Approximate dynamic programming with $(\min, +)$ linear function approximation for Markov decision processes

Contributions:
I am the first author of this paper and my contributions are listed as under.

1. Formulating the projected Bellman equation (PBE) in the $(\min, +)$ basis.
2. Proposing novel approximate dynamic programming (ADP) algorithms based on the PBE in the $(\min, +)$ basis.
3. The error analysis for the proposed ADP algorithms.

The paper makes fundamental contributions by proposing for the first time in literature novel ADP algorithms based on $(\min, +)$ basis.

The paper was presented in the 53^{rd} IEEE conference on Control and Decision, Los Angeles, December 2014.

Approximate dynamic programming with $(\min, +)$ linear function approximation for Markov decision processes

Chandrashekar L[1] and Shalabh Bhatnagar[2]

***Abstract*— Markov Decision Process (MDP) is a useful framework to study problems of optimal sequential decision making under uncertainty. Given any MDP the aim here is to find the optimal action selection mechanism i.e., the optimal policy. Typically, the optimal policy (u^*) is obtained by substituting the optimal value-function (J^*) in the Bellman equation. Alternatively, u^* is also obtained by learning the optimal state-action value function Q^* known as the Q value-function. However, it is difficult to compute the exact values of J^* or Q^* for MDPs with large number of states. Approximate Dynamic Programming (ADP) methods address this difficulty by computing lower dimensional approximations of J^*/Q^*. Most ADP methods employ linear function approximation (LFA), i.e., the approximate solution lies in a subspace spanned by a family of pre-selected basis functions. The approximation is obtained via a linear least squares projection of higher dimensional quantities and the L_2 norm plays an important role in convergence and error analysis. In this paper, we discuss ADP methods for MDPs based on LFAs in the $(\min, +)$ algebra. Here the approximate solution is a $(\min, +)$ linear combination of a set of basis functions whose span constitutes a subsemimodule. Approximation is obtained via a projection operator onto the subsemimodule which is different from linear least squares projection used in ADP methods based on conventional LFAs. MDPs are not $(\min, +)$ linear systems, nevertheless, we show that the monotonicity property of the projection operator helps us establish the convergence of our ADP schemes. We also discuss future directions in ADP methods for MDPs based on the $(\min, +)$ LFAs.**

I. Introduction

Markov Decision Process (MDP) is a framework to pose optimal sequential decision making problems. In particular, an MDP is a four tuple $< S, A, P, g >$, where S is the state space, A is the action space, P is the probability transition kernel and g is the reward function. In this paper, we consider MDPs with $|S| = n$ states and $|A| = d$ actions. A policy is a map $u\colon S \to A$ and under u action $u(s)$ is chosen in states $s \in S$. The value of a state s under policy u is denoted by $J_u(s)$, and $J_u = (J_u(s), s \in S) \in \mathrm{R}^n$ is the value function corresponding to policy u. Solving an MDP involves solving two sub-problems namely *prediction* and *control*. The *prediction* problem constitutes evaluating the value function $J_u \in \mathrm{R}^n$ of a given policy u. The *control* problem involves coming up with desirable policies. The optimal value function is the fixed point of the Bellman equation given as $J^* = TJ^*$, where T is the Bellman operator. Once J^* is known u^* can be obtained by substituting J^* in the Bellman equation. The problem of prediction and control can be addressed simultaneously by computing $Q^* \in \mathrm{R}^{n\times d}$, known as the Q value function. Q^* encodes both J^* and u^*, and obeys the equation $Q^* = HQ^*$, where H is the Q-Bellman operator. When n is small, conventional methods such as value iteration (VI), policy iteration (PI) and linear programming (LP) can be used to compute J^*/Q^* and u^*. VI computes J^*/u^* and is based on the contraction property of T/H. PI computes J_{u_n} (*prediction*) of policy u_n based on which obtains an improved policy u_{n+1} (*control*) such that the sequence $u_n \to u^*$ as $n \to \infty$. The LP method computes J^* by representing the BE as a set of linear inequalities.

When n is large computing exact values of J^*/u^* is not possible and we need to resort to Approximate Dynamic Programming (ADP) methods that compute an approximate value-function $\tilde{J}$ and a sub-optimal policy $\tilde{u}$. Most ADP methods employ linear function approximators (LFAs), i.e., approximate the value function is given by $\tilde{J} = \Phi r^*$, where Φ is a $n\times k$ feature matrix and $r^* \in \mathrm{R}^k$ ($k << n$) is a weight vector to be computed. Approximate policy iteration (API) is the ADP analogue of PI in that it computes $\tilde{J}_u \approx J_u$ that obeys the Projected Bellman Equation (8) and then uses $\tilde{J}_u$ to perform policy improvement. An important shortcoming is that only $||J_u - \tilde{J}_u||_2$ can be bounded under certain restricted assumptions. However in order to guarantee policy improvement one needs to bound $||J_u - \tilde{J}_u||_\infty$. Consequently policy improvement is not guaranteed and API might not produce a sequence of convergent policies always and even in the cases when API converges to a sub-optimal policy $\tilde{u}$ its performance cannot be ascertained. Also, there is no convergent ADP based on conventional LFA to compute approximate J^*/Q^*. Approximate Linear Programming (ALP) [1] offers guarantee on the approximate policy but is limited by the large number of constraints. As a result, ADP methods based on conventional LFAs either do not address both the prediction and control problems (in the case of API) or do not alleviate the curse altogether (in the case of ALP).

