



# Reinforcement learning and adaptive optimization of a class of Markov jump systems with completely unknown dynamic information

DOI:

[10.1007/s00521-019-04180-2](https://doi.org/10.1007/s00521-019-04180-2)

## Document Version

Accepted author manuscript

[Link to publication record in Manchester Research Explorer](#)

## Citation for published version (APA):

He, S., Zhang, M., Fang, H., Liu, F., Luan, X., & Ding, Z. (2019). Reinforcement learning and adaptive optimization of a class of Markov jump systems with completely unknown dynamic information. *Neural Computing and Applications*. <https://doi.org/10.1007/s00521-019-04180-2>

## Published in:

Neural Computing and Applications

## Citing this paper

Please note that where the full-text provided on Manchester Research Explorer is the Author Accepted Manuscript or Proof version this may differ from the final Published version. If citing, it is advised that you check and use the publisher's definitive version.

## General rights

Copyright and moral rights for the publications made accessible in the Research Explorer are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

## Takedown policy

If you believe that this document breaches copyright please refer to the University of Manchester's Takedown Procedures [<http://man.ac.uk/04Y6Bo>] or contact [openresearch@manchester.ac.uk](mailto:openresearch@manchester.ac.uk) providing relevant details, so we can investigate your claim.



# Reinforcement learning and adaptive optimization of a class of Markov jump systems with completely unknown dynamic information

Shuping He, Maoguang Zhang, Haiyang Fang, Fei Liu, Xiaoli Luan, Zhengtao Ding

**Abstract**—In this paper, an online adaptive optimal control problem of a class of continuous-time Markov jump linear systems (MJLSs) is investigated by using a parallel reinforcement learning (RL) algorithm with completely unknown dynamics. Before collecting and learning the subsystems information of states and inputs, the exploration noise is firstly added to describe the actual control input. Then, a novel parallel RL algorithm is used to parallelly compute the corresponding  $N$  coupled algebraic Riccati equations (AREs) by online learning. By this algorithm, we will not need to know the dynamic information of the MJLSs. The convergence of the proposed algorithm is also proved. Finally, the effectiveness and applicability of this novel algorithm is illustrated by two simulation examples.

**Index Terms**—Markov jump linear systems (MJLSs); adaptive optimal control; online; reinforcement learning (RL); coupled algebraic Riccati equations (AREs).

## I. INTRODUCTION

Markov jump linear systems (MJLSs), firstly proposed by Krasovskii and Lidskii [1] in 1961, can be considered as a kind of multi-model stochastic systems. In MJLSs, it contains two mechanisms, i.e., the modes and the states. The modes are jumping dynamics, modeled by finite-state Markov chains. The states are continuous or discrete, modeled by a set of differential or difference equations. With the development of control science and stochastic theory, MJLSs have been widely concerned and many research results are available, such as stochastic stability and stabilizability [2-4], controllability [5-9] and robust estimation and filtering [10, 12].

In recent years, the adaptive optimal control problem has become a focused issue in controllers design and many related

works have been published. For example, the authors in [13] studied the adaptive surface optimal control methods for strict-feedback systems. Then the observer-based adaptive fuzzy control law was proposed for nonlinear nonstrict-feedback systems [14]. A general method to solve the optimal control problem related to the related algebraic Riccati equations (AREs) was studied in [15]. In [16], Kleinman considered an offline iterative technique to solve the AREs. In [17, 18], the iterative algorithm and the temporal difference method were respectively proposed to solve the discrete-time coupled AREs. To get the solutions of a set of coupled AREs for MJLSs, a Lyapunov iteration method [19] was derived. For more related researches, we can refer to [20-24]. However, all these algorithms are offline and have limitations to implement when the system dynamics are unknown.

Reinforcement learning (RL) and approximate/adaptive dynamic programming (ADP) algorithms are widely applied to solve the optimal control problems [25-28]. Moreover, we can use RL and ADP algorithm to obtain the optimal controller solutions through online learning. In [29], an online feedback control scheme is designed for linear systems by applying RL and ADP method. Then, it was developed to solve the zero-sum differential game [30]. In [31] and [32], an adaptive policy iteration algorithm is considered for continuous-time and discrete-time nonlinear systems respectively. Then the reinforcement  $Q$ -learning for optimal tracking control problem was studied for linear discrete-time systems with unknown dynamics [33] and nonlinear discrete-time systems with unknown nonaffine dead-zone input [34]. Applying the  $H_\infty$  state feedback scheme, an online policy update algorithm was proposed [35]. For more results on this studying, we refer readers to [36-44].

It is worth pointing out the relevant methods related to RL and ADP are mostly used to solve the optimal controller design problems for general linear/nonlinear dynamics. Due to the complexity of MJLSs which contain states and modes, the mode-based switching subsystems have coupling relationships. Obviously, the RL and ADP algorithms proposed above can not directly solve the adaptive optimal control problems of MJLSs. To study the online control problem for MJLSs, the authors in [45] considered an online adaptive optimal control scheme by applying a novel policy iteration algorithm. Then, the relevant results were developed to the Itô stochastic systems [46] and the  $H_\infty$  optimal problem [47] of MJLSs. But in these results, we must partially know the system dynamics. Therefore, these published results can be considered as partial model-free ones.

This work was supported in part by the National Natural Science Foundation of China under Grant 61673001, 61722306, the Foundation for Distinguished Young Scholars of Anhui Province under Grant 1608085J05, the Key Support Program of University Outstanding Youth Talent of Anhui Province under Grant gxydZD2017001, the State Key Program of National Natural Science Foundation of China under Grant 61833007 and the 111 Project under Grant B12018.

S. He, M. Zhang and H. Fang are with School of Electrical Engineering and Automation, Anhui University, Hefei 230601, China. S. He is also with Institute of Physical Science and Information Technology, Anhui University, Hefei, 230601, China (e-mail: shuping.he@ahu.edu.cn, fangocean1996@gmail.com).

F. Liu and X. Luan are with the Key Laboratory of Advanced Process Control for Light Industry (Ministry of Education), Institute of Automation, Jiangnan University, Wuxi 214122, China. E-mail: fliu@jiangnan.edu.cn, xiaoli\_luan@126.com.

Z. Ding is School of Electrical and Electronic Engineering, The University of Manchester, Manchester M13 9PL, UK. E-mail: zhengtao.ding@manchester.ac.uk.

In this paper, we propose a novel parallel RL algorithm to solve the optimal control problem of MJLSs with completely unknown dynamics. This algorithm is based on the subsystem transformation technique and the parallel RL algorithm. The subsystem transformation technique is used to obtain the  $N$  AREs. Then we can parallelly solve the corresponding  $N$  AREs by collecting and learning the subsystems information of states and inputs. Comparing with the published results in [45] and [46], we will not use the system dynamics in this paper and the exploration noise is added to describe the actual control input. Moreover, the solutions can be conducted directly in one step by repeated iteration and computation without using policy evaluation and improvement. It is clear that the iterative algorithm in this paper can be effective to study the online optimal control problems for MJLSs. The convergence has also been proved. In order to demonstrate the effectiveness of the designed method, two simulation examples are given at last.

