Journal of Systems Engineering and Electronics

Vol. 33, No. 5, October 2022, pp.1186 – 1194

Hybrid Q-learning for data-based optimal control of non-linear switching system

LI Xiaofeng[1,2], DONG Lu[3], and SUN Changyin[1,2,*]

1. School of Automation, Southeast University, Nanjing 210096, China; 2. School of Artificial Intelligence, Anhui University, Hefei 230601, China; 3. School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China

Abstract: In this paper, the optimal control of non-linear switching system is investigated without knowing the system dynamics. First, the Hamilton-Jacobi-Bellman (HJB) equation is derived with the consideration of hybrid action space. Then, a novel data-based hybrid Q-learning (HQL) algorithm is proposed to find the optimal solution in an iterative manner. In addition, the theoretical analysis is provided to illustrate the convergence and optimality of the proposed algorithm. Finally, the algorithm is implemented with the actor-critic (AC) structure, and two linear-in-parameter neural networks are utilized to approximate the functions. Simulation results validate the effectiveness of the data-driven method.

Keywords: switching system, hybrid action space, optimal control, reinforcement learning, hybrid Q-learning (HQL).

DOI: 10.23919/JSEE.2022.000114

1. Introduction

In contrast to the conventional non-linear system, the dynamics of switching system can be described by an interaction of a discrete switching policy and several continuous subsystems [1]. The properties of stability, controllability, and observability have been well studied in existing literatures [2,3]. Besides stability, optimality is more preferred for designing controller of real-world applications. The optimal control problem aims to find an admissible policy that can stabilize the controlled system as well as optimizing the predefined performance [4]. In general, the optimal solution can be obtained by solving the corresponding Hamilton-Jacobi-Bellman (HJB) equation. The family of classical methods includes the classical variational method, Pontryagin’s maximum principle, and dynamic programming [5]. In particular, as for discrete-time dynamics systems, dynamic programming method has been successfully applied in many fields of engineering. However, it suffers from the “curse of dimensionality” problem so that the computation cost is very high with the increasing of system dimensions [6].

In recent years, the optimal control problem of switching systems has attained much attention since many real-world applications from aerospace systems to traffic signal control system can be addressed as switching organisms [7–10]. In general, the related work can be divided into two categories. The switching system with autonomous subsystems has attained much attention from researchers. Without considering the control input, the task is simplified to find the optimal switching scheduling. A kind of gradient projection-based methods is proposed for general continuous-time non-linear hybrid systems. The local minima of cost function is found along the direction of the gradient [11,12]. In [13], researchers considered the optimal scheduling problem of linear switching systems with pre-specified mode sequence. The optimal switching time instances are determined by using calculus of variations method. Note that it is required to fix and know the active mode sequence for these non-linear programming-based methods. So the planning process should be re-computed if the initial state is changed.

As for the switching system with controlled subsystems, it is required to co-design the switching policy and the control policies of subsystems to optimize the performance function. In [14], a direct search scheme based on the Luus-Jakola optimization technique was proposed to address the optimal control of general switched linear quadratic systems. Also, the sequence of active mode is pre-fixed which simplifies the control problem of non-autonomous switching systems. In addition, a two point boundary value differential algebraic equation (DAE) is solved to explore the optimal solutions numerically [15]. In [16−18], discretization-based algorithms were pro-

Manuscript received January 13, 2021.

*Corresponding author.

This work was supported by the National Key R&D Program of China (2018AAA0101400), the Natural Science Foundation of Jiangsu Province of China (BK20202006), and the National Natural Science Foundation of China (61921004;62173251).

posed which divides the state and input space with a finite number of options. However, the above planning based algorithms also suffer from high computation cost and limited range of initial state.

Recently, the reinforcement learning (RL) method has been utilized to learn the optimal policy of Markov decision process (MDP) by interacting with the environment [19–21]. The actor-critic (AC) structure is commonly employed to implement the algorithm, where the critic network approximates the value function and the actor network approximates the control policy [22–24]. Value iteration [25] and policy iteration [26] are two typical classes of model-based methods which require to know the accurate system dynamics. In [27], researchers proposed a model-free algorithm for the optimal control of unknown non-linear system by pre-training a model network. In addition, a series of data-based schemes were proposed to learn the optimal policy completely based on interactive data [28–31]. Considering its adaptive property and feedback formulation, a novel RL scheme is proposed to determine the optimal scheduling for switching systems which achieves good performance. In [32], the problem of multi therapeutic human immunodeficiency virus treatment was formulated as to find the optimal solution of a finite-horizon autonomous switching system. The optimal value function was learned by using the value iteration (VI) based method. Then, the decision can be made by simply comparing several scalar values. Moreover, researchers extended this work to general autonomous nonlinear switching systems with rigorous convergence proof [33]. In addition, switching cost penalty and minimum dwell time constraint were considered [34,35]. Note that the systems in [32–35] all take the finite-horizon objective functions with terminal state constraints. However, the controlled switching system is rarely studied for RL control design. In [36], researchers proposed a model-based algorithm to co-design the optimal policy of the non-linear switching system with control constraint. In [37], a neural network was first trained to learn the model and an iterative algorithm was designed to generate a sequence of Q-functions which finally converges to the optimal solution.

Considering the complex dynamics of controlled switching system, it is rather difficult to obtain the exact system dynamics. While the model can be identified by training a model network, the model error can not be neglected. In this paper, a novel hybrid RL algorithm is proposed to learn the optimal policy of non-linear switching systems. The main contributions are as follows.

(i) Considering the hybrid policy of discrete switching signal and continuous control input, the corresponding HJB equation is constructed based on the Bellman's optimality principle.

(ii) An iterative RL algorithm is proposed to find the optimal hybrid policy without knowing the system dynamics or pre-training the model network.

(iii) The convergence proof of iterative Q-functions is provided.

The rest of this paper is organized as follows. In Section 2, we first analyse the hybrid nature of action space and derive the transformed HJB equation. Section 3 proposes the design of hybrid RL algorithm as well as the detailed implementation steps with AC structure. The convergence proof is given in Section 4. In Section 5, two numerical examples are provided to demonstrated the performance of the proposed method. Finally, conclusions are provided in Section 6.

2. Problem formulation

Consider the general non-linear switching system with the following dynamics:

$$x_{k+1} = f_{v_k}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{1}$$

where $\boldsymbol{x}_k \in \boldsymbol{\Omega}_x \subseteq \mathbf{R}^n$ and $\boldsymbol{u}_k \in \boldsymbol{\Omega}_u \subseteq \mathbf{R}^m$ denote the system states and control parameters, respectively. The subscript k denotes the index of time step. Both $\boldsymbol{\Omega}_x$ and $\boldsymbol{\Omega}_u$ are compact and connected sets. The notation v denotes the index of subsystem and there are a number of P subsystems. The notation $\mathcal{P} = \{1, 2, \cdots, P\}$ denotes the set of available subsystems. It is assumed that $f_v : \boldsymbol{\Omega}_x \times \boldsymbol{\Omega}_u \to \boldsymbol{\Omega}_x$ is Lipschitz continuous with $f_v(0,0) = 0$.

