Information Sciences 411 (2017) 66–83

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Multi-step heuristic dynamic programming for optimal control of nonlinear discrete-time systems

Biao Luo[a,*], Derong Liu[b], Tingwen Huang[d], Xiong Yang[c], Hongwen Ma[a]

[a] The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] The School of Automation, Guangdong University of Technology, Guangzhou 510006, China
[c] The School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China
[d] Texas A&M University at Qatar, PO Box 23874, Doha, Qatar

ARTICLE INFO

Article history:
Received 13 October 2016
Revised 1 March 2017
Accepted 6 May 2017
Available online 15 May 2017

Keywords:
Optimal control
Multi-step heuristic dynamic programming
Adaptive dynamic programming
Nonlinear systems
Discrete-time
Neural networks

ABSTRACT

Policy iteration and value iteration are two main iterative adaptive dynamic programming frameworks for solving optimal control problems. Policy iteration converges fast while requiring an initial stabilizing control policy, which is a strict constraint in practice. Value iteration avoids the requirement of initial admissible control policy while converging much slowly. This paper tries to utilize the advantages of policy iteration and value iteration, and avoids their drawbacks at the same time. Therefore, a multi-step heuristic dynamic programming (MsHDP) method is developed for solving the optimal control problem of nonlinear discrete-time systems. MsHDP speeds up value iteration and avoids the requirement of initial admissible control policy in policy iteration at the same time. The convergence theory of MsHDP is established by proving that it converges to the solution of the Bellman equation. For implementation purpose, the actor-critic neural network (NN) structure is developed. The critic NN is employed to estimate the value function and its NN weight vector is computed with a least-square scheme. The actor NN is used to estimate the control policy and a gradient descent method is proposed for updating its NN weight vector. According to the comparative simulation studies on two examples, the effectiveness and advantages of MsHDP are verified.

1. Introduction

The bottleneck of nonlinear optimal control applications is that it depends on the solution of the Bellman equation [4,11,17], which is extremely difficult to obtain analytically or even approximately. In the past few years, adaptive dynamic programming (ADP) [2,6,7,9,10,12,13,15,19–30,33,36,39–60] has appeared as a promising method to solve the optimal control problems. Policy iteration and value iteration are two important frameworks for developing iterative ADP methods. Generally speaking, iterative ADP methods start from an initial condition, and then conduct policy evaluation and policy improvement iteratively till the convergence is achieved to design the optimal control policy. For the implementation purpose, some approximation structures, such as neural networks, are required to estimate the value function and the control policy. It

* Corresponding author.
E-mail addresses: biao.luo@hotmail.com (B. Luo), derongliu@foxmail.com (D. Liu), tingwen.huang@qatar.tamu.edu (T. Huang), xiong.yang@tju.edu.cn (X. Yang), mahongwen2012@ia.ac.cn (H. Ma).

http://dx.doi.org/10.1016/j.ins.2017.05.005
0020-0255/

is noted that the availability of a good initial condition will affect the practicability of the iterative ADP methods to some extent.

Policy iteration is one of the most popular iterative ADP framework for solving optimal control problems, which requires to start from an initial admissible control policy. The advantage of policy iteration is that it achieves a fast convergence rate, and also it guarantees that all obtained iterative control policies are admissible. In [6], the Bellman equation was successively approximated with a series of generalized Bellman equations, and the control policy update was derived based on the small perturbation assumption. For the nonlinear tracking control problem with input constraint [12], by using a discounted performance index and an augmented system, an online policy iteration algorithm was presented to learn the solution of tracking control. The convergence and stability properties of the policy iteration algorithm were analyzed for discrete-time nonlinear systems in [20]. Generalized policy iteration algorithm was proposed in [47], where an iterative loop rather than one step was employed to evaluate the value function of the current control policy. In the literature, it is noted that most of policy iteration methods focused on discrete-time systems. For continuous-time systems, Kleinman [14] proposed the famous policy iteration algorithm to solve the algebraic Riccati equation of linear continuous-time systems, and Vrabie et al. [38] proved that it is theoretically a Newton method. The algebraic Riccati equation is converted to a sequences of Lyapunov equations, which are linear matrix equations. This method was extended to nonlinear systems, where the continuous-time Hamilton–Jacobi–Bellman equation was successively approximated with a sequence of linear equations [34], which were solved with Galerkin's approximation method and its convergence was analyzed in [3]. The policy iteration methods were also employed to solve the optimal control problem of constrained systems [1] and partially unknown systems [37,38]. To provide an easy-to-check persistence of excitation condition, the experience replay technique was employed in policy iteration algorithm [31]. Although policy iteration achieves faster convergence, the requirement of an initial stabilizing control policy is a strict constraint in practice, which greatly restricts its applications.

As another iterative ADP framework, value iteration avoids requiring an initial stabilizing control policy. It starts from an arbitrary initial positive semi-definite value function, which is easy to obtain and makes the value iteration algorithms much more practical. Over the past few years, value iteration has received extensive attention and considerable works have been reported [2,8,13,59]. For the linear quadratic tracking (LQT) control problem [13], both policy iteration and value iteration methods were presented by using the data of the input, output and reference trajectory. In [2], the value iteration algorithm is proposed for affine nonlinear discrete-time systems and its convergence is proved rigorously. The actor-critic NN structure was used for implementation purpose, where actor and critic NNs were employed to approximate control policy and value function, respectively. The value iteration method and its convergence were also studied for the nonlinear Markov jump systems [59]. Feng et al. [8] proved that the iterative control laws after finite iterations can guarantee the stability of the closed-loop system. However, one serious drawback of the value iteration methods is that it converges much slower than policy iteration, and thus it requires great computational effort.

From the above discussion, it is found that both policy iteration and value iteration approaches have advantages and disadvantages. Policy iteration converges fast while suffering from the requirement of an initial admissible control policy. Value iteration starts from an easy-to-realize initial value function while it achieves a slow convergence. To overcome their drawbacks and utilize their merits at the same time, we develop a promising method, named multi-step heuristic dynamic programming (MsHDP), to solve the optimal control problem of nonlinear discrete-time systems. Through experimental simulations, the results show that MsHDP achieves much better performance than policy iteration and value iteration. The rest of the paper is arranged as follows. Section 2 gives some preliminaries about nonlinear optimal control theory and some backgrounds are given in Section 3. The MsHDP method is developed in Section 4. Section 5 demonstrates the comparative simulation results and Section 6 gives a brief conclusion.

Notation: $\mathbb{R}^n$ is the set of n-dimensional Euclidean space and $\|\cdot\|$ denotes its norm. The superscript T is used for the transpose and I denotes the identify matrix of appropriate dimension. For a symmetric matrix M, $M > (\geq) 0$ means that it is a positive definite (semi-definite) matrix. $\|v\|_M^2 \triangleq v^{\mathsf{T}} M v$ for some real vector v and symmetric matrix $M > (\geq) 0$ with appropriate dimensions. Let Ω be a compact set and $x \in \Omega$.

2. Optimal control problem

Consider the following nonlinear discrete-time system:

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

where $x(k) \in \mathbb{R}^n$ is the state and $u(k) \in \mathbb{R}^p$ is the control input. $f(x)$ is a continuous nonlinear vector function with $f(0) = 0$ and $g(x)$ is a continuous nonlinear matrix function with $g(0) = 0$. It is assumed that the system (1) is stabilizable on the set Ω.

The objective of optimal control problem is to design a state feedback control law $u(k) = u(x(k))$, such that the equilibrium point of the closed-loop system (1) is asymptotically stable, and minimize the following infinite horizon performance index:

$$J_u(x(0)) \triangleq \sum_{l=0}^{\infty} \mathcal{R}(x(l), u(x(l))), \tag{2}$$

where $\mathcal{R}(x(l), u(x(l))) \triangleq \left[\|x(l)\|_Q^2 + \|u(x(l))\|_R^2\right]$ with $Q, R > 0$. That is, the optimal control problem is formatted as:

$$\min_u J_u(x(0) = x) \tag{3}$$

and then the optimal control is determined as:

$$u^*(x) \triangleq \arg\min_u J_u(x) \tag{4}$$

for all $x \in \Omega$. For all $x(k) \in \Omega$, it follows from (2) that

$$\begin{aligned} J_u(x(k)) &= \sum_{l=k}^{\infty} \mathcal{R}(x(l), u(x(l))) = \mathcal{R}(x(k), u(x(k))) + \sum_{l=k+1}^{\infty} \mathcal{R}(x(l), u(x(l))) \\ &= \mathcal{R}(x(k), u(x(k))) + J_u(x(k+1)), \end{aligned} \tag{5}$$

where $J_u(0) = 0$. From the optimal control theory [4,17], the optimal value function $J^*(x) \triangleq J_{u^*}(x)$ is the solution of the following Bellman equation

$$J^*(x(k)) = \min_{u(x)} \{\mathcal{R}(x(k), u(x(k))) + J^*(x(k+1))\} \tag{6}$$

and the optimal control is

$$\begin{aligned} u^*(x(k)) &= \arg\min_{u(x)} \{\mathcal{R}(x(k), u(x(k))) + J^*(x(k+1))\} \\ &= \arg\min_{u(x)} \{\mathcal{R}(x(k), u(x(k))) + J^*(f(x(k)) + g(x(k))u(x(k)))\}. \end{aligned} \tag{7}$$

Thus,

$$\frac{\partial}{\partial u}\{\mathcal{R}(x(k), u(x(k))) + J^*(x(k+1))\}\Big|_{u=u^*} = 0. \tag{8}$$

According to (7) and (8),

$$u^*(x(k)) = -\frac{1}{2} R^{-1} g^{\mathrm{T}}(x(k)) \frac{\partial J^*(x(k+1))}{\partial x(k+1)}. \tag{9}$$

Substituting (9) into (6), the Bellman equation is given by

$$J^*(x(k)) = J^*(x(k+1)) + \|x(k)\|_Q^2 + \frac{1}{4} \frac{\partial J^{*\mathrm{T}}(x(k+1))}{\partial x(k+1)} g(x(k)) R^{-1} g^{\mathrm{T}}(x(k)) \frac{\partial J^*(x(k+1))}{\partial x(k+1)}. \tag{10}$$

Note that the optimal control (9) depends on the solution $J^*(x)$ of (10). The Bellman equation is a nonlinear difference equation, which is extremely difficult to solve analytically or even approximately.

