A neural network based online learning and control approach for Markov jump systems

Xiangnan Zhong, Haibo He, Huaguang Zhang, Zhanshan Wang

ABSTRACT

In this paper, we propose an optimal online control method for discrete-time nonlinear Markov jump systems (MJSs). The Markov chain and the weighted sum technique are introduced to convert the Markov jumping problem into an optimal control problem. We then use adaptive dynamic programming (ADP) to accomplish online learning and control with specific learning algorithm and detailed stability analysis, including the convergence of the performance index function sequence and the existence of the corresponding admissible control input. Neural networks are applied to implement this ADP approach and online learning method is used to tune the weights of the critic and the action networks. Two different numerical examples are given to demonstrate the effectiveness of the proposed method.

1. Introduction

Adaptive dynamic programming (ADP) has been widely recognized as one of the “core methodologies” to achieve optimal control in stochastic process in a general case to achieve intelligent control [1,2]. Taking the advantage of approximating the solutions of optimal control problems in equivalent to solve the Hamilton–Jacobi–Bellman (HJB) equation, this method has attracted significantly increasing attention in recent years. Extensive efforts and promising results in both theoretical research and engineering applications have been achieved over the past decades. Among these achievements, we highlight Al-Tamimi et al. [3], Abu-Khalaf et al. [4], Wei and Liu [5–7], Lewis and Vanwoudakis [8,9], He et al. [10–12], Zhang et al. [13–15], Si et al. [16–18], He and Jagannathan [19–21], Seiffert et al. [22], Zhong et al. [23,24] and Lin et al. [25,26] from the theoretical perspective that are closely related to the research presented in this paper. These achievements cover a large variety of problems, including system stability, convergence proof, optimal control, and state prediction. Interested readers can refer to the two important handbooks [27,28] on ADP for many other successful architectures, algorithms and challenging engineering applications.

On the other hand, there has been extensive interest in the stability analysis and controller design of the Markov jump systems (MJSs) over the past decades due to its powerful modeling capability for power systems [29,30], aerospace systems [31], and manufacturing systems [32–34]. In practice, random parameters change may exist in these systems resulting from sudden environmental disturbances, abrupt changes of the operating point, or component failure or repairs. These make the systems that cannot be easily modeled. The studies of MJSs build a bridge between these architecture systems and the theoretical analysis. However, many of the research in this field completely depend on the accuracy system functions [35–39], which narrows the range of application of this powerful modeling method. How to solve the problems of MJSs without the knowledge of system functions is a challenging topic.

This paper proposes an optimal control method for a class of discrete-time nonlinear Markov jump systems without the requirement of system functions by using ADP technique. Note that, generally, ADP can be categorized into three typical structures which are heuristic dynamic programming (HDP), dual heuristic dynamic programming (DHP), and globalized dual heuristic dynamic programming (GDHP). This paper is focused on the HDP technique for stabilizing the MJSs. The main contribution of this work is to introduce the ADP method into the field of MJSs by transforming the MJSs control problem with multi-subsystem into a single-objective optimal control problem. Moreover, unlike the traditional method to solve the MJSs problem, such as linear matrix inequality, our approach based on ADP technique is an adaptive and learning process, which means when the parameters...
of the systems are changed, our approach can still find the optimal controller adaptively. Besides, the convergence of the proposed ADP approach is provided in detail. Two numerical examples are presented to verify the validity of the proposed method.

The rest of this paper is organized as follows. In Section 2, we formulate the MJJs control problem analyzed in this paper. The performance index function for the whole MJJs is obtained by combining the performance index functions for the subsystems using Markov chain and weighted sum technique. The ADP method for discrete-time nonlinear MJJs is established in Section 3. Section 4 presents the detailed convergence analysis of the performance index function and the existence of the admissible control for the proposed optimal control scheme. Neural networks are used in Section 5 to implement this ADP scheme. The online learning method is also applied in this section to tune the weights of the critic and the action networks. In Section 6, two numerical examples, including one with two jumping modes and one with four jumping modes, are presented to demonstrate the effectiveness of the proposed approach. Finally, Section 7 concludes the paper.