The $(\min, +)$ $\big((\max, +)\big)$ algebra differs from conventional algebra, in that '$+$' and '$\times$' operators are replaced by '$\min$' ('$\max$') and '$+$' respectively. It is known that finite horizon deterministic optimal control problems based on maximizing a reward criterion are $(\max, +)$ linear transformations. A lot of work has been reported in the literature [2], [3], [4], [5] that make use of $(\max, +)$ basis to compute approximate value-functions for deterministic optimal con-

Work was supported by projects from Department of Science and Technology, Government of India as well as Xerox Corporation, USA.

[1]Chandrashekar L is a researcher with the Department of Computer Science and Automation Indian Institute of Science, Bangalore-560012, India. chandrul@csa.iisc.ernet.in

[2]Shalabh Bhatnagar is a faculty member with the Department of Computer Science and Automation Indian Institute of Science, Bangalore-560012, India. shalabh@csa.iisc.ernet.in

trol problems. The primary focus of the paper is to explore ADP schemes for MDPs based on $(\min, +)$ LFAs as opposed to conventional LFAs. The organization of the paper and our specific contributions in this paper are listed below.

1) In section II we briefly introduce the MDP setting and then discuss the basic solution methods.
2) In section III we discuss the ADP methods based on conventional LFAs and their shortcomings.
3) In section IV we introduce the $(\min, +)$ linear basis and introduce the PBE in the $(\min, +)$ basis.
4) In sections V and VI we present the main contribution of this paper namely the Approximate Q Iteration (AQI) and Variational Approximate Q Iteration schemes respectively. Both AQI and VAQI make use of the projected Bellman operator (denoted by Π_M) and the variational form of the projected Bellman operator (denoted by Π_M^W) respectively to compute $\tilde{Q} \approx Q^*$. We show that Π_M and Π_M^W are contraction maps in the L_∞-norm. Since AQI and VAQI schemes compute an approximate Q value they address the prediction and control problems, and using the L_∞ norm contraction property of Π_M and Π_M^W one can show convergence of these schemes and provide error bounds for the sub-optimal policies obtained by them.

II. Markov Decision Processes

An MDP is characterized by its state space S, action space A, the reward function $g\colon S \times A \to \mathrm{R}$, and the probability of transition from state s to s' under action a denoted by $p_a(s, s')$. The reward for selecting an action a in state s is denoted by $g_a(s)$. We consider MDP with state space $S = \{1, 2, \ldots, n\}$ and action set $A = \{1, 2, \ldots, d\}$. For simplicity, we assume that all actions $a \in A$ are feasible in all states $s \in S$. A policy is a map $u\colon S \to A$ that describes the action selection mechanism[1]. Under a policy u the MDP is a Markov chain and we denote its probability transition kernel by $P_u = (p_{u(i)}(i, j), i = 1 \text{ to } n, j = 1 \text{ to } n)$. The expected discounted reward starting from state s when following policy u is denoted by $J_u(s)$ and is defined as

$$J_u(s) = \mathbf{E}[\sum_{t=0}^{\infty} \alpha^t g_{a_t}(s_t) | s_0 = s, u].$$

Here $\{s_t\}$ is the trajectory or sequence of states of the Markov chain under u when $s_0 = s$, and $a_t = u(s_t), \forall t \geq 0$. Also $\alpha \in (0, 1)$ is a given discount factor. We call $J_u = (J_u(s), \forall s \in S) \in \mathrm{R}^n$ the value-function corresponding to policy u. The optimal policy denoted as u^* is given by $u^* = (u^*(s), s \in S)$ with

$$u^*(s) = \arg\max_{u \in U} J_u(s), \forall s \in S,$$

where U is the set of all SDPs. The optimal value function is then $J^*(s) = J_{u^*}(s), \forall s \in S$. The optimal value function and optimal policy are related by the Bellman Equation as below:

$$J^*(s) = \max_{a \in A}(g_a(s) + \alpha \sum_{s'=1}^{n} p_a(s, s') J^*(s')), \tag{1a}$$

$$u^*(s) = \arg\max_{a \in A}(g_a(s) + \alpha \sum_{s'=1}^{n} p_a(s, s') J^*(s')). \tag{1b}$$

Usually, J^* is computed first and u^* is obtained by substituting J^* in (1b). One can also define the state-action value-function of a policy u, i.e., the Q value-function as follows:

$$Q_u(s, a) = \mathbf{E}[\sum_{t=0}^{\infty} \alpha^t g_{a_t}(s_t) | s_0 = s, a_0 = a],$$

with $a_t = u(s_t), \forall t > 0$. The optimal Q function obeys the Q Bellman equation given below:

$$Q^*(s, a) = g_a(s) + \alpha \sum_{s'} p(s, s') \max_{a'} Q^*(s', a'),$$

$\forall s \in S, a \in A$. It is easy to see that $J^*(s) = \max_a Q^*(s, a)$. The optimal policy can be computed as $u^*(s) = \arg\max_a Q^*(s, a)$. Thus, computing Q^* addresses both the problems of *prediction* and *control* since it encodes both J^* as well as u^*.

