Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Global Adaptive Dynamic Programming for Continuous-Time Nonlinear Polynomial Systems ⋆

Yu Jiang * **Zhong-Ping Jiang** *

* *Department of Electrical and Computer Engineering, Polytechnic School of Engineering, New York University, Brooklyn, NY 11201 USA (e-mail: yu.jiang@nyu.edu, zjiang@nyu.edu).*

Abstract: This paper presents a novel adaptive sub-optimal control method for continuous-time nonlinear polynomial systems from a perspective of adaptive dynamic programming (ADP). This is achieved by relaxing the problem of solving an Hamilton-Jacobi-Bellman (HJB) equation into an optimization problem, which is solved via a new policy iteration method. The proposed methodology distinguishes from previously known nonlinear ADP methods in that the neural network approximation is avoided and that the resultant control policy is globally stabilizing, instead of semiglobally or locally stabilizing. Furthermore, in the absence of the a prior knowledge of the system dynamics, an online learning method is devised to implement the proposed policy iteration technique by generalizing the current ADP theory. Finally, the proposed method is applied to a jet engine surge control problem.

1. INTRODUCTION

Adaptive/approximate dynamic programming (ADP) is a non-model-based and biologically-inspired method for computing online optimal control policies for uncertain systems. It was developed with the aim to avoid the two obstacles encountered in implementing classical dynamic programming [Bellman, 1957], namely, the so-called *curse of dimensionality* and the requirement on knowing the perfect system knowledge. The foundational work of ADP can be traced back to [Werbos, 1974]. ADP has been extensively studied for Markov decision processes [Bertsekas and Tsitsiklis, 1996, Powell, 2007], as well as dynamic systems [Lewis and Vrabie, 2009, Wang et al., 2009]. Stability issues regarding ADP when it is applied for dynamic systems are addressed by Balakrishnan et al. [2008], Vrabie et al. [2013]. A robustification of ADP, known as Robust-ADP or RADP, is recently developed by taking into account dynamic uncertainties [Jiang and Jiang, 2013].

To achieve online approximation of the cost function and the control policy, neural networks are widely used in the previous ADP architecture. Although neural networks can be used as universal approximators [Hornik et al., 1989, Park and Sandberg, 1991], they have at least two major limitations for ADP-based online implementations. First, a large number of basis functions comprising the neural network are usually required if high approximation accuracy is desired. Hence, it may incur a huge computational burden for the learning system. Besides, it is not trivial to specify the type of basis functions, when the target function to be approximated is unknown. Second, neural network approximations generally are effective only on some compact sets, but not in the entire state space. Therefore, the resultant control policy may not provide global asymptotic stability for the closed-loop system. In addition, the compact set, on which the uncertain functions of interest are to be approximated, has to be carefully quantified before one applies the online learning method, such that stability can be assured during the learning process.

The main purpose of this paper is to develop a novel ADP methodology that not only finds online a suboptimal control policy for uncertain continuous-time nonlinear polynomial systems, but at the same time guarantees the global asymptotic stability. Our main contribution is threefold. First, we relax the problem of solving an Hamilton-Jacobi-Bellman (HJB) equation into an optimization problem, of which each feasible solution provides a sub-optimal and globally stabilizing control policy. Second, a novel policy iteration method is proposed to find a local minimum of the above-mentioned optimization problem. Third, we develop a way in which the proposed policy iteration can be implemented online, when the system dynamics is not perfectly known. Proofs of lemmas and theorems in this paper are omitted due to space limitation, but they can be found in [Jiang and Jiang, 2014].

Notations: Throughout this paper, we use $\mathcal{C}^1$ to denote the set of all continuously differentiable functions. $\mathcal{P}$ denotes the set of all functions in $\mathcal{C}^1$ that are also positive definite and proper. $\mathbb{R}_+$ indicates the set of all non-negative real numbers. For any vector $u \in \mathbb{R}^m$ and any positive definite matrix $R \in \mathbb{R}^{m\times m}$, we define $|u|_R^2$ as $u^T R u$. A feedback control policy u is said to be globally stabilizing, if under this control policy, the closed-loop system is globally asymptotically stable (GAS) at the origin [Khalil, 2002]. For any non-negative integers d_1, d_2 satisfying $d_2 \geq d_1$, $[x]_{d_1,d_2}$ is the vector of all $\binom{n+d_2}{d_2} - \binom{n+d_1}{d_1}$ distinct monic monomials in $x \in \mathbb{R}^n$ with degree no less than d_1 and no greater than d_2, and arranged in lexicographic order [Cox,

⋆ This work has been supported in part by the National Science Foundation, under Grants ECCS-1101401 and ECCS-1230040.