The main contributions of this paper can be concluded as:

- 1) A novel parallel RL algorithm with completely unknown dynamic information is first proposed to solve the optimal control problem of MJLSs;
- 2) To deal with the jumping modes, we use the subsystem transformation technique to decompose the MJLSs.
- 3) Compared with the results in [19], we apply the online collecting and learning the subsystems information of states and inputs to obtain the dynamics information. Then we optimize and update the iterative results to obtain the optimal control policy. The designed PI algorithm is online and does not require the dynamic information of the MJLSs.

This paper contains the following five sections. The problem formulation and preliminaries are introduced in Section II. The novel parallel iteration algorithm is proposed in Subsection III-A and the online implementation of the algorithm is given in Subsection III-B; Subsection III-C gives the convergence proof of the designed algorithm. Two simulation examples are shown in Section IV. A conclusion is given in Section V.

**Notations:** Throughout this paper,  $\mathfrak{R}^n$  is used to denote an  $n$ -dimensional real matrix;  $\lambda(A)$  denotes the eigenvalue of  $A$ ;  $\text{rank}(A)$  denotes the rank of  $A$ ;  $Z_+$  denotes a non-negative positive integer set;  $\mathbf{E}\{\cdot\}$  shows the mathematical expectation of stochastic processes or vectors;  $\|\cdot\|$  represents the Euclidean norm;  $\otimes$  is used to represent the Kronecker product which has the following properties: (1)  $(A \otimes B)^\top = A^\top \otimes B^\top$ , (2)  $x^\top P x = (x^\top \otimes x^\top) \text{vec}(P)$ . For a matrix  $A \in \mathfrak{R}^{n \times n}$ , we denote  $\text{vec}(A) = [a_1^\top, a_2^\top, \dots, a_n^\top]^\top$ , where  $a_i \in \mathfrak{R}^n$  is the  $i$ th column of  $A$ .

## II. BACKGROUNDS AND PRELIMINARIES

### A. Problem Description

Consider a class of continuous-time Markov jump linear systems (MJLSs) described by:

$$\begin{cases} \dot{x}(t) = A(r(t))x(t) + B(r(t))u(t) \\ x(0) = x_0, r(0) = r_0, t = 0 \end{cases} \quad (1)$$

with the infinite horizon performance index:

$$J(x(t)) = \mathbf{E} \left\{ \int_0^\infty [x^\top(t)Q_i x(t) + u^\top(t)R_i u(t)] dt \mid t=0, x_0, r_0 \right\} \quad (2)$$

where  $x(t) \in \mathfrak{R}^n$  is the system state with the initial value  $x_0$ ;  $u(t) \in \mathfrak{R}^m$  is the control input;  $A(r(t))$ ,  $B(r(t))$  are the coefficient matrices,  $Q(r(t))$  and  $R(r(t))$  are the weight matrices which are mode-dependent and positive-definite; the pair  $(A(r(t)), Q^{1/2}(r(t)))$  is supposed to be observable;  $r(t)$  is a continuous-time and discrete-state right continuous Markov stochastic process which represents the system mode and takes on values in a discrete set  $M = \{1, 2, \dots, N\}$ ;  $r_0$  represents the initial mode. Here, MJLSs (1) is assumed to be stochastically stabilizable.

For MJLSs (1), the transition probabilities of each mode satisfy:

$$P_r \{r(t + \Delta t) = j \mid r(t) = i\} = \begin{cases} \pi_{ij} \Delta t + o(\Delta t) & j \neq i \\ 1 + \pi_{ii} \Delta t + o(\Delta t) & j = i \end{cases} \quad (3)$$

where  $i, j \in M$ ,  $\Delta t > 0$  and  $\lim_{\Delta t \rightarrow 0} \frac{o(\Delta t)}{\Delta t} = 0$ ;  $\pi_{ij} \geq 0$  ( $i \neq j$ ) represents the transition rate from mode  $i$  to mode  $j$  at time  $t \rightarrow t + \Delta t$  and has  $\pi_{ii} = -\sum_{i \neq j} \pi_{ij}$ . For simplicity, when  $r(t) = i$ ,  $A(r(t))$ ,  $B(r(t))$ ,  $Q(r(t))$ ,  $R(r(t))$  can be labeled as  $A_i$ ,  $B_i$ ,  $Q_i$ ,  $R_i$  respectively.

For a given feedback control policy  $u(t) = -K_i x(t)$ , we define the following cost function as:

$$V(x(t)) = \mathbf{E} \left\{ \int_t^\infty [x^\top(t)Q_i x(t) + u^\top(t)R_i u(t)] dt \mid t, x(t), r(t) \right\} \quad (4)$$

The optimal control problem for MJLSs (1) is to find a mode-dependent admissible feedback control policy, which minimizes the performance index, i.e.

$$\begin{aligned} V^*(x(t)) &= \min_{u(t)} J(x(t)) \\ &= \min_{u(t)} \mathbf{E} \left\{ \int_0^\infty [x^\top(t)Q_i x(t) + u^\top(t)R_i u(t)] dt \mid t=0, x_0, r_0 \right\} \end{aligned} \quad (5)$$

Then the cost function (4) can be denoted as  $V(x(t)) = x_{i,t}^\top P_i x_{i,t}$ , where  $x_{i,t}$  is the state of the  $i$ th jump mode;  $P_i$  is a positive-definite and mode-dependent matrix which can be obtained through solving the following coupled algebraic Riccati equations (AREs):

$$\hat{A}_i^\top P_i + P_i \hat{A}_i + Q_i + \sum_{j=1, j \neq i}^N \pi_{ij} P_j - P_i B_i R_i^{-1} B_i^\top P_i = 0 \quad (6)$$

where  $\hat{A}_i = A_i + \frac{\pi_{ii}}{2} I$ . Then, the optimal feedback control policy can be determined by:

$$u(t) = -K_i x(t) = -R_i^{-1} B_i^\top P_i x(t). \quad (7)$$

**Definition 1:** MJLSs (1) is stochastically stable (SS), if for any initial conditions  $x_0$  and  $r_0$ , the following formula is satisfied:

$$\lim_{t \rightarrow \infty} \mathbf{E} \|x(t)\|^2 = 0. \quad (8)$$

Inspired by [48], according to the subsystem transformation technique, we can re-write MJLSs (1) as the following  $N$  re-constructed subsystems:

$$\begin{cases} \dot{x}_{i,t} = \hat{A}_i x_{i,t} + B_i u_{i,t} \\ x_i(0) = x_{i,0}, t_0 = 0 \end{cases} \quad (9)$$

In order to guarantee the stabilizability of the  $N$  re-constructed subsystems in equation (9) by the optimal feedback control policy (7), the triples  $(A_i, B_i, \sqrt{Q_i})$  are assumed to be stabilizable-detectable (Gajic & Borno [19]).

