Received: 4 January 2018 | Revised: 21 October 2018 | Accepted: 23 October 2018
DOI: 10.1002/oca.2477

RESEARCH ARTICLE

WILEY

LQ control of unknown discrete-time linear systems—A novel approach and a comparison study

Nan Li | Ilya Kolmanovsky | Anouck Girard

Department of Aerospace Engineering, University of Michigan, Ann Arbor, Michigan

Correspondence
Nan Li, Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109-2140.
Email: nanli@umich.edu

Summary

In this paper, we propose a novel approach to the linear quadratic (LQ) optimal control of unknown discrete-time linear systems. We first describe an iterative procedure for minimizing a partially unknown static function. The procedure is based on simultaneous updates in the estimation of unknown parameters and in the optimization of controllable inputs. We then use the procedure for control optimization in unknown discrete-time dynamic systems—we consider applications to the finite-horizon and the infinite-horizon LQ control of linear systems in detail. To illustrate the approach, an example of the pitch attitude control of an aircraft is considered. We also compare our proposed approach to several other approaches to finite/infinite-horizon LQ control problems with unknown dynamics from the literature, including extremum seeking and adaptive dynamic programming/reinforcement learning. Our proposed approach is competitive with these approaches in speed of convergence and in implementation and computational complexity.

KEYWORDS

LQ control, optimal control, unknown system

1 | INTRODUCTION

An optimal control problem typically involves constructing an open-loop control function or a closed-loop/feedback control policy that minimizes a specified cost function subject to the dynamics of the system to be controlled.[1] A state-space model of the system in the form of differential or difference equations is typically used to represent the dynamic coupling between control inputs and system outputs, imposed as a constraint when control optimization is performed. Such a model may be derived based on physics or obtained via system identification.[2]

In practical applications, the system dynamics may be uncertain or evolve over time, due to, for instance, (i) the system operating point changing, such as the trim condition of an aircraft; (ii) some unmodeled factors that influence the system dynamics changing, such as weather or ambient temperature and humidity for an automotive engine; or (iii) the system parameters changing, eg, as a result of aging or part-to-part variability. Such evolution in dynamics may cause the model used in the derivation of the control strategy to become inaccurate and degrade the control performance. In particular, a control strategy may be designed based on a linearization about an operating point of a nonlinear model. When the system operating point changes, the original linearization becomes inaccurate, and the original control may no longer provide satisfactory performance. While gain scheduling/gain interpolation may be used, scheduling on all possible parameters that can cause variability is often impractical, as these parameters may be numerous and some of them may not be measured.

To cope with the lack of accurate models or the changes in dynamics, control techniques such as adaptive control[3] and robust optimal control[4] can be applied. The goal of adaptive control is, in principle, different from that of optimal control. On the one hand, adaptive control usually concerns itself with the adaptation of a fixed-form controller to initially uncertain system parameters to achieve typical control requirements such as stabilization and reference tracking. On the other hand, the goal of optimal control is to find a control that minimizes a specified cost function, while stabilization and reference tracking may be achieved through the cost function design. Robust optimal control pursues such a goal for systems with uncertain parameters: it optimizes the control to minimize the worst cost (in a min-max formulation)[5,6] or the cost expectation (in a probability-weighted-average formulation)[7] over an uncertainty set.

An alternative technique is optimal control for unknown systems, which is also referred to as model-free optimal control, ie, optimal control without the need for prior knowledge of the system dynamics. Differently from the problem setting of robust optimal control, model-free optimal control typically pursues control optimization to minimize the true cost associated with a given but unknown system.

Model-free optimal control approaches relying on learning and dynamic programming have been developed, for example, in the works of Dierks et al,[8] Lewis and Vamvoudakis,[9] and Wang et al.[10] In the work of Dierks et al,[8] a neural network is used to learn the plant dynamics online; then, an adaptive dynamic programming (ADP) algorithm is exploited to obtain an optimal control law off-line based on the learned neural network model. In the work of Lewis and Vamvoudakis,[9] policy iteration and value iteration reinforcement learning (RL) algorithms are developed to learn an optimal controller off-line from a sufficiently rich set of measured input/output data. In the work of Wang et al,[10] three neural networks are used to identify the plant model, approximate the value function and its derivative, and compute the control, respectively, based on an actor-critic scheme. The works of Lewis and Vrabie[11] and Recht[12] provide surveys on the application of ADP and RL for feedback control. Such approaches attempt to approximately solve the Hamilton-Jacobi-Bellman equation and create an optimal feedback policy. Their scalability may be limited due to the "curse of dimensionality."[13]

Alternatively, a known low-complexity model that approximates the dynamics of a possibly unknown high-complexity system with an error may be used to solve for the control, where the low-complexity model and the control solution are iteratively refined to improve their matches to the original system and the optimal control, respectively.[14] A possible option for the low-complexity model is a linear time-varying model defined in a neighborhood of the current state-and-input trajectory pair. It can be identified through perturbation-based sensitivity analysis and then used to estimate gradient information for updating control to decrease cost.[15,16]