3. Backgrounds

It is known that policy iteration [16,20,31,46,47] and value iteration [2,16,18,44,45,60] algorithms are two main frameworks for solving the optimal control problems. To analyze the advantages and disadvantages of policy iteration and value iteration, it is necessary to have a brief review of the two algorithms. First, the policy iteration algorithm is given as follows.

Algorithm 1 Policy Iteration.

- *Step 1:* Choose an initial admissible control policy $u^{(0)}(x)$ and let $i = 0$.
- *Step 2:* (**Policy evaluation**) Solve the equation

$$V^{(i)}(x(k)) = \mathcal{R}(x(k), u^{(i)}(x(k))) + V^{(i)}(x(k+1)) \tag{11}$$

  for value function $V^{(i)}(x)$.
- *Step 3:* (**Policy improvement**) Update control policy by

$$u^{(i+1)}(x(k)) = -\frac{1}{2} R^{-1} g^{\mathrm{T}}(x(k)) \frac{\partial V^{(i)}(x(k+1))}{\partial x(k+1)}. \tag{12}$$

- *Step 4:* If $V^{(i)}(x) \equiv V^{(i-1)}(x)$, terminate iteration, else, let $i = i + 1$, go back to Step 2 and continue. □

Note that the policy iteration algorithm requires the initial control policy $u^{(0)}(x)$ to be admissible, which is a strict condition. For many complicated practical systems, this condition is difficult to satisfy, which restricts the application of policy iteration. Till present, finding an admissible control policy for policy iteration still remains an open issue when studying the optimal control problems with ADP method. Even so, policy iteration algorithm still has an important advantage that it converges fast once an initial admissible control is determined.

Next, we give an introduction to value iteration. The procedure of the algorithm is presented as follows.

Algorithm 2 Value iteration.

- *Step 1:* Choose $V^{(0)}(x)$ and let $i = 0$.
- *Step 2:* (**Policy improvement**) Update control policy by

$$u^{(i)}(x(k)) = -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x(k))\frac{\partial V^{(i)}(x(k+1))}{\partial x(k+1)}. \tag{13}$$

- *Step 3:* (**Policy evaluation**) Calculate

$$V^{(i+1)}(x(k)) = \mathcal{R}(x(k), u^{(i)}(x(k))) + V^{(i)}(x(k+1)). \tag{14}$$

- *Step 4:* If $V^{(i+1)}(x) \equiv V^{(i)}(x)$, terminate iteration, else, let $i = i + 1$, go back to Step 2 and continue. □

For the value iteration algorithm, its initial condition is easy to satisfy, which makes the algorithm much more practicable than the policy iteration algorithm. However, value iteration suffers from the lower convergence speed compared with policy iteration.

4. Development of multi-step heuristic dynamic programming

First, the MsHDP algorithm is developed to utilize their advantages and avoid disadvantages, where the multi-step scheme is employed for policy evaluation. For the implementation of the MsHDP algorithm, the actor-critic structure is developed by using actor and critic NNs to approximate control policy and value function, and then the NN weight update laws are derived. Subsequently, the developed MsHDP method is simplified for linear quadratic regulation (LQR) problem.

4.1. Multi-step heuristic dynamic programming

It is observed that both policy iteration and value iteration algorithms have advantages and disadvantages. Policy iteration converges fast while suffering from the requirement of an initial admissible control policy. Value iteration starts from an easy-to-realize initial value function while it achieves a slow convergence. Therefore, one natural question is whether it is possible to achieve the trade-off between value iteration and policy iteration. In other words, it is desired to find a method to achieve the goal that converges faster than value iteration and avoids the requirement of initial admissible control policy at the same time. To achieve such goal, the following MsHDP algorithm is proposed.

Algorithm 3 Multi-step heuristic dynamic programming.

- *Step 1:* Give $V^{(0)}(x)$ and let $i = 0$.
- *Step 2:* (**Policy improvement**) Update control policy $u^{(i)}(x)$ by

$$u^{(i)}(x(k)) = -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x(k))\frac{\partial V^{(i)}(x(k+1))}{\partial x(k+1)}. \tag{15}$$

- *Step 3:* (**Multi-step policy evaluation**) Calculate

$$V^{(i+1)}(x(k)) = \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l), u^{(i)}(x(l))) + V^{(i)}(x(k+\beta)). \tag{16}$$

- *Step 4:* If $V^{(i+1)}(x) \equiv V^{(i)}(x)$, terminate iteration, else, let $i = i + 1$, go back to Step 2 and continue. □

It is necessary to have a discussion about the differences among the value iteration, policy iteration and the MsHDP algorithm. Moreover, the advantages of using "multi-step" in the MsHDP algorithm and their reasons will be analyzed. From Algorithms 1–3, it is noted that policy iteration, value iteration and MsHDP algorithm use the same scheme for policy improvement. That is, for the obtained value function $V(x)$, the control policy is improved with:

$$\mu(x) = \arg\min_{u(x)}\{\mathcal{R}(x(k), u(x(k))) + V(x(k+1))\} \tag{17}$$

i.e.,

$$\mu(x) = -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x(k))\frac{\partial V(x(k+1))}{\partial x(k+1)}. \tag{18}$$

However, three algorithms behaves differently for their policy evaluation operation. For the policy evaluation in policy iteration, it follows from (11) that

$$\begin{aligned} V^{(i)}(x(k)) &= \mathcal{R}(x(k), u^{(i)}(x(k))) + V^{(i)}(x(k+1)) \\ &= \mathcal{R}(x(k), u^{(i)}(x(k))) + \mathcal{R}(x(k+1), u^{(i)}(x(k+1))) + V^{(i)}(x(k+2)) \\ &\vdots \\ &= \sum_{l=0}^{\infty} \mathcal{R}(x(l), u^{(i)}(x(l))). \end{aligned} \tag{19}$$

This means that $V^{(i)}(x)$ is the cost function of the control policy $u^{(i)}(x)$. Note that in (19), $V^{(i)}(x)$ is essentially an infinite-sum of utility function $\mathcal{R}(x(l), u^{(i)}(x(l)))$. That is why the initial control policy $u^{(0)}(x)$ and all iterative control policies $u^{(i)}(x)$ in the policy iteration should be stabilizing. Otherwise, $V^{(i)}(x)$ will go to infinity, and thus it becomes meaningless. For the value iteration algorithm, it follows from (14) that $V^{(i+1)}(x)$ is essentially not the cost function of the control policy $u^{(i)}(x)$. $V^{(i+1)}(x)$ uses only one-step utility function of the current control policy $u^{(i)}(x)$, and then uses the previous value function $V^{(i)}(x)$ thereafter. Hence, it is not strange that policy iteration converges faster than value iteration. In this way, for the value iteration algorithm, the finiteness of $V^{(i+1)}(x)$ can be guaranteed for a finite $V^{(i)}(x)$, and thus the requirement of initial admissible control policy is removed. For Algorithm 3, MsHDP uses multi-step scheme for policy evaluation, which is different from that in the value iteration algorithm with one-step scheme for policy evaluation.

From the discussions above, it is found that the main differences between policy iteration, value iteration and MsHDP algorithm are that they use different schemes for policy evaluation. To summarize, in policy iteration, $V^{(i)}(x)$ is essentially an infinite-sum of utility function, i.e., $\sum_{l=0}^{\infty} \mathcal{R}(x(l), u^{(i)}(x(l)))$. Thus, it has the advantage of fast convergence and have disadvantage of the requirement of an initial stabilizing control policy. In value iteration, $V^{(i)}(x)$ is the sum of the previous $V^{(i-1)}(x)$ and the one-step utility function, i.e., $\mathcal{R}(x(k), u^{(i-1)}(x(k)))$. Then, it has the advantage of without requiring an initial stabilizing control policy and has the disadvantage of slow convergence. In the MsHDP algorithm, it uses the same iterative scheme as value iteration to avoid requiring an initial stabilizing control policy. Moreover, it sums multi-step utility function, i.e., $\sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l), u^{(i)}(x(l)))$, to accelerate the convergence compared with value iteration. Therefore, MsHDP algorithm can achieve a trade-off between policy iteration and value iteration. It has the advantage of without requiring an initial stabilizing control policy compared with policy iteration, and has the advantage of fast convergence compared with value iteration.

To show the convergence of Algorithm 3, it is proved in the following Theorem 1 that the generated sequence $\{V^{(i)}(x)\}$ converges to the solution $J^*(x)$ of Bellman Eq. (6).

Theorem 1. *Let the sequence $\{V^{(i)}(x)\}$ be generated by Algorithm 3. If the condition $V^{(0)}(x(k)) \geqslant \min_{u(x)}\{\mathcal{R}(x(k), u(x(k))) + V^{(0)}(x(k+1))\}$ holds, then,*

(1) For all i,

$$V^{(i+1)}(x(k)) \leqslant \min_{u(x)}\{\mathcal{R}(x(k), u(x(k))) + V^{(i)}(x(k+1))\}. \leqslant V^{(i)}(x(k)) \tag{20}$$

(2) $\lim_{i\to\infty} V^{(i)}(x(k)) = J^*(x)$.

Proof. With the condition, we have

$$\begin{aligned} V^{(1)}(x(k)) &= \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l), u^{(0)}(x(l))) + V^{(0)}(x(k+\beta)) \\ &= \sum_{l=k}^{k+\beta-2} \mathcal{R}(x(l), u^{(0)}(x(l))) + \mathcal{R}(x(k+\beta-1), u^{(0)}(x(k+\beta-1))) + V^{(0)}(x(k+\beta)) \\ &= \sum_{l=k}^{k+\beta-2} \mathcal{R}(x(l), u^{(0)}(x(l))) + \min_{u(x)}\{\mathcal{R}(x(k+\beta-1), u(x(k+\beta-1))) + V^{(0)}(x(k+\beta))\} \\ &\leqslant \sum_{l=k}^{k+\beta-2} \mathcal{R}(x(l), u^{(0)}(x(l))) + V^{(0)}(x(k+\beta-1)) \\ &\vdots \\ &\leqslant \mathcal{R}(x(k), u^{(0)}(x(k))) + V^{(0)}(x(k+1)) \\ &= \min_{u(x)}\{\mathcal{R}(x(k), u(x(k))) + V^{(0)}(x(k+1))\}. \end{aligned} \tag{21}$$

Combining the condition and (21) yields that

$$V^{(1)}(x(k)) \leqslant \min_{u(x)}\{\mathcal{R}(x(k),u(x(k))) + V^{(0)}(x(k+1))\} \leqslant V^{(0)}(x(k)) \tag{22}$$

which implies that the conclusion (20) holds for $i = 0$.

