Artif Intell Rev
DOI 10.1007/s10462-017-9548-4

Comprehensive comparison of online ADP algorithms for continuous-time optimal control

Yuanheng Zhu[1,2] · **Dongbin Zhao**[1,2]

Abstract Online learning is an important property of adaptive dynamic programming (ADP). Online observations contain plentiful dynamics information, and ADP algorithms can utilize them to learn the optimal control policy. This paper reviews the research of online ADP algorithms for the optimal control of continuous-time systems. With the intensive study, ADP has been developed towards model free and data efficient. After separately introducing the algorithms, we compare their performance on the same problem. This paper is desired to provide a comprehensive understanding of continuous-time online ADP algorithms.

Keywords Adaptive dynamic programming · Policy iteration · Integral reinforcement learning · Experience replay · Off-policy

1 Introduction

With decades of development, adaptive dynamic programming (ADP) (Lewis and Vrabie 2009; Wang et al. 2009; Zhang et al. 2012, 2013; Song et al. 2015; Zhu et al. 2016a, 2017a; Zhao et al. 2016) has now become a powerful method in the field of control theory for the optimal control. ADP is first proposed by Werbos (Werbos 1977), who incorporates the idea of reinforcement learning (RL) (Kaelbling et al. 1996; Sutton and Barto 1998; Ribeiro 2002) from the field of computational intelligence. It interacts with the system and learns the optimal control policy with the target of minimizing certain cost criteria. In the past, ADP mainly focuses on discrete-time (DT) systems with stochastic or deterministic transition

✉ Dongbin Zhao
dongbin.zhao@ia.ac.cn

Yuanheng Zhu
yuanheng.zhu@ia.ac.cn

1 The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

2 University of Chinese Academy of Sciences, Beijing, China

Published online: 24 February 2017

dynamics (Al-Tamimi et al. 2008; Wang et al. 2012; Zhao and Zhu 2015). In the physical world, many problems are continuous-time (CT), which makes it difficult to directly apply DT ADP algorithms to these problems.

From a mathematical viewpoint, to find the optimal control for CT systems one can solve the Hamilton–Jacobi–Bellman (HJB) equation (Bardi and Capuzzo-Dolcetta 2008; Beard et al. 1997), which is a first-order, nonlinear partial differential equation (PDE). In general, it is intractable to give a universal solution. So approximation technique has to be used to approach the solution over a compact set. Neural networks (NNs) are among the most widely used approximations. In most cases, one network is constructed to evaluate the control performance, termed as critic, and another network approximates the policy, termed as actor. When the system dynamics is known, the HJB equation can be devided into a sequence of linear PDEs by policy iteration (PI) (Beard et al. 1998; Abu-Khalaf and Lewis 2005). The coefficients are computed offline. However, this method requires sampling the dynamics, so algorithms lack interactions with the system. This problem can be overcomed by online learning. Another advantage of online learning is that it helps algorithm avoid training on unusual states and save computation resources.

Murray et al. (2002) execute a given stabilizing policy on the system and evaluate its performance by observations. The policy is then updated. After iterating between the two phases, the optimal policy is obtained. In their implementation, state derivatives must be known. After that, Vrabie and Lewis (2009) introduce integral reinforcement learning (IRL) to PI method. They use only partial system dynamics and online trajectories to implement their algorithm. The input gain matrix is needed. Motivated by that, a complete model-free method is developed by Jiang and Jiang (2014) without any dynamics knowledge. Probing noise is inserted in dynamics, so trajectories contain more dynamics information, and the algorithm can learn the optimal solution without any knowledge of dynamics.

One common feature of the above mentioned algorithms is that the policy evaluation phase and the policy improvement phase are separately conducted. In other words, when the critic is updated, the actor holds constant, and vice versa. To simplify the process, Vamvoudakis and Lewis (2010) propose a synchronous policy iteration (SPI) algorithm. The critic and the actor are updated synchronously. They further prove that the system states and critic/actor NN errors are uniformly ultimately bounded (UUB), which illustrates the convergence of the learning. The full system dynamics is needed. In many practical applications, the precise dynamics is usually unknown. One solution is to construct identifier NNs to model dynamics, such as Bhasin et al. (2013), Modares et al. (2013). The update of the critic and the actor is implemented on the basis of the identified dynamics. Notice that online trajectories contain the complete dynamics information. So the more efficient approach is to design direct online ADP algorithm that learns the optimal solution using online data. Vamvoudakis et al. (2011, 2014) combine their SPI algorithm with IRL technique. Their updating laws for critic and actor use online trajectories, so the internal dynamics is no longer needed. Modares et al. (2014) further introduce experience replay (ER) technique to accelerate the convergence rate. Past observations are repeatedly utilized to train the critic and the actor. In the literature, actuation saturation problem is particularly considered. However, input gain matrix is supposed to be known in both algorithms. Inspired by the works of Jiang and Jiang (2014), we develop a complete model-free SPI algorithm to solve the optimal tracking problems (Zhu et al. 2016b). The convergence rate is further improved by ER technique.

Even though online ADP algorithms have been fully developed, the systematic comparison of these algorithms in the perspectives of methodology and experiments are rare. This paper aims to summarize the state-of-the-art online ADP algorithms for the optimal control of CT systems. Their performance is observed in solving the same problem. Their dynamics

dependency and learning speed are also compared. The paper is organized as follows. In Sect. 2, we briefly describe the optimal control problem of CT systems. In Sect. 3, the latest online ADP algorithms are reviewed. The comparison in solving the same problem is give in Sect. 4. In the end we have the discussion and conclusion.

Notations

Throughout this paper, we use $\mathbb{R}$, $\mathbb{R}^n$, $\mathbb{R}^{n\times m}$ to denote the sets of real numbers, vectors and matrices. $\|\cdot\|$ denotes the Euclidean norm for vectors, or the induced matrix norm for matrices. $\|z\|_{\max}$ represents the upper bound of a variable vector or matrix z in the norm sense.

2 Optimal control and HJB equation

The continuous-time system considered here is described by

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

where the state $x(t) \in \mathbb{R}^n$, the control $u(t) \in \mathbb{R}^m$, the internal dynamics $f(x(t)) \in \mathbb{R}^n$, and the input gain matrix $g(x(t)) \in \mathbb{R}^{n\times m}$. We assume $f(0) = 0$ and f, g are Lipschitz continuous on a compact set $\Omega \in \mathbb{R}^n$ that contains the origin. In addition, we assume (1) is stabilizable on Ω, i.e. there exists a continuous control function $u(t)$ rendering the system asymptotically stable.

For a linear system, it is easy to verify the global asymptotic stability. But for a nonlinear system, it is generally difficult to guarantee the global asymptotic stability. It is because there may exist the discontinuity of state time derivatives and cost gradient at some points due to the dynamics nonlinearity. In Jiang and Jiang (2015), Zhu et al. (2017b), authors study the global optimal control and the global H_∞ optimal control for nonlinear CT systems. Their research is based on sum of squares (SOS) theory, but is out of scope of this paper. We here consider the general cases and restrict the state space Ω to a compact set so that the asymptotic stability and differential continuity is guaranteed.

