7th International Conference on Advanced Computational Intelligence
Mount Wuyi, Fujian, China; March 27-29, 2015

Neural-Network-Based Finite Horizon Optimal Control for Partially Unknown Linear Continuous-Time Systems

Chao Li, Hongliang Li, and Derong Liu

Abstract— **In this paper, we establish a neural-network-based online learning algorithm to solve the finite horizon linear quadratic regulator (FHLQR) problem for partially unknown continuous-time systems. To solve the FHLQR problem with partially unknown system dynamics, we develop a time-varying Riccati equation. A critic neural network is used to approximate the value function and the online learning algorithm is established using the policy iteration technique to solve the time-varying Riccati equation. An integral policy iteration method and a tuning law are used when the algorithm is implemented without the knowledge of the system drift dynamics. We give a simulation example to show the effectiveness of this algorithm.**

I. Introduction

THE purpose of the optimal regulator is to obtain an optimal control law that minimizes the value function and moves the system states to the origin. The objective in finite horizon controller design is to seek a control law which satisfies the system demands over a specified time interval. In the field of optimal control theory [1], [2], the finite horizon liner quadratic regulator (FHLQR) is an important problem. The FHLQR problem tries to find a control law that not only minimizes a predefined value function, but also moves the states to the origin and satisfies a final condition constraint over a specified time interval. The standard solution of the optimal control law to the FHLQR problem can be obtained by solving a differential equation backward using the exact system dynamics and boundary conditions. This procedure is a kind of backward-time schemes which are not practical for real-time control and generally offline methods which require the system dynamics completely. An ideal FHLQR optimal control law using forward-in-time control design and partial knowledge of the system dynamics can overcome this weakness.

Dynamic programming (DP) [3] provides a principled method for determining optimal control policies for dynamic systems. Due to the nature of exhaustive search, DP is often computationally and it also requires the accurate system representation. Among the methods of solving the optimal control problem, adaptive dynamic programming (ADP) has received increasing attention owing to its learning and optimal capabilities [4]–[15]. Reinforcement learning (RL) is another computational method and it can interactively find an optimal policy [18]–[21]. The ADP and RL schemes relax the need for a complete and accurate model of the process to be controlled in DP by using compact parameterized function representations whose parameters are adjusted through adaption. In the existing literatures of ADP-based optimal control, either policy iteration (PI) or value iteration is utilized to solve the Bellman equation or the Hamilton-Jcaobi-Bellman equation. Liu et al. [22] extended the PI algorithm to nonlinear optimal control problem with unknown dynamics and discounted cost function. Wang et al. [23] investigated a neural-network-based robust optimal control design for a class of uncertain nonlinear systems via ADP approach. Wang et al. established a novel strategy to design a robust controller for a class of continuous-time nonlinear systems with uncertainties in [24]. Vrabie and Lewis [25] derived an integral RL method to obtain direct adaptive optimal control for nonlinear input-affine continuous-time systems with partially unknown dynamics. Jiang and Jiang [26] presented a novel PI approach for continuous-time linear systems with completely unknown dynamics. Lee et al. [27], [28] presented an integral Q-learning algorithm for continuous-time systems without the exact knowledge of the system dynamics.

Although ADP-based and RL-based algorithms are widely used to solve the infinite horizon optimal regulator problem, there are few results about the FHLQR problem. The FHLQR problem is more challenging since the solution is time-varying and a terminal constraint has to be satisfied. The novelty of this paper is that we establish an online learning algorithm to solve the FHLQR problem with partially unknown system dynamics. To obtain the optimal control law with partially unknown system dynamics, we develop a time-varying Riccati equation. Using the PI technique, we establish an online learning algorithm to solve the time-varying Riccati equation. To implement this algorithm, a critic neural network (NN) is used to approximate the value function. An integral PI method and a tuning law are implemented to obtain the optimal control policy. The effectiveness of the optimal control law is demonstrated by a simulation example.

The rest of this paper is organized as follows. In Section II, we present the FHLQR problem and its standard solution. In Section III, we establish an online learning algorithm using PI and NN to obtain the solution of the time-varying Riccati equation with partially unknown system dynamics. In Section IV, a simulation example is provided to illustrate the effectiveness of the derived optimal control law. In Section V, we conclude the paper with a few remarks.

The authors are with The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (email: {lichao2012, hongliang.li, derong.liu}@ia.ac.cn).

This work was supported in part by the National Natural Science Foundation of China under Grants 61034002, 61233001, 61273140, and 61304086.

978-1-4799-7259-3/15/$31.00 ©2015 IEEE

II. Problem Formulation

Consider the linear time-invariant continuous-time system

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the system state vector, $u(t) \in \mathbb{R}^r$ is the control input vector, and the matrices A and B have appropriate dimensionalities.