Assume that the conclusion (20) holds for index $i-1$, i.e.,

$$V^{(i)}(x(k)) \leqslant \min_{u(x)}\{\mathcal{R}(x(k),u(x(k))) + V^{(i-1)}(x(k+1))\} \leqslant V^{(i-1)}(x(k)). \tag{23}$$

Then,

$$\begin{aligned}
V^{(i)}(x(k)) &= \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l),u^{(i-1)}(x(l))) + V^{(i-1)}(x(k+\beta)) \\
&\geqslant \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l),u^{(i-1)}(x(l))) + \min_{u(x)}\{\mathcal{R}(x(k+\beta),u(x(k+\beta))) + V^{(i-1)}(x(k+\beta+1))\} \\
&= \sum_{l=k}^{k+\beta} \mathcal{R}(x(l),u^{(i-1)}(x(l))) + V^{(i-1)}(x(k+\beta+1)) \\
&= \mathcal{R}(x(k),u^{(i-1)}(x(k))) + V^{(i)}(x(k+1)) \\
&\geqslant \min_{u(x)}\{\mathcal{R}(x(k),u(x(k))) + V^{(i)}(x(k+1))\}.
\end{aligned} \tag{24}$$

By using (16) and (24), we have

$$\begin{aligned}
V^{(i+1)}(x(k)) &= \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l),u^{(i)}(x(l))) + V^{(i)}(x(k+\beta)) \\
&= \sum_{l=k}^{k+\beta-2} \mathcal{R}(x(l),u^{(i)}(x(l))) + \mathcal{R}(x(k+\beta-1),u^{(i)}(x(k+\beta-1))) + V^{(i)}(x(k+\beta)) \\
&= \sum_{l=k}^{k+\beta-2} \mathcal{R}(x(l),u^{(i)}(x(l))) + \min_{u(x)}\{\mathcal{R}(x(k+\beta-1),u(x(k+\beta-1))) + V^{(i)}(x(k+\beta))\} \\
&\leqslant \sum_{l=k}^{k+\beta-2} \mathcal{R}(x(l),u^{(i)}(x(l))) + V^{(i)}(x(k+\beta-1)) \\
&\vdots \\
&\leqslant \mathcal{R}(x(k),u^{(i)}(x(k))) + V^{(i)}(x(k+1)) \\
&= \min_{u(x)}\{\mathcal{R}(x(k),u(x(k))) + V^{(i)}(x(k+1))\}.
\end{aligned} \tag{25}$$

Then, the combination of (24) and (25) shows that the conclusion (20) holds for all i.

2) From (20), $\{V^{(i)}(x)\}$ is a nonincreasing sequence and lower bounded by $V^{(i)}(x) \geq 0$. Considering the bounded monotone sequence always has a limit, denote the limit by $V^{(\infty)}(x) \triangleq \lim_{i\to\infty} V^{(i)}(x)$. Taking limit on (20) yields

$$V^{(\infty)}(x(k)) \leqslant \min_{u(x)}\{\mathcal{R}(x(k),u(x(k))) + V^{(\infty)}(x(k+1))\} \leqslant V^{(\infty)}(x(k))$$

i.e.,

$$V^{(\infty)}(x(k)) = \min_{u(x)}\{\mathcal{R}(x(k),u(x(k))) + V^{(\infty)}(x(k+1))\}. \tag{26}$$

Note that (26) is essentially the same as the Bellman Eq. (6), i.e., $V^{(\infty)}(x) = J^*(x)$. □

Remark 1. For policy iteration and value iteration of continuous-time systems, Lewis and Vrabie [16] have given the important statement that "The reinforcement learning time interval T need not be the same at each iteration. T can be changed depending on how long it takes to get meaningful information from the observations." The thought of the statement is similar to the multi-step policy evaluation in MsHDP to some extent. However, the reference [16] did not give further discussion and theoretical analysis about this issue. For the MsHDP developed in this paper, compared with policy iteration and value

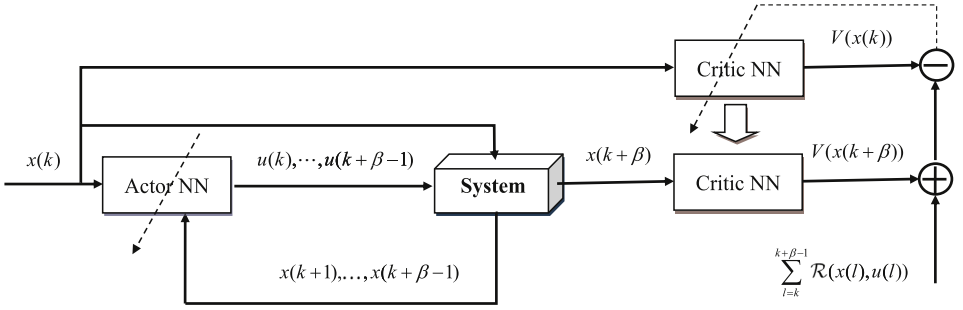

Fig. 1. The actor-critic NN strucute of MsHDP algorithm.

iteration, the differences and advantages of MsHDP are analyzed properly. Moreover, the convergence theory of MsHDP is established by proving that it converges to the solution of the Bellman equation. □

4.2. Implementation of MsHDP with neural networks

In this subsection, the actor-critic NN structure is developed to implement the MsHDP algorithm. The actor-critic NN structure is shown in Fig. 1. The critic NN is employed to approximate the value function. Let $\Phi(x) \triangleq [\phi_1(x), \cdots, \phi_{L_V}(x)]^{\mathrm{T}}$ be the critic NN activation function vector, where L_V is the number of neurons in the critic NN hidden layer. Then, the output of the critic neural network is given by

$$\hat{V}^{(i)}(x) = \sum_{j=1}^{L_V} \theta_{V,j}^{(i)} \phi_j(x) = \Phi^{\mathrm{T}}(x)\theta_V^{(i)}, \tag{27}$$

where $\theta_V^{(i)} \triangleq [\theta_{V,}^{(i)}, \cdots, \theta_{V,L_V}^{(i)}]^{\mathrm{T}}$ denotes the critic NN weight vector. By using (27), it follows from (16) that

$$\Phi^{\mathrm{T}}(x(k))\theta_V^{(i+1)} = \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l), u^{(i)}(x(l))) + \Phi^{\mathrm{T}}(x(k+\beta))\theta_V^{(i)}. \tag{28}$$

To compute the unknown critic NN weight vector $\theta_V^{(i+1)}$, the least-square method is developed. Let $\mathcal{S}_M \triangleq \{x_{[j]} | x_{[j]}(k) \in \Omega, j = 1, 2, \ldots, M\}$ be the sample set on domain Ω, where M is the size of $\mathcal{S}_M$. For all $x_{[j]}(k)$, $x_{[j]}(k+\beta)$ denotes the β-step forward state of the system (1) started from $x_{[j]}(k)$. For each sample $x_{[j]}$ in $\mathcal{S}_M$, the Eq. (28) becomes

$$\Phi^{\mathrm{T}}(x(k))\theta_V^{(i+1)} = \eta_{[j]}^{(i)}, j = 1, 2, \ldots, M, \tag{29}$$

where $\eta_{[j]}^{(i)} \triangleq \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l), u^{(i)}(x(l))) + \Phi^{\mathrm{T}}(x_{[j]}(k+\beta))\theta_V^{(i)}$. Note that (29) is a linear equation with respect to the unknown the critic NN weight vector $\theta_V^{(i+1)}$. Then, $\theta_V^{(i+1)}$ can be computed with the following least-square scheme

$$\theta_V^{(i+1)} = (Z^{\mathrm{T}}Z)^{-1}Z^{\mathrm{T}}\eta^{(i)}, \tag{30}$$

where $Z \triangleq [\Phi(x_{[1]}(k)), \cdots, \Phi(x_{[M]}(k))]^{\mathrm{T}}$ and $\eta^{(i)} \triangleq [\eta_{[1]}^{(i)}, \cdots, \eta_{[M]}^{(i)}]^{\mathrm{T}}$.

Next, the actor NN is applied to estimate the control policy. Let $\Psi(x) \triangleq [\psi_1(x), \cdots, \psi_{L_u}(x)]^{\mathrm{T}}$ be the actor NN activation function vector, where L_u is the number of neurons in the actor NN hidden layer. Then, the output of the actor neural network is given by

$$\hat{u}^{(i)}(x) = \sum_{j=1}^{L_u} \theta_{u,j}^{(i)} \psi_j(x) = \Psi^{\mathrm{T}}(x)\theta_u^{(i)}, \tag{31}$$

where $\theta_u^{(i)} \triangleq [\theta_{u,}^{(i)}, \cdots, \theta_{u,L_u}^{(i)}]^{\mathrm{T}}$ is the actor NN weight vector. By using the same method for computing the actor NN weight vector $\theta_u^{(i+1)}$ in [2], the gradient descent method for updating $\theta_u^{(i+1)}$ is given by

$$\theta_u^{(i)}|_{m+1} = \theta_u^{(i)}|_m - \alpha\Psi(x(k))\left[2R\hat{u}^{(i)}(x(k), \theta_u^{(i)}|_m) + g^{\mathrm{T}}(x(k))\frac{\partial\Phi(x(k+1))}{\partial x(k+1)}\theta_V^{(i)}\right], \tag{32}$$

where $\alpha > 0$ is the gain. With the increase of m, the actor NN weight vector $\theta_u^{(i)}|_m$ will approach its ideal value $\theta_u^{(i)}$.