A. Basic Solution Methods

It is important to note that J^* and Q^* are the fixed points of maps T and H respectively defined as follows: For $J \in \mathrm{R}^n$ and $Q \in \mathrm{R}^{n \times d}$

$$(TJ)(s) = \max_{a \in A}(g_a(s) + \alpha \sum_{j=1}^{n} p_a(s, s') J(s')), \tag{2a}$$

$$(HQ)(s, a) = (g_a(s) + \alpha \sum_{j=1}^{n} p_a(s, s') \max_{a \in A} Q(s', a)), \tag{2b}$$

T and H are called the Bellman and Q-Bellman operators respectively. Given $J \in \mathrm{R}^n, Q \in \mathrm{R}^{n \times d}$, TJ and HQ are the 'one-step' greedy value-functions. We summarize certain useful properties of H in the following lemmas (see [7] for proofs).

Lemma 1: H is a max-norm contraction operator, i.e., given $Q_1, Q_2 \in \mathbf{R}^{n \times d}$

$$||HQ_1 - HQ_2||_\infty \leq \alpha ||Q_1 - Q_2||_\infty. \tag{3}$$

Corollary 1: Q^* is a unique fixed point of H.

Lemma 2: H is a monotone map, i.e., given $Q_1, Q_2 \in \mathrm{R}^{n \times d}$ such that $Q_1 \geq Q_2$, it follows that $HQ_1 \geq HQ_2$.

Lemma 3: Given $Q \in \mathrm{R}^{n \times d}$, $k \in \mathrm{R}$ and $\mathbf{1} \in \mathrm{R}^{n \times d}$ - a vector with all entries 1, we have

$$H(Q + k\mathbf{1}) = HQ + \alpha k \mathbf{1}. \tag{4}$$

It is easy to check that Lemmas1, 2 and 3 hold for T as well ([7]).

Value iteration (VI) is the most basic method to compute J^*/Q^* and works using the fixed point iterations

$$J_{n+1} = TJ_n, \tag{5a}$$

$$Q_{n+1} = HQ_n. \tag{5b}$$

[1] A policy thus defined is known as stationary deterministic policy (SDP). Policies can also be non-stationary and randomized. However since there exists an optimal policy that is SDP ([6]) we restrict our treatment to SDPs alone.

Iterations in (5) constitute an exact method, and the contraction property of the Bellman operator ensures that $J_n \to J^*$ in (5a) and $Q_n \to Q^*$ in (5b), respectively as $n \to \infty$. They are also referred to as *look-up-table* methods or full state representation methods, as opposed to methods employing function approximation. u^* can be computed by substituting J^* in (1b). Another basic solution method is Policy Iteration (PI) presented in Algorithm 1. Alternatively, J^* can also be

Algorithm 1 Policy Iteration

1: Start with any policy u_0
2: **for** $i = 0, 1, \ldots, n$ **do**
3: Evaluate policy u_i by computing J_{u_i}.
4: Improve policy $u_{i+1}(s) = \arg\max_a(g_a(s) + \alpha \sum_{s'} p_a(s, s')J_{u_i}(s'))$.
5: **end for**
6: **return** u_n

computed using the linear programming (LP) formulation given by

$$\min \; c^\top J \tag{6}$$
$$s.t \quad J(s) \geq g(s, a) + \alpha \sum_{s'} p_a(s, s')J(s'), \forall s \in S, a \in A;$$

VI and PI form the basis of ADP methods explained in the next section.

III. Approximate Dynamic Programming in conventional LFAs

The phenomenon called *curse-of-dimensionality* denotes the fact that the number of states grows exponentially in the number of state variables. Due to the *curse*, as the number of variables increase, it is hard to compute exact values of J^*/Q^* and u^*. Approximate Dynamic Programming (ADP) methods make use of (1) and dimensionality reduction techniques to compute an approximate value-function $\tilde{J}$. Then $\tilde{J}$ can be used to obtain an approximate policy $\tilde{u}$ which is greedy with respect to $\tilde{J}$ as follows:

$$\tilde{u}(s) = \arg\max_a(g_a(s) + \alpha \sum_{s'} p_a(s, s')\tilde{J}(s')). \tag{7}$$

A bound on the error between the value-function $J_{\tilde{u}}$ corresponding to the greedy policy in (7) and the optimal value function is given by the following result.

Lemma 4: Let $\tilde{J} = \Phi r^*$ be the approximate value function and $\tilde{u}$ be as in (7), then

$$||J_{\tilde{u}} - J^*||_\infty \leq \frac{2}{1-\alpha}||J^* - \tilde{J}||_\infty.$$

Proof: See [7]. ■

Thus a good ADP method is one that addresses both the *prediction* (i.e., computing $\tilde{J}$) and the *control* (i.e., computing $\tilde{u}$) problems with desirable approximation guarantees.