2. Problem statement

The discrete-time nonlinear MJJs can be described by the following equation:

\[ x(k+1) = F(x(k), u(k), \theta(k)), \quad k \geq 0 \tag{1} \]

where \( x(k) \in \mathbb{R}^n \) is the system state with the initial state \( x(0) \) and \( u(k) \in \mathbb{R}^m \) is the control vector. \( F(x(k), u(k), \theta(k)) \) denotes the unknown system function and \( F(0,0,0) = 0 \). We assume that \( F(x(k), u(k), \theta(k)) \) is Lipschitz continuous. \( \theta(k) \geq 0 \) is the discrete-time Markov chain, which refers to the active mode of the whole system in each step and takes values in a finite set \( S = \{1, 2, \ldots, l\} \), where \( l \) is the number of the subsystems. The elements in the Markov chain are given by

\[ p_{ij} = \text{Prob}(\theta(k+1) = j | \theta(k) = i) \tag{2} \]

which denotes the transition probability that the next active subsystem is the \( j \)th one given that the current active subsystem is the \( i \)th one. Hence, we know \( p_{ii} \geq 0, \forall i,j \in S \) and \( \sum_{j=1}^{l} p_{ij} = 1 \).

Define the performance index function for each subsystem as follows:

\[ J_i(x(k), \theta(k)) = \sum_{t=k}^{\infty} \alpha^{-t} U_i(x(t), u(t), \theta(t)) \tag{3} \]

where the utility function \( U_i(x(t), u(t), \theta(t)) = Q_i(x(t), \theta(t)) + u(t)^T R_i \theta(t) u(t) \) is positive definite, i.e., \( U_i(x(t), u(t), \theta(t)) > 0 \) if and only if \( x(t) = 0 \) and \( u(t) = 0 \); otherwise \( U_i(x(t), u(t), \theta(t)) > 0 \). And, \( 0 < \alpha < 1 \) is the discount factor.

In the following part, we use \( F_i(x(k), u(k)), J_i(x(k)), U_i(x(k), u(k)), Q_i(x(k), \theta(k)) \) to represent the notation \( F(x(k), u(k), \theta(k)), J(x(k), u(k), \theta(k)), U(x(k), u(k), \theta(k)), Q(x(k), \theta(k)) \) to simplify the presentation.

For optimal control problem, it is desired to find an optimal control \( u^*(k) \) to minimize the performance index function for system (1). However, due to the existence of the transition probabilities (2), we cannot just add all the performance index functions of the subsystems to act as that of the whole MJJs. Here, we use

\[
\begin{align*}
J(x(k)) &= \sum_{i=1}^{l} p_{i} J_i(x(k)) + p_{12} J_2(x(k)) + \cdots + p_{1l} J_l(x(k)) \\
J_2(x(k)) &= p_{21} J_1(x(k)) + p_{22} J_2(x(k)) + \cdots + p_{2l} J_l(x(k)) \\
&\vdots \\
J_l(x(k)) &= p_{l1} J_1(x(k)) + p_{l2} J_2(x(k)) + \cdots + p_{ll} J_l(x(k)) \\
\end{align*}
\tag{4}
\]

to reconstruct the performance index function according to the Markov chain (2). Then, by using the weighted sum technique, the final performance index function for MJJs is obtained as

\[ J(x(k)) = \omega_1 J_1(x(k)) + \omega_2 J_2(x(k)) + \cdots + \omega_l J_l(x(k)) \tag{5} \]

where \( \omega_i > 0 \) is the weight vector and \( \sum_{i=1}^{l} \omega_i = 1 \).

Hence, the control vector \( u(k) \) needs to be found to minimize the performance index function (5) and make the MJJs achieve stability. Note that this control law must not only stabilize the system on the compact set \( \Omega \subset \mathbb{R}^n \), but also guarantee that (5) is finite, which is called admissible control.

**Definition 1.** A control law is said to be an admissible control with respect to (5) on \( \Omega \), if \( u(k) \) is continuous on \( \Omega \) and can stabilize system (1) for all \( x(0) \in \Omega \), \( u(k) = 0 \) as \( x(k) = 0 \), and for \( \forall x(k), J(x(k)) \) is finite.

3. ADP approach for optimal control problem of nonlinear MJJs

In this section, the ADP approach for discrete-time nonlinear MJJs is presented based on the performance index function (5). Eq. (5) can be expanded as

\[
J(x(k)) = \omega_1 J_1(x(k)) + \omega_2 J_2(x(k)) + \cdots + \omega_l J_l(x(k)) \\
= (\omega_1 p_{11} + \omega_2 p_{21} + \cdots + \omega_l p_{l1}) J_1(x(k)) \\
+ (\omega_1 p_{12} + \omega_2 p_{22} + \cdots + \omega_l p_{l2}) J_2(x(k)) \\
+ \cdots + (\omega_1 p_{1l} + \omega_2 p_{2l} + \cdots + \omega_l p_{ll}) J_l(x(k)) \\
= D_1 J_1(x(k)) + D_2 J_2(x(k)) + \cdots + D_l J_l(x(k)) \\
= \sum_{i=1}^{l} \sum_{k=1}^{\infty} (\alpha^{-k} D_i U_i(x(t), u(t))) \\
\tag{6}
\]

where \( D_i = \sum_{j=1}^{l} \omega_j p_{ij} > 0 \). Hence, Eq. (6) is positive definite, i.e., the above performance index function serves as a Lyapunov function.