### B. An offline parallel policy iteration algorithm (OPPIA)

According to the assumptions mentioned above, the exact solution  $P_i^*$  of the positive-definite and mode-dependent matrix  $P_i$  can be obtained by solving AREs (6). Notice that AREs (6) is a set of nonlinear algebraic equations and it is difficult to be solved directly, especially for large dimensional matrices. To tackle this problem, some algorithms have been proposed by Kleinman [16] and Gajic & Borno [19]. For MJLSs (1), we give the following Lemma 1.

**Lemma 1.** (Gajic & Borno [19]). For each subsystems, we give an initial stabilizable value  $K_{i(0)}$ . The solutions of the following decoupled algebraic Lyapunov equations are equivalent to the solutions of AREs (6).

$$\begin{aligned} (\hat{A}_i - B_i K_{i(k)})^T P_{i(k+1)} + P_{i(k+1)} (\hat{A}_i - B_i K_{i(k)}) \\ = -Q_{i(k)} - P_{i(k)} B_i R_i^{-1} B_i^T P_{i(k)} \end{aligned} \quad (10)$$

In order to solve the  $N$  parallel decoupled algebraic Lyapunov equations in (10), we give the following Algorithm 1.

---

#### Algorithm 1. OPPIA

---

**Step 1.** Subsystem Transformation. Re-construct MJLSs (1) into the  $N$  independent subsystems in equation (9).

**Step 2.** Give  $N$  initial stabilizable values, i.e., when  $k=0$ , let  $\{P_{1(0)}, \dots, P_{N(0)}\}$ ,  $\lambda(\hat{A}_i - B_i K_{i(k)}) < 0$ .

**Step 3. Parallel Policy Iteration:**

Solve the  $N$  parallel decoupled algebraic Lyapunov equations in (10).

**Parallel Policy Update:**

Derive  $Q_{i(k)}$  and  $K_{i(k)}$  by the following recursions:

$$\begin{cases} K_{i(k)} = R_i^{-1} B_i^T P_{i(k-1)} \\ Q_{i(k)} = Q_i + \sum_{j=1, j \neq i}^N \pi_{ij} P_{j(k-1)} \geq 0 \end{cases} \quad (11)$$

**Step 4.** Compute  $P_{i(k)}$  and  $K_{i(k)}$  by solving step 3, until

$\|P_{(k+1)} - P_{(k)}\| \leq \varepsilon$  for  $k \geq 1$ ; Otherwise, set  $k = k + 1$ , go to Step 2 and continue the iteration process.

**Step 5.** Obtain the optimal control policy  $u_{i,t} = -K_{i(k)} x_{i,t}$ .

---

**Remark 1:** Due to the fact that MJLSs (1) can be decomposed into  $N$  independent subsystems, the solutions can be obtained by parallel solving  $N$  decoupled algebraic Lyapunov linear equations. Thus the solutions to the  $N$  parallel nonlinear AREs (6) is numerically approximated. By Algorithm 1, the exact solutions of  $P_i^*$  can be obtained while  $k \rightarrow \infty$ . Notice that the method in Algorithm 1 is offline and the knowledge of the dynamics  $A_i, B_i$  are all needed.

## III. MAIN RESULTS

In this part, we will propose a parallel reinforcement learning (RL) algorithm to design the feedback control policy for MJLSs (1). This is a novel online learning strategy which will not require the knowledge of the coefficient matrices, i.e.,  $A_i$  and  $B_i$ . For each solution step, we can solve  $N$  parallel decoupled algebraic Lyapunov equations in (10). In the end, the convergence of this new algorithm is also proved.

### A. A parallel RL algorithm with completely unknown dynamics

For the offline Algorithm 1, the  $N$  subsystems dynamics are required to solve the decoupled algebraic Lyapunov equations in (10). For practical applications, this algorithm will be difficult to achieve if we do not know the exact knowledge in advance. To overcome these weakness, a new online parallel RL algorithm is developed to study the optimal solutions of MJLSs with completely unknown dynamics. For the  $i$ th subsystem, the unique symmetric positive-definite solutions of  $P_{i(k)}$  and the state feedback gains  $K_{i(k)}$  ( $\forall k \in \mathbb{Z}_+$ ) should satisfy equations (10)-(11).

Recalling to equation (9), we rewrite the system state equation as:

$$\begin{cases} \dot{x}_{i,t} = \bar{A}_i x_{i,t} + B_i (u_{i,t} + K_{i(k)} x_{i,t}) \\ x_i(0) = x_{i,0}, t_0 = 0 \end{cases} \quad (12)$$

where  $\bar{A}_i = \hat{A}_i - B_i K_{i(k)}$  and the pairs  $(\hat{A}_i, B_i)$  are stabilizable.

Then, the novel algorithm is given as follows:

---

#### Algorithm 2 A parallel RL algorithm with completely unknown dynamics.

---

**Step 1.** Subsystem Transformation. Re-construct MJLSs (1) into the  $N$  independent subsystems in equation (12).

**Step 2.** Give a initial stabilizable value  $K_{i(0)}$  and employ  $u_{i,t} = -K_{i(0)} x_{i,t} + e_i$  as the control input in time interval  $[t_0, t_f]$ .

**Step 3.** Solve equations (10)-(11) by using the online parallel RL algorithm:

$$\begin{aligned} x_{i,t+T}^T P_{i(k)} x_{i,t+T} - x_{i,t}^T P_{i(k)} x_{i,t} = - \int_t^{t+T} x_{i,\tau}^T \bar{Q}_{i(k)} x_{i,\tau} d\tau \\ + 2 \int_t^{t+T} [(u_{i,\tau} + K_{i(k)} x_{i,\tau})^T R_i K_{i(k+1)} x_{i,\tau}] d\tau \end{aligned} \quad (13)$$

where  $\bar{Q}_{i(k)} = Q_i + \sum_{j=1, j \neq i}^N \pi_{ij} P_{j(k-1)} + K_{i(k)}^T R_i K_{i(k)}$ .

**Step 4.** Compute  $P_{i(k)}$  and  $K_{i(k)}$  in equation (13), until  $\|P_{(k+1)} - P_{(k)}\| \leq \varepsilon$  for  $k \geq 1$ ; Otherwise, set  $k = k + 1$ , go to Step 2 and continue the iteration process.

**Step 5.** Obtain the optimal control policy  $u_{i,t} = -K_{i(k)} x_{i,t}$ .

---

Notice that the  $N$  subsystems in equation (9) are reconstructed into  $N$  new continuous-time subsystems in equation (12). Next, we will give an online implementation of the parallel RL algorithm to solve equations (13). In equation (12),  $A_i$  and  $\pi_{ii}$  are embedded in the reconstructed

subsystem states;  $B_i$  is embeded in the reconstructed subsystem control inputs, which can be measured online. For real-time learning the information of  $x_{i,t}$  and  $u_{i,t}$ , we can choose  $u_{i,t} = -K_{i(k)}x_{i,t} + e_i$  as a true control input, where  $e_i$  is an additional exploration noise.

**Remark 2.** Solving equation (13), we can get a series of solutions, where  $P_{i(k)}$ ,  $K_{i(k)}$  satisfies (10)-(11),  $x_{i,t}$  is the solution of equation (12) with the arbitrary control input  $u_{i,t}$ .