In contrast to the conventional non-linear systems, the controller of switching system need to co-design the switching signal and control input. Consequently, the control signal at each time step is a tuple of $(v, \boldsymbol{u}_v)$ where the subscript of $\boldsymbol{u}_v$ denotes the coupling between active mode and control input. Then, the action space can be formulated by

$$A = \cup_{v \in \mathcal{P}} \{(v, \boldsymbol{u}) | v \in \mathcal{P}, \boldsymbol{u} \in \boldsymbol{\Omega}_u\}. \tag{2}$$

Afterwards, the performance function is defined as follows:

$$J = \sum_{k=0}^{\infty} U(\boldsymbol{x}_k, v_k, \boldsymbol{u}_k) \tag{3}$$

where the cost function is defined by $U(\boldsymbol{x}, v, \boldsymbol{u}) = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{Q}_v \boldsymbol{x} + \boldsymbol{u}^{\mathrm{T}} \boldsymbol{R}_v \boldsymbol{u}$ where $\boldsymbol{Q}_v \in \mathbf{R}^{n \times n}$and $\boldsymbol{R}_v \in \mathbf{R}^{m \times m}$ are positive definite matrices.

Let $\pi_v(\boldsymbol{x})$ and $\pi_{u_v}(\boldsymbol{x})$ denote the policies of discrete action and its corresponding continuous parameters, respectively. For notation clarity, we utilize $\pi(\boldsymbol{x})$ to represent the hybrid control policy, i.e., $\pi(\boldsymbol{x}) = (\pi_v(\boldsymbol{x}), \pi_{u_v}(\boldsymbol{x}))$. In order to derive the algorithm, we first introduce a

Q-function with respect to any given policy $\pi(\boldsymbol{x})$ as follows:

$$Q^{\pi}(\boldsymbol{x}_k, v_k, \boldsymbol{u}_k) = U(\boldsymbol{x}_k, v_k, \boldsymbol{u}_k) + \sum_{l=k+1}^{\infty} U(\boldsymbol{x}_l, \pi_v(x_l), \pi_{u_v}(\boldsymbol{x}_l)). \quad (4)$$

That is to say, the Q-function denotes the accumulated costs if the system starts in state $\boldsymbol{x}$ and takes an arbitrary hybrid action $(v,\boldsymbol{u})$, and then taking hybrid actions generated by the hybrid policy $\pi(\boldsymbol{x})$ thereafter.

Afterwards, based on the Bellman's optimality principle [38], the corresponding HJB equation can be obtained:

$$Q^{*}(\boldsymbol{x}_k, v_k, \boldsymbol{u}_k) = U(\boldsymbol{x}_k, v_k, \boldsymbol{u}_k) + \min_{v \in \mathcal{P}} \inf_{\boldsymbol{u} \in \boldsymbol{\Omega}_u} Q^{*}(\boldsymbol{x}_{k+1}, v, \boldsymbol{u}) \quad (5)$$

where $Q^{*}(\boldsymbol{x}, v, \boldsymbol{u})$ denotes the optimal Q-function of $\pi^{*}(\boldsymbol{x})$. For notation simplicity, we let $\boldsymbol{x}$, v, and $\boldsymbol{u}$ denote the current state, discrete action, and continuous parameters while $\boldsymbol{x}'$, v', and $\boldsymbol{u}'$ denote the state, discrete action, and continuous parameters at the next time step, respectively.

3. Hybrid Q-learning algorithm and its convergence analysis

In order to co-design the policies of switching signal and control input, a novel hybrid Q-learning (HQL) algorithm is proposed in this section. In addition, the implementation details of AC structure is provided by using linear-in-parameter (LIP) neural network (NN) as function approximator.

3.1 HQL algorithm

The algorithm starts with the initial Q-functions, i.e., $Q_v^{(0)} = 0, \forall v \in \mathcal{P}$. For each $v \in \mathcal{P}$, its corresponding continuous parameters policy can be obtained by taking the infimum over $\boldsymbol{\Omega}_u$:

$$\pi_{u_v}^{(0)}(\boldsymbol{x}) = \arg \inf_{\boldsymbol{u} \in \boldsymbol{\Omega}_u} Q_v^{(0)}(\boldsymbol{x}, v, \boldsymbol{u}). \quad (6)$$

Then, the Q-function can be updated by

$$Q_v^{(1)}(\boldsymbol{x}, v, \boldsymbol{u}) = U(\boldsymbol{x}, v, \boldsymbol{u}) + \min_{v' \in \mathcal{P}} Q_{v'}^{(0)}(\boldsymbol{x}', v', \pi_{u_{v'}}^{(0)}(\boldsymbol{x}')). \quad (7)$$

For $i = 1, 2, \cdots$, one iterates between

$$\pi_{u_v}^{(i)}(\boldsymbol{x}) = \arg \inf_{\boldsymbol{u} \in \boldsymbol{\Omega}_u} Q_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}), \quad (8)$$

and

$$Q_v^{(i+1)}(\boldsymbol{x}, v, \boldsymbol{u}) = U(\boldsymbol{x}, v, \boldsymbol{u}) + \min_{v' \in \mathcal{P}} Q_{v'}^{(i)}(\boldsymbol{x}', v', \pi_{u_{v'}}^{(i)}(\boldsymbol{x}')). \quad (9)$$

Consequently, the HQL algorithm generates a sequence of Q-function $\{Q_v^{(i)}\}_{i=0}^{\infty}$ that will converge to the optimal solution of (5). Once the optimal Q-function is obtained, the optimal continuous policy can be computed by substituting Q_v^{*} into (8) while the optimal discrete action can be simply determined be comparing the value of different Q-functions. The convergence proof is given in Section 4.

3.2 Implementation with the AC structure

Consider the continuous state space, and LIP NNs are employed as function approximators. Specifically, for each mode, there exists a corresponding actor network and critic network.

Let $Q_v(\boldsymbol{x}, v, \boldsymbol{u}; \boldsymbol{W}_{c,v})$ denote the output of the critic network so that

$$Q_v(\boldsymbol{x}, v, \boldsymbol{u}; \boldsymbol{W}_{c,v}) = \boldsymbol{W}_{c,v}^{\mathrm{T}} \boldsymbol{\phi}_v(\boldsymbol{x}, \boldsymbol{u}) \quad (10)$$

and let $\mu_v(\boldsymbol{x}; \boldsymbol{W}_{a,v})$ denote the output of the actor network so that

$$\mu_v(\boldsymbol{x}; \boldsymbol{W}_{a,v}) = \boldsymbol{W}_{a,v}^{\mathrm{T}} \boldsymbol{\sigma}_v(\boldsymbol{x}) \quad (11)$$

where $\boldsymbol{W}_{c,v}$ and $\boldsymbol{W}_{a,v}$ denote the weights of the critic and actor networks, respectively. In addition, $\boldsymbol{\phi}_v(\cdot)$ and $\boldsymbol{\sigma}_v(\cdot)$ represent the activation function of the critic and actor networks, respectively.

