SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

June 2014, Vol. 57 068202:1–068202:10
doi: 10.1007/s11432-013-4954-y

A new self-learning optimal control laws for a class of discrete-time nonlinear systems based on ESN architecture

SONG RuiZhuo*, XIAO WenDong & SUN ChangYin

School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083*, China*

Received December 13, 2013; accepted February 13, 2014

Abstract A novel self-learning optimal control method for a class of discrete-time nonlinear systems is proposed based on iteration adaptive dynamic programming (ADP) algorithm. It is proven that the iteration costate functions converge to the optimal one, and a detailed convergence analysis of the iteration ADP algorithm is given. Furthermore, echo state network (ESN) architecture is used as the approximator of the costate function for each iteration. To ensure the reliability of the ESN approximator, the ESN mean square training error is constrained in the satisfactory range. Two simulation examples are given to demonstrate that the proposed control method has a fast response speed due to the special structure and the fast training process.

Keywords adaptive dynamic programming, discrete-time, optimal control, ESN, costate function

Citation Song R Z, Xiao W D, Sun C Y. A new self-learning optimal control laws for a class of discrete-time nonlinear systems based on ESN architecture. Sci China Inf Sci, 2014, 57: 068202(10), doi: 10.1007/s11432-013-4954-y

1 Introduction

Adaptive dynamic programming (ADP) is a very famous self-learning method, aiming to avoid the "curse of dimensionality" closely related to dynamic programming method [1,2]. In the last few years, ADP algorithms were developed in depth by Liu et al. [3–6], Powell [7], Jagannathan et al. [8,9], Lewis et al. [10–13], Murray et al. [14], Si et al. [15,16], and so on. Dual heuristic dynamic programming (DHP) is one of the most common algorithms in ADP, which derives from the gradient formalization of the Hamilton-Jacobi- Bellman (HJB) equation, meaning that the critic network in the ADP structure is used to approximate the gradient of the performance index function with respect to the system dynamic. The feedforward neural networks such as back propagation (BP) neural networks and radial basis function (RBF) neural networks are used as the approximator of ADP algorithm by most researchers [17–20]. But the BP neural network as a local search optimization method that usually converges to the local minimum points. As for RBF neural network, the center point is difficult to be determined.

Many neural networks are used in the control problem of nonlinear systems [21–28]. For neural networks, the two major categories are feedforward network (FNN) and recurrent network (RNN). For the

*Corresponding author (Email: ruizhuosong@163.com)

former, the implementation is static input-output mappings. It can approximate arbitrary nonlinear functions by arbitrary accuracy. Furthermore, the RNN has at least one cyclic path. The theoretical result shows that RNN can approximate arbitrary dynamical systems with arbitrary accuracy [29]. The key part of RNN performance lies in the activities of recurrent units and their variations with time [30], i.e. network state and network dynamics. So the RNN has universal approximation capability [31,32].

Echo state network (ESN) is a novel approach for RNN supervised training, which overcomes some obstacles in many other approaches for training RNNs, such as slow convergence, complex implementation of the learning algorithms, and the suboptimal solutions [33]. As we know, ESN is a constructive method for supervised training of RNN. The learning process is simple and fast. The basic idea of ESN is to use a large number of "reservoirs" as the supplier of some useful dynamics, and the desired output can be combined from the "reservoirs". Recursive least square method is used for the online training [34]. The ESN was proposed by Jaeger [35,36], and developed by many scholars. Prokhorov [37] discussed the ESN in a broader context of RNN applications, and highlighted challenges in practical applications. Rodan et al. [38] presented the minimal complexity of reservoir construction for obtaining competitive models. Xia et al. [39] introduced an augmented ESN, which was used as the nonlinear adaptive filter for the complex-valued signals. Koprinkova-Hristova et al. [34] investigated the possibility for adaptive critic online training using ESN.

Though the qualities of ESN have been studied by many researchers, and the ESN theory has made great progress, to our knowledge, how to design the optimal control laws using the ESN in the framework of ADP is still an open problem. There are the following three difficulties. 1) Difficulty in designing the optimal control laws and the iteration control algorithm, 2) difficulty in proving the convergence of the costate function, 3) difficulty in implementing the control scheme using ESN. In this paper, these difficulties will be overcome one by one. First, the optimal control laws based on DHP algorithm are established for the nonlinear systems. Then, the convergence analysis of iteration DHP algorithm is given. It is proven that the control law makes the system asymptotically stable. Furthermore, some theorems are given to demonstrate the boundedness and convergence of the iteration costate function. And then, the background and training method of ESN are given. The implementation scheme for DHP using ESN architecture is also proposed. At last, the simulation examples are provided to demonstrate the effectiveness of the proposed implementation scheme.