This fact enables us to choose  $u_{i,t} = -K_{i(k)}x_{i,t} + e_i$  as a real input signal for learning. Obviously, this operation will not affect the convergence of the learning process.

**Remark 3.** In Algorithm 2, we choose a exploration noise and assume that it is a nonzero measurable bounded signal. In fact, the choice of the exploration noise will not affect the algorithm. We add the exploration noise as a excitation signal to satisfy the persistence of the excitation conditions. For convenience, we always choose the exponentially decreasing probing noise and the sum of sinusoidal signals as the exploration noise.

**Remark 4.** In equation (13), the subsystems internal dynamics information is not needed, i.e., the dynamic matrices of  $A_i$ ,  $B_i$  are all not included. In order to solve the equation (13) by the online RL algorithm, we should continuously sample the states and the inputs information online from the  $N$  subsystems at each iteration step. In fact, the dynamics information is already included in the sample information. Therefore, the online Algorithm 2 did not required the dynamics information, i.e., Algorithm 2 as a completely model-free algorithm.

### B. Online implementation of the novel parallel RL algorithm

To find a set of pairs  $(P_{i(k)}, K_{i(k+1)})$  which satisfy equations (10)-(11), the novel online parallel RL implement approach is derived. It observes that the dynamics matrices  $A_i$ ,  $B_i$  and the transfer matrix element  $\pi_{ii}$  are not included in Algorithm 2. According to the Kronecker product representation, the left side of equation (13) can be rewritten as:

$$\begin{aligned} & x_{i,t+T}^T P_{i(k)} x_{i,t+T} - x_{i,t}^T P_{i(k)} x_{i,t} \\ &= \bar{P}_{i(k)}^T \left[ x_{i,t+T}^T \otimes x_{i,t+T}^T - x_{i,t}^T \otimes x_{i,t}^T \right] = \bar{P}_{i(k)}^T \delta_{ix} \end{aligned} \quad (14)$$

where  $\bar{P}_{i(k)}$  is a set of positive-definite and mode-dependent matrices;  $\bar{P}_i$  is a column vector-valued matrix whose internal elements are produced by  $P_i$  and can be given as:

$$\bar{P}_i = [p_{i11}, 2p_{i12}, \dots, 2p_{i1n}, p_{i22}, 2p_{i23}, \dots, p_{im}]^T, \quad (15)$$

$$x_i^T \otimes x_i^T = [x_{i1}^2, x_{i1}x_{i2}, \dots, x_{i1}x_{in}, x_{i2}^2, x_{i2}x_{i3}, \dots, x_{i2}x_{in}, \dots, x_{in}^2]^T. \quad (16)$$

Using the similar method as equation (14), we have:

$$x_{i,t}^T \bar{Q}_{i(k)} x_{i,t} = (x_{i,t}^T \otimes x_{i,t}^T) \text{vec}(\bar{Q}_{i(k)}) \quad (17)$$

where  $\bar{Q}_{i(k)} = Q_{i(k)} + K_{i(k)}^T R_i K_{i(k)}$ . Then, the right side of equation (13) can be rewritten as:

$$\begin{aligned} & (u_{i,t} + K_{i(k)}x_{i,t})^T R_i K_{i(k+1)} x_{i,t} \\ &= [(x_{i,t}^T \otimes x_{i,t}^T)(I_n \otimes K_{i(k)}^T R_i) + (x_{i,t}^T \otimes u_{i,t}^T)(I_n \otimes R_i)] \text{vec}(K_{i(k+1)}) \end{aligned} \quad (18)$$

Meanwhile, we define the following operators:

$$I_{ixx} = \begin{bmatrix} \int_{t_0}^{t_1} x_{i,\tau}^T \otimes x_{i,\tau}^T d\tau \\ \int_{t_1}^{t_2} x_{i,\tau}^T \otimes x_{i,\tau}^T d\tau \\ \vdots \\ \int_{t_{l-1}}^{t_l} x_{i,\tau}^T \otimes x_{i,\tau}^T d\tau \end{bmatrix}, \quad (19)$$

$$I_{ixu} = \begin{bmatrix} \int_{t_0}^{t_1} x_{i,\tau}^T \otimes u_{i,\tau}^T d\tau \\ \int_{t_1}^{t_2} x_{i,\tau}^T \otimes u_{i,\tau}^T d\tau \\ \vdots \\ \int_{t_{l-1}}^{t_l} x_{i,\tau}^T \otimes u_{i,\tau}^T d\tau \end{bmatrix}. \quad (20)$$

Thus, equation (13) can be described as:

$$\Theta_{i(k)} \begin{bmatrix} \bar{P}_{i(k)} \\ \text{vec}(K_{i(k+1)}) \end{bmatrix} = \Xi_{i(k)} \quad (21)$$

where  $\Theta_{i(k)} \in \mathfrak{R}^{l \times \frac{1}{2}n(n+1)+nm}$ ,  $\Xi_{i(k)} \in \mathfrak{R}^l$  and they can be described as:

$$\Theta_{i(k)} = \begin{bmatrix} \delta_{ixx} & -2[I_{ixx}(I_n \otimes K_{i(k)}^T R_i) + I_{ixu}(I_n \otimes R_i)] \end{bmatrix}, \quad (22)$$

$$\Xi_{i(k)} = -I_{ixx} \text{vec}(\bar{Q}_{i(k)}). \quad (23)$$

For equation (21), if  $\Theta_{i(k)}$  has full column rank, the least-squares solutions of equation (13) can be solved by:

$$\begin{bmatrix} \bar{P}_{i(k)} \\ \text{vec}(K_{i(k+1)}) \end{bmatrix} = (\Theta_{i(k)}^T \Theta_{i(k)})^{-1} \Theta_{i(k)}^T \Xi_{i(k)} \quad (24)$$

By formula (24), we can obtain  $P_{i(k)}$  and  $K_{i(k+1)}$ . To compute the matrices of  $I_{ixx}$  and  $I_{ixu}$ , we can use  $\frac{n(n+1)}{2} + nm$  integrators to collect the information of the states and the control inputs of each subsystems. In order to guarantee that Algorithm 2 can be online implemented at each step,  $\Theta_{i(k)}$  is required to have full column rank for all  $k \in Z_+$ , i.e., there is an integer  $l_0 > 0$ , for  $\forall l > l_0$ , the following condition holds:

$$\text{rank}([I_{ixx}, I_{ixu}]) = \frac{n(n+1)}{2} + nm. \quad (25)$$

### C. The convergence proof

In order to prove the convergence of Algorithm 2, we establish the following Theorem 1.

**Theorem 1.** For the novel parallel RL Algorithm 2, we give a stabilizable value  $K_{i(0)}$ , where  $\forall i \in M$  and assume that  $\Theta_{i(k)}$  has the full column rank; then the solutions of equation (13) satisfy equations (10)-(11).

**Proof:** Give a stabilizable value  $K_{i(0)}$ , i.e.,  $A_i - B_i K_{i(0)}$  is a Hurwitz. According to Algorithm 1, the exact solutions  $P_i^* = \lim_{k \rightarrow \infty} P_{i(k)}$  can be obtained by the decoupled algebraic Lyapunov equations in (10), and the feedback gain

$K_i^* = \lim_{k \rightarrow \infty} K_{i(k)}$  can be uniquely determined by equation (11).