Let $\mathcal{D}$ denote a data buffer with memory size M. To begin with, the HQL algorithm needs to sample a few transitions from the state and hybrid action spaces and stores them into $\mathcal{D}$. Specifically, according to uniform random distribution, we sample M states from $\boldsymbol{\Omega}_x$ and M parameters from $\boldsymbol{\Omega}_u$. By substituting $(\boldsymbol{x}, v, \boldsymbol{u})$ into (1), one can receive the corresponding cost function U_d and next state $\boldsymbol{x}_{d+1}$. Then, the tuple of transitions $\{(\boldsymbol{x}_d, v_d, \boldsymbol{u}_d, U_d, \boldsymbol{x}_{d+1})\}_{d=1}^{D}$ are stored into $\mathcal{D}$. Note that although with the same $\boldsymbol{x}_d$ and $\boldsymbol{u}_d$, by selecting different v_d, one can obtain different U_d and $\boldsymbol{x}_{d+1}$ so that all subsystems can be explored sufficiently.

First, the critic networks are initialized with $Q_v^{(0)}(\boldsymbol{x}, v, \boldsymbol{u}; \boldsymbol{W}_{c,v}) = 0$. For each mode, a batch of transitions $\{(\boldsymbol{x}_b, v_b, \boldsymbol{u}_b, U_b, \boldsymbol{x}_{b+1})\}_{b=1}^{B}$ are randomly sampled from $\mathcal{D}$, where B denotes the batch size. According to (8), for any iteration i, the target value of actor network is

$$\mu_v^{(i)}(\boldsymbol{x}) = \arg \inf_{\boldsymbol{u} \in \boldsymbol{\Omega}_u} Q_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}). \quad (12)$$

Define

$$\boldsymbol{\sigma}_v = [\boldsymbol{\sigma}_v(\boldsymbol{x}_1), \boldsymbol{\sigma}_v(\boldsymbol{x}_2), \cdots, \boldsymbol{\sigma}_v(\boldsymbol{x}_B)],$$

and

$$\boldsymbol{\mu}_v^{(i)} = \left[\boldsymbol{\mu}_v^{(i)}(\boldsymbol{x}_1), \boldsymbol{\mu}_v^{(i)}(\boldsymbol{x}_2), \cdots, \boldsymbol{\mu}_v^{(i)}(\boldsymbol{x}_B)\right].$$

Then, by using the least square method (LSM), the weights of actor network can be computed by

$$\boldsymbol{W}_{a,v}^{(i)} = (\boldsymbol{\sigma}_v \boldsymbol{\sigma}_v^{\mathrm{T}})^{-1} \boldsymbol{\mu}_v^{(i)}. \quad (13)$$

Afterwards, the target value of critic network can be obtained by

$$\boldsymbol{y}_v^{(i+1)} = U(\boldsymbol{x}_b, v_b, \boldsymbol{u}_b) + \min_{v \in \mathcal{P}} (\boldsymbol{W}_{c,v}^{(i)})^{\mathrm{T}} \boldsymbol{\phi}_v(\boldsymbol{x}_{b+1}, \boldsymbol{\mu}_v^{(i)}(\boldsymbol{x}_{b+1})). \quad (14)$$

Define

$$\boldsymbol{\phi}_v = [\boldsymbol{\phi}_v(\boldsymbol{x}_1, \boldsymbol{u}_1), \boldsymbol{\phi}_v(\boldsymbol{x}_2, \boldsymbol{u}_2), \cdots, \boldsymbol{\phi}_v(\boldsymbol{x}_B, \boldsymbol{u}_B)]$$

and

$$\boldsymbol{y}_v^{(i+1)} = \left[\boldsymbol{y}_v^{(i+1)}(\boldsymbol{x}_1, \boldsymbol{u}_1), \boldsymbol{y}_v^{(i+2)}(\boldsymbol{x}_2, \boldsymbol{u}_2), \cdots, \boldsymbol{y}_v^{(i+1)}(\boldsymbol{x}_B, \boldsymbol{u}_B)\right].$$

Consequently, by using LSM, the weights of actor network can be computed by

$$\boldsymbol{W}_{c,v}^{(i+1)} = (\boldsymbol{\phi}_v \boldsymbol{\phi}_v{}^{\mathrm{T}})^{-1} \boldsymbol{y}_v^{(i+1)}. \quad (15)$$

Remark 1 By using the LIP NNs with linear independent polynomial basis functions, the weights can be updated as the one shot solution based on the LSM at each iteration step. Since the training is an iterative process, this can significantly accelerate the convergence procedure. In addition, it is worth noting that the proposed algorithm is not limited to LIP NNs, one can utilize multilayer perceptrons or even deep NNs, for improving the approximation capability of the NN.

Motivated by [20], the target networks are utilized to stabilize the training process. The detailed implementation steps of HQL algorithm are given in Algorithm 1.

Algorithm 1 HQL algorithm

```
Initialize the normalized Q networks of discrete action $v$ with $Q_v^{(0)}(\boldsymbol{x}, v, \boldsymbol{u}; \boldsymbol{W}_{c,v})$;
Initialize the target networks $Q'^{(0)}_v(\boldsymbol{x}, v, \boldsymbol{u}; \boldsymbol{W}_{c,v'})$ with $\boldsymbol{W}_{c,v'} \leftarrow \boldsymbol{W}_{c,v}$;
Initialize the data buffer $\mathcal{D} \leftarrow \varnothing$;
Randomly sample $\{\boldsymbol{x}_d\}_{d=1}^M$ from $\boldsymbol{\Omega}_x$ and $\{\boldsymbol{u}_d\}_{d=1}^M$ from $\boldsymbol{\Omega}_u$, respectively;
For $v = 1, 2, \cdots, P$ do
    Execute hybrid action $\{(v, \boldsymbol{u}_d)\}_{d=1}^M$ and receive $\{U_d\}_{d=1}^M$ and $\{\boldsymbol{x}_{d+1}\}_{d=1}^M$, where $\boldsymbol{x}_{d+1} = f_v(\boldsymbol{x}_d, \boldsymbol{u}_d)$;
    Store the collected transitions $\{(\boldsymbol{x}_d, v, \boldsymbol{u}_d, U_d, \boldsymbol{x}_{d+1})\}_{d=1}^M$ into the $\mathcal{D}$;
End For
For $i = 1, 2, \cdots, I$ do
    For $v = 1, 2, \cdots, P$ do
        Sample a batch of $B$ transitions $\{(\boldsymbol{x}_b, v_b, \boldsymbol{u}_{v_b}, U_b, \boldsymbol{x}_{b+1} | v_b = v)\}_{b=1}^B$ from $\mathcal{D}$;
        Update the actor network according to (13);
        Update the critic network according to (15);
    End For
    Update the weights of target networks by $\boldsymbol{W}_{c,v}^{(i+1)'} \leftarrow \boldsymbol{W}_{c,v}^{(i+1)}$
End For
```

4. Convergence analysis

The convergence proof derived by extending the theoretical analysis in [36]. Before proceeding, the following definition and assumption are given.