The rest of this paper is organized as follows. In Section 2, an overview of iteration DHP algorithm is provided, and the convergence analysis is given. In Section 3, the training method of ESN is presented in detail. The implementation process of DHP based on ESN architecture is proposed. In Section 4, two examples about linear and nonlinear systems are given to demonstrate the advantage of the ESN method. Finally, Section 5 concludes the paper.

2 The iteration DHP algorithm

2.1 A general framework for iteration DHP algorithm

Consider the following discrete-time nonlinear dynamical systems:

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

where $F(x(k), u(k)) = f(x(k)) + g(x(k))u(k)$, $F(0,0) = 0$. The state $x(k) \in \mathbb{R}^n$, $f(x(k)) \in \mathbb{R}^n$, $g(x(k)) \in \mathbb{R}^{n\times m}$ and the control $u(k) \in \mathbb{R}^m$. $F(x(k), u(k))$ is function smooth and Lipschitz continuous on a compact set $\Omega \in \mathbb{R}^n$. Here we assume that the system state is completely controllable and bounded on Ω. Define the following optimal control problem:

$$\inf_{\{u(k)\}_{k=0}^{\infty}} J(x(k), u(k)) = \inf_{\{u(k)\}_{k=0}^{\infty}} \left\{ \sum_{k=0}^{\infty} (x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k)) \right\}, \tag{2}$$

where Q and R are positive definite.

To solve the optimal control problem (2), the following assumption is necessary.

Assumption 1. For system (1), there exists constant matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times n}$, such that

$$A \leqslant \frac{\partial F(x(k),u(k))}{\partial x(k)} \leqslant B \tag{3}$$

holds, implying $\frac{\partial F(x(k),u(k))}{\partial x(k)} - A$ and $B - \frac{\partial F(x(k),u(k))}{\partial x(k)}$ are positive semi-definite.

Let $J^*(x(k)) = \inf_{\{u(k)\}_{k=0}^{\infty}} J(x(k),u(k))$ and $u^*(k)$ denote the optimal performance index function and the corresponding optimal control law, respectively. Based on Bellman's principle of optimality, we can have the following HJB equation:

$$J^*(x(k)) = \inf_{u(k)} \{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + J^*(x(k+1))\}, \tag{4}$$

and the optimal controller $u^*(k)$ satisfies

$$u^*(k) = \arg\inf_{u(k)} \{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + J^*(x(k+1))\}. \tag{5}$$

Define the costate function $\sigma(x(k+1)) = \frac{\mathrm{d}J(x(k+1))}{\mathrm{d}x(k+1)}$. Then the following relationship holds [40]:

$$\sigma(x(k)) = 2Qx(k) + \left(\frac{\partial F(x(k),u(k))}{\partial x(k)}\right)^{\mathrm{T}} \sigma(x(k+1)). \tag{6}$$

So we have

$$\sigma^*(x(k)) = 2Qx(k) + \left(\frac{\partial F(x(k),u(k))}{\partial x(k)}\right)^{\mathrm{T}} \sigma^*(x(k+1)), \tag{7}$$

and

$$u^*(k) = -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x(k))\sigma^*(x(k+1)), \tag{8}$$

where $\sigma^*(x(k+1)) = \frac{\mathrm{d}J^*(x(k+1))}{\mathrm{d}x(k+1)}$.

To get the optimal control law, the following iteration DHP algorithm is used:

$$\sigma^{[i+1]}(x(k)) = 2Qx(k) + \left(\frac{\partial F(x(k),u(k))}{\partial x(k)}\right)^{\mathrm{T}} \sigma^{[i]}(x(k+1)), \tag{9}$$

and

$$u^{[i]}(k) = -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x(k))\sigma^{[i]}(x(k+1)), \tag{10}$$

where $\sigma^{[i]}(x(k+1)) = \frac{\mathrm{d}J^{[i]}(x(k+1))}{\mathrm{d}x(k+1)}$, in which

$$J^{[i+1]}(x(k)) = \inf_{u(k)} \{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + J^{[i]}(x(k+1))\}, \tag{11}$$

with $J^{[0]}(x(k)) = 0$.

Below, a detailed convergence analysis of the proposed iteration DHP algorithm will be given.

2.2 Convergence analysis of the iteration DHP algorithm

Lemma 1 [41]**.** Define $\lim_{i\to\infty} J^{[i]}(x(k)) = J^*(x(k))$. Then $J^*(x(k))$, $\forall k$, satisfies HJB equation, i.e.,

$$J^*(x(k)) = \inf_{u(k)} \{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + J^*(x(k+1))\}. \tag{12}$$

It is clear that $J^{[i+1]}(x(k))$ is convergent [42], and the limitation satisfies HJB equation. Next, we will analyze the convergence of the itaration costate function.

Theorem 1. Define $J^{[i+1]}(x(k))$ as in (11) and $J^{[0]}(\cdot)=0$. Define $\sigma^{[i+1]}(x(k))$ as in (9). And $\sigma^*(x(k))$ is the optimal costate function as in (7). Then $\lim_{i\to\infty}\sigma^{[i]}(x(k))=\sigma^*(x(k))$ holds.