LFA has been widely employed for its simplicity and ease of computation. Here, one typically lets $\tilde{J} \in V \subset \mathrm{R}^n$, where V is the subspace spanned by a set of preselected basis functions $\{\phi_i \in \mathrm{R}^n, i = 1, \ldots, k\}$. Let Φ be the $n \times k$ matrix with columns $\{\phi_1, \ldots, \phi_k\}$, and $V = \{\Phi r | r \in \mathrm{R}^k\}$, then the approximate value function $\tilde{J}$ is of the form $\tilde{J} = \Phi r^*$ for some $r^* \in \mathrm{R}^k$. Here, r^* is a weight vector to be learnt, and due to dimensionality reduction ($k << n$), computing $r^* \in \mathrm{R}^k$ is easier than computing $J^* \in \mathrm{R}^n$.

We now discuss popular ADP methods namely approximate policy evaluation (APE) and approximate policy iteration (API), which are analogous to VI and PI. APE and API are based on solving the projected Bellman equation which is analogous to Bellman equation in (1).

A. Approximate Policy Evaluation

J^* or Q^* are usually not known and hence finding their projections onto V is not possible. Nevertheless, one may use the Bellman operator and the linear least squares projection operator to write down a projected Bellman equation (PBE) as below:

$$\Phi r^* = \Pi T_u \Phi r^*, \tag{8}$$

where $\Pi = \Phi(\Phi^\top D \Phi)^{-1}\Phi^\top$ is the projection operator, D is a diagonal matrix with all diagonal entries strictly greater than 0 and T_u is the Bellman operator restricted to a policy u and is given by

$$(T_u J)(s) = g_{u(s)}(s) + \alpha \sum_{s'} p_{u(s)}(s, s')J(s'), J \in \mathrm{R}^n.$$

The approximate policy evaluation (APE) in this setting is the following iteration:

$$\Phi r_{n+1} = \Pi T_u \Phi r_n. \tag{9}$$

Here, $\Pi T_u \colon \mathrm{R}^n \to \mathrm{R}^n$ is the projected Bellman operator and the convergence of (9) can be established by showing that ΠT_u is a contraction map.

Lemma 5: If $\Pi = \Phi(\Phi^\top D \Phi)^{-1}\Phi^\top$, and D is a diagonal matrix with the i^{th} diagonal entry being the stationary probability of visiting state i under policy u, then ΠT_u is a contraction map with factor α.

Proof: See [7]. ■

Corollary 2: The iteration in (9) converges to Φr^* such that $\Phi r^* = \Pi T_u \Phi r^*$.

The error bound for Φr^* is given by

$$||J_u - \Phi r^*||_D \leq \frac{1}{\sqrt{1-\alpha^2}}||J_u - \Pi J_u||_D. \tag{10}$$

One is inclined to think that iteration (9) being an ADP analogue of (5a) would yield an approximation to J^*. However, it is important to note that (9) only computes $\tilde{J}_u$ because (9) contains T_u and not T. Since the operator ΠT might not be a contraction map in the L_2 norm, T_u cannot be replaced by T in iteration (9), and the PBE in (8).

B. Approximate Policy Iteration

Approximate Policy Iteration (Algorithm 2) tackles both prediction and control problems, by performing APE and policy improvement at each step.

Algorithm 2 Approximate Policy Iteration (API)

1: Start with any policy u_0
2: **for** $i = 0, 1, \ldots, n$ **do**
3: Approximate policy evaluation $\tilde{J}_i = \Phi r_i^*$, where $\Phi r_i^* = \Pi T_{u_i} \Phi r_i^*$.
4: Improve policy $u_{i+1}(s) = \arg\max_a (g_a(s) + \alpha \sum_{s'} p_a(s, s') \tilde{J}_i(s'))$.
5: **end for**
6: **return** Suboptimal policy $\tilde{u}$=u_n

The performance guarantee of API can be stated as follows:

Lemma 6: If at each step i one can guarantee that $||\tilde{J}_i - J_{u_i}||_\infty \leq \delta$, then one can show that $\lim_{n\to\infty} ||J_{u_i} - J^*|| \leq \frac{2\delta\alpha}{(1-\alpha)^2}$.

Note that the error bound required by Lemma 6 is in the L_∞ norm, whereas (10) is only in the L_2 norm. So API cannot guarantee an approximate policy improvement at each step which is a shortcoming. Also, even though each evaluation step (line 3 of Algorithm 2) converges, the sequence $u_n, n \geq 0$ is not guaranteed to converge. This is known as policy *chattering* and is another shortcoming of conventional LFAs. Thus the problem of *control* is only partially addressed by API, i.e, suffers in both convergence and performance guarantees.

C. LFAs for Q value-function

To alleviate the shortcomings in API, it is then natural to look for an approximation method that computes a policy in a more direct fashion. Since we know that by computing Q^* we obtain the optimal policy directly, it is a good idea to approximate Q^*. The PBE version of (5b) is a plausible candidate and the iterations are given by

$$\Phi r_{n+1} = \Pi H \Phi r_n. \tag{11}$$

The above scheme runs into the following problems:

1) The H operator (2b) contains a $\max$ term, and it is not straightforward to show that ΠH is a contraction map in L_2 norm, and consequently one cannot establish the convergence of iterates in (11).
2) The issue pertaining to $\max$ operator can be alleviated by restricting H to a policy u, i.e., by considering iterations of the form

$$\Phi r_{n+1} = \Pi H_u \Phi r_{n+1}, \tag{12}$$

where H_u is the Q-Bellman analog of the operator T_u. However, iterates in (12) attempt to approximate Q_u and not Q^*, which means the problem of *control* is unaddressed.