Definition 1 [36] The hybrid policy $(\pi_v(\boldsymbol{x}), \pi_{u_v}(\boldsymbol{x}))$ is defined to be admissible within $\boldsymbol{\Omega}_x$ if there exists an upper bound $\mathcal{Z}(\boldsymbol{x})$ for its performance function

$$J(x_0) = \sum_{k=0}^{\infty} U(\boldsymbol{x}_k, \pi_v(\boldsymbol{x}_k), \pi_{u_v}(\boldsymbol{x}_k)) \leqslant \mathcal{Z}(\boldsymbol{x}_k).$$

Assumption 1 For the controlled switching system, there exists at least one admissible hybrid policy $(\pi_v(\boldsymbol{x}), \pi_u(\boldsymbol{x}))$ within $\boldsymbol{\Omega}_x$.

Lemma 1 Let $\{Q_v^{(i)}\}_{i=0}^{\infty}$ denote the Q-function sequence and $\{\pi_{u_v}^{(i)}\}_{i=0}^{\infty}$ denote the sequence of continuous parameter policy generated by (8) and (9), respectively. Given an arbitrary hybrid policy $(\varpi_v(\boldsymbol{x}), \varpi_{u_v}(\boldsymbol{x}))$, the corresponding sequence $\{\boldsymbol{\Pi}_v^{(i)}\}_{i=0}^{\infty}$ satisfies

$$\boldsymbol{\Pi}_v^{(i+1)}(\boldsymbol{x}, v, \boldsymbol{u}) = U(\boldsymbol{x}, v, \boldsymbol{u}) + \Pi^{(i)}(\boldsymbol{x}', \boldsymbol{\varpi}_v^{(i)}(\boldsymbol{x}'),, \varpi_{u_v}^{(i)}(\boldsymbol{x}')) \quad (16)$$

where $\boldsymbol{x}' = f_v(\boldsymbol{x}, \boldsymbol{u})$. If $\Pi^{(0)} = Q_v^{(0)} = 0$ holds for any $v \in \mathcal{P}$, there is $\Pi_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}) \geqslant Q_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}), \forall i$.

Proof Because $(\pi_v^{(i)}(\boldsymbol{x}), \pi_{u_v}^{(i)}(\boldsymbol{x}))$ minimize the right-hand side of (9) while $(\varpi_v(\boldsymbol{x}), \varpi_{u_v}(\boldsymbol{x}))$ is arbitrarily chosen, and because $\Pi_v^{(0)} = Q_v^{(0)} = 0, \forall v \in \mathcal{P}$, it is straightforward to know that $\Pi_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}) \geqslant Q_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}), \forall i$ by induction. □

Lemma 2 Let $\{Q_v^{(i)}\}_{i=0}^{\infty}$ denote the Q-function sequence and $\{\pi_{u_v}^{(i)}\}_{i=0}^{\infty}$ denote the sequence of continuous parameter policy generated by (8) and (9) with $Q_v^{(0)} = 0$, respectively. If Assumption 1 holds, there exists a finite upper bound $\mathcal{Z}(\boldsymbol{x})$ satisfying

$$Q_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}) \leqslant \mathcal{Z}(x), \quad \forall v \in \mathcal{P}. \quad (17)$$

Proof Let $(\vartheta_v(\boldsymbol{x}), \vartheta_{u_v}(\boldsymbol{x}))$ denote an arbitrary admissible hybrid policy and let $\boldsymbol{\Theta}_v^{(0)}(\boldsymbol{x}, v, \boldsymbol{u}) = Q_v^{(0)}(\boldsymbol{x}, v, \boldsymbol{u}) = 0$, where $\boldsymbol{\Theta}_v^{(i+1)}(\boldsymbol{x}, v, \boldsymbol{u})$ is generated by

$$\boldsymbol{\Theta}_v^{(i+1)}(\boldsymbol{x}, v, \boldsymbol{u}) = U(\boldsymbol{x}, v, \boldsymbol{u}) + \boldsymbol{\Theta}_{v'}^{(i)}(\boldsymbol{x}', \vartheta_v(\boldsymbol{x}'), \vartheta_{u_v}(\boldsymbol{x}')). \quad (18)$$

Motivated by the Lemma 2 in [35], we can obtain

$$\boldsymbol{\Theta}_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}) \leqslant U(\boldsymbol{x}, v, \boldsymbol{u}) + \sum_{l=1}^{\infty} U(\boldsymbol{x}', \vartheta_v(\boldsymbol{x}'), \vartheta_{u_v}(\boldsymbol{x}')). \quad (19)$$

Because $(\vartheta_v(\boldsymbol{x}), \vartheta_{u_v}(\boldsymbol{x}))$ is an admissible hybrid policy, there is

$$\boldsymbol{\Theta}_v^{(i)}(\boldsymbol{x}, v, \boldsymbol{u}) \leqslant \mathcal{Z}(\boldsymbol{x}). \quad (20)$$

According to Lemma 1, we have

$$Q_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant \boldsymbol{\Theta}_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant \mathcal{Z}(\boldsymbol{x}). \tag{21}$$

□

Theorem 1 Let $\{Q_v^{(i)}\}_{i=0}^{\infty}$ denote the Q-function sequence and $\{\pi_{u_v}^{(i)}\}_{i=0}^{\infty}$ denote the sequence of continuous parameter policy generated by (8) and (9) with $Q_v^{(0)}=0$, respectively. If Assumption 1 holds, the generated Q-function sequence is non-decreasing, i.e., $Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) \geqslant Q_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}), \forall v \in \mathcal{P}$.

Proof Let $(\varphi_v(\boldsymbol{x}),\varphi_{u_v}(\boldsymbol{x}))$ denote an arbitrary hybrid policy. Starting with $\Phi_v^{(0)}=Q_v^{(0)}=0$, its corresponding Q-function $\Phi_v^{(i)}$ is defined by

$$\Phi_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) = U(\boldsymbol{x},v,\boldsymbol{u}) + \Phi_{v'}^{(i)}(\boldsymbol{x}',\varphi_v(\boldsymbol{x}'),\varphi_{u_v}(\boldsymbol{x}')) \tag{22}$$

where the subscript $v'=\Phi_v(\boldsymbol{x}')$.