Proof. According to the definitions of the costate function and the vector function derivative, we can have

$$\begin{aligned}\sigma^{[i]}(x(k)) &= \frac{\mathrm{d}J^{[i]}(x(k))}{\mathrm{d}x(k)}\\ &= \lim_{\Delta x\to 0}\frac{J^{[i]}(x(k)+\Delta x)-J^{[i]}(x(k))}{\Delta x}.\end{aligned} \tag{13}$$

Letting $i\to\infty$, we obtain

$$\begin{aligned}\lim_{i\to\infty}\sigma^{[i]}(x(k)) &= \lim_{i\to\infty}\lim_{\Delta x\to 0}\frac{J^{[i]}(x(k)+\Delta x)-J^{[i]}(x(k))}{\Delta x}\\ &= \lim_{\Delta x\to 0}\lim_{i\to\infty}\frac{J^{[i]}(x(k)+\Delta x)-J^{[i]}(x(k))}{\Delta x}\\ &= \lim_{\Delta x\to 0}\frac{J^*(x(k)+\Delta x)-J^*(x(k))}{\Delta x}\\ &= \frac{\mathrm{d}J^*(x(k))}{\mathrm{d}x(k)}.\end{aligned} \tag{14}$$

As $\sigma^*(x(k))=\frac{\mathrm{d}J^*(x(k))}{\mathrm{d}x(k)}$, so (14) can be rewritten as

$$\lim_{i\to\infty}\sigma^{[i]}(x(k))=\sigma^*(x(k)). \tag{15}$$

The proof is completed.

From Theorem 1, we can conclude that the DHP algorithm is convergent, which means the iteration control law $u^{[i]}(k)$ also converges to $u^*(k)$. Here we give the following theorems to demonstrate the asymptotic stability of system (1) under $u^*(k)$.

Theorem 2. Let $u^*(k)$ be as in (5) and $J^*(x(k))$ be as in (4). Then the optimal control $u^*(k)$ stabilizes the system (1) asymptotically.

Proof. As Q and R are both positive definite matrices, $J^*(x(k))$ is considered as the candidate Lyapunov function.

With expression (4), we obtain

$$J^*(x(k+1))-J^*(x(k))=-\left\{x^{\mathrm{T}}(k)Qx(k)+(u^*(k))^{\mathrm{T}}Ru^*(k)\right\}\leqslant 0. \tag{16}$$

So the feedback system (1) is asymptotically stable, which means that the state $x(k)$ is convergent, i.e., as $k\to\infty$, $x(k)\to 0$. The proof is completed.

Base on Theorem 2, we will prove that the costate function $\sigma^{[i]}$ is bounded.

Theorem 3. Let $\sigma^{[i+1]}(x(k))$ be as in (9) for system (1). Then there exists a constant $Y>0$, such that the norm of $\sigma^{[i+1]}(x(k))$ is bounded by Y, $\forall x(k)$, i.e., $|\sigma^{[i+1]}(x(k))|<Y$, $\forall i$.

Proof. Since $J^{[0]}(x(k))=0$, we can easily know that $\sigma^{[0]}(x(k))=0$. So $|\sigma^{[0]}(x(k))|<Y$, for $i=0$.

Suppose that $|\sigma^{[j]}(x(k))|\leqslant Y_0<Y$, for $i=j$. Then for $i=j+1$, we have

$$\begin{aligned}|\sigma^{[j+1]}(x(k))| &= \left|2Qx(k)+\left(\frac{\partial F(x(k),u(k))}{\partial x(k)}\right)^{\mathrm{T}}\sigma^{[j]}(x(k+1))\right|\\ &\leqslant |2Qx(k)|+\left|\left(\frac{\partial F(x(k),u(k))}{\partial x(k)}\right)^{\mathrm{T}}\right|Y_0.\end{aligned} \tag{17}$$

As the state $x(k)$ is bounded, by Assumption 1, we have

$$|\sigma^{[j+1]}(x(k))|\ \leqslant |2Qx(k)|+\left|\left(\frac{\partial F(x(k),u(k))}{\partial x(k)}\right)^{\mathrm{T}}-B^{\mathrm{T}}+B^{\mathrm{T}}\right|Y_0$$

$$\begin{aligned}&\leqslant |2Qx(k)| + |B^{\mathrm{T}}|Y_0\\&< Y.\end{aligned} \tag{18}$$

This completes the proof. So $\sigma^{[i]}(x(k))$ is convergent in the region $(-Y, Y)$.

Theorem 4. For system (1), let $\sigma^{[i+1]}(x(k))$ be as in (9). Then for $k \to \infty$, we have $\lim_{k\to\infty} \sigma^{[i]}(x(k)) = 0$, $\forall i$.

