Invariant Adaptive Dynamic Programming for Discrete-Time Optimal Control

Yuanheng Zhu, *Member, IEEE*, Dongbin Zhao, *Senior Member, IEEE*, and Haibo He, *Fellow, IEEE*

***Abstract*—For systems that can only be locally stabilized, control laws and their effective regions are both important. In this paper, invariant policy iteration is proposed to solve the optimal control of discrete-time systems. At each iteration, a given policy is evaluated in its invariantly admissible region, and a new policy and a new region are updated for the next iteration. Theoretical analysis shows the method is regionally convergent to the optimal value and the optimal policy. Combined with sum-of-squares polynomials, the method is able to achieve the near-optimal control of a class of discrete-time systems. An invariant adaptive dynamic programming algorithm is developed to extend the method to scenarios where system dynamics is not available. Online data are utilized to learn the near-optimal policy and the invariantly admissible region. Simulated experiments verify the effectiveness of our method.**

***Index Terms*—Adaptive dynamic programming, discrete-time systems, invariant admissibility, optimal control, policy iteration, sum of squares.**

I. Introduction

AFTER decades of development, adaptive dynamic programming (ADP) [1], [2] has been proved to be a powerful tool in the field of optimal control. In comparison with dynamic programming (DP) [3], ADP avoids the curse of dimensionality by combining with approximation techniques, such as Galerkin approximation [4], neural network [5], fuzzy system [6], polynomials [7], and so forth. The original complicated value/policy functions are approximated with much fewer parameters in a compact set. In the computational intelligence community, researchers prefer reinforcement learning (RL) to refer to algorithms that solve the optimal control problems based on rewards, and in most cases, ADP and RL are interchangeable. Successful applications of ADP/RL include optimal control [8], $\mathcal{H}_\infty$ control [7], multiagent system [9], interconnected system [10], robust control [11], tracking control [12], event-triggered control [13], saturation control [14], time-delayed control [15], global stabilization [16], to name a few.

In ADP, two most commonly used techniques are value iteration (VI) [8], [17] and policy iteration (PI) [18], [19]. The former starts from an initial value function and iterates on value functions to reach the optimal one. Different to VI, PI starts from an initial admissible policy and iterates on the policy until it converges to the optimal one. Two steps are included at each iteration: 1) policy evaluation and 2) policy improvement. A big advantage of PI is that the policy at every iteration is always a stabilizing control law for the system, making it more suitable for online implementation. In the past, PI has been fully studied for continuous-time systems [20]–[23], but in the recent years considerable efforts have been made to discrete-time PI. In [24], PI was applied to discrete-time nonlinear systems, and the optimal stabilizing control was obtained under the convergence theorem. Convergence of approximate PI was investigated in [25]. Discrete-time infinite horizon problems of optimal control to a terminal set of states were studied in [26]. The uniqueness of the solution of Bellman equation is established, and the convergence of VI and PI is both provided. Q functions were introduced in [27] and [28] to solve the optimal policies without knowing system models.

One important fact cannot be ignored is that many systems can only be locally stabilized, not globally. ADP confronts the same issue when value/policy functions are approximated in a finite region. In that case, PI may synthesize a new policy that is no long admissible in the originally prescribed region. Continuing the iterative process in the old region may cause unpredictable results. As a consequence, regional PI should update both policy and effective region at each iteration. In [29], an invariantly admissible PI is proposed for continuous-time nonlinear systems. For each new policy, a new region is defined such that the policy is still invariantly admissible inside of it. The next iteration continues on the basis of new results. For discrete-time systems, to the best of our knowledge, there has been no literature on this topic except [30]. At each iteration of [30], policies are locally updated over a sequence of state sets, but the sets are manually specified beforehand, limiting its application.

To deal with the regionality appearing in discrete-time optimal control, an invariant PI is proposed in this paper. At each policy improvement step, an invariantly admissible region is defined for the improved policy. The region is

Manuscript received September 17, 2018; revised February 3, 2019; accepted April 9, 2019. Date of publication April 29, 2019; date of current version October 15, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61603382. This paper was recommended by Associate Editor Y.-J. Liu. *(Corresponding author: Haibo He.)*

Y. Zhu and D. Zhao are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: yuanheng.zhu@ia.ac.cn; dongbin.zhao@ia.ac.cn).

H. He is with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI 02881 USA (e-mail: he@ele.uri.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2019.2911900

a lower-level set of value function, and the policy always steers the system inside it and eventually to the zero point. The next policy evaluation for the new policy is performed on the new region. It is proved that invariant PI regionally converges to the optimal solution of discrete-time optimal control problems. Based on that a specific type of discrete-time systems is studied. The system has polynomial nonlinearity in input-gain dynamics. With quadratic value definition and sum-of-squares (SOSs) relaxation [31], [32], the policy is expressed as a ratio of polynomials, and value/policy coefficients are searched in the SOS polynomial space. Theoretical analysis proves the algorithm is convergent and the optimality gap is bounded. Furthermore, a model-free version of the algorithm is designed. Its implementation requires no knowledge of system dynamics, so it is applicable to dynamics unavailable cases. Numerical simulations verify the effectiveness of the new algorithm.

The contributions of this paper are threefold.

1) Regional PI of discrete-time systems is studied for the first time. An invariantly admissible region is defined at each policy improvement step for the new policy, and the next policy evaluation step is performed on it. It ensures the correctness of the learning process in comparison to the existing discrete-time PI [18], [24]–[26], [28], [33], [34] that uses a constant region.
2) SOS is introduced in discrete-time ADP to approximate value functions and define constraints. In contrast to tradition ADP [5], [6], [8], [18], [19], [24], [33], the value/policy coefficients are searched in SOS polynomial space such that the positivity of value functions and the Lyapunov condition are satisfied.
3) For the discrete-time systems that have polynomial input dynamics, a model-free algorithm is developed such that the near-optimal policy and the invariantly admissible region are learned based on data. This feature distinguishes itself from other works like [35]–[38] that require complete or partial dynamics knowledge.

The remainder of this paper is organized as follows. In Section II, the preliminary of discrete-time optimal control is introduced. In Section III, the invariant PI is proposed to learn the optimal policy and the invariantly admissible region. Discrete-time systems with polynomial input dynamics are specified in Section IV and SOS polynomials are adopted for the implementation of invariant PI. To deal with dynamics unavailable cases, a model-free algorithm is developed in Section V to learn the polynomial coefficients based on data. Numerical experiments are simulated in Section VI, and the conclusion is reached in the end.

Notation: $\mathbb{R}^n$ is the real vector space of dimension n and $\mathbb{R}^{n\times n}$ is the real matrix space of size $n \times n$. $\|\cdot\|$ is the vector norm or induced matrix norm. I is the unit matrix. Throughout this paper, all matrices and vectors are compatibly dimensioned. For two sets Ω_1 and Ω_2, $\Omega_1 \subseteq \Omega_2$ means Ω_1 is a subset of Ω_2, and $\partial\Omega_1$ is the boundary of Ω_1. $\mathcal{C}(\Omega)$ is the set of all continuous functions in Ω, and $\mathcal{P}(\Omega)$ is the set of all functions that are positive definite and proper in $\mathcal{C}(\Omega)$. $\mathbb{R}[x]$ defines the set of all polynomials in x with real coefficients. $\deg(\cdot)$ is the degree of a given polynomial. SOS is the set of all SOSs polynomials.

II. Discrete-Time Optimal Control

The generalized discrete-time nonlinear systems can be described by

$$x_{k+1} = f(x_k, u_k) \tag{1}$$

where step $k \geq 0$, states $x_k, x_{k+1} \in \mathbb{R}^n$, control input $u_k \in \mathbb{R}^m$, dynamics $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. n, and m denote dimensions of state and control space. We assume f is a continuous function and has $f(0, 0) = 0$. In the sequel, when necessary the subscript k is attached to highlight the time-order relationship of variables. Otherwise, it is omitted for simplicity.

A *policy* μ specifies the control actions at each step, $\mu = \{u_0, u_1, \ldots, \}$, and its control performance is evaluated by the *cost*

$$J(x_0; \mu) = \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k \tag{2}$$

where Q and R are symmetric positive-definite matrices. In this paper, we are interested in state-feedback policies, i.e., $\mu : \mathbb{R}^n \to \mathbb{R}^m$. For ease of notation, we use μ_k and x_k^μ to denote the control and state at the kth step of a system trajectory that is generated under μ.

Definition 1 (Invariantly Admissible): Given a state-feedback policy μ and a region $\Omega \subseteq \mathbb{R}^n$ that contains the origin, if:

1) μ is continuous in Ω;
2) starting from any $x_0 \in \Omega$, μ stabilizes the system, i.e., $\lim_{k\to\infty} x_k^\mu = 0$, and

$$\forall x_0 \in \Omega \Rightarrow x_k^\mu \in \Omega \quad \forall k \geq 0 \tag{3}$$

3) $\forall x_0 \in \Omega, J(x_0; \mu) < +\infty$, then μ is called an *invariantly admissible* policy and Ω is its invariantly admissible region, denoted by $\mu \in \mathcal{A}_I(\Omega)$.

Definition 1 can be seen as a discrete-time version of invariant admissibility given in [29]. From (2), given a policy $\mu \in \mathcal{A}_I(\Omega)$, its cost satisfies the Lyapunov equation

$$V(x_k) - V\left(x_{k+1}^\mu\right) - x_k^T Q x_k - \mu_k^T R \mu_k = 0, V(0) = 0 \tag{4}$$

for any $x_k \in \Omega$. The solution V is also called *value* function, and its uniqueness can be illustrated by the following lemma.

Lemma 1: Given a region Ω and a policy $\mu \in \mathcal{A}_I(\Omega)$, the Lyapunov (4) has a unique solution in the continuously differentiable function set $\mathcal{C}(\Omega)$.

The proof is given in the Appendix. Based on the lemma, the cost of an invariantly admissible policy can be obtained by solving the Lyapunov (4) in its invariantly admissible region. From the cost definition, V is continuous and positive definite in Ω.