Then, it follows from Lemma 1 that $Q_v^{(i)} \leqslant \Phi_v^{(i)}, \forall i$ holds. Afterwards, assume that $\varphi_v(\boldsymbol{x})=\pi_v^{(i+1)}(\boldsymbol{x})$ and $\varphi_{u_v}(\boldsymbol{x})=\pi_{u_v}^{(i+1)}(\boldsymbol{x})$, we have

$$\Phi_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) = U(\boldsymbol{x},v,\boldsymbol{u}) + \Phi_{v'}^{(i)}(\boldsymbol{x}',\varphi_v(\boldsymbol{x}'),\varphi_{u_v}(\boldsymbol{x}')). \tag{23}$$

By using mathematical induction method, the inequality $Q_v^{(i+1)} \geqslant \Phi_v^{(i)}$ can be proved to hold for any i. To begin with, for $i=0$, there is

$$Q_v^{(1)}(\boldsymbol{x},v,\boldsymbol{u}) - \Phi_v^{(0)}(\boldsymbol{x},v,\boldsymbol{u}) = U(\boldsymbol{x},v,\boldsymbol{u}) \geqslant 0. \tag{24}$$

If $i \geqslant 1$, assume that $Q_v^{(i)} \geqslant \Phi_v^{(i-1)}$ holds for $\forall i-1$. Then, by subtracting $\Phi_v^{(i)}$ from $Q_v^{(i+1)}$, one has

$$Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) - \Phi_v(\boldsymbol{x},v,\boldsymbol{u}) = Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) - \Phi_v(\boldsymbol{x},v,\boldsymbol{u}) \geqslant 0. \tag{25}$$

Moreover, one knows $Q_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant \Phi_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u})$ according to Lemma 1. Consequently, it follows that

$$Q_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}). \tag{26}$$

□

Theorem 2 Let $\{Q_v^{(i)}\}_{i=0}^{\infty}$ denote the Q-function sequence and $\{\pi_{u_v}^{(i)}\}_{i=0}^{\infty}$ denote the sequence of continuous parameter policy generated by (8) and (9) with $Q_v^{(0)}=0$, respectively. If Assumption 1 holds, then, there exists a limit function of $\{Q_v^{(i)}\}_{i=0}^{\infty}$, i.e., $Q_v^{(\infty)}=\lim\limits_{i\to\infty} Q_v^{(i)}$. In addition, $Q_v^{(\infty)}$ is the solution of (5) and the sequence of $\{\pi_{u_v}^{(i)}\}_{i=0}^{\infty}$ converges to $\pi_{u_v}^{*}$.

Proof According to Theorem 1 and Lemma 2, it can be known that $\{Q_v^{(i)}\}_{i=0}^{\infty}$ is non-decreasing and upper bounded by a finite function. Therefore, one knows that

$$Q_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}). \tag{27}$$

Let $(\omega_v(\boldsymbol{x}),\omega_{u_v}(\boldsymbol{x}))$ denotes an arbitrary hybrid policy. From Lemma 1, one knows that

$$Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant U(\boldsymbol{x},v,\boldsymbol{u}) + Q_{v'}^{(i)}(\boldsymbol{x}',\omega_v(\boldsymbol{x}'),\omega_{u_v}(\boldsymbol{x}')) \tag{28}$$

where $v'=\omega_v(\boldsymbol{x}')$. Then, it follows from Theorem 1 that

$$Q_v^{(i+1)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant U(\boldsymbol{x},v,\boldsymbol{u}) + Q_{v'}^{(\infty)}(\boldsymbol{x}',\omega_v(\boldsymbol{x}'),\omega_{u_v}(\boldsymbol{x}')). \tag{29}$$

Let $i\to\infty$, we have

$$Q_v^{(\infty)}(\boldsymbol{x},v,\boldsymbol{u}) \leqslant U(\boldsymbol{x},v,\boldsymbol{u}) + Q^{(\infty)}(\boldsymbol{x}',\omega_v(\boldsymbol{x}'),\omega_{u_v}(\boldsymbol{x}')). \tag{30}$$

Because $(\omega_v(\boldsymbol{x}),\omega_{u_v}(\boldsymbol{x}))$ is arbitrarily chosen, we have

$$Q_v^{(\infty)}(\boldsymbol{x}_k,v_k,\boldsymbol{u}_k) \leqslant U(\boldsymbol{x}_k,v_k,\boldsymbol{u}_k) + \min_{v'\in\mathcal{P}} \inf_{\boldsymbol{u}\in\boldsymbol{\Omega}_u} Q^{(\infty)}(\boldsymbol{x}_{k+1},v,\boldsymbol{u}). \tag{31}$$

On the other hand, from (9), we can obtain

$$Q_v^{(i)}(\boldsymbol{x},v,\boldsymbol{u}) = U(\boldsymbol{x},v,\boldsymbol{u}) + \min_{v'\in\mathcal{P}} \inf_{\boldsymbol{u}'\in\boldsymbol{\Omega}_u} Q^{(i-1)}(\boldsymbol{x}',v',\boldsymbol{u}'). \tag{32}$$

Then, it follows from Theorem 1 that

$$Q_v^{(\infty)}(\boldsymbol{x},v,\boldsymbol{u}) \geqslant U(\boldsymbol{x},v,\boldsymbol{u}) + \min_{v'\in\mathcal{P}} \inf_{\boldsymbol{u}'\in\boldsymbol{\Omega}_u} Q^{(i-1)}(\boldsymbol{x}',v',\boldsymbol{u}'). \tag{33}$$

Let $i\to\infty$, one gets

$$Q_v^{(\infty)}(\boldsymbol{x},v,\boldsymbol{u}) \geqslant U(\boldsymbol{x},v,\boldsymbol{u}) + \min_{v'\in\mathcal{P}} \inf_{\boldsymbol{u}'\in\boldsymbol{\Omega}_u} Q^{(\infty)}(\boldsymbol{x}',v',\boldsymbol{u}'). \tag{34}$$

Based on (31) and (34), it is straightforward to know that

$$Q_v^{(\infty)}(\boldsymbol{x},v,\boldsymbol{u}) = U(\boldsymbol{x},v,\boldsymbol{u}) + \min_{v'\in\mathcal{P}} \inf_{\boldsymbol{u}'\in\boldsymbol{\Omega}_u} Q^{(\infty)}(\boldsymbol{x}',v',\boldsymbol{u}'). \tag{35}$$

Therefore, one knows that $Q^{(\infty)}$ is the solution of the HJB equation. Once Q_v^{*} is known, according to (8), the optimal policy $\pi_{u_v}^{*}$ can be obtained. □

5. Numerical analysis

Two simulation examples are provided to evaluate the performance of the HQL algorithm. The code is run on Matlab 2018a with Intel Core i7 3.2 GHz processor.