Proof. For $i = 0$, we have $\sigma^{[0]}(x(k)) = 0$, $\forall x(k)$. That is, $\lim_{k\to\infty} \sigma^{[0]}(x(k)) = 0$.

Suppose that $\lim_{k\to\infty} \sigma^{[j]}(x(k)) = 0$, for $i = j$. From Theorem 2, we have $x(k) \to 0$ and $u(k) \to 0$, as $k \to \infty$. So we can get $x(k+1) \to 0$, as $k \to \infty$. Then for $i = j+1$, we have

$$\begin{aligned}\lim_{k\to\infty} \sigma^{[j+1]}(x(k)) &= \lim_{k\to\infty} \left(2Qx(k) + \left(\frac{\partial F(x(k), u(k))}{\partial x(k)} - B\right)^{\mathrm{T}} \sigma^{[j]}(x(k+1))\right) + B^{\mathrm{T}}\sigma^{[j]}(x(k+1))\\&\leqslant \lim_{k\to\infty}(2Qx(k)) + \lim_{k\to\infty}(B^{\mathrm{T}}\sigma^{[j]}(x(k+1)))\\&= 0.\end{aligned} \tag{19}$$

On the other hand, we have

$$\begin{aligned}\lim_{k\to\infty} \sigma^{[j+1]}(x(k)) &= \lim_{k\to\infty} \left(2Qx(k) + \left(\frac{\partial F(x(k), u(k))}{\partial x(k)} - A\right)^{\mathrm{T}} \sigma^{[j]}(x(k+1))\right) + A^{\mathrm{T}}\sigma^{[j]}(x(k+1))\\&\geqslant \lim_{k\to\infty}(2Qx(k)) + \lim_{k\to\infty}(A^{\mathrm{T}}\sigma^{[j]}(x(k+1)))\\&= 0.\end{aligned} \tag{20}$$

So we can get

$$0 \leqslant \lim_{k\to\infty} \sigma^{[j+1]}(x(k)) \leqslant 0, \tag{21}$$

which means $\lim_{k\to\infty} \sigma^{[j+1]}(x(k)) = 0$. Thus, we have $\lim_{k\to\infty} \sigma^{[i]}(x(k)) = 0$. This completes the proof.

Theorems 2–4 show that feedback system (1) is asymptotically stable under the control $u^*(k)$. When $k \to \infty$, the state $x(k) \to 0$ and the costate function $\sigma^{[i]}(x(k)) \to 0$. After a comprehensive analysis of the optimal control and the iteration costate function, the implementation method based on ESN will be given.

3 Implementation of iteration DHP algorithm using ESN architecture

In this section, we introduce the implementation process of getting the costate function $\sigma^{[i]}$ using ESN architecture. First, we give a brief introduction of ESN architecture.

3.1 Introduction of ESN architecture

In this paper, the ESN consists of K input units $h(k) = (h_1(k), h_2(k), \ldots, h_K(k))^{\mathrm{T}}$, N internal units $s(k) = (s_1(k), s_2(k), \ldots, s_N(k))^{\mathrm{T}}$ and L output units $y(k) = (y_1(k), y_2(k), \ldots, y_L(k))^{\mathrm{T}}$, where k is the time step.The basic network architecture used in this paper is as in Figure 1.

Here the connection weights for input units are denoted by $W_{\mathrm{in}} \in \mathbb{R}^{N\times K}$, the connection weights for internal units are denoted by $W \in \mathbb{R}^{N\times N}$, and the connection weights for output units are denoted by $W_{\mathrm{out}} \in \mathbb{R}^{L\times(K+N+L)}$. $W_{\mathrm{back}} \in \mathbb{R}^{N\times L}$ denotes the weights between the output units and the internal units and is the optionally project.

The activation of internal units is updated according to

$$s(k+1) = \phi(W_{\mathrm{in}}h(k+1) + Ws(k) + W_{\mathrm{back}}y(k)), \tag{22}$$

where ϕ is the reservoir activation function. The output is computed by

$$y(k+1) = \varphi(W_{\mathrm{out}}(h(k+1), s(k+1), y(k))), \tag{23}$$

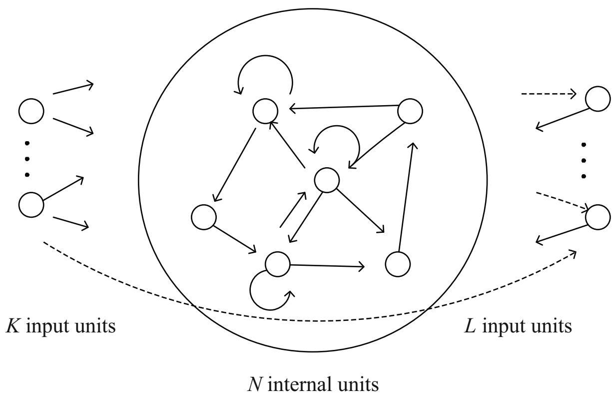

Figure 1 The basic ESN architecture.

