Soft Comput (2014) 18:1645–1653
DOI 10.1007/s00500-013-1170-z

METHODOLOGIES AND APPLICATION

# Neural-network-based approach to finite-time optimal control for a class of unknown nonlinear systems

**Ruizhuo Song · Wendong Xiao · Qinglai Wei · Changyin Sun**

Published online: 10 November 2013


**Abstract** This paper proposes a novel finite-time optimal control method based on input–output data for unknown nonlinear systems using adaptive dynamic programming (ADP) algorithm. In this method, the single-hidden layer feed-forward network (SLFN) with extreme learning machine (ELM) is used to construct the data-based identifier of the unknown system dynamics. Based on the data-based identifier, the finite-time optimal control method is established by ADP algorithm. Two other SLFNs with ELM are used in ADP method to facilitate the implementation of the iterative algorithm, which aim to approximate the performance index function and the optimal control law at each iteration, respectively. A simulation example is provided to demonstrate the effectiveness of the proposed control scheme.

**Keywords** Adaptive dynamic programming · Approximate dynamic programming · Unknown nonlinear systems · Optimal control · Data-based

## 1 Introduction

The linear optimal control problem with a quadratic cost function is probably the most well-known control problem (Duncan et al. 1999; Gabasov et al. 2000), and it can be translated into Riccati equation. While the optimal control of nonlinear systems is usually a challenging and difficult problem (Jin et al. 2012; Zhang et al. 2011e). Furthermore, comparing with the known system dynamics case, it is more intractable to solve the optimal control problem of the unknown system dynamics. Generally speaking, most actual systems are nearly far too complex to present the perfect mathematical models. Whenever no model is available to design the system controller nor is easy to produce, a standard way is resorting to data-based techniques (Guardabassi and Savaresi 2000): (1) on the basis of input-output data, the model of the unknown system dynamics is identified; (2) on the basis of the estimated model of the system dynamics, the controller is designed by model-based design techniques.

It is well known that neural network is an effective tool to implement intelligent identification based on input–output data, due to the properties of nonlinearity, adaptivity, self-learning and fault tolerance (Jagannathan 2006; Yu 2009; Fernández-Navarro et al. 2013; Richert et al. 2013; Maji et al. 2013). In which, single-hidden-layer feed-forward neural network (SLFN) is one of the most useful types (Huang et al. 2006b). Hornik (1991) proved that if the activation function is continuous, bounded, and non-constant, then continuous mappings can be approximated by SLFNs with additive hidden nodes over compact input sets. Leshno et al. (1993) improved the results of Hornik (1991) and proved that SLFNs with additive hidden nodes and with a non-polynomial activation function can approximate any continuous target functions. In Huang et al. (2006b) it is proven in theory that SLFNs with randomly generated additive and a broad type of activation functions can universally approximate any continuous target functions in any compact subset of the Euclidean space. For SLFN training, there are three main approaches: (1) gradient-descent based, for example back-propagation (BP) method (Zhang et al. 2008); (2) least square based, for

Communicated by D. Liu.

R. Song · W. Xiao · C. Sun
School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China

Q. Wei (✉)
The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
e-mail: rzsong@126.com

example extreme learning machine (ELM) method in this paper; (3) standard optimization method based, for example support vector machine (SVM). While, the learning speed of feed-forward neural networks is in general far slower than required and it has been a major bottleneck in their applications for past decades (Huang and Siew 2004). Two key reasons are: (1) the slow gradient-based learning algorithms are extensively used to train neural networks, (2) all the parameters of the networks are tuned iteratively by using such learning algorithms. Unlike conventional neural network theories, in this paper, the ELM method is used to train SLFN. Such SLFN can be as the universal approximator, one may simply randomly choose hidden nodes and then only need to adjust the output weights linking the hidden layer and the output layer. For given network architectures, ELM does not require human-intervened parameters, so ELM has fast convergence and can be easily used.

Based on the SLFN identifier, the finite-time optimal control method is presented in this paper. It is worth noting that although dynamic programming is very useful in solving the optimal control problems (Murray and Cox 2002; Seiffertt et al. 2008; He and Jagannathan 2007; Zhao et al. 2008), it is often computationally untenable to run dynamic programming (Bellman 1957). Thus, adaptive/approximate dynamic programming (ADP) algorithm was proposed in (Werbos 1992, 2008) as an effective intelligent control method and has played an important role in seeking solutions for the optimal control problem (Zhang et al. 2011a,b,c,d; Liu and Zhang 2005). Furthermore, most works about ADP adopted feed-forward neural networks with gradient descent-based methods to approximate the critic network and action network (Wei and Liu 2012; Liu and Wei 2013; Song et al. 2010; Song and Zhang 2013). Moreover, most research results about ADP focus on infinite-time optimal control problems. But the system cannot really be stabilized until the time reaches infinity. While for finite-time control problems, the system must be stabilized to zero within finite time. The controller design of finite-time problems still presents a challenge to control engineers as the lack of methodology and the control step is difficult to determine. Few results relate to the finite-time optimal control based on ADP algorithm. As we know that Wang et al. (2011) solved the finite-horizon optimal control problem for a class of discrete-time nonlinear systems using ADP algorithm. But the method in Wang et al. (2011) adopts the BP networks to obtain the optimal control, which has slow convergence speed.

In this paper, we will design the finite-time optimal controller based on SLFN with ELM for unknown nonlinear systems. First, the identifier is established by the input–output data. Upon the data-based identifier, the optimal control method is proposed. We prove that the iterative performance index function converges to optimum, and the optimal control is also obtained. Compared to other popular implementation methods such as BP, the SLFN with ELM has the fast response speed and is fully automatic. It means that except for target errors and the allowed maximum number of hidden nodes, no control parameters need to be manually tuned by users.