The optimal control objective is to find the *optimal policy* μ^* that achieves the minimum cost among all policies

$$V^* = \min J(\cdot; \mu).$$

It is not hard to see that μ^* has the largest invariantly admissible region, denoted by Ω^*. V^* is also called the *optimal value* function and satisfies the Bellman [3] in Ω^*

$$V^*(x_k) - \min_{u_k}\left[x_k^T Q x_k + u_k^T R u_k + V^*(x_{k+1})\right] = 0$$
$$V^*(0) = 0. \qquad (5)$$

If Ω^* covers the whole state space $\mathbb{R}^n$, the system is globally stabilizable, that is to say it can be stabilized to equilibrium from any initial state. The corresponding Bellman equation has a unique continuously differentiable solution. Otherwise, the system is only regionally stabilizable. Literature like [39] and [40] have studied the uniqueness condition of Bellman equation, and here the following assumption is made.

Assumption 1: In the region Ω^*, the Bellman (5) has a unique solution in $\mathcal{C}(\Omega^*)$.

The optimal policy is formulated based on V^* by

$$\mu^*(x_k) = \arg\min_{u_k}\left[x_k^T Q x_k + u_k^T R u_k + V^*(x_{k+1})\right].$$

Remark 1: In the previous discrete-time PI research [18], [24], [25], [28], [33], the concept of admissibility is mostly used, but here we introduce the invariant admissibility. The difference exists in the restriction of trajectories as illustrated in the second condition of Definition 1. Invariant admissibility requires trajectories stay in the invariantly admissible region, while traditional admissibility does not concern that. This restriction is necessary because the Lyapunov (4) and the Bellman (5) require value functions to be well defined for both x_k and x_{k+1}. If x_{k+1} is outside the region, unexpected errors are brought to the solutions. Therefore, in order to evaluate the correct value function of a policy, one needs to know its invariantly admissible region.

III. Invariant Policy Iteration

To obtain the optimal value and the optimal policy, an invariant PI is proposed to solve the Bellman (5). The method iteratively evaluates the value function of a given policy in its invariantly admissible region, and then updates the policy and the region for the next iteration. The detailed steps are listed in Algorithm 1.

Algorithm 1 Invariant PI

Given a region $\Omega^{(1)} \subseteq \Omega^*$ and a policy $\mu^{(1)}$ that have $\mu^{(1)} \in \mathcal{A}_I(\Omega^{(1)})$. Repeat the following two steps for $i \geq 1$

1) (**Policy evaluation**) Formulate the following Lyapunov equation in $\Omega^{(i)}$ and solve for $V^{(i)} \in \mathcal{C}(\Omega^{(i)})$ with policy $\mu^{(i)}$

$$V^{(i)}(x_k) - V^{(i)}\left(x_{k+1}^{(i)}\right) - x_k^T Q x_k - \left(\mu_k^{(i)}\right)^T R \mu_k^{(i)} = 0,$$
$$V^{(i)}(0) = 0 \qquad (6)$$

where the superscript (i) indicates the variables are related to the ith iteration.

2) (**Policy improvement**) Since $V^{(i)}$ is continuously positive definite in $\Omega^{(i)}$, there exists $d^{(i+1)} > 0$ such that

$$\Omega^{(i+1)} = \left\{x \in \mathbb{R}^n | V^{(i)}(x) \leq d^{(i+1)}\right\}$$

is a compact set and has $\Omega^{(i+1)} \subseteq \Omega^{(i)}$. Define the new policy in $\Omega^{(i+1)}$ as

$$\mu^{(i+1)}(x_k) = \arg\min_{u_k}\left[x_k^T Q x_k + u_k^T R u_k + V^{(i)}(x_{k+1})\right]. \qquad (7)$$

In the algorithm, $d^{(i+1)}$ defines the sublevel set of $V^{(i)}$ for the new policy region $\Omega^{(i+1)}$, and the region is required to be the subset of $\Omega^{(i)}$. If the invariantly admissible region keeps constant, i.e., $\Omega^{(i)} = \Omega^{(i+1)}, \forall i \geq 1$ (for example the globally stabilizable systems), invariant PI is equivalent to traditional PI in [18], [24], [25], [28], and [33]. The following theorem illustrates the convergence of invariant PI.

Theorem 1: Given the initial $\mu^{(1)} \in \mathcal{A}_I(\Omega^{(1)})$, a sequence of values $\{V^{(i)}\}$, policies $\{\mu^{(i)}\}$, and regions $\{\Omega^{(i)}\}$ are produced by invariant PI. Under Assumption 1, the following statements are true $\forall i \geq 1$:

1) $\mu^{(i)} \in \mathcal{A}_I(\Omega^{(i)})$;
2) $0 \leq V^*(x) \leq V^{(i)}(x) \leq V^{(i-1)}(x) \leq \cdots \leq V^{(1)}(x)$, $\forall x \in \Omega^{(i)}$;
3) let $\Omega^\infty = \bigcap_{i=1}^\infty \Omega^{(i)}$. In Ω^∞, the sequence $\{V^{(i)}\}$ converges to V^*.

Proof:

1) The statement is proved by induction. First, it is true for $i = 1$. Then assume that it holds for $i > 1$ and prove it is true for $(i + 1)$. According to the definition, $\Omega^{(i+1)} \subseteq \Omega^{(i)}$. From (6) and (7) the following inequality holds $\forall x_k \in \Omega^{(i+1)}$:

$$V^{(i)}(x_k) \geq x_k^T Q x_k + \left(\mu_k^{(i+1)}\right)^T R \mu_k^{(i+1)} + V^{(i)}\left(x_{k+1}^{(i+1)}\right) \qquad (8)$$

and implies $V^{(i)}(x_{k+1}^{(i+1)}) \leq V^{(i)}(x_k) \leq d^{(i+1)}$ and $x_{k+1}^{(i+1)} \in \Omega^{(i+1)}$. It is concluded that $\mu^{(i+1)}$ governs the system within $\Omega^{(i+1)}$, and (8) holds for all points in the trajectory. By [41, Th. 2.1], $V^{(i)}$ is a Lyapunov function and $\mu^{(i+1)}$ is a stabilizing policy in $\Omega^{(i+1)}$. Recursively extending (8) yield

$$V^{(i)}(x_0) \geq \sum_{k=0}^{N}\left(\left(x_k^{(i+1)}\right)^T Q x_k^{(i+1)} + \left(\mu_k^{(i+1)}\right)^T R \mu_k^{(i+1)}\right) + V^{(i)}\left(x_{N+1}^{(i+1)}\right).$$

When $N \to \infty$, the right-hand side becomes the cost of $\mu^{(i+1)}$ and it has $J(x_0; \mu^{(i+1)}) = V^{(i+1)}(x_0) \leq V^{(i)}(x_0)$.

The conditions of invariant admissibility are satisfied for $\mu^{(i+1)}$ in $\Omega^{(i+1)}$. By induction, the statement is true for any $i \geq 1$.

2) From the above analysis, $\Omega^{(i)} \subseteq \Omega^{(i-1)} \subseteq \cdots \subseteq \Omega^{(1)}$ and $\{V^{(i)}\}$ is a nonincreasing sequence in $\Omega^{(i)}$. The lower bound of $V^{(i)}$ follows the definition of V^*.
3) In Ω^∞, $\{V^{(i)}\}$ is convergent and its limit $V^\infty = \lim_{i\to\infty} V^{(i)}$ satisfies the Bellman (5). Under the uniqueness assumption, $V^\infty = V^*$ in Ω^∞. ■

In invariant PI algorithm, the invariantly admissible region is updated at each iteration, and it has $\Omega^{(i+1)} \subseteq \Omega^{(i)}$. A special case is discrete-time linear quadratic optimal control. The value function is quadratic in state and the policy is linearly state-feedback. For discrete-time linear systems, any stabilizing policy is invariantly admissible in the whole state space. In this case, the invariant PI algorithm has $\Omega^{(i)} = \Omega^{(i+1)} = \mathbb{R}^n$.

Reviewing Algorithm 1, the value function is obtained by solving the Lyapunov (6). An alternative way is to convert the Lyapunov equation into an inequality and relax the evaluation process to an optimization problem. Define a Lyapunov operator $\mathcal{L}$ such that for continuous functions $V : \mathbb{R}^n \to \mathbb{R}$ and $\mu : \mathbb{R}^n \to \mathbb{R}^m$, let

$$\mathcal{L}(V, \mu, x_k) = V(x_k) - V\big(x_{k+1}^{\mu}\big) - x_k^T Q x_k - \mu_k^T R \mu_k. \quad (9)$$

The Bellman equation can be rewritten by

$$\max_{\mu} \mathcal{L}\big(V^*, \mu, x_k\big) = 0.$$

About $\mathcal{L}$, we have the following lemma.

Lemma 2: Given a region Ω containing the origin and a policy μ. If there exists a function $V \in \mathcal{P}(\mathbb{R}^n)$ that is radially unbounded and satisfies the inequality

$$\mathcal{L}(V, \mu, x) \geq 0, \quad \forall x \in \Omega$$

then there exists a positive constant d such that $\Omega' = \{x \in \mathbb{R}^n | V(x) \leq d\}$ is a subset of Ω, and μ is an invariantly admissible policy in Ω'.

Proof: Since V is continuously positive definite and radially unbounded, there exists d that makes $\Omega' \subseteq \Omega$. According to the Lyapunov operator definition $\forall x_k \in \Omega'$

$$\mathcal{L}(V, \mu, x_k) = V(x_k) - V\big(x_{k+1}^{\mu}\big) - x_k^T Q x_k - \mu_k^T R \mu_k \geq 0.$$

The conclusion is drawn following the proof of Theorem 1-1). ■

From the above analysis, the invariant admissibility of a policy μ in a region Ω can be proved by finding a function V that satisfies the conditions in Lemma 2. Along the trajectory generated by μ in Ω, adding up (9) yield

$$V(x_0) - J(x_0; \mu) = \sum_{k=0}^{\infty} \mathcal{L}(V, \mu, x_k) \geq 0. \quad (10)$$

In other words, V is an overestimate of the cost of μ. Based on that the ith policy evaluation in invariant PI can be replaced by the optimization problem

$$\min \int_{\Omega} V(x)dx \quad (11)$$
$$\text{s.t.} \quad V \in \mathcal{P}\big(\mathbb{R}^n\big) \quad (12)$$
$$\mathcal{L}\Big(V, \mu^{(i)}, x\Big) \geq 0 \quad \forall x \in \Omega^{(i)} \quad (13)$$
$$V^{(i-1)}(x) - V(x) \geq 0 \quad \forall x \in \Omega^{(i)} \quad (14)$$

where $\mu^{(i)}$ is the given policy, $\Omega^{(i)}$ is its invariantly admissible region, $V^{(i-1)}$ is the result of last iteration, and Ω is a subset of $\Omega^{(i)}$ representing the area that desires to be optimized. The real cost $J(\cdot; \mu)$ is the minimum feasible solution to the optimization problem. As a consequence, (11)–(14) are equivalent to the Lyapunov (6). Now the relaxed invariant PI algorithm for the discrete-time optimal control is presented in Algorithm 2. Due to the equivalence, the convergence conclusion of Theorem 1 also holds for the new algorithm.