The equivalent equation of (6) is given by the Bellman equation:

\[
J(x(k)) = \sum_{i=1}^{l} (D_i U_i(x(k), u(k))) + \sum_{i=1}^{l} \sum_{k=1}^{\infty} \alpha^{-k} D_i U_i(x(t), u(t)) \\
= \sum_{i=1}^{l} (D_i U_i(x(k), u(k))) + \alpha \sum_{i=1}^{l} \sum_{k=1}^{\infty} \alpha^{-k} D_i U_i(x(t), u(t)) \\
= D^T U(x(k), u(k)) + df^*(x(k+1)) \\
\tag{7}
\]

where \( D = (D_1, D_2, \ldots, D_l)^T \), \( U_i(x(k), u(k)) = (U_1(x(k), u(k)), U_2(x(k), u(k)), \ldots, U_l(x(k), u(k)))^T \). Depending on Bellman’s optimality principle, the optimal performance index function \( J^*(x(k)) \) is time invariant and satisfies the discrete-time HJB equation:

\[
J^*(x(k)) = \min_{u(k)} \left[ D^T U(x(k), u(k)) + df^*(x(k+1)) \right] \\
\tag{8}
\]

Besides, the optimal control \( u^*(k) \) satisfies the first-order necessary condition, which is obtained by gradient of the right-hand side of (8) with respect to \( u(k) \) as

\[
\alpha D^T U(x(k), u(k)) + \alpha \left( \frac{\partial J^*(x(k+1))}{\partial x(k+1)} \right)^T \frac{\partial J^*(x(k+1))}{\partial x(k+1)} = 0 \\
\tag{9}
\]

Therefore, the optimal control policy can be expressed as

\[
u^*(k) = \frac{\alpha}{2} \sum_{i=1}^{m} D_i R_i^{-1} \left( \frac{\partial f_i(x(k), u(k))}{\partial u(k)} \right)^T \frac{\partial J^*(x(k+1))}{\partial x(k+1)} \\
\tag{10}
\]

where \( J^*(x(k)) \) is solved in the following HJB equation:

\[
J^*(x(k)) = \sum_{i=1}^{l} D_i Q_i(x(k)) + \alpha^2 \frac{1}{4} \left( \frac{\partial J^*(x(k+1))}{\partial u(k)} \right)^T \left( \frac{\partial J^*(x(k+1))}{\partial u(k)} \right) \\
\tag{11}
\]
\[
\left( \sum \limits_{i=1}^{\infty} D_i \right)^{-1} \left( \frac{\partial F_i(x(k), u(k))}{\partial u(k)} \right)^T \frac{\partial^2 \Phi(x(k+1))}{\partial u(k)^2} + \alpha \frac{\partial^2 \Phi(x(k+1))}{\partial u(k)^2} \tag{11}
\]

in which \( i \in S \).

Note that for the general nonlinear optimal control problem, this HJB equation (11) cannot be solved exactly. In this paper, ADP method is established to approximate the optimal performance index function \( J^*(x(k)) \).

Set the initial value of the performance index function as \( J^0(x(0)) = 0 \), and hence we solve for \( u^0(k) \) as
\[
u^0(k) = \arg \min_{u(k)} \left[ D^T U(x(k), u(k)) + \alpha J^0(x(k+1)) \right]
\tag{12}
\]

Depending on (12), we can update the performance index function by computing
\[
J^1(x(k)) = \min_{u(k)} \left[ D^T U(x(k), u(k)) + \alpha J^0(x(k+1)) \right] = D^T U(x(k), u^0(k)) + \alpha J^0(x(k+1))
\tag{13}
\]

Because \( J^0(x) = 0 \), it follows
\[
J^1(x(k)) = D^T U(x(k), u^0(k))
\tag{14}
\]

Consider (12) and (13). Therefore, we obtain the ADP algorithm for optimal control problem of nonlinear MJSs by iterating between
\[
u^{(n)}(k) = \arg \min_{u(k)} \left[ D^T U(x(k), u(k)) + \alpha J^{(n)}(x(k+1)) \right]
\tag{15}
\]

and
\[
J^{(n+1)}(x(k)) = \min_{u(k)} \left[ D^T U(x(k), u(k)) + \alpha J^{(n)}(x(k+1)) \right] = D^T U(x(k), u^{(n)}(k)) + \alpha J^{(n)}(x(k+1))
\tag{16}
\]

where \( k \) is the time index, \( i \) is the index of the active subsystem at time \( k \), and \( n \) is the iteration index.