2007]. Also, $\mathbb{R}[x]_{d_1,d_2}$ denotes the set of all polynomials in $x \in \mathbb{R}^n$ with degree no less than d_1 and no greater than d_2. In addition, $\mathbb{R}[x]^m_{d_1,d_2}$ denotes the set of m-dimensional vectors, of which each entry is a polynomial in $\mathbb{R}[x]_{d_1,d_2}$. ∇V refers to the gradient of a differentiable function $V : \mathbb{R}^n \to \mathbb{R}$.

2. PROBLEM FORMULATION AND PRELIMINARIES

2.1 Problem formulation

Consider the nonlinear system

$$\dot{x} = f(x) + g(x)u \tag{1}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the control input, $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are polynomial mappings with $f(0) = 0$.

In conventional optimal control theory [Lewis et al., 2012], the common objective is to find a control policy u that minimizes certain performance index. In this paper, the performance index to be minimized is given by

$$J(x_0, u) = \int_0^\infty r(x(t), u(t))dt, \quad x(0) = x_0 \tag{2}$$

where $r(x, u) = Q(x)+u^T Ru$, with $Q(x)$ a positive definite function, and R is a symmetric positive definite matrix.

Assumption 2.1. Consider system (1). There exist $V_0 \in \mathcal{P}$ and $u_1 : \mathbb{R}^n \to \mathbb{R}^m$, such that

$$\mathcal{L}(V_0(x), u_1(x)) \geq 0, \quad \forall x \in \mathbb{R}^n \tag{3}$$

where, for any $V \in \mathcal{C}^1$ and $u \in \mathbb{R}^m$,

$$\mathcal{L}(V, u) = -\nabla V^T(x)(f(x) + g(x)u) - r(x, u). \tag{4}$$

Under Assumption 2.1, the closed-loop system composed of (1) and $u = u_1(x)$ is GAS at the origin, with a well-defined Lyapunov function V_0. Further, u_1 is also known as an *admissible* control policy [Beard et al., 1997], since it is easy to show $J(x_0, u_1) \leq V_0(x_0), \quad \forall x_0 \in \mathbb{R}^n$.

2.2 Optimality and stability

Here, we recall a basic result connecting optimality and global asymptotic stability in nonlinear systems [Sepulchre et al., 1997]. First, let us give the following assumption.

Assumption 2.2. There exists $V^o \in \mathcal{P}$, such that the Hamilton-Jacobi-Bellman (HJB) equation holds

$$\mathcal{H}(V^o) = 0 \tag{5}$$

where

$$\mathcal{H}(V) = \nabla V^T(x)f(x) + Q(x) - \frac{1}{4}\nabla V^T(x)g(x)R^{-1}g^T(x)\nabla V(x).$$

Under Assumption 2.2, it is easy to see that V^o is a well-defined Lyapunov function for the closed-loop system comprised of (1) and the following control law

$$u^o(x) = -\frac{1}{2}R^{-1}g^T(x)\nabla V^o(x). \tag{6}$$

Hence, this closed-loop system is GAS at $x = 0$ [Khalil, 2002]. Then, according to [Sepulchre et al., 1997, Theorem 3.19], u^o is the optimal control policy, and the value function $V^o(x_0)$ gives the optimal cost at the initial condition $x(0) = x_0$, i.e.,

$$V^o(x_0) = \min_u J(x_0, u) = J(x_0, u^o), \quad \forall x_0 \in \mathbb{R}^n. \tag{7}$$

2.3 Conventional policy iteration

The nonlinear HJB equation (5) is almost impossible to be solved analytically in general. As a result, numerical methods are developed to approximate the solution. In particular, the following policy iteration method is widely used [Saridis and Lee, 1979].

1) *Initialization* Find u_1 that satisfies Assumption 2.1.
2) *Policy evaluation:* For $i = 1, 2, \cdots$, solve for the cost function $V_i(x) \in \mathcal{C}^1$, with $V_i(0) = 0$, from the following partial differential equation.
$$\mathcal{L}(V_i(x), u_i(x)) = 0. \tag{8}$$
3) *Policy improvement:* Update the control policy by
$$u_{i+1}(x) = -\frac{1}{2}R^{-1}g^T(x)\nabla V_i(x). \tag{9}$$

The following result is a direct extension of [Saridis and Lee, 1979, Theorem 4], in which $g(x)$ is a constant matrix and only stabilization over compact set is considered.

Theorem 2.1. Suppose Assumptions 2.1 and 2.2 hold, and the solution $V_i(x) \in \mathcal{C}^1$ satisfying (8) exists, for $i = 1, 2, \cdots$. Let $V_i(x)$ and $u_{i+1}(x)$ be the functions generated from (8) and (9). Then, the following properties hold, $\forall i = 0, 1, \cdots$.