Linear quadratic (LQ) optimal control is one of the most fundamental problems in optimal control theory and practice. The control of many engineering systems can be formulated as a finite-horizon or an infinite-horizon LQ control problem.[17] For infinite-horizon LQ problems, due to the fact that the optimal solution, as a state-feedback policy, and the Bellman value,[13] as a function of the states, are time invariant, approaches based on model-free policy/value iteration RL can be effective. This route has been pursued in other works.[18-23] On the other hand, many practical tasks, such as spacecraft landing or docking, involve maneuvers over finite-time duration, which lead to optimal control problems defined over a finite horizon. Fewer results exist for the finite-horizon LQ control of unknown systems. Approaches based on ADP and RL have recently been investigated in the works of Zhao et al[24] and Fong et al[25]; an approach based on multiparameter extremum seeking (ES)[26] has been proposed in the work of Frihauf et al[27] and extended to handle measurement noise in the work of Liu et al.[28]

In this paper, we also consider the LQ optimal control of unknown discrete-time linear systems. Our contributions include the following. (i) We propose a novel iterative approach that can be applied to both finite-horizon and infinite-horizon LQ problems. (ii) Our approach involves simultaneous updates in the estimates of unknown parameters of a model, which represents the system dynamics, and in the control. A similar-in-spirit but different-in-detail strategy based on the interplay between system identification and control optimization is discussed in the works of Dean et al,[29,30] where only infinite-horizon LQ problems are considered. On the other hand, our approach can also be applied to finite-horizon LQ problems. Furthermore, the updates in our approach rely only on input and cost measurements, ie, our approach does not rely on full-state measurements. (iii) Probing signals are usually needed in model-free optimal control approaches.[8,10,18-25,27,28] On the other hand, our approach can achieve a significant cost decrease even without using probing signals, especially when applied to finite-horizon LQ problems. (iv) Our approach is easy to understand and implement, and it exhibits fast convergence and low computational complexity.

This paper is organized as follows. In Section 2, we describe a problem of minimizing a partially unknown function and propose an iterative approach to treat the problem. In Section 3, we apply the approach to the finite-horizon LQ optimal control of unknown discrete-time linear systems; in addition, we present an algorithm based on ES for comparison. In Section 4, we apply the approach to the infinite-horizon LQ optimal control of unknown discrete-time linear systems; in addition, we present an algorithm based on RL and an algorithm based on ES for comparison. In Section 5, we illustrate the approach with an example representing the pitch attitude control of an aircraft and compare its performance to the other algorithms. This paper is concluded in Section 6.

The notations used in this paper are standard. $\mathbb{R}^{m_1\times m_2}$ represents the set of $m_1 \times m_2$ matrices with real entries; $\mathbb{Z}_{\geq m}$ represents the set of integers that are no less than m. For a scalar twice-continuously-differentiable real-valued function $\beta(\alpha^1, \alpha^2) \in C^2(\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}, \mathbb{R})$, where $\alpha^1 \in \mathbb{R}^{n_1}$ and $\alpha^2 \in \mathbb{R}^{n_2}$, the operator ∇ is defined as

$$\nabla_{\alpha^i}\beta = \frac{\partial \beta}{\partial \alpha^i} = \left[\frac{\partial \beta}{\partial \alpha^i_1}, \frac{\partial \beta}{\partial \alpha^i_2}, \dots, \frac{\partial \beta}{\partial \alpha^i_{n_i}}\right] \in \mathbb{R}^{1\times n_i},$$

$$\nabla_{\alpha^i,\alpha^j}\beta = \nabla_{\alpha^j}(\nabla_{\alpha^i}\beta)^\top = \left[\nabla^\top_{\alpha^j}\left(\frac{\partial \beta}{\partial \alpha^i_1}\right), \nabla^\top_{\alpha^j}\left(\frac{\partial \beta}{\partial \alpha^i_2}\right), \dots, \nabla^\top_{\alpha^j}\left(\frac{\partial \beta}{\partial \alpha^i_{n_i}}\right)\right]^\top \in \mathbb{R}^{n_i\times n_j},$$

where $i \in \{1, 2\}, j \in \{1, 2\}$, and α^i_k denotes the kth component of α^i. For a matrix $M \in \mathbb{R}^{m_1\times m_2}$, the operator $\mathrm{vec}(M)$ is defined by stacking the columns of M, ie,

$$\mathrm{vec}(M) = \mathrm{vec}\left(\left[M_1, M_2, \dots, M_{m_2}\right]\right) = \left[M_1^\top, M_2^\top, \dots, M_{m_2}^\top\right]^\top \in \mathbb{R}^{m_1 m_2},$$

where M_i denotes the ith column of M. We also define the operator $\mathrm{vec}^{-1}(\cdot, \mathbb{R}^{m_1\times m_2}) : \mathbb{R}^{m_1 m_2} \to \mathbb{R}^{m_1\times m_2}$ as the inverse of $\mathrm{vec}(\cdot)$, such that $\mathrm{vec}^{-1}(\mathrm{vec}(M), \mathbb{R}^{m_1\times m_2}) = M$. When without ambiguity, we simply write $\mathrm{vec}^{-1}(\cdot)$. For a matrix pair (M, N), the operator $M \otimes N$ is defined as the Kronecker product of M and N.[31] For a symmetric matrix with real entries $M = M^\top$, $\lambda_{\min}(M)$ (or $\lambda_{\max}(M)$) represents the smallest (or largest) eigenvalue of M. Furthermore, we use $\|\cdot\|$ to represent the vector 2-norm and its induced matrix norm. I_n represents the identity matrix of $\mathbb{R}^{n\times n}$.