D. Approximate Linear Programming

An ADP method making use of LFA based on the LP formulation (6) of the MDP is the ALP formulation given by

$$\min \; c^\top \Phi r \tag{13}$$
$$s.t \quad \Phi r(s) \geq g_a(s) + \alpha \sum_{s'} p_a(s, s')(\Phi r)(s'), \forall s \in S, a \in A.$$

The ALP computes an approximate value function $\tilde{J}$ and a sub-optimal policy $\tilde{u}$ can be obtained by substituting $\tilde{J}$ in the BE. The ALP method does offer performance guarantees [1] for $\tilde{u}$, however it is limited by the large number of constraints. We conclude the section by making the following observations about ADPs based on conventional LFAs.

- Methods such as APE, API which make use of PBE in (8) neither have guaranteed convergence nor performance bounds for the sub-optimal policy $\tilde{u}$.
- The ALP, while it offers performance guarantees for $\tilde{u}$, is limited by the large number of constraints.

We now present the main contribution of the paper namely the Approximate Q Iteration (AQI) and Variational Approximate Q Iteration (VAQI) schemes. In particular, VAQI does not suffer from the aforementioned shortcomings of the ADP methods based on conventional LFAs.

IV. $(\min, +)$ LINEAR FUNCTIONS

The $\mathbf{R}_{\min}$ semiring is obtained by replacing multiplication $(\times)$ by $+$, and addition $(+)$ by $\min$.

Definition 1:

$$\text{Addition:} \qquad x \oplus y = \min(x, y).$$
$$\text{Multiplication:} \qquad x \otimes y = x + y.$$

Henceforth we use, $(+, \times)$ and $(\oplus, \otimes)$ to respectively denote the conventional and $\mathbf{R}_{\min}$ addition and multiplication respectively. A Semimodule over a semiring can be defined in a similar manner to vector spaces over fields. In particular we are interested in the semimodule $\mathcal{M} = \mathbf{R}_{\min}^n$ (since $J^* \in \mathbf{R}_{\min}^n$). Given $u, v \in \mathbf{R}_{\min}^n$, and $\lambda \in \mathbf{R}_{\min}$, we define addition and scalar multiplication as follows:

Definition 2:

$$(u \oplus v)(i) = \min\{u(i), v(i)\} = u(i) \oplus v(i), \forall i = 1, 2, \ldots, n.$$
$$(u \otimes \lambda)(i) = u(i) \otimes \lambda = u(i) + \lambda, \forall i = 1, 2, \ldots, n.$$

Similarly one can define the $\mathrm{R}_{\max}$ semiring which has the operators $\max$ as addition and $+$ as multiplication.

It is a well known fact that deterministic optimal control problems with cost/reward criterion are $(\min, +)$/$(\max, +)$ linear ([2], [3], [4], [5]). However, it is straightforward to show that the Bellman operator T (as well as H) in (2) corresponding to infinite horizon discounted reward MDPs is neither $(\min, +)$ linear nor $(\max, +)$ linear. Nevertheless, the motivation behind developing ADP methods based on $(\min, +)$ LFAs is to explore them as an alternative to the conventional basis representation.

Given a set of basis functions $\{\phi_i, i = 1, \ldots, k\}$, we define the $(\min, +)$ linear span to be $\mathcal{V} = \{v | v = \Phi \otimes r \stackrel{def}{=} \min(\phi_1 + r(1), \ldots, \phi_k + r(k)), r \in \mathbf{R}_{\min}^k\}$. Thus $\mathcal{V}$ is a subsemimodule. In the context of value function approximation, we would want to project quantities in $\mathbf{R}_{\min}^n$ onto $\mathcal{V}$. The $(\min, +)$ projection operator Π_M works as follows ([2], [8], [3]):

$$\Pi_M u = \min\{v | v \in \mathcal{V}, v \geq u\}, \forall u \in \mathcal{M}. \tag{14}$$

We can write the PBE in the $(\min,+)$ basis:

$$v = \Pi_M T v, v \in \mathcal{V}$$
$$\Phi \otimes r^* = \min\{\Phi \otimes r^* \in \mathcal{V} | \Phi \otimes r^* \geq T\Phi \otimes r^*\}. \quad (15)$$

Thus by making use of $(\min,+)$ LFAs and the projection operator Π_M we aim to find the minimum upper bound to J^*/Q^*.

V. Approximate Q Iteration

We now present an ADP scheme based on solving the PBE in $(\min,+)$ basis to compute $\tilde{Q}(\approx Q^*)$. Our ADP scheme successfully addresses the two shortcomings of the ADP scheme in conventional basis. First, we establish contraction of the projected Bellman operator in the L_∞ norm. This enables us to show that our recursion to compute $\tilde{Q}$ converges. We compute a greedy policy $\tilde{u}(s) = \max_a \tilde{Q}(s,a)$, and an approximate value function $\tilde{J}(s) = \max_a \tilde{Q}(s,a)$. Secondly, we also bound $||\tilde{Q} - Q^*||$ in the max norm which along with Lemma 4 gives a guarantee for $J_{\tilde{u}}$.