4. Convergence analysis of the proposed ADP approach

In this section, the detailed stability analysis, including the convergence of the proposed performance index function of the MJSs and the existence of the corresponding control input, is provided. Let us start with a lemma which is useful in the following proof.

**Lemma 1.** [3] Let \( \eta(k) \) be any stabilizing and admissible control policy and \( \Lambda^0(x) = J^0(x) = 0 \), where \( \Lambda^0(x) \) is updated as
\[
\Lambda^{(n+1)}(x(k)) = D^T U(x(k), \eta(k)) + \alpha \Lambda^{(n)}(x(k+1))
\tag{17}
\]

where \( U(x(k), \eta(k)) = (U_1(x(k), \eta(k)), U_2(x(k), \eta(k)), \ldots, U_s(x(k), \eta(k)))^T \), \( U_i(x(k), \eta(k)) = \eta_i(k) R_i \), \( i \in S \). Then, \( \Lambda^0(x(k)) \leq \Lambda^{(n)}(x(k)) \).

This lemma can be easily proved by noticing that \( J^{(n)}(x(k)) \) is the result when control \( u(k) \) minimizes the right-hand side of (17).

Now, three important theorems are presented to show the stability of the proposed ADP approach.

**Theorem 1.** Let the performance index function sequence \( J^{(n)}(x(k)) \) be defined as in (16). If \( J^{(0)}(x(k)) = 0 \), then \( J^{(n)}(x(k)) \) is a monotonically non-decreasing sequence, i.e., \( J^{(n)}(x(k)) \leq J^{(n+1)}(x(k)) \).

**Proof.** The new sequence \( \Lambda^{(n)}(x(k)) \) is defined as Eq. (17). If \( \eta(k) = u^{(n)}(k) \), it follows that
\[
\Lambda^{(n)}(x(k)) = D^T U(x(k), \eta(k)) + \alpha \Lambda^{(n-1)}(x(k+1)) = D^T U(x(k), u^{(n)}(k)) + \alpha \Lambda^{(n-1)}(x(k+1)) \tag{18}
\]

In the following part, we prove \( J^{(n+1)}(x(k)) \geq \Lambda^{(n)}(x(k)) \) by mathematical induction. Let us start with \( n = 0 \). We know that \( J^{(0)}(x(k)) = \Lambda^{(0)}(x(k)) = 0 \), then
\[
J^{(1)}(x(k)) = \Lambda^{(0)}(x(k)) = D^T U(x(k), u^{(0)}(k)) \geq 0 \tag{19}
\]

Thus, for \( n = 0 \), we can obtain
\[
J^{(1)}(x(k)) \geq \Lambda^{(0)}(x(k)) \tag{20}
\]

Now, we assume it holds for \( n - 1 \), i.e.,
\[
J^{(n)}(x(k)) - \Lambda^{(n-1)}(x(k)) \geq 0 \tag{21}
\]

By subtracting (18) from (16), it follows
\[
J^{(n+1)}(x(k)) - \Lambda^{(n)}(x(k)) = D^T U(x(k), u^{(n)}(k)) + \alpha \Lambda^{(n-1)}(x(k+1)) - (D^T U(x(k), u^{(n)}(k)) + \alpha \Lambda^{(n-1)}(x(k+1))) = \alpha \Lambda^{(n-1)}(x(k+1)) - \Lambda^{(n-1)}(x(k+1)) \geq 0 \tag{22}
\]

which completes the proof of \( J^{(n+1)}(x(k)) \geq \Lambda^{(n)}(x(k)) \).

On the other side, we obtain \( \Lambda^{(n)}(x(k)) \leq \Lambda^{(n)}(x(k)) \) from Lemma 1, hence \( J^{(n)}(x(k)) \leq \Lambda^{(n)}(x(k)) \leq \Lambda^{(n)}(x(k)) \) for any \( n = 0, 1, 2, 3, \ldots \), which means \( J^{(n)}(x(k)) \leq \Lambda^{(n)}(x(k)) \) for any iterative step, i.e., \( J^{(n)}(x(k)) \) is a monotonically non-decreasing sequence. The conclusion holds.