For  $\forall t > 0$ , we choose the cost function as  $V(x(t)) = x_{i,t}^T P_i x_{i,t}$ .

Then, for the new re-constructed  $i$ th  $N$  independent subsystems in equation (12) and along the solutions of (12) by (10) and (11), the following is satisfied:

$$\begin{aligned} & x_{i,t+T}^T P_{i(k)} x_{i,t+T} - x_{i,t}^T P_{i(k)} x_{i,t} \\ &= \int_t^{t+T} \left[ x_{i,\tau}^T (\bar{A}_i^T P_{i(k)} + P_{i(k)} \bar{A}_i) x_{i,\tau} + 2(u_{i,\tau} + K_{i(k)} x_{i,\tau})^T B_i^T P_{i(k)} \right] d\tau \quad (26) \\ &= - \int_t^{t+T} x_{i,\tau}^T \bar{Q}_{i(k)} x_{i,\tau} d\tau + \int_t^{t+T} 2(u_{i,\tau} + K_{i(k)} x_{i,\tau})^T R_i K_{i(k+1)} x_{i,\tau} d\tau \end{aligned}$$

Since  $\Theta_{i(k)}$  has full column rank, the least-squares solutions in equation (24) is an online implement of equation (13). Therefore, the solutions of equation (13) is equivalent to the iteration solutions of equations (10) and (11). The proof is completed.

The flowchart of Algorithm 2 is given in Fig. 1.

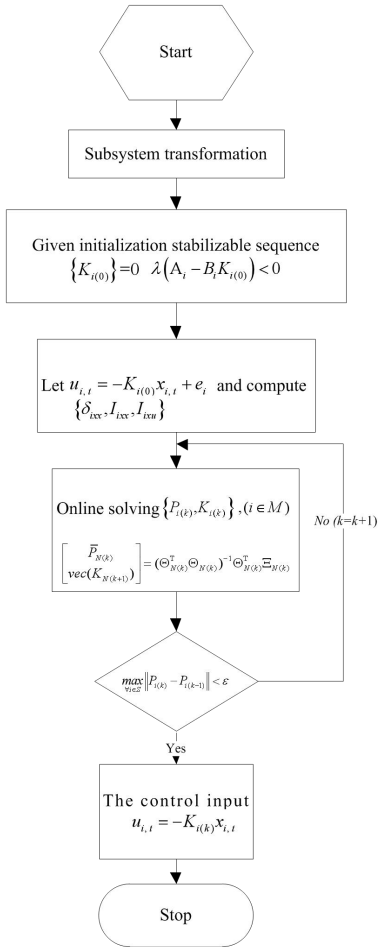


Fig. 1. The Flowchart of Algorithm 2

**Remark 5.** By the online implementation in Algorithm 2, we can see the differences between the offline Algorithm 1 and the online Algorithm 2. In Algorithm 1 we apply the offline iterative algorithm to solve the Riccati functions and equations directly, while in Algorithm 2, we use the online sampling and learning method. Moreover, the equations (10)-(11) in Algorithm 1 depend on the dynamics information. But in

Algorithm 2, we do not need any dynamics information by equation (13).

#### IV. Simulation Examples

**Example 1:** To show the the applicability of Algorithm 2, we consider a fourth order jump linear model with two jump modes as:

$$N = 2, \quad \Pi = \begin{bmatrix} -3 & 3 \\ 2.5 & -2.5 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} -2.2361 & -1.1358 & 1 & 0.6324 \\ -0.1024 & -3 & 0.3835 & 0.85 \\ 0.7112 & 11.2346 & -36.8199 & 4 \\ 1.0692 & 13.4230 & 20.1185 & -12.1801 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} -1.5326 & -1.2436 & 0.5458 & 0.7136 \\ -0.8 & -2.9346 & 0.0920 & 0.42 \\ 11.1634 & 23 & -26.4655 & -1.8347 \\ 25 & 8.3132 & -3.8714 & -31.4631 \end{bmatrix},$$

$$B_1 = B_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad Q_1 = Q_2 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

In the simulation, we set  $R_1 = R_2 = 1$ , the error value  $\epsilon$  as  $10^{-15}$  and the initial stabilizing feedback gain as  $K_{i(0)} = 0$ . According to the OPPIA in Algorithm 1, AREs (6) can be directly solved. After the 14th iterations, the exactly solutions are obtained:

$$P_1^* = \begin{bmatrix} 0.3505 & -0.0320 & 0.0180 & 0.0191 \\ -0.0320 & 0.5943 & 0.0572 & 0.0845 \\ 0.0180 & 0.0572 & 0.0268 & 0.0232 \\ 0.0191 & 0.0845 & 0.0232 & 0.0503 \end{bmatrix},$$

$$P_2^* = \begin{bmatrix} 0.5574 & -0.0483 & 0.0140 & 0.0227 \\ -0.0483 & 0.4057 & 0.0164 & 0.0111 \\ 0.0140 & 0.0164 & 0.0196 & -0.0003 \\ 0.0227 & 0.0111 & -0.0003 & 0.0178 \end{bmatrix}.$$

The OPPIA in Algorithm 1 is offline and the subsystems dynamics matrices are all needed. To show the effectiveness and applicability of Algorithm 2, we use the same conditions mentioned above. For the two subsystems, we select the initial states as  $x_{10} = x_{20} = [0 \quad -0.1 \quad 0.1 \quad 0]^T$  and the controller  $u_{i(k)} = -K_{i(k)} x_i + e_i$  in the time interval  $[0, 2]$  with the exploration noises as  $e_1 = e_2 = 100 \sum_{n=1}^{100} \sin(\omega_n t)$ , where  $\omega_n$  is a number and can be randomly selected in  $[-500, 500]$ . For the two subsystems, the information of the states and inputs is sampled every 0.01s, and the optimal controller can be updated every 0.1s. By the given iteration criterion, we can get

the final calculation results of Algorithm 2 in step 4, after 7th iterations as:

$$P_{1(7)} = \begin{bmatrix} 0.3505 & -0.0320 & 0.0180 & 0.0191 \\ -0.0320 & 0.5943 & 0.0572 & 0.0845 \\ 0.0180 & 0.0572 & 0.0268 & 0.0232 \\ 0.0191 & 0.0845 & 0.0232 & 0.0503 \end{bmatrix},$$

$$P_{2(7)} = \begin{bmatrix} 0.5574 & -0.0483 & 0.0140 & 0.0227 \\ -0.0483 & 0.4057 & 0.0164 & 0.0111 \\ 0.0140 & 0.0164 & 0.0196 & -0.0003 \\ 0.0227 & 0.0111 & -0.0003 & 0.0178 \end{bmatrix}.$$

The optimal state feedback controller gains are:

$$K_{1(7)} = [0.3557 \quad 0.7040 \quad 0.1253 \quad 0.1771],$$

$$K_{2(7)} = [0.5458 \quad 0.3849 \quad 0.0497 \quad 0.0513].$$

Then, we get the simulation results by using MATLAB. The following Figs. 2-5 show the convergence of Algorithm 2. The convergence of  $P_1$  and  $P_2$  to the optimal values are shown in Figs. 2-3. The elements of matrices  $P_1$  and  $P_2$  are updated which are shown in Figs. 4-5.