We are interested in solving the following PBE:

$$\Phi \otimes r^* = \Pi_M H \Phi \otimes r^*. \quad (16)$$

Since we want to approximate Q^*, Φ is an $nd \times k$ feature matrix, and $Q^*(s,a) \approx \tilde{Q}(s,a) = \phi^{(s-1)\times d + a} \otimes r^*$, where ϕ^i is the i^{th} row of Φ. The Approximate Q Iteration (AQI) algorithm is given by

$$\Phi \otimes r_{n+1} = \Pi_M H \Phi \otimes r_n. \quad (17)$$

The following results help us to establish the fact that the projected Bellman operator $\Pi_M H \colon \mathbf{R}_{\min}^{n\times d} \to \mathbf{R}_{\min}^{n\times d}$ is a contraction map in the L_∞ norm. In the discussion that follows, $\mathbf{1} \in \mathrm{R}^n \times d$ is a vector with all components equal to 1.

Lemma 7: For $Q_1, Q_2 \in \mathbf{R}_{\min}^{n\times d}$, such that $Q_1 \geq Q_2$, we have $\Pi_M Q_1 \geq \Pi_M Q_2$.

Proof: Follows from the definition of Π_M in (14). ■

Lemma 8: Let $Q \in \mathbf{R}_{\min}^{n\times d}$, $V_1 = \Pi_M Q$ be its projection onto $\mathcal{V}$. For $k \in \mathrm{R}$ the projection of $Q + k\mathbf{1}$ is $V_2 = \Pi_M Q + k\mathbf{1}$.

Proof: We know that since $V_1 \geq Q$, $V_1 + k\mathbf{1} \geq Q + k\mathbf{1}$, and from the definition of the Π_M in (14) $V_2 \leq V_1 + k\mathbf{1}$. Similarly $V_2 - k\mathbf{1} \geq Q$, so $V_1 \leq V_2 - k\mathbf{1}$. ■

Theorem 9: The projected Bellman operator $\Pi_M H$ is a contraction map in L_∞ norm with modulus α.

Proof: Let $Q_1, Q_2 \in \mathbf{R}_{\min}^{n\times d}$, and $\epsilon \stackrel{def}{=} ||Q_1 - Q_2||_\infty$, then by Lemma 7 we have

$$\begin{aligned}\Pi_M H Q_1 - \Pi_M H Q_2 &\leq \Pi_M H(Q_2 + \epsilon\mathbf{1}) - \Pi_M H Q_2 \\ &= \Pi_M(H Q_2 + \alpha\epsilon\mathbf{1}) - \Pi_M H Q_2 \\ &= \alpha\epsilon\mathbf{1}. \end{aligned} \quad (18)$$

Here the equality in second line is due to Lemma 3, and the final line follows from Lemma 8. Similarly $\Pi_M H Q_2 - \Pi_M H Q_1 \leq \alpha\epsilon\mathbf{1}$, so it follows that $||\Pi_M H Q_1 - \Pi_M H Q_2||_\infty \leq \alpha ||Q_1 - Q_2||_\infty$. ■

Corollary 3: AQI in (17) converges to a fixed point r^* such that $\Phi \otimes r^* = \Pi_M H \Phi \otimes r^*$.

VI. Variational Approximate Q Iteration

The projection operator Π_M used in (17) is exact. Let $v = \Pi_M u$, then for test vectors $w_i \in \mathbf{R}_{\min}^n, i = 1, \ldots, m$ it follows from the definition of Π_M that

$$w_i^\top v \geq w_i^\top u$$

where the dot product $x^\top y = \min_{i=1}^n (x(i) + y(i))$. Let W denote the $nd \times m$ test matrix whose columns are w_i. Now we shall define the approximate projection operator to be

$$\Pi_M^W u = \min\{v \in \mathcal{V} | W^\top v \geq W^\top u\}. \quad (19)$$

The superscript in Π_M^W denotes the test matrix W. The Variational Approximate Q Iteration (VAQI) is given by

$$\Phi \otimes r_{n+1} = \Pi_M^W H \Phi \otimes r_n. \quad (20)$$

Lemmas 7, 8, and Theorem 9 continue to hold if Π_M is replaced with Π_M^W. Thus by Corollary 3, we know that (20) converges to a unique fixed point r_W^* such that $\Phi \otimes r_W^* = \Pi_M^W H \Phi \otimes r_W^*$.

VII. Error Analysis

Theorem 10: Let $\tilde{r}$ be such that $\tilde{r} = \arg\min_r ||Q^* - \Phi \otimes r||_\infty$. Let r_W^* be the fixed point of the iterates in (20), then

$$\begin{aligned}||Q^* - \Phi \otimes r_W^*||_\infty \leq \frac{2}{1+\alpha}(&||Q^* - \Phi \otimes \tilde{r}||_\infty \\ &+ ||\Phi \otimes \tilde{r} - \Pi_M^W \Phi \otimes \tilde{r}||_\infty). \end{aligned} \quad (21)$$