From the above theorem, we know that the performance index function sequence (16) for MJSs is monotonically non-decreasing. The following theorem shows that this non-decreasing sequence has an upper bound.

**Theorem 2.** Define the performance index function sequence as (16). If the MJSs (1) are controllable, then there exists an upper bound \( C(x(k)) \) such that \( 0 \leq J^{(n)}(x(k)) \leq J^\infty(x(k)) \leq C(x(k)) \).

**Proof.** From Eq. (3), we know that the elements in the obtained performance index function sequence \( J^{(n)}(x(k)) \) for MJSs are all positive values. Therefore, the left-hand side of the conclusion \( 0 \leq J^{(n)}(x(k)) \leq J^\infty(x(k)) \) holds. Now, we prove that this positive sequence has an upper bound.

Motivated by the research in [3,7], we obtain the following equations:
\[
\Lambda^{(n+1)}(x(k)) = D^T U(x(k), \eta(k)) + \alpha \Lambda^{(n)}(x(k+1)) = D^T U(x(k), \eta(k)) + \alpha \Lambda^{(n-1)}(x(k+1)) + \alpha^2 \Lambda^{(n-2)}(x(k+2))
\]

Because \( \Lambda^{(0)}(x) = 0 \), it follows that
\[
\Lambda^{(n+1)}(x(k)) = \sum_{i=0}^{n} \alpha^i D^T U(x(k+i), \eta(k+i)) = \sum_{i=0}^{n+k} \alpha^i D^T U(x(k+i), \eta(k+i)) \tag{24}
\]

Letting \( n \to \infty \), \( \lim_{n \to \infty} \Lambda^{(n+1)}(x(k)) = \Lambda^\infty(x(k)) \), Eq. (24) becomes
\[
\Lambda^\infty(x(k)) = \sum_{i=k}^{\infty} \alpha^i D^T U(x(k+i), \eta(k+i)) \tag{25}
\]

By using Lemma 1, we know \( J^{(n)}(x(k)) \leq \Lambda^{(n)}(x(k)) \), which can be rewritten as \( J^\infty(x(k)) \leq \Lambda^\infty(x(k)) \) when \( n \to \infty \). Combining this and...
Consider the performance index function and the networks. First, we prove that the performance index function (26) can converge to its optimal value. Consider the definition of the upper bound $C(x(k))$. Because $\eta(k)$ is defined as an admissible control, if $\eta(k)$ is the infinite step of the control input, it follows that

$$J^\infty(x(k)) = C(x(k)) \geq f^*(x(k))$$

(27)

On the other hand, since $f^0(x(k)) \leq A^0(x(k))$, which can be rewritten as $J^\infty(x(k)) \leq A^\infty(x(k)) = \sum_{t=1}^{\infty} \alpha_t^{-1} \frac{d}{D}U(x(t), \eta(t))$, we obtain

$$J^\infty(x(k)) \leq \sum_{t=1}^{\infty} \alpha_t^{-1} \frac{d}{D}U(x(t), u^*(t))$$

(28)

by setting $\eta(k) = u^*(k)$, which means $f^0(x(k)) \leq f^*(x(k))$. From (27), we know $f^0(x(k)) \leq f^*(x(k))$. Hence, $J^\infty(x(k)) = f^*(x(k))$, i.e., $f^0(x(k))$ converge to the optimal value $f^*(x(k))$.

Next, the convergence of the corresponding control law sequence $u^{n}(k)$ is shown as follows. From (16), we have

$$J^{n+1}(x(k)) = D^\frac{1}{2}U(x(k), u^{n}(k)) + \alpha f^0(x(k+1))$$

(29)

When $n \to \infty$, the above equation becomes

$$\lim_{n \to \infty} J^{n+1}(x(k)) = \lim_{n \to \infty} D^\frac{1}{2}U(x(k), u^{n}(k)) + \alpha \lim_{n \to \infty} f^0(x(k+1))$$

(30)

As shown above, $\lim_{n \to \infty} f^0(x(k)) = f^*(x(k))$, we can further write that

$$J^*(x(k)) = D^\frac{1}{2}U(x(k), u^*(k)) + \alpha f^*(x(k+1))$$

(31)

Since $f^*(x(k)) = D^\frac{1}{2}U(x(k), u^*(k)) + \alpha f^*(x(k+1))$, we can conclude $u^*(k) = u^*(k)$. Hence, the conclusion holds.