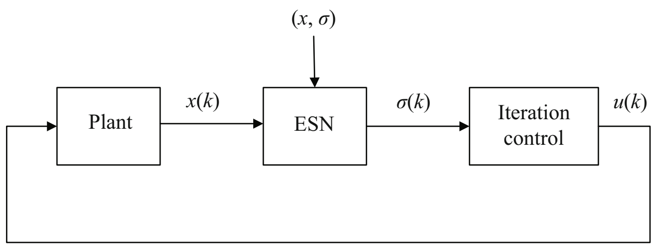

Figure 2 The process of DHP algorithm using ESN.

where φ is the output transfer function, $(h(k+1), s(k+1), y(k))$ is the concatenation of the input, internal, and previous output activation vectors. Especially for $k = 0$, the network output is $y_0 = 0$. In this paper, we let $y(k+1) = \varphi(W_{\text{out}} s(k+1))$ for convenience.

3.2 The training process of ESN

In this paper, the ESN training process is presented as follows.

1) Give the training input/output data length T and the sequence $(h(1), y(1))$, $(h(2), y(2))$, $\ldots$, $(h(T), y(T))$. Give randomly generated input weights W_{in} and output back propagation weights W_{back}. Give arbitrary network state s_0, K, N and L.

2) Let W_0 be a random internal weight matrix W_0, and let a be the spectral radius of W_0. Then we have $W_1 = 1/aW_0$, and $W = \alpha W_1$, where $0 < \alpha < 1$. So we get the internal units weight matrix W.

3) The network is driven by the training data from 0 to T.

4) Given the washout time K_0, collect the network states $(s(K_0), s(K_0+1), \ldots, s(T))^{\mathrm{T}}$ for $T/10 < K_0 < T$, as the new row into $M \in \mathbb{R}^{(T-K_0+1)\times N}$.

5) Collect $(\varphi^{-1}(y(K_0)), \varphi^{-1}(y(K_0+1)), \ldots, \varphi^{-1}(y(T)))^{\mathrm{T}}$ as the new row into $U \in \mathbb{R}^{(T-K_0+1)\times L}$.

6) Get the output weight matrix $W_{\text{out}} = (M^{-1}U)^{\mathrm{T}}$, which minimizes the the mean square training error (MSE) between the desired output and the actual output.

7) Set the output of the ESN at $y(k+1) = \varphi(W_{\text{out}} s(k+1))$.

This completes the ESN training.

3.3 The implementation process of DHP algorithm

In this paper, the ESN is used for getting the costate function $\sigma^{[i]}(x(k))$ for each iteration. The process of the algorithm implementation is as shown in Figure 2.

To implement the process of DHP algorithm, the system state is used as the input of ESN, and the costate function $\sigma^{[i]}(x(k))$ is considered as the ESN output. First, the T pairs input/output sequences are used to train the ESN and get the output weight matrix W_{out}. So for time step k, we can get $\sigma^*(x(k))$, and $u^*(k)$. Driving the system plant by $u^*(k)$, we obtain $x(k+1)$, and repeating the training process, we have $\sigma^*(x(k+1))$, $u^*(k+1)$ and $x(k+2)$. When $k \to \infty$, the state $x(k) \to 0$, the costate function

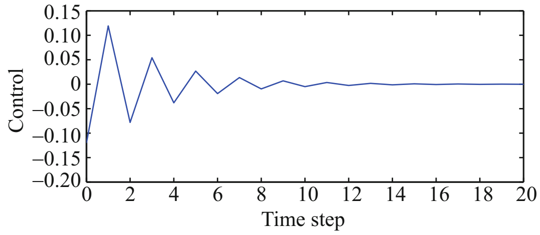

Figure 3 The control trajectory of system (24).

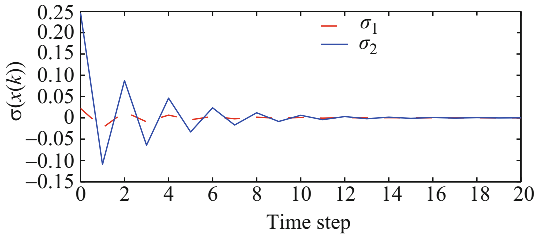

Figure 4 The trajectory of $\sigma(x(k))$.

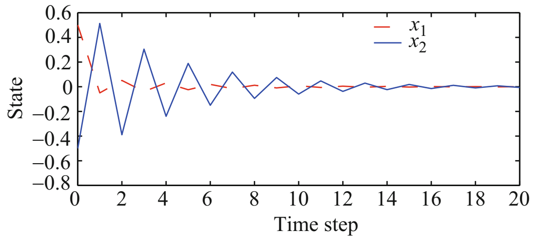

Figure 5 The state trajectories of system (24) by the proposed method with ESN.