(i) Example 1

First, the HQL algorithm is applied to a linear switching system with two modes:

$$\begin{cases} \boldsymbol{x}_{k+1} = \begin{bmatrix} 0.1 & 2 \\ 1 & 1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \boldsymbol{u}_k, & v=1 \\ \boldsymbol{x}_{k+1} = \begin{bmatrix} 1 & 0.5 \\ 0 & 1.1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \boldsymbol{u}_k, & v=2 \end{cases}. \tag{36}$$

The domains of interest are selected as $\boldsymbol{\Omega}_x=\{\boldsymbol{x}\in\mathbf{R}^2 : |\boldsymbol{x}_i|\leqslant 5, \forall i\}$ and $\Omega_u=\{u\in\mathbf{R} : |u|\leqslant 5\}$. The cost function is defined as $U(\boldsymbol{x},v,\boldsymbol{u})=6\times(\boldsymbol{x}_1^2+\boldsymbol{x}_2^2)+\boldsymbol{u}^2$. The LIP NNs are employed to implement the algorithm with the following activation functions:

$$\boldsymbol{\phi}_v(\boldsymbol{x},\boldsymbol{u}) = [\boldsymbol{x}_1^2, \boldsymbol{x}_1\boldsymbol{x}_2, \boldsymbol{x}_1\boldsymbol{u}, \boldsymbol{x}_2^2, \boldsymbol{x}_2\boldsymbol{u}, \boldsymbol{u}^2]^{\mathrm{T}},$$

$$\boldsymbol{\sigma}_v(\boldsymbol{x}) = [\boldsymbol{x}_1, \boldsymbol{x}_2]^{\mathrm{T}}.$$

To begin with, for each subsystem, 500 transitions are randomly sampled from the state and action spaces. During the iteration process, a batch of 300 samples are randomly selected from the data buffer to train the networks. The maximum iteration number is 100 and the training process will be completed if $\|\boldsymbol{W}_{c,v}^{(j+1)}-\boldsymbol{W}_{c,v}^{(j)}\|_2 \leqslant 10^{-6}, \forall v \in \{1,2\}$ is satisfied.

The evolution process of the critic network weights are shown in Fig. 1 and Fig. 2 which verifies the convergence proof. It is shown that the elements converge after five iteration steps. The training process takes 0.619 9 s.

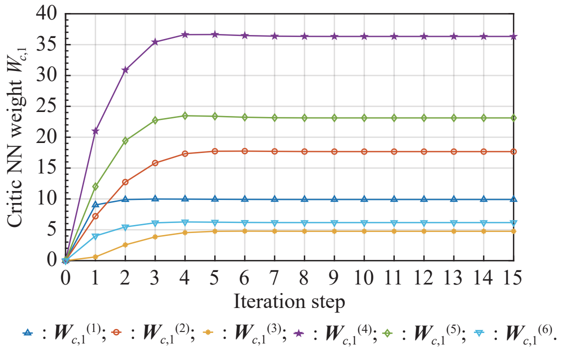

Fig. 1 Evolution process of critic network weight $\boldsymbol{W}_{c,1}$

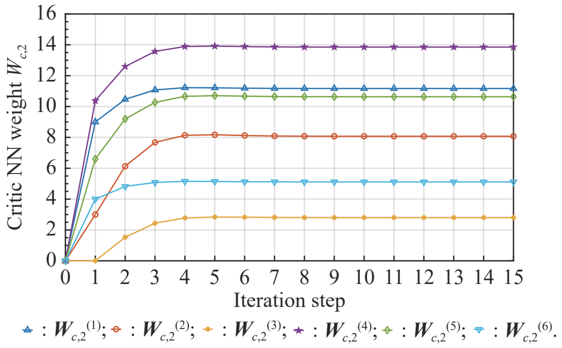

Fig. 2 Evolution process of critic network weight $\boldsymbol{W}_{c,2}$

Let the initial state be $\boldsymbol{x}_0=[5,-5]^{\mathrm{T}}$. The trajectories of system states and hybrid control input under the trained policy are given in Fig. 3–Fig. 5, respectively. The states converge to the origin after six time steps.

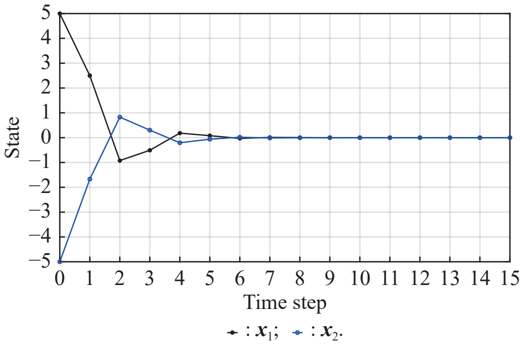

Fig. 3 Trajectory of system state with $\boldsymbol{x}_0$=[5, −5]$^{\mathrm{T}}$

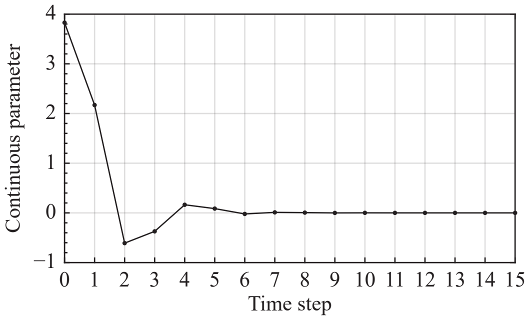

Fig. 4 Trajectory of continuous parameter with $\boldsymbol{x}_0$=[5, −5]$^{\mathrm{T}}$

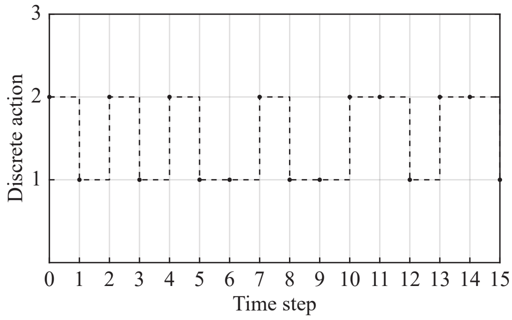

Fig. 5 Trajectory of discrete action with $\boldsymbol{x}_0$=[5, −5]$^{\mathrm{T}}$

(ii) Example 2

Next, a non-linear scalar system [34] is selected:

$$\begin{cases}\dot{\boldsymbol{x}}=f_1(\boldsymbol{x})+g_1(\boldsymbol{x})\boldsymbol{u}=-\boldsymbol{x}+\boldsymbol{u}, & v=1\\ \dot{\boldsymbol{x}}=f_2(\boldsymbol{x})+g_2(\boldsymbol{x})\boldsymbol{u}=-\boldsymbol{x}^3+\boldsymbol{u}, & v=2\end{cases}. \tag{37}$$