Algorithm 2 Relaxed Invariant PI

Give an initial invariantly admissible policy $\mu^{(1)}$ and its region $\Omega^{(1)}$. For $i \geq 1$,

1) Formulate the optimization problem (11)–(14) based on $\mu^{(i)}$ and $\Omega^{(i)}$, and denote the optimal solution as $V^{(i)}$. Note that when $i = 1$, the constraint (14) is removed.
2) Define a new region

$$\Omega^{(i+1)} = \Big\{x \in \mathbb{R}^n | V^{(i)}(x) \leq d^{(i+1)}\Big\} \quad (15)$$

such that $\Omega^{(i+1)} \subseteq \Omega^{(i)}$, and update a new policy in $\Omega^{(i+1)}$

$$\mu^{(i+1)}(x_k) = \arg\min_{u_k} \Big[x_k^T Q x_k + u_k^T R u_k + V^{(i)}(x_{k+1})\Big]. \quad (16)$$

Remark 2: In the relaxed invariant PI, the policy evaluation is relaxed to search in the constrained function space such that (11) is minimized. Generally speaking, checking inequalities is an NP-hard problem, so some specific constraints are needed to make the problem feasible. In the next section, a class of discrete-time polynomial systems is specified. Quadratic value functions and SOSs polynomials are introduced into the relaxed invariant PI, and the near-optimal policy and the invariantly admissible region are synthesized.

IV. Near-Optimal Control of Class of Discrete-Time Systems

The discrete-time system is assumed to be input-affine and have a linear drift dynamics

$$x_{k+1} = F x_k + g(x_k) u_k \quad (17)$$

where $F \in \mathbb{R}^{n\times n}$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$. g is further assumed to be polynomial. Now, we use SOSs theory to address inequality constraints appearing in the relaxed invariant PI algorithm.

A polynomial $p(x)$ is called SOS if it can be written as the sum of squares of polynomials, that is

$$p(x) = \sum_{j=1}^{N} \big(p_j(x)\big)^2.$$

SOS polynomials are naturally globally positive, but the converse is not true. According to the polynomial theory [31], determining if a polynomial is SOS is equivalent to find a symmetric positive matrix W such that

$$p(x) = \big(z_p(x)\big)^T W z_p(x)$$

where z_p is a vector of monomials in x. In this way, SOS decomposition is transformed to a semidefinite programming (SDP) and numerous SDP toolboxes are developed to solve it.

For system (17), quadratic value functions are specified, $V(x) = x^T W_V x$. If $W_V > 0$, V is globally positive. Differentiating the right-hand side of (16) toward u_k and assigning to zero yield the explicit formula of updated policy in policy improvement step

$$\mu'(x) = -\big(R + (g(x))^T W_V g(x)\big)^{-1} (g(x))^T W_V F x.$$

Note that R and W_V are both positive definite, so the above formula is valid. The invariantly admissible region is expressed by $\{x | x^T W_V x \le d\}$. The policy function is a ratio of polynomials. For simplicity, policies of system (17) in the sequel are all expressed in the form

$$\mu(x) = \frac{\beta(x)}{\alpha(x)}$$

with $\alpha \in \mathbb{R}[x]$ and $\beta = [\beta_1, \ldots, \beta_m]^T$, where $\beta_1, \ldots, \beta_m \in \mathbb{R}[x]$. After inserting V and μ into the Lyapunov operator, we have

$$\begin{aligned}
&\mathcal{L}(V, \mu, x) \\
&= \frac{1}{(\alpha(x))^2}\Big[(\alpha(x))^2 x^T W_V x \\
&\qquad - (Fx\alpha(x) + g(x)\beta(x))^T W_V (Fx\alpha(x) + g(x)\beta(x)) \\
&\qquad - (\alpha(x))^2 x^T Q x - (\beta(x))^T R \beta(x)\Big].
\end{aligned}$$

SOS theory provides a computationally feasible way to address globally positive constraints. However, for certain cases, global positivity may be too restrictive if just local inequalities are required. Such as the constraints in (13) and (14). To cope with that $\mathcal{S}$-procedure is introduced [42]. Let a compact set be described by a polynomial inequality, i.e., $\Omega = \{x | b(x) \ge 0, b \in \mathbb{R}[x]\}$. If one desires to find a polynomial $p(x)$ that is positive over Ω, a sufficient condition is the existence of an SOS polynomial multiplier $\lambda(x) \ge 0$ such that $p(x) - \lambda(x) b(x) \in$ SOS.

According to the above analysis, a quadratic SOSs PI (QSPI) algorithm is proposed in Algorithm 3 for the optimal control of discrete-time system (17). It is based on relaxed invariant PI, and uses SOS polynomials for the inequality constraints. The algorithm stops when the norm of the difference between $W_V^{(i)}$ and $W_V^{(i-1)}$ is less than a threshold, or the policy evaluation has no feasible solution. The former indicates the algorithm has converged and the latter means the policy cannot be further improved by QSPI algorithm. Take the final synthesized policy $\mu^{(i)}$ as the near-optimal policy and output the invariantly admissible region $\Omega^{(i)}$.

Algorithm 3 QSPI

Given an initial policy $\mu^{(1)}(x) = [(\beta^{(1)}(x))/(\alpha^{(1)}(x))]$ that is invariantly admissible in a region described by $\Omega^{(1)} = \{x | b^{(1)}(x) \ge 0, b^{(1)} \in \mathbb{R}[x]\}$. For each $i \ge 1$,

1) (**Policy evaluation**) Define $V(x) = x^T W_V x$ and a polynomial $\lambda(x)$, whose coefficients are solved by the SOS optimization

$$\max \int_\Omega V(x)dx \tag{18}$$
$$\text{s.t.}\quad V(x) \in SOS \tag{19}$$
$$\lambda(x) \in SOS \tag{20}$$
$$\Big(\alpha^{(i)}(x)\Big)^2 \mathcal{L}\Big(V, \mu^{(i)}, x\Big) - \lambda(x) b^{(i)}(x) \in SOS \tag{21}$$
$$V^{(i-1)}(x) - V(x) \in SOS. \tag{22}$$

Note that for $i = 1$, the constraint (22) is removed. Denote the optimal solution as $W_V^{(i)}$, and let $V^{(i)}(x) = x^T W_V^{(i)} x$.

2) (**Policy improvement**) Update the policy $\mu^{(i+1)}(x) = [(\beta^{(i+1)}(x))/(\alpha^{(i+1)}(x))]$ with

$$\alpha^{(i+1)}(x) = \det\Big(R + (g(x))^T W_V^{(i)} g(x)\Big)$$
$$\beta^{(i+1)}(x) = -\mathrm{adj}\Big(R + (g(x))^T W_V^{(i)} g(x)\Big) \cdot (g(x))^T W_V^{(i)} F x$$

$\det(\cdot)$ and $\mathrm{adj}(\cdot)$ denote the determinant and adjugate of a given matrix. The invariantly admissible region is defined as $\Omega^{(i+1)} = \{x | b^{(i+1)}(x) \ge 0\}$ with

$$b^{(i+1)}(x) = \min_{y \in \partial\Omega^{(i)}} V^{(i)}(y) - V^{(i)}(x).$$

To make SOS constraints valid, polynomials in constraints (19)–(33) should satisfy

$$\deg(\alpha) + 1 \ge \max\left\{\deg(g) + \deg(\beta), \frac{1}{2}\deg(\lambda) + \frac{1}{2}\deg(b)\right\}.$$

Theorem 2: Apply the QSPI algorithm to discrete-time system (17), and suppose at the ith iteration have obtained the values $\{V^{(1)}, \ldots, V^{(i)}\}$, policies $\{\mu^{(1)}, \ldots, \mu^{(i+1)}\}$, and regions $\{\Omega^{(1)}, \ldots, \Omega^{(i+1)}\}$. Then:

1) $\mu^{(i+1)} \in \mathcal{A}_I(\Omega^{(i+1)})$;
2) $0 \le V^*(x) \le V^{(i)}(x) \le V^{(i-1)} \le \cdots \le V^{(1)}(x)$, $\forall x \in \Omega^{(i)}$;
3) for any $x_0 \in \Omega^{(i+1)}$, the cost of $\mu^{(i+1)}$ satisfies

$$\begin{aligned}
J\Big(x_0; \mu^{(i+1)}\Big) &= V^*(x_k) - \sum_{k=0}^{\infty} \mathcal{L}\Big(V^*, \mu^{(i+1)}, x_k^{(i+1)}\Big) \\
&\le V^{(i)}(x_0).
\end{aligned}$$

Proof:

1) First, we prove $\Omega^{(i+1)}$ is a subset of $\Omega^{(i)}$. Denote y_0 as the point on the boundary $\partial\Omega^{(i)}$ that has $V^{(i)}(y_0) = \min_{y \in \partial\Omega^{(i)}} V^{(i)}(y)$. If $\Omega^{(i+1)}$ is not a subset of $\Omega^{(i)}$,

there exists a point $y_1 \in \partial\Omega^{(i)}$ that is in the interior of $\Omega^{(i+1)}$. According to the definition of $\Omega^{(i+1)}$, $V^{(i)}(y_1) < V^{(i)}(y_0)$ which is contradict to the definition of y_0. Hence, $\Omega^{(i+1)} \subseteq \Omega^{(i)}$, and according to Lemma 2, $\mu^{(i+1)}$ is invariantly admissible in $\Omega^{(i+1)}$.

2) The statement is true following constraint (22).