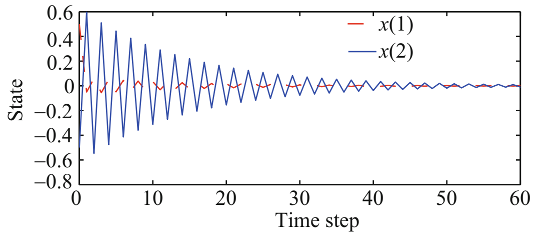

Figure 6 The state trajectories of system (24) by the method with BP.

$\sigma^{[i]}(x(k)) \to 0$ and the control law $u^{[i]}(k) \to 0$. The feedback system is asymptotically stable under the control law u^*.

4 Simulation study

In this section, two examples are used to demonstrate the detailed processes of ESN. The first example is a linear dynamical system. The second example is a nonlinear system.

4.1 Example 1

Consider the linear system as follows:

$$x(k+1) = Ax(k) + Bu(k), \tag{24}$$

where $A = \left[\begin{smallmatrix} 0 & 0.1 \\ 0.3 & -0.9 \end{smallmatrix}\right]$ and $B = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$.

Obviously, Assumption 1 holds for system (24). To train ESN, we select the internal units $N = 100$, input units $K = 2$ and output units $L = 2$. The weight matrixes W_0, W_{in} and W_{back} are chosen from $(-0.15, 0.15)$ randomly. According to Subsection 3.2, we can get W. We select $\phi = \tan h$ and $\varphi = \tan h$. The initial input of internal units s_0 is selected from $(-0.1, 0.1)$. Firstly, we train $T = 400$ pairs input/output data, and select the washout time K_0 as 201. Then we can obtain W_{out}. After 30 times iteration, the optimal control for time step k is reached. For initial system state $x(0) = [0.5; -0.5]$, the system runs 20 steps. To verify the approximation effect, the mean squared training error is calculated within 3.5e$-$31. So we get the control trajectory in Figure 3. Figure 4 shows that for any time step k, the costate function $\sigma^{[i]}(x(k))$ is bounded, and it converges to zero as $k \to \infty$. To verify the approximation effect, the system state trajectories obtained by the proposed method with ESN are shown in Figure 5, which converge within 20 time steps. To compare the control effect, we use BP neural network as the approximator. The initial weights of BP neural network are chosen randomly from $[-0.1, 0.1]$, and the learning rate is 0.02. Then for the same initial system state $x(0) = [0.5; -0.5]$, the system state trajectories are obtained by the method with BP neural network as in Figure 6, which converge within 60 time steps. The feedback control system in Figure 5 has a faster response speed than the one in Figure 6. Therefore, the results obtained by the algorithm in this paper are satisfactory. Obviously the ESN is able to implement the iteration DHP algorithm and has good effect.

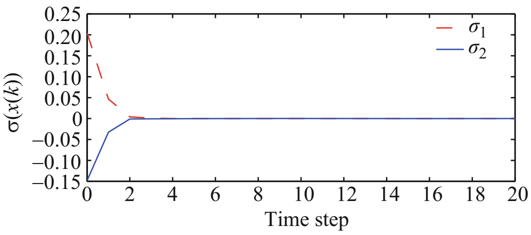

Figure 7 The trajectory of $\sigma(x(k))$.

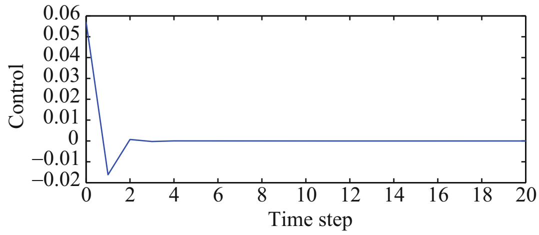

Figure 8 The control trajectory of system (25).

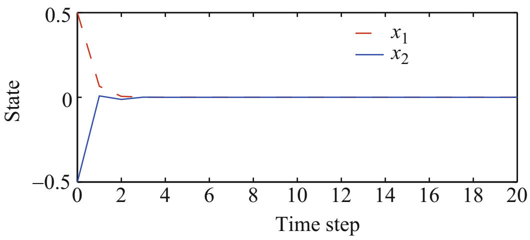

Figure 9 The state trajectories of system (25) by the proposed method with ESN.

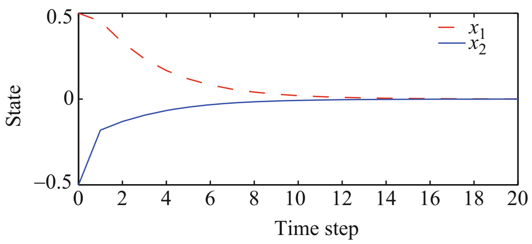

Figure 10 The state trajectories of system (25) by the method with BP.