2 | PROBLEM FORMULATION AND METHODOLOGY

2.1 | Problem formulation

We describe our approach to the LQ optimal control of unknown discrete-time linear systems by first considering the following optimization problem:

$$\min_x f(x, v^*), \tag{1}$$

where the function $f \in C^2(\mathbb{R}^{n_x} \times \mathbb{R}^{n_v}, \mathbb{R})$, that is, $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}$ is a twice-continuously-differentiable real-valued function of the variable $x \in \mathbb{R}^{n_x}$ and the parameter $v^* \in \mathbb{R}^{n_v}$.

We make the following assumptions.

Assumption 1. The true value of $v^* \in \mathbb{R}^{n_v}$ is unknown.

Assumption 2. For each value of $x \in \mathbb{R}^{n_x}$, the value of $f(x, v^*)$ can be measured.

Assumption 3. For each value of $\bar{v} \in \mathbb{R}^{n_v}$, the minimizer

$$\bar{x} \in \operatorname*{argmin}_x f(x, \bar{v}) \tag{2}$$

exists and can be computed. In particular, the minimizer $\bar{x}$ satisfies the first-order and second-order necessary conditions for optimality, ie,

$$\nabla_x f(\bar{x}, \bar{v}) = 0, \quad \nabla_{x,x} f(\bar{x}, \bar{v}) \succeq 0. \tag{3}$$

Assumptions 1, 2, and 3 characterize the function f as a gray-box–type system—only the structure of the system is known, but the parameters of the system are unknown. Such a problem formulation covers a range of practical situations. For instance, it may represent the control optimization for an unknown dynamic system, where the variable x represents the sequence of control inputs or the parameters of a feedback law to be optimized, and the parameter v^* represents the parameters of a model that reflects the dynamic coupling between inputs and outputs. Details are discussed in Sections 3 and 4.

A common approach to treat the above problem involves first identifying the value of v^* (eg, by sampling inputs x, measuring outputs $f(x, v^*)$, and exploiting known information about f to estimate v^*) and then solving the optimization problem (1) with the identified value of v^* substituted in. Depending on the application, this sequential identification and optimization process, when used for control, can have several potential drawbacks. Firstly, both stages of the process need to be repeated when parameters of the system change due to a change in operating conditions, aging, etc. This can entail a significant effort and may require the system to be temporarily decommissioned in order to perform the identification. Secondly, sampling inputs in the identification phase may not instantly benefit the optimization and performance improvement until the identification is completed. Thirdly, the inputs used for identification, as well as the conditions under which the identification is performed, and those occurring during the system online operation may be different. This can cause situations where the identified v^* does not translate into good control performance. For these and other reasons, alternative procedures, including those that perform optimization without the need of knowing v^* [26-28] and those that perform identification and optimization concurrently,[10,29,30] are of interest.

2.2 | Procedure for minimizing a partially unknown function

In this paper, we propose an iterative procedure to solve problem (1) under Assumptions 1, 2, and 3. Our approach involves concurrently updating the estimate of $x^* \in \operatorname{argmin}_x f(x, v^*)$ and the estimate of v^*.

Let (x^k, v^k) denote the estimate of (x^*, v^*) at the kth iteration. At the $(k+1)$st iteration, we first minimize $f(x, v^k)$ according to the first-order and second-order necessary conditions for optimality (3), ie, based on the conditions

$$\nabla_x f\left(x^{k+1}, v^k\right) = 0, \quad \nabla_{x,x} f\left(x^{k+1}, v^k\right) \geq 0, \tag{4}$$

to obtain the minimizer x^{k+1} as the updated estimate of x^*.

We then update the estimate of v^*. The Taylor expansion of f at (x^{k+1}, v^k) to the second order yields

$$\begin{aligned} f\left(x^{k+1}, v^*\right) = {} & f\left(x^{k+1}, v^k\right) + \nabla_v f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) \\ & + \frac{1}{2}\left(v^* - v^k\right)^\top \nabla_{v,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) + H.O.T., \end{aligned} \tag{5}$$

$$\begin{aligned} f\left(x^k, v^*\right) = {} & f\left(x^{k+1}, v^k\right) + \nabla_x f\left(x^{k+1}, v^k\right)\left(x^k - x^{k+1}\right) + \nabla_v f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) \\ & + \frac{1}{2}\begin{bmatrix} x^k - x^{k+1} \\ v^* - v^k \end{bmatrix}^\top \begin{bmatrix} \nabla_{x,x} f\left(x^{k+1}, v^k\right) & \nabla_{x,v} f\left(x^{k+1}, v^k\right) \\ \nabla_{x,v}^\top f\left(x^{k+1}, v^k\right) & \nabla_{v,v} f\left(x^{k+1}, v^k\right) \end{bmatrix} \begin{bmatrix} x^k - x^{k+1} \\ v^* - v^k \end{bmatrix} + H.O.T., \end{aligned} \tag{6}$$

where $H.O.T.$ stands for "higher-order terms."

