Automatica 50 (2014) 1167–1175

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

# Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics☆

Bahare Kiumarsi[a,1], Frank L. Lewis[b], Hamidreza Modares[a], Ali Karimpour[a], Mohammad-Bagher Naghibi-Sistani[a]

[a] Department of Electrical Engineering, Ferdowsi University of Mashhad, Mashhad 91775-1111, Iran

[b] University of Texas at Arlington Research Institute, 7300 Jack Newell Blvd. S., Ft. Worth, TX 76118, USA

ARTICLE INFO

*Article history:*
Received 19 February 2013
Received in revised form
15 August 2013
Accepted 3 January 2014
Available online 22 February 2014

*Keywords:*
Linear quadratic tracker
Reinforcement learning
Policy iteration
Algebraic Riccati equation

ABSTRACT

In this paper, a novel approach based on the Q-learning algorithm is proposed to solve the infinite-horizon linear quadratic tracker (LQT) for unknown discrete-time systems in a causal manner. It is assumed that the reference trajectory is generated by a linear command generator system. An augmented system composed of the original system and the command generator is constructed and it is shown that the value function for the LQT is quadratic in terms of the state of the augmented system. Using the quadratic structure of the value function, a Bellman equation and an augmented algebraic Riccati equation (ARE) for solving the LQT are derived. In contrast to the standard solution of the LQT, which requires the solution of an ARE and a noncausal difference equation simultaneously, in the proposed method the optimal control input is obtained by only solving an augmented ARE. A Q-learning algorithm is developed to solve online the augmented ARE without any knowledge about the system dynamics or the command generator. Convergence to the optimal solution is shown. A simulation example is used to verify the effectiveness of the proposed control scheme.



## 1. Introduction

The linear quadratic tracker (LQT) has been a main method for control systems' design. The objective of the LQT is to design a controller in such a way that the output tracks a reference trajectory in an optimal sense by minimizing a predefined performance index. Traditional solutions to the LQT consist of two components; a feedback term obtained by solving an algebraic Riccati equation (ARE) and a feedforward term obtained by solving a noncausal difference equation (Lewis, Vrabie, & Syrmos, 2012). These solutions are normally performed offline, noncausal and require complete knowledge of the system dynamics.

Reinforcement learning (RL) (Barto, Powell, & Wunch, 2004; Bertsekas & Tsitsiklis, 1996; Powell, 2009; Sutton & Barto, 1998) as a class of machine learning methods has been widely used in several disciplines to find an optimal policy in an uncertain environment. In the control system society, RL techniques were first employed by Werbos (1989, 1991, 1992) to seek solutions to the optimal regulator problem for discrete-time systems. Later, considerable research was conducted for developing RL techniques to find optimal feedback solutions for both discrete-time and continuous-time systems. Moreover, RL methods have been used to find the solution to zero-sum game problems (Al-Tamimi, Lewis, & Abu-Khalaf, 2007; Zhang, Wei, & Liu, 2011). The interested reader is referred to Lewis and Liu (2013), Lewis, Vrabie, and Vamvoudakis (2012), Lewis, Vamvoudakis, and Vrabie (2013), Lewis and Vrabie (2009) and Zhang, Liu, Luo, and Wang (2012) and the references therein for details of the existing RL methods for solving optimal control and zero-sum game problems. Among the existing RL methods, the policy iteration (PI) technique (Bertsekas & Tsitsiklis, 1996; White & Sofge, 1992) has been widely used for designing feedback controllers. In particular, PI algorithms are used to solve the linear quadratic regulator (LQR) problem for both discrete-time systems (Al-Tamimi, Lewis, & Abu-Khalaf, 2008; Bradtke, Ydestie, & Barto, 1994; Lewis & Vamvoudakis, 2011) and continuous-time

☆ This work is supported by NSF grant ECCS-1128050, NSF grant IIS-1208623, ONR grant N00014-13-1-0562, AFOSR EOARD Grant #13-3055, China NNSF grant 61120106011, and China Education Ministry Project 111 (No. B08015). The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Shuzhi Sam Ge under the direction of Editor Miroslav Krstic.

*E-mail addresses:* b_kiomarsi@yahoo.com (B. Kiumarsi), lewis@uta.edu (F.L. Lewis), reza_modares@yahoo.com (H. Modares), a_karimpoure@yahoo.com (A. Karimpour), mb-naghibi@um.ac.ir (M.-B. Naghibi-Sistani).

[1] Tel.: +98 9153826014; fax: +98 5118172233.

http://dx.doi.org/10.1016/j.automatica.2014.02.015
0005-1098/

systems (Baird, 1994; Jiang & Jiang, 2012; Lee, Park, & Choi, 2012; Vrabie, Pastravanu, Abu-Khalaf, & Lewis, 2009). It is well known that solving the LQR requires solving an ARE. To find a solution to the ARE, the PI technique starts with an admissible control policy and then iteratively alternates between policy evaluation and policy improvement steps until there is no change in the value or the policy.

To avoid the requirement for knowledge of the system dynamics, in Bradtke et al. (1994), a PI algorithm is developed that converges to the optimal solution of the discrete-time LQR problem using $Q$-functions (Sutton & Barto, 1998; Watkins, 1989). $Q$-learning does not require any knowledge of the system dynamics. Although $Q$-learning has been effectively applied to the LQR, because of the additional complexity caused by computing the feedforward term in the LQT, RL techniques have not been used to solve the LQT. Note that although some RL-based algorithms are developed to find the solution to the optimal tracking problem for both nonlinear discrete-time systems (Wang, Liu, & Wei, 2012; Zhang, Wei, & Luo, 2008) and nonlinear continuous-time systems (Dierks & Jagannathan, 2010; Zhang, Cui, Zhang, & Luo, 2011), these methods employ the dynamic inversion concept to find the feedforward part of the control input a priori and they only use the RL to find the optimal feedback part of the control input. However, the dynamic inversion technique requires the control input matrix be invertible and complete knowledge of the system dynamics be known or identified a priori.

This paper develops an online model-free solution using a reinforcement $Q$-learning algorithm to the infinite-horizon LQT for discrete-time systems. Full state feedback is assumed available for control. It is assumed that the reference trajectory is generated by a linear command generator. Although the value function for the LQT is not quadratic in general, it is shown that for the given command generator and the reward function, the LQT value function is quadratic in the state of the system and the reference trajectories. The quadratic nature of the value function for the LQT allows development of a Bellman equation which uses only knowledge of the state of the system and the reference trajectories to find the value related to a control policy. Then, an augmented system composed of the original system dynamics and the command generator dynamics is formed. Based on this augmented system, an augmented ARE is derived whose solution yields the solution to the LQT. That is, once the augmented ARE is solved, both the feedback and feedforward terms of the control input are obtained simultaneously. Finally, a $Q$-learning algorithm is proposed to solve the LQT without requiring any knowledge of the augmented system dynamics. It is verified that starting from an admissible control policy, the proposed $Q$-learning algorithm converges to the optimal control solution.

This paper is organized as follows. Review of the standard solution for the LQT problem is given in Section 2. An alternative approach for formulating the infinite-horizon LQT in a causal manner is presented in Section 3. In Section 4 offline and online PI algorithms are developed to solve the LQT. A novel $Q$-learning algorithm is proposed in Section 5 to solve the LQT without any knowledge of the augmented system dynamics. Simulation results of the mentioned algorithms are presented in Section 6.