1) $V^o(x) \leq V_{i+1}(x) \leq V_i(x)$, $\forall x \in \mathbb{R}^n$;
2) u_{i+1} is globally stabilizing;
3) $J(x_0, u_i)$ is finite, $\forall x_0 \in \mathbb{R}^n$;
4) $\{V_i(x)\}_{i=1}^\infty$ and $\{u_i(x)\}_{i=1}^\infty$ converge pointwise to $V^o(x)$ and $u^o(x)$, respectively.

3. SUBOPTIMAL CONTROL WITH RELAXED HJB EQUATION

In this section, we consider an auxiliary optimization problem, which allows us to obtain a suboptimal solution to the minimization problem (2) subject to (1). For simplicity, we will omit the arguments of functions whenever there is no confusion in the context.

Problem 3.1. (Relaxed optimal control problem).

$$\min_V \int_{\mathbb{R}^n} w(x)V(x)dx \tag{10}$$
$$\text{s.t. } \mathcal{H}(V) \leq 0 \tag{11}$$
$$V \in \mathcal{P} \tag{12}$$

where $w(x)$, also recognized as the *state-relevance* weighting function [de Farias and Van Roy, 2003], is a positive semidefinite function taking positive values only on some predefined compact set $\Omega \subset \mathbb{R}^n$.

Theorem 3.1. Under Assumptions 2.1 and 2.2, the following properties hold.

1) Problem 3.1 has a nonempty feasible set.
2) Let V be a feasible solution to Problem 3.1. Then, $\bar{u} = -\frac{1}{2}R^{-1}g^T\nabla V$ is globally stabilizing.
3) For any $x_0 \in \mathbb{R}^n$, an upper bound of the cost of the closed-loop system comprised of (1) and $\bar{u}$ is given by $V(x_0)$, i.e., $J(x_0, \bar{u}) \leq V(x_0)$.
4) Along the trajectories of the closed-loop system (1) and $\bar{u}$, the following inequalities hold for any $x_0 \in \mathbb{R}^n$:

$$V(x_0) + \int_0^\infty \mathcal{H}(V(x(t)))dt \leq V^{\mathrm{o}}(x_0) \leq V(x_0). \quad (13)$$

5) V^{o} as defined in (7) is a global optimal solution to Problem 3.1.

Remark 3.1. A feasible solution V to Problem 3.1 may not necessarily be the true cost function associated with the control policy $\bar{u}$. However, by Theorem 3.1, we see V can be viewed as an upper bound or an *overestimate* of the actual cost, inspired by the concept of *underestimator* [Wang and Boyd, 2010]. Further, V serves as a Lyapunov function for the closed-loop system and can be more easily parameterized than the actual cost function. For simplicity, V is still called the *cost function*, in the remainder of the paper.

4. SOS-BASED POLICY ITERATION FOR POLYNOMIAL SYSTEMS

The inequality constraint (11) contained in Problem 3.1 provides us the freedom of specifying desired analytical forms of the cost function. However, solving (11) is nontrivial in general. Fortunately, due to the developments in sum of squares (SOS) programming [Blekherman et al., 2013, Parrilo, 2000], the computational burden can be significantly reduced, if inequality constraints can be restricted to SOS constraints. The purpose of this section is to develop a novel policy iteration method for polynomial systems using SOS-based methods [Blekherman et al., 2013, Parrilo, 2000].

4.1 Polynomial parametrization

Assumption 4.1. There exist integers $d > 0$, $d_1 \geq 0$, and $r > 0$, such that

(1) all entries of $f(x)$ belong to $\mathbb{R}[x]_{1,d}$ and all entries of $g(x)$ belong to $\mathbb{R}[x]_{0,d_1}$;
(2) in addition to being positive definite, the weighting function $Q(x)$ satisfies $Q(x) \in \mathbb{R}[x]_{2,2d}$;
(3) there exist $V_0 : \mathbb{R}^n \to \mathbb{R}$ and $u_1 : \mathbb{R}^n \to \mathbb{R}^m$, such that $V_0 \in \mathbb{R}[x]_{2,2r} \cap \mathcal{P}$, $u_1 \in \mathbb{R}[x]_{1,d}^m$, and $\mathcal{L}(V_0, u_1)$ is SOS; and
(4) the inequality holds:

$$d \geq (2r-1) + d_1. \quad (14)$$

Remark 4.1. It is easy to see that, Assumption 4.1 holds only if Assumption 2.1 holds. In addition, under Assumption 4.1, we know that $\mathcal{L}(V_0, u_1) \in \mathbb{R}[x]_{2,2d}$. Indeed, by (14), it follows that

$$\mathcal{L}(V_0, u_1) \in \mathbb{R}[x]_{2,\max\{(2r-1)+d+(d_1+d),2d\}} = \mathbb{R}[x]_{2,2d}.$$

Remark 4.2. Notice that the inequality (14) can be assumed without loss of generality. Indeed, if it does not hold, we can always find $\tilde{d} > \max\{d, (2r-1)+d_1\}$. As a result, Assumption 4.1 holds with d replaced by $\tilde{d}$.