The rest of this paper is organized as follows. In Sect. 2, the problem formulation is presented. In Sect. 3, the identifier is developed based on the input–output data. In Sect. 4, the iterative ADP algorithm and the convergence proof are given. In Sect. 5, an example is given to demonstrate the effectiveness of the proposed control scheme. In Sect. 6, the conclusion is drawn.

## 2 Motivations and problem formulation

Consider the following unknown discrete-time nonlinear systems

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

where the state $x(k) \in \Re^n$ and the control $u(k) \in \Re^m$. $F(x(k), u(k))$ is unknown continuous function. Assume that the state is completely controllable and bounded on $\Omega$, and $F(0, 0) = 0$. The finite-time performance index function is defined as follows:

$$J(x(k), U(k, K)) = \sum_{i=k}^{K} \{x^T(i)Qx(i) + u^T(i)Ru(i)\}, \tag{2}$$

where $Q$ and $R$ are positive definite matrices, $K$ is the finite positive integer, the control sequence $U(k, K) = (u(k), u(k+1), \ldots, u(K))$ is finite-time admissible (Wang et al. 2011). The length of $U(k, K)$ is defined as $(K - k + 1)$.

This paper is desired to find the optimal control for system (1) based on performance index function (2). Since the system dynamics is completely unknown, the optimal problem cannot be solved directly. Therefore, it is desirable to propose a novel method that does not need the exact system dynamics but only the input-output data, which can be obtained during the operation of the system. In this paper, we propose a data-based optimal control scheme using SLFN with ELM and ADP method for general unknown nonlinear systems. The design of proposed controller is divided into two steps:

1. The unknown nonlinear system dynamics is identified by SLFN identification scheme.
2. The optimal controller is designed based on the data-based identifier.

In the following sections, we will discuss the establishment of the data-based identifier and the controller design in details.

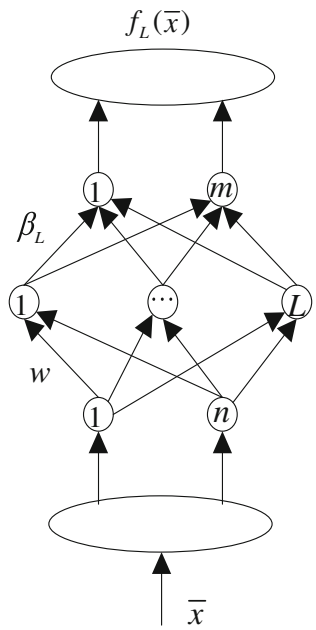


**Fig. 1** The basic SLFN architecture

## 3 The data-based identifier

In this section, the ELM method is introduced and the data-based identifier is established. The structure of SLFN is in Fig. 1.

For $N_1$ arbitrary distinct samples $(\bar{x}(i), \bar{y}(i))$, where $\bar{x}(i) \in \Re^{n_1}$, $\bar{y}(i) \in \Re^{m_1}$, $i = 1, \ldots, N_1$. The weight vectors between the input neurons and the $j$th hidden neuron are $w_j \in \Re^{n_2}$. The weight vectors between the output neurons and the $j$th hidden neuron are $\bar{\beta}_j \in \Re^{m_1}$, which will be designed by ELM method (Zhang et al. 2007). The number of hidden neurons is $L$. The threshold of the $j$th hidden neuron is $b_j$. The hidden layer activation function $g_L(\bar{x})$ is infinitely differentiable, then the mathematically model of SLFN is (Huang and Siew 2004)

$$f_L(\bar{x}(i)) = \sum_{j=1}^{L} \bar{\beta}_j g_L(w_j^T \bar{x}(i) + b_j), \quad i = 1, \ldots, N_1. \quad (3)$$

Unlike the traditional popular implementations in SLFN, in this paper, ELM is used to adjust the output weights. In theory, Refs.Tamura and Tateishi (1997) and Huang (2003) show that the input weights and hidden neurons biases of SLFN do not need be adjusted during training and one may simply randomly assign values to them. To be convenient for explanation, let $\beta_L = [\bar{\beta}_1, \ldots, \bar{\beta}_L]^T_{L\times m_1}$, $\bar{Y} = [\bar{y}(1), \ldots, \bar{y}(N_1)]^T_{N_1\times m_1}$, and

$$H = [h(\bar{x}(1)), \ldots, h(\bar{x}(N_1))]^T = \begin{bmatrix} G(w_1, b_1, \bar{x}(1)) \cdots G(w_L, b_L, \bar{x}(1)) \\ \vdots \\ G(w_1, b_1, \bar{x}(N_1)) \cdots G(w_L, b_L, \bar{x}(N_1)) \end{bmatrix}_{N_1\times L},$$

where $G(w_j, b_j, \bar{x}(i)) = g_L(w_j^T \bar{x}(i) + b_j)$. So we have

$$H\beta_L = \bar{Y}. \quad (4)$$

Based on least-square method, it can be obtained that

$$\beta_L = H^+ \bar{Y}, \quad (5)$$

where $H^+ = (H^T H)^{-1} H^T$.

For SLEN in (3), the output weight $\beta_L$ is the only value we want to obtain. In the following, a theorem is given to prove that $\beta_L$ exists, which means that $H$ is invertible.