The objective of the FHLQR problem is to find the optimal control policy $u^*(t)$ to control the system (1) in such a way that the state $x(t)$ goes to the origin as close as possible during the interval $[t_0, t_f]$ with minimum expenditure of control effort. For this, we choose the value function as

$$V(t) = \frac{1}{2}x^\mathsf{T}(t_f)Fx(t_f) + \frac{1}{2}\int_t^{t_f} \left[x^\mathsf{T}(\tau)Qx(\tau) + u^\mathsf{T}(\tau)Ru(\tau)\right] \mathrm{d}\tau$$

where $t \in [t_0, t_f]$, t_f is the fixed final time, $F \in \mathbb{R}^{n\times n}$, $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{r\times r}$ are positive definite symmetric matrices, and $x^\mathsf{T}(t)Qx(t) + u^\mathsf{T}(t)Ru(t)$ is the utility function.

The standard solution of the optimal control $u^*(t)$ to the FHLQR problem is given as [1]

$$u^*(t) = -R^{-1}B^\mathsf{T}P(t)x^*(t) \tag{2}$$

where $x^*(t)$ is the optimal system state, $P(t)$ can be obtained by solving the matrix differential Riccati equation (DRE),

$$\dot{P}(t) = -P(t)A - A^\mathsf{T}P(t) + P(t)BR^{-1}B^\mathsf{T}P(t) - Q \tag{3}$$

with the final condition $P(t_f) = F$.

Using the optimal state, the optimal value function $V^*(t)$ can be represented as

$$V^*(t) = \frac{1}{2}x^{*\mathsf{T}}(t)P(t)x^*(t).$$

By solving the differential equation (3) backward using the boundary condition, we can obtain the standard solution of the FHLQR problem. Once the system dynamics and the value function are specified, we can independently compute $P(t)$ before the system operates in the forward direction from its initial condition.

***Remark** 1:* The feedback part of the control input is calculated in a backward-time manner which is not practical for real-time control. The standard solution described in this section is a kind of offline methods which require the system dynamics completely. To obtain the time-varying control input online with partial knowledge of the system dynamics, we establish an online learning algorithm.

***Remark** 2:* We do not need the controllability condition on the system for solving the optimal feedback control. As long as we deal with a finite time system, the contribution of those uncontrollable states to the value function is still a finite quantity.

III. The Online Learning Algorithm and Its Implementation

In this section, we establish a NN-based online learning algorithm to obtain the solution of the FHLQR problem with partially unknown system dynamics. Compared with the infinite horizon problem, a time-varying Riccati equation is developed. The online algorithm consists of an online integral PI method and an online tuning law for different time intervals of the time-varying Riccati equation.

For the system (1), we consider a value function with infinite horizon

$$\Lambda(t) = \frac{1}{2}\int_t^{\infty} [x^\mathsf{T}(\tau)Qx(\tau) + u^\mathsf{T}(\tau)Ru(\tau)]\mathrm{d}\tau. \tag{4}$$

According to the optimal theory [1], the optimal control with respect to this value function is given by

$$\mu^*(t) = -R^{-1}B^\mathsf{T}\bar{P}x^*(t) \tag{5}$$

where $\bar{P} \in \mathbb{R}^{n\times n}$ is a constant positive definite symmetric matrix. $\bar{P}$ is the solution of the nonlinear matrix algebraic Riccati equation (ARE)

$$\bar{P}A + A^\mathsf{T}\bar{P} + Q - \bar{P}BR^{-1}B^\mathsf{T}\bar{P} = 0. \tag{6}$$

Using the constant matrix $\bar{P}$, the value function can be represented in a quadratic form as

$$\Lambda(t) = \frac{1}{2}x^\mathsf{T}(t)\bar{P}x(t). \tag{7}$$

Now we consider the relationship between the solution of the matrix DRE (3) and the solution of the matrix ARE (6). We make a simple time transformation $\tau = t_f - t$. Then, in τ scale we can consider the final time t_f as the "starting time", $P(t_f)$ as the "initial condition", and $\bar{P}$ as the "steady-state solution" of the matrix DRE. As $t_f \to \infty$, the "transient solution" is pushed to near t_f which is at infinity. Then for the begining time interval, the matrix $P(t)$ becomes a steady state, i.e., a constant matrix $\bar{P}$ which is the solution of the ARE (6), as shown in Fig. 1.