## 2. Review of standard solution of the LQT problem

In this section we review the standard solution for the linear quadratic tracker (LQT) problem. It is assumed here that the reference trajectory approaches zero as time goes to infinity.

Consider the linear discrete-time (DT) system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k \end{aligned} \tag{1}$$

where $x_k \in \mathbb{R}^n$ is the measured state, $u_k \in \mathbb{R}^m$ is the control input, $y_k \in \mathbb{R}^p$ is the output and $A$, $B$ and $C$ are constant matrices with compatible dimensions.

For the infinite-horizon LQT problem, the goal is to design an optimal controller for the system (1) which ensures that the output $y_k$ tracks a reference trajectory $r_k$ and guarantees stability. This can be achieved by minimizing the following infinite-horizon performance index

$$\begin{aligned} J_k &= \frac{1}{2}\sum_{i=k}^{\infty} U_i \\ &= \frac{1}{2}\sum_{i=k}^{\infty}\left[(Cx_i - r_i)^T Q(Cx_i - r_i) + u_i^T R u_i\right] \end{aligned} \tag{2}$$

where $\bar{r}_k = \{r_k, r_{k+1}, \ldots\}$, $U_i$ is the utility function at time step $i$, and $Q > 0$ and $R > 0$ are symmetric matrices.

The standard solution using the calculus of variation is provided as follows. Considering the system (1) with the performance index (2), the costate equation is given by (Lewis, Vrabie, & Syrmos, 2012)

$$\lambda_k = A^T\lambda_{k+1} + C^T QCx_k - C^T Qr_k \tag{3}$$

where $\lambda_k$ is the costate variable. The stationarity condition for finding the optimal control is

$$0 = B^T\lambda_{k+1} + Ru_k. \tag{4}$$

Therefore, the optimal control is

$$u_k = -R^{-1}B^T\lambda_{k+1}. \tag{5}$$

It is clear that the optimal control is a linear costate feedback, but because of the last term in the costate equation, it is no longer possible to express it as a linear state feedback as for the LQ regulator. However, $u_k$ can be expressed as a combination of a linear state variable feedback plus a term depending on $r_k$ (Lewis, Vrabie, & Syrmos, 2012). Thus,

$$\lambda_k = Sx_x - v_k^{SS} \tag{6}$$

for some as yet unknown auxiliary sequence $v_k^{SS}$ and gain $S$. This will turn out to be a valid assumption if a consistent equation can be found for $v_k^{SS}$. Using (1), (3), (5) and (6), and some manipulations yields

$$u_k = -K_x x_k + K_v v_k^{SS} \tag{7}$$

where $v_k^{SS} = \lim_{T\to\infty} v_k$ with

$$v_k = (A - BK_x)^T v_{k+1} + C^T Qr_k, \quad v(T) = 0 \tag{8}$$

and

$$K_x = (B^T SB + R)^{-1}B^T SA \tag{9}$$

$$K_v = (B^T SB + R)^{-1}B^T \tag{10}$$

where $S$ is obtained from solving the following algebraic Riccati equation (ARE)

$$C^T QC - S + A^T SA - A^T SB(B^T SB + R)^{-1}B^T SA = 0. \tag{11}$$

Sufficient conditions for the existence of a solution $S = S^T > 0$ to the ARE are $(A, B)$ stabilizable and $(A, \sqrt{Q}C)$ observable (Lewis, Vrabie, & Syrmos, 2012).

**Remark 1.** From (7) it is observed that the control input consists of a feedback term linear in $x_k$ plus a feedforward term independent of $x_k$. The gain $K_x$ of the first term depends on the solution of the ARE (11) and the second term depends on the difference equation (8). A drawback of this formulation of the LQT problem is the need to solve for $v_k$ backwards in time. That is, the standard LQT solution is noncausal.

**Remark 2.** Note that the assumption that the reference trajectory approaches zero as time goes to infinity is essential for minimizing the performance index (2). This is because the control input contains a part depending on the reference trajectory which makes (2) unbounded if the reference trajectory does not approach zero. Therefore, the meaning of minimality is lost. In the subsequent sections it is shown that we can relax this restrictive assumption by using a discount factor in the performance index.

**Remark 3.** A disadvantage to the standard LQT solution in Section 2 is that it can only be used for a class of reference trajectories that are generated by an asymptotically stable command generator. Another disadvantage of this solution is the need to compute the noncausal signal $v_k$ using backward recursion (8). Therefore, the infinite-horizon LQT problem has not received much attention in the literature.

## 3. Causal solution to the LQT problem and quadratic form of the LQT value function

In this section, we propose an alternative approach for formulating the infinite-horizon LQT problem in a causal manner. First, it is assumed that the reference trajectory is generated by a linear command generator and it is shown that in this case the value function of the LQT problem can be expressed as a quadratic form in terms of $x_k$ and $r_k$. Then, a Bellman equation is developed for the LQT, and an augmented LQT ARE is given. This allows us to use reinforcement learning (RL) to solve the LQT problem online in Section 4.

### 3.1. Quadratic form for the LQT value function

Before proceeding, the following assumption is made.

**Assumption 1.** The reference trajectory for the LQT problem is produced by the command generator model

$$r_{k+1} = Fr_k. \tag{12}$$

This command generator model does not assume that $F$ is Hurwitz. As such, it can generate a large class of useful command trajectories, including unit step (useful, e.g., in position command), sinusoidal waveforms (useful, e.g., in hard disk drive control), the ramp (useful in velocity tracking systems, e.g., satellite antenna pointing), and more.

Based on the system dynamics (1) and the reference trajectory dynamics (12), construct the augmented system

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} A & \mathbf{0} \\ \mathbf{0} & F \end{bmatrix} \begin{bmatrix} x_k \\ r_k \end{bmatrix} + \begin{bmatrix} B \\ \mathbf{0} \end{bmatrix} u_k \equiv TX_k + B_1 u_k \tag{13}$$

where the augmented state is

$$X_k = \begin{bmatrix} x_k \\ r_k \end{bmatrix}. \tag{14}$$

The performance index (2) can be only used if $F$ is Hurwitz. In practice this is not true, for instance, tracking of unit step and sinusoidal commands. In the following, it is shown that by introducing a discount factor in the performance index one can implement the infinite-horizon LQT even for the cases that the command generator dynamics $F$ is not Hurwitz. Consider the following discounted performance index or value function

$$\begin{aligned} V(x_k, \bar{r}_k) &= \frac{1}{2}\sum_{i=k}^{\infty} \gamma^{i-k} U_i \\ &= \frac{1}{2}\sum_{i=k}^{\infty} \gamma^{i-k} \left[ (Cx_i - r_i)^T Q (Cx_i - r_i) + u_i^T R u_i \right] \end{aligned} \tag{15}$$

where $0 < \gamma \leq 1$ is the discount factor. Note that $\gamma = 1$ can be only used if one knows a priori that the reference trajectory is generated by an asymptotically stable command generator system. That is, if $F$ in (12) is Hurwitz.

Note that the value function (15) can be written in terms of the augmented state as

$$V(x_k, \bar{r}_k) = \frac{1}{2}\sum_{i=k}^{\infty} \gamma^{i-k} \left[ X_i^T Q_1 X_i + u_i^T R u_i \right] \tag{16}$$

where

$$Q_1 = C_1^T Q C_1 \tag{17}$$

with $C_1 = [C \; -I]$.

The next lemma shows that the value function is quadratic in the state of the augmented system.