**Theorem 1** *If SLFN is defined as in (3), let the hidden neurons number is $L$. For $N_1$ arbitrary distinct input samples $\bar{x}(i)$ and any given $w_j$ and $b_j$, we have $H$ in (4) is invertible.*

*Proof* As input samples $\bar{x}(i)$ are distinct, for any vector $w_j$ according to any continuous probability distribution, then with probability one, $w_j^T x(1), w_j^T x(2), \ldots, w_j^T x(N_1)$ are different from each other. Define the $j$th column of $H$ is $c(j) = [g_L(w_j^T \bar{x}(1) + b_j), g_L(w_j^T \bar{x}(2) + b_j), \ldots, g_L(w_j^T \bar{x}(N_1) + b_j)]^T$, we can have $c(j)$ does not belong to any subspace whose dimension is less than $N_1$ (Huang 2003). It means that for any given $w_j$ and $b_j$, according to any continuous probability distribution, $H$ in (4) can be made full-rank, i.e., $H$ is invertible.

Therefore, the SLFN with ELM method is summarized as follows (Huang et al. 2011):

Step 1. Given a training set $(\bar{x}(i), \bar{y}(i))$, $i = 1, \ldots, N_1$, hidden node output function $G(w_j, b_j, \bar{x}(i))$ and hidden node number $L$.
Step 2. Given arbitrary hidden node parameters $(w_j, b_j)$, $j = 1, \ldots, L$.
Step 3. Calculate the hidden layer output matrix $H$.
Step 4. According to (5) to calculate $\beta_L$.

*Remark 1* ELM algorithm can work with wide type of activation functions, such as sigmoidal functions, radial basis, sine, cosine and exponential functions et al. The feed-forward networks with arbitrary input weights and hidden layer biases can universally approximate any continuous functions on any compact input sets (Huang et al. 2006a).

*Remark 2* It is important to point out that $\beta_L$ in (5) has the smallest norm among all the least-squares solutions of $H\beta_L = \bar{Y}$. As the input weights and hidden neurons biases of SLFN are simply randomly assigned values. So training an SLFN is simply equivalent to finding a least-squares solution of the linear system $H\beta_L = \bar{Y}$. Although almost all learning algorithms wish to reach the minimum training error, however, most of them cannot reach it because of local minimum or infinite training iteration is usually not allowed in applications (Huang et al. 2006a). Fortunately, the special unique solution $\beta_L$ in (5) has the smallest norm among all the least squares solutions.

## 4 Derivation of the iterative ADP algorithm with convergence analysis

For the unknown nonlinear system (1), the data-based identifier is established. Then we can design the iterative ADP algorithm to get the solution of the finite-time optimal control problem.

First, the derivations of the optimal control $u^*(k)$ and $J^*(x(k))$ are given in details. It is known that, for the case of finite horizon optimization, the optimal performance index function $J^*(x(k))$ satisfies (Wang et al. 2011)

$$J^*(x(k)) = \inf_{U(k)} \{J(x(k), U(k, K))\}, \tag{6}$$

where $U(k, K)$ stands for a finite-time control sequence. The length of the control sequence is not assigned.

According to Bellman's optimality principle, the following Hamilton–Jacobi–Bellman (HJB) equation

$$\begin{aligned} J^*(x(k)) = \inf_{u(k)} \{&x^T(k)Qx(k) + u^T(k)Ru(k) \\ &+J^*(x(k+1))\} \end{aligned} \tag{7}$$

holds.

Define the law of optimal control sequence starting at $k$ by

$$U^*(k, K) = arg \inf_{U(k,K)} \{J(x(k), U(k, K))\}, \tag{8}$$

and the law of optimal control vector by

$$\begin{aligned} u^*(k) = arg \inf_{u(k)} \{&x^T(k)Qx(k) \\ &+u^T(k)Ru(k) + J^*(x(k+1))\}. \end{aligned} \tag{9}$$

Therefore, we can have

$$J^*(x(k)) = x^T(k)Qx(k) + u^{*T}(k)Ru^*(k) + J^*(x(k+1)). \tag{10}$$

Based on the above preparation, the finite-time ADP method for unknown system is proposed. The iterative procedure is as follows.

For the iterative step $i = 1$, the performance index function is computed as

$$\begin{aligned} &V^{[1]}(x(k)) \\ &= \inf_{u(k)} \{x^T(k)Qx(k) + u^T(k)Ru(k) + V^{[0]}(x(k+1))\} \\ &= x^T(k)Qx(k) + u^{[1]T}(k)Ru^{[1]}(k) + V^{[0]}(x(k+1)), \end{aligned} \tag{11}$$

where

$$u^{[1]}(x(k)) = \arg \inf_{u(k)} \{x^T(k)Qx(k) + u^T(k)Ru(k) + V^{[0]}(x(k+1))\}, \tag{12}$$

and $V^{[0]}(x(k+1))$ has two expression forms according to two different cases.

If for $x(k)$, there exists $U(k, K) = (u(k))$, s.t. $F(x(k), u(k)) = 0$, then $V^{[0]}(x(k+1))$ is

$$V^{[0]}(x(k+1)) = J(x(k+1), U^*(k+1, k+1)) = 0, \quad \forall x(k+1), \tag{13}$$

where $U^*(k+1, k+1) = (0)$. In this situation, the restrict term $F(x(k), u^{[1]}(k)) = 0$ for (11) is necessary.

If for $x(k)$, there exists $U(k, \bar{K}) = (u(k), u(k+1), \ldots, u(\bar{K}))$, s.t. $F(x(k), U(k, \bar{K})) = 0$, then $V^{[0]}(x(k+1))$ is

$$V^{[0]}(x(k+1)) = J(x(k+1), U^*(k+1, K)), \tag{14}$$

where $U^*(k+1, K) = (u^*(k+1), u^*(k+1), \ldots, u^*(K))$.