For notational simplicity, we denote the dimensions of $[x]_{1,r}$, $[x]_{1,d}$, $[x]_{2,2r}$, and $[x]_{2,2d}$ by n_r, n_d, n_{2r}, and n_{2d}, respectively. By Blekherman et al. [2013], we know $n_r = \binom{n+r}{r} - 1$, $n_d = \binom{n+d}{d} - 1$, $n_{2r} = \binom{n+2r}{2r} - n - 1$, and $n_{2d} = \binom{n+2d}{2d} - d - 1$.

4.2 SOS-programming-based policy iteration

Now, we are ready to propose a relaxed policy iteration scheme. Similar as in other policy-iteration-based iterative schemes, an initial globally stabilizing (and admissible) control policy has been assumed in Assumption 4.1.

1) *Policy evaluation:* For $i = 1, 2, \cdots$, solve for an optimal solution $p_i \in \mathbb{R}^{n_{2r}}$ to the following optimization program, and denote $V_i = p_i^T[x]_{2,2r}$.

$$\min_{p \in \mathbb{R}^{n_{2r}}} \quad \int_{\mathbb{R}^n} w(x)V(x)dx \quad (15)$$

$$\text{s.t.} \quad V := p^T[x]_{2,2r} \quad (16)$$

$$\mathcal{L}(V, u_i) \in \Sigma_{2,2d} \quad (17)$$

$$V_{i-1} - V \in \Sigma_{2,2r} \quad (18)$$

where $\Sigma_{2,2d}$ and $\Sigma_{2,2r}$ denote the sets of all SOS polynomials in $\mathbb{R}[x]_{2,2d}$ and $\mathbb{R}[x]_{2,2r}$, respectively.
2) *Policy improvement:* Update the control policy by

$$u_{i+1} = -\frac{1}{2}R^{-1}g^T\nabla V_i. \quad (19)$$

Then, go to Step 1) with $i \leftarrow i + 1$.

Remark 4.3. The optimization problem (15)-(18) is a well defined SOS program [Blekherman et al., 2013]. Indeed, the objective function (15) is linear in p, since for any $V = p^T[x]_{2,2r}$, we have $\int_{\mathbb{R}^n} w(x)V(x)dx = c^T p$, with $c = \int_{\mathbb{R}^n} w(x)[x]_{2,2r}dx$. In addition, notice that since the objective function is nonnegative, its optimal value must be finite.

Theorem 4.1. Under Assumptions 2.2 and 4.1, the following are true, for $i = 1, 2, \cdots$.

1) The SOS program (15)-(18) has a feasible solution.
2) The closed-loop system comprised of (1) and $u = u_i$ is GAS at the origin.
3) $V_i \in \mathcal{P}$. In particular, the following inequalities hold:

$$V^{\mathrm{o}}(x_0) \leq V_i(x_0) \leq V_{i-1}(x_0), \quad \forall x_0 \in \mathbb{R}^n. \quad (20)$$

4) There exists $V^*(x)$ satisfying $V^*(x) \in \mathbb{R}[x]_{2,2r} \cap \mathcal{P}$, such that, for any $x_0 \in \mathbb{R}^n$, $\lim_{i\to\infty} V_i(x_0) = V^*(x_0)$.
5) Along the solutions of the system (1) with

$$u^* = -\frac{1}{2}R^{-1}g^T\nabla V^*, \quad (21)$$

the following inequalities hold:

$$0 \leq V^*(x_0) - V^{\mathrm{o}}(x_0) \leq -\int_0^\infty \mathcal{H}(V^*(x(t)))dt. \quad (22)$$

4.3 An equivalent SDP implementation

In [Blekherman et al., 2013], it has been shown that any SOS program can be reformulated as a semidefinite program (SDP) [Vandenberghe and Boyd, 1996]. Indeed, we can always find two linear mappings $\iota : \mathbb{R}^{n_{2r}} \times \mathbb{R}^{m\times n_r} \to \mathbb{R}^{n_{2d}}$ and $\kappa : \mathbb{R}^{n_{2r}} \to \mathbb{R}^{m\times n_r}$, such that given $p \in \mathbb{R}^{n_{2r}}$ and $K \in \mathbb{R}^{m\times n_r}$,

$$\iota(p,K)^T[x]_{2,2d} = \mathcal{L}(p^T[x]_{2,2r}, K[x]_{1,2r-1}) \tag{23}$$

$$\kappa(p)^T[x]_{1,2r-1} = -\frac{1}{2}R^{-1}g^T\nabla(p^T[x]_{2,2r}) \tag{24}$$