For the implementation procedure of the MsHDP algorithm, it requires to compute the critic and actor NN weight vectors with (30) and (32) iteratively until the desired convergence accuracy is achieved. The implementation of the MsHDP method

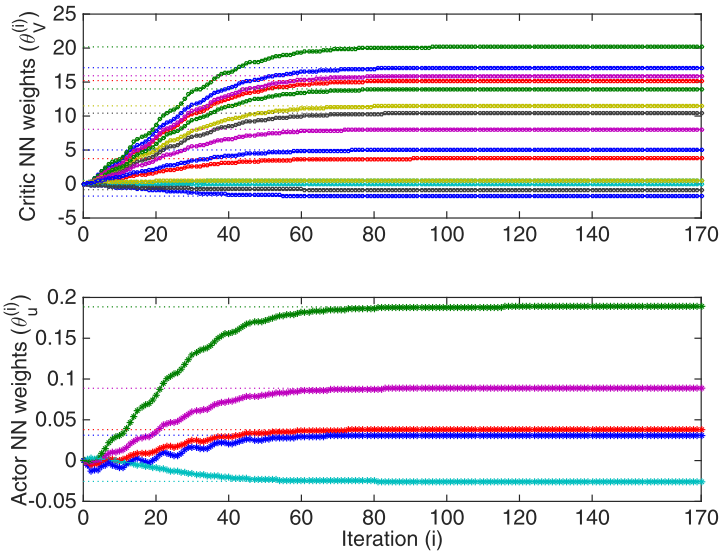

Fig. 2. For example 1, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of value iteration (i.e., $\beta = 1$) .

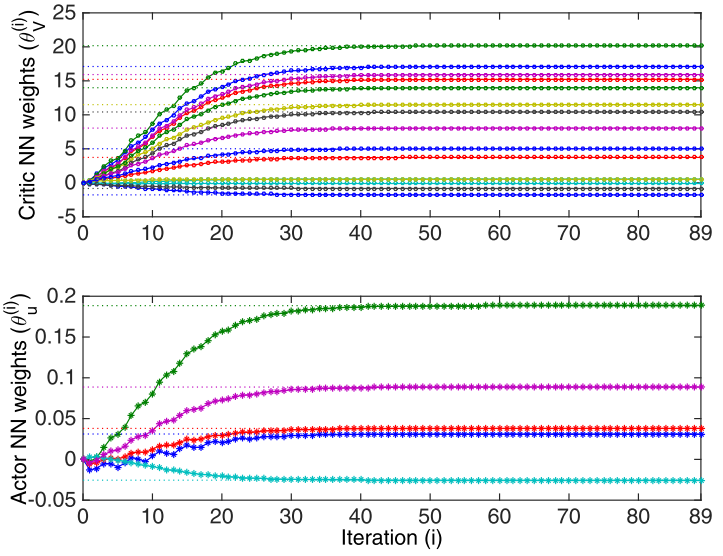

Fig. 3. For example 1, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of MsHDP with $\beta = 2$.

is realized offline by using the set $\mathcal{S}_M$ pre-collected on domain Ω before learning. After the convergence of the MsHDP algorithm is achieved, the converged control policy is employed for real-time system control with (31). The implementation of the MsHDP algorithm is presented as follows.

Algorithm 4 (Implementation Procedure of MsHDP)**.**

- ► *Step 1:* Collect the sample data set $\mathcal{S}_M$. Give $V^{(0)}(x)$ and let $i = 0$.
- ► *Step 2:* Compute the actor NN weight vector $\theta_u^{(i)}$ with (32).
- ► *Step 3:* Compute the critic NN weight vector $\theta_V^{(i+1)}$ with (30).
- ► *Step 4:* If $\|\theta_u^{(i)} - \theta_u^{(i-1)}\| \leqslant \epsilon_1$ and $\|\theta_V^{(i+1)} - \theta_V^{(i)}\| \leqslant \epsilon_2$ (ϵ_1 and ϵ_2 are small positive numbers), terminate the iteration, else, let $i = i + 1$, go back to Step 2 and continue. □

Remark 2. In machine learning community, the thought of multi-step policy evaluation has been applied in modified policy iteration [5,32,35]. Compared with modified policy iteration, the MsHDP method proposed in this paper has the following differences and contributions. First, modified policy iteration was mainly for Markov decision processes with discounted

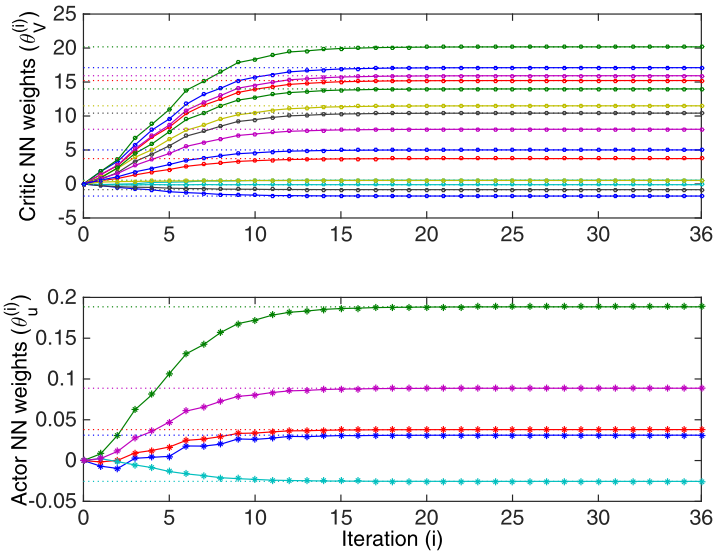

Fig. 4. For example 1, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of MsHDP with $\beta = 5$.

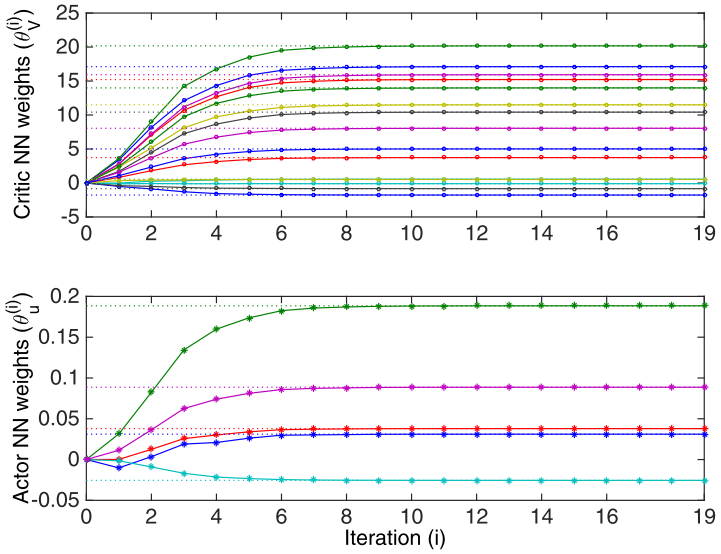

Fig. 5. For example 1, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of MsHDP with $\beta = 10$.

factor. MsHDP is developed for solving the optimal control problems without discount factor in the performance index. The main objective of MsHDP is to overcome the disadvantage of policy iteration requiring the initial admissible control policy and accelerate value iteration. Second, the convergence of the developed MsHDP algorithm is proved in Theorem 1, where MsHDP converges to the solution of the Bellman equation of optimal control problem. In addition, although modified policy iteration has been well studied in machine learning community, its thought has rarely been introduced to solve optimal control problems in control community. In the past few years, many reinforcement learning methods, such as, policy iteration and value iteration, have been introduced to handle control design problems. The use of multi-step policy evaluation in the MsHDP to solve optimal control problems is still new in control community, and it is meaningful and important. □

4.3. MsHDP for special case: the LQR problem

To help readers understand the developed MsHDP method, it is presented for the LQR problem in this subsection. Consider the linear version of system (1)

$$x(k+1) = Ax(k) + Bu(k), \tag{33}$$

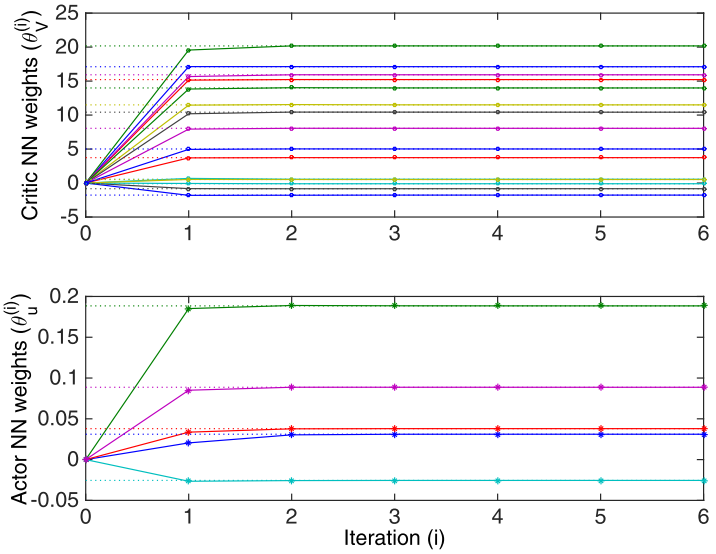

Fig. 6. For example 1, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of MsHDP with $\beta = 30$.

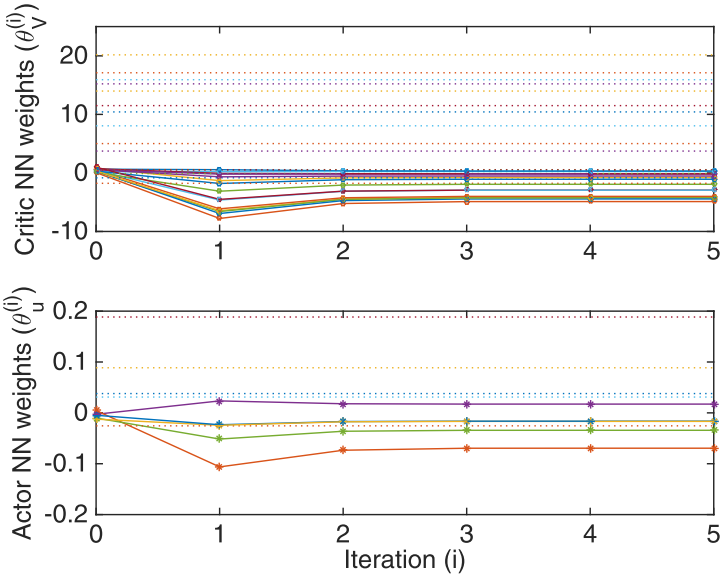

Fig. 7. For example 1, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of policy iteration.