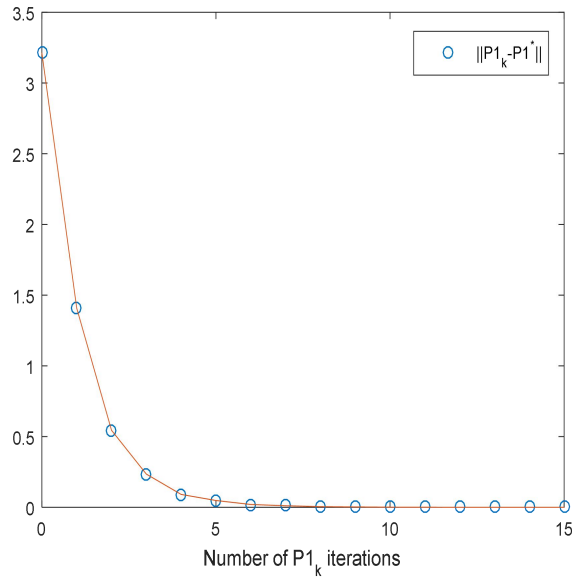


Fig. 2. The convergence of  $P_{1(k)}$  to the optimal value  $P_1^*$

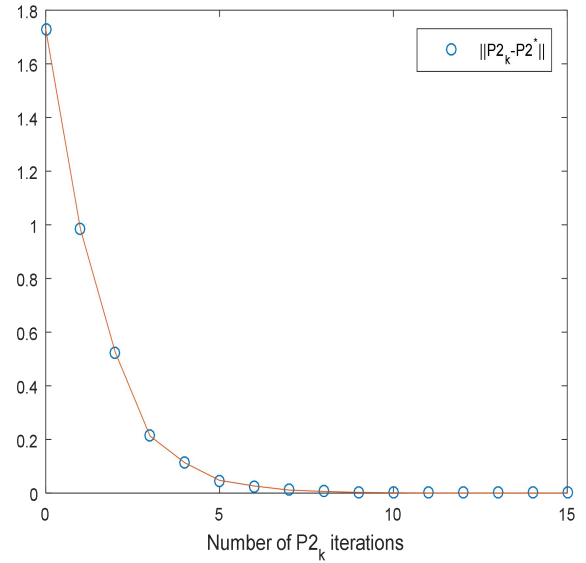


Fig. 3. The convergence of  $P_{2(k)}$  to the optimal value  $P_2^*$

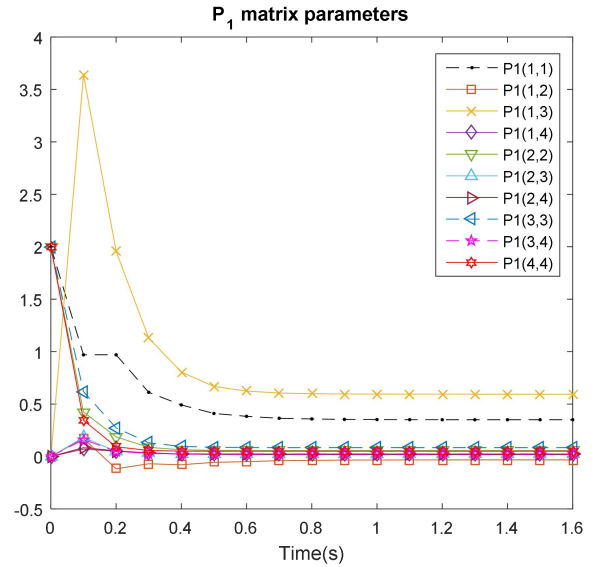


Fig. 4. The updated elements of  $P_{1(k)}$  at each step in Algorithm 2



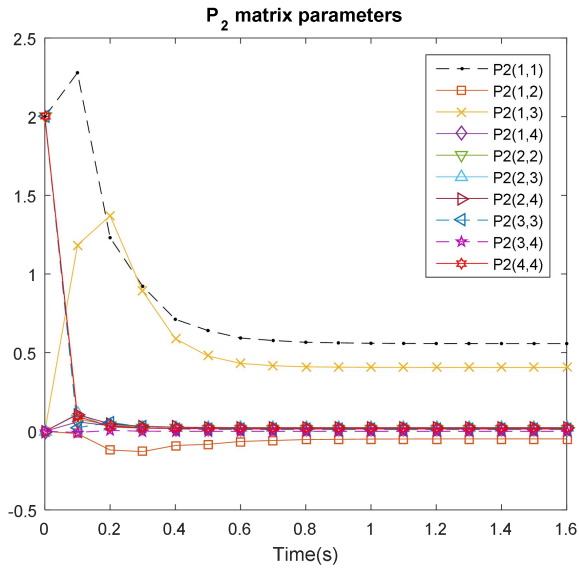


Fig. 5. The updated elements of  $P_{2(k)}$  at each step in Algorithm 2

**Example 2:** In order to further study the proposed approach, we give the following comparative results. Applying the parallel RL algorithm proposed in this paper and simulating the experiment with the same parameters as [45], we can get the final solutions after 7 iterations:

$$P_{1(7)} = \begin{bmatrix} 0.2408 & 0.0705 & 0.0393 & 0.0182 \\ 0.0705 & 0.0308 & 0.0085 & 0.0064 \\ 0.0393 & 0.0085 & 0.0157 & 0.0025 \\ 0.0182 & 0.0064 & 0.0025 & 0.0016 \end{bmatrix},$$

$$P_{2(7)} = \begin{bmatrix} 0.5026 & 0.1343 & 0.0518 & 0.0097 \\ 0.1343 & 0.0485 & 0.0138 & 0.0026 \\ 0.0518 & 0.0138 & 0.0193 & 0.0002 \\ 0.0097 & 0.0026 & 0.0002 & 0.0003 \end{bmatrix}.$$