**Lemma 1** (*Quadratic Form for the Value Function*)**.** *For the infinite-horizon LQT problem, under Assumption 1, for any fixed stabilizing policy*

$$u_i = K_x x_i + K_r r_i \tag{18}$$

*the value function (15) can be written as*

$$V(x_k, \bar{r}_k) = V(x_k, r_k) = V(X_k) = \frac{1}{2} X_k^T P X_k \tag{19}$$

*for some matrix $P = P^T > 0$.*

**Proof.** Using (18) in (15) yields

$$\begin{aligned} V(x_k, \bar{r}_k) &= \frac{1}{2}\sum_{i=k}^{\infty} \gamma^{i-k} [(Cx_i - r_i)^T Q (Cx_i - r_i) \\ &\quad + (K_x x_i + K_r r_i)^T R (K_x x_i + K_r r_i)] \\ &= \frac{1}{2}\sum_{i=0}^{\infty} \gamma^{i} [x_{i+k}^T (C^T Q C + K_x^T R K_x) x_{i+k} \\ &\quad + x_{i+k}^T (-C^T Q + K_x^T R K_r) r_{i+k} + r_{i+k}^T (-QC \\ &\quad + K_r^T R K_x) x_{i+k} + r_{i+k}^T (Q + K_r^T R K_r) r_{i+k}]. \end{aligned} \tag{20}$$

Note that using (18), the solution of system dynamics (1) and reference trajectory (12) for specific initial condition $x_k$ and $r_k$ are

$$r_{i+k} = F^i r_k \tag{21}$$

$$x_{i+k} = G^i x_k + M r_k \tag{22}$$

where $G = A + BK_x$ and $M = \sum_{n=0}^{i-1} G^{i-n-1} B K_r F^n$.
Putting (21) and (22) in (20) results in

$$V(x_k, r_k) = \frac{1}{2} x_k^T P_{11} x_k + \frac{1}{2} x_k^T P_{12} r_k + \frac{1}{2} r_k^T P_{21} x_k + \frac{1}{2} r_k^T P_{22} r_k \tag{23}$$

where

$$P_{11} = \sum_{i=0}^{\infty} \gamma^i \left[ (G^i)^T (C^T Q C + K_x^T R K_x) G^i \right] \tag{24}$$

$$\begin{aligned} P_{12} &= \sum_{i=0}^{\infty} \gamma^i [(G^i)^T (-C^T Q + K_x^T R K_r) F^i \\ &\quad + (G^i)^T (C^T Q C + K_x^T R K_x) M] \end{aligned} \tag{25}$$

$$\begin{aligned} P_{21} &= \sum_{i=0}^{\infty} \gamma^i [(F^i)^T (-QC + K_r^T R K_x) G^i \\ &\quad + M^T (C^T Q C + K_x^T R K_x) G^i] \end{aligned} \tag{26}$$

$$P_{22} = \sum_{i=0}^{\infty} \gamma^i [M^T(-C^T Q + K_x^T R K_r)F^i + M^T(C^T Q C + K_x^T R K_x)M + (F^i)^T(-QC + K_r^T R K_x)M + (F^i)^T(Q + K_r^T R K_r)F^i]. \quad (27)$$

Therefore (19) holds with

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}. \quad (28)$$

This completes the proof. □

### 3.2. Bellman equation and ARE for the LQT problem

In this subsection we derive an LQT Bellman equation and an augmented LQT ARE in terms of $P$ in (19).

On the basis of (15) and (19) we have

$$V(x_k, r_k) = \frac{1}{2}(Cx_k - r_k)^T Q(Cx_k - r_k) + \frac{1}{2}u_k^T R u_k + \frac{\gamma}{2}\sum_{i=k+1}^{\infty} \gamma^{i-(k+1)}\left[(Cx_i - r_i)^T Q(Cx_i - r_i) + u_i^T R u_i\right] \quad (29)$$

which yields the LQT Bellman equation

$$V(x_k, r_k) = \frac{1}{2}(Cx_k - r_k)^T Q(Cx_k - r_k) + \frac{1}{2}u_k^T R u_k + \gamma V(x_{k+1}, r_{k+1}). \quad (30)$$

Using (19) in (30) we obtain the LQT Bellman equation in terms of value function kernel matrix $P$ as

$$X_k^T P X_k = X_k^T Q_1 X_k + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1} \quad (31)$$

where $Q_1$ is defined in (17).

Define the LQT Hamiltonian

$$H(X_k, u_k) = X_k^T Q_1 X_k + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1} - X_k^T P X_k \quad (32)$$

or equivalently

$$H(X_k, u_k) = X_k^T Q_1 X_k + u_k^T R u_k + \gamma V(X_{k+1}) - V(X_k). \quad (33)$$

The next theorem shows how the LQT problem can be solved in a causal manner using an augmented ARE.

**Theorem 1** (*ARE for the causal solution of the LQT problem*)**.** *Under Assumption 1 and using (19), any optimal policy for the LQT problem has the form*

$$u_k = -K_1 X_k \quad (34)$$

*where*

$$K_1 = (R + \gamma B_1^T P B_1)^{-1}\gamma B_1^T P T \quad (35)$$

*and $P$ satisfies the augmented LQT ARE*

$$Q_1 - P + \gamma T^T P T - \gamma^2 T^T P B_1 (R + \gamma B_1^T P B_1)^{-1} B_1^T P T = 0. \quad (36)$$

**Proof.** A necessary condition for optimality (Lewis, Vrabie, & Vamvoudakis, 2012) is the stationary condition

$$\frac{\partial H(X_k, u_k)}{\partial u_k} = 2Ru_k + \gamma \frac{\partial X_{k+1}}{\partial u_k}^T \frac{\partial V(X_{k+1})}{\partial X_{k+1}} = 2Ru_k + 2\gamma B_1^T P X_{k+1} = 0.$$

Then

$$u_k = -(R + \gamma B_1^T P B_1)^{-1}\gamma B_1^T P T X_k. \quad (37)$$

Substituting (13) and (37) in the Bellman equation (31) results in the LQT ARE (36). □

In the following theorem, the stability of tracking error for the optimal control input, given by solving the LQT ARE (36), is discussed. It is shown that the convergence of the tracking error to zero cannot be guaranteed because of using the discount factor in the value function. However, it is discussed that by choosing a proper discount factor and a weighting matrix $Q$ in the value function, one can make the tracking error as small as desired.

**Theorem 2** (*Stability and Optimality of the LQT ARE Solution*)**.** *Consider the LQT problem for the system (1) with the command generator (12) and the value function (15). Define $\bar{e}_k = \gamma^{k/2} e_k$, where $e_k = Cx_k - r_k$ is the tracking error at sample time $k$. Then, the optimal control input obtained by solving the LQT ARE (36) asymptotically stabilizes $\bar{e}_k$. Moreover, it minimizes the value function (15) over all stabilizing controls.*

**Proof.** We first show that $\bar{e}_k$ is asymptotically stable. Consider the augmented system (13) with the state $X_k$. Define the new state $\bar{X}_k = \gamma^{k/2} X_k$. Since $e_k = [C \; -I]X_k$ and $[C \; -I] \neq 0$, if $\bar{X}_k$ goes to zero, then $\bar{e}_k$ goes to zero. In the following, it is shown that $\bar{X}_k$ and consequently $\bar{e}_k$ converges to zero as $k$ goes to infinity.