The subject of interest is to find a state-feedback control policy that minimizes a prescribed performance criterion. For a policy $u = u(x(t))$, its *value function* is defined as an infinite horizon integral cost

$$V(x(0)) = \int_0^\infty (x^T Q x + u^T R u) d\tau \tag{2}$$

where Q and R are symmetric positive definite matrices.

Definition 1 *(Admissible)* [Beard et al. (1997), Vrabie and Lewis (2009)] A control policy $u(x)$ is defined as admissible with respect to (2) on Ω, denoted by $u \in \Psi(\Omega)$, if $u(x)$ is continuous on Ω, $u(0) = 0$, $u(x)$ stabilizes (1) on Ω and $V(x_0)$ is finite $\forall x_0 \in \Omega$.

The optimal control is to find the optimal admissible policy $u^* \in \Psi(\Omega)$ that has the lowest value for every state, called the *optimal policy*. The corresponding value function is called the *optimal value function*, denoted by $V^*(x) = \min_{u\in\Psi(\Omega)} V(x)$. We assume there exists a unique solution to the optimal control problem and V^* is continuously differentiable on Ω, i.e. $V^* \in \mathcal{C}^1(\Omega)$. For ease of expression, x is omitted in functions if there is no confusion in the context.

An infinitesimal equivalent to the value function definition (2) is the Bellman equation

$$\nabla V^T(f+gu)+x^TQx+u^TRu=0,\ V(0)=0$$

Define the Hamiltonian function as follows

$$H(x,\nabla V,u)=\nabla V^T(f+gu)+x^TQx+u^TRu \tag{3}$$

where ∇ denotes the partial derivative operator, i.e. $\nabla V=\partial V/\partial x$. The sufficient condition for the optimality is provided by the famous Hamilton–Jacobi–Bellman equation

$$\min_{u\in\Psi(\Omega)} H(x,\nabla V^*,u)=0 \tag{4}$$

According to the stationary condition, the optimal policy is constructed by the optimal value function in the form

$$u^*(x)=-\frac{1}{2}R^{-1}g^T(x)\nabla V^*(x) \tag{5}$$

After substituting into (4), the HJB equation becomes

$$[\nabla V^*]^Tf-\frac{1}{4}[\nabla V^*]^TgR^{-1}g^T\nabla V^*+x^TQx=0,\ V^*(0)=0$$

Once the solution to the HJB equation is obtained, the optimal control policy is known following (5). So the optimal control now becomes solving the HJB equation, which apparently is a nonlinear partial differential equation. It is difficult or impossible to give an analytical solution even for simple cases. An efficient approach is by policy iteration method, which involves a two-step iteration. Given an admissible policy u_0, calculate the value of the current policy in the *policy evaluation* step with

$$[\nabla V^{(i)}]^T(f+gu^{(i)})+x^TQx+[u^{(i)}]^TRu^{(i)}=0,\ V^{(i)}(0)=0 \tag{6}$$

and update in the *policy improvement* step to produce a new policy with

$$u^{(i+1)}(x)=-\frac{1}{2}R^{-1}g^T(x)\nabla V^{(i)}(x) \tag{7}$$

According to the existing literature (Beard et al. 1997; Abu-Khalaf and Lewis 2005), the new policy is admissible and has better performance. After iterating the two steps, the value and the policy sequences converge to V^* and u^*. Compared to the nonlinear nature of the HJB equation, the formula in (6) is linear and it is feasible to solve. In ADP, a critic NN is constructed to approximate the value function, while an actor NN is constructed to approximate the policy.

On the basis of PI method, numerous online algorithms are proposed to solve the optimal control of CT systems using ADP. The critic and the actor are tuned based on observations of online trajectories. Next section reviews the state-of-the-art online ADP algorithms and analyzes their own features.

3 Online ADP algorithms

3.1 Synchronous policy iteration algorithm

SPI algorithm is proposed by Vamvoudakis and Lewis (2010). In their algorithm, V is approximated by the critic NN with

$$\hat{V}(x) = \hat{W}_1^T \phi_1(x), \tag{8}$$

where $\hat{W}_1 \in \mathbb{R}^{K_1}$ are the critic NN coefficients and $\phi_1 \in \mathbb{R}^{K_1}$ include the basis functions. K_1 denotes the number of neurons in the hidden layer. According to (5), the corresponding policy is formulated by

$$u(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1(x) \hat{W}_1.$$

But they use another NN to represent the actor with

$$\hat{u}(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla \phi_1(x) \hat{W}_2 \tag{9}$$

where $\hat{W}_2$ are the actor NN coefficients. After substituting (8) and (9) into the Hamiltonian function (3), we get the approximation error as

$$e = \hat{W}_1^T \nabla \phi_1^T (f + g\hat{u}) + x^T Q x + \hat{u}^T R \hat{u} \tag{10}$$

The error reflects the accuracy of the critic and the actor towards the optimal solutions. Hence one can update the coefficients based on the error using gradient descent method. In the algorithm, the actor is applied on the system to produce online trajectories. The critic is tuned using the following updating law

$$\dot{\hat{W}}_1 = -\alpha_1 \frac{\sigma_1}{m_s^2} \left(\sigma_1^T \hat{W}_1 + x^T Q x + \hat{u}^T R \hat{u} \right) \tag{11}$$

where α_1 is the learning rate, $\sigma_1 = \nabla \phi_1^T (f + g\hat{u})$, and $m_s = \sigma_1^T \sigma_1 + 1$ is for normalization. The actor on one hand aims to approximate the critic coefficients $\hat{W}_1$, and on the other hand should stabilize the system in the compact set. So Vamvoudakis and Lewis (2010) design the following actor law

$$\dot{\hat{W}}_2 = -\alpha_2 \left\{ \left(F_2 \hat{W}_2 - F_1 \frac{\sigma_1^T}{m_s} \hat{W}_1 \right) - \frac{1}{4} \bar{D}_1(x) \hat{W}_2 \frac{\sigma_1^T}{m_s^2} \hat{W}_1 \right\} \tag{12}$$

where α_2 is the learning rate and $F_1 > 0$, $F_2 > 0$ are tuning parameters. $\bar{D}_1$ is defined as $\bar{D}_1 = \nabla \phi_1 g R^{-1} g^T \nabla \phi_1^T$.

By using Lyapunov analysis, Vamvoudakis and Lewis (2010) prove that under the update laws (11) and (12), and the control input (9), the system states and the critic/actor errors are uniformly ultimately bounded. Note that when the system is at rest ($x = 0$), $e = 0$ and $\dot{\hat{W}}_1 = 0$. The update stops working. In order to guarantee the convergence, persistency of excitation (PE) condition is necessary, requiring the system be persistently excited. So probing noise is added in the control input to excite the system.