For the iterative step $i > 1$, the performance index function is updated as follows

$$\begin{aligned} V^{[i+1]}(x(k)) &= \inf_{u(k)} \{x^T(k)Qx(k) + u^T(k)Ru(k) \\ &\quad +V^{[i]}(x(k+1))\} \\ &= x^T(k)Qx(k) + u^{[i+1]T}(k)Ru^{[i+1]}(k) \\ &\quad +V^{[i]}(x(k+1)), \end{aligned} \tag{15}$$

where

$$\begin{aligned} u^{[i+1]}(x(k)) = \arg \inf_{u(k)} \{&x^T(k)Qx(k) + u^T(k)Ru(k) \\ &+V^{[i]}(x(k+1))\}. \end{aligned} \tag{16}$$

In the above recurrent iterative procedure, the index $i$ is the iterative step and $k$ is the time step. The optimal control and optimal performance index function can be obtained by the iterative ADP algorithm (11)–(16).

In the following part, we will present the convergence analysis of the iterative ADP algorithm (11)–(16).

**Theorem 2** *For an arbitrary state vector $x(k)$, the performance index function $V^{[i+1]}(x(k))$ is obtained by the iterative ADP algorithm (11)–(16), then $\{V^{[i+1]}(x(k))\}$ is a monotonically nonincreasing sequence for $i \geq 1$, i.e., $V^{[i+1]}(x(k)) \leq V^{[i]}(x(k)), \forall i \geq 1$.*

*Proof* The mathematical induction is used to prove the theorem.

First, for $i = 1$, we can have $V^{[1]}(x(k))$ in (11), $V^{[0]}(x(k+1))$ in (13), and the finite-time admissible control sequence $U^*(k, k+1) = (u^{[1]}(k), U^*(k+1, k+1)) = (u^{[1]}(k), 0)$. For $i = 2$, we have

$$\begin{aligned} V^{[2]}(x(k)) &= x^T(k)Qx(k) + u^{[2]T}(k)Ru^{[2]}(k) \\ &\quad +V^{[1]}(x(k+1)). \end{aligned} \tag{17}$$

From (11), we have

$$\begin{aligned} V^{[1]}(x(k+1)) = \inf_{u(k+1)} \{&x^T(k+1)Qx(k+1) \\ &+u^T(k+1)Ru(k+1) + V^{[0]}(x(k+2))\}. \end{aligned} \tag{18}$$

So (17) can be expressed as

$$V^{[2]}(x(k)) = \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + \inf_{u(k+1)}\{x^T(k+1)Qx(k+1) + u^T(k+1)Ru(k+1)\} + V^{[0]}(x(k+2))\}. \tag{19}$$

So (19) can be written as

$$V^{[2]}(x(k)) = \inf_{U(k,k+1)}\sum_{l=k}^{k+1}\{x^T(l)Qx(l) + u^T(l)Ru(l)\}. \tag{20}$$

If $U(k, k+1)$ in (20) is defined as $U(k, k+1) = (u(k), u(k+1)) = (u^{[1]}(k), 0)$, then we have

$$\sum_{l=k}^{k+1}\{x^T(l)Qx(l) + u^T(l)Ru(l)\} = x^T(k)Qx(k) + u^{[1]T}(k)Ru^{[1]}(k) = V^{[1]}(x(k)). \tag{21}$$

So according to (20) and (21), we have $V^{[2]}(x(k)) \leq V^{[1]}(x(k))$, for $i = 1$.

Second, we assume that for $i = j - 1$, the following expression

$$V^{[j]}(x(k)) \leq V^{[j-1]}(x(k)) \tag{22}$$

holds.

Then according to (15), for $i = j$, we have

$$V^{[j+1]}(x(k)) = \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + \inf_{u(k+1)}\{x^T(k+1)Qx(k+1) + u^T(k+1)Ru(k+1) + \cdots + \inf_{u(k+j)}\{x^T(k+j)Qx(k+j) + u^T(k+j)Ru(k+j)\}\cdots\}\}. \tag{23}$$

So we can obtain

$$V^{[j+1]}(x(k)) = \inf_{U(k)}\sum_{l=k}^{k+j}\{x^T(l)Qx(l) + u^T(l)Ru(l)\}. \tag{24}$$

If we let $U(k, k+j) = (u^{[j]}(k), \ldots, u^{[1]}(k+j-1), 0)$ in (24), then we can get

$$\sum_{l=k}^{k+j}\{x^T(l)Qx(l) + u^T(l)Ru(l)\} = x^T(k)Qx(k) + u^{[j]T}(k)Ru^{[j]}(k) + x^T(k+1)Qx(k+1) + u^{[j-1]T}(k+1)Ru^{[j-1]}(k+1) + \cdots + x^T(k+j-1)Qx(k+j-1) + u^{[1]T}(k+j-1)Ru^{[1]}(k+j-1) + x^T(k+j)Qx(k+j). \tag{25}$$

As mentioned in the iterative algorithm, the restrict term $F(x(k), u^{[1]}(k)) = 0, \forall x(k)$ for (11) is necessary. So we can get

$$x(k+j) = F(x(k+j), u^{[1]}(k+j)) = 0. \tag{26}$$

Thus, we have

$$\sum_{l=k}^{k+j}\{x^T(l)Qx(l) + u^T(l)Ru(l)\} = V^{[j]}(x(k)). \tag{27}$$

Therefore, we obtain

$$V^{[j+1]}(x(k)) \leq V^{[j]}(x(k)). \tag{28}$$

For the situation (14), it can easily be proven according to the above method. Therefore, we can conclude that $V^{[i+1]}(x(k)) \leq V^{[i]}(x(k)), \forall i$.