Consider the following Lyapunov function

$$V(\bar{X}_k) = \frac{1}{2}\bar{X}_k^T P \bar{X}_k \quad (38)$$

where $P$ is the solution of the LQT ARE (36). Then we have

$$V(\bar{X}_{k+1}) - V(\bar{X}_k) = \frac{1}{2}\bar{X}_{k+1}^T P \bar{X}_{k+1} - \frac{1}{2}\bar{X}_k^T P \bar{X}_k. \quad (39)$$

Using $X_k = \gamma^{-k/2}\bar{X}_k$ and the control input (34) in (13), one has

$$\bar{X}_{k+1} = \gamma^{1/2} T \bar{X}_k + \gamma^{1/2} B_1 \bar{u}_k = \gamma^{1/2}(T - B_1(R + \gamma B_1^T P B_1)^{-1}\gamma B_1^T P T)\bar{X}_k \quad (40)$$

where $\bar{u}_k = -K_1 \bar{X}_k$. Putting (40) in (39) and adding and subtracting $K_1^T R K_1$ and some manipulations yields

$$V(\bar{X}_{k+1}) - V(\bar{X}_k) = \frac{1}{2}\bar{X}_k^T[-P + \gamma T^T P T - \gamma^2 T^T P B_1 (R + \gamma B_1^T P B_1)^{-1} B_1^T P T - K_1^T R K_1]\bar{X}_k \quad (41)$$

where $K_1$ is defined in (35). From (36) one has

$$-P + \gamma T^T P T - \gamma^2 T^T P B_1 (R + \gamma B_1^T P B_1)^{-1} B_1^T P T = -Q_1. \quad (42)$$

Putting (42) in (41) yields

$$V(\bar{X}_{k+1}) - V(\bar{X}_k) = \frac{1}{2}\bar{X}_k^T(-Q_1 - K_1^T R K_1)\bar{X}_k < 0. \quad (43)$$

This completes the proof of the stability.

To show the optimality, note that

$$\frac{1}{2}(\bar{X}_\infty^T P \bar{X}_\infty - \bar{X}_k^T P \bar{X}_k) = \frac{1}{2}\sum_{i=k}^{\infty}[\bar{X}_{i+1}^T P \bar{X}_{i+1} - \bar{X}_i^T P \bar{X}_i]$$
$$= \frac{1}{2}\sum_{i=k}^{\infty}[\gamma(T\bar{X}_i + B_1\bar{u}_i)^T P(T\bar{X}_i + B_1\bar{u}_i) - \bar{X}_i^T P \bar{X}_i]$$
$$= \frac{1}{2}\sum_{i=k}^{\infty}[\bar{X}_i^T(\gamma T^T P T - P)\bar{X}_i + \gamma \bar{X}_i^T T^T B_1 \bar{u}_i + \gamma \bar{u}_i^T B_1^T P T \bar{X}_i + \gamma \bar{u}_i^T B_1^T P B_1 \bar{u}_i]. \quad (44)$$

Using the LQT ARE (36) in (44) and since $\bar{X}_\infty = 0$, one has

$$\frac{1}{2}\bar{X}_k^T P \bar{X}_k + \frac{1}{2}\sum_{i=k}^{\infty}[\bar{X}_i^T(-Q_1 + \gamma^2 T^T P B_1 (R + \gamma B_1^T P B_1)^{-1} B_1^T P T)\bar{X}_i + \gamma \bar{X}_i^T T^T P B_1 \bar{u}_i + \gamma \bar{u}_i^T B_1^T P T \bar{X}_i + \gamma \bar{u}_i^T B_1^T P B_1 \bar{u}_i] = 0. \quad (45)$$

On the other hand, the value function (15) in terms of $\bar{X}_k$ and $\bar{u}_k$ can be written as

$$V(\bar{X}_k, \bar{u}_k) = \frac{1}{2}\gamma^{-k}\sum_{i=k}^{\infty}[\bar{X}_i^T Q_1 \bar{X}_i + \bar{u}_i^T R \bar{u}_i]. \tag{46}$$

In fact, minimizing the value function (15) with respect to the system (13) is equivalent to minimizing the value function (46) with respect to (40).

Multiplying the right-hand side of (45) by $\gamma^{-k}$ and adding its result to (46) yields

$$\begin{aligned} &V(\bar{X}_k, \bar{u}_k) \\ &= \frac{1}{2}X_k^T P X_k + \frac{\gamma^{-k}}{2}\sum_{i=k}^{\infty}[\bar{X}_i^T(\gamma^2 T^T P B_1 (R+\gamma B_1^T P B_1)^{-1} B_1^T P T)\bar{X}_i \\ &\quad + \gamma \bar{X}_i^T T^T P B_1 \bar{u}_i + \gamma \bar{u}_i^T B_1^T P T \bar{X}_i + \bar{u}_i^T (R+\gamma B_1^T P B_1)\bar{u}_i]. \end{aligned} \tag{47}$$

Completing the square gives

$$\begin{aligned} V(\bar{X}_k, \bar{u}_k) = \frac{1}{2}X_k^T P X_k + \frac{\gamma^{-k}}{2}\sum_{i=k}^{\infty}&[\bar{u}_i + (R+\gamma B_1^T P B_1)^{-1} \\ &\times \gamma B_1^T P T \bar{X}_i]^T (R+\gamma B_1^T P B_1) \\ &\times [\bar{u}_i + (R+\gamma B_1^T P B_1)^{-1}\gamma B_1^T P T \bar{X}_i]. \end{aligned} \tag{48}$$

Since $R > 0$, (46) achieves its minimum when $\bar{u}_k = -K_1 \bar{X}_k$, where $K_1$ is given in (35). Consequently, $u_k = -K_1 X_k$ minimizes the value function (15) and this completes the proof of the optimality. □

**Remark 4.** Theorem 2 shows that the tracking error is bounded when the optimal control input obtained by the LQT ARE is applied to the system. Moreover, Eq. (43) shows that the larger the $Q$ in the value function is the faster the tracking error decreases (see (17)). Therefore, by choosing a smaller discount factor and/or larger $Q$ one can make the tracking error as small as desired before the value of $\gamma^i$ becomes very small. Simulation results in Section 6 confirm this conclusion.

## 4. Reinforcement learning to solve LQT online

In this section we use the causal LQT formulation of Section 3 to develop RL algorithms for the LQT, where the value function and control law are updated by recursive iterations online using data measured along the system trajectories.

For an arbitrary stabilizing gain $K_1$ in (34), the augmented LQT Bellman equation (31) becomes the LQT Lyapunov equation

$$Q_1 - P + K_1^T R K_1 + \gamma(T - B_1 K_1)^T P (T - B_1 K_1) = 0. \tag{49}$$

Instead of directly solving the LQT ARE (36), the following policy iteration (PI) algorithm based on repeated solutions of (49) can be employed.

**Algorithm 1** (*Offline Policy Iteration for LQT Solution*)**.** Initialization: Start with a stabilizing control policy $K_1$.

1. Policy evaluation, solve for $P^{j+1}$ using the LQT Lyapunov equation

$$P^{j+1} = Q_1 + (K_1^j)^T R K_1^j + \gamma(T - B_1 K_1^j)^T P^{j+1}(T - B_1 K_1^j). \tag{50}$$

2. Policy improvement

$$K_1^{j+1} = (R + \gamma B_1^T P^{j+1} B_1)^{-1}\gamma B_1^T P^{j+1} T. \tag{51}$$

This algorithm is an extension of Hewer's method (Hewer, 1971) to the LQT problem. The proof there shows that $P^j$ in Algorithm 1 converges to the solution to the LQT ARE (36) and that $K_1$ is stabilizing at each step.