3.2 Actor-critic-identifier SPI algorithm

In the updating laws of the SPI algorithm, the system dynamics f and g is supposed to be known. When considering unknown or uncertain problems, many researchers resort to the identification techniques to model the system dynamics. An actor-critic-identifier structure is proposed, and many kinds of identifier NNs are used. Bhasin et al. (2013) use the following multi-layer dynamic neural network (MLDNN) identifer to approximate the system (1)

$$\dot{\hat{x}} = \hat{W}_f^T \hat{\sigma}_f + g(x)\hat{u} + \mu$$

where $\hat{x} \in \mathbb{R}^n$ is the DNN state, $\hat{\sigma}_f = \sigma(\hat{V}_f^T \hat{x}) \in \mathbb{R}^{L_f+1}$, $\hat{W}_f$ and $\hat{V}_f$ are weight estimates, and $\mu \in \mathbb{R}^n$ is the robust integral of sign of the error (RISE) feedback term. Under the tuning law given by Bhasin et al. (2013), the estimated state and derivatives infinitely approach the true values when $t \to \infty$. Another identifer NN structure used by Modares et al. (2013) transforms the dynamics into a filtered regressor form. More details are available in the reference.

Since the identification method has been deeply developed independent of ADP and has gained tremendous success for various nonlinear CT and DT systems. Its detailed introduction is out of the scope of this paper. Interesting readers are suggested to the review works of Hunt et al. (1992), Cochocki and Unbehauen (1993).

3.3 Integral reinforcement learning SPI algorithm

The identification process additionally increases the computational complexity and extends the learning time. It is more desired to develop direct online ADP algorithms that learn the critic and the actor using online trajectories. For this reason, Vamvoudakis et al. (2014) combine integral reinforcement learning (IRL) with their SPI algorithm and propose the algorithm which we denote as SPI-IRL. Reviewing the Hamiltonian error defined by (10), along the system evolution $\dot{x} = f + g\hat{u}$, after integrating both sides over interval $[t - T, t]$, the integral Hamiltonian error is defined as

$$\delta = \hat{W}_1^T \left[\phi_1(x(t)) - \phi_1(x(t-T))\right] + \int_{t-T}^{t} \left(x^T Q x + \hat{u}^T R \hat{u}\right) d\tau \tag{13}$$

Based on the error, the critic can be updated by using the gradient descent method

$$\dot{\hat{W}}_1 = -\alpha_1 \frac{\Delta\phi_1}{m_s^2} \left(\hat{W}_1^T \Delta\phi_1 + \int_{t-T}^{t} (x^T Q x + \hat{u}^T R \hat{u}) d\tau\right) \tag{14}$$

where $\Delta\phi_1$ is defined as $\Delta\phi_1 = \phi_1(x(t)) - \phi_1(x(t-T))$, and $m_s = \Delta\phi_1^T \Delta\phi_1 + 1$. The actor updating law resembles the original SPI algorithm but with minor adjustment as follows

$$\dot{\hat{W}}_2 = -\alpha_2 \left\{ \left(F_2 \hat{W}_2 - F_1 T \frac{\Delta\phi_1^T}{m_s} \hat{W}_1\right) - \frac{1}{4} \bar{D}_1 \hat{W}_2 \frac{\Delta\phi_1^T}{m_s^2} \hat{W}_1 \right\} \tag{15}$$

where F_1, F_2, and $\bar{D}_1$ are defined in the same way as the SPI algorithm. It is also proved that with the critic/actor and their updating laws, the system states and the critic/actor errors are UUB.

Reviewing (11), the critic convergence rate of SPI is mainly determined by $\frac{\sigma_1 \sigma_1^T}{m_s^2}$, more precisely the minimum eigenvalue of $\frac{\sigma_1 \sigma_1^T}{m_s^2}$. As for SPI-IRL, the critic convergence rate is

determined by $\frac{\Delta\phi_1 \Delta\phi_1^T}{m_s^2}$ according to (14). It is known $\Delta\phi_1$ is the integration of σ_1 between interval $t - T$ and t, i.e.

$$\Delta\phi_1(t) = \int_{t-T}^{t} \sigma_1(\tau) d\tau$$

So $\Delta\phi_1(t)$ generally contains more data information than $\sigma_1(t)$. The minimum eigenvalue of $\frac{\Delta\phi_1 \Delta\phi_1^T}{m_s^2}$ is larger than the one of $\frac{\sigma_1 \sigma_1^T}{m_s^2}$. Hence SPI-IRL converges faster than SPI.

After the combination of IRL and SPI, the internal drift dynamics f is no longer needed, but the input gain matrix g is still necessary to define the actor. In addition, in the above updating laws, only instant online observations are utilized. Old observations in the past time, which also contain the complete dynamics information, are discarded. If the past data can be reutilized, the algorithm will exhibit high data efficiency and high learning speed.

3.4 Integral reinforcement learning and experience replay SPI algorithm

To reutilize the past data, Modares et al. (2014) adopt the idea of concurrent learning in the adaptive control, and propose the experience-replay based online algorithm. Their work is based on the SPI-IRL algorithm but considers actuation constraints. To keep the consistence of the study scope, some manipulations are made to their algorithm to fit the problem herein. Reviewing the integral Hamiltonian error in (13), it only uses the instant observation to define the error. Past observations are also capable of defining errors. Suppose the past data are stored in a history stack, with the time intervals $\{[t_j - T, t_j]\}_{j=1}^{N}$. For the past time t_j, together with the critic and the actor coefficients, its error is defined by

$$\delta_j = \hat{W}_1^T \left[\phi_1(x(t_j)) - \phi_1(x(t_j - T))\right] + \int_{t_j - T}^{t_j} \left(x^T Q x + \hat{u}^T R \hat{u}\right) d\tau$$

We have the experience-replay based gradient-descent updating law for the critic as follows

$$\begin{aligned} \dot{\hat{W}}_1 = &- \frac{\alpha_1}{N+1} \frac{\Delta\phi_1}{m_s^2} \left(\hat{W}_1^T \Delta\phi_1 + \int_{t-T}^{t} (x^T Q x + \hat{u}^T R \hat{u}) d\tau \right) \\ &- \sum_{j=1}^{N} \frac{\alpha_1}{N+1} \frac{\Delta\phi_{1j}}{m_{sj}^2} \left(\hat{W}_1^T \Delta\phi_{1j} + \int_{t_j - T}^{t_j} (x^T Q x + \hat{u}^T R \hat{u}) d\tau \right) \end{aligned} \tag{16}$$

where ϕ_{1j} is defined by the past data as $\phi_{1j} = \phi_1(x(t_j)) - \phi_1(x(t_j - T))$, and $m_{sj} = \Delta\phi_{1j}^T \Delta\phi_{1j} + 1$. The updating law of the actor is the same as (15).