Combining (4), (5), and (6), we obtain

$$\begin{aligned} f\left(x^{k+1}, v^*\right) - f\left(x^k, v^*\right) = {} & \left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) \\ & - \frac{1}{2}\left(x^{k+1} - x^k\right)^\top \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right) + H.O.T. \end{aligned} \tag{7}$$

In particular, the $H.O.T.$ in the Taylor expansions (5), (6), and (7) are characterized by

$$H.O.T. = O\left(\left\| \begin{bmatrix} x^{k+1} - x^k \\ v^* - v^k \end{bmatrix} \right\|^3\right). \tag{8}$$

We now treat the $H.O.T.$ in (7) as some unknown noise $w^{k+1} \in \mathbb{R}$ and obtain

$$\begin{aligned} f\left(x^{k+1}, v^*\right) - f\left(x^k, v^*\right) = {} & \left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) \\ & - \frac{1}{2}\left(x^{k+1} - x^k\right)^\top \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right) + w^{k+1}, \end{aligned} \tag{9}$$

which is a linear equation in v^*.

Assumption 4. There exists $\epsilon \geq 0$ such that $|w^{k+1}| \leq \epsilon$, for all $k \in \mathbb{Z}_{\geq 0}$.

We note that ϵ in Assumption 4 is typically a small number. This is reasonable when (i) x^{k+1} is sufficiently close to x^k, eg, through a sufficiently small step size value in the update of the estimate of v^*, and (ii) v^k is sufficiently close to v^*, eg, through a sufficiently good initial guess v^0. We also note that when f is, at most, quadratically dependent on (x, v^*), $\epsilon = 0$.

We write (9) as

$$\zeta^{k+1} = \phi^{k+1} v^* + w^{k+1}, \tag{10}$$

where

$$\phi^{k+1} = \left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right) \in \mathbb{R}^{1\times n_v}, \tag{11}$$

$$\zeta^{k+1} = \zeta_1^{k+1} + \zeta_2^{k+1} \in \mathbb{R}, \tag{12}$$

$$\zeta_1^{k+1} = f\left(x^{k+1}, v^*\right) - f\left(x^k, v^*\right) + \frac{1}{2}\left(x^{k+1} - x^k\right)^\top \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right) \in \mathbb{R}, \tag{13}$$

$$\zeta_2^{k+1} = \left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right) v^k = \phi^{k+1} v^k \in \mathbb{R}. \tag{14}$$

Define

$$\Phi^{k+1} = \begin{bmatrix} \phi^1 \\ \vdots \\ \phi^{k+1} \end{bmatrix}, \quad Z^{k+1} = \begin{bmatrix} \zeta^1 \\ \vdots \\ \zeta^{k+1} \end{bmatrix}, \quad W^{k+1} = \begin{bmatrix} w^1 \\ \vdots \\ w^{k+1} \end{bmatrix}. \tag{15}$$

The least squares estimate of v^* is

$$v^{k+1} = \left(\left(\Phi^{k+1}\right)^\top \Phi^{k+1}\right)^{-1} \left(\Phi^{k+1}\right)^\top Z^{k+1}, \tag{16}$$

assuming that $(\Phi^{k+1})^\top \Phi^{k+1}$ is of full rank.

The least squares estimate (16) can be computed recursively using the recursive least squares (RLS) algorithm, ie,

$$\Pi^{k+1} = \Pi^k - \frac{\Pi^k \left(\phi^{k+1}\right)^\top \phi^{k+1} \Pi^k}{1 + \phi^{k+1} \Pi^k \left(\phi^{k+1}\right)^\top}, \tag{17}$$

$$v^{k+1} = v^k + \lambda^{k+1} \Pi^{k+1} \left(\phi^{k+1}\right)^\top \left(\zeta^{k+1} - \phi^{k+1} v^k\right), \tag{18}$$

where $\lambda^{k+1} \in [0, 1]$ is an update step size, and $\Pi^k = \left((\Phi^k)^\top \Phi^k\right)^{-1} \in \mathbb{R}^{n_v \times n_v}$.

After both the updated estimate of x^*, x^{k+1}, and the updated estimate of v^*, v^{k+1}, are obtained, we let $k \leftarrow k + 1$ and proceed with the next iteration.

We now describe the convergence properties of the iterates $\{v^k\}_{k=0}^\infty$.

Proposition 1. *Suppose that the update step size λ^{k+1} is selected to satisfy*

$$0 \le \lambda^{k+1} \le \min\left(\frac{\Gamma(1-\gamma)^k}{\left\|\Pi^{k+1}\left(\phi^{k+1}\right)^\top\right\| \left|\zeta_1^{k+1}\right|}, 1\right), \tag{19}$$

for all $k \in \mathbb{Z}_{\ge 0}$, where $\Gamma > 0$ and $\gamma \in (0, 1)$ are design parameters. Then, $\{v^k\}_{k=0}^\infty$ converges, ie, there exists $v^\infty \in \mathbb{R}^{n_v}$ such that

$$\lim_{k\to\infty} v^k = v^\infty. \tag{20}$$

Proof. Equation (18) can be written as

$$\begin{aligned} v^{k+1} - v^k &= \lambda^{k+1} \Pi^{k+1} \left(\phi^{k+1}\right)^\top \left(\zeta^{k+1} - \phi^{k+1} v^k\right) \\ &= \lambda^{k+1} \Pi^{k+1} \left(\phi^{k+1}\right)^\top \zeta_1^{k+1}. \end{aligned} \tag{21}$$

By (19), we obtain

$$\left\|v^{k+1} - v^k\right\| \le \lambda^{k+1} \left\|\Pi^{k+1}\left(\phi^{k+1}\right)^\top\right\| \left|\zeta_1^{k+1}\right| \le \Gamma(1-\gamma)^k. \tag{22}$$