3) Since $\mu^{(i+1)} \in \mathcal{A}_I(\Omega^{(i+1)})$, starting from any $x_0 \in \Omega^{(i+1)}$, $\mu^{(i+1)}$ governs the system inside $\Omega^{(i+1)}$. Along the trajectory, apply the Lyapunov operator definition

$$\begin{aligned}
&\mathcal{L}\Big(V^*, \mu^{(i+1)}, x_0\Big)\\
&\quad = V^*(x_0) - V^*\Big(x_1^{(i+1)}\Big) - x_0^T Q x_0\\
&\qquad - \Big(\mu_0^{(i+1)}\Big)^T R \mu_0^{(i+1)}\\
&\mathcal{L}\Big(V^*, \mu^{(i+1)}, x_1^{(i+1)}\Big)\\
&\quad = V^*\Big(x_1^{(i+1)}\Big) - V^*\Big(x_2^{(i+1)}\Big)\\
&\qquad - \Big(x_1^{(i+1)}\Big)^T Q x_1^{(i+1)} - \Big(\mu_1^{(i+1)}\Big)^T R \mu_1^{(i+1)}\\
&\vdots
\end{aligned}$$

Summing up both sides yield

$$\sum_{k=0}^{\infty} \mathcal{L}\Big(V^*, \mu^{(i+1)}, x_k^{(i+1)}\Big) = V^*(x_0) - J\Big(x_0; \mu^{(i+1)}\Big).$$

Similarly, replacing V^* by $V^{(i)}$, the above equation becomes

$$\sum_{k=0}^{\infty} \mathcal{L}\Big(V^{(i)}, \mu^{(i+1)}, x_k^{(i+1)}\Big) = V^{(i)}(x_0) - J\Big(x_0; \mu^{(i+1)}\Big).$$

According to the definition of $\mu^{(i+1)}$, $\mathcal{L}(V^{(i)}, \mu^{(i+1)}, x_k) \geq \mathcal{L}(V^{(i)}, \mu^{(i)}, x_k)$. From constraint (21), $\mathcal{L}(V^{(i)}, \mu^{(i)}, x_k) \geq 0$. The conclusion $J(x_0; \mu^{(i+1)}) \leq V^{(i)}(x_0), \forall x_0 \in \Omega^{(i+1)}$ is reached. ■

Remark 3: From Theorem 2-3), $V^{(i)}$ is proved to be an overestimate of $J(\cdot; \mu^{(i+1)})$, and the suboptimality of $\mu^{(i+1)}$ is established. The minimization in (18) plays a role of lowering the overestimate as much as possible. In addition, through the iteration of QSPI algorithm, values $V^{(1)}, \ldots, V^{(i)}$ are decreasing, so the optimality gap is further reduced.

Remark 4: In QSPI algorithm, V and μ act as the critic and the actor in the framework of RL and ADP. In the previous literature, neural networks are mostly used to approximate these two functions [5], [8], [18], [19], [24], [33]. The critic weights are trained to minimize the Lyapunov function error or Bellman equation error, while the actor weights are searched along the gradient descent of the right-hand side of (16). Technically, it is difficult to verify the positivity of NN-based value functions. In our algorithm, the value and policy coefficients are searched in the SOS polynomial space, so the invariant admissibility is ensured.

Remark 5: A special case for QSPI algorithm is the globally stabilizable systems. In that case, $\mathcal{S}$-procedure is no longer needed and $\lambda(x)$ in the algorithm is set to zero. The synthesized policy at each iteration globally stabilizes the system and the invariantly admissible region covers the whole state space.

V. Invariant Adaptive Dynamic Programming

A drawback of QSPI algorithm is the dependence on the knowledge of system dynamics F and g. Many cases, in practical applications, assume dynamics is unknown or uncertain. To cope with that an invariant ADP algorithm is proposed to learn the near-optimal policy for (17) based on data.

Reviewing the policy evaluation of QSPI algorithm, constraint (21) implies that there exists a polynomial $L(x)$ such that

$$L(x) = \Big(\alpha^{(i)}(x)\Big)^2 \mathcal{L}\Big(V, \mu^{(i)}, x\Big) - \lambda(x) b^{(i)}(x) \tag{23}$$

and $L(x) \in$ SOS. Rewrite L in the form $L(x) = (z_L(x))^T W_L z_L(x)$, where z_L is a vector of monomials in x, and W_L is a symmetric positive-definite matrix whose coefficients are to be determined. Similarly, define λ in the form $\lambda(x) = (z_\lambda(x))^T W_\lambda z_\lambda(x)$, where z_λ is a monomial vector and $W_\lambda \geq 0$ is the matrix to be determined.

Suppose a tuple (x_k, u_k, x_{k+1}) is observed from the system, and it has $x_{k+1} = Fx_k + g(x_k)u_k$. Given the policy $\mu^{(i)}$, the observation can be expressed by

$$x_{k+1} = Fx_k + g_k \mu_k^{(i)} + g_k\Big(u_k - \mu_k^{(i)}\Big).$$

For ease of notation, we use g_k and $\mu_k^{(i)}$ to denote the value of g and $\mu^{(i)}$ with input x_k. After inserting $(Fx_k + g_k\mu_k^{(i)})$ into (23), the following equality holds:

$$\begin{aligned}
L_k = \Big(\alpha_k^{(i)}\Big)^2 \Big[& x_k^T W_V x_k - x_{k+1}^T W_V x_{k+1} + u_k^T g_k^T W_V g_k u_k\\
& - \Big(\mu_k^{(i)}\Big)^T g_k^T W_V g_k \mu_k^{(i)} + 2\Big(u_k - \mu_k^{(i)}\Big)^T g_k^T W_V F x_k\\
& - x_k^T Q x_k - \Big(\mu^{(i)}\Big)^T R \mu_k^{(i)}\Big] - \lambda_k b_k^{(i)}.
\end{aligned} \tag{24}$$

Given a vector h and a matrix W, rearrange the elements of h and W in the form

$$\bar{h} = \Big[h_1^2, h_1h_2, h_1h_3, \ldots, h_2^2, h_2h_3, \ldots\Big]^T \tag{25}$$

$$w = \hat{W} = [W_{11}, 2W_{12}, 2W_{13}, \ldots, W_{22}, 2W_{23}, \ldots]^T. \tag{26}$$

The transformation is invertible, i.e., given $\bar{h}$ and w, one can uniquely determine h and W. Then the quadratic $h^T W h$ can be rewritten as $h^T W h = \bar{h}^T w$. On the same principle, rewrite polynomials V, L, and λ

$$\begin{aligned}
V(x) &= \bar{x}^T w_V\\
L(x) &= (\bar{z}_L(x))^T w_L\\
\lambda(x) &= (\bar{z}_\lambda(x))^T w_\lambda
\end{aligned}$$

where $\bar{x}$, $\bar{z}_L$, and $\bar{z}_\lambda$ are rearranged from x, z_L, and z_λ following (25), and w_V, w_L, and w_λ are rearranged from W_V, W_L, and W_λ following (26).

If the system degree is available, the unknown g can be approximated by

$$g(x) = \left[G_1 z_g(x), G_2 z_g(x), \dots, G_m z_g(x)\right]$$

where z_g is the monomial vector with degrees up to $\deg(g)$, and $G_1, \dots, G_m$ are uncertain coefficients. Let $G = [G_1, \dots, G_m]$. Introduce the Kronecker product operator $\otimes$ and transform the following terms in (24) into the linear form:

$$u_k^T g_k^T W_V g_k u_k = \xi_k^T W_\alpha \xi_k = \bar{\xi}_k^T w_\alpha$$
$$\left(\mu_k^{(i)}\right)^T g_k^T W_V g_k \mu_k^{(i)} = \eta_k^T W_\alpha \eta_k = \bar{\eta}_k^T w_\alpha$$

where $\xi_k = (I \otimes z_{gk}) u_k$, $\eta_k = (I \otimes z_{gk}) \mu_k^{(i)}$, $W_\alpha = G^T W_V G$, and $w_\alpha = \hat{W}_\alpha$. Using the Kronecker product property $\mathrm{vec}(XYZ) = (Z^T \otimes X)\mathrm{vec}(Y)$ where $\mathrm{vec}(\cdot)$ denotes the vectorization of a matrix

$$g_k^T W_V F x_k = \left(I \otimes z_{gk}\right)^T W_\beta x_k$$
$$\left(u_k - \mu_k^{(i)}\right)^T g_k^T W_V F x_k = \zeta_k^T w_\beta$$

where $W_\beta = G^T W_V F$, $\zeta_k = x_k \otimes ((I \otimes z_{gk})(u_k - \mu_k^{(i)}))$, and $w_\beta = \mathrm{vec}(W_\beta)$. Based on the above transformation, (24) now becomes

$$\bar{z}_{Lk}^T w_L = \left(\alpha_k^{(i)}\right)^2 \Big[(\bar{x}_k - \bar{x}_{k+1})^T w_V + \left(\bar{\xi}_k - \bar{\eta}_k\right)^T w_\alpha + 2\zeta_k^T w_\beta - x_k^T Q x_k - \left(\mu_k^{(i)}\right)^T R \mu_k^{(i)} \Big] - b_k^{(i)} \bar{z}_{\lambda k}^T w_\lambda. \tag{27}$$

The equality in (27) holds for arbitrary online observations. Define a data set $\{(x_l, u_l, x_{l+1})\}$ and let

$$A_1 = \begin{bmatrix} \vdots \\ \left(\alpha_l^{(i)}\right)^2 (\bar{x}_l - \bar{x}_{l+1})^T \\ \vdots \end{bmatrix}, A_2 = \begin{bmatrix} \vdots \\ -b_l^{(i)} \bar{z}_{\lambda l}^T \\ \vdots \end{bmatrix}$$

$$B = \begin{bmatrix} \vdots \\ \left(\alpha_l^{(i)}\right)^2 \left[-x_l^T Q x_l - \left(\mu_l^{(i)}\right)^T R \mu_l^{(i)} \right] \\ \vdots \end{bmatrix}$$

$$C_1 = \begin{bmatrix} \vdots \\ \bar{z}_{Ll}^T \\ \vdots \end{bmatrix}, \quad C_2 = \begin{bmatrix} \vdots \\ -\left(\alpha_l^{(i)}\right)^2 \left(\bar{\xi}_l - \bar{\eta}_l\right)^T \\ \vdots \end{bmatrix}$$

$$C_3 = \begin{bmatrix} \vdots \\ -2\left(\alpha_l^{(i)}\right)^2 \zeta_l^T \\ \vdots \end{bmatrix}.$$