Then, by properties of SOS constraints [Blekherman et al., 2013], the polynomial $\iota(p,K)^T[x]_{2,2d}$ is SOS if and only if there exists a symmetric and positive semidefinite matrix $L \in \mathbb{R}^{n_d\times n_d}$, such that

$$\iota(p,K)^T[x]_{2,2d} = [x]_{1,d}^T L[x]_{1,d}. \tag{25}$$

Furthermore, there exist linear mappings $M_P : \mathbb{R}^{n_r\times n_r} \to \mathbb{R}^{n_{2r}}$ and $M_L : \mathbb{R}^{n_d\times n_d} \to \mathbb{R}^{n_{2d}}$, such that, for any vectors $p \in \mathbb{R}^{n_{2r}}$, $l \in \mathbb{R}^{n_{2d}}$, and symmetric matrices $P \in \mathbb{R}^{n_r\times n_r}$ and $L \in \mathbb{R}^{n_d\times n_d}$, the following implications are true.

$$p^T[x]_{2,2r} = [x]_{1,r}^T P[x]_{1,r} \Longleftrightarrow p = M_P(P) \tag{26}$$

$$l^T[x]_{2,2d} = [x]_{1,d}^T L[x]_{1,d} \Longleftrightarrow l = M_L(L) \tag{27}$$

Under Assumptions 2.2 and 4.1, the proposed policy iteration can be reformulated as follows.

1) Let $i = 1$. Let $p_0 \in \mathbb{R}^{n_{2r}}$ and $K_1 \in \mathbb{R}^{m\times n_d}$ satisfy $V_0 = p_0^T[x]_{2,2r}$ and $u_1 = K_1[x]_{1,d}$.
2) Solve for an optimal solution $(p_i, P_i, L_i) \in \mathbb{R}^{n_{2r}} \times \mathbb{R}^{n_r\times n_r} \times \mathbb{R}^{n_d\times n_d}$ to the following problem.

$$\min_{p,P,L} \quad c^T p \tag{28}$$

$$\text{s.t.} \quad \iota(p,K_i) = M_L(L) \tag{29}$$

$$p_{i-1} - p = M_P(P) \tag{30}$$

$$P = P^T \geq 0 \tag{31}$$

$$L = L^T \geq 0 \tag{32}$$

where $c = \int_{\mathbb{R}^n} w(x)[x]_{2,2r}dx$.
3) Go to Step 2) with $K_{i+1} = \kappa(p_i)$ and $i \leftarrow i+1$.

Remark 4.4. The optimization problem (28)-(32) is a well-defined SDP problem, since it has a linear objective function subject to linear equality and inequality constraints. It can be directly solved using, for example, Matlab-based solver CVX [Grant and Boyd, 2013].

Corollary 4.1. Under Assumptions 2.2 and 4.1, the following are true.

(1) The optimization problem (28)-(32) has at least one feasible solution, for $i = 1, 2, \cdots$.
(2) Denote $V_i = p_i^T[x]_{2,2r}$, $u_{i+1} = K_i[x]_{1,d}$, for $i = 0, 1, \cdots$. Then, the sequences $\{V_i\}_{i=0}^{\infty}$ and $\{u_i\}_{i=1}^{\infty}$ satisfy the properties 2)-5) in Theorem 4.1.

5. GLOBAL ADAPTIVE DYNAMIC PROGRAMMING FOR UNCERTAIN POLYNOMIAL SYSTEMS

The proposed policy iteration method requires the perfect knowledge of the mappings ι and κ, which can be determined if f and g are known exactly. In practice, precise system knowledge may be difficult to obtain. Hence, in this section, we develop an online learning method based on the idea of ADP to implement the iterative scheme with real-time data, instead of identifying the system dynamics.

To begin with, consider the system

$$\dot{x} = f + g(u_i + e) \tag{33}$$

where u_i is a feedback control policy and e is a bounded time-varying function, known as the exploration noise, added for the learning purpose.

Lemma 5.1. Consider system (33). Suppose u_i is a globally stabilizing control policy and there exists $V_{i-1} \in \mathcal{P}$, such that $\nabla V_{i-1}^T(f + gu_i) + u_i^T R u_i \leq 0$. Then, the system (33) is forward complete.

By Lemma 5.1 and Theorem 4.1, we immediately have the following Proposition.

Proposition 5.1. Under Assumptions 2.2 and 4.1, let u_i be a feedback control policy obtained at the i-th iteration step in the proposed policy iteration algorithm (15)-(19) and e be a bounded time-varying function. Then, the closed-loop system (1) with $u = u_i + e$ is forward complete.