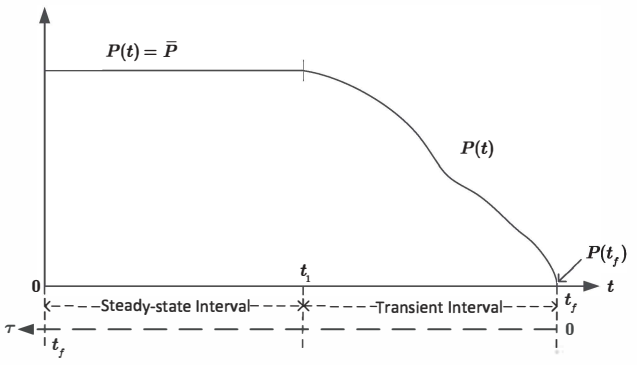

Fig. 1. Interpretation of the Constant Matrix $\bar{P}$.

According to Fig. 1, the matrix $P(t)$ in DRE becomes the constant matrix $\bar{P}$ during the steady-state interval. We give

the following time-varying Riccati equation to solve $P(t)$ during the interval $[t_0, t_f]$,

$$P(t) = \begin{cases} \bar{P}, & t \in [t_0, t_1) \\ P(t), & t \in [t_1, t_f] \end{cases}$$

where t_1 is the terminal time of the steady-state interval. We will establish the online integral PI method to calculate the steady-state interval solution $\bar{P}$ and the online tuning law to solve the transient interval solution $P(t)$.

A. Steady-State Interval Solution

In this subsection, we discuss the online integral PI method during the steady-state interval and its implementation. $\bar{P}$ is the solution of the ARE (6). To obviate the need for the complete knowledge of the system dynamics, the Integral Reinforcement Learning (IRL) algorithm [25] can be used to solve for ARE. The IRL is a PI method which uses an equivalent formulation of the Lyapunov equation that does not involve the system dynamics. Hence, it is central to the development of the online integral PI method for continuous-time systems. To obtain the IRL Bellman equation, note that for time interval $\Delta t > 0$, the value function satisfies

$$\begin{aligned} \Lambda(t) = \Lambda(t + \Delta t) \\ + \frac{1}{2} \int_t^{t+\Delta t} [x^\mathsf{T}(\tau) Q x(\tau) + u^\mathsf{T}(\tau) R u(\tau)] \mathrm{d}\tau. \end{aligned}$$

The representation (7) yields the IRL Bellman equation

$$\begin{aligned} & x(t)^\mathsf{T} \bar{P} x(t) - x^\mathsf{T}(t + \Delta t) \bar{P} x(t + \Delta t) \\ & = \int_t^{t+\Delta t} [x^\mathsf{T}(\tau) Q x(\tau) + u^\mathsf{T}(\tau) R u(\tau)] \mathrm{d}\tau. \end{aligned} \tag{8}$$

The last term of (8) is known as the integral reinforcement.

Equation (8) which is derived from (4) and (5) plays an important role in relaxing the assumption of knowing the system dynamics, since A is not appear in the equation. It means that the algorithm can be implemented without knowing the system dynamic A, but the knowledge of B is still required.

***Remark** 3:* We solve the linear quadratic problem over infinite horizon to obtain the "steady-state solution" $\bar{P}$ in this subsection. The admissibility of control is required to guarantee the existence of $\bar{P}$. So an admissible control is needed to implement this online integral PI method.

We will discuss the NN-based implementation of the established online integral PI method. A critic NN is used to approximate the value function. We assume that for the system, $\Lambda(t)$ is represented on a compact set Ω by single-layer NN as

$$\Lambda(t) = \frac{1}{2} x(t)^\mathsf{T} \bar{P} x(t) = \frac{1}{2} p^\mathsf{T} \chi(t)$$

where

$$\begin{aligned} p^\mathsf{T} &= [p_{11}, p_{12}, \ldots, p_{1n}, p_{22}, p_{23}, \ldots, p_{n-1,n}, p_{nn}], \\ \chi^\mathsf{T}(t) &= [x_1^2, 2x_1x_2, \ldots, 2x_1x_n, x_2^2, \ldots, 2x_{n-1}x_n, x_n^2]. \end{aligned}$$

p_{ij} is the i-row j-column element of $\bar{P}$, $p \in \mathbb{R}^{\frac{n(n+1)}{2}}$ is unknown bounded ideal weight parameters which will be determined by the established integral PI method, and $\chi(t) \in \mathbb{R}^{\frac{n(n+1)}{2}}$ is the continuously differentiable activation functions. Since the ideal weights are unknown, the outputs of the critic NN is

$$\Lambda^{(i)}(t) = \frac{1}{2} (\hat{p}^i)^\mathsf{T} \chi(t) = \Lambda(t) - \varepsilon^i \tag{9}$$

where $\hat{p}^i$ is the current estimated weight vector and $\varepsilon^i \in \mathbb{R}$ is the bounded NN approximation errors.