In comparison to the original SPI-IRL algorithm in which only the current observation defines the updating law, the ER-based algorithm improves the data utilization. After analyzing the dynamics of critic errors, the following matrix determines the convergence of the learning process

$$\bar{H} = \frac{\Delta\phi_1 \Delta\phi_1^T}{m_s^2} + \sum_{j=1}^{N} \frac{\Delta\phi_{1j} \Delta\phi_{1j}^T}{m_{sj}^2}$$

To ensure the convergence, matrix $\bar{H}$ must be full rank. So the original PE condition required by the above algorithms now is replaced by a more easily-checked full rank condition.

3.5 Robust ADP algorithm

With the introduction of IRL technique, the internal dynamics f is no longer needed in ADP algorithms. But the input gain matrix g must be known. To overcome this, Jiang and Jiang (2014) develop a robust ADP (RADP) algorithm for the uncertain CT system. One of their contributions is to employ the off-policy idea. Review the i-th policy iteration in (6) and (7). Consider an arbitrary control input u_s and execute it on the system to produce solutions $(f + gu_s)$. Differentiate the value function V_i along the solutions and utilize the relationship $[\nabla V^{(i)}]^T g = -2[u^{(i+1)}]^T R$,

$$\begin{aligned}\dot{V}^{(i)} &= [\nabla V^{(i)}]^T \left(f + gu^{(i)} + g(u_s - u^{(i)})\right) \\ &= -2[u^{(i+1)}]^T R(u_s - u^{(i)}) - x^T Qx - [u^{(i)}]^T Ru^{(i)}\end{aligned}$$

After applying the IRL technique and making some manipulations, we get the following equation over an arbitrary interval $[t - T, t)$

$$\begin{aligned}&V^{(i)}(x(t)) - V^{(i)}(x(t-T)) \\ &+ \int_{t-T}^{t} \left(2[u^{(i+1)}]^T R(u_s - u^{(i)}) + x^T Qx + [u^{(i)}]^T Ru^{(i)}\right) d\tau = 0\end{aligned} \tag{17}$$

Note that by solving the above equation, we get the value function $V^{(i)}$ and the improved policy $u^{(i+1)}$ at one calculation. In addition, computing the equation needs no knowledge of dynamics, making it completely model-free. After getting the new policy $u^{(i+1)}$, we continue the next iteration until finding the converged optimal policy u^*. The closed-loop stability of the system is ensured by the following assumption about the control u_s.

Assumption 1 Suppose the control u_s is a stabilizing input such that the closed-loop system remains in the compact set Ω for any starting state $x(0) \in \Omega_0$, where $\Omega_0 \subseteq \Omega$.

At the beginning of Sect. 2, we have mentioned the state space is defined in a compact set Ω. The core formula (17) of Robust ADP algorithm indicates that value functions and policies are learned along the system data under the control input u_s. To guarantee the validity of data, the control input should force the system within the predefined compact set.

When using the actor-critic structure to approximate the value and the policy functions, the algorithm first collects online data from the system, and then calculates the NN coefficients based on the data. The critic and the actor are approximated by NNs independently with

$$\begin{aligned}\hat{V}^{(i)}(x) &= \hat{W}_1^T \phi_1(x) \\ \hat{u}^{(i+1)}(x) &= \hat{W}_2^T \phi_2(x)\end{aligned}$$

where $\hat{W}_1 \in \mathbb{R}^{K_1}$ and $\hat{W}_2 \in \mathbb{R}^{K_2 \times m}$ are coefficients of the critic and the actor, and ϕ_1 and ϕ_2 contain the basis functions separatively. After inserting into (17), a new error is defined as

$$\begin{aligned}e = &\hat{W}_1^T \left[\phi_1(x(t)) - \phi_1(x(t-T))\right] \\ &+ \int_{t-T}^{t} \left(2\phi_2^T \hat{W}_2 R(u_s - u^{(i)}) + x^T Qx + [u^{(i)}]^T Ru^{(i)}\right) d\tau\end{aligned}$$

By using the Kronecker product $\otimes$, the equation is rewritten into a linear form with

$$\begin{aligned}e = &\hat{W}_1^T \left[\phi_1(x(t)) - \phi_1(x(t-T))\right] + \mathbf{vec}(\hat{W}_2)^T \int_{t-T}^{t} 2\left[R(u_s - u^{(i)})\right] \otimes \phi_2 d\tau \\ &+ \int_{t-T}^{t} \left(x^T Qx + [u^{(i)}]^T Ru^{(i)}\right) d\tau\end{aligned}$$

where **vec**$(\cdot)$ is the vectorizing operator that stacks the matrix elements into a column. Given a sequence of online data with the time intervals $\{[t_j - T, t_j]\}_{j=1}^{N}$, a vector of errors are defined by

$$\begin{aligned}\begin{bmatrix} e_1 \\ \vdots \\ e_N \end{bmatrix} &= \begin{bmatrix} \Delta\phi_1^T(x(t_1)), \left[\int_{t_1-T}^{t_1} 2\big[R(u_s - u^{(i)})\big] \otimes \phi_2 d\tau\right]^T \\ \vdots \\ \Delta\phi_1^T(x(t_N)), \left[\int_{t_N-T}^{t_N} 2\big[R(u_s - u^{(i)})\big] \otimes \phi_2 d\tau\right]^T \end{bmatrix} \begin{bmatrix} \hat{W}_1 \\ \mathbf{vec}(\hat{W}_2) \end{bmatrix} \\ &\quad + \begin{bmatrix} \int_{t_1-T}^{t_1} (x^T Q x + [u^{(i)}]^T R u^{(i)}) d\tau \\ \vdots \\ \int_{t_N-T}^{t_N} (x^T Q x + [u^{(i)}]^T R u^{(i)}) d\tau \end{bmatrix} \\ &= \Theta \bar{W} + \Xi \end{aligned}$$

where $\bar{W} = [\hat{W}_1^T, \mathbf{vec}(\hat{W}_2)^T]^T$. Based on the least-squares principle, the coefficients are determined by

$$\bar{W} = -\left(\Theta^T \Theta\right)^{-1} \Theta^T \Xi \tag{18}$$

In order to get the final result, matrix $\Theta^T \Theta$ must be full rank. So during the algorithm, one needs to check the eigenvalues of $\Theta^T \Theta$ and add new observations into the history stack until meeting the full-rank condition. After computing the solution, a new policy is obtained with $u^{(i+1)} = \hat{W}_2^T \phi_2$. Then, the iteration is continued until the final converged optimal policy is obtained.

Jiang and Jiang (2014) give a complete convergence proof of their RADP algorithm. Even though the algorithm is model-free, its implementation is not in a real-time manner. The critic and the actor are not tuned along the system evolution. Their coefficients are computed and updated following a batch process with a group of online data. In the next, we present our latest research that designs a synchronous online ADP algorithm without any knowledge of dynamics.