From Theorem 2, it is clear that the iterative performance index function is convergent. So we can define the limitation of the sequence $\{V^{[i+1]}(x(k))\}$ is $V^o(x(k))$. In the next theorem, we will prove that $V^o(x(k))$ satisfies the HJB equation.

**Theorem 3** *Let* $V^o(x(k)) = \lim_{i\to\infty} V^{[i+1]}(x(k))$, *then* $V^o(x(k))$ *satisfies*

$$V^o(x(k)) = \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + V^o(x(k+1))\}. \tag{29}$$

*Proof* According to (15) and (16), for any admissible control vector $\eta(k)$, we have

$$V^{[i+1]}(x(k)) \leq x^T(k)Qx(k) + \eta^T(k)R\eta(k) + V^{[i]}(x(k+1)). \tag{30}$$

From Theorem 2, we can obtain

$$V^o(x(k)) \leq V^{[i+1]}(x(k)). \tag{31}$$

So it can be obtained that

$$V^o(x(k)) \leq x^T(k)Qx(k) + \eta^T(k)R\eta(k) + V^{[i]}(x(k+1)). \tag{32}$$

Let $i \to \infty$, (32) can be written as

$$V^o(x(k)) \leq x^T(k)Qx(k) + \eta^T(k)R\eta(k) + V^o(x(k+1)). \tag{33}$$

As $\eta(k)$ is any admissible control, so we can obtain

$$V^o(x(k)) \leq \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + V^o(x(k+1))\}. \tag{34}$$

On the other side, according to the definition $V^o(x(k)) = \lim_{i\to\infty} V^{[i+1]}(x(k))$, there exists a positive integer $p$ and an arbitrary positive number $\varepsilon$, such that

$$V^{[p]}(x(k)) \geq V^o(x(k)) \geq V^{[p]}(x(k)) - \varepsilon. \tag{35}$$

From (15), we have

$$V^{[p]}(x(k)) = x^T(k)Qx(k) + u^{[p]T}(k)Ru^{[p]}(k) + V^{[p-1]}(x(k+1)). \tag{36}$$

So according to (35) and (36), we have

$$V^o(x(k)) \geq x^T(k)Qx(k) + u^{[p]T}(k)Ru^{[p]}(k) + V^{[p-1]}(x(k+1)) - \varepsilon. \quad (37)$$

As $V^{[p-1]}(x(k+1)) \geq V^o(x(k+1)), \forall p$, then (37) is written as follows

$$V^o(x(k)) \geq x^T(k)Qx(k) + u^{[p]T}(k)Ru^{[p]}(k) + V^o(x(k+1)) - \varepsilon. \quad (38)$$

Hence we have

$$V^o(x(k)) \geq \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + V^o(x(k+1)) - \varepsilon\}. \quad (39)$$

Since $\varepsilon$ is arbitrary, we can have

$$V^o(x(k)) \geq \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + V^o(x(k+1))\}. \quad (40)$$

Thus from (34) and (40), we have

$$V^o(x(k)) = \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + V^o(x(k+1))\}. \quad (41)$$

From Theorems 2 and 3, it can be concluded that $V^o(x(k))$ is the optimal performance index function and $V^o(x(k)) = J^*(x(k))$. So we can have the following corollary.

**Corollary 1** *Let the iterative algorithm be expressed as (11)-(16). Then we have the iterative control $u^{[i]}(k)$ converge to the optimal control $u^*(k)$, as $i \to \infty$, i.e.,*

$$u^*(k) = arg \inf_{u(k)}\{x^T(k)Qx(k) + u^T(k)Ru(k) + J^*(x(k+1))\}. \quad (42)$$

In this section, the iterative control algorithm is proposed for data-based unknown systems with convergence analysis. In next section, the neural network implementation of the iterative control algorithm will be presented.

### 4.1 Neural network implementation of the iterative control algorithm

The input–output data are used to identify the unknown nonlinear system, until the identification error is in the satisfied precision range. Then the data-based identifier is used for the controller design. The diagram of the whole structure is shown in Fig. 2.

In Fig. 2, the SLENs module is the identifier, the action network module is used to approximate the iterative control $u^{[i]}(k)$, and the critic network module is used to approximate the iterative performance index function. The SLFNs with ELM are used in the ADP algorithm, i.e. action network and critic network. The detailed implementation steps are as follows.

Step 1. Train the identifier by input–output data.
Step 2. Choose an error bound $\varepsilon$, and choose randomly an initial state $x(0)$.
Step 3. Calculate the initial finite-time admissible control sequence for $x(0)$, which is $U(0, K) = (u(0), \ldots, u(K))$. The corresponding state sequence is $(x(0), \ldots, x(K+1))$, where $x(K+1) = 0$.
Step 4. For the state $x(K)$, run the iterative ADP algorithm (11)–(13) for $i = 1$, and (15)–(16) for $i > 1$. If $|V^{[i+1]}(x(K)) - V^i(x(K))| < \varepsilon$, then stop.
Step 5. For the state $x(k), k = K-1, \ldots, 0$, run the iterative ADP algorithm (11)–(12) and (14) for $i = 1$, (15)–(16) for $i > 1$. Until $|V^{[i+1]}(x(k)) - V^i(x(k))| < \varepsilon$.
Step 6. Stop.

## 5 Simulation study

To evaluate the performance of our iterative ADP algorithm for the data-based identifier, an example is provided in this section.

Consider the following nonlinear system (Wang et al. 2011).