Using the expression (9), the IRL Bellman equation (8) can be rewritten in a general form

$$\psi_k^\mathsf{T} \hat{p}^i = \theta_k \tag{10}$$

with

$$\begin{aligned} \theta_k &= \int_{t+(k-1)\Delta t}^{t+k\Delta t} \left[x^\mathsf{T}(\tau) Q x(\tau) + u^{(i)\mathsf{T}}(\tau) R u^{(i)}(\tau) \right] \mathrm{d}\tau \\ \psi_k &= \chi(t + (k-1)\Delta t) - \chi(t + k\Delta t) \end{aligned}$$

where the measurement time is from $t+(k-1)\Delta t$ to $t+k\Delta t$, Δt is the time interval. Since equation (10) is only a one-dimensional equation, we cannot guarantee the uniqueness of the solution. Similar to [27], we use the least-square-based method to solve the parameter vector over a compact set Ω. For any positive integral K, we denote $\Phi = [\psi_1, \psi_2, \ldots, \psi_K]$ and $\Theta = [\theta_1, \theta_2, \ldots, \theta_K]^\mathsf{T}$. Then, we have the following K-dimensional equation

$$\Phi^\mathsf{T} \hat{p}^i = \Theta.$$

If Φ^T has full column rank, the weight parameters can be solved by

$$\hat{p}^i = (\Phi \Phi^\mathsf{T})^{-1} \Phi \Theta. \tag{11}$$

Therefore, we need to guarantee that the number of collected points K satisfies $K \geq \mathrm{rank}(\Phi) = \frac{n(n+1)}{2}$, which will make $(\Phi\Phi^\mathsf{T})^{-1}$ exist. The least squares problem in (11) can be solved in real time by collecting enough data points generated from the system.

B. Transient Interval Solution

In this subsection, we will derive an online tuning law to obtain the solution $P(t)$ of the DRE with the final condition $P(t_f) = F$ during the time interval $[t_1, t_f]$. We assume that the value function $V(t)$ is represented by single-layer NN as

$$V(t) = \frac{1}{2} x(t)^\mathsf{T} P(t) x(t) = \frac{1}{2} p^\mathsf{T}(t) \chi(t).$$

We define the ideal time-varying weights of the critic network as

$$p^\mathsf{T}(t) = [p_{11}, p_{12}, \ldots, p_{1n}, p_{22}, p_{23}, \ldots, p_{n-1,n}, p_{nn}]$$

where we omit the time t in the elements of $P(t)$.

When we consider the time-varying function $P(t)$ for the Bellman equation (8), there is a residual error caused by the estimated value function. We assume that $P(t)$ is a constant

matrix during the time interval $[t, t+\Delta t]$. The residual error can be expressed as

$$\begin{aligned} e_1(t) =& x^\mathsf{T}(t+\Delta t)P(t)x(t+\Delta t) - x(t)^\mathsf{T}P(t)x(t) \\ &+\int_t^{t+\Delta t} [x^\mathsf{T}(\tau)Qx(\tau) + u^\mathsf{T}(\tau)Ru(\tau)]\mathrm{d}\tau. \end{aligned}$$

By defining the expressions

$$\begin{aligned} \theta(t) =& \int_t^{t+\Delta t} \left[x^\mathsf{T}(\tau)Qx(\tau) + u^\mathsf{T}(\tau)Ru(\tau)\right] \mathrm{d}\tau \\ \psi(t) =& \chi(t) - \chi(t+\Delta t), \end{aligned}$$

the residual error $e_1(t)$ can be rewritten as

$$e_1(t) = \theta(t) - \psi(t)^\mathsf{T}\hat{p}(t).$$

Next, the terminal constraint $P(t_f) = F$ need to be satisfied. The constraint error is given as

$$e_2(t) = f - \hat{p}(t)$$

where f is defined as

$$f^\mathsf{T} = [f_{11}, f_{12}, \ldots, f_{1n}, f_{22}, f_{23}, \ldots, f_{n-1,n}, f_{nn}]$$

where f_{ij} is the element of the terminal constraint matrix. In order to minimize both the residual error and the constraint error, we give the following online parameters tuning law

$$\hat{p}(t+\Delta t) = \hat{p}(t) + \alpha\frac{\psi(t)e_1(t)}{\psi^\mathsf{T}(t)\psi(t)+1} + \alpha\frac{e_2(t)}{(1+t_f-t)^c} \quad (12)$$

where the learning rate α satisfies $0 < \alpha < 1$, c is a predefined constant.

Theorem 1: The parameters update law of the value function is given by (12). Within the finite time interval $[t_1, t_f]$, there exists a positive constant learning rate $0 < \alpha < 1$ such that the value function parameter estimation error is bounded.