3.6 Off-policy SPI algorithm

Now take a look at (17). If $u^{(i)}$ equals u^*, then $V^{(i)} = V^*$ and $u^{(i+1)} = u^*$. The equation becomes the following integral off-policy HJB equation

$$V^*(x(t)) - V^*(x(t-T)) + \int_{t-T}^{t} \left(2[u^*]^T R u_s - [u^*]^T R u^* + x^T Q x\right) d\tau = 0 \tag{19}$$

Similarly, we define the critic NN and the actor NN for V^* and u^* with

$$\begin{aligned} \hat{V}^*(x) &= \hat{W}_1^T \phi_1(x) \\ \hat{u}^*(x) &= \hat{W}_2^T \phi_2(x) \end{aligned}$$

After inserting into (19), we have

$$\begin{aligned} e(t) &= \hat{W}_1^T \left[\phi_1(t) - \phi_1(t-T)\right] \\ &\quad + \int_{t-T}^{t} \left(2\phi_2^T \hat{W}_2 R u_s - \phi_2^T \hat{W}_2 R \hat{W}_2^T \phi_2 + x^T Q x\right) d\tau \end{aligned} \tag{20}$$

Under the representation of Kronecker product, define the following

$$\begin{aligned} \Delta\phi_1(t) &= \phi_1(t) - \phi_1(t-T) \\ \mu(t) &= \int_{t-T}^{t} (Ru_s) \otimes \phi_2 d\tau \\ D(t) &= \int_{t-T}^{t} \left(\phi_2^T \otimes R\right) \otimes \phi_2 d\tau \\ p(t) &= \int_{t-T}^{t} x^T Q x d\tau \end{aligned} \tag{21}$$

and let

$$\eta(t) = 2\mu(t) - 2D(t)\mathbf{vec}(\hat{W}_2)$$

Then e is rewritten as

$$e(t) = \hat{W}_1^T \Delta\phi_1(t) + \mathbf{vec}(\hat{W}_2)^T \eta(t) + \mathbf{vec}(\hat{W}_2)^T D(t)\mathbf{vec}(\hat{W}_2) + p(t)$$

Under the gradient descent method, the updating laws for the critic and the actor have

$$\begin{aligned} \dot{\hat{W}}_c &= -\alpha_1 \frac{\Delta\phi_1(t)}{m_s^2(t)} e(t) \\ \mathbf{vec}(\dot{\hat{W}}_2) &= -\alpha_2 \frac{\eta(t)}{m_s^2(t)} e(t) \end{aligned}$$

where m_s is defined as $m_s = \Delta\phi_1^T \Delta\phi_1 + \eta^T \eta + 1$. The off-policy based online algorithm without system dynamics is proposed. We term it as SPI-IRL-OffPo algorithm to indicate it is a SPI algorithm based on IRL and off-policy techniques.

Inspired by Modares et al. (2014), past data are reutilized to accelerate the learning rate by the experience replay technique. Under the current coefficients $\hat{W}_1$ and $\hat{W}_2$, for arbitrary time t_j, the past data form the errors as follows

$$e(t_j) = \hat{W}_1^T \Delta\phi_1(t_j) + \mathbf{vec}(\hat{W}_2)^T \eta(t_j) + \mathbf{vec}(\hat{W}_2)^T D(t_j)\mathbf{vec}(\hat{W}_2) + p(t_j)$$

where

$$\eta(t_j) = 2\mu(t_j) - 2D(t_j)\mathbf{vec}(\hat{W}_2)$$

and $\Delta\phi_1(t_j)$, $\mu(t_j)$, $D(t_j)$, $p(t_j)$ are defined in the same way as (21). Then the experience-replay based algorithm is designed on the basis of a history data set $\left\{\left(\Delta\phi_1(t_j), \mu(t_j), D(t_j), p(t_j)\right)\right\}_{j=1}^{N}$ with the following updating laws

$$\dot{\hat{W}}_1 = -\frac{\alpha}{N+1}\left[\frac{\Delta\phi_1(t)}{m_s^2(t)} e(t) + \sum_{j=1}^{N} \frac{\Delta\phi_1(t_j)}{m_s^2(t_j)} e(t_j)\right] \tag{22}$$

$$\mathbf{vec}(\dot{\hat{W}}_2) = -\frac{\alpha}{N+1}\left[\frac{\eta(t)}{m_s^2(t)} e(t) + \sum_{j=1}^{N} \frac{\eta(t_j)}{m_s^2(t_j)} e(t_j)\right] \tag{23}$$

The new algorithm is termed as SPI-IRL-OffPo-ER. The convergence property is given as follows. According to the Weirstrass high-order approximation theorem, suppose the optimal

value function V^* and the optimal policy u^* are expressed by NNs over the compact set Ω as

$$V^*(x) = W_1^T \phi_1(x) + \varepsilon_1(x)$$
$$u^*(x) = W_2^T \phi_2(x) + \varepsilon_2(x)$$

where $W_1 \in \mathbb{R}^{K_1}$ and $W_2 \in \mathbb{R}^{K_2 \times m}$ represent the ideal coefficients of V^* and u^*, and $\varepsilon_1 \in \mathbb{R}$ and $\varepsilon_2 \in \mathbb{R}^m$ are the approximation errors. If the hidden neurons increase to infinity, the approximation errors are reduced to zero, i.e. $\varepsilon_1 \to 0$, $\varepsilon_2 \to 0$ when $K_1 \to \infty$, $K_2 \to \infty$. After inserting into the integral off-policy HJB equation (19), we get

$$\begin{aligned} \varepsilon_{HJB}(t) = {} & W_1^T [\phi_1(t) - \phi_1(t-T)] \\ & + \int_{t-T}^{t} \left(2\phi_2^T W_2 R u_s - \phi_2^T W_2 R W_2^T \phi_2 + x^T Q x\right) d\tau \end{aligned} \tag{24}$$

where

$$\begin{aligned} \varepsilon_{HJB}(t) = {} & -\varepsilon_1(t) + \varepsilon_1(t-T) \\ & + \int_{t-T}^{t} \left(-2\varepsilon_2^T R u_s + 2\varepsilon_2^T R W_2^T \phi_2 + \varepsilon_2^T R \varepsilon_2\right) d\tau \end{aligned}$$

Assumption 2 (a) g is bounded by a constant

$$\|g(x)\| \leq \|g\|_{\max}$$

(b) The critic and the actor NN approximation errors are bounded on the compact Ω

$$\|\varepsilon_1\| \leq \|\varepsilon_1\|_{\max}$$
$$\|\varepsilon_2\| \leq \|\varepsilon_2\|_{\max}$$

(c) The critic and the actor NN basis functions are bounded

$$\|\phi_1(x)\| \leq \|\phi_1\|_{\max}$$
$$\|\phi_2(x)\| \leq \|\phi_2\|_{\max}$$

Under Assumption 2, ε_{HJB} is also bounded. The following theorem gives the convergence property of SPI-IRL-OffPo-ER. Note that in our previous work Zhu et al. (2016b), we have used the algorithm to the optimal tracking control of nonlinear CT systems. Here we study the optimal control of nonlinear CT systems. For the sake of completeness, the proof of Theorem 1 is also presented.