Then, we can rewrite (27) in the matrix form

$$A \begin{bmatrix} w_V \\ w_\lambda \end{bmatrix} + B = C \begin{bmatrix} w_L \\ w_\alpha \\ w_\beta \end{bmatrix}$$

with $A = [A_1\ A_2]$ and $C = [C_1\ C_2\ C_3]$. If C is a full column rank, coefficients w_L, w_α, and w_β are uniquely determined by w_V and w_λ

$$\begin{bmatrix} w_L \\ w_\alpha \\ w_\beta \end{bmatrix} = \left(C^T C\right)^{-1} C^T \left(A \begin{bmatrix} w_V \\ w_\lambda \end{bmatrix} + B \right).$$

The SOS optimization in the policy evaluation of QSPI algorithm can now be reformulated in a model-free way. Given $\mu^{(i)} \in \mathcal{A}_I(\Omega^{(i)})$, collect sufficient data $\{(x_l, u_l, x_{l+1})\}^{(i)}$ that make C full column rank. Optimize the determinant variables W_V and W_λ by the SDP program

$$\max \int_\Omega x^T W_V x dx \tag{28}$$
$$\text{s.t.} \begin{bmatrix} w_L \\ w_\alpha \\ w_\beta \end{bmatrix} = \left(C^T C\right)^{-1} C^T \left(A \begin{bmatrix} w_V \\ w_\lambda \end{bmatrix} + B \right) \tag{29}$$
$$W_V \geq 0 \tag{30}$$
$$W_L \geq 0 \tag{31}$$
$$W_\lambda \geq 0 \tag{32}$$
$$W_V \leq W_V^{(i-1)}. \tag{33}$$

When $i = 1$, constraint (33) is removed. It is clear to see that w_α and w_β play an intermediate role in the SDP program. But they are used to synthesize the new policy in the policy improvement step. Denote the optimal solution as $W_V^{(i)}$ and $W_\lambda^{(i)}$, and calculate $W_\alpha^{(i+1)}$ and $W_\beta^{(i+1)}$ by (29). The dominant and numerator of the fractional policy is updated by

$$\alpha^{(i+1)}(x) = \det\Big(R + \left(I \otimes z_g(x)\right)^T W_\alpha^{(i+1)} \left(I \otimes z_g(x)\right)\Big)$$
$$\beta^{(i+1)}(x) = -\mathrm{adj}\Big(R + \left(I \otimes z_g(x)\right)^T W_\alpha^{(i+1)} \left(I \otimes z_g(x)\right)\Big) \times \left(I \otimes z_g(x)\right)^T W_\beta^{(i+1)} x.$$

The invariantly admissible region is updated by $\Omega^{(i+1)} = \{x | b^{(i+1)}(x) \geq 0\}$ with

$$b^{(i+1)}(x) = \min_{y \in \partial\Omega^{(i)}} y^T W_V^{(i)} y - x^T W_V^{(i)} x.$$

Remark 6: The whole process of IADP algorithm is summarized in Fig. 1. Note that the data set that formulates the SDP program (28)–(33) is repeatedly utilized at different iterations to increase data efficiency. Moreover, a necessary condition to ensure the solvability of the SDP program is that the matrix C is full-rank in columns. To this end, the system needs to be excited by noised control signals to generate a variety of observations. The control input comprises two parts, $u_k = \mu_k + e_k$, one of which is a stabilizing policy μ_k while the other is noise e_k. In the literature, random noise

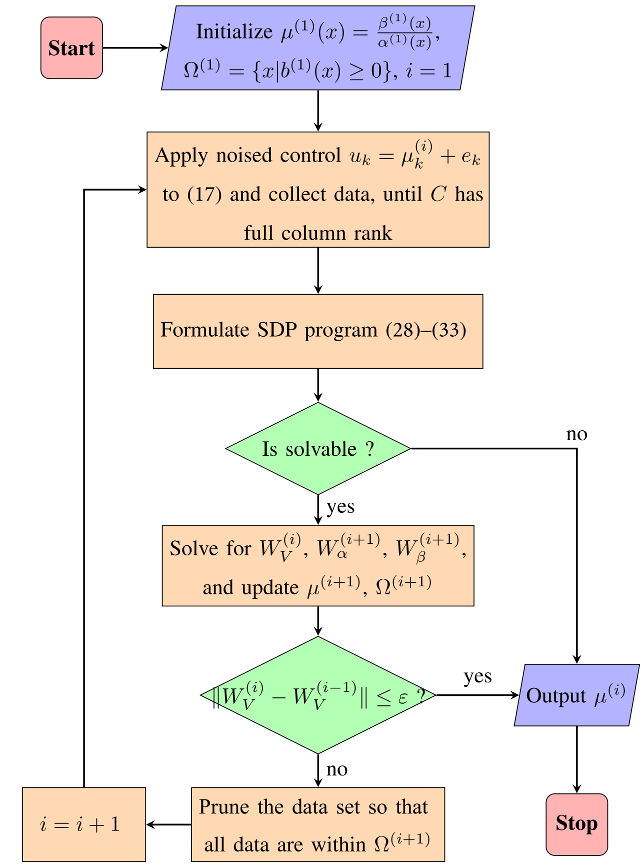


Fig. 1. Flowchart of IADP algorithm.

and sinusoidal signals are mostly used. In our experiments, we use the current policy $\mu^{(i)}$ plus random noise to excite the system. During the online learning process, if the system is disturbed outside the current invariantly admissible region, reset it to the origin and continue collecting data. In our experiments, the SDP program is solved by SOSTOOLS MATLAB toolbox [43].

Remark 7: Before starting QSPI or IADP algorithm, an initial policy $\mu^{(1)}$ and its region $\Omega^{(1)}$ are required. Synthesizing stabilizing control laws and their regions for discrete-time systems has been investigated by many works, including dynamics-known cases [35], [36] and dynamics-uncertain cases [37], [38]. QSPI and IADP algorithms can further optimize these results and find near-optimal policies and invariantly admissible regions. The difference is that QSPI requires the knowledge of system dynamics while IADP is model-free.

Remark 8: The process of learning from data that are not generated by interested policies is called off-policy learning. For CT systems, off-policy learning plays an important role in designing model-free ADP algorithms [16], [22], [44]. For discrete-time systems, the more traditional approach is to define Q functions [27], [28], which take both state and control as input. Q function usually has more parameters than value function, and the additional parameters are used in the policy improvement step to produce new policy. This is similar to the role of w_α and w_β in the SDP program of IADP algorithm.

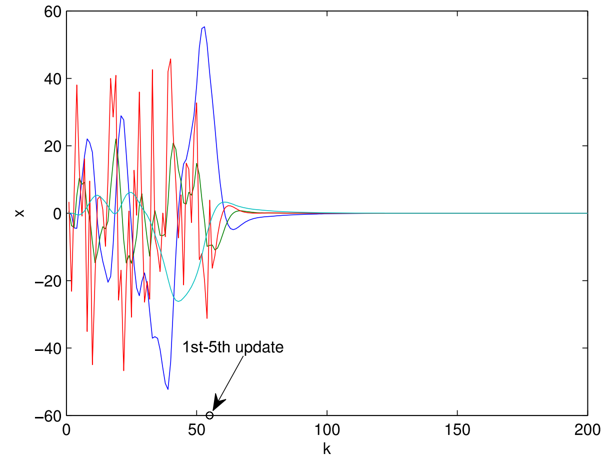


Fig. 2. Linear experiment: online trajectories.

VI. Numerical Simulations

A. Linear Dynamics

The first experiment considers the model of load frequency control [20], whose discrete-time dynamics is

$$x_{k+1} = \begin{bmatrix} 0.970 & 0.663 & 0.085 & -0.044 \\ -0.076 & 0.672 & 0.158 & -0.146 \\ -0.395 & -0.166 & 0.237 & -0.740 \\ 0.059 & 0.021 & 0.002 & 0.999 \end{bmatrix} x_k + \begin{bmatrix} 0.044 \\ 0.146 \\ 0.740 \\ 0.0007 \end{bmatrix} u_k.$$

Note that when the system uses quadratic cost, the optimal control becomes the discrete-time linear quadratic regulator problem. Let the cost selects $Q = I_4$ and $R = 1$. The optimal value function is equal to

$$V^*(x) = x^T \begin{bmatrix} 5.385 & 4.919 & 0.748 & 4.897 \\ 4.919 & 8.763 & 1.420 & 4.061 \\ 0.748 & 1.420 & 1.316 & 0.452 \\ 4.897 & 4.061 & 0.452 & 24.102 \end{bmatrix} x$$

and the optimal state-feedback policy has $\mu^*(x) = -K^* x$

$$K^* = [0.369 \quad 1.087 \quad 0.352 \quad -0.076].$$

The system is self-stabilizable, so when running the IADP algorithm, the initial globally admissible policy selects $\mu^{(1)} = 0$. Random noise with uniform distribution in $[-50, 50]$ is added into current policies to excite the system. The monomial vector z_L is defined up to degree 2. z_g is defined to be constant 1. SOS polynomial $\lambda(x)$ is set to zero because the system is globally stabilizable. The state trajectories of the online process is given in Fig. 2. Once the data set satisfies the full-rank condition, IADP algorithm formulates the SDP program and solves for the new policy. The time of each iteration is marked on the time axis, and the algorithm converges at the fifth iteration. It is observed that after the first iteration, the full-rank condition is persistently satisfied by the existing data for the rest four iterations. After that the converged policy takes over the control input and random noise is stopped.

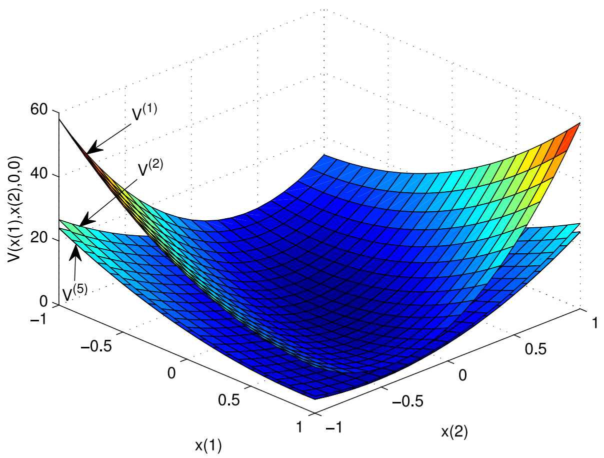


Fig. 3. Linear experiment: value functions at different iterations.