Proof: We consider the following definite Lyapunov function candidate given as

$$\Pi(t) = \tilde{p}^\mathsf{T}(t)\tilde{p}(t)$$

where $\tilde{p}(t) = p(t) - \hat{p}(t)$. Using this expression, we have

$$\begin{aligned} e_1(t) &= \psi^\mathsf{T}(t)p(t) - \psi^\mathsf{T}(t)\hat{p}(t) = \psi^\mathsf{T}(t)\tilde{p}(t), \\ e_2(t) &= f - [p(t) - \tilde{p}(t)] = f - p(t) + \tilde{p}(t). \end{aligned}$$

We define $\tilde{p}(t+\Delta t) = p(t) - \hat{p}(t+\Delta t)$ and have

$$\begin{aligned} \tilde{p}(t+\Delta t) &= \tilde{p}(t) + \hat{p}(t) - \hat{p}(t+\Delta t) \\ &= \tilde{p}(t) - \alpha\frac{\psi(t)e_1(t)}{\psi^\mathsf{T}(t)\psi(t)+1} - \alpha\frac{e_2(t)}{(1+t_f-t)^c}. \end{aligned}$$

Then using online parameters tuning law (12), first difference of $\Pi(t)$ can be derived as

$$\begin{aligned} &\Delta\Pi(t) = \tilde{p}^\mathsf{T}(t+\Delta t)\tilde{p}(t+\Delta t) - \tilde{p}^\mathsf{T}(t)\tilde{p}(t) \\ &= \tilde{p}^\mathsf{T}(t)\tilde{p}(t) - 2\alpha\frac{\tilde{p}^\mathsf{T}(t)\psi(t)e_1(t)}{\psi^\mathsf{T}(t)\psi(t)+1} - 2\alpha\frac{\tilde{p}^\mathsf{T}(t)e_2(t)}{(1+t_f-t)^c} \\ &\quad + \alpha^2\frac{\psi^\mathsf{T}(t)\psi(t)e_1^2(t)}{[\psi^\mathsf{T}(t)\psi(t)+1]^2} + \alpha^2\frac{e_2^\mathsf{T}(t)e_2(t)}{(1+t_f-t)^{2c}} \\ &\quad + 2\alpha^2\frac{\psi^\mathsf{T}(t)e_1(t)e_2(t)}{[\psi^\mathsf{T}(t)\psi(t)+1](1+t_f-t)^c} - \tilde{p}^\mathsf{T}(t)\tilde{p}(t) \\ &\leq -\alpha(1-\alpha)\left[\frac{\psi^\mathsf{T}(t)\psi(t)}{\psi^\mathsf{T}(t)\psi(t)+1} + \frac{1}{(1+t_f-t)^c}\right]\tilde{p}^\mathsf{T}(t)\tilde{p}(t) \\ &\quad + 2\alpha^2\frac{\|f - p(t) + \tilde{p}(t)\|^2}{(1+t_f-t)^c} \\ &\leq -\alpha(1-\alpha)\left[\frac{\Psi}{\Psi+1} + \Xi\right]\tilde{p}^\mathsf{T}(t)\tilde{p}(t) + \Upsilon \end{aligned}$$

where $\Psi = \min_{t\in[t_1,t_f]}[\psi^\mathsf{T}(t)\psi(t)]$, $\Xi = \frac{1}{(1+t_f-t_1)^c}$. Since learning rate α is selected as $0 < \alpha < 1$, the first term of $\Delta\Pi(t)$ is negative, the second term $\Upsilon = 2\alpha^2\frac{\|f-p(t)+\tilde{p}(t)\|^2}{(1+t_f-t)^c}$ is bounded.Using standard Lyapunov theory, the value function parameter estimation error can be proven to be bounded with a bound which is dependent upon initial condition of the system and the fixed final time instant t_f.

Assume that the initial value function parameter estimation error is bounded such that $\|\tilde{p}(t_1)\|^2 \leq \Gamma_0$. According to standard Lyapunov theory, value function parameter estimation error at time t can be expressed as

$$\begin{aligned} \Pi(t) =& \Delta\Pi(t) + \Delta\Pi(t-\Delta t) + \cdots + \Delta\Pi(t_1) + \Pi(t_1) \\ =& \sum_{i=0}^{N_t-1} \Delta\Pi(t_1 + i\Delta t) + \Pi(t_1) \end{aligned}$$

where $N_t = \lceil\frac{t-t_1}{\Delta t}\rceil$, $\lceil x\rceil$ is the ceiling operation represents the smallest integer not less than x. Note that Δt is a small sampling interval. The bound for the value function parameter estimation error Γ_t can be expressed as