Considering Theorems 1–3, we know that the proposed performance index function sequence (16) can monotonically non-decrease to the optimal value $f^*(x(k))$ and the corresponding admissible control input can also converge to its optimal value, i.e., as $n \to \infty$, $f^0(x(k)) \to f^*(x(k))$ and $u^{n}(k) \to u^*(k)$.

5. Implementation of the ADP method by using neural networks

Neural networks are used to implement the proposed ADP method in this paper. The architecture of the implementation process is shown in Fig. 1. Two neural networks are included in this structure [18]. One is the critic network, which is applied to approximate the performance index function of MJSS, and another one is the action network, which is to estimate the control law sequence of the proposed systems. These two networks are used iteratively to obtain the optimal performance index function and control law for MJSS. Online learning is applied to tune the weights of these two networks. The detailed implementation process of these two networks is presented as follows.

The approximated performance index function and control law can be formulated as

$$\hat{J}(x(k)) = W_{c2}^T(k)\Psi(y(k))$$

(32)

and

$$\hat{u}(k) = W_{a2}^T(k)\Psi(t(k))$$

(33)

where $W_{c2}(k)$ and $W_{a2}(k)$ are the weight matrices between the hidden and the output layer of the critic and the action networks, respectively. $y_k = W_{c1}^T(k)x^T(k), \hat{u}(k)$ in which $W_{c1}(k)$ denotes the weight matrix between the input and the hidden layer of the critic network, $t_k = W_{a1}^T(k)x(k)$, where $W_{a1}(k)$ denotes the weight matrix between the input and the hidden layer of the action network. $\Psi(\cdot)$ is a sigmoid function

$$\Psi(\cdot) = \frac{1}{1+e^{-\cdot}}$$

(34)

First, for critic network, define the error function according to [16]

$$e_{c}(k) = \hat{J}(x(k)) - \hat{J}(x(k-1)) - D^\frac{1}{2}U(x(k), \hat{u}(k))$$

(35)

The goal is to minimize the squared error:

$$E_c(k) = \frac{1}{2} e_{c}^2(k)$$

(36)

Hence, we obtain the critic network weight adjustments for the hidden to the output layer:

$$\Delta W_{c2}(k) = \beta_c \left( - \frac{\partial E_{c}(k)}{\partial W_{c2}(k)} \right)$$

(37)

and for the input to the hidden layer

$$\Delta W_{c1}(k) = \beta_c \left( - \frac{\partial E_{c}(k)}{\partial W_{c1}(k)} \right)$$

(38)

where $\beta_c > 0$ is the learning rate of the critic network. Next, adjusting the action network weights is to minimize the error:

$$e_{a}(k) = \hat{J}(x(k)) - U_c(x(k))$$

(39)

$$E_a(k) = \frac{1}{2} e_{a}^2(k)$$

where $U_c$ is the desired ultimate objective. Therefore, we obtain the action network weight adjustments for the hidden to the output layer:

$$\Delta W_{a2}(k) = \beta_a \left( - \frac{\partial E_{a}(k)}{\partial W_{a2}(k)} \right)$$

(40)

and for the input to the hidden layer

$$\Delta W_{a1}(k) = \beta_a \left( - \frac{\partial E_{a}(k)}{\partial W_{a1}(k)} \right)$$

(41)
The Markov chain is
\[
H = \begin{pmatrix} 0.4 & 0.6 \\ 0.7 & 0.3 \end{pmatrix}
\] (43)

Depending on the Markov chain (43) and letting the weight vector be \( \omega = [0.2, 0.8]^T \), we adopt the proposed method in this paper to transform this two-mode jumping problem into a single-objective optimal control problem. Set the discount factor as \( \alpha = 0.95 \). According to (6), we obtain
\[
D_1 = 0.2*0.4 + 0.8*0.6 = 0.56
\]
and
\[
D_2 = 0.2*0.7 + 0.8*0.3 = 0.38
\] (45)

Therefore, the performance index function for MJS (42) can be calculated as
\[
J(x(k)) = 0.56J_1(x(k)) + 0.38J_2(x(k))
\] (46)
where \( J_i(x(k)), i \in \{1, 2\} \) can obtained from Eq. (3) with
\[
Q_i(x(k)) = 0.2x^T(k)x(k)
\] (47)
and
\[
R_i = 0.2
\] (48)

During the implementation process of the proposed ADP approach, the initial weights of both the critic and the action networks are chosen randomly in \([0, 1]\). Furthermore, set the learning rates of both networks to be \( \beta_a = \beta_c = 0.01 \), and the initial state of the system to be \([0.6, -0.6]^T\).