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times p}$. For the LQR problem of the system (33) with performance index (2), its optimal value function is $J^*(x) = x^{\mathrm{T}}Px$, where $P \geq 0$. It follows from (9) that

$$\begin{aligned} u^*(x(k)) &= -R^{-1}B^{\mathrm{T}}Px(k+1) \\ &= -R^{-1}B^{\mathrm{T}}P[Ax(k) + Bu^*(x(k))]. \end{aligned}$$

Then,

$$u^*(x) = Kx, \tag{34}$$

where $K = -(R + B^{\mathrm{T}}PB)^{-1}B^{\mathrm{T}}PA$ is the optimal control gain. According to (6) and (34), we have that P satisfies the following algebraic Riccati equation:

$$P = A^{\mathrm{T}}PA + Q - A^{\mathrm{T}}PB(R + B^{\mathrm{T}}PB)^{-1}B^{\mathrm{T}}PA. \tag{35}$$

For the MsHDP algorithm, the policy improvement (15) is written as

$$u^{(i)}(x) = K^{(i)}x, \tag{36}$$

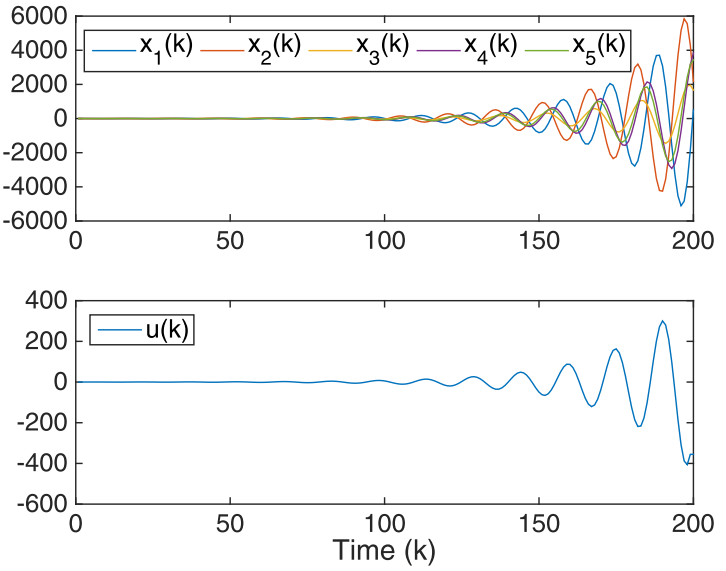

Fig. 8. For example 1, the trajectories of $x(k)$ and $u(k)$ under the control obtained by policy iteration.

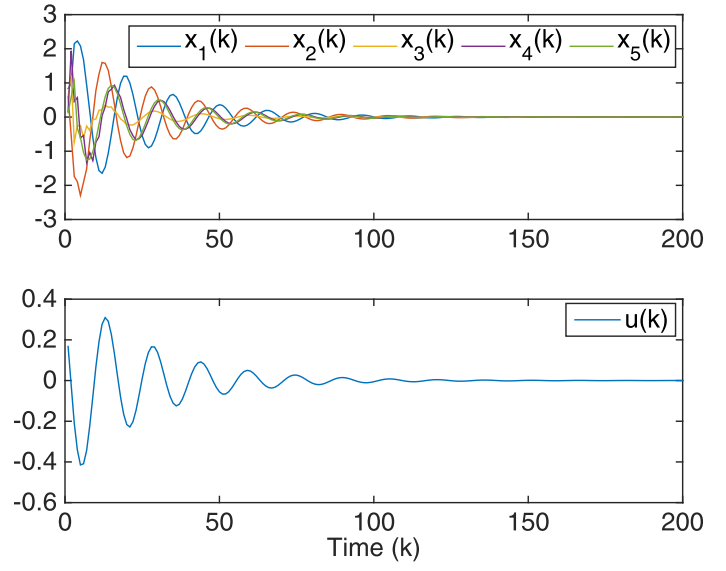

Fig. 9. For example 1, the trajectories of $x(k)$ and $u(k)$ under the control obtained by MsHDP with $\beta = 30$.

where $K^{(i)} = -(R + B^{\mathsf{T}}P^{(i)}B)^{-1}B^{\mathsf{T}}P^{(i)}A$. The policy evaluation (16) is given by:

$$x^{\mathsf{T}}(k)P^{(i+1)}x(k) = \sum_{l=k}^{k+\beta-1} \mathcal{R}(x(l), u^{(i)}(x(l))) + x^{\mathsf{T}}(k+\beta)P^{(i)}x(k+\beta). \tag{37}$$

The least-square scheme (30) is used to compute the unknown matrix $P^{(i+1)}$ of the Eq. (15), and the adaptive law (31) is used to compute $K^{(i)}$. For this LQR problem, the solution of the algebraic Riccati Eq. (35) is a positive symmetric matrix denoted by $P = \{p_{ij}\}_{n\times n} \in \mathbb{R}^{n\times n}$. Since $p_{ij} = p_{ji}$ for $i \neq j$, the matrix P contains $n(n+1)/2$ different parameters. Comparing (15) and (16) with (36) and (37), we have that $V^{(i)}(x) = x^{\mathsf{T}}P^{(i)}x = \sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij}^{(i)}x_i x_j$. Considering $p_{ij} = p_{ji}$, we can rewrite $V^{(i)}(x)$ with the form (27) as $V^{(i)}(x) = \Phi^{\mathsf{T}}(x)\theta_V^{(i)}$, where $\theta_V^{(i)} = [p_{1,1}^{(i)}, 2p_{1,2}^{(i)}, \cdots, 2p_{1,n}^{(i)}, p_{2,2}^{(i)}, \cdots, 2p_{2,n}^{(i)}, \cdots, p_{n,n}^{(i)}]^{\mathsf{T}}$ and $\Phi(x) = [x_1^2, x_1x_2, \cdots, x_1x_n, x_2^2, \cdots, x_2x_n, \cdots, x_n^2]^{\mathsf{T}}$. Similarly, the control policy (36) can also be rewritten with the form (31) as $u^{(i)}(x) = K^{(i)}x = \Psi^{\mathsf{T}}(x)\theta_u^{(i)}$, where $\theta_u^{(i)} = (K^{(i)})^{\mathsf{T}}$ and $\Psi(x) = x^{\mathsf{T}}$.

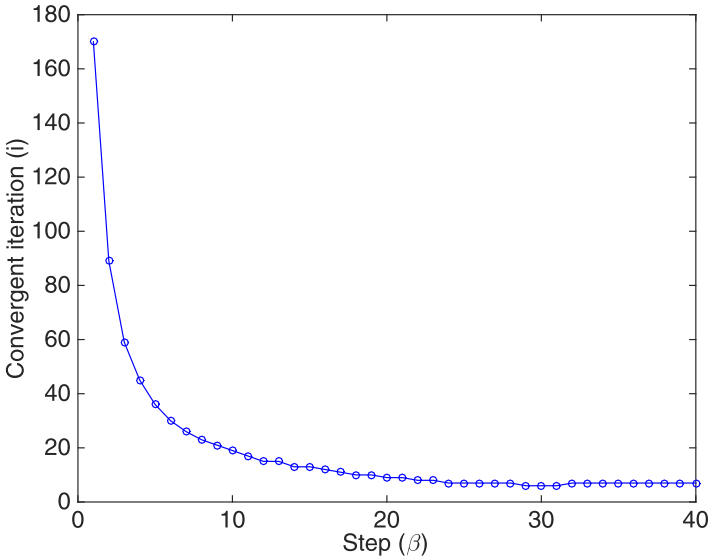

Fig. 10. For example 1, the relationship between the convergent iteration i and the step β by using MsHDP.

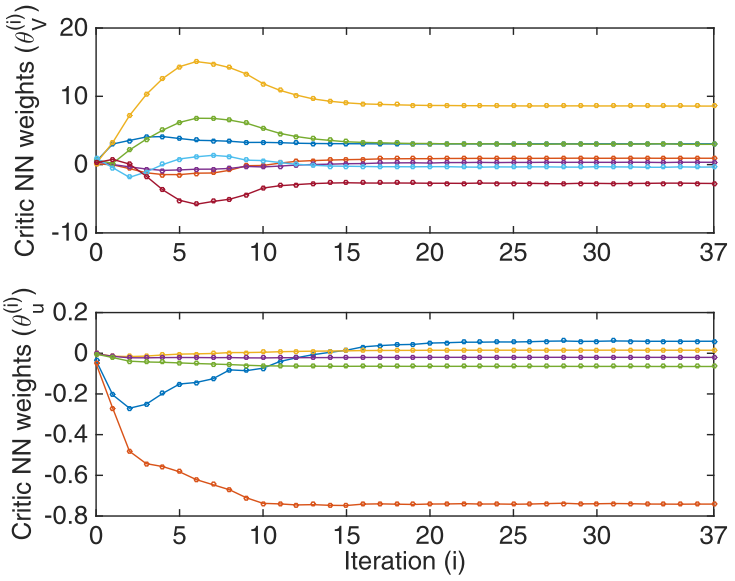

Fig. 11. For example 2, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of value iteration ($\beta = 1$).

5. Simulations studies

In this section, the effectiveness of the developed MsHDP method is verified through comparative simulation studies using two examples. For nonlinear systems, the explicit optimal control law is usually unavailable. To show that the MsHDP can find the optimal control policy, it is first tested on a linear system, and then it is applied to a nonlinear system.