The domains of interest are selected as $\Omega_x=\{x\in\mathbf{R}: |x|\leqslant 3\}$ and $\Omega_u=\{u\in\mathbf{R}: |u|\leqslant 5\}$. In addition, the system is discretized by using Euler method with $\Delta t=0.005$ s. The cost function is defined as $U(\boldsymbol{x},v,\boldsymbol{u})=\boldsymbol{x}^2+\boldsymbol{u}^2$. The activation functions of the LIP NNs are

$$\boldsymbol{\phi}_v(\boldsymbol{x},\boldsymbol{u})=[\boldsymbol{u},\boldsymbol{x}\boldsymbol{u},\boldsymbol{x}^2\boldsymbol{u},\boldsymbol{x}^3\boldsymbol{u},\boldsymbol{x}^4\boldsymbol{u},\boldsymbol{x}^5\boldsymbol{u},\boldsymbol{u}^2,\boldsymbol{x},\boldsymbol{x}^2,\boldsymbol{x}^3,\boldsymbol{x}^4,\boldsymbol{x}^5,\boldsymbol{x}^6]^{\mathrm{T}},$$

$$\boldsymbol{\sigma}_v(\boldsymbol{x})=[1,\boldsymbol{x},\boldsymbol{x}^2,\boldsymbol{x}^3,\boldsymbol{x}^4,\boldsymbol{x}^5]^{\mathrm{T}}.$$

To begin with, for each subsystem, 300 transitions are randomly sampled from the state and action spaces. During the iteration process, a batch of 200 samples are randomly selected from the data buffer to train the networks. The maximum iteration number is 100 and the training process will be completed if $\|\boldsymbol{W}_{c,v}^{(j+1)}-\boldsymbol{W}_{c,v}^{(j)}\|_2 \leqslant 10^{-6}, \forall v \in \{1,2\}$ is satisfied.

The evolution process of the critic network weights are shown in Fig. 6 and Fig. 7 which verifies the convergence proof. It is shown that the elements converge after 150 iteration steps. The training process takes 1.729 1 s.

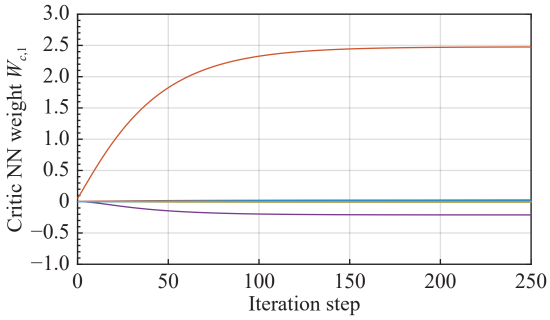

Fig. 6 Evolution process of critic network weight $W_{c,1}$

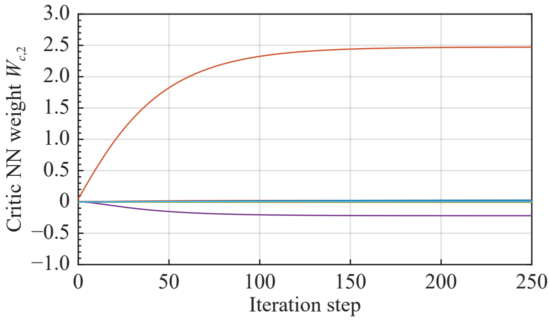

Fig. 7 Evolution process of critic network weights $W_{c,2}$

Let the initial state be $\boldsymbol{x}_0 = 3$ and apply the trained hybrid controller for 2.5 s. The trajectory of system state is shown in Fig. 8−Fig. 10 show the trajectories of control input and switching signal, respectively.

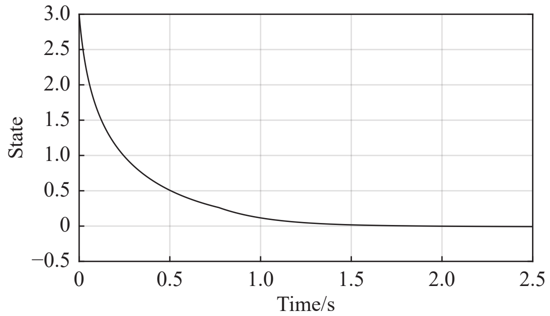

Fig. 8 Trajectory of system state with x_0=3

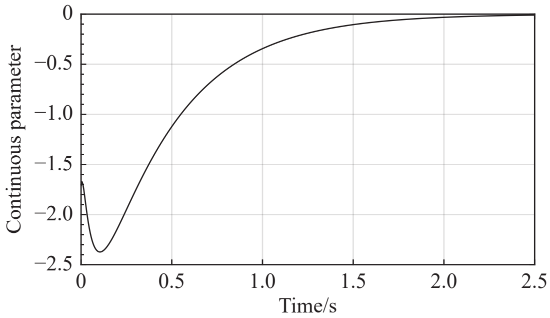

Fig. 9 Trajectory of continuous parameter with x_0=3

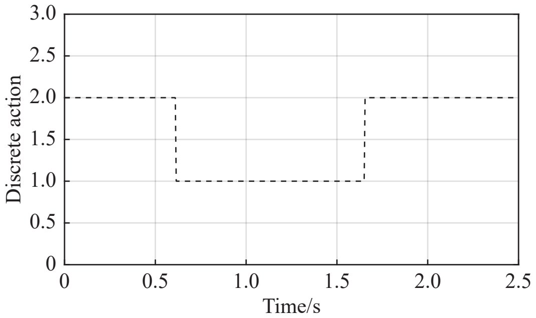

Fig. 10 Trajectory of discrete action with x_0=3

6. Conclusions

In this paper, a novel hybrid reinforcement learning with the AC structure is designed to find the co-designed optimal policy of the controlled switching system. The generated Q-functions will iteratively converge to the optimal solution of the derived HJB equation without knowing or identifying the system dynamics. The effectiveness of our algorithm is verified by two numerical examples.

References

[1] LIBERZON D. Switching in systems and control. Boston: Birkhauser, 2003.

[2] TANWANI A, SHIM H, LIBERZON D. Observability for switched linear systems: characterization and observer design. IEEE Trans. on Automatic Control, 2013, 58(4): 891–904.

[3] RINEHART M, DAHLEH M, REED D, et al. Suboptimal control of switched systems with an application to the disc engine. IEEE Trans. on Control Systems Technology, 2008, 16(2): 189–201.

[4] KOUVELAS A, ABOUDOLAS K, PAPAGEORGIOU M, et al. A hybrid strategy for real-time traffic signal control of urban road networks. IEEE Trans. on Intelligent Transportation Systems, 2011, 12(3): 884–894.

[5] BRYSON A E. Optimal control−1950 to 1985. IEEE Control System Magazine, 1996, 16(3): 26–33.

[6] LIU D R, XUE S, ZHAO B, et al. Adaptive dynamic programming for control: a survey and recent advances. IEEE Trans. on System, Man, and Cybernetics: System, 2021, 51(1): 142–160.

[7] SOLER M, OLIVARES A, STAFFETTI E, et al. Framework for aircraft trajectory planning toward an efficient air traffic management. Journal of Aircraft, 2012, 49(1): 341–348.