The proposed ADP approach is applied to stabilize this MJS. The stochastic jumping trajectory between mode 1 and mode 2 is presented in Fig. 2. It is clear that the system randomly jumps between these two modes. During this process, the control policy and the system states trajectories are shown in Figs. 3 and 4, respectively. We can see that the obtained control policy can make variables \( x_1 \) and \( x_2 \) converge to their equilibrium points, even though the system mode is jumping randomly. Moreover, when
the system reaches its stability (after 10 time steps), the system states do not change any more, even though the system mode is still jumping. The evolution of the performance index function at $k=1$ is presented in Fig. 5. We can observe that, during this process, the performance index function is monotonically non-decreasing to the optimal value just like the proof in Theorems 1–3. Therefore, the simulation results reveal that the proposed ADP method is effectiveness.

Next, consider the following discrete-time nonlinear MJS with four jumping modes:

$$
\begin{align*}
\text{mode 1} & : \quad \begin{cases} 
  x_1(k+1) = -x_1(k) + \sin(x_1(k)) + u(k) \\
  x_2(k+1) = -x_2(k) + \sin(x_2(k))
\end{cases} \\
\text{mode 2} & : \quad \begin{cases} 
  x_1(k+1) = -x_1(k) + \sin(x_1(k)) + u(k) \\
  x_2(k+1) = -1 + \cos(x_1(k))u(k)
\end{cases} \\
\text{mode 3} & : \quad \begin{cases} 
  x_1(k+1) = -\sin(0.5x_2(k))u(k) \\
  x_2(k+1) = -\sin(0.9x_1(k))
\end{cases} \\
\text{mode 4} & : \quad \begin{cases} 
  x_1(k+1) = -\sin(x_1(k))u(k) \\
  x_2(k+1) = 1 - \cos(x_1(k))
\end{cases}
\end{align*}
$$

(49)

Set the Markov chain as

$$
H = \begin{pmatrix}
0.1 & 0.6 & 0.2 & 0.1 \\
0.4 & 0.3 & 0.1 & 0.2 \\
0.2 & 0.2 & 0.5 & 0.1 \\
0.2 & 0.4 & 0.3 & 0.1
\end{pmatrix}
$$

(50)

and the weight vector as

$$
\omega = [0.3, 0.1, 0.1, 0.5]^T
$$

(51)

Assume that this system has unknown system functions. We use the proposed ADP approach to find the optimal control policy. According to (6), we obtain

$$
\begin{align*}
D_1 &= 0.3x_1 + 0.1x_2 + 0.6x_3 + 0.9x_4 + 0.5x_5 + 0.1 = 0.16 \\
D_2 &= 0.3x_1 + 0.1x_2 + 0.3x_3 + 0.5x_4 + 0.2 = 0.26 \\
D_3 &= 0.3x_1 + 0.2 + 0.1x_2 + 0.1x_3 + 0.5x_4 + 0.1 = 0.18 \\
D_4 &= 0.3x_1 + 0.2 + 0.1x_2 + 0.4 + 0.1x_3 + 0.3 + 0.5x_4 + 0.1 = 0.18
\end{align*}
$$

(52) (53) (54) (55)

With these parameters, the performance index function for the discrete-time nonlinear MJS with four jumping modes can be calculated as

$$
J(x(k)) = 0.16J_1(x(k)) + 0.26J_2(x(k)) + 0.18J_3(x(k)) + 0.18J_4(x(k))
$$

(56)

where $J_i(x(k)), i \in \{1, 2, 3, 4\}$ can obtained from Eq. (3) with $Q_i(x(k)) = 0.2x_i^T(k)x(k)$

(57)

and

$$
R_i = 0.2
$$

(58)

During the implementation process, the initial weights of the critic and the action networks are randomly chosen in $[0, 1]$. And set the learning rates of both networks as $\beta_c = \beta_a = 0.01$, the discount factor as $\alpha = 0.95$, and the initial value of the jumping system state as $[-0.6, 0.6]^T$. With the obtained performance index function (56), the proposed ADP approach is applied to stabilize the MJS (49). Simulation results are provided in Figs. 6–9.