5.1. Example 1: linear system

Consider the linear system (33) with the system matrices given by

$$A = \begin{bmatrix} 0.8336 & -0.1844 & 0.4933 & 0.2188 & 0.3701 \\ -0.6447 & 0.4335 & -0.1820 & -0.3976 & 0.1668 \\ 0.9231 & 1.2643 & -0.3826 & -0.1813 & -0.2393 \\ 1.0231 & 0.9135 & 0.7174 & -0.7530 & 0.9509 \\ 0.0386 & 0.2487 & 0.4597 & 0.1943 & 0.3258 \end{bmatrix},$$

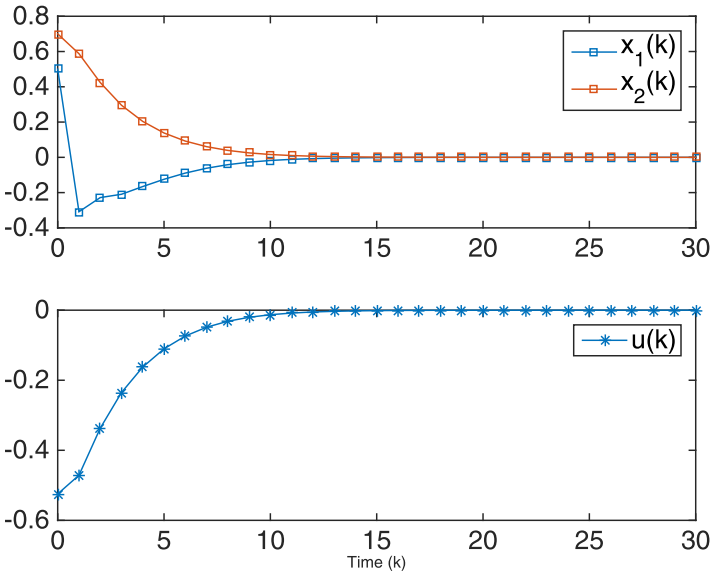

Fig. 12. For example 2, trajectories of the state $x(k)$ and control $u(k)$ of the closed-loop system under control obtained with value iteration.

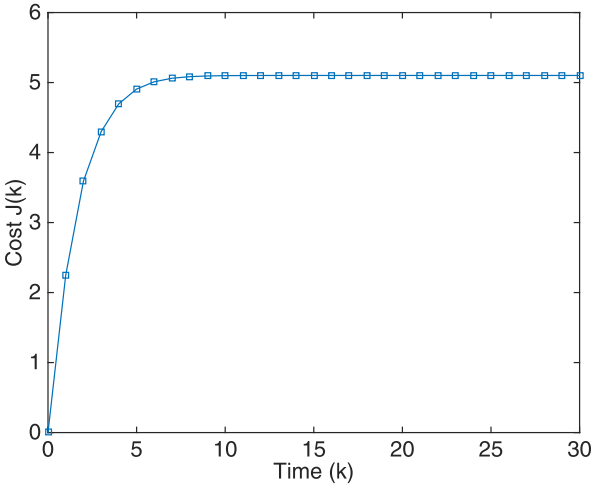

Fig. 13. For example 2, trajectory of the real cost $J(k)$ of the closed-loop system under control obtained with value iteration.

$$B = [0.0065, -0.5830, -0.0492, 0.6807, 0.2954]^{\mathrm{T}}.$$

For the performance index (2), let $Q = 0.1I$ and $R = 20$. For this LQR problem, its algebraic Riccati equation can be solved with the MATLAB command DARE. The optimal value function is $J^*(x) = x^{\mathrm{T}}Px$ with P given by

$$P = \begin{bmatrix} 17.1107 & 10.0862 & 7.6090 & 0.2768 & 7.9625 \\ 10.0862 & 11.4994 & 5.2113 & -0.8873 & 6.9936 \\ 7.6090 & 5.2113 & 3.7341 & -0.0517 & 4.0247 \\ 0.2768 & -0.8873 & -0.0517 & 0.5199 & -0.4168 \\ 7.9625 & 6.9936 & 4.0247 & -0.4168 & 5.0128 \end{bmatrix} \tag{38}$$

and the optimal control is $u^*(x) = Kx$ with K given by

$$K = [0.0311, 0.1885, 0.0379, -0.0255, 0.0887]^{\mathrm{T}}. \tag{39}$$

To solve this problem with the developed MsHDP method, value iteration and policy iteration, the critic NN activation functions are selected as $\Phi(x) = [x_1^2,\ x_1x_2,\ x_1x_3,\ x_1x_4,\ x_1x_5,\ x_2^2,\ x_2x_3,\ x_2x_4,\ x_2x_5,\ x_3^2,\ x_3x_4,\ x_3x_5,\ x_4^2,\ x_4x_5,\ x_5^2]^{\mathrm{T}}$, and the actor NN activation functions are selected as $\Psi(x) = x = [x_1,\ x_2,\ x_3,\ x_4,\ x_5]^{\mathrm{T}}$. From (38) and (39), the ideal critic NN weight vector $\theta_V^* = K = [p_{11},\ 2p_{12},\ 2p_{13},\ 2p_{14},\ 2p_{15},\ p_{22},\ 2p_{23},\ 2p_{24},\ 2p_{25},\ p_{33},\ 2p_{34},\ 2p_{35},\ p_{44},\ 2p_{45},\ p_{55}]^{\mathrm{T}} = [17.1107,\ 20.1725,\ 15.2181,$

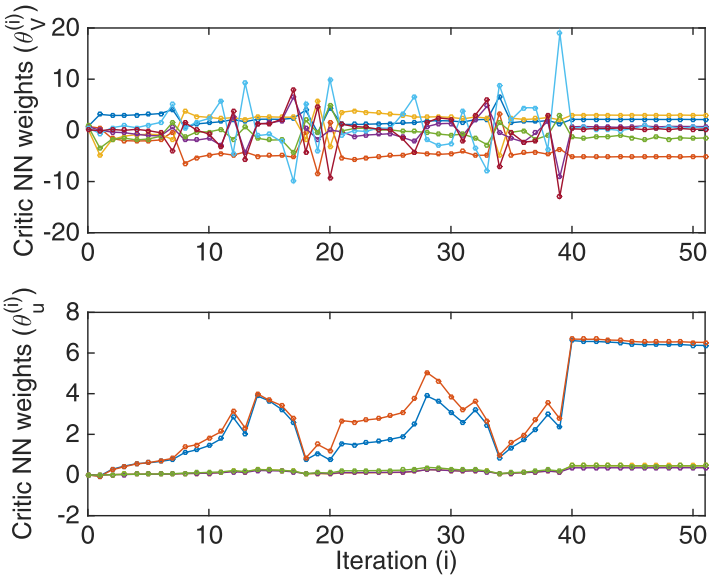

Fig. 14. For example 2, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of policy iteration.

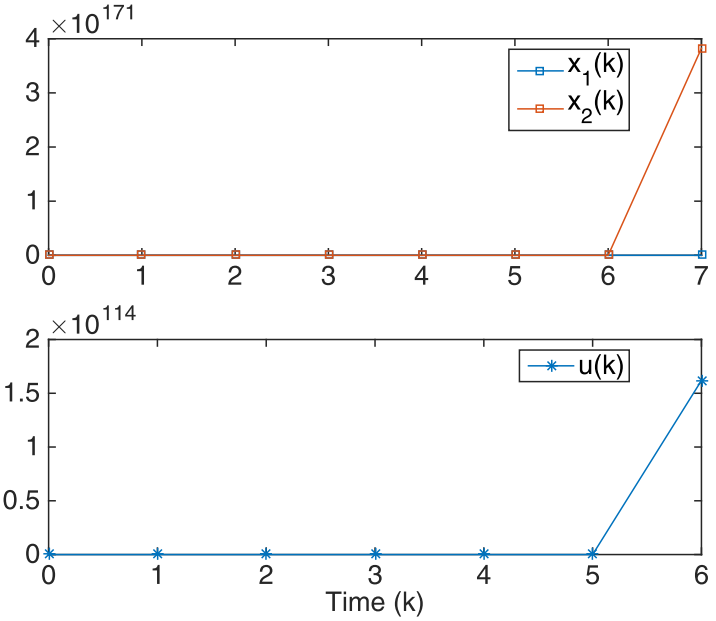

Fig. 15. For example 2, trajectories of the state $x(k)$ and control $u(k)$ of the closed-loop system under control obtained with policy iteration.

0.5535, 15.9250, 11.4994, 10.4227, −1.7747, 13.9872, 3.7341, −0.1034, 8.0495, 0.5199, −0.8336,5.0128]$^{\mathrm{T}}$, and the ideal actor NN weight vector is $\theta_u^* = [0.0311, 0.1885, 0.0379, -0.0255, 0.0887]^{\mathrm{T}}$.

Value iteration (i.e., $\beta = 1$) and policy iteration are employed for comparative simulation studies with MsHDP with 4 different number of steps: $\beta = 2, 5, 10, 30$. Figs. 2–7 show the comparative results, where the dotted lines represent ideal values of the critic NN weight vector θ_V^* and the actor NN weight vector θ_u^*. It is observed from simulation results that the critic NN weight vector $\theta_V^{(i)}$ and the actor NN weight vector $\theta_u^{(i)}$ obtained from value iteration and MsHDP converge to the ideal values (dotted lines) of θ_V^* and θ_u^*. From Fig. 2, it is shown that value iteration achieves convergence at $i = 170$th iteration. Figs. 3–6 show that the convergence was achieved at $i = 89$th, $i = 36$th, $i = 19$th and $i = 6$th iteration when using MsHDP with $\beta = 2$, $\beta = 5$, $\beta = 10$ and $\beta = 30$, respectively. Obviously, MsHDP converges much faster than value iteration. The results obtained with policy iteration is given in Fig. 7, which shows that $\theta_V^{(i)}$ and $\theta_u^{(i)}$ do not converge to their ideal values (dotted lines). This means that policy iteration cannot be used for optimal control design if initial stabilizing control is not available. By using the converged control obtained with policy iteration, the state and control of the closed-loop system

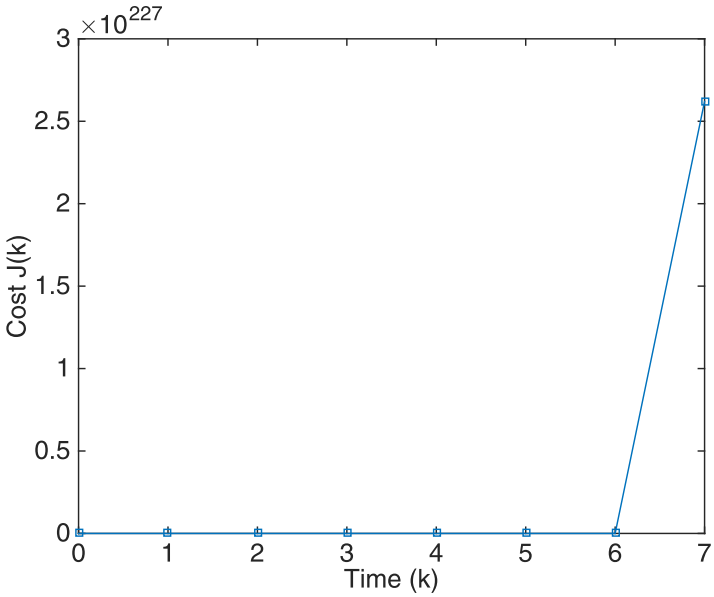

Fig. 16. For example 2, trajectory of the real cost $J(k)$ of the closed-loop system under control obtained with policy iteration.