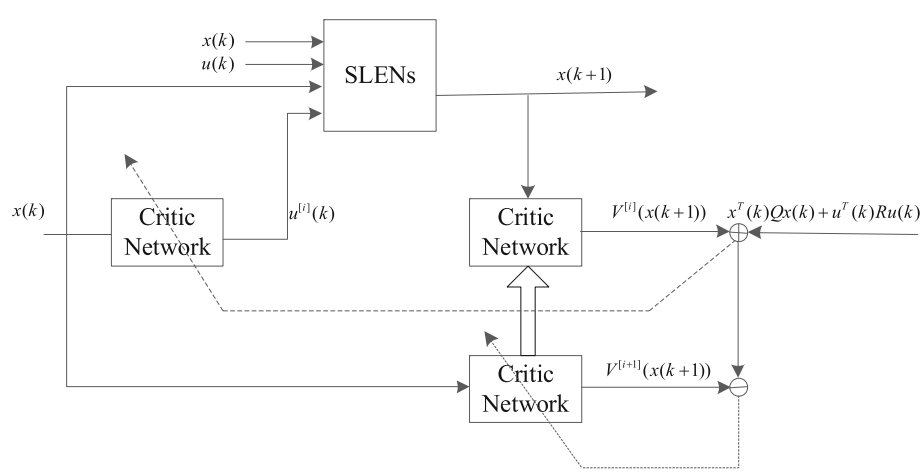


**Fig. 2** The basic structure of the proposed control method

$$x(k+1) = x(k) + sin(0.1x^2(k) + u(k)), \tag{43}$$

where the state $x(k) \in \Re$ and the control $u(k) \in \Re$.

In this paper, 5,000 sampling points are used to train the SLFN identifier. The number of hidden neurons is 20. The weight vectors between the input neurons and the $j$th hidden neuron are selected in (0, 1). The threshold of the hidden neuron is selected in (0, 1). For 50 test points, we get the identification results in Fig. 3. The dashed line is the test points, the solid line is the output of the data-based identifier, and the line with sign "×" is the identification error. From Fig. 3, we can see that the identifier reconstructs the unknown nonlinear system accurately.

Based on the identification results, the optimal ADP controller is designed. The initial state for system (43) is $x(0) = 1.5$. For implementation the proposed iteration algorithm in this paper, two neural networks which are SLFNs with ELM are used to approximate the action network and critic network, respectively. To demonstrate the effectiveness of the proposed scheme, we implement the iterative algorithm by neural networks with ELM and BP methods, respectively. The maximal iteration step is 50 for two kinds of neural networks. The convergence precision of ELM is $10^{-6}$, and the convergence precision of BP is $10^{-4}$. We get the simulation results in Figs. 4, 5, 6, 7, 8, 9. Figures 4 and 5 are trajectories of the iterative performance index function obtained by ELM and BP, respectively. In Fig. 4, after 5 iterative steps, the iterative performance index function is convergent. While in Fig. 5, it costs 15 iterative steps. Figures 6 and 7 are the state trajectories obtained by ELM method and BP method, respectively. Figures 8 and 9 are the control trajectories obtained by ELM method and BP method, respectively. By ELM method, the trajectories of state and control are convergent after 4 time steps. While by BP method, it costs 15 time steps. From the figures we can see that the results of ELM method are faster and smoother than the results of BP method. It can be concluded that the learning speed of ELM method is faster than BP method while obtaining better generalization performance.