4.2 Example 2

Consider the following discrete nonlinear system:

$$\begin{aligned} x(k+1) &= F(x(k), u(k)) \\ &= f(x(k)) + g(x(k))u(k), \end{aligned} \tag{25}$$

where $f(x(k)) = \begin{bmatrix} 0.7x_1(k)\exp{(x_2(k))^2} \\ 0.3(x_2(k))^3 - 0.3x_1(k) \end{bmatrix}$, $g(x(k)) = \begin{bmatrix} 0.5 \\ 0.8 \end{bmatrix}$.

For system (25), as the states are bounded, we can say Assumption 1 holds. In the ESN training process, the internal units is $N = 100$, input units is $K = 2$ and output units is $L = 2$. The weight matrixes W_0, W_{in}, W_{back} and the initial input of internal units s_0 are selected from $(-0.15, 0.15)$, respectively. We select $T = 400$ pairs input/output data, and washout time $K_0 = 201$ to train the ESN and get W_{out}. The internal units activation function and output units activation function are selected as $\phi = \tan h$ and $\varphi = \tan h$. To implement the DHP algorithm, the maximal iteration step is chosen as 30. The system plant runs 20 steps with the initial system state $x(0) = [0.5; -0.5]$. The mean squared training error is obtained within $1.4\mathrm{e}-31$. Then we get the trajectories of the costate function σ as given in Figure 7. They converge to zero as $k \to \infty$. The control trajectory is shown in Figure 8. The trajectories of the system state obtained by the proposed method with ESN are shown in Figure 9, which converge within 5 time steps. Similarly, to compare the control effect, we use BP neural network as the approximator for system (25). The initial weights of BP neural network are chosen randomly from $[-0.01, 0.01]$, and the learning rate is 0.01. Then for the same initial system state $x(0) = [0.5; -0.5]$, the system state trajectories are obtained by the method with BP neural network in Figure 10, which converge within 20 time steps. Obviously, the response speed in Figure 9 is faster than that in Figure 10. From the figures, we can see that the results obtained by the algorithm in this paper are more satisfactory than others.

5 Conclusion

A novel iteration DHP control scheme based on ESN architecture for a class of discrete-time nonlinear systems was proposed in this paper. First, the iteration DHP algorithm and the convergence analysis were given. Then, the fundamental training process of ESN, and the implementation process of DHP algorithm are proposed. At last, the simulation examples are given to validate the presented optimal control algorithm.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61304079, 61125306, 61034002), Beijing Natural Science Foundation (Grant No. 4143065), China Postdoctoral Science Foundation (Grant No. 2013M530527) and the Open Research Project from SKLMCCS (Grant No. 20120106).

References

1 Werbos P J. A Menu of Designs for Reinforcement Learning Over Time, in Neural Networks for Control. Massachusetts: MIT Press, 1991. 67–95
2 Werbos P J. Approximate Dynamic Programming for Real-Time Control and Neural Modeling, in Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches. New York: Van Nostrand Reinhold, 1992.
3 Liu D, Javaherian H, Kovalenko O, et al. Adaptive critic learning techniques for engine torque and air-fuel ratio control. IEEE Trans Syst Man Cybern B Cybern, 2008, 38: 988–993
4 Liu D, Xiong X, Zhang Y. Action-dependent adaptive critic designs. In: Proceedings of International Joint Conference on Neural Networks, Washington, 2001. 2: 990–995
5 Liu D, Zhang H. A neural dynamic programming approach for learning control of failure avoidance problems. Int J Intell Syst, 2005, 10: 21–32
6 Liu D, Zhang Y, Zhang H. A self-learning call admission control scheme for CDMA cellular networks. IEEE Trans Neural Netw, 2005, 16: 1219–1228
7 Powell W B. Approximate Dynamic Programming: Solving the Curses of Dimensionality. New York: Wiley, 2009
8 Zheng C, Jagannathan S. Generalized Hamilton-Jacobi-Bellman formulation-based neural network control of affine nonlinear discrete-time systems. IEEE Trans Neural Netw, 2008, 19: 90–106
9 He P, Jagannathan S. Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. IEEE Trans Syst Man Cybern B Cybern, 2007, 37: 425–436
10 Al-Tamimi A, Lewis F L, Abu-Khalaf M. Model-free Q-learning designs for linear discrete-time zero-sum games with application to H-infinity control. Automatica, 2007, 43: 473–481
11 Vrabie D, Pastravanu O, Abu-Khalaf M, et al. Adaptive optimal control for continuous-time linear systems based on policy iteration. Automatica, 2009, 45: 477–484
12 Vamvoudakis K G, Lewis F L. Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. Automatica, 2010, 46: 878–888
13 Abu-Khalaf M, Lewis F L. Nearly optimal control laws for nonlinear systems withsaturating actuators using a neural network HJB approach. Automatica, 2005, 41: 779–791
14 Murray J J, Cox C J, Lendaris G G, et al. Adaptive dynamic programming. IEEE Trans Syst Man Cybern C Appl Rev, 2002, 32: 140–153
15 Si J, Wang Y T. On-line learning control by association and reinforcement. IEEE Trans Neural Netw, 2001, 12: 264–276
16 Enns R, Si J. Helicopter trimming and tracking control using direct neural dynamic programming. IEEE Trans Neural Netw, 2003, 14: 929–939
17 Zhang H G, Wei Q L, Luo Y H. 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 B Cybern, 2008, 38: 937–942
18 Zhang H G, Luo Y H, Liu D R. Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints. IEEE Trans Neural Netw, 2009, 20: 1490–1503
19 Wang F Y, Jin N, Liu D R, et al. Adaptive dynamic programming for finite horizon optimal control of discrete-time nonlinear systems with ε-error bound. IEEE Trans Neural Netw, 2011, 22: 24–36
20 Zhang H G, Wei Q L, Liu D R. An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. Automatica, 2011, 47: 207–214
21 Chang W D, Hwang R C, Hsieh J G. Stable direct adaptive neural controller of nonlinear systems based on single auto-tuning neuron. Neurocomputing, 2002, 48: 541–554
22 Du H B, Chen X C. NN-based output feedback adaptive variable structure control for a class of non-affine nonlinear systems: A nonseparation principle design. Neurocomputing, 2009, 72: 2009–2016
23 Song R Z, Zhang H G, Luo Y H, et al. Optimal control laws for time-delay systems with saturating actuators based on heuristic dynamic programming. Neurocomputing, 2010, 73: 3020–3027
24 Wei Q L, Zhang H G, Dai J. Model-free multiobjective approximate dynamic programming for discrete-time nonlinear systems with general performance index functions. Neurocomputing, 2009, 72: 1839–1848
25 Li X, Xian B, Diao C, et al. Output feedback control of hypersonic vehicles based on neural network and high gain observer. Sci China Inf Sci, 2011, 54: 429–447