Suppose there exist $p \in \mathbb{R}^{n_{2r}}$ and $K_i \in \mathbb{R}^{m\times n_k}$ such that $V = p^T[x]_{2,2r}$ and $u_i = K_i[x]_{1,d}$. Then, along the solutions of the system (33), it follows that

$$\begin{aligned}\dot{V} &= \nabla V^T(f + gu_i) + \nabla V^T ge \\ &= -r(x,u_i) - \mathcal{L}(V,u_i) + 2(\frac{1}{2}R^{-1}g^T\nabla V)^T Re \\ &= -r(x,u_i) - \iota(p,K_i)^T[x]_{2,2d} - 2[x]_{1,d}^T\kappa(p)^T Re \end{aligned} \tag{34}$$

where the last row is obtained by (23) and (24).

Now, integrating the terms in (34) over the interval $[t, t+\delta t]$, we have

$$\begin{aligned} & p^T\left([x(t)]_{2,2r} - [x(t+\delta t)]_{2,2r}\right) \\ &= \int_t^{t+\delta t} \left[r(x,u_i) + \iota(p,K_i)^T[x]_{2,2d} \right. \\ &\quad \left. + 2[x]_{1,d}^T\kappa(p)^T Re\right] dt \end{aligned} \tag{35}$$

Eq. (35) implies that, given $p \in \mathbb{R}^{n_{2r}}$, $\iota(p,K_i)$ and $\kappa(p)$ can be directly calculated by using real-time online data, without knowing the precise knowledge of f and g.

Indeed, define

$$\sigma_e = -\left[\, [x]_{2,2d}^T \;\; 2[x]_{1,d}^T \otimes e^T R \,\right]^T \in \mathbb{R}^{n_{2d}+mn_d},$$

$$\Phi_i = \left[\int_{t_{0,i}}^{t_{1,i}} \sigma_e dt \;\; \int_{t_{1,i}}^{t_{2,i}} \sigma_e dt \;\cdots\; \int_{t_{q_i-1,i}}^{t_{q_i,i}} \sigma_e dt\right]^T \in \mathbb{R}^{q_i\times(n_{2d}+mn_d)},$$

$$\Xi_i = \left[\int_{t_{0,i}}^{t_{1,i}} r(x,u_i)dt \;\; \int_{t_{1,i}}^{t_{2,i}} r(x,u_i)dt \;\cdots\; \int_{t_{q_i-1,i}}^{t_{q_i,i}} r(x,u_i)dt\right]^T \in \mathbb{R}^{q_i},$$

$$\Theta_i = \left[\, [x]_{2,2r}|_{t_{0,i}}^{t_{1,i}} \;\; [x]_{2,2r}|_{t_{1,i}}^{t_{2,i}} \;\cdots\; [x]_{2,2r}|_{t_{q_i-1,i}}^{t_{q_i,i}} \,\right]^T \in \mathbb{R}^{q_i\times n_{2r}}.$$

Then, (35) implies

$$\Phi_i \begin{bmatrix} \iota(p, K_i) \\ \text{vec}(\kappa(p)) \end{bmatrix} = \Xi_i + \Theta_i p. \tag{36}$$

Assumption 5.1. For each $i = 1, 2, \cdots$, there exists an integer q_{i0}, such that when $q_i \geq q_{i0}$ the following rank condition holds.

$$\text{rank}(\Phi_i) = n_{2d} + mn_d. \tag{37}$$

Remark 5.1. The rank condition (37) is in the spirit of persistency of excitation (PE) in adaptive control [*e.g.* Ioannou and Sun, 1996, Tao, 2003] and is a necessary condition for parameter convergence.

Given $p \in \mathbb{R}^{n_{2r}}$ and $K_i \in \mathbb{R}^{m \times n_d}$, suppose Assumption 5.1 is satisfied and $q_i \geq q_{i0}$ for all $i = 1, 2, \cdots$. Then, it is easy to see that the values of $\iota(p, K_i)$ and $\kappa(p)$ can be uniquely determined from

$$\begin{bmatrix} \iota(p, K_i) \\ \text{vec}(\kappa(p)) \end{bmatrix} = \left(\Phi_i^T \Phi_i\right)^{-1} \Phi_i^T \left(\Xi_i + \Theta_i p\right) \tag{38}$$

Now, we are ready to develop the ADP-based online implementation algorithm for the proposed policy iteration method.

1) *Initialization:*
   Let p_0 be the constant vector such that $V_0 = p_0^T [x]_{2,2r}$, and let $i = 1$.
2) *Collect online data:*
   Apply $u = u_i + e$ to the system and compute the data matrices Φ_i, Ξ_i, and Θ_i, until the rank condition (37) in Assumption 5.1 is satisfied.
3) *Policy evaluation and improvement:*
   Find an optimal solution (p_i, K_{i+1}, P_i, L_i) to the following optimization problem

$$\min_{p,K,P,L} \quad c^T p \tag{39}$$

$$\text{s.t.} \quad \begin{bmatrix} M_L(L) \\ \text{vec}(K) \end{bmatrix} = \left(\Phi_i^T \Phi_i\right)^{-1} \Phi_i^T \left(\Xi_i + \Theta_i p\right) \tag{40}$$

$$p_{i-1} - p = M_P(P) \tag{41}$$

$$P = P^T \geq 0 \tag{42}$$

$$L = L^T \geq 0 \tag{43}$$

   Then, denote $V_i = p_i^T [x]_{2,2r}$, $u_{i+1} = K_{i+1}[x]_{1,d}$, and go to Step 2) with $i \leftarrow i + 1$.