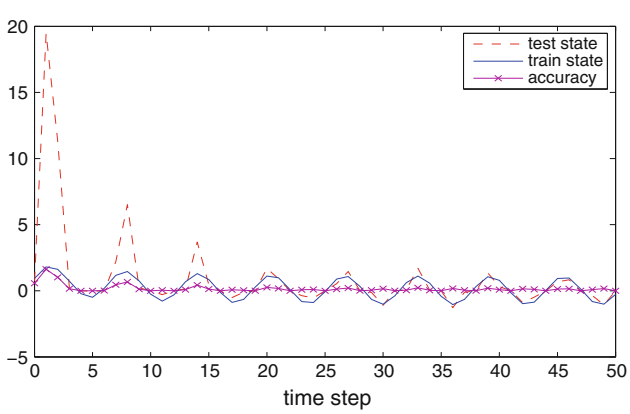


**Fig. 3** The train results of the data-based identifier

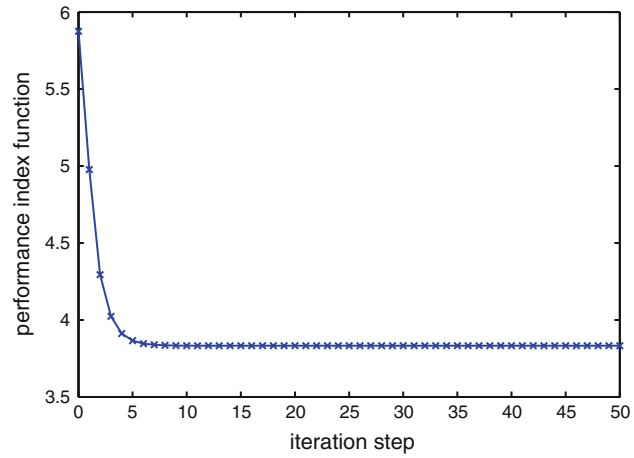


**Fig. 4** The performance index function obtained by ELM method

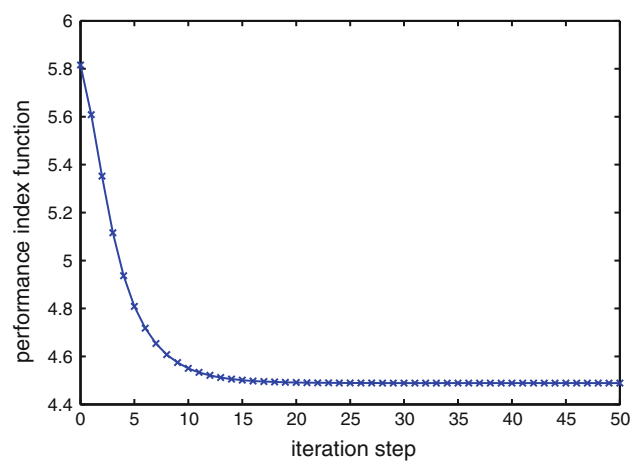


**Fig. 5** The performance index function obtained by BP method

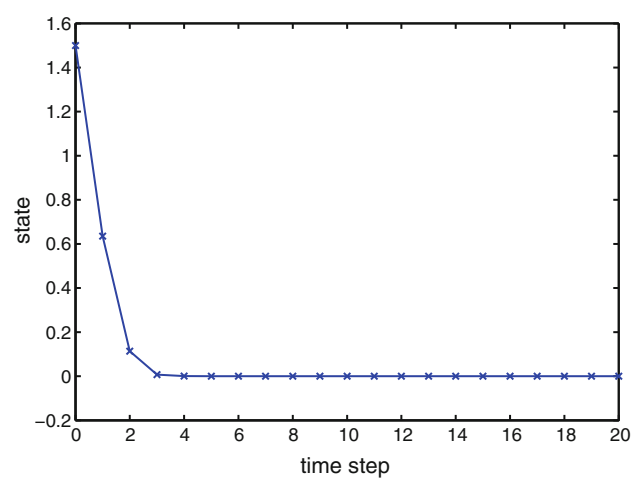


**Fig. 6** The state obtained by ELM method

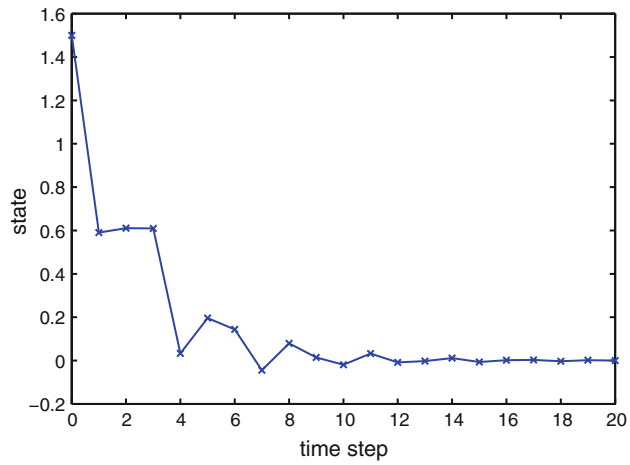


**Fig. 7** The state obtained by BP method

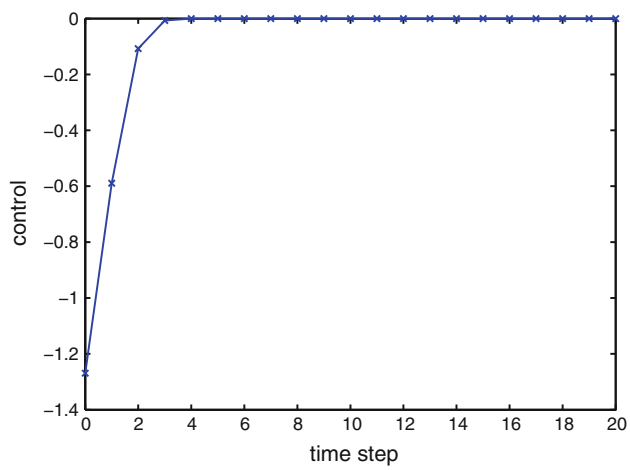


**Fig. 8** The control obtained by ELM method

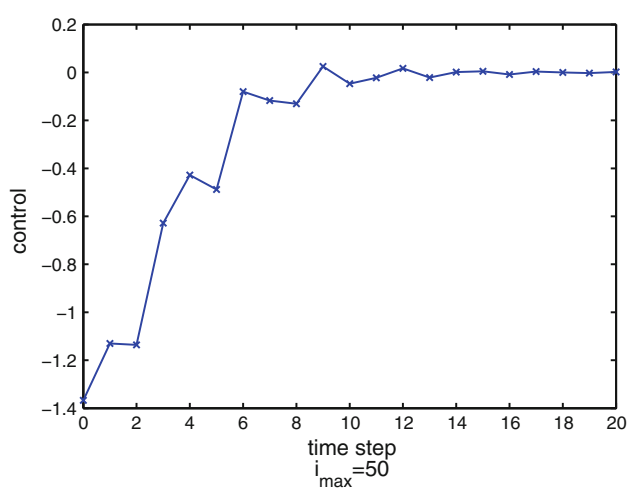


**Fig. 9** The control obtained by BP method

## 6 Conclusions

This paper studied the ELM method for optimal control of unknown nonlinear systems. Using the input–output data, a data-based identifier was established. The finite-time optimal control scheme was proposed based on iterative ADP algorithm. The results of theorems showed that the proposed iterative algorithm was convergent. The simulation study have demonstrated the effectiveness of the proposed control algorithm.

**Acknowledgments** This work was supported in part by the Open Research Project from SKLMCCS (Grant no. 20120106), the Fundamental Research Funds for the Central Universities (Grant no. FRF-TP-13-018A), the China Postdoctoral Science Foundation (Grant no. 2013M530527), and the National Natural Science Foundation of China (Grants no. 61304079, 61125306, 61034002, 61374105), and Beijing Natural Science Foundation (Grant no. 4132078).