[8] GANS N R, HUTCHINSON S A. Stable visual servoing through hybrid switched-system control. IEEE Trans. on Robotics, 2007, 23(3): 530–540.

[9] LI X F, DONG L, XUE L, et al. Hybrid reinforcement learning for optimal control of non-linear switching system. IEEE Trans. on Neural Networks and Learning Systems, 2022.

[10] SARGENT R. Optimal control. Journal of Computational and Applied Mathematics, 2000, 124(1): 361–371.

[11] AXELSSON H, EGERSTEDT M, WARDI Y, et al. Algorithm for switching-time optimization in hybrid dynamical systems. Proc. of the IEEE International Conference on Control and Automation Intelligent Control, 2005: 256–261.

[12] EGERSTEDT M, WARDI Y, AXELSSON H. Transition-time optimization for switched-mode dynamical systems. IEEE Trans. on Automatic Control, 2006, 51(1): 110–115.

[13] LI S T, LIU X, TAN Y, et al. Optimal switching time control of discrete-time switched autonomous systems. International Journal of Innovative Computing, Information and Control, 2015, 11(6): 2043–2050.

[14] LUUS R, CHEN Y. Optimal switching control via direct search optimization. Proc. of the IEEE International Symposium on Intelligent Control, 2003: 371–376.

[15] XU X P, ANTSAKLIS P J. Optimal control of switched systems based on parameterization of the switching instants. IEEE Trans. on Automatic Control, 2004, 49(1): 2–16.

[16] SAKLY M, SAKLY A, MAJDOUB N, et al. Optimization of switching instants for optimal control of linear switched systems based on genetic algorithms. IFAC Proceedings Volumes, 2009, 42(19): 249–253.

[17] LONG R, FU J M, ZHANG L Y. Optimal control of switched system based on neural network optimization. Proc. of the International Conference on Intelligent Computing, 2008: 799–806.

[18] RUNGGER M, STURSBERG O. A numerical method for hybrid optimal control based on dynamic programming. Nonlinear Analysis: Hybrid Systems, 2011, 5(2): 254–274.

[19] SUTTON R S, BARTO A G. Reinforcement Learning: an introduction. Cambridge: MIT Press, 2018.

[20] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning. Nature, 2015, 518(7540): 529–533.

[21] SILVER D, HUBERT T, SCHRITTWIESER J, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. Science , 2018, 362(6419): 1140–1144.

[22] BERTSEKAS D P. Neuro-dynamic programming. Belmont: Athena Scientific, 1996.

[23] LEWIS F L, VRABIE D. Reinforcement learning and adaptive dynamic programming for feedback control. IEEE Circuits and Systems Magazine, 2009, 9(3): 32–50.

[24] SI J, WANG Y T. Online learning control by association and reinforcement. IEEE Trans. on Neural networks, 2001, 12(2): 264–276.

[25] LI X F, DONG L, SUN C Y. Data-based optimal tracking of autonomous nonlinear switching systems. IEEE/CAA Journal of Automatica Sinica, 2021, 8(1): 227–238.

[26] AL-TAMIMI A, LEWIS F L, ABU-KHALAF M. Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. IEEE Trans. on Systems, Man, and Cybernetics, Part B (Cybernetics), 2008, 38(4): 943–949.

[27] MU C X, WANG D, HE H B. Novel iterative neural dynamic programming for data-based approximate optimal control design. Automatica, 2017, 81: 240–252.

[28] LUO B, WU H N, HUANG T W, et al. Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design. Automatica , 2014, 50(12): 3281–3290.

[29] ZHANG H G, SONG R Z, WEI Q L, et al. Optimal tracking control for a class of nonlinear discrete-time systems with time delays based on heuristic dynamic programming. IEEE Trans. on Neural Networks, 2011, 22(12): 1851–1862.

[30] ZHANG H G, LUO Y H, LIU D R. Neural-network-based near optimal control for a class of discrete-time affine nonlinear systems with control constraints. IEEE Trans. on Neural Networks, 2009, 20(9): 1490–1503.

[31] DONG L, ZHONG X N, SUN C Y, et al. Adaptive event-triggered control based on heuristic dynamic programming for nonlinear discrete-time systems. IEEE Trans. on Neural Networks and Learning Systems, 2016, 28(7): 1594–1605.

[32] HEYDARI A. Optimal switching of DC-DC power converters using approximate dynamic programming. IEEE Trans. on Neural Networks and Learning Systems, 2016, 29(3): 586–596.

[33] HEYDARI A. Optimal switching with minimum dwell time constraint. Journal of the Franklin Institute, 2017, 354(11): 4498–4518.

[34] LIU D R, WANG D, ZHAO D B. Neural-network based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming. IEEE Trans. on Automation Science and Engineering, 2012, 9(3): 628–634.

[35] ZHANG H G, QIN C, LUO Y H. Neural-network-based constrained optimal control scheme for discrete-time switched nonlinear system using dual heuristic programming. IEEE Trans. on Automation Science and Engineering, 2014, 11(3): 839–849.

[36] MU C X, LIAO K, REN L, et al. Approximately optimal control of discrete-time nonlinear switched systems using globalized dual heuristic programming. Neural Processing Letters, 2020, 52(2): 1089–1108.

[37] GU S X, LILLICRAP T, SUTSKEVER I, et al. Continuous deep Q-learning with model-based acceleration. Proc. of the International Conference on Machine Learning, 2016: 2829–2838.

[38] LEWIS F L, VRABIE D, SYRMOS V L. Optimal control. New Jersey: John Wiley & Sons, 2012.

Biographies

LI Xiaofeng was born in 1990. He received his B.S. degree and M.S. degree in engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2012 and 2016, respectively, and his Ph.D. degree in control science and engineering from Southeast University, Nanjing, China, in 2021. He is working as a postdoctoral researcher with the School of Artificial Intelligence, Anhui University, Heifei, China. He was a joint Ph.D. student with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA, from 2018 to 2019. His current research interests include reinforcement learning, adaptive dynamic programming, robot system, and optimal control.

E-mail: 230169413@seu.edu.cn

DONG Lu was born in 1990. She received her B.S. degree in physics and Ph.D. degree in electrical engineering from Southeast University, Nanjing, China in 2012 and 2017, respectively. She is currently an associate professor with the School of Cyber Science and Engineering, Southeast University, Nanjing, China. Her current research interests include adaptive dynamic programming, event-triggered control, nonlinear system control, and optimization.

E-mail: ldong90@seu.edu.cn

SUN Changyin was born in 1975. He received his B.S. degree in applied mathematics from the College of Mathematics, Sichuan University, Chengdu, China, in 1996, and M.S. and Ph.D. degrees in electrical engineering from Southeast University, Nanjing, China, in 2001 and 2004, respectively. He is currently a professor with the School of Automation, Southeast University, Nanjing, China. His current research interests include intelligent control, flight control, and optimal theory.
E-mail: cysun@seu.edu.cn