$$\begin{aligned} &\Gamma_t = \|\tilde{p}(t)\|^2 = \Pi(t) = \sum_{i=0}^{N_t-1} \Delta\Pi(t_1 + i\Delta t) + \Pi(t_1) \\ &\leq \sum_{i=0}^{N_t-1} [-\beta(1-\beta)^i\Pi(t_1)] + \sum_{i=1}^{N_t-1} [\beta(1-\beta)^{i-1}\Upsilon] + \Pi(t_1) \\ &\leq (1-\beta)^{N_t}\Gamma_0 + [1-(1-\beta)^{N_t-1}]\Upsilon \end{aligned}$$

where $\beta = \alpha(1-\alpha)\left[\frac{\Psi}{\Psi+1} + \Xi\right]$, since $0 < \alpha < 1$, we know $0 < \beta < 1$. The value function estimation error Γ_t is dependent upon initial bound Γ_0 and Υ.

The proof is completed.

We have already obtained the Riccati coefficient matrix $P(t)$ during the interval $t \in [t_0, t_f]$ using the online integral PI method and the online tuning law. Then we will describe the online learning algorithm as Algorithm 1 which can be used to solve the FHLQR problem with partially unknown system dynamics.

Algorithm 1 Online Learning Algorithm

Part I: Steady-State Interval

1: Give a small positive real number ϵ. Let $i = 0$ and start with $\bar{P}^{(0)}$ which makes the control policy $u^{(0)}(t)$ is admissible.

2: **Policy Evaluation:**
Based on the Riccati coefficients $\bar{P}^{(i)}$ and control policy $u^{(i)}(t)$, solve the following Bellman equation for $\bar{P}^{(i+1)}$

$$x(t)^{\mathsf{T}}\bar{P}^{(i+1)}x(t) - x(t+\Delta t)^{\mathsf{T}}\bar{P}^{(i+1)}x(t+\Delta t) = \int_t^{t+\Delta t}[x^{\mathsf{T}}(\tau)Qx(\tau) + u^{(i)\mathsf{T}}(\tau)Ru^{(i)}(\tau)]\mathrm{d}\tau.$$

3: **Policy Improvement:**
Update the control policy using

$$u^{(i+1)}(t) = -R^{-1}B^{\mathsf{T}}\bar{P}^{(i+1)}x(t).$$

4: If $\|\bar{P}^{(i+1)} - \bar{P}^{(i)}\| \leq \epsilon$, set $t_1 = t$, obtain the steady-state solution, and go to Part II; else, set $i = i + 1$ and go to Step 2.

Part II: Transient Interval

1: Start with $\bar{P}$ when $t = t_1$.

2: **Policy Evaluation:**
Update $P(t + \Delta t)$ using

$$\hat{p}(t+\Delta t) = \hat{p}(t) + \alpha\frac{\psi(t)e_1(t)}{\psi^{\mathsf{T}}(t)\psi(t)+1} + \alpha\frac{e_2(t)}{(1+t_f-t)^c}.$$

3: **Policy Improvement:**
Update the control policy using

$$u(t+\Delta t) = -R^{-1}B^{\mathsf{T}}P(t+\Delta t)x(t+\Delta t).$$

4: Repeat Step 2 and Step 3 while $t < t_f$.

Remark 4: This algorithm is a kind of PI algorithms which consist of policy evaluation and policy improvement. For the two different time intervals, the policy evaluation is implemented using (8) and (12), the policy improvement is implemented using (2) where the knowledge of system dynamics B is required.

IV. Simulation

In this section, we provide a simulation example to demonstrate the effectiveness of the online learning algorithm. Compared with the standard solution, the algorithm derived in Section III is implemented online without the knowledge of A. We use this algorithm to obtain the feedback control law and plot all the time histories of optimal states and control.

We consider the following second order example to illustrate the linear quadratic regulator system. A second order plant

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -2 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

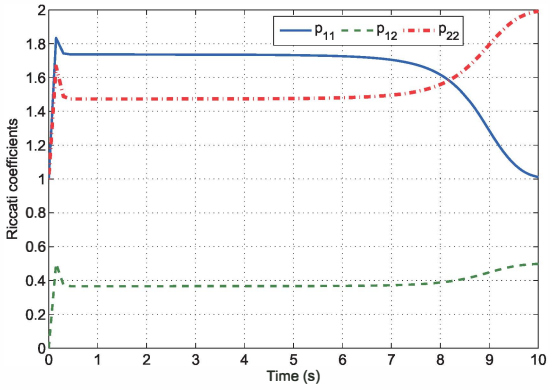

Fig. 2. Evolutions of the Riccati coefficients $P(t)$.