## References

Bellman RE (1957) Dynamic programming. Princeton University Press, Princeton

Duncan TE, Guo L, Pasik-Duncan B (1999) Adaptive continuous-time linear quadratic gaussian control. IEEE Trans Autom Control 44(9):1653–1662

Fernández-Navarro F, Hervás-Martínez C, Gutierrez PA (2013) Generalised Gaussian radial basis function neural networks. Soft Comput 17:519–533

Gabasov R, Kirillova FM, Balashevich NV (2000) Open-loop and closed-loop optimization of linear control systems. Asian J Control 2(3):155–168

Guardabassi GO, Savaresi SM (May 2000) Virtual reference direct design method: an off-line approach to data-based control system design. IEEE Trans Autom Control 45(5):954–959

He P, Jagannathan S (2007) Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. IEEE Trans Syst Man Cybern Part B: Cybern 37(2):425–436

Hornik K (1991) Approximation capabilities of multilayer feedforward networks. Neural Netw 4:251–257

Huang G-B (2003) Learning capability and storage capacity of two-hidden-layer feedforward networks. IEEE Trans Neural Netw 14(2):274–281

Huang G-B, Siew C-K (2004) Extreme learning machine: RBF network case. In: The Proceedings of the eighth international conference on control, automation, robotics and vision (ICARCV 2004), vol 2. Kunming, Dec 6–9. p 1029–1036

Huang G-B, Zhu Q-Y, Siew C-K (2006a) Extreme learning machine: theory and applications. Neurocomputing 70:489–501

Huang G-B, Chen L, Siew C-K (2006b) Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes. IEEE Trans Neural Netw 17(4):879–892

Huang G-B, Wang DH, Lan Y (2011) Extreme learning machines: a survey. Int J Mach Learn Cybern 2(2):107–122

Jagannathan S (2006) Neural network control of nonlinear discrete-time systems. CRC Press, Boca Raton

Jin X, Yang G, Peng L (2012) Robust adaptive tracking control of distributed delay systems with actuator and communication failures. Asian J Control 14(5):1282–1298

Leshno M, Lin VY, Pinkus A, Schocken S (1993) Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Netw 6:861–867

Liu D, Zhang H (2005) A neural dynamic programming approach for learning control of failure avoidance problems. Int J Intell Control Syst 10(1):21–32

Liu D, Wei Q (2013) Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems. IEEE Trans Syst Man Cybern Part B: Cybern 43(2):779–789

Maji K, Pratihar DK, Nath AK (2013) Analysis and synthesis of laser forming process using neural networks and neuro-fuzzy inference system. Soft Comput 17:849–865

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

Richert D, Masaud K, Macnab CJB (2013) Discrete-time weight updates in neural-adaptive control. Soft Comput 17:431–444

Seiffertt J, Sanyal S, Wunsch DC (2008) Hamilton-Jacobi-Bellman equations and approximate dynamic programming on time scales. IEEE Trans Syst Man Cybern Part B: Cybern 38(4):918–923

Song R, Zhang H (2013) The finite-horizon optimal control for a class of time-delay affine nonlinear system. Neural Comput Appl 22(2):229–235

Song R, Zhang H, Luo Y, Wei Q (2010) Optimal control laws for time-delay systems with saturating actuators based on heuristic dynamic programming. Neurocomputing 73(16–18):3020–3027

Tamura S, Tateishi M (1997) Capabilities of a fourlayered feedforward neural network: four layers versus three. IEEE Trans Neural Netw 8(2):251–255

Wang F, Jin N, Liu D, Wei Q (2011) Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with $\epsilon$-error bound. IEEE Trans Neural Netw 22:24–36

Wei Q, Liu D (2012) An iterative $\varepsilon$-optimal control scheme for a class of discrete-time nonlinear systems with unfixed initial state. Neural Netw 32:236–244

Werbos PJ (1992) Approximate dynamic programming for real-time control and neural modeling, chap. 13. In: White DA, Sofge DA (eds) Handbook of intelligent control: neural, fuzzy, and adaptive approaches. Van Nostrand Reinhold, New York

Werbos PJ (2008) ADP: The key direction for future research in intelligent control and understanding brain intelligence. IEEE Trans Syst Man Cybern Part B: Cybern 38(4):898–900

Yu W (2009) Recent advances in intelligent control systems. Springer-Verlag, London

Zhang R, Huang G-B, Sundararajan N, Saratchandran P (2007) Multicategory classification using extreme learning machine for microarray gene expression cancer diagnosis. IEEE/ACM Trans Comput Biol Bioinform 4(3):485–495

Zhang H, Wei Q, Luo Y (2008) 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: Cybern 38(4):937–942

Zhang H, Cui L, Zhang X, Luo Y (2011a) Data-Driven Robust Approximate Optimal Tracking Control for Unknown General Nonlinear Systems Using Adaptive Dynamic Programming Method. IEEE Trans Neural Netw 22(12):2226–2236

Zhang H, Wei Q, Liu D (2011b) An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. Automatica 47(1):207–214

Zhang H, Song R, Wei Q (2011c) 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):1851–1862

Zhang H, Wei Q, Liu D (2011d) An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. Automatica 47(1):207–214

Zhang H, Song R, Wei Q, Zhang T (2011e) 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):1851–1862

Zhao Y, Patek SD, Beling PA (2008) Decentralized Bayesian search using approximate dynamic programming methods. IEEE Trans Syst Man Cybern Part B: Cybern 38(4):970–975