Then, the series of nonnegative terms, ie,

$$\sum_{k=0}^\infty \left\|v^{k+1} - v^k\right\| \le \Gamma \sum_{k=0}^\infty (1-\gamma)^k = \frac{\Gamma}{\gamma} < \infty, \tag{23}$$

is convergent, and as a result, for any $\epsilon > 0$, there exists $k^* \in \mathbb{Z}_{\ge 0}$ such that

$$\sum_{k=k^*}^\infty \left\|v^{k+1} - v^k\right\| \le \epsilon. \tag{24}$$

Then, by the triangle inequality, for any $k_1 \in \mathbb{Z}_{\geq k^*}$ and any $k_2 \in \mathbb{Z}_{\geq k_1}$, we have

$$\left\| v^{k_2} - v^{k_1} \right\| \leq \sum_{k=k_1}^{k_2-1} \left\| v^{k+1} - v^k \right\| \leq \sum_{k=k^*}^{\infty} \left\| v^{k+1} - v^k \right\| \leq \epsilon. \tag{25}$$

That is, we get

$$\lim_{k_1,k_2\to\infty} \left\| v^{k_2} - v^{k_1} \right\| = 0. \tag{26}$$

Therefore, $\{v^k\}_{k=0}^{\infty}$ is a Cauchy sequence in $\mathbb{R}^{n_v}$ and, thus, converges. □

Remark 1. In the implementation, a constant update step size $\lambda^{k+1} = \lambda \in [0, 1]$ for all $k \in \mathbb{Z}_{\geq 0}$ is used, which usually achieves the convergence of the iterates. Note that such a constant step size can satisfy (19) over an arbitrarily large number of iterations if $\Gamma > 0$ is selected to be sufficiently large and $\gamma \in (0, 1)$ is selected to be sufficiently small. The feasibility of using a constant update step size is also supported by the analysis of the robustness of our approach to step size selection through simulations in Section 5.1.

We next provide an error estimate of our approach for the case when the step size is equal to 1.

Proposition 2. *Suppose that (i) $\lambda^{k+1} = 1$ for all $k \in \mathbb{Z}_{\geq 0}$ and (ii) Assumption 4 holds. Then, (I) the estimation error of v^* at the kth iteration is bounded by*

$$\left\| v^k - v^* \right\| \leq \sqrt{\frac{k}{\lambda_{\min}\left(\left(\Phi^k\right)^\top \Phi^k\right)}}\, \varepsilon, \tag{27}$$

where the right-hand side is unbounded if $(\Phi^k)^\top \Phi^k$ is not of full rank.

Suppose further that (iii) in the Taylor expansion of $\nabla_x f$ at (x^{k+1}, v^k) to the first order,

$$\nabla_x^\top f(x^*, v^*) = \nabla_x^\top f\left(x^{k+1}, v^k\right) + \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^* - x^{k+1}\right) + \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) + \tilde{w}^{k+1}, \tag{28}$$

where $\tilde{w}^{k+1} \in \mathbb{R}^{n_x}$ and is characterized by $\tilde{w}^{k+1} = O\left(\left\| \begin{bmatrix} x^* - x^{k+1} \\ v^* - v^k \end{bmatrix} \right\|^2\right)$, it holds that $\|\tilde{w}^{k+1}\| \leq \delta$, for all $k \in \mathbb{Z}_{\geq 0}$. Then, (II) the estimation error of x^* at the $(k+1)$st iteration is bounded by

$$\left\| x^{k+1} - x^* \right\| \leq \left\| \left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1} \nabla_{x,v} f\left(x^{k+1}, v^k\right) \Pi^k \left(\Phi^k\right)^\top \right\| \sqrt{k}\varepsilon + \left\| \left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1} \right\| \delta, \tag{29}$$

where the right-hand side is unbounded if $\nabla_{x,x} f(x^{k+1}, v^k)$ is not of full rank.

Suppose further that (iv) there exists $\xi > 0$, such that $|f(x_1, v^*) - f(x_2, v^*)| \leq \xi \|x_1 - x_2\|$, for all $x_1, x_2 \in \mathbb{R}^{n_x}$. Then, (III)

$$\begin{aligned} &\left| f\left(x^{k+1}, v^*\right) - f(x^*, v^*) \right| \leq \\ &\xi \left(\left\| \left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1} \nabla_{x,v} f\left(x^{k+1}, v^k\right) \Pi^k \left(\Phi^k\right)^\top \right\| \sqrt{k}\varepsilon + \left\| \left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1} \right\| \delta \right). \end{aligned} \tag{30}$$

Proof. (I) v^k and v^* respectively satisfy

$$\left(\left(\Phi^k\right)^\top \Phi^k\right) v^k = \left(\Phi^k\right)^\top Z^k, \tag{31}$$

$$\left(\left(\Phi^k\right)^\top \Phi^k\right) v^* = \left(\Phi^k\right)^\top \left(Z^k - W^k\right). \tag{32}$$

Then, we have

$$\left(\left(\Phi^k\right)^\top \Phi^k\right)\left(v^k - v^*\right) = \left(\Phi^k\right)^\top W^k. \tag{33}$$

Multiply by $(v^k - v^*)^\top$ on both sides to obtain

$$\left\| \Phi^k \left(v^k - v^*\right) \right\|^2 = \left(\Phi^k \left(v^k - v^*\right)\right)^\top W^k. \tag{34}$$