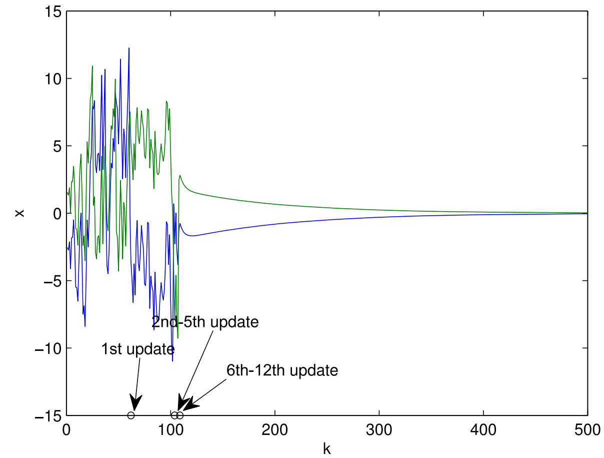


Fig. 4. Bilinear experiment: online trajectories.

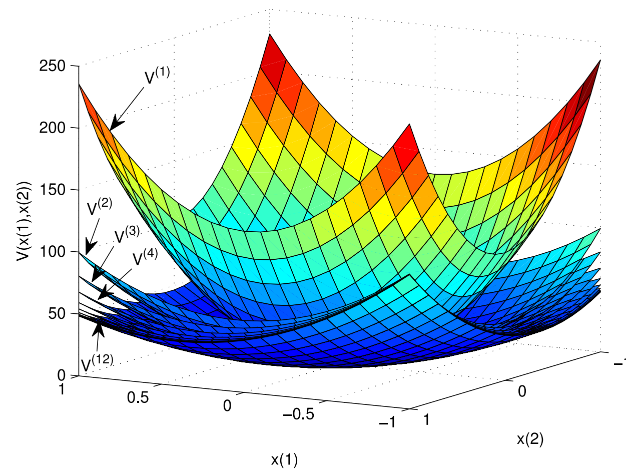


Fig. 5. Bilinear experiment: value functions at different iterations.

Value functions at each iteration are depicted in Fig. 3. The third and forth dimensions of state are set to zero for illustration. By SOS polynomials, the value functions are globally positive. The nonincreasing property of the value functions is consistent with our theorem. The final converged value function has

$$V^{(5)}(x) = x^T \begin{bmatrix} 5.385 & 4.919 & 0.748 & 4.897 \\ 4.919 & 8.763 & 1.420 & 4.061 \\ 0.748 & 1.420 & 1.316 & 0.452 \\ 4.897 & 4.061 & 0.452 & 24.102 \end{bmatrix} x$$

and the converged policy has

$$\mu^{(5)}(x) = -0.369x(1) - 1.087x(2) - 0.352x(3) + 0.076x(4).$$

B. Bilinear Dynamics

The second experiment considers a discrete-time bilinear system [35] with dynamics

$$\begin{aligned} x_{k+1} &= Fx_k + g(x_k)u_k \\ &= \begin{bmatrix} 1 & 0.01 \\ 0.01 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.001x_k(1) + 0.09 \\ -0.004x_k(2) + 0.09 \end{bmatrix} u_k. \end{aligned}$$

Based on the method provided by [35], an initial stabilizing policy $\mu^{(1)}(x) = [(\beta^{(1)}(x))/(\alpha^{(1)}(x))]$ is synthesized

$$\begin{aligned} \alpha^{(1)}(x) &= 0.450(x(1))^2 - 0.395x(1)x(2) + 0.0007x(1) \\ &\quad + 0.425(x(2))^2 - 0.012x(2) + 1.0 \\ \beta^{(1)}(x) &= -0.024(x(1))^2 - 0.024x(1)x(2) - 4.083x(1) \\ &\quad - 0.047(x(2))^2 - 4.094x(2) \end{aligned}$$

and the stabilizing region is $\Omega^{(1)} = \{x \in \mathbb{R}^2 | 120 - (x(1))^2 - (x(2))^2 \geq 0\}$. Starting from that IAPI algorithm learns a near-optimal policy and its invariantly admissible region based on data. The cost function is defined with $Q = I_2$ and $R = 1$. Random noise with uniform distribution in $[-50, 50]$ is added to excite the system. The monomial vectors z_L, z_λ, and z_g are defined up to degrees 3, 2, and 1, respectively.

The state trajectories of the online learning process are given in Fig. 4. The first iteration happens at the 62th step when the full-rank condition is fulfilled. After that the old data set fails in providing enough data for the next iteration. More data are collected under the new policy plus random noise. At the 104th step, from the 2nd to the 5th iterations are performed. At the 109th step, the rest iterations are performed until the algorithm reaches the convergence. After that the converged policy takes over the control input and random noise is removed. Value functions and the invariantly admissible regions at each iteration are plotted in Figs. 5 and 6. It is observed that the new region is always an interior of the previous one. The final converged policy $\mu^{(12)}(x) = [(\beta^{(12)}(x))/(\alpha^{(12)}(x))]$ and its region have

$$\begin{aligned} \alpha^{(12)}(x) &= 0.00004(x(1))^2 + 0.0001x(1)x(2) + 0.004x(1) \\ &\quad + 0.0007(x(2))^2 - 0.019x(2) + 1.392 \\ \beta^{(12)}(x) &= -0.036(x(1))^2 - 0.044x(1)x(2) - 1.950x(1) \\ &\quad + 0.167(x(2))^2 - 2.448x(2) \\ \Omega^{(12)} &= \left\{ x \,\middle|\, \begin{aligned} &2533 - 36.364(x(1))^2 + 29.939x(1)x(2) \\ &- 41.955(x(2))^2 \geq 0 \end{aligned} \right\}. \end{aligned}$$

We select 20 equivalently distributed points along the boundary of $\Omega^{(12)}$ as starting states, and depict the phase portraits of the system under the converged policy by IADP in Fig. 7(a). It is obvious that the policy makes $\Omega^{(12)}$ an invariant

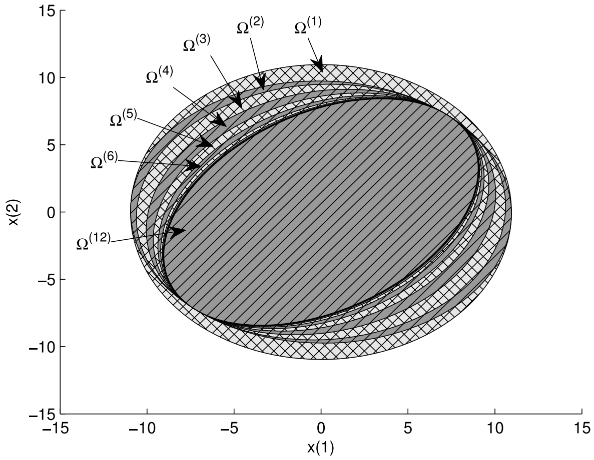


Fig. 6. Bilinear experiment: invariantly admissible regions at different iterations.

region. Vatani *et al.* [35] also gave an algorithm to improve the control performance of the synthesized controller. For comparison, the phase portraits under the initial policy and the improved policy by [35] are both depicted. The accumulated costs along the 20 trajectories by three policies are 37941.065, 44406.807, and 37728.085, respectively. The performance of IADP policy is quite close to the improved policy by [35], but our implementation does not need the knowledge of exact dynamics.

C. Nonlinear Dynamics

Now we add more nonlinearity to the dynamics in the previous experiment. The input gain matrix is set to

$$g(x_k) = \begin{bmatrix} -0.001(x_k(1))^2 + 0.001x_k(1) + 0.09 \\ 0.004(x_k(2))^2 - 0.004x_k(2) + 0.09 \end{bmatrix}$$

and the drift dynamics is unchanged. The initial stabilizing policy is $\mu^{(1)}(x) = [(\beta^{(1)}(x))/(\alpha^{(1)}(x))]$ with

$$\begin{aligned}\alpha^{(1)}(x) = {} & 0.0004(x(1))^4 - 0.0009(x(1))^3x(2) - 0.0009(x(1))^3 \\ & + 0.002(x(1))^2(x(2))^2 + 0.002(x(1))^2x(2) \\ & - 0.077(x(1))^2 - 0.008x(1)(x(2))^3 \\ & - 0.004x(1)(x(2))^2 \\ & + 0.061x(1)x(2) + 0.078x(1) + 0.013(x(2))^4 \\ & + 0.009(x(2))^3 + 0.028(x(2))^2 \\ & - 0.147x(2) + 3.515 \\ \beta^{(1)}(x) = {} & 0.001(x(1))^3 + 0.004(x(1))^2x(2) - 0.001(x(1))^2 \\ & + 0.008x(1)(x(2))^2 + 0.006x(1)x(2) - 0.090x(1) \\ & - 0.046(x(2))^3 - 0.046(x(2))^2 - 0.575x(2)\end{aligned}$$

and the stabilizing region is unchanged. The monomial vectors z_L, z_λ, and z_g are defined up to degrees 5, 3, and 2. The rest parameters follow the previous experiment.