Theorem 1 *Let W_1 be the ideal critic coefficients of V^*, W_2 be the ideal actor coefficients of u^*, and $\hat{W}_1$ and $\hat{W}_2$ be the estimations of W_1 and W_2 in the critic and the actor. Let $\rho = [\Delta\phi_1^T, \eta^T]^T$ and assume signal $\bar{\rho} = \rho/m_s$ is persistently exciting. Let Assumptions 1 and 2 hold. If the integral interval T satisfies the requirement as described in the sequel and NN coefficients are appropriately initialized, the errors $\tilde{W}_1 = W_1 - \hat{W}_1$ and $\tilde{W}_2 = W_2 - \hat{W}_2$ converge exponentially to a residual set in the neighbor of zero under the laws provided by (22) and (23).*

Proof Subtract (24) from (20)

$$e(t) = -\tilde{W}_1^T \Delta\phi_1(t) - \mathbf{vec}(\tilde{W}_2)^T \eta(t) + \mathbf{vec}(\tilde{W}_2)^T D(t) \mathbf{vec}(\tilde{W}_2) + \varepsilon_{HJB}(t)$$

The above equation also holds for past time t_j. Define the Lyapunov candidate $L = \frac{1}{2}\tilde{W}_1^T\alpha^{-1}\tilde{W}_1 + \frac{1}{2}\mathbf{vec}(\tilde{W}_2)^T\alpha^{-1}\mathbf{vec}(\tilde{W}_2)$. Its time derivative has

$$\begin{aligned}\dot{L} &= \tilde{W}_1^T\alpha^{-1}\dot{\tilde{W}}_1 + \mathbf{vec}(\tilde{W}_2)^T\alpha^{-1}\mathbf{vec}(\dot{\tilde{W}}_2)\\ &= \frac{1}{N+1}\left\{\tilde{Z}^T\frac{\rho(t)}{m_s^2(t)}\Big[-\tilde{Z}^T\rho(t) + \mathbf{vec}(\tilde{W}_2)^T D(t)\mathbf{vec}(\tilde{W}_2) + \varepsilon_{HJB}(t)\Big]\right.\\ &\quad\left. + \sum_{j=1}^{N}\tilde{Z}^T\frac{\rho(t_j)}{m_s^2(t_j)}\Big[-\tilde{Z}^T\rho(t_j) + \mathbf{vec}(\tilde{W}_2)^T D(t_j)\mathbf{vec}(\tilde{W}_2) + \varepsilon_{HJB}(t_j)\Big]\right\}\end{aligned}$$

where $\tilde{Z} = \left[\tilde{W}_1^T, \mathbf{vec}(\tilde{W}_2)^T\right]^T$. Under the assumptions that ϕ_2 is bounded and $\bar{\rho}$ satisfies the PE condition, D is bounded by

$$D(t) \le T\kappa\bar{\rho}(t)\bar{\rho}^T(t)$$

where $0 < \kappa(t) < \infty$. Then the following is inferred

$$\mathbf{vec}(\tilde{W}_2)^T D(t)\mathbf{vec}(\tilde{W}_2) \le T\kappa(t)\tilde{Z}^T\bar{\rho}(t)\bar{\rho}^T(t)\tilde{Z}$$

Define a large bound B and set T to a small value such that for arbitrary $\|\tilde{Z}\| < B$, $T\kappa(t)\left|\tilde{Z}^T\bar{\rho}(t)\right| < \varepsilon_T < 1$ where ε_T is a fixed constant. $\dot{L}$ becomes as follows

$$\begin{aligned}\dot{L} &\le -\frac{1-\varepsilon_T}{N+1}\tilde{Z}^T\left[\bar{\rho}(t)\bar{\rho}^T(t) + \sum_{j=1}^{N}\bar{\rho}(t_j)\bar{\rho}^T(t_j)\right]\tilde{Z}\\ &\quad + \frac{1}{N+1}\tilde{Z}^T\left[\frac{\rho(t)}{m_s^2(t)}\varepsilon_{HJB}(t) + \sum_{j=1}^{N}\frac{\rho(t_j)}{m_s^2(t_j)}\varepsilon_{HJB}(t_j)\right]\\ &\le -\frac{1-\varepsilon_T}{N+1}\lambda_{\min}(\bar{H})\|\tilde{Z}\|^2 + \|\tilde{Z}\|\,\|\varepsilon_{HJB}\|_{\max}\end{aligned}$$

where

$$\bar{H} = \bar{\rho}(t)\bar{\rho}^T(t) + \sum_{j=1}^{N}\bar{\rho}(t_j)\bar{\rho}^T(t_j)$$

So the estimation error $\tilde{Z}$ converges exponentially to the residual set $\left\{\tilde{Z} : \|\tilde{Z}\| \le \frac{(N+1)\|\varepsilon_{HJB}\|_{\max}}{(1-\varepsilon_T)\lambda_{\min}(\bar{H})}\right\}$. The proof is complete. □

The convergence region is related to approximation errors. If the value function and the policy are explicitly approximated by NNs, i.e. $\varepsilon_1 = 0$ and $\varepsilon_2 = 0$, then $\varepsilon_{HJB} = 0$, and $\hat{W}_1$ and $\hat{W}_2$ converge explicitly to the optimal solutions.

The convergence rate of the learning process depends on the minimum eigenvalue of $\bar{H}$, which is lower bounded by the minimum eigenvalue of the following H

$$H = \sum_{j=1}^{N}\bar{\rho}(t_j)\bar{\rho}^T(t_j)$$

The history data set can be updated by letting newly observed data replace some old data to increase $\lambda_{\min}(H)$. It will be illustrated in the simulated experiments that experience replay

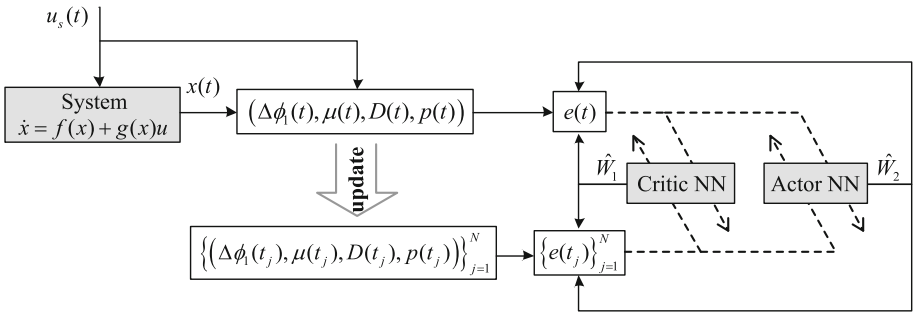

Fig. 1 The structure of the SPI-IRL-OffPo-ER algorithm

Table 1 Comparison of online ADP algorithms

	Dynamics dependency	Learning mode	Data reutilization
SPI	f, g	Real time	No
SPI-IRL	g	Real time	No
SPI-IRL-ER	g	Real time	Yes
RADP	None	Batch process	Yes
SPI-IRL-OffPo	None	Real time	No
SPI-IRL-OffPo-ER	None	Real time	Yes

will remarkably accelerate the learning rate in comparison to non-experience-replay algorithms. The structure of SPI-IRL-OffPo-ER is depicted in Fig. 1. With the IRL and off-policy methods, no dynamics is needed, and the critic/actor are tuned along system trajectories in real time. Past data are repeatedly utilized by experience replay.