By the Cauchy-Schwarz inequality, we have

$$\left\| \Phi^k \left(v^k - v^*\right) \right\|^2 = \left(\Phi^k \left(v^k - v^*\right)\right)^\top W^k \leq \left\| \Phi^k \left(v^k - v^*\right) \right\| \left\| W^k \right\|. \tag{35}$$

Then, by Assumption 4, we obtain

$$\left\|\Phi^k\left(v^k - v^*\right)\right\| \leq \left\|W^k\right\| = \sqrt{\sum_{t=1}^{k}(w^t)^2} \leq \sqrt{k\varepsilon^2} = \sqrt{k}\varepsilon. \tag{36}$$

The left-hand side can be bounded by

$$\sqrt{\lambda_{\min}\left(\left(\Phi^k\right)^\top \Phi^k\right)}\left\|v^k - v^*\right\| \leq \left\|\Phi^k\left(v^k - v^*\right)\right\|. \tag{37}$$

Therefore, we get

$$\left\|v^k - v^*\right\| \leq \sqrt{\frac{k}{\lambda_{\min}\left(\left(\Phi^k\right)^\top \Phi^k\right)}}\varepsilon. \tag{38}$$

(II) By the first-order necessary conditions for optimality (3) and (4) and Equations (28) and (33), we have

$$\begin{aligned}\nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^*\right) &= -\nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^k - v^*\right) + \tilde{w}^{k+1} \\ &= -\nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(\left(\Phi^k\right)^\top \Phi^k\right)^{-1}\left(\Phi^k\right)^\top W^k + \tilde{w}^{k+1} \\ &= -\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^k\left(\Phi^k\right)^\top W^k + \tilde{w}^{k+1}.\end{aligned} \tag{39}$$

Assume that $\nabla_{x,x} f(x^{k+1}, v^k)$ is of full rank, then

$$x^{k+1} - x^* = -\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\left(\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^k\left(\Phi^k\right)^\top W^k - \tilde{w}^{k+1}\right). \tag{40}$$

By (ii) and (iii), we obtain

$$\begin{aligned}\left\|x^{k+1} - x^*\right\| &\leq \left\|\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^k\left(\Phi^k\right)^\top\right\|\left\|W^k\right\| + \left\|\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\right\|\left\|\tilde{w}^{k+1}\right\| \\ &\leq \left\|\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^k\left(\Phi^k\right)^\top\right\|\sqrt{k}\varepsilon + \left\|\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\right\|\delta.\end{aligned} \tag{41}$$

Then, (III) follows from (II) and (iv). □

Remark 2. As the number of data points, k, increases in the RLS algorithm, it usually holds that $\lambda_{\min}((\Phi^k)^\top \Phi^k)$ increases and goes to infinity, whereas $\|\Pi^k\|$ decreases and goes to zero. If $k/\lambda_{\min}((\Phi^k)^\top \Phi^k)$ decreases as k increases, the error bound (27) shrinks, which can be monitored at run time. Similarly, the growth or decrease of the error bounds (29) and (30) can be monitored at run time. The bounds (27), (29), and (30) can be used to define the criteria to terminate the iterations. For instance, if there exists $k^* \in \mathbb{Z}_{\geq 0}$ such that $\|v^{k^*} - v^*\| \leq \eta_1$, or $\|x^{k^*} - x^*\| \leq \eta_2$, or $|f(x^{k^*}, v^*) - f(x^*, v^*)| \leq \eta_3$, where $\eta_1, \eta_2, \eta_3 > 0$ are user specified, then iterations terminate at k^*. Otherwise, iterations terminate after the maximum number of iterations is reached.

Sometimes, a monotone nonincrease in the cost values $\left\{f(x^{k+1}, v^*)\right\}_{k\in\mathbb{Z}_{\geq 0}}$ during the iterations is desired. For the case of control, this may facilitate maintenance of stability, supposing the initial control is stabilizing. In what follows, we provide a guideline for selecting the update step size so that the cost does not increase after the current iteration, which can be checked before the iteration is performed.

The guideline is developed based on several approximations. In particular, we make the following assumptions.

Assumption 5.

1. In (9), $w^{k+1} = 0$.
2. In the following equation obtained based on the Taylor expansion of f at (x^{k+1}, v^k) to the second order,

$$\begin{aligned}f\left(x^{k+2}, v^*\right) - f\left(x^{k+1}, v^*\right) = &\left(x^{k+2} - x^{k+1}\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) \\ &+ \frac{1}{2}\left(x^{k+2} - x^{k+1}\right)^\top \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+2} - x^{k+1}\right) + \bar{w}^{k+1},\end{aligned} \tag{42}$$

the higher-order terms $\bar{w}^{k+1} = 0$.

3. In the Taylor expansion of $\nabla_x f$ at (x^{k+1}, v^k) to the first order,

$$\nabla_x^\top f\left(x^{k+2}, v^{k+1}\right) = \nabla_x^\top f\left(x^{k+1}, v^k\right) + \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+2} - x^{k+1}\right) + \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^{k+1} - v^k\right) + \hat{w}^{k+1}, \tag{43}$$

the higher-order terms $\hat{w}^{k+1} = 0$.

