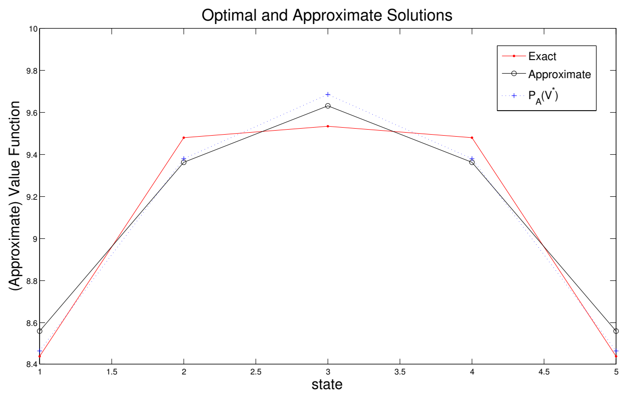

Fig. 3. Exact and Approximate Solutions to the 5-state Chain-walk Problem

problem is plotted against its projection in the approximation architecture and the solution obtained by Algorithm 1.

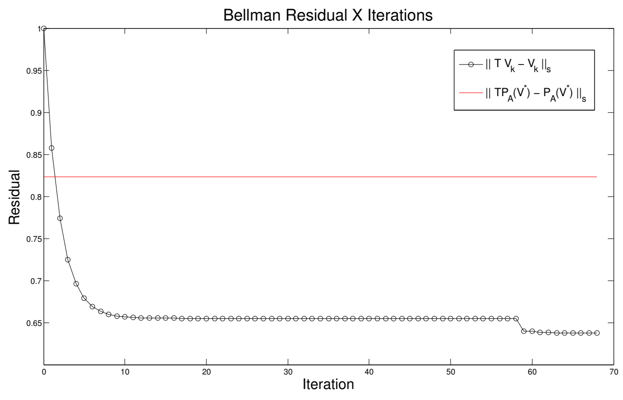

Fig. 4. Bellman Residual for the 10-state Chain-walk Problem

Figure 4 depicts the Bellman residual for the 10-state chain-walk example. For this Example, Algorithm 1 converges in 68 iterations. Once again, the Bellman residual is decreased considerably if compared to its counterpart at the value function projection $\mathcal{P}_A(V^*)$. However, one can observe in Figure 5 that the solution to the proposed algorithm is more distant from the value function than $\mathcal{P}_A(V^*)$ in terms of the span semi-norm. This seemingly counter-intuitive fact may happen when the residual is not a tight bound to the distance to the value function, i.e. when the upper bound in (7) not a tight bound.

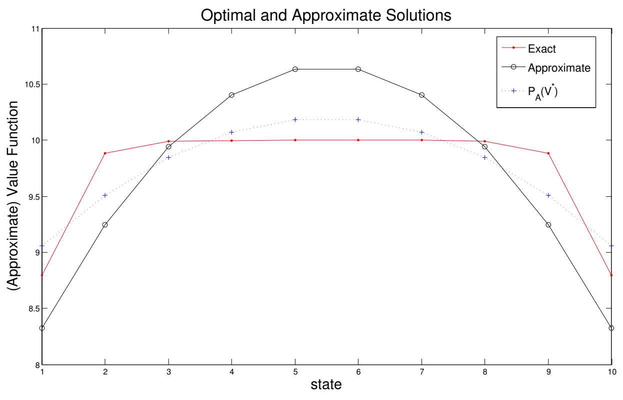

Fig. 5. Exact and Approximate Solutions to the 10-state Chain-walk Problem

VII. Concluding Remarks

This paper showed that the Bellman residual is a convex function in the Banach space of real-valued functions. This fact motivates the introduction of an approximate value iteration (AVI) algorithm with local search devised to solve large scale dynamic programming problems. Defining an approximate problem with a convex objective function allows one to use derivative-free local search procedures and extend residual algorithms to the context of convex approximation spaces that are non-differentiable with respect to the parameters.

The proposed algorithm is applied to a class of problems previously explored in the literature for the purpose of benchmarking. The solution to the proposed problems illustrate the strengths and drawbacks of the method. The results illustrate the potential of the proposed algorithm to find low-residual approximate solutions. Considering that the objective function is an upper bound to the distance to the value function, the quality of the approximate solution is also a function of the tightness of this bound.

VIII. Acknowledgments

This work was supported by CNPq, Brazil, under Grants No. 301740/2007-0, 470527/2007-2, FAPERJ under Grant No. E-26/100.579/2007 and FAPESP under Grant No. 03/06736-7.

References

[1] E. F. Arruda and J. B. R. do Val. Approximate dynamic programming based on expansive projections. In *Proceedings of the 45^{th} IEEE International Conference on Decision and Control*, pages 5537–5542, San Diego, 2006.

[2] L. C. Baird. Residual algorithms: reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Machine Learning*, pages 30–37, Tahoe City CA, 1995.

[3] L. C. Baird and A. Moore. Gradient descent for general reinforcement learning. *Advances in Neural Information Processing Systems 11*, 1999.

[4] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: Theory and algorithms*. John Wiley & Sons, New York, 2 edition, 1993.

[5] R. Bellman. *Dynamic programming*. Princeton University Press, Princeton, NJ, 1957.

[6] D. P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, Belmont, 2 edition, 1995.

[7] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, Belmont, 1996.

[8] J. A. Boyan and A. W Moore. Generalization in reinforcement learning: Safely approximating the value function. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 369–376. MIT Press, Cambridge MA, 1995.

[9] G. J. Gordon. Stable function approximation in dynamic programming. In *Proceedings of the 12th International Conference on Machine Learning*, pages 261–268, Tahoe City CA, 1995.

[10] D. Koller and R. Parr. Policy iteration for factored MDPs. In *Proceedings of the* 16^{th} *Conference on Uncertainty in Artificial Intelligence*, pages 2130–2155, Stanford CA, 1998.

[11] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

[12] M. L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, New York, 1994.

[13] J. Si, A. Barto, W. Powell, and D. Wunsch. *Handbook of learning and approximate dynamic programming*. John Wiley & Sons-IEEE Press, Piscataway NJ, 2004.

[14] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, 1998.

[15] G. Tesauro. Practical issues in temporal difference learning. *Machine Learning*, 8(3):257–277, 1992.