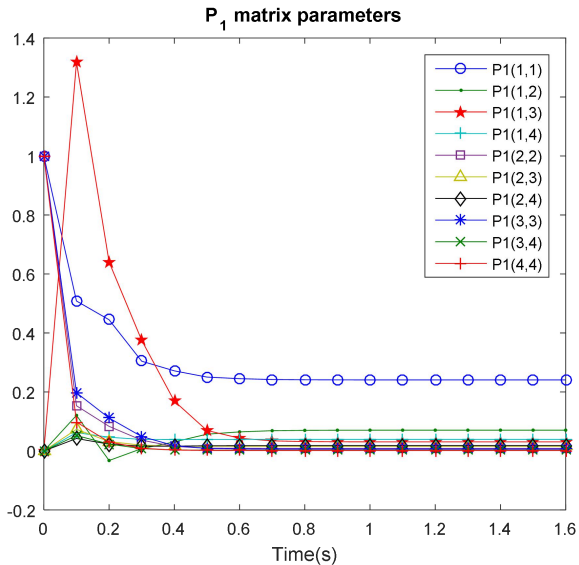


Fig. 6. The updated elements of  $P_{1(k)}$  using the parallel RL algorithm

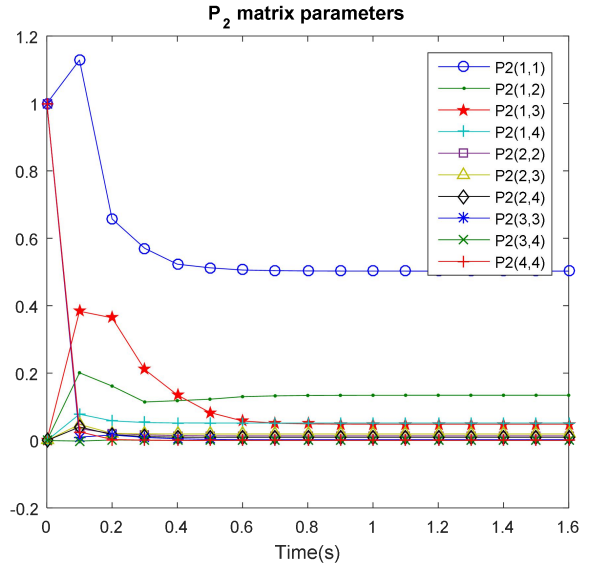


Fig. 7. The updated elements of  $P_{2(k)}$  using the parallel RL algorithm

Comparing with the online algorithms proposed in [45], the simulation errors between the calculated results proposed in this paper can nearly converge to zero. It shows that the parallel RL algorithm in this paper gives a more accurate solutions. It is also seen from Fig. 6 and Fig. 7 that the convergence rate of the parallel RL algorithm is also faster than the results in [45].

**Remark 6.** In the two simulation examples, we solve the positive-definite solution  $P_{i(k)}$  by the online reinforcement learning method. First, we obtain  $N$  coupled Riccati equations by decomposing the subsystems. According to the online implementation process, we need to collect the states and the inputs of each subsystems, and satisfy the condition of the full rank of the least square equation. Then we iteratively solve the least square formula, until the final optimal solution satisfying the limited precision requirement is obtained.

## V. CONCLUSION

A novel online parallel RL algorithm is proposed with completely unknown dynamics. It can solve the AREs by collecting the information of the states and the inputs of each of the subsystems. Therefore, this novel parallel RL algorithm can parallelly compute the corresponding  $N$  coupled AREs with completely unknown dynamics. The convergence of the parallel RL algorithm is also proved. Two simulation results illustrate the applicability and effectiveness of the designed algorithm. In the future, the RL algorithm and the relevant optimal algorithm can be used in other aspects, such as artificial intelligence, industrial informatics, pattern processing and wireless automation devices.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.



## REFERENCES

- [1] N. N. Krasovskii and E. A. Lidskii, "Analysis design of controller in systems with random attributes-part 1," *Automation and Remote Control*, vol. 22, pp. 1021-1025, Jan. 1961.
- [2] X. Luan, B. Huang, and F. Liu, "Higher order moment stability region for Markov jump systems based on cumulant generating function," *Automatica*, vol. 93, pp. 389-396, Jul. 2018.
- [3] L. Zhang and E. K. Boukas, "Stability and stabilization of Markovian jump linear systems with partly unknown transition probabilities," *Automatica*, vol. 45, no. 2, pp. 463-468, Feb. 2009.
- [4] P. Shi and F. Li, "A survey on Markovian jump systems: Modeling and design," *International Journal of Control, Automation and Systems*, vol. 13, no. 1, pp. 1-16, Feb. 2015.
- [5] Y. Wang, Y. Xia, H. Shen, and P. Zhou, "SMC design for robust stabilization of nonlinear Markovian jump singular systems," *IEEE Transactions on Automatic Control*, In press, 2017, DOI: 10.1109/TAC.2017.2720970.
- [6] H. Li, P. Shi, D. Yao, and L. Wu, "Observer-based adaptive sliding mode control for nonlinear Markovian jump systems," *Automatica*, vol. 64, no. 1, pp. 133-142, Feb. 2016.
- [7] Y. Kao, J. Xie, C. Wang, and H. R. Karimi, "A sliding mode approach to  $H_\infty$  non-fragile observer-based control design for uncertain Markovian neutral-type stochastic systems," *Automatica*, vol. 52, pp. 218-226, Feb. 2015.
- [8] P. Shi, M. Liu, and L. Zhang, "Fault-tolerant sliding mode observer synthesis of Markovian jump systems using quantized measurements," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 9, pp. 5910-5918, Sept. 2015.
- [9] Y. Ma, X. Jia, and D. Liu, "Robust finite-time  $H_\infty$  control for discrete-time singular Markovian jump systems with time-varying delay and actuator saturation," *Applied and Computational Mathematics*, vol. 286, pp. 213-227, Aug. 2016.
- [10] Z. Mao, B. Jiang, and P. Shi, " $H_\infty$  fault detection filter design for networked control systems modelled by discrete Markovian jump systems," *IET Control Theory and Applications*, vol. 1, no. 5, pp. 1336-1343, May 2007.
- [11] P. Shi, F. Li, L. Wu, and C. C. Lim, "Neural network-based passive filtering for delayed neutral-type semi-markovian jump systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 9, pp. 2101-2114, Sept. 2017.
- [12] F. Li, L. Wu, P. Shi, and C. C. Lim, "State estimation and sliding mode control for semi-Markovian jump systems with mismatched uncertainties," *Automatica*, 51 (2015) 385-393.
- [13] H. Ma, H. Liang, Q. Zhu, and C. K. Ahn, "Adaptive dynamic surface control design for uncertain nonlinear strict-feedback systems with unknown control direction and disturbances," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, In press, 2018, DOI:10.1109/TSMC.2018.2855170.
- [14] H. Ma, Q. Zhou, L. Bai, and H. Liang, "Observer-based adaptive fuzzy fault-tolerant control for stochastic nonstrict-feedback nonlinear systems with input quantization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, In press, 2018, DOI: 10.1109/TSMC.2018.2833872
- [15] G. Tao, "Adaptive control design and analysis," *Wiley-IEEE Press*, 2003.
- [16] D. Kleinman, "On an iterative technique for Riccati equation computations," *IEEE Transactions on Automatic Control*, vol. 13, no. 1, pp. 114-115, Feb. 1968.
- [17] L. Lu and W. Lin, "An iterative algorithm for the solution of the discrete-time algebraic Riccati equation," *Linear Algebra and its Applications*, vol. 188-189, no. 1, pp. 465-488, Jul.-Aug.1993.
- [18] O. L. V. Costa and J. C. C. Aya, "Temporal difference methods for the maximal solution of discrete-time coupled algebraic Riccati equations," *In Proceedings of the American Control Conference, San Diego, CA, USA, IEEE Press*, pp. 1791-1795, 1999.
- [19] Z. Gajic and I. Borno, "Lyapunov iterations for optimal control of jump linear systems at steady state," *IEEE Transactions on Automatic Control*, vol. 40, no. 11, pp. 1971-1975, Nov. 1975.
- [20] W. He, Y. Dong, and C. Sun, "Adaptive neural impedance control of a robotic manipulator with input saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 3, pp. 334-344, Mar. 2016.
- [21] H. Shen, Y. Men, Z. Wu, and J. H. Park, "Nonfragile  $H_\infty$  control for fuzzy Markovian jump systems under fast sampling singular perturbation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, In press, DOI:10.1109/TSMC.2017.2758381, 2017.
- [22] Y. Xu, R. Lu, H. Peng, K. Xie, and A. Xue, "Asynchronous dissipative state estimation for stochastic complex networks with quantized jumping coupling and uncertain measurements," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 2, pp. 268-277, Feb. 2017.
- [23] J. Cheng, J. H. Park, and H. R. Karimi, "A flexible terminal approach to sampled-data exponentially synchronization of Markovian neural networks with time-varying delayed signals," *IEEE Transactions on Cybernetics*, vol. 48, no. 8, pp. 2232-2244, Aug. 2018.
- [24] D. Zhai, L. An, X. Li, and Q. Zhang, "Adaptive fault-tolerant control for nonlinear systems with multiple sensor faults and unknown control directions," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 9, pp. 4436-4446, Sep. 2018.
- [25] D. Zhai, L. An, D. Ye, and Q. Zhang, "Adaptive reliable  $H_\infty$  static output feedback control against Markovian jumping sensor failures," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 3, pp. 631-644, Mar. 2018.
- [26] D. Liu, Q. Wei, and P. Yan, "Generalized policy iteration adaptive dynamic programming for discrete-time nonlinear systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 12, pp. 1577-1591, May 2015.
- [27] Q. Wei and D. Liu, "Stable iterative adaptive dynamic programming algorithm with approximation errors for discrete-time nonlinear systems," *Neural Computing and Applications*, vol. 24, no. 6, pp. 1355-1367, May 2014.
- [28] Y. Liang, H. Zhang, G. Xiao, and H. Jiang, "Reinforcement learning-based online adaptive controller design for a class of unknown nonlinear discrete-time systems with time delays," *Neural Computing and Applications*, vol. 30, no. 6, pp. 1733-1745, Sep. 2018.
- [29] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, vol. 9, no. 3, pp. 32-50, Dec. 2009.
- [30] D. Vrabie and F. L. Lewis, "Adaptive dynamic programming for online solution of a zero-sum differential game," *IET Control Theory and Applications*, vol. 9, no. 3, pp. 353-360, Mar. 2011.
- [31] D. Vrabie and F. L. Lewis, "Adaptive optimal control algorithm for continuous-time nonlinear systems based on policy iteration," *In Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, pp. 73-79, 2009.
- [32] W. Guo, J. Si, F. Liu, and S. Mei, "Policy approximation in policy iteration approximate dynamic programming for discrete-time nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 2794-2807, Jul. 2018.
- [33] B. Kiumarsi, F. L. Lewis, H. Modares, A. Karimpour, and M. B. Naghibi-Sistani, "Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics," *Automatica*, vol. 50, no. 4, pp. 1167-1175, Apr. 2014.
- [34] Liu, Y. J, S. Li, C. T. Tong, and C. L. P. Chen, "Adaptive reinforcement learning control based on neural approximation for nonlinear discrete-time systems with unknown nonaffine dead-zone input," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 1, pp. 295-305, Jan. 2019.
- [35] H. N. Wu and B. Luo, "Simultaneous policy update algorithms for learning the solution of linear continuous-time  $H_\infty$  state feedback control," *Information Sciences*, vol. 222, no. 11, pp. 472-485, Feb. 2013.