The comparison of the above mentioned algorithms is summarized in Table 1. SPI tunes the critic and the actor in real time. SPI-IRL extends SPI with integral reinforcement learning to eliminate the dependency of f. SPI-IRL-ER extends SPI-IRL with experience replay to reutilize past data. RADP makes the algorithm completely model-free, but the critic and the actor are learned by a batch process. Our proposed SPI-IRL-OffPo-ER combines their advantages.

4 Numerical evaluation

The benchmark considered here for numerical evaluation is the F16 aircraft plant (Stevens and Lewis 2003; Vamvoudakis and Lewis 2010) with the linear dynamics

$$\dot{x} = \begin{bmatrix} -1.01887 & 0.90506 & -0.00215 \\ 0.82225 & -1.07741 & -0.17555 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

The cost matrices select $Q = I$ and $R = I$. As the system is linear and the cost is quadratic, the problem reduces to the linear quadratic regulator (LQR) problem and the optimal controller is linearly state-feedback in the form $u^*(x) = -K^*x$, whose gains are determined by solving the algebraic Riccati equation (ARE). The optimal value function is quadratic in the state

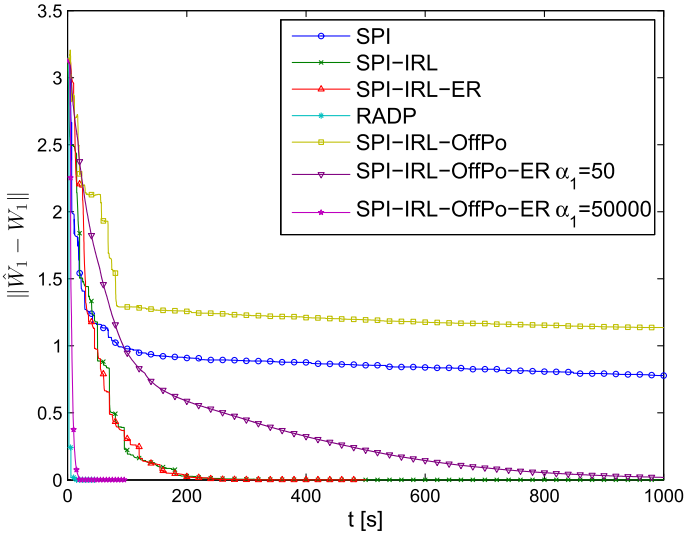

Fig. 2 Trajectories of critics by different online ADP algorithms

variables. So the critic and actor NNs select the following basis functions

$$\phi_1(x) = [x_1^2 \quad x_1x_2 \quad x_1x_3 \quad x_2^2 \quad x_2x_3 \quad x_3^2]^T$$
$$\phi_2(x) = [x_1 \quad x_2 \quad x_3]^T$$

and the optimal solutions are as follows

$$W_1 = [1.4245, 2.3364, -0.2705, 1.4349, -0.3002, 0.4329]^T$$
$$W_2 = [0.2057, 1.9089, 1.4145]^T$$

Now we apply the 6 algorithms listed in Table 1 to solve the optimal control problem. To make a fair comparison, the critic learning rates in the synchronous algorithms all select 50, and the actor learning rates are set to 1. As for RADP, we check the full-rank condition of $\Theta^T\Theta$ (18) online. Once the condition is held, we compute (18) and start the next iteration. The rest parameters are set according to their references.

Trajectories of the critic and the actor coefficients are depicted in Figs. 2 and 3. For ease of comparison we only depict the first 1000 s trajectories in these two figures. In Fig. 4, the 10,000 s trajectories with logarithmic time axis is plotted. All algorithms tend to converge to the optimal solutions if sufficient time is provided. It is because the value and the policy are explicitly approximated by NNs. SPI-IRL-OffPo algorithm has the slowest convergence rate. When it is combined with experience replay, the learning speed is improved notably, faster than SPI algorithm, but still slower than SPI-IRL and SPI-IRL-ER. The reasons are twofold. First, it does not learn from the original HJB Eq. (2), but from the off-policy HJB Eq. (19), which makes the computation less straightforward. Second, the system dynamics is completely unknown, also increasing the learning difficulty. But SPI-IRL-OffPo-ER algorithm allows much larger learning rates during the implementation, which is rarely observed in other algorithms since large learning rates can easily diverge their learning process. We repeat the SPI-IRL-OffPo-ER but set α_1 to 50,000. The learning speed is almost as fast as RADP. It is also observed that SPI is slower than SPI-IRL, and the reason has been elaborated in the above section. In Fig. 4, it is observed that the update of RADP is a jumping process,

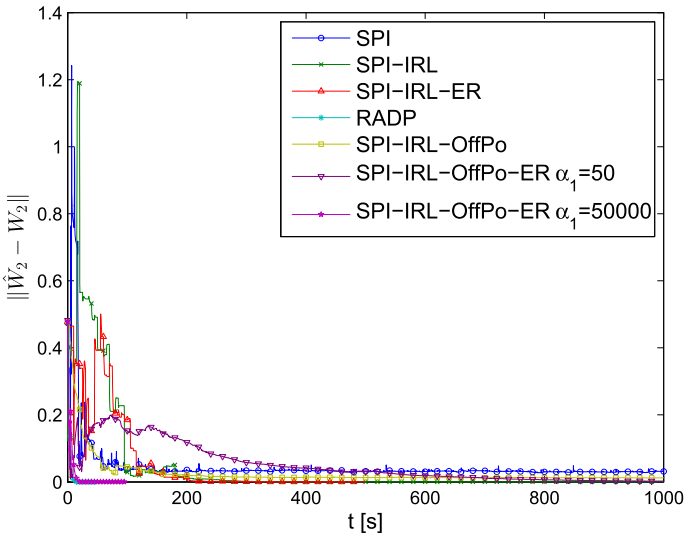

Fig. 3 Trajectories of actors by different online ADP algorithms

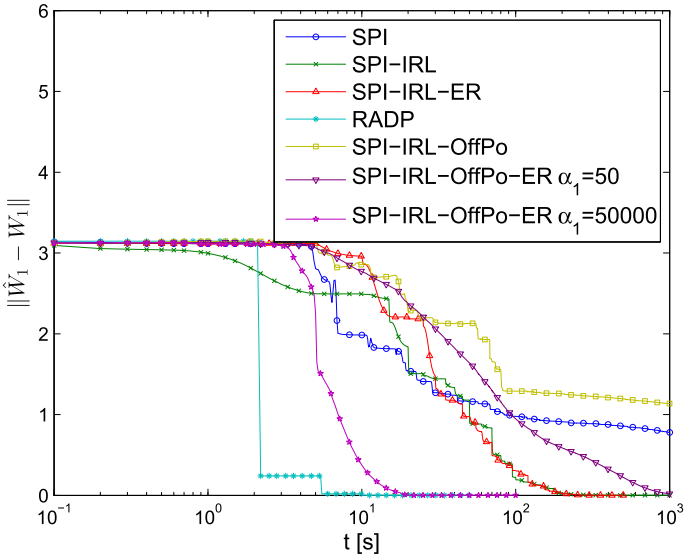

Fig. 4 Trajectories of critics by different online ADP algorithms with logarithmic time axis

since its critic and actor are learned in a batch manner. While the rest algorithms all update coefficients in real time.