The Lyapunov equation (50) in Algorithm 1 evaluates a fixed control policy in an offline manner and it requires complete knowledge of the system dynamics. However, one can use the Bellman equation (31), instead of the Lyapunov equation (50), to evaluate a control policy in an online manner and without requiring knowledge of the system dynamics. The next algorithm uses the LQT Bellman equation (31) to solve the LQT online.

**Algorithm 2** (*Online Policy Iteration for LQT Solution*)**.** Initialization: Start with a stabilizing control policy $K_1$.

1. Policy evaluation, solve for $P^{j+1}$ using the LQT Bellman equation.

$$X_k^T P^{j+1} X_k = X_k^T (Q_1 + (K_1^j)^T R K_1^j) X_k + \gamma X_{k+1}^T P^{j+1} X_{k+1}. \tag{52}$$

2. Policy improvement.

$$K_1^{j+1} = (R + \gamma B_1^T P^{j+1} B_1)^{-1}\gamma B_1^T P^{j+1} T. \tag{53}$$

Policy iteration Algorithm 2 can be implemented online using least-squares (LS) using the data tuple $X_k, X_{k+1}$ and $\rho_k$ measured along the system trajectories with $\rho_k = X_k^T\left(Q_1 + (K_1^j)^T R K_1^j\right)X_k$. In fact (52) is a scalar equation and $P$ is a symmetric $(n+p)\times(n+p)$ matrix with $(n+p)\times(n+p+1)/2$ independent element. Therefore at least $(n+p)\times(n+p+1)/2$ data tuples are required before (52) can be solved using LS. Both batch LS and recursive LS methods can be used to perform the policy evaluation step (52) (Lewis, Vrabie, & Vamvoudakis, 2012). The system dynamics $(T, B_1)$ is not needed to solve the Bellman equation (52), but must be known to update the control policy using (53).

To obviate the requirement for complete knowledge of the system dynamics, a $Q$-learning algorithm is developed in the next section (see Algorithm 3) to solve the LQT problem.

**Remark 5.** The requirement for initial stabilizing policy can be avoided by using the value iteration algorithm (Lewis & Vrabie, 2009).

## 5. Q-learning to solve the LQT online

The online LQT policy iteration Algorithm 2 requires knowledge of the system dynamics $(T, B_1)$. In this section a $Q$-learning algorithm (Landelius & Knutsson, 1996; Watkins, 1989; Werbos, 1992) is developed that solves the LQT ARE (36) online without requiring any knowledge of the system dynamics $(A, B_1)$ or command generator dynamics $(F)$.

### 5.1. Q-function for the LQT

Based on the LQT Bellman equation (31), the discrete-time LQT $Q$-function is defined as

$$Q(x_k, r_k, u_k) = \frac{1}{2}X_k^T Q_1 X_k + \frac{1}{2}u_k^T R u_k + \frac{1}{2}\gamma X_{k+1}^T P X_{k+1} \tag{54}$$

where $Q_1$ is defined in (17).

By using augmented system dynamics (13), (54) becomes

$$\begin{aligned} Q(X_k, u_k) &= \frac{1}{2}X_k^T Q_1 X_k + \frac{1}{2}u_k^T R u_k \\ &\quad + \frac{1}{2}\gamma(TX_k + B_1 u_k)^T P (TX_k + B_1 u_k) \\ &= \frac{1}{2}\begin{bmatrix} X_k \\ u_k \end{bmatrix}^T \begin{bmatrix} Q_1 + \gamma T^T P T & \gamma T^T P B_1 \\ \gamma B_1^T P T & R + \gamma B_1^T P B_1 \end{bmatrix}\begin{bmatrix} X_k \\ u_k \end{bmatrix}. \end{aligned} \tag{55}$$

Therefore, define

$$Q(X_k, u_k) = \frac{1}{2}\begin{bmatrix} X_k \\ u_k \end{bmatrix}^T H \begin{bmatrix} X_k \\ u_k \end{bmatrix} = \frac{1}{2}\begin{bmatrix} X_k \\ u_k \end{bmatrix}^T \begin{bmatrix} H_{XX} & H_{Xu} \\ H_{uX} & H_{uu} \end{bmatrix} \begin{bmatrix} X_k \\ u_k \end{bmatrix} \tag{56}$$

for kernel matrix $H = H^T$.

Applying $\frac{\partial Q(X_k, u_k)}{\partial u_k} = 0$ to (56) yields

$$u_k = -H_{uu}^{-1} H_{uX} X_k \tag{57}$$

and to (55) yields

$$u_k = -(R + \gamma B_1^T P B_1)^{-1} \gamma B_1^T P T X_k \tag{58}$$

as in Eq. (34).

Eq. (58) requires knowledge of the augmented system dynamics $(T, B_1)$ to compute the LQT control. On the other hand, (57) requires knowledge only of the $Q$-function matrix kernel $H$. RL is used in the next subsection to determine the kernel matrix $H$ online without knowing the augmented system dynamics using data measured along the system trajectories.

### 5.2. Q-learning for LQT

Based on the definition of $Q$-function (54), one can introduce a $Q$-learning algorithm to solve the LQT ARE (36) online without knowing the augmented system dynamics $(T, B_1)$.

The infinite-horizon $Q$-function is given by (54). Hence the $Q$-function satisfies the Bellman equation

$$Q(X_k, u_k) = \frac{1}{2} X_k^T Q_1 X_k + \frac{1}{2} u_k^T R u_k + \gamma Q(X_{k+1}, u_{k+1}) \tag{59}$$

where the policy $u_{k+1} = -K_1 X_{k+1}$ is followed after time $k$. Define

$$Z_k = \begin{bmatrix} X_k \\ u_k \end{bmatrix} \tag{60}$$

to write (56) as

$$Q(X_k, u_k) = \frac{1}{2} Z_k^T H Z_k. \tag{61}$$

Substituting (61) into (59), the $Q$-function Bellman equation (59) becomes

$$Z_k^T H Z_k = X_k^T Q_1 X_k + u_k^T R u_k + \gamma Z_{k+1}^T H Z_{k+1}. \tag{62}$$

Policy iteration is especially easy to implement in terms of the $Q$-function, as follows.

**Algorithm 3** (*LQT Policy Iteration Solution Using the LQT Q-Function*)**.**

1. *Policy evaluation.*

$$Z_k^T H^{j+1} Z_k = X_k^T Q_1 X_k + (u_k^j)^T R (u_k^j) + \gamma Z_{k+1}^T H^{j+1} Z_{k+1}. \tag{63}$$

2. *Policy improvement.*

$$u_k^{j+1} = -(H_{uu}^{-1})^{j+1} H_{uX}^{j+1} X_k. \tag{64}$$

Note that in contrast to the Bellman equation (52) in Algorithm 2, the control input appears in quadratic form of the $Q$-function Bellman equation (63). Therefore, in contrast to Algorithm 2, the policy improvement step (64) in Algorithm 3, which is given by minimizing the $Q$-function (63) with respect to the control input, can be carried out in terms of the learned kernel matrix $H^{j+1}$ without resorting to the system dynamics.