Apply IADP algorithm to the system. The online trajectories are presented in Fig. 8. Since nonlinearity is increased, the number of determinant variables is also increased and more data are needed to formulate the model-free SDP program. The learning time is longer than the previous experiment. After nine iterations, the algorithm reaches the convergence. Value functions and invariantly admissible regions at each iteration are depicted in Figs. 9 and 10. The final converged policy has $\mu^{(9)}(x) = [(\beta^{(9)}(x))/(\alpha^{(9)}(x))]$ with

$$\begin{aligned}\alpha^{(9)}(x) = {} & 0.00004(x(1))^4 - 0.00008(x(1))^3 \\ & + 0.0002(x(1))^2(x(2))^2 - 0.0002(x(1))^2x(2) \\ & - 0.003(x(1))^2 - 0.0002x(1)(x(2))^2\end{aligned}$$

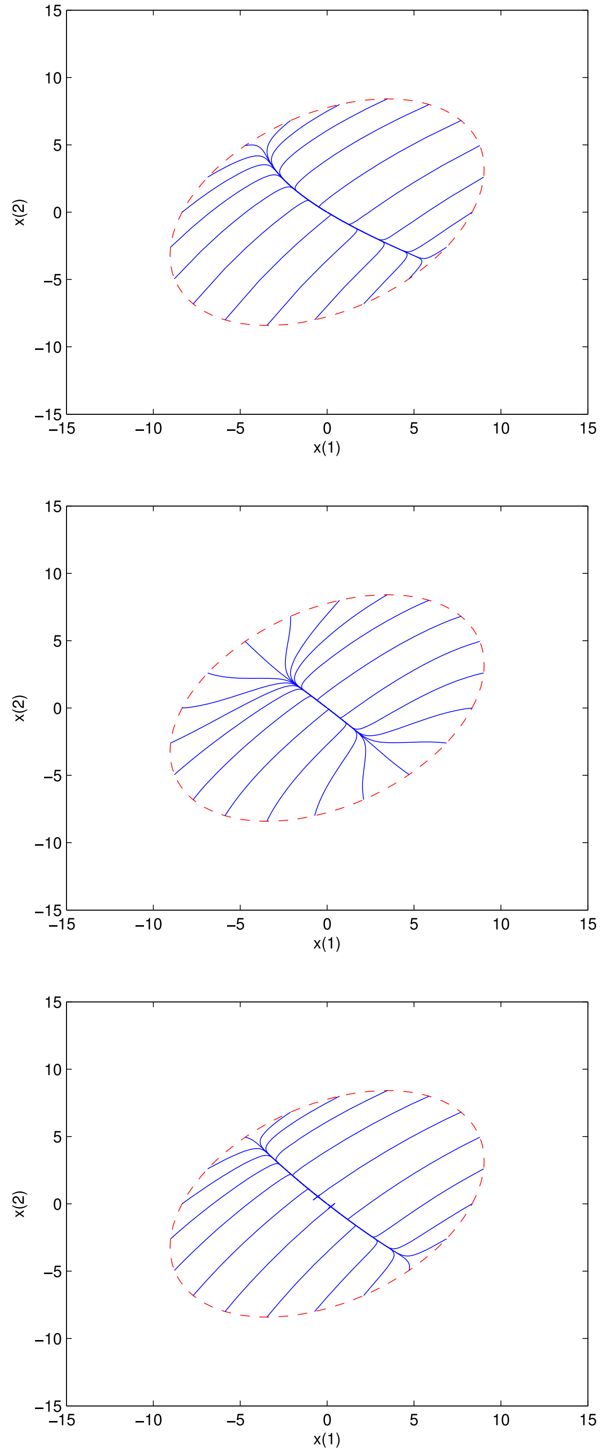


Fig. 7. Bilinear experiment: comparison of different controllers. From the top to the bottom are phase portraits under IADP policy, initial policy, improved policy by [35].

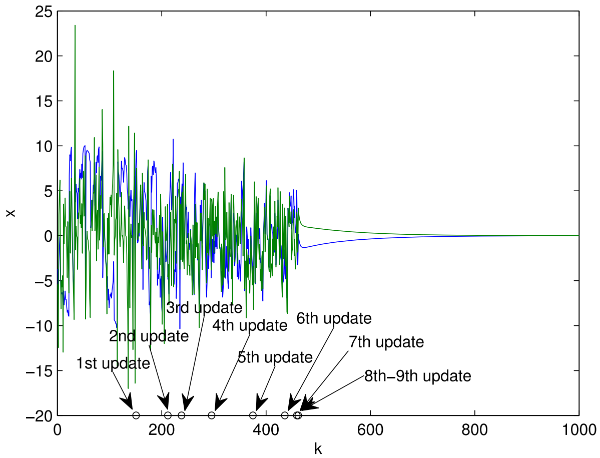


Fig. 8. Nonlinear experiment: online trajectories.

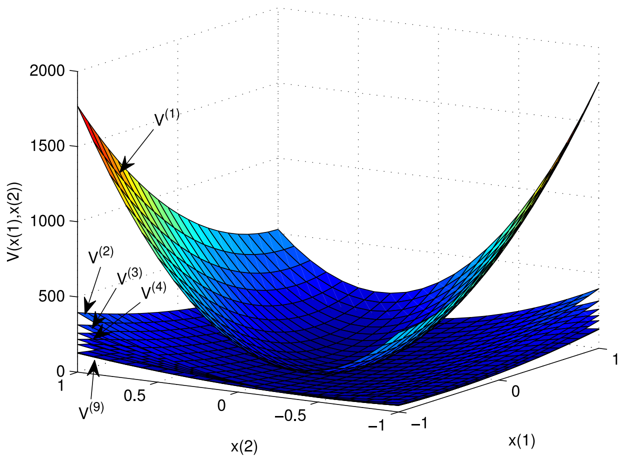


Fig. 9. Nonlinear experiment: value functions at different iterations.

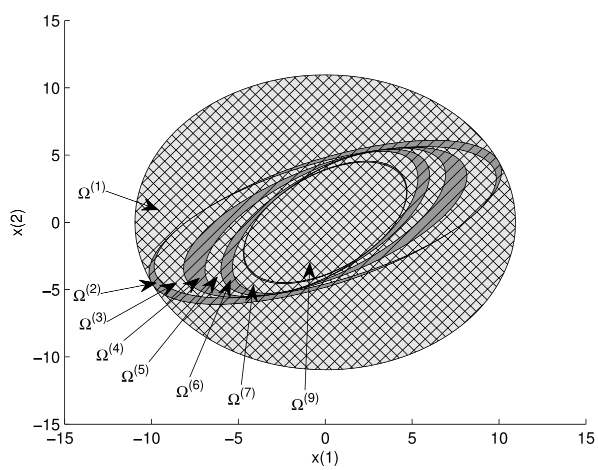


Fig. 10. Nonlinear experiment: invariantly admissible regions at different iterations.

$$
\begin{aligned}
&+ 0.0002x(1)x(2)\\
&+ 0.003x(1) + 0.0007(x(2))^4 - 0.001(x(2))^3\\
&+ 0.016(x(2))^2 - 0.015x(2) + 1.326\\
\beta^{(9)}(x) = {}& 0.040(x(1))^3 - 0.020(x(1))^2x(2) - 0.040(x(1))^2\\
&+ 0.082x(1)(x(2))^2 - 0.061x(1)x(2) - 1.728x(1)\\
&- 0.168(x(2))^3 + 0.168(x(2))^2 - 1.931x(2)
\end{aligned}
$$

and the invariantly admissible region is $\Omega^{(9)} = \{x|643.208 - 39.895(x(1))^2 + 41.822x(1)x(2) - 42.179(x(2))^2 \geq 0\}$.

VII. Conclusion

Invariant PI is studied to deal with the regionality appearing in the optimal control of discrete-time systems. Both policies and their invariantly admissible regions are updated at each iteration. Then, the QSPI algorithm is proposed to learn near-optimal policies for a class of discrete-time systems. SOS polynomials are used to ensure the feasibility of optimization. To achieve model-free learning, the IADP algorithm is further developed. Numerical experiments demonstrate that the algorithm can learn near-optimal policies and invariantly admissible regions based on data.

Only input-gain nonlinearity is considered in the algorithm, and internal dynamics is required to be linear. In many cases the whole dynamics is nonlinear and the optimal control becomes more complicated. One possible solution is to use a fuzzy model to describe the nonlinear dynamics by a group of linear dynamics with nonlinear fuzzy rules [45]. The controller design on these linear dynamics may formulate a feasible controller for the original nonlinear systems. Our future research will focus on the optimal control of systems with arbitrary nonlinear dynamics.

Appendix
Proof of Lemma 1

Since μ is invariantly admissible in Ω, the continuity of dynamics implies that $J(\cdot;\mu)$ is finite and continuous in Ω, and satisfies the Lyapunov equation. Now we use contradiction to prove the uniqueness. Suppose there exist two different solutions to (4), i.e., $V_1, V_2 \in \mathcal{C}(\Omega)$. There exists at least one point $x_0 \in \Omega$ such that the difference between V_1 and V_2 is nonzero, i.e., $\epsilon = |V_1(x_0) - V_2(x_0)| > 0$.

From (4)

$$
\begin{aligned}
V_1(x_0) - V_2(x_0) &= V_1\left(x_1^{\mu}\right) - V_2\left(x_1^{\mu}\right)\\
&\;\;\vdots\\
&= V_1\left(x_k^{\mu}\right) - V_2\left(x_k^{\mu}\right).
\end{aligned}
$$

When $k \to \infty$, $x_k^{\mu} \to 0$ and $V_1(x_k^{\mu}) \to 0$, $V_2(x_k^{\mu}) \to 0$. That means the difference $V_1(x_0) - V_2(x_0)$ can be arbitrarily close to zero, which contradicts our hypothesis. By contradiction, $J(\cdot;\mu)$ is the unique solution to (4).

References

[1] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits Syst. Mag.*, vol. 9, no. 3, pp. 32–50, Aug. 2009.

[2] H. Zhang, D. Liu, Y. Luo, and D. Wang, *Adaptive Dynamic Programming for Control: Algorithms and Stability*. London, U.K.: Springer, 2012.

[3] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA, USA: Athena Sci., 1995.

[4] R. W. Beard and T. W. McLain, "Successive Galerkin approximation algorithms for nonlinear optimal and robust control," *Int. J. Control*, vol. 71, no. 5, pp. 717–743, 1998.

[5] J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 264–276, Mar. 2001.

[6] Y. Zhu, D. Zhao, and D. Liu, "Convergence analysis and application of fuzzy-HDP for nonlinear discrete-time HJB systems," *Neurocomputing*, vol. 149, pp. 124–131, Feb. 2015.

[7] Y. Zhu, D. Zhao, X. Yang, and Q. Zhang, "Policy iteration for H_∞ optimal control of polynomial nonlinear systems via sum of squares programming," *IEEE Trans. Cybern.*, vol. 48, no. 2, pp. 500–509, Feb. 2018.

[8] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 943–949, Aug. 2008.