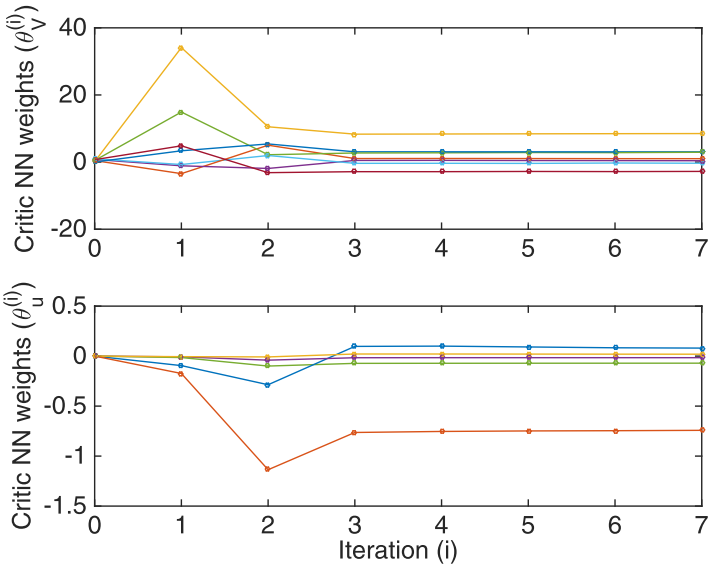

Fig. 17. For example 2, the critic NN weights $\theta_V^{(i)}$ and actor NN weights $\theta_u^{(i)}$ of MsHDP with $\beta = 5$.

is given in Fig. 8, which shows the closed-loop system becomes unstable. For comparison, the converged control obtained from MsHDP ($\beta = 30$) was also used for simulation, and the result was shown in Fig. 9.

To further investigate how the number of steps β affects the performance of MsHDP, we conduct simulation of MsHDP with β changing from 1 to 40. Fig. 10 demonstrate the simulation result, from which the converged step of MsHDP decreases sharply with β. This means that the increase of the step β can greatly improve the performance of MsHDP compared with value iteration (i.e., $\beta = 1$).

Through the simulation on the linear system, the effectiveness of the developed MsHDP is verified. From the simulation results, MsHDP overcomes the problem of policy iteration that requires the initial stabilizing control policy. Moreover, it is observed that MsHDP greatly improves the performance compared with value iteration, and better performance can be achieved by increasing the step β. Next, the developed MsHDP is employed to solve the optimal control problem of a nonlinear system.

5.2. Example 2: nonlinear system

Consider the following nonlinear system

$$\begin{cases} x_1(k+1) = & -0.15x_1(k) + 0.705x_2^2(k) - 0.015x_1^3(k) + 1.10u(k), \\ x_2(k+1) = & 1.33x_2(k) + 0.27x_1^2(k) + 0.133x_2^3(k) + 0.87u(k), \end{cases} \tag{40}$$

with $x(0) = [0.5, 0.7]^{\mathrm{T}}$. For the performance index (2), let $Q = 3I$ and $R = 0.1$. To show the real cost generated by using control u, define the real cost with respect to time as

$$J(k) \triangleq \sum_{l=0}^{k} \mathcal{R}(x(l), u(l)). \tag{41}$$

Select the critic NN activation functions as $\Phi(x) = [x_1^2,\ x_1x_2,\ x_2^2,\ x_1^3,\ x_1^2x_2,\ x_1x_2^2,\ x_2^3]^{\mathrm{T}}$, and the actor NN activation functions as $\Psi(x) = [x_1,\ x_2,\ x_1^2,\ x_1x_2,\ x_2^2]^{\mathrm{T}}$. Value iteration, policy iteration and MsHDP ($\beta = 5$) are employed to solve this optimal control problem, respectively.

By using value iteration, the critic NN weight vector $\theta_V^{(i)}$ and the actor NN weight vector $\theta_u^{(i)}$ are given in Fig. 11, where value iteration achieves convergence at $i = 37$th iteration. With the converged control policy, simulation is conducted on the closed-loop system, and the state and control are shown in Fig. 12. The real cost $J(k)$ is given in Fig. 13, where it converges to 5.1015.

By using policy iteration, the critic NN weight vector $\theta_V^{(i)}$ and the actor NN weight vector $\theta_u^{(i)}$ are given in Fig. 14, where it achieves convergence at $i = 51$th iteration. With the converged control policy, simulation is conducted on the closed-loop system. The system is interrupted at time $k = 7$ because the closed-loop system becomes unstable. Figs. 15 and 16 show the state and control and the real cost $J(k)$, respectively. From these figures, it is observed that policy iteration fails to solve this problem.

Next, the developed MsHDP ($\beta = 5$) is employed to solve this problem. Fig. 17 shows the critic NN weight vector $\theta_V^{(i)}$ and the actor NN weight vector $\theta_u^{(i)}$, where MsHDP achieves convergence at $i = 7$th iteration. By using the converged control policy, simulation is conducted on the closed-loop system. The trajectories of the state and control and the real cost are similar to that in Figs. 12 and 13, which are omitted here. The real cost (41) converges to $J(k) = 5.0847$ as k increases.

Remark 3. Through simulations with the MsHDP, we found that there exists a maximum for the step size β, denoted by $\overline{\beta}$. That is to say, if $1 \leqslant \beta \leqslant \overline{\beta}$, the MsHDP converges to the solution of the Bellman equation, and then solves the optimal control problem. Otherwise, it may not. For specifical practical problems, the maximum $\overline{\beta}$ is different for different systems. Therefore, it is difficult and also not necessary to determine the maximum $\overline{\beta}$ exactly. Experiences will be helpful to suggest a appropriate step size β satisfying $1 \leqslant \beta \leqslant \overline{\beta}$. □

6. Conclusions

The MsHDP method has been developed to solve the optimal control problem of nonlinear discrete-time systems in this paper. After detailed review and analysis of the policy iteration and value iteration algorithms, it is found that the advantages and disadvantages of the two algorithms result from their differences in policy evaluation. To overcome the drawbacks of both algorithms, the MsHDP algorithm has been developed by using the multi-step policy evaluation scheme, which starts from an arbitrary initial positive semi-definite value function and thus avoids the requirement of initial stabilizing control policy. The actor-critic structure has been developed to implement the MsHDP algorithm, where actor and critic NNs are employed to approximate the control policy and value function, respectively. To help readers understand the developed MsHDP method, it has been employed to solve a special case: the LQR problem. Through the comparative simulation studies with value iteration, policy iteration and MsHDP, the results demonstrate that MsHDP converges much faster than value iteration and also avoids the initial stabilizing control policy requirement in policy iteration.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 61533017, U1501251, 61374105, 61503377, 61233001, the Early Career Development Award of SKLMCCS, and in part by the NPRP grant #NPRP 9 166-1-031 from the Qatar National Research Fund (a member of Qatar Foundation). The authors would like to thank anonymous reviewers for their valuable comments.

References

[1] M. Abu-Khalaf, F.L. Lewis, Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach, Automatica 41 (5) (2005) 779–791.
[2] A. Al-Tamimi, F.L. Lewis, M. Abu-Khalaf, Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof, IEEE Trans. Syst. Man Cybern. Part B 38 (4) (2008) 943–949.
[3] R.W. Beard, G.N. Saridis, J.T. Wen, Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation, Automatica 33 (12) (1997) 2159–2177.
[4] D.P. Bertsekas, Dynamic programming and optimal control, 1, Nashua: Athena Scientific, 2005.