Note that in our simulation, the learning rates between SPI-IRL and SPI-IRL-ER are very close. It is against the results of Modares et al. (2014) where SPI-IRL-ER exhibits faster learning rate. The discrepancy is due to the adjustment of updating laws. In this paper, the updating law (16) of SPI-IRL-ER is normalized by the history stack size. While in Modares et al. (2014), the law is not normalized, so the learning rate is improved partly because of the increase of the learning rate.

5 Conclusion

In this paper, we review the state-of-the-art online ADP algorithms. The series of synchronous policy iteration algorithms update the critic and the actor in real time during the learning process. The RADP algorithm updates the critic and the actor with a batch process and needs no knowledge of dynamics. We present the SPI-IRL-OffPo algorithm that combines the advantages of two kinds of algorithms, and its learning rate is significantly improved by experience replay.

Acknowledgements This work is supported partly by National Natural Science Foundation of China (61603382, 61573353, 61533017), and partly by the Early Career Development Award of SKLMCCS.

References

Abu-Khalaf M, Lewis FL (2005) Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach. Automatica 41(5):779–791

Al-Tamimi A, Lewis FL, Abu-Khalaf M (2008) Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. IEEE Trans Syst Man Cybern Part B Cybern 38(4):943–949

Bardi M, Capuzzo-Dolcetta I (2008) Optimal control and viscosity solutions of Hamilton–Jacobi–Bellman equations. Springer, NewYork

Beard R, McLain T et al (1998) Successive Galerkin approximation algorithms for nonlinear optimal and robust control. Int J Control 71(5):717–743

Beard RW, Saridis GN, Wen JT (1997) Galerkin approximations of the generalized Hamilton–Jacobi–Bellman equation. Automatica 33(12):2159–2177

Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor–critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. Automatica 49(1):82–92

Cochocki A, Unbehauen R (1993) Neural networks for optimization and signal processing, 1st edn. Wiley, NewYork, NY

Hunt K, Sbarbaro D, Zbikowski R, Gawthrop P (1992) Neural networks for control systemsa survey. Automatica 28(6):1083–1112

Jiang Y, Jiang ZP (2014) Robust adaptive dynamic programming and feedback stabilization of nonlinear systems. IEEE Trans Neural Netw Learn Syst 25(5):882–893

Jiang Y, Jiang ZP (2015) Global adaptive dynamic programming for continuous-time nonlinear systems. IEEE Trans Autom Control 60(11):2917–2929

Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement learning: a survey. J Artif Intell Res 4:237–285

Lewis FL, Vrabie D (2009) Reinforcement learning and adaptive dynamic programming for feedback control. IEEE Circuits Syst Mag 9(3):32–50

Modares H, Lewis FL, Naghibi-Sistani MB (2013) Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks. IEEE Trans Neural Netw Learn Syst 24(10):1513–1525

Modares H, Lewis FL, Naghibi-Sistani MB (2014) Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. Automatica 50(1):193–202

Murray JJ, Cox CJ, Lendaris GG, Saeks R (2002) Adaptive dynamic programming. IEEE Trans Syst Man Cybern Part C Appl Rev 32(2):140–153

Ribeiro C (2002) Reinforcement learning agents. Artif Intell Rev 17(3):223–250

Song R, Lewis F, Wei Q, Zhang HG, Jiang ZP, Levine D (2015) Multiple actor–critic structures for continuous-time optimal control using input–output data. IEEE Trans Neural Netw Learn Syst 26(4):851–865

Stevens BL, Lewis FL (2003) Aircraft control and simulation. Wiley, Hoboken

Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge

Vamvoudakis K, Vrabie D, Lewis F (2011) Online adaptive learning of optimal control solutions using integral reinforcement learning. In: IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL), pp 250–257

Vamvoudakis KG, Lewis FL (2010) Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem. Automatica 46(5):878–888

Vamvoudakis KG, Vrabie D, Lewis FL (2014) Online adaptive algorithm for optimal control with integral reinforcement learning. Int J Robust Nonlinear Control 24(17):2686–2710

Vrabie D, Lewis F (2009) Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. Neural Netw 22(3):237–246

Wang D, Liu D, Wei Q, Zhao D, Jin N (2012) Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. Automatica 48(8):1825–1832

Wang FY, Zhang H, Liu D (2009) Adaptive dynamic programming: an introduction. IEEE Comput Intell Mag 4(2):39–47

Werbos PJ (1977) Advanced forecasting methods for global crisis warning and models of intelligence. Gen Syst Yearb 22(6):25–38

Zhang H, Liu D, Luo Y, Wang D (2012) Adaptive dynamic programming for control: algorithms and stability. Springer, NewYork

Zhang H, Cui L, Luo Y (2013) Near-optimal control for nonzero-sum differential games of continuous-time nonlinear systems using single-network ADP. IEEE Trans Cybern 43(1):206–216

Zhao D, Zhu Y (2015) MEC—a near-optimal online reinforcement learning algorithm for continuous deterministic systems. IEEE Trans Neural Netw Learn Syst 26(2):346–356

Zhao D, Zhang Q, Wang D, Zhu Y (2016) Experience replay for optimal control of nonzero-sum game systems with unknown dynamics. IEEE Trans Cybern 46(3):854–865

Zhu Y, Zhao D, He H, Ji J (2016a) Event-triggered optimal control for partially-unknown constrained-input systems via adaptive dynamic programming. IEEE Trans Ind Electron PP(99):1

Zhu Y, Zhao D, Li X (2016b) Using reinforcement learning techniques to solve continuous-time non-linear optimal tracking problem without system dynamics. IET Control Theory Appl 10(12):1339–1347

Zhu Y, Zhao D, Li X (2017a) Iterative adaptive dynamic programming for solving unknown nonlinear zero-sum game based on online data. IEEE Trans Neural Netw Learn Syst 28(3):714–725

Zhu Y, Zhao D, Yang X, Zhang Q (2017b) Policy iteration for H_∞ optimal control of polynomial nonlinear systems via sum of squares programming. IEEE Trans Cybern PP(99):1–10