[9] H. Zhang, J. Zhang, G.-H. Yang, and Y. Luo, "Leader-based optimal coordination control for the consensus problem of multiagent differential games via fuzzy adaptive dynamic programming," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 1, pp. 152–163, Feb. 2015.

[10] X. Yang and H. He, "Adaptive dynamic programming for decentralized stabilization of uncertain nonlinear large-scale systems with mismatched interconnections," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published. doi: 10.1109/TSMC.2018.2837899.

[11] Y. Jiang and Z.-P. Jiang, "Robust adaptive dynamic programming and feedback stabilization of nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 882–893, May 2014.

[12] D. Wang, D. Liu, Y. Zhang, and H. Li, "Neural network robust tracking control with adaptive critic framework for uncertain nonlinear systems," *Neural Netw.*, vol. 97, pp. 11–18, Jan. 2018.

[13] Y. Zhu, D. Zhao, H. He, and J. Ji, "Event-triggered optimal control for partially unknown constrained-input systems via adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4101–4109, May 2017.

[14] Y. Zhu, D. Zhao, X. Li, and D. Wang, "Control-limited adaptive dynamic programming for multi-battery energy storage systems," *IEEE Trans. Smart Grid*, to be published. doi: 10.1109/TSG.2018.2854300.

[15] H. Zhang, Y. Liu, G. Xiao, and H. Jiang, "Data-based adaptive dynamic programming for a class of discrete-time systems with multiple delays," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published. doi: 10.1109/TSMC.2017.2758849.

[16] Y. Jiang and Z.-P. Jiang, "Global adaptive dynamic programming for continuous-time nonlinear systems," *IEEE Trans. Autom. Control*, vol. 60, no. 11, pp. 2917–2929, Nov. 2015.

[17] Q. Wei, F. L. Lewis, D. Liu, R. Song, and H. Lin, "Discrete-time local value iteration adaptive dynamic programming: Convergence analysis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 6, pp. 875–891, Jun. 2018.

[18] D. Liu, Q. Wei, and P. Yan, "Generalized policy iteration adaptive dynamic programming for discrete-time nonlinear systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 12, pp. 1577–1591, Dec. 2015.

[19] Y. Zhu, D. Zhao, and X. Li, "Using reinforcement learning techniques to solve continuous-time non-linear optimal tracking problem without system dynamics," *IET Control Theory Appl.*, vol. 10, no. 12, pp. 1339–1347, Aug. 2016.

[20] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, 2009.

[21] Y. Zhu, D. Zhao, and X. Li, "Iterative adaptive dynamic programming for solving unknown nonlinear zero-sum game based on online data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 714–725, Mar. 2017.

[22] Y. Jiang and Z.-P. Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics," *Automatica*, vol. 48, no. 10, pp. 2699–2704, 2012.

[23] Q. Zhang, D. Zhao, and Y. Zhu, "Event-triggered H_∞ control for continuous-time nonlinear system via concurrent learning," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 7, pp. 1071–1081, Jul. 2017.

[24] D. Liu and Q. Wei, "Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 621–634, Mar. 2014.

[25] Y. Zhu, D. Zhao, H. He, and J. Ji, "Convergence proof of approximate policy iteration for undiscounted optimal control of discrete-time systems," *Cogn. Comput.*, vol. 7, no. 6, pp. 763–771, 2015.

[26] D. P. Bertsekas, "Value and policy iterations in optimal control and adaptive dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 500–509, Mar. 2017.

[27] B. Luo, D. Liu, and H.-N. Wu, "Adaptive constrained optimal control design for data-based nonlinear discrete-time systems with critic-only structure," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2099–2111, Jun. 2018.

[28] B. Luo, D. Liu, T. Huang, and D. Wang, "Model-free optimal tracking control via critic-only Q-learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 10, pp. 2134–2144, Oct. 2016.

[29] J. Y. Lee, J. B. Park, and Y. H. Choi, "Integral reinforcement learning for continuous-time input-affine nonlinear systems with simultaneous invariant explorations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 5, pp. 916–932, May 2015.

[30] Q. Wei, D. Liu, Q. Lin, and R. Song, "Discrete-time optimal control via local policy iteration adaptive dynamic programming," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3367–3379, Oct. 2017.

[31] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," *Math. Program.*, vol. 96, no. 2, pp. 293–320, 2003.

[32] S. Prajna, A. Papachristodoulou, and F. Wu, "Nonlinear control synthesis by sum of squares optimization: A Lyapunov-based approach," in *Proc. 5th Asian Control Conf.*, vol. 1, Jul. 2004, pp. 157–165.

[33] B. Luo, D. Liu, H.-N. Wu, D. Wang, and F. L. Lewis, "Policy gradient adaptive dynamic programming for data-based optimal control," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3341–3354, Oct. 2017.

[34] B. Luo, Y. Yang, and D. Liu, "Adaptive Q-learning for data-based optimal output regulation with experience replay," *IEEE Trans. Cybern.*, vol. 48, no. 12, pp. 3337–3348, Dec. 2018.

[35] M. Vatani, M. Hovd, and S. Olaru, "Control design and analysis for discrete time bilinear systems using sum of squares methods," in *Proc. 53rd IEEE Conf. Decis. Control*, Los Angeles, CA, USA, Dec. 2014, pp. 3143–3148.

[36] J. Xu, L. Xie, and Y. Wang, "Synthesis of discrete-time nonlinear systems: A SOS approach," in *Proc. 26th Amer. Control Conf.*, New York, NY, USA, Jul. 2007, pp. 4829–4834.

[37] D. Coutinho, M. Fu, and A. Trofino, "Guaranteed cost analysis and control for a class of uncertain nonlinear discrete-time systems," in *Proc. Amer. Control Conf.*, vol. 3, 2003, pp. 2341–2346.

[38] G. Pin, L. Magni, T. Parisini, and D. M. Raimondo, "Robust receding—Horizon control of nonlinear systems with state dependent uncertainties: An input-to-state stability approach," in *Proc. Amer. Control Conf.*, Seattle, WA, USA, Jun. 2008, pp. 1667–1672.

[39] J. P. Rincón-Zapatero and C. Rodríguez-Palmero, "Existence and uniqueness of solutions to the Bellman equation in the unbounded case," *Econometrica*, vol. 71, no. 5, pp. 1519–1555, 2003.

[40] T. Kamihigashi, "Existence and uniqueness of a fixed point for the Bellman operator in deterministic dynamic programming," Res. Inst. Econ. Bus. Admin., Kobe Univ., Kobe, Japan, Rep. DP2011-23, 2011.

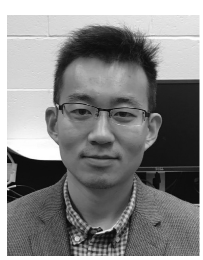
[41] A. Bacciotti and A. Biglio, "Some remarks about stability of nonlinear discrete-time control systems," *Nonlin. Differ. Equ. Appl.*, vol. 8, no. 4, pp. 425–438, Nov. 2001.

[42] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. Philadelphia, PA, USA: SIAM, 1994.

[43] A. Papachristodoulou *et al.*, "SOSTOOLS version 3.00 sum of squares optimization toolbox for MATLAB," *arXiv preprint arXiv:1310.4716*, 2013. [Online]. Available: http://arxiv.org/abs/1310.4716

[44] Y. Zhu, D. Zhao, and Z. Zhong, "Adaptive optimal control of heterogeneous CACC system with uncertain dynamics," *IEEE Trans. Control Syst. Technol.*, to be published. doi: 10.1109/TCST.2018.2811376.

[45] K. Tanaka, H. Ohtake, and H. O. Wang, "Guaranteed cost control of polynomial fuzzy systems via a sum of squares approach," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 561–567, Apr. 2009.



Yuanheng Zhu (M'15) received the B.S. degree in automation from Nanjing University, Nanjing, China, in 2010, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2015.

From 2015 to 2017, he was an Assistant Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, where he is currently an Associate Professor. From 2017 to 2018, he was a Visiting Scholar with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA. His current research interests include optimal control, adaptive dynamic programming, reinforcement learning, automatic driving, and game intelligence.

Dr. Zhu served as the Travel Grant Chair of IEEE Computational Intelligence Society in 2016. He also organized several special sessions in some international conferences.

Dongbin Zhao (M'06–SM'10) received the B.S., M.S., Ph.D. degrees in material processing engineering from the Harbin Institute of Technology, Harbin, China, in 1994, 1996, and 2000, respectively.

He was a Post-Doctoral Fellow with Tsinghua University, Beijing, China, from 2000 to 2002. Since 2002, he has been a Professor with the Institute of Automation, Chinese Academy of Sciences, Beijing, and also a Professor with the University of Chinese Academy of Sciences, Beijing. From 2007 to 2008, he was also a Visiting Scholar with the University of Arizona, Tucson, AZ, USA. He has published 4 books, and about 80 international journal papers. His current research interests include deep reinforcement learning, computational intelligence, adaptive dynamic programming, autonomous driving, robotics, intelligent transportation systems, and smart grids.

Dr. Zhao serves as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, and *IEEE Computation Intelligence Magazine*. He is the Chair of Beijing Chapter, and was the Chair of Adaptive Dynamic Programming and Reinforcement Learning Technical Committee from 2015 to 2016 and Multimedia Subcommittee from 2015 to 2016 of IEEE Computational Intelligence Society. He works as several guest editors of renowned international journals, and is involved in organizing many international conferences.

Haibo He (SM'11–F'18) received the B.S. and M.S. degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical engineering from Ohio University, Athens, OH, USA, in 2006.

From 2006 to 2009, he was an Assistant Professor with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA. He is currently the Robert Haas Endowed Chair Professor with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingstown, RI, USA. His current research interests include adaptive learning and control, computational intelligence, machine learning, data mining, and various applications.

Dr. He was a recipient of the IEEE International Conference on Communications Best Paper Award in 2014, the IEEE Computational Intelligence Society (CIS) Outstanding Early Career Award in 2014, the National Science Foundation CAREER Award in 2011, and the Providence Business News "Rising Star Innovator Award" in 2011. He was the Chair of IEEE CIS Emergent Technologies Technical Committee in 2015 and IEEE CIS Neural Networks Technical Committee in 2013 and 2014. He served as the General Chair of 2014 IEEE Symposium Series on Computational Intelligence (IEEE SSCI14, Orlando, FL, USA). He is currently the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.