26 Xu B, Gao D, Wang S. Adaptive neural control based on HGO for hypersonic flight vehicles. Sci China Inf Sci, 2011, 54: 511–520
27 Wang M, Zhang S, Chen B, et al. Direct adaptive neural control for stabilization of nonlinear time-delay systems. Sci China Inf Sci, 2010, 53: 800–812
28 Huang Z, Wang X, Sannay M. Self-excitation of neurons leads to multiperiodicity of discrete-time neural networks with distributed delays. Sci China Inf Sci, 2011, 54: 305–317
29 Jaeger H. A Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the Echo State Network Approach. Bremen: International University Bremen, 2002
30 Čerňanský M. Feed-forward Echo State Networks. In: Proceedings of International Joint Conference on Neural Networks, Montreal, 2005. 1479–1482
31 Liu Z, Zhang H, Zhang Q. Novel stability analysis for recurrent neural networks with multiple delays via line integral-type L-K functional. IEEE Trans Neural Netw, 2010, 21: 1710–1718
32 Zhang H, Liu Z, Huang G, et al. Novel weighting-delay-based stability criteria for recurrent neural networks with time-varying delay. IEEE Trans Neural Netw, 2010, 21: 91–106
33 Lukoševičius M, Popovici D, Jaeger H, et al. Time warping invariant echo state networks, 2006. Available form: http://jpubs.jacobs-university.de/bitstream/579/149/1/twiesn_iubtechreport.pdf
34 Koprinkova-Hristova P, Oubbati M, Palm G. Adaptive critic design with echo state network. In: Proceedings of the IEEE International Conference on Systems Man and Cybernetics, Istanbul, 2010. 1010–1015
35 Jaeger H. The Echo State Approach to Analysing and Training Recurrent Neural Networks. GMD Report 148, GMD-German National Research Institute for Computer Science. 2001
36 Jaeger H. Short Term Memory in Echo State Networks. GMD Report 152, GMD-German National Research Institute for Computer Science. 2002
37 Prokhorov D. Echo state networks: appeal and challenges. In: Proceedings of the International Joint Conference on Neural Networks, Montreal, 2005. 1463–1466
38 Rodan A, Tiňo P. Minimum complexity echo state network. IEEE Trans Neural Netw, 2011, 22: 131–144
39 Xia Y L, Jelfs B, van Hulle Marc M, et al. An augmented echo state network for nonlinear adaptive filtering of complex noncircular signals, IEEE Trans Neural Netw, 2011, 22: 74–83
40 Lin W S. Optimality and convergence of adaptive optimal control by reinforcement synthesis. Automatica, 2008, 44: 2716–2723
41 Zhang H G, Song R Z, Wei Q L, et al. Optimal tracking control for a class of nonlinear discrete-time systems with time delays based on heuristic dynamic programming. IEEE Trans Neural Netw, 2011, 22: 1851–1862
42 Al-Tamimi A, Lewis F L. Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. IEEE Trans Syst Man Cybern B Cybern, 2007, 38: 943–949