Lemma 5.2. Under Assumption 5.1, (p_i, K_{i+1}, P_i, L_i) is an optimal solution to the optimization problem (39)-(43) if and only if (p_i, P_i, L_i) is an optimal solution to the optimization problem (28)-(32) and $K_{i+1} = \kappa(p_i)$.

Theorem 5.1. Under Assumptions 2.1, 4.1 and 5.1, the following properties hold.

(1) The optimization problem (39)-(43) has a nonempty feasible set.
(2) The sequences $\{V_i\}_{i=1}^{\infty}$ and $\{u_i\}_{i=1}^{\infty}$ satisfy the properties 2)-5) in Theorem 4.1.

Remark 5.2. Notice that both V_0 and u_1 satisfying Assumption 4.1 have to be determined without knowing exactly f and g. In practice, we can find polynomial mappings $\bar{f}, \underline{f}, \bar{g}, \underline{g}$, such that $\underline{f} \leq f \leq \bar{f}$ and $\underline{g} \leq g \leq \bar{g}$. Thus, it is possible to find u_1 by using robust nonlinear control methods [Krstic et al., 1995, Taware and Tao, 2003]. Then, we solve V_0 from the following robust feasibility problem in SOS programming

$$-\nabla V_0^T (\tilde{f} + \tilde{g} u_1) - Q - u_1^T R u_1 \in \Sigma_{2,2d}, \tag{44}$$

for all $\tilde{f}$ and $\tilde{g}$ such that $\underline{f} \leq \tilde{f} \leq \bar{f}$ and $\underline{g} \leq \tilde{g} \leq \bar{g}$. This problem, if solvable, can be converted into a robust linear matrix inequality and efficiently solved using MATLAB-based solvers, such as the LMI control toolbox [Gahinet et al., 1994] or CVX [Grant and Boyd, 2013].

6. APPLICATION

Consider the following model of jet engine surge dynamics [Greitzer, 1976, Krstic et al., 1998].

$$\dot{x}_1 = -x_2 - \frac{3}{2} x_1^2 - \frac{1}{2} x_1^3 \tag{45}$$

$$\dot{x}_2 = \frac{1}{\beta^2} u \tag{46}$$

where x_1 and x_2 represent the scaled annulus-averaged flow and plenum pressure rise in error coordinates, respectively. u is the control input, and the constant β is assumed to be unknown belonging to $[0.7, 0.9]$. The cost is specified as $J(x_0, u) = \int_0^{\infty} (0.1 x_1^2 + x_2^2 + 0.1 u^2) dt$.

Using the technique in Krstic et al. [1998], we are able to find an initial stabilizing control policy $u_1 = 50x_1 - 2x_2$, and a related cost function V_0 satisfying Assumption 4.1 is obtained by solving the feasibility problem (44), with $r = 2$, $d = 3$, and $d_1 = 0$. For simulation, select $x_1(0) = 3$ and $x_2(0) = -4$.

The proposed online learning scheme is applied to improve the control policy every one second for four times. In this simulation, we set $\beta = 0.8$, which is assumed to be unknown to the learning system. The exploration noise is the sum of 25 sinusoidal waves with different frequencies, and it is turned off after four iterations. Simulation results are shown in Figures 1 and 2. It can be seen that the post-learning cost function is remarkably improved compared with the one obtained in the first policy evaluation step.

7. CONCLUSIONS

In this paper, a global ADP method for continuous-time nonlinear polynomial systems has been proposed for the first time. This method solves online an optimization problem which is a relaxation of the problem of solving HJBs. A novel policy iteration technique has been developed. Different from policy iteration methods in the past literature, this new iterative technique does not attempt to solve a partial differential equation at each iteration step. Instead, it solves a semi-definite programming problem. Compared with neural-network-based ADP schemes, the proposed method departs from the approximation technique using a large number of basis functions and hopefully yields significant computational benefit. In addition, the resultant control policy is globally stabilizing, while the previously known neural-networks-based ADP methods only yield semi-globally or locally stabilizing controllers.

It will be interesting to extend the proposed methodology for more general (deterministic or stochastic) nonlinear systems. Some preliminary work has been accomplished by Jiang and Jiang [2014].