Proposition 3. *Suppose that Assumption 5 holds. Then, if the update step size* $\lambda^{k+1} \in [0, 1]$ *is selected such that*

$$\Sigma_1^{k+1} - \lambda^{k+1}\Sigma_2^{k+1} \geq 0, \tag{44}$$

where

$$\begin{aligned}\Sigma_1^{k+1} = &\Big(\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\\ &\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Big) + \Big(\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top\\ &\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Big)^\top,\end{aligned} \tag{45}$$

$$\begin{aligned}\Sigma_2^{k+1} = &\left(\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\right)\Big(\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\\ &\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^{k+1}\Big)\left(\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\right),\end{aligned} \tag{46}$$

then the cost does not increase after the $(k+1)$*st iteration, ie,*

$$f\left(x^{k+2}, v^*\right) \leq f\left(x^{k+1}, v^*\right). \tag{47}$$

Proof. By the first-order necessary conditions for optimality $\nabla_x f(x^{k+2}, v^{k+1}) = \nabla_x f(x^{k+1}, v^k) = 0$, (43) yields

$$x^{k+2} - x^{k+1} = -\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^{k+1} - v^k\right). \tag{48}$$

Substituting (48) into (42), we get

$$\begin{aligned}f\left(x^{k+2}, v^*\right) - f\left(x^{k+1}, v^*\right) = &-\left(v^{k+1} - v^k\right)^\top \nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right)\\ &+ \frac{1}{2}\left(v^{k+1} - v^k\right)^\top \nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^{k+1} - v^k\right).\end{aligned} \tag{49}$$

The update law (18) yields

$$\begin{aligned}v^{k+1} - v^k = &\lambda^{k+1}\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\\ &\left(f\left(x^{k+1}, v^*\right) - f\left(x^k, v^*\right) + \frac{1}{2}\left(x^{k+1} - x^k\right)^\top \nabla_{x,x} f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\right).\end{aligned} \tag{50}$$

Substituting (9) with $w^{k+1} = 0$ into (50) yields

$$v^{k+1} - v^k = \lambda^{k+1}\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right). \tag{51}$$

Substituting (51) into (49), we obtain

$$\begin{aligned}&f\left(x^{k+2}, v^*\right) - f\left(x^{k+1}, v^*\right)\\ &= -\lambda^{k+1}\left(v^* - v^k\right)^\top \nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^{k+1}\\ &\quad \nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right) + \frac{\left(\lambda^{k+1}\right)^2}{2}\left(v^* - v^k\right)^\top\\ &\quad \nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top \nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\\ &\quad \left(\nabla_{x,x} f\left(x^{k+1}, v^k\right)\right)^{-1}\nabla_{x,v} f\left(x^{k+1}, v^k\right)\Pi^{k+1}\nabla_{x,v}^\top f\left(x^{k+1}, v^k\right)\left(x^{k+1} - x^k\right)\left(x^{k+1} - x^k\right)^\top\\ &\quad \nabla_{x,v} f\left(x^{k+1}, v^k\right)\left(v^* - v^k\right)\\ &= -\frac{\lambda^{k+1}}{2}\left(v^* - v^k\right)^\top\left(\Sigma_1^{k+1} - \lambda^{k+1}\Sigma_2^{k+1}\right)\left(v^* - v^k\right),\end{aligned} \tag{52}$$

where Σ_1^{k+1} and Σ_2^{k+1} are defined in (45) and (46).

If $\Sigma_1^{k+1} - \lambda^{k+1}\Sigma_2^{k+1} \geq 0$, then $f(x^{k+2}, v^*) - f(x^{k+1}, v^*) \leq 0$ for any $v^* \in \mathbb{R}^{n_v}$. □

Note that in expression (52), all parameter values, except for the unknown v^* and the controllable update step size λ^{k+1}, are already known before the update of v^{k+1} at the $(k+1)$st iteration. Therefore, if λ^{k+1} is selected such that (44) holds, which can be checked before the $(k+1)$st iteration is performed, then, under Assumption 5, the cost is guaranteed to not increase after the $(k+1)$st iteration.

In practice, a monotone nonincrease in the cost values $\left\{f(x^{k+1}, v^*)\right\}_{k\in\mathbb{Z}_{\geq 0}}$ can be achieved by solving the following linear matrix inequality problem[32]:

$$\begin{aligned} &\text{maximize:} && \lambda \in [0,1], \\ &\text{subject to:} && \Sigma_1^{k+1} - \lambda \Sigma_2^{k+1} \geq 0, \end{aligned} \tag{53}$$

and set $\lambda = 0$ if the constraint is infeasible, to select the update step size λ^{k+1}, for each $k \in \mathbb{Z}_{\geq 0}$.

Remark 3. If the update step sizes λ^{k+1} are selected by solving (53) for all $k \in \mathbb{Z}_{\geq 0}$, then, under Assumption 5, the sequence $\left\{f(x^{k+1}, v^*)\right\}_{k\in\mathbb{Z}_{\geq 0}}$ is monotone nonincreasing. Since $\left\{f(x^{k+1}, v^*)\right\}_{k\in\mathbb{Z}_{\geq 0}}$ is also lower bounded by $f(x^*, v^*)$, by the monotone convergence theorem, $\left\{f(x^{k+1}, v^*)\right\}_{k\in\mathbb{Z}_{\geq 0}}$ converges. This provides another sufficient condition, aside from Proposition 1, for the convergence of the iterates.