- [36] C. Mu, D. Wang, and H. He, "Data-driven finite-horizon approximate optimal control for discrete-time nonlinear systems using iterative HDP approach," *IEEE Transactions on Cybernetics*, In press, 2017, DOI: 10.1109/TCYB.2017.2752845.
- [37] X. He, T. Huang, J. Yu, C. Li, and Y. Zhang, "A continuous-time algorithm for distributed optimization based on multiagent networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, In press, 2017, DOI: 10.1109/TSMC.2017.2780194.
- [38] X. Yang, H. He, and Y. Liu, "Adaptive dynamic programming for robust neural control of unknown continuous-time nonlinear systems," *IET Control Theory and Applications*, vol. 11, no. 14, pp. 2307-2316, Nov. 2017.
- [39] W. Xu, Z. Huang, L. Zuo, and H. He, "Manifold-based reinforcement learning via locally linear reconstruction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 4, pp. 934-947, Apr. 2017.
- [40] M. M. Alipour, S. N. Razavi, M. R. F. Derakhshi, and M. A. Balafar, "A hybrid algorithm using a genetic algorithm and multiagent reinforcement learning heuristic to solve the traveling salesman problem," *Neural Computing and Applications*, vol. 30, no. 9, pp. 2935-2951, Nov. 2018.
- [41] Y. Zhu and D. Zhao, "A data-based online reinforcement learning algorithm satisfying probably approximately correct principle," *Neural Computing and Applications*, vol. 26, no. 4, pp. 775-787, May. 2015.
- [42] L. Tang, Y. Liu, and S. Tong, "Adaptive neural control using reinforcement learning for a class of robot manipulator," *Neural Computing and Applications*, vol. 25, no. 1, pp. 135-141, Jul. 2014.
- [43] C. Mu, Z. Ni, C. Sun, and H. He, "Air-breathing hypersonic vehicle tracking control based on adaptive dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 584-598, Mar. 2017.
- [44] X. Xie, D. Yue, and S. Hu, "Fault estimation observer design of discrete-time nonlinear systems via a joint real-time scheduling law," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 7, pp. 1451-1463, Jul. 2017.
- [45] S. He, J. Song, Z. Ding, and F. Liu, "Online adaptive optimal control for continuous-time Markov jump linear systems using a novel policy iteration algorithm," *IET Control Theory and Applications*, vol. 9, no. 10, pp. 1536-1543, Oct. 2015.
- [46] J. Song, S. He, F. Liu, Y. Niu, and Z. Ding, "Data-driven policy iteration algorithm for optimal control of continuous-time Itô stochastic systems with Markovian jumps," *IET Control Theory and Applications*, vol. 10, no. 12, pp. 1431-1439, Dec. 2016.
- [47] J. Song, S. He, Z. Ding, and F. Liu, "A new iterative algorithm for solving  $H_\infty$  control problem of continuous-time Markovian jumping linear systems based on online implementation," *International Journal of Robust and Nonlinear Control*, vol. 26, no. 17, pp. 3737-3754, Mar. 2016.
- [48] Z. Gajic and I. Borno, "General transformation for block diagonalization of weakly coupled linear systems composed of N-subsystems," *IEEE Transactions on Circuits and Systems I Fundamental Theory & Applications*, vol. 47, no. 6, pp. 909-912, Jun. 2000.