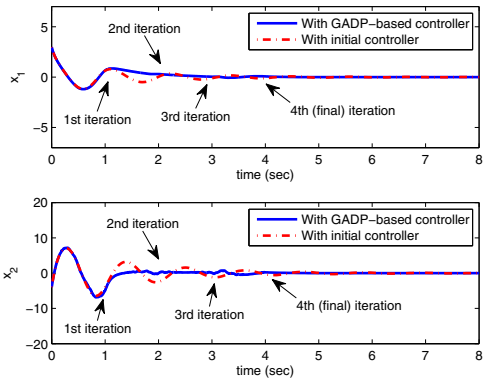

Fig. 1. Trajectories of the state variables

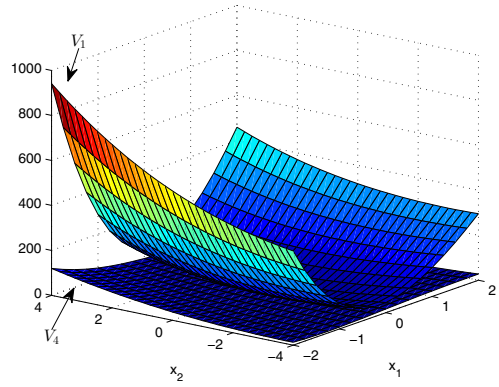

Fig. 2. Comparison of the cost functions

REFERENCES

S. N. Balakrishnan, J. Ding, and F. L. Lewis. Issues on stability of ADP feedback controllers for dynamical systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 38(4):913–917, 2008.

R. W. Beard, G. N. Saridis, and J. T. Wen. Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation. *Automatica*, 33(12):2159–2177, 1997.

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

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Nashua, NH, 1996.

G. Blekherman, P. A. Parrilo, and R. R. Thomas, editors. *Semidefinite Optimization and Convex Algebraic Geometry*. SIAM, Philadelphia, PA, 2013.

D. A. Cox. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2007.

D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

P. Gahinet, A. Nemirovskii, A. J. Laub, and M. Chilali. The LMI control toolbox. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, volume 3, pages 2038–2041, 1994.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.0 beta. *http://cvxr.com/cvx*, 2013.

E. M. Greitzer. Surge and rotating stall in axial flow compressors, Parts I and II. *Journal of Engineering for Power*, 98:190–217, 1976.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

P. A. Ioannou and J. Sun. *Robust Adaptive Control*. Prentice-Hall, Upper Saddle River, NJ, 1996.

Y. Jiang and Z. P. Jiang. Global adaptive dynamic programming for continuous-time nonlinear systems. *ArXiv:1401.0020 [math.DS]*, 2014. URL `http://arxiv.org/pdf/1401.0020.pdf`.

Z. P. Jiang and Y. Jiang. Robust adaptive dynamic programming for linear and nonlinear systems: An overview. *European Journal of Control*, 19(5):417–425, 2013.

H. K. Khalil. *Nonlinear Systems, 3rd Edition*. Prentice Hall, Upper Saddle River, NJ, 2002.

M. Krstic, I. Kanellakopoulos, and P. V. Kokotovic. *Nonlinear and Adaptive Control Design*. Wiley, NY, 1995.

M. Krstic, D. Fontaine, P. V. Kokotovic, and J. D. Paduano. Useful nonlinearities and global stabilization of bifurcations in a model of jet engine surge and stall. *IEEE Trans. Auto. Contr.*, 43(12):1739–1745, 1998.

F. L. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

F. L. Lewis, D. Vrabie, and V. L. Syrmos. *Optimal Control, 3rd ed.* Wiley, New York, 2012.

J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.

P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, Pasadena, CA, 2000.

W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, New York, 2007.

G. N. Saridis and C.-S. G. Lee. An approximation theory of optimal control for trainable manipulators. *IEEE Transactions on Systems, Man and Cybernetics*, 9(3): 152–159, 1979.

R. Sepulchre, M. Jankovic, and P. Kokotovic. *Constructive Nonlinear Control*. Springer Verlag, New York, 1997.

G. Tao. *Adaptive Control Design and Analysis*. Wiley, 2003.

A. Taware and G. Tao. *Control of Sandwich Nonlinear Systems*. Springer, 2003.

L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

D. Vrabie, K. G. Vamvoudakis, and F. L. Lewis. *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*. The Institution of Engineering and Technology, London, UK, 2013.

F.-Y. Wang, H. Zhang, and D. Liu. Adaptive dynamic programming: an introduction. *IEEE Computational Intelligence Magazine*, 4(2):39–47, 2009.

Y. Wang and S. Boyd. Approximate dynamic programming via iterated Bellman inequalities. *Manuscript preprint*, 2010.

P. Werbos. *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard Univ. Comm. Appl. Math., 1974.