Remark 4. In the iterative procedure, at the $(k+1)$st iteration, $k \in \mathbb{Z}_{\geq 0}$, on the one hand, x^{k+1} is required to satisfy the necessary conditions for optimality (4); on the other hand, x^k can be arbitrary. Therefore, (i) to initialize the iterative procedure, $k = 0$, we start from an initial guess v^0, compute $x^1 \in \operatorname{argmin}_x f(x, v^0)$, and then create x^0 by adding a small perturbation to x^1. (ii) At the $(k+1)$st iteration, $k \in \mathbb{Z}_{\geq 1}$, we may add a small random perturbation $e^k \in \mathbb{R}^{n_x}$ to the x^k obtained from the kth iteration and use the perturbed $x^k \leftarrow x^k + e^k$ at the $(k+1)$st iteration. Such perturbations play the role of probing signals and generate a persistency of excitation[33] to the RLS algorithm, to promote the RLS estimate to approach v^*.[34,35] We have found that such perturbations are not necessary in the application of our approach to finite-horizon LQ problems, but can benefit the application of our approach to infinite-horizon LQ problems. Further details can be found in Sections 3 and 4.

2.3 | Examples

2.3.1 | Minimization of an unknown quadratic function

Consider the minimization of the following function:

$$f(x, v^*) = ax^2 + bx + 1, \quad v^* = (a, b) = (1, -2). \tag{54}$$

Clearly, the minimizer is $x^* = 1$, and the corresponding function value is $f(x^*, v^*) = 0$.

Suppose we do not know $v^* = (a, b)$. We can use our proposed approach to treat this problem. In particular, we start from an initial guess $v^0 = (a^0, b^0) = (2, 4)$. At the $(k+1)$st iteration, the minimizer x^{k+1} is computed from $x^{k+1} = -b^k/(2a^k)$; the update step size λ^{k+1} is selected by solving the linear matrix inequality problem (53). The evolutions of x^{k+1} and $f(x^{k+1}, v^*)$ over the iterations are plotted in Figure 1A,B. It can be observed that x^{k+1} converges to $x^* = 1$ and $f(x^{k+1}, v^*)$ converges to $f(x^*, v^*) = 0$. Furthermore, we plot the history of λ^{k+1} and the history of the smallest eigenvalue of $\Sigma_1^{k+1} - \lambda^{k+1}\Sigma_2^{k+1}$ over the iterations in Figure 1C. In this example, the solution to (53) is 1 for all $k \in \mathbb{Z}_{\geq 0}$, which can be observed from Figure 1C.1, and the matrix $\Sigma_1^{k+1} - \lambda^{k+1}\Sigma_2^{k+1}$ is positive semidefinite for all $k \in \mathbb{Z}_{\geq 0}$, which can be observed from Figure 1C.2. This guarantees the monotone nonincrease in the cost values $\left\{f(x^{k+1}, v^*)\right\}_{k\in\mathbb{Z}_{\geq 0}}$, which is verified in Figure 1B.

2.3.2 | Minimization of the Rosenbrock function with unknown parameters

The second example we consider is minimizing the Rosenbrock function, ie,

$$f(x, v^*) = (a - x_1)^2 + b\left(x_2 - x_1^2\right)^2, \quad x = (x_1, x_2), \quad v^* = (a, b) = (1, 100). \tag{55}$$

The Rosenbrock function is often used as a benchmark problem to test optimization algorithms.[36] Note that it is non-convex and admits a global minimum at $(x_1^*, x_2^*) = (a, a^2)$. We use the Rosenbrock function to show that the procedure described in Section 2.2 can be successful for functions that are not quadratic.

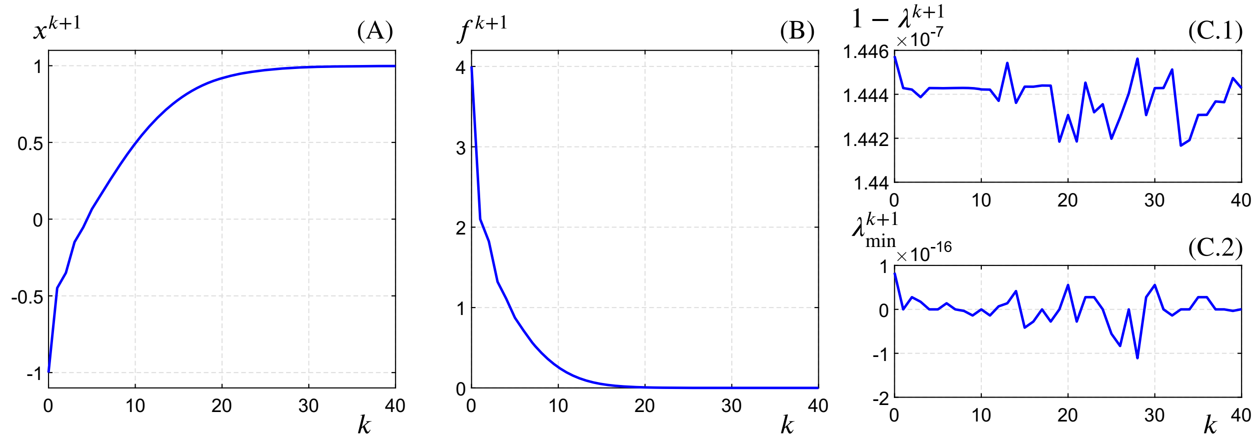

FIGURE 1 Quadratic function minimization. A, The plot of (k, x^{k+1}); B, The plot of $(k, f(x^{k+1}, v^*))$; C, The plots of $