The convergence of Algorithm 3 can be proven as in Al-Tamimi et al. (2007). Note that, policy iteration using $Q$-function is performed online and can be implemented without requiring any knowledge of the augmented system dynamics based on Least-squares (LS) using the data tuple $Z_k$, $Z_{k+1}$ and $\rho_k$ measured along the system trajectories with $\rho_k = X_k^T Q_1 X_k + (u_k^j)^T R u_k^j$. In fact (63) is a scalar equation and $H$ is a symmetric $(n+p+m) \times (n+p+m)$ matrix with $(n+p+m) \times (n+p+m+1)/2$ independent elements. Therefore at least $(n+p+m) \times (n+p+m+1)/2$ data tuples are required before (63) can be solved using LS. Both batch LS and recursive LS methods can be used to perform the policy evaluation step (63) (Lewis, Vrabie, & Vamvoudakis, 2012).

**Remark 6.** Policy iteration based adaptive optimal control schemes require a persistent excitation condition (PE) (Al-Tamimi et al., 2007; Bradtke et al., 1994; Lewis & Vrabie, 2009; Vrabie et al., 2009), to ensure the sufficient exploration of the state space. If the state almost converges to the desired position and becomes stationary, the PE is no longer satisfied. An exploratory signal consisting of sinusoids of varying frequencies can be added to the control input to ensure PE qualitatively.

## 6. Simulation results

In this section, a simulation example is carried out to illustrate the design procedures and verify the effectiveness of the proposed scheme.

A linear system is considered as

$$x_{k+1} = \begin{bmatrix} -1 & 2 \\ 2.2 & 1.7 \end{bmatrix} x_k + \begin{bmatrix} 2 \\ 1.6 \end{bmatrix} u_k \qquad y_k = \begin{bmatrix} 1 & 2 \end{bmatrix} x_k. \tag{65}$$

The open-loop poles are $z_1 = -2.1445$ and $z_2 = 2.8445$, so the system is unstable.

The performance index is considered as (15) with $Q = 6$, $R = 1$ and $\gamma = 0.8$. It is supposed that the sinusoid reference trajectory is generated by the command generator dynamics given by

$$r_{k+1} = -r_k. \tag{66}$$

### 6.1. Policy iteration using value function

In this subsection Algorithms 1 and 2, which use value function structure (19) to evaluate the performance of a policy, are applied for the system (65) and the reference trajectory (66).

The optimal matrix $P$ satisfying the ARE (36) for this problem is

$$P^* = \begin{bmatrix} 133.3840 & 16.0531 & 31.1402 \\ 16.0531 & 25.1604 & -10.8271 \\ 31.1402 & -10.8271 & 18.4825 \end{bmatrix}. \tag{67}$$

First, offline policy iteration Algorithm 1 is implemented as in (50) and (51). Fig. 1 shows that the $P$ matrix parameters converge to their optimal values. After 12 iterations the $P$ matrix parameters converge to

$$P = \begin{bmatrix} 133.3840 & 16.0531 & 31.1402 \\ 16.0531 & 25.1604 & -10.8271 \\ 31.1402 & -10.8271 & 18.4825 \end{bmatrix}. \tag{68}$$

The results of applying the optimal control given by substituting the $P$ matrix (68) in (34), (35) to the system (65) are now presented. Fig. 2 shows that the output $y_k$ tracks the reference trajectory $r_k$ and guarantees the stability for the offline policy iteration Algorithm 1. The optimal control signal input is shown in Fig. 3.

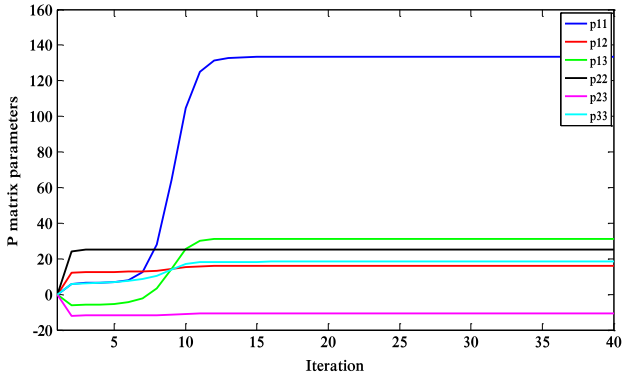


**Fig. 1.** Convergence of the $P$ matrix parameters to their optimal values for offline PI Algorithm 1.

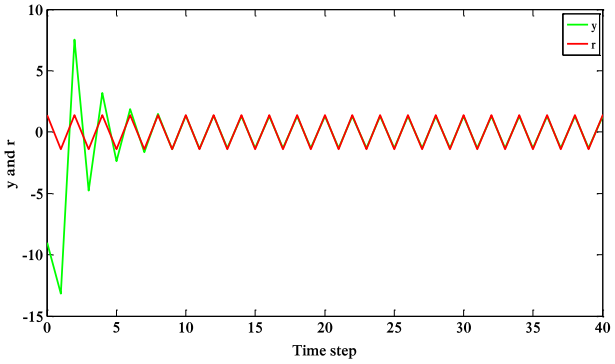


**Fig. 2.** Evaluation of the output and the reference trajectory for offline PI Algorithm 1.

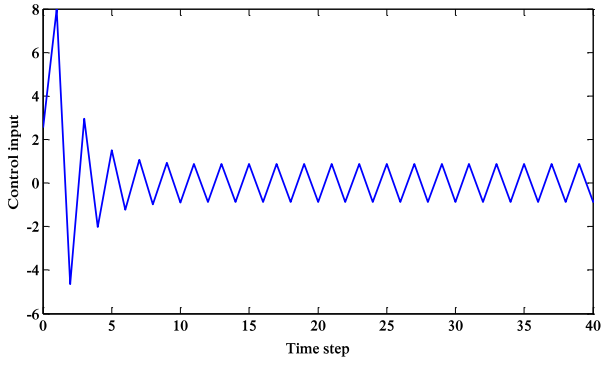


**Fig. 3.** The control input during learning.

Next, the online policy iteration Algorithm 2 is implemented as in (52) and (53). PE was ensured by adding a probing noise to the control input. Fig. 4 shows how the $P$ matrix parameters converge to their optimal values. After 20 iterations the $P$ matrix parameters converge to

$$P = \begin{bmatrix} 133.3840 & 16.0531 & 31.1402 \\ 16.0531 & 25.1604 & -10.8271 \\ 31.1402 & -10.8271 & 18.4825 \end{bmatrix}.$$

Comparing this $P$ matrix with the $P^*$ matrix, it is seen that the online Algorithm 2 converges very close to the optimal controller.

### 6.2. Policy iteration using Q-function

The policy iteration Algorithm 3, which uses the $Q$-function to evaluate a policy, is implemented as in (63) and (64).