[5] D.P. Bertsekas, Lambda-policy iteration: a review and a new implementation, in: F.L. Lewis, D. Liu (Eds.), Reinforcement Learning and Approximate Dynamic Programming for Feedback Control, John Wiley & Sons, 2012.
[6] Z. Chen, S. Jagannathan, Generalized Hamilton–Jacobi–Bellman formulation-based neural network control of affine nonlinear discrete-time systems, IEEE Trans. Neural Netw. 19 (1) (2008) 90–106.
[7] T. Dierks, S. Jagannathan, Online optimal control of affine nonlinear discrete-time systems with unknown internal dynamics by using time-based policy update, IEEE Trans. Neural Netw. Learn. Syst. 23 (7) (2012) 1118–1129.
[8] T. Feng, H. Zhang, Y. Luo, J. Zhang, Stability analysis of heuristic dynamic programming algorithm for nonlinear systems, Neurocomputing 149, Part C (2015) 1461–1468. http://dx.doi.org/10.1016/j.neucom.2014.08.046.
[9] Y. Fu, T. Chai, Online solution of two-player zero-sum games for continuous-time nonlinear systems with completely unknown dynamics, IEEE Trans. Neural Netw. Learn Syst. 27 (12) (2016) 2577–2587, doi:10.1109/TNNLS.2015.2496299.
[10] P. He, S. Jagannathan, Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints, IEEE Trans. Syst. Man Cybern. Part B 37 (2) (2007) 425–436.
[11] D.G. Hull, Optimal Control Theory for Applications, Troy, NY: Springer, 2003.
[12] B. Kiumarsi, F. Lewis, Actor-critic-based optimal tracking for partially unknown nonlinear discrete-time systems, IEEE Trans. Neural Netw. Learn. Syst. 26 (1) (2015) 140–151, doi:10.1109/TNNLS.2014.2358227.
[13] B. Kiumarsi, F. Lewis, M.-B. Naghibi-Sistani, A. Karimpour, Optimal tracking control of unknown discrete-time linear systems using input–output measured data, IEEE Trans. Cybern. 45 (12) (2015) 2770–2779, doi:10.1109/TCYB.2014.2384016.
[14] D.L. Kleinman, On an iterative technique for riccati equation computations, IEEE Trans. Autom. Control 13 (1) (1968) 114–115.
[15] J.Y. Lee, J.B. Park, Y.H. Choi, On integral generalized policy iteration for continuous-time linear quadratic regulations, Automatica 50 (2) (2014) 475–489. http://dx.doi.org/10.1016/j.automatica.2013.12.009.
[16] F.L. Lewis, D. Vrabie, Reinforcement learning and adaptive dynamic programming for feedback control, IEEE Circuits Syst. Mag. 9 (3) (2009) 32–50.
[17] F.L. Lewis, D. Vrabie, V.L. Syrmos, Optimal Control, Hoboken, NJ: John Wiley & Sons, 2013.
[18] H. Li, D. Liu, Optimal control for discrete-time affine non-linear systems using general value iteration, IET Control Theory Appl. 6 (18) (2012) 2725–2736.
[19] D. Liu, D. Wang, D. Zhao, Q. Wei, N. Jin, Neural-network-based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming, IEEE Trans. Autom. Sci. Eng. 9 (3) (2012) 628–634.
[20] D. Liu, Q. Wei, Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems, IEEE Trans. Neural Netw. Learn. Syst. 25 (3) (2014) 621–634.
[21] Y.J. Liu, J. Li, S. Tong, C.L.P. Chen, Neural network control-based adaptive learning design for nonlinear systems with full-state constraints, IEEE Trans. Neural Netw. Learn. Syst. 27 (7) (2016) 1562–1571.
[22] Y.-J. Liu, L. Tang, S. Tong, C. Chen, D.-J. Li, Reinforcement learning design-based adaptive tracking control with less learning parameters for nonlinear discrete-time MIMO systems, IEEE Trans. Neural Netw. Learn. Syst. 26 (1) (2015) 165–176, doi:10.1109/TNNLS.2014.2360724.
[23] B. Luo, T. Huang, H.-N. Wu, X. Yang, Data-driven H_∞ control for nonlinear distributed parameter systems, IEEE Trans. Neural Netw. Learn. Syst. 26 (11) (2015) 2949–2961, doi:10.1109/TNNLS.2015.2461023.
[24] B. Luo, D. Liu, T. Huang, D. Wang, Model-free optimal tracking control via critic-only q-learning, IEEE Trans. Neural Netw. Learn. Syst. 27 (10) (2016) 2134–2144.
[25] B. Luo, D.L.H.-N. Wu, D. Wang, F.L. Lewis, Policy gradient adaptive dynamic programming for data-based optimal control, IEEE Trans. Cybern. PP (99) (2017) 1–14, doi:10.1109/TCYB.2016.2623859.
[26] B. Luo, H.-N. Wu, T. Huang, Off-policy reinforcement learning for H_∞ control design, IEEE Trans. Cybern. 45 (1) (2015) 65–76.
[27] B. Luo, H.-N. Wu, T. Huang, D. Liu, Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design, Automatica 50 (12) (2014) 3281–3290.
[28] B. Luo, H.-N. Wu, T. Huang, D. Liu, Reinforcement learning solution for HJB equation arising in constrained optimal control problem, Neural Netw. 71 (2015) 150–158. http://dx.doi.org/10.1016/j.neunet.2015.08.007.
[29] B. Luo, H.-N. Wu, H.-X. Li, Data-based suboptimal neuro-control design with reinforcement learning for dissipative spatially distributed processes, Industrial & Engineering Chemistry Research 53 (29) (2014) 8106–8119.
[30] B. Luo, H.-N. Wu, H.-X. Li, Adaptive optimal control of highly dissipative nonlinear spatially distributed processes with neuro-dynamic programming, IEEE Trans. Neural Netw. Learn. Syst. 26 (4) (2015) 684–696.
[31] H. Modares, F.L. Lewis, M.-B. Naghibi-Sistani, Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks, IEEE Trans. Neural Netw. Learn. Syst. 24 (10) (2013) 1513–1525.
[32] M.L. Puterman, M.C. Shin, Modified policy iteration algorithms for discounted markov decision problems, Manage. Sci. 24 (11) (1978) 1127–1137.
[33] A. Sahoo, H. Xu, S. Jagannathan, Near optimal event-triggered control of nonlinear discrete-time systems using neurodynamic programming, IEEE Trans. Neural Netw. Learn. Syst. 27 (9) (2015) 1801–1815, doi:10.1109/TNNLS.2015.2453320.
[34] G.N. Saridis, C.-S. G. Lee, An approximation theory of optimal control for trainable manipulators, IEEE Trans. Syst. Man Cybern. 9 (3) (1979) 152–159.
[35] B. Scherrer, V. Gabillon, M. Ghavamzadeh, M. Geist, Approximate modified policy iteration, in: Proceedings of the 29 the International Conference on Machine Learning, Edinburgh, Scotland, UK, 2012, pp. 1–21.
[36] R. Song, F. Lewis, Q. Wei, H. Zhang, Z. Jiang, D. Levine, Multiple actor-critic structures for continuous-time optimal control using input-output data, IEEE Trans. Neural Netw. Learn. Syst. 26 (4) (2015) 851–865, doi:10.1109/TNNLS.2015.2399020.
[37] D. Vrabie, F.L. Lewis, Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems, Neural Netw. 22 (3) (2009) 237–246.
[38] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, F.L. Lewis, Adaptive optimal control for continuous-time linear systems based on policy iteration, Automatica 45 (2) (2009) 477–484.
[39] D. Wang, D. Liu, Q. Wei, D. Zhao, N. Jin, Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming, Automatica 48 (8) (2012) 1825–1832.
[40] F.-Y. Wang, N. Jin, D. Liu, Q. Wei, Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with ε-error bound, IEEE Trans. Neural Netw. 22 (1) (2011) 24–36.
[41] H. Wang, T. Huang, X. Liao, H. Abu-Rub, G. Chen, Reinforcement learning for constrained energy trading games with incomplete information, IEEE Trans. Cybern. PP (99) (2016) 1–13, doi:10.1109/TCYB.2016.2539300.
[42] H. Wang, T. Huang, X. Liao, H. Abu-Rub, G. Chen, Reinforcement learning in energy trading game among smart microgrids, IEEE Trans. Ind. Electron. 63 (8) (2016) 5109–5119, doi:10.1109/TIE.2016.2554079.
[43] Q. Wei, D. Liu, An iterative ϵ-optimal control scheme for a class of discrete-time nonlinear systems with unfixed initial state, Neural Netw. 32 (2012) 236–244.
[44] Q. Wei, D. Liu, H. Lin, Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems, IEEE Trans. Cybern. 46 (3) (2016) 840–853, doi:10.1109/TCYB.2015.2492242.
[45] Q. Wei, D. Liu, Q. Lin, Discrete-time local value iteration adaptive dynamic programming: admissibility and termination analysis, IEEE Trans. Neural Netw. Learn. Syst. PP (99) (2016) 1–13, doi:10.1109/TNNLS.2016.2593743.
[46] Q. Wei, D. Liu, Q. Lin, R. Song, Discrete-time optimal control via local policy iteration adaptive dynamic programming, IEEE Trans. Cybern. PP (99) (2016) 1–13, doi:10.1109/TCYB.2016.2586082.
[47] Q. Wei, D. Liu, X. Yang, Infinite horizon self-learning optimal control of nonaffine discrete-time nonlinear systems, IEEE Trans. Neural Netw. Learn. Syst. 26 (4) (2015) 866–879, doi:10.1109/TNNLS.2015.2401334.

[48] H.-N. Wu, B. Luo, Simultaneous policy update algorithms for learning the solution of linear continuous-time H_∞ state feedback control, Inf. Sci. 222 (2013) 472–485.
[49] B. Xu, C. Yang, Z. Shi, Reinforcement learning output feedback NN control using deterministic learning technique, IEEE Trans. Neural Netw. Learn. Syst. 25 (3) (2014) 635–641.
[50] H. Xu, Q. Zhao, S. Jagannathan, Finite-horizon near-optimal output feedback neural network control of quantized nonlinear discrete-time systems with input constraint, IEEE Trans. Neural Netw. Learn. Syst. 26 (8) (2015) 1776–1788.
[51] Q. Yang, S. Jagannathan, Reinforcement learning controller design for affine nonlinear discrete-time systems using online approximators, IEEE Trans. Syst. Man Cybern. Part B 42 (2) (2012) 377–390.
[52] H. Zhang, H. Liang, Z. Wang, T. Feng, Optimal output regulation for heterogeneous multiagent systems via adaptive dynamic programming, IEEE Trans. Neural Netw. Learn. Syst. 28 (1) (2017) 18–29, doi:10.1109/TNNLS.2015.2499757.
[53] H. Zhang, Y. Luo, D. Liu, Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints, IEEE Trans. Neural Netw. 20 (9) (2009) 1490–1503.
[54] H. Zhang, C. Qin, Y. Luo, Neural-network-based constrained optimal control scheme for discrete-time switched nonlinear system using dual heuristic programming, IEEE Trans. Autom. Sci. Eng. 11 (3) (2014) 839–849, doi:10.1109/TASE.2014.2303139.
[55] H. Zhang, R. Song, Q. Wei, T. Zhang, Optimal tracking control for a class of nonlinear discrete-time systems with time delays based on heuristic dynamic programming, IEEE Trans. Neural Netw. 22 (12) (2011) 1851–1862.
[56] H. Zhang, Q. Wei, Y. Luo, A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm, IEEE Trans. Syst. Man Cybern. Part B 38 (4) (2008) 937–942.
[57] D. Zhao, Z. Xia, D. Wang, Model-free optimal control for affine nonlinear systems with convergence analysis, IEEE Trans. Autom. Sci. Eng. 12 (4) (2015) 1461–1468, doi:10.1109/TASE.2014.2348991.
[58] Q. Zhao, H. Xu, S. Jagannathan, Neural network-based finite-horizon optimal control of uncertain affine nonlinear discrete-time systems, IEEE Trans. Neural Netw. Learn. Syst. 26 (3) (2015) 486–499, doi:10.1109/TNNLS.2014.2315646.
[59] X. Zhong, H. He, H. Zhang, Z. Wang, Optimal control for unknown discrete-time nonlinear markov jump systems using adaptive dynamic programming, IEEE Trans. Neural Netw. Learn. Syst. 25 (12) (2014) 2141–2155, doi:10.1109/TNNLS.2014.2305841.
[60] Y. Zhu, D. Zhao, D. Liu, Convergence analysis and application of fuzzy-HDP for nonlinear discrete-time HJB systems, Neurocomputing 149, Part A (2015) 124–131. http://dx.doi.org/10.1016/j.neucom.2013.11.055.