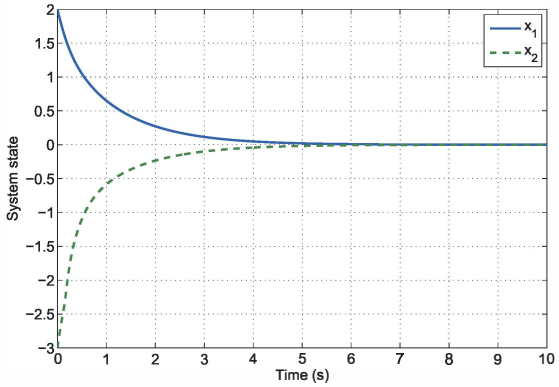

Fig. 3. Evolutions of the system state $x(t)$.

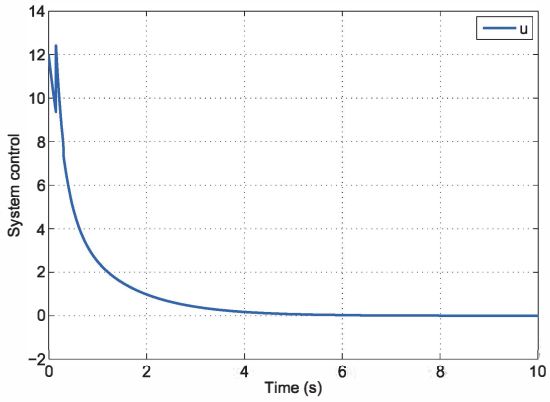

Fig. 4. Evolutions of the system control $u(t)$.

is to be controlled to minimize the value function with parameters

$$F = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix}, \quad Q = \begin{bmatrix} 2 & 3 \\ 3 & 5 \end{bmatrix}, \quad R = 0.25.$$

The initial condition $x(0) = [2, -3]^{\mathsf{T}}$. The final time t_f is specified at 10s and the final state $x(t_f)$ is free.

Now we assume that the system drift dynamics is unknown, that is, we can not use the knowledge of A when the online learning algorithm is applying. Algorithms 1 is

implemented online to solve the FHLQR problem. The 2×2 symmetric Riccati coefficient matrix $P(t)$ can be represented as

$$P(t) = \begin{bmatrix} p_{11}(t) & p_{12}(t) \\ p_{12}(t) & p_{22}(t) \end{bmatrix}.$$

The activation functions are chosen as

$$\chi^{\mathsf{T}}(t) = [x_1^2, 2x_1x_2, x_2^2].$$

The weight parameters of the critic NN can be represented as

$$p^{\mathsf{T}}(t) = [p_{11}(t), p_{12}(t), p_{22}(t)].$$

Using the online integral PI method, we solve the "steady-state solution" $\bar{P}$ of the matrix DRE. To implement this algorithm, we let the integral $K = 3$, the period time $\Delta t = 0.05$s and the initial weights as $p^{(0)\mathsf{T}} = [1,\ 0,\ 1]$. The least squares problem is solved after 3 samples are acquired, and the weights of the critic NN are updated every 0.15s. It is clear that the weights approximately converge to the steady ones after five updates at $t = 0.75$s in Fig. 2.

To implement the online parameters tuning law, we let the period time $\Delta t = 0.05$s, learning rate $\alpha = 0.6$ and the constant $c = 4$. We obtain the near optimal solution $P(t)$ of the matrix DRE during the time interval $[1.8, 10]$s. The system states and control are obtained at the same time interval. Figs. 2, 3 and 4 illustrate the evolutions of the Riccati coefficients, system states and optimal control with partial system dynamics. It is clear that using the derived algorithm the states $x_1(t)$ and $x_2(t)$ go to origin during the simulation.

V. Conclusion

A neural-network-based online learning algorithm was established using PI to solve the FHLQR problem for partially unknown linear time-invariant continuous-time systems. Compared with the infinite horizon problem, the time-varying Riccati equation was developed to obtain the optimal control with partially unknown system dynamics. The online learning algorithm consists of an online integral PI method and an online tuning law for different time intervals of the time-varying Riccati equation. A simulation example was given to show the efficiency of the proposed algorithm.

References

[1] F. L. Lewis, D. Vrabie, and V. Syrmos, *Optimal Control*, Third edition, Wiley, New Jersey, 2012.

[2] D. S. Naidu, *Optimal Control Systems*, CRC Press, New York, 2003.

[3] R. Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[4] F. Y. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: An introduction," *IEEE Computational Intelligence Magazine*, vol. 4, no. 2, pp. 39–47, May 2009.

[5] H. Zhang, Q. Wei, and D. Liu, "An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games," *Automatica*, vol. 47, no. 1, pp. 207–214, Jan. 2011.