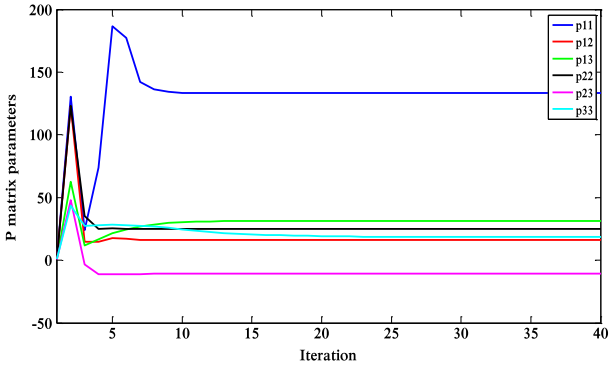


**Fig. 4.** Convergence of the $P$ matrix parameters to their optimal values for online PI Algorithm 2.

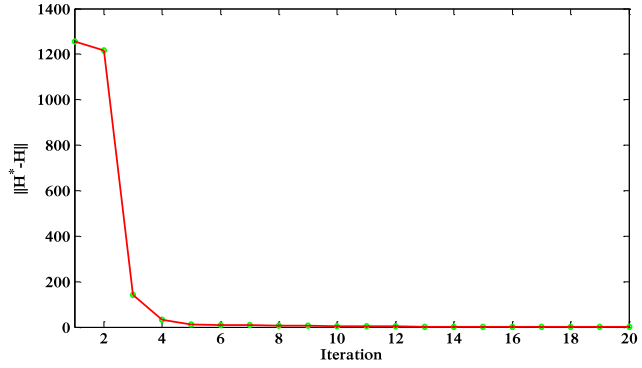


**Fig. 5.** Convergence of $H$ matrix to its optimal values $H^*$ during the learning process.

The optimal $P$ matrix for this problem by solving the ARE (36) is (67). Using this $P$ matrix and considering the definition of $Q$-function in (55) and (56) and definitions of $B_1$ and $T$ in (13), the optimal $H$ function becomes

$$H^* = \begin{bmatrix} 153.6214 & -91.4595 & 37.9679 & -106.6035 \\ -91.4595 & 596.3286 & -47.0995 & 566.3383 \\ 37.9679 & -47.0995 & 20.7860 & -35.9657 \\ -106.6035 & 566.3383 & -35.9657 & 561.5493 \end{bmatrix}.$$

Consequently, using (58) for the optimal control $u_k^* = -K^* X_k$, the control gain $K^*$ is given as

$$K^* = \begin{bmatrix} -0.1898 & 1.0085 & -0.0640 \end{bmatrix}.$$

Now, Algorithm 3 is used to solve the problem. It is assumed that the dynamics $T$ and $B_1$ are completely unknown. For the purpose of demonstrating the algorithm, the initial state of the augmented system is chosen as $X_0 = [5 \quad -5 \quad 5]^T$ and initial control input is chosen as $K_0 = [0.3 \quad 1.3 \quad 0.75]$. In each iteration, 21 data samples are collected to perform the LS. PE was ensured by adding a probing noise to the control input. Figs. 5 and 6 show norm of the difference of the optimal and the computed $H$ matrices as well as norm of the difference between the optimal control gain and the computed gain, respectively. After 6 iterations the $H$ matrix parameters and the control gain converge to

$$H = \begin{bmatrix} 153.6214 & -91.4595 & 37.9681 & -106.6935 \\ -91.4595 & 596.3286 & -47.1000 & 566.3383 \\ 37.9681 & -47.1000 & 19.0389 & -35.9662 \\ -106.6935 & 566.3383 & -35.9662 & 561.5493 \end{bmatrix}$$

and

$$K = \begin{bmatrix} -0.1898 & 1.0085 & -0.0640 \end{bmatrix}.$$

Fig. 7 shows the output of the system and the reference trajectory during the learning process. Fig. 8 shows the probing noise injected

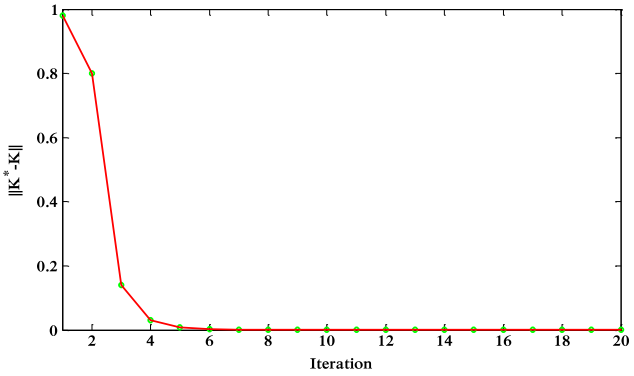


**Fig. 6.** Convergence of $K$ matrix to its optimal values $K^*$ during the learning process.

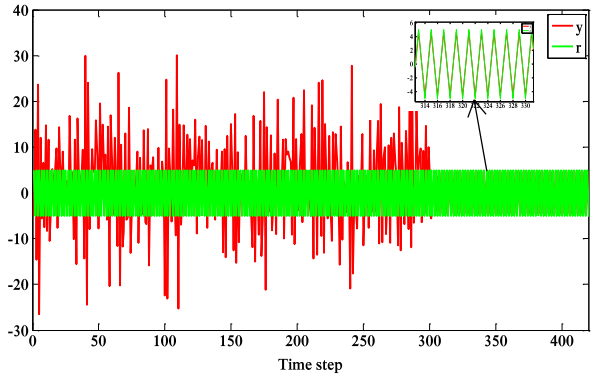


**Fig. 7.** Evaluation of the output and the reference trajectory during the learning process.

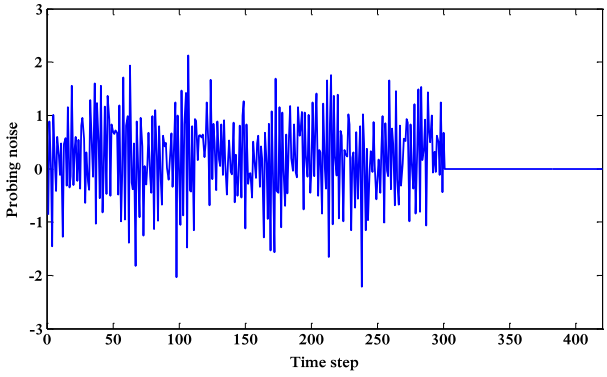


**Fig. 8.** Probing noise during the learning process.

to the control input during the learning process. It is clear that after 300 time step the PE condition is no longer needed. Therefore, probing noise is turned off. Thereafter, the output of the system is very close to the reference trajectory as it is required.

## 7. Conclusion

In this paper, a causal solution of the LQT problem using reinforcement learning was presented. It was shown that the value function has quadratic form in terms of the state and the reference trajectory. On the basis of this value function, an LQT ARE was obtained and a $Q$-learning algorithm was developed to solve the LQT ARE online without requiring knowledge of system dynamics. The simulation results have shown that the proposed formulation for the LQT problem gives good tracking performance.

Future research efforts will focus on how to implement previous architecture to solve the LQT using only the measured input, output and reference trajectory data. Moreover, the value iteration implementation of this architecture will be developed to avoid the requirement of an admissible control policy.

## References


Al-Tamimi, A., Lewis, F. L., & Abu-Khalaf, M. (2007). Model-free Q-learning designs for linear discrete-time zero-sum games with application to $H$-infinity control. *Automatica*, *43*(3), 473–481.

Al-Tamimi, A., Lewis, F. L., & Abu-Khalaf, M. (2008). Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, *38*(4), 943–949.

Baird, L. C. III (1994). Reinforcement learning in continuous time: advantage updating. In *Proc. of ICNN*.

Barto, J. Si. A., Powell, W., & Wunch, D. (2004). *Handbook of learning and approximate dynamic programming*. John Wiley.

Bertsekas, D. P., & Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. MA: Athena Scientific.

Bradtke, S. J., Ydestie, B. E., & Barto, A. G. (1994). Adaptive linear quadratic control using policy iteration. In: *Proc. of ACC* (pp. 3475–3476).

Dierks, T., & Jagannathan, S. (2010). Optimal control of affine nonlinear continuous-time systems. In *Proc. Am. control conf.* (pp. 1568–1573).