Proof: Let $\epsilon = ||Q^* - \Phi \otimes \tilde{r}||_\infty$. By contraction property of H (Lemma 1) we know that

$$||HQ^* - H\Phi \otimes \tilde{r}||_\infty \leq \alpha\epsilon.$$

So we have $||\Phi \otimes \tilde{r} - H\Phi \otimes \tilde{r}||_\infty \leq (1+\alpha)\epsilon$. Then

$$\begin{aligned}||\Phi \otimes \tilde{r} - \Pi_M^W H \Phi \otimes \tilde{r}||_\infty &= ||\Phi \otimes \tilde{r} - \Pi_M^W \Phi \otimes \tilde{r}||_\infty \\ &\quad + ||\Pi_M^W \Phi \otimes \tilde{r} - \Pi_M^W H \Phi \otimes \tilde{r}||_\infty.\end{aligned}$$

$$\begin{aligned}\text{Now, } \Pi_M^W \Phi \otimes \tilde{r} - \Pi_M^W H \Phi \otimes \tilde{r} &\leq \Pi_M^W \Phi \otimes \tilde{r} \\ &\quad - \Pi_M^W(\Phi \otimes \tilde{r} - (1+\alpha)\epsilon) \\ &= (1+\alpha)\epsilon.\end{aligned}$$

Similarly $\Pi_M^W H \Phi \otimes \tilde{r} - \Pi_M^W \Phi \otimes \tilde{r} \leq (1+\alpha)\epsilon$, and hence

$$||\Phi \otimes \tilde{r} - \Pi_M^W H \Phi \otimes \tilde{r}||_\infty \leq (1+\alpha)\epsilon + \beta, \quad (22)$$

where $\beta = ||\Phi \otimes \tilde{r} - \Pi_M^W \Phi \otimes \tilde{r}||_\infty$. Now consider the iterative scheme in (20) with $r_0 = \tilde{r}$, and

$$\begin{aligned}||Q^* - \Phi \otimes r_W^*||_\infty &= ||Q^* - \Phi \otimes r_0 + \Phi \otimes r_0 \\ &\quad - \Phi \otimes r_1 + \ldots - \Phi \otimes r_W^*||_\infty \\ &\leq ||Q^* - \Phi \otimes r_0||_\infty + \\ &\quad ||\Phi \otimes r_0 - \Phi \otimes r_1||_\infty \\ &\quad + ||\Phi \otimes r_1 - \Phi \otimes r_2||_\infty + \ldots \\ &\leq \epsilon + (1+\alpha)\epsilon + \beta + \alpha((1+\alpha)\epsilon + \beta) \\ &\quad + \ldots \\ &= \epsilon(\frac{1+\alpha}{1-\alpha} + 1) + \frac{\beta}{1-\alpha} \\ &= \frac{2\epsilon + \beta}{1-\alpha}.\end{aligned}$$

■

The error bound for AQI is obtained by letting $\beta = 0$ in Theorem 10.

Corollary 4: Let r^* and r^*_W be the fixed points of iterations (17) and (20) respectively, and let $\tilde{Q} = \Phi \otimes r^*$, $\tilde{Q}_W = \Phi \otimes r^*_W$. The greedy policies $\tilde{u}(s) = \arg\max_a \tilde{Q}(s,a)$ and $\tilde{u}_W(s) = \arg\max_a \tilde{Q}_W(s,a)$ obey

$$||J_{\tilde{u}} - J^*|| \leq \frac{2(2\epsilon)}{(1-\alpha)^2}, ||J_{\tilde{u}_W} - J^*|| \leq \frac{2(2\epsilon+\beta)}{(1-\alpha)^2}, \quad (23)$$

where $\epsilon = \min_r ||Q^* - \Phi \otimes r||_\infty$.

Proof: Follows from Theorem 10, Lemma 4, and the observation that the term β is the error due to the usage of Π^W_M. Thus for solution to (16), $\beta = 0$. ■

VIII. Experiments

We test our ADP schemes on a randomly generated MDP with 100 states, i.e., $S = \{1, 2, \ldots, 100\}$, and action set $A = \{1, \ldots, 5\}$. The reward $g_a(s)$ is a random integer between 1 and 10, and we let the discount factor $\alpha = 0.9$. We now describe feature selection, where $\{\phi_j, j = 1, \ldots, k\}, \phi_j \in \mathbf{R}^{n\times d}_{\min}$ and $\{\phi^i, i = 1, \ldots, n \times d\}, \phi^i \in \mathbf{R}^k_{\min}$ denote the columns and rows respectively of the feature matrix Φ. The feature corresponding to a state-action pair (s,a) is given by $\phi^{(s-1)\times d+a}$. Let ϕ^x, ϕ^y be features corresponding to state-action pairs (s_x, a_x) and (s_y, a_y) respectively, then

$$< \phi^x, \phi^y >= \phi^x(1) \otimes \phi^y(1) \oplus \ldots \oplus \phi^x(k) \otimes \phi^y(k). \quad (24)$$

We desire the following in the feature matrix Φ.

1) Features ϕ^i should have unit norm, i.e., $||\phi^i|| =< \phi^i, \phi^i >= \mathbf{0}$, since $\mathbf{0}$ is the multiplicative identity in the $(\min, +)$ algebra.
2) For dissimilar state-action pairs, we prefer $< \phi^x, \phi^y >= +\infty$, since $+\infty$ is the additive identity in $(\min, +)$ algebra.