In Fig. 6, we can clearly observe that the active mode of this MJS is jumping randomly among the above four modes. The trajectories of the control policy and the system states are presented in Figs. 7 and 8, respectively. We know even though the mode jumps

![Fig. 6. Jumping mode evolution among four jumping modes.](image-url)

Please cite this article as: X. Zhong et al., A neural network based online learning and control approach for Markov jump systems, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2014.01.060
randomly, the state variables $x_1$ and $x_2$ can converge to and stay at the equilibrium point.

The evolution of the performance index function sequence at $k=1$ is shown in Fig. 9. We can see that this sequence is monotonically non-decreasing and achieves its optimal value at about 100th iterative step. Hence, the results reveal that the proposed method in this paper is effective for discrete-time nonlinear MJS even with high modes.

Note that this paper uses two different numerical examples to demonstrate the effectiveness of the proposed ADP method for discrete-time nonlinear MJSs. For each example the evolution of the performance index function sequence of the whole jumping system at $k=1$ is presented. We can observe that the performance index function sequence is monotonically increasing at the first several iterative steps, then unchanged when it reaches the optimal value. The purpose of showing the evolution of the performance index function of iterative steps is to prove that the performance index function can achieve the optimal value at each time step.

7. Conclusion

This paper developed and analyzed an optimal control approach for a class of discrete-time nonlinear MJSs with unknown system functions based on ADP techniques. The MJSs control problem was converted into a single-objective optimal control problem. The detailed stability analysis was proposed, including the convergence of the performance index function sequence and the existence of the corresponding admissible control. Neural networks were applied for implementation. Online learning was used to tune the weights of the critic and the action networks. Two numerical examples were presented to validate the effectiveness of the proposed control approach.

Acknowledgments

This work was supported in part by the National Science Foundation under Grant ECCS 1053717, Army Research Office under Grant W911NF-12-1-0378, NSF-DFG Collaborative Research on “Autonomous Learning” (a supplement Grant to CNS 1117314). This work was also supported in part by the National Natural Science Foundation of China under Grants 51228701 and 61034005, and the IAPI Fundamental Research Funds under Grant 2013ZCX01-07.

References

Xiangnan Zhong received her B.S. and M.S. degrees in automation and control engineering from Northeastern University, Shenyang, China, in 2010 and 2012, respectively. She is currently a Ph.D. student in the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, RI, USA. Her research interests include adaptive dynamic programming, reinforcement learning, and neural network.

Haibo He received the B.S. and M.S. degrees in electrical engineering from Huazhong University of Science and Technology, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical engineering from Ohio University, in 2006. He is currently an Associate Professor at the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island. From 2006 to 2009, he was an Assistant Professor at the Department of Electrical and Computer Engineering, Stevens Institute of Technology. His current research interests include adaptive dynamic programming, computational intelligence, self-adaptive systems, machine learning, data mining, embedded intelligent system design (VLSI/FPGA), and various applications such as smart grid, cognitive radio, and sensor networks. He has published over 50 articles in international journals and conferences.

Huaxiang Zhang received the B.S. degree in control engineering from Northeastern University, Shenyang, China, in 1982 and 1985, respectively. He received the Ph.D. degree in thermal power engineering and automation from Southeast University, Nanjing, China, in 1991. He joined the Department of Automatic Control, Northeastern University, Shenyang, China, in 1992, as a Postdoctoral Fellow for two years. Since 1994, he has been a Professor and Head of the Institute of Electric Automation, School of Information Science and Engineering, Northeastern University, Shenyang, China. His main research interests are fuzzy control, stochastic system control, neural networks based control, nonlinear control, and their applications. He has authored and coauthored over 200 journal and conference papers, four monographs and the in-copyright 20 patents. He is the Chair of the Adaptive Dynamic Programming & Reinforcement Learning Technical Committee on IEEE Computational Intelligence Society, he is the Associate Editor of Automatica, IEEE Transactions on Fuzzy Systems, IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Cybernetics, IEEE Transactions on Industrial Informatics, and several other journals.

Zhanshan Wang received the M.S. degree in control theory and control engineering from Fushun Petroleum Institute (now Liaoning Shihua University), Fushun, China, in 2001. He received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2006. He is now a professor in Northeastern University. His research interests include stability analysis of recurrent neural networks, synchronization of complex networks, fault diagnosis, fault tolerant control and nonlinear control.

**Please cite this article as:** X. Zhong, et al., A neural network based online learning and control approach for Markov jump systems, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2014.01.060