Hewer, G. A. (1971). An iterative technique for the computation of steady state gains for the discrete optimal regulator. *IEEE Transactions on Automatic Control*, *16*(4), 382–384.

Jiang, Y., & Jiang, Z. P. (2012). Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica*, *48*, 2699–2704.

Landelius, T., & Knutsson, H. (1996). Greedy adaptive critics for LQR problem: convergence proof. Technical report, Linkoping, Sweden, Computer vision laboratory.

Lee, J. Y., Park, J. B., & Choi, Y. H. (2012). Integral $Q$-learning and explorized policy iteration for adaptive optimal control of continuous-time linear systems. *Automatica*, *48*, 2850–2859.

Lewis, F. L., & Liu, D. (Eds.) (2013). *Reinforcement learning and approximate dynamic programming for feedback control*. Hoboken, NJ: Wiley.

Lewis, F. L., & Vamvoudakis, K. (2011). Reinforcement learning for partially observable dynamic processes: adaptive dynamic programming using measured output data. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, *41*(1), 14–23.

Lewis, F. L., Vamvoudakis, K., & Vrabie, D. (2013). *Optimal adaptive control and differential games by reinforcement learning principles*. London: Institution of Engineering and Technology.

Lewis, F. L., & Vrabie, D. (2009). Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, *9*(3), 32–50.

Lewis, F. L., Vrabie, D., & Syrmos, V. (2012). *Optimal control*. John Wiley.

Lewis, F. L., Vrabie, D., & Vamvoudakis, K. G. (2012). Reinforcement learning and feedback control using natural decision methods to design optimal adaptive controllers. *IEEE Systems Magazine*, *32*(6), 76–105.

Powell, W. B. (2009). *Approximate dynamic programming: solving the curses of dimensionality*. John Wiley.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning—an introduction*. Cambridge, MT: MIT Press.

Vrabie, D., Pastravanu, O., Abu-Khalaf, M., & Lewis, F. L. (2009). Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica*, *45*, 477–484.

Wang, D., Liu, D., & Wei, Q. (2012). Finite-horizon neurooptimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach. *Neurocomputing*, *78*, 14–22.

Watkins, C. J. C. H. (1989). Learning from delayed rewards. *Ph.D. thesis*, England: University of Cambridge.

Werbos, P. J. (1989). Neural networks for control and system identification. In: *Proc. of CDC* (pp. 260–265).

Werbos, P. J. (1991). A menu of designs for reinforcement learning over time. In *Neural networks for control* (pp. 67–95). Cambridge, MA: MIT Press.

Werbos, P. J. (1992). Approximate dynamic programming for real-time control and neural modeling. In D. A. White, & D. A. Sofge (Eds.), *Handbook of intelligent control*. New York: Van Nostrand Reinhold.

White, D. A., & Sofge, D. A. (Eds.) (1992). *Handbook of intelligent control*. New York: Van Nostrand Reinhold.

Zhang, H., Cui, L., Zhang, X., & Luo, X. (2011). Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method. *IEEE Transactions on Neural Networks*, *22*, 2226–2236.

Zhang, H., Liu, D., Luo, Y., & Wang, D. (2012). *Adaptive dynamic programming for control-algorithms and stability*. London: Springer-Verlag.

Zhang, H., Wei, Q., & Liu, D. (2011). An iterative approximate dynamic programming method to solve for a class of nonlinear zero-sum differential games. *Automatica*, *47*(1), 207–214.

Zhang, H., Wei, Q., & Luo, Y. (2008). A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *38*, 937–942.

**Bahare Kiumarsi** received the B.S. degree from Shahrood University of Technology, Iran, 2009 and the M.S. degree from Ferdowsi University of Mashhad, Iran, 2013. From August 2012 to December 2013, she was a Visiting Scholar with the University of Texas at Arlington, TX, USA, where she is currently working toward the Ph.D. degree. Her research interests include optimal control, reinforcement learning and neural networks.

**Frank L. Lewis** is a Member of National Academy of Inventors, Fellow IEEE, Fellow IFAC, Fellow UK Institute of Measurement and Control, PE Texas, and UK Chartered Engineer. He is also a UTA Distinguished Scholar Professor, UTA Distinguished Teaching Professor, and Moncrief-O'Donnell Chair at The University of Texas at Arlington Research Institute. He is also an IEEE Control Systems Society *Distinguished Lecturer*. He obtained the Bachelor's Degree in Physics/EE and the MSEE at Rice University, the M.S. in Aeronautical Engineering from Univ. W. Florida, and the Ph.D. at Ga. Tech. He works in feedback control, reinforcement learning, intelligent systems, and distributed control systems. He is the author of 6 US patents, 273 journal papers, 375 conference papers, 15 books, 44 chapters, and 11 journal special issues. He received the Fulbright Research Award, NSF Research Initiation Grant, ASEE *Terman Award*, Int. Neural Network Soc. *Gabor Award* 2009, UK Inst Measurement and Control *Honeywell Field Engineering Medal* 2009. He received IEEE Computational Intelligence Society *Neural Networks Pioneer Award* 2012. He was a Distinguished Foreign Scholar, Nanjing Univ. Science and Technology. He was also Project 111 Professor at Northeastern University, China. He received Outstanding Service Award from Dallas IEEE section and was selected as Engineer of the Year by Ft. Worth IEEE Section. He was listed in Ft. Worth Business Press Top 200 Leaders in Manufacturing. He received the 2010 IEEE Region 5 Outstanding Engineering Educator Award and the 2010 UTA Graduate Dean's Excellence in Doctoral Mentoring Award. He was elected to UTA Academy of Distinguished Teachers in 2012. He served on the NAE Committee on Space Station in 1995. He is the Founding Member of the Board of Governors of the Mediterranean Control Association. He helped win the IEEE Control Systems Society Best Chapter Award (as Founding Chairman of DFW Chapter), the National Sigma Xi Award for Outstanding Chapter (as President of UTA Chapter), and the US SBA Tibbetts Award in 1996 (as Director of ARRI's SBIR Program).

**Hamidreza Modares** received the B.S. degree from University of Tehran in 2004 and the M.S. degree from Shahrood University of Technology in 2006. He is currently working toward the Ph.D. degree at Ferdowsi University of Mashhad. He joined Shahrood University of Technology as a University Lecturer from 2006 to 2009. From August 2012, he has been a Visiting Scholar with the University of Texas at Arlington Research Institute. His research interests include optimal control, reinforcement learning, approximate dynamic programming, neural adaptive control and pattern recognition.

**Ali Karimpour** was born in Mashhad, Iran, in 1964. He received the B.Sc. and M.Sc. degrees in electrical engineering from the Ferdowsi University of Mashhad, Iran, in 1987 and 1990, respectively. He received the Ph.D. degree in electrical engineering from Ferdowsi University of Mashhad, Iran. He is currently working as an Associate Professor at the Ferdowsi University of Mashhad. His research interests include multivariable control, hybrid systems, renewable energies, power system stability and control structure design in power systems.

**Mohammad-Bagher Naghibi Sistani** received the B.Sc. and M.Sc. degrees in control engineering with honors from the University of Tehran, Tehran, Iran, in 1991 and 1995, respectively and the Ph.D. degree from the department of Electrical Engineering at Ferdowsi University of Mashhad in 2005. He was a Lecturer at Ferdowsi University of Mashhad from 2001 to 2005, where he is now Assistant Professor. His research interests are artificial intelligence, reinforcement learning and control systems.