Keeping these in mind, we design the feature matrix Φ for the random MDP as in (25). For state-action pair (s,a) let $x = (s-1) \times d + a$, then the feature

$$\phi^x(i) = \begin{cases} 0 & : g_a(s) \in [g_{\min} + \frac{(i-1)L}{k}, g_{\min} + \frac{(i)L}{k}] \\ 1000 & : g_a(s) \notin [g_{\min} + \frac{(i-1)L}{k}, g_{\min} + \frac{(i)L}{k}], \end{cases} \quad \forall i = 1, \ldots, k. \quad (25)$$

We use 1000 in place of $+\infty$. Note that in (25) state-action pairs with similar rewards have similar features. The results are plotted in Fig. 1 and Fig. 2. Here J^* is the optimal value-function, $\tilde{J}_{EP}(s) = \max_a \tilde{Q}_{EP}(s,a)$, where $\tilde{Q}_{EP}$ is the value obtained via the iterative scheme in (17) and subscript EP denotes the fact that the projection employed in *exact* (Π_M). $\tilde{J}_W(s) = \max_a \tilde{Q}_W(s,a)$, where $\tilde{Q}_W$ is the value obtained via the iterative scheme in (20) and subscript W denotes the fact that the projection employed is Π^W_M. $u_{EP}(s) = \arg\max_a \tilde{Q}_{EP}(s,a)$ and $u_W(s) = \arg\max_a \tilde{Q}_W(s,a)$ are greedy policies and $J_{u_{EP}}$, J_{u_W} are their respective value functions. We also evaluated the performance of an *arbitrary* policy u_{arbt}, wherein we chose a fixed but arbitrary action for each state. We observed that $||J^* - J_{u_{arbt}}||_\infty = 40.5$.

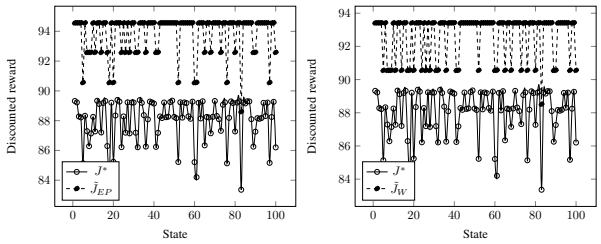

Fig. 1. $||J^* - \tilde{J}_{EP}||_\infty = 6.47, ||J^* - \tilde{J}_W||_\infty = 6.35$

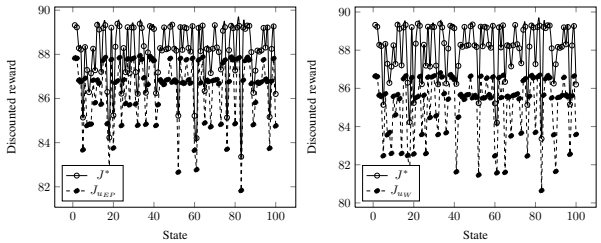

Fig. 2. $||J^* - J_{u_{EP}}||_\infty = 2.61, ||J^* - J_{u_W}||_\infty = 5.61$

IX. Conclusion

We developed two ADP schemes AQI and VAQI based on the $(\min, +)$ basis to compute an approximation to Q^*. The convergence of AQI and VAQI followed from the L_∞ norm contraction property of the projected Bellman operators $\Pi_M H$ and $\Pi^W_M H$. We also showed that the approximation error is bounded in the L_∞ norm (Theorem 10), and characterized the performance of the greedy policy (Corollary 4). The preliminary experiments on randomly generated MDPs show that AQI and VAQI converge in practice. We also observe that the policies computed by AQI and VAQI perform much better than arbitrary policies.

References

[1] D. P. de Farias and B. V. Roy, "The linear programming approach to approximate dynamic programming," *Operations Research*, vol. 51, no. 6, pp. 850–865, 2003.

[2] M. Akian, S. Gaubert, and A. Lakhoua, "The max-plus finite element method for solving deterministic optimal control problems: basic properties and convergence analysis," *SIAM Journal on Control and Optimization*, vol. 47, no. 2, pp. 817–848, 2008.

[3] W. M. McEneaney, A. Deshpande, and S. Gaubert, "Curse-of-complexity attenuation in the curse-of-dimensionality-free method for hjb pdes," in *American Control Conference, 2008*. IEEE, 2008, pp. 4684–4690.

[4] W. M. McEneaney and L. J. Kluberg, "Convergence rate for a curse-of-dimensionality-free method for a class of hjb pdes," *SIAM Journal on Control and Optimization*, vol. 48, no. 5, pp. 3052–3079, 2009.

[5] S. Gaubert, W. McEneaney, and Z. Qu, "Curse of dimensionality reduction in max-plus based approximation methods: Theoretical estimates and improved pruning algorithms," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. IEEE, 2011, pp. 1054–1061.

[6] M. Puterman, *Markov Decision Processes: Discrete Stochastic Programming*. New York: John Wiley, 1994.

[7] D. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Belmont,MA: Athena Scientific, 2007, vol. II.

[8] G. Cohen, S. Gaubert, and J.-P. Quadrat, "Kernels, images and projections in dioids," in *Proceedings of WODES96*, 1996, pp. 151–158.