[6] D. Liu and Q. Wei, "Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 779–789, Apr. 2013.

[7] F. Y. Wang, N. Jin, D. Liu, and Q. Wei, "Adaptive dynamic programming for finite horizon optimal control of discrete-time nonlinear systems with ϵ-error bound," *IEEE Transactions on Neural Networks*, vol. 22, no. 1, pp. 24–36, Jan. 2011.

[8] D. Liu, H. Javaherian, O. Kovalenko, and T. Huang, "Adaptive critic learning techniques for engine torque and air-fuel ratio control," *IEEE Transactions on Systems, Man, and CyberneticsCPart B: Cybernetics*, vol. 38, no. 4, pp. 988–993, Aug. 2008.

[9] Y. Jiang and Z. P. Jiang, "Robust adaptive dynamic programming for large-scale systems with an application to multimachine power systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 10, pp. 693–697, Oct. 2012.

[10] D. Zhao, Z. Zhang, and Y. Dai, "Self-teaching adaptive dynamic programming for gomoku," *Neurocomputing*, vol. 78, no. 1, pp. 23–29, Jan. 2012.

[11] Y. Jiang and Z. P. Jiang, "Robust adaptive dynamic programming with an application to power systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 7, pp. 1150–1156, July 2013.

[12] D. Wang and D. Liu, "Neuro-optimal control for a class of unknown nonlinear dynamic systems using SN-DHP technique," *Neurocomputing*, vol. 121, pp. 218–225, Dec. 2013.

[13] H. Li and D. Liu, "Optimal control for discrete-time affine nonlinear systems using general value iteration," *IET Control Theory & Applications*, vol. 6, no. 18, pp. 2725–2736, Dec. 2012.

[14] D. Liu, D. Wang, and X. Yang, "An iterative adaptive dynamic programming algorithm for optimal control of unknown discretetime nonlinear systems with constrained inputs," *Information Sciences*, vol. 220, pp. 331–342, Jan. 2013.

[15] D. Liu, H. Li, and D. Wang, "Online synchronous approximate optimal learning algorithm for multiplayer nonzero-sum games with unknown dynamics," *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 44, no. 8, pp. 1015–1027, Aug. 2014

[16] H. Li, D. Liu, D. Wang, and X. Yang, "Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 3, pp. 706–714, July 2014.

[17] D. Liu, D. Wang, and H. Li, "Decentralized stabilization for a class of continuous-time nonlinear interconnected systems using online learning optimal control approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 418–428, Feb. 2014

[18] 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, Mar. 2009.

[19] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, Cambridge Univ Press, 1998.

[20] J. Si, A. Barto, W. Powell, and D. Wunsch, *Handbook of Learning and Approximate Dynamic Programming*, Los Alamitos: IEEE Press, 2004.

[21] S. J. Bradtke, B. E. Ydstie, and A. G. Barto, "Adaptive linear quadratic control using policy iteration," in *Proceedings of American Control Conference*, Baltimore, Jun. 1994, pp. 3475–3479.

[22] D. Liu, X. Yang, and H. Li, "Adaptive optimal control for a class of continuous-time affine nonlinear systems with unknown internal dynamics," *Neural Computing and Applications*, vol. 23, no. 7–8, pp. 1843–1850, Dec. 2013.

[23] D. Wang, D. Liu, H. Li, and H. Ma, "Neural-network-based robust optimal control design for a class of uncertain nonlinear systems via adaptive dynamic programming," *Information Sciences*, vol. 282, pp. 167–179, Oct. 2014.

[24] D. Wang, D. Liu, and H. Li, "Policy iteration algorithm for online design of robust control of a class of continuous-time nonlinear systems," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 627–632, Apr. 2014.

[25] D. Vrabie and F. Lewis, "Neural network approach to continuoustime direct adaptive optimal control for partially unknown nonlinear systems," *Neural Networks*, vol. 22, no. 3, pp. 237–246, Mar. 2009.

[26] Y. Jiang and Z. P. Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics," *Automatica*, vol. 48, no. 10, pp. 2699–2704, Oct. 2012.

[27] J. Y. Lee, J. B. Park, and Y. H. Choi, "Integral Q-learning and explorized policy iteration for adaptive optimal control of continuous-time linear systems," *Automatica*, vol. 48, no. 11, pp. 2850–2859, Nov. 2012.

[28] J. Y. Lee, J. B. Park, and Y. H. Choi, "Integral reinforcement learning with explorations for continuous-time nonlinear systems," in *Proceedings of The 2012 International Joint Conference on Neural Networks*, Brisbane, QLD, June 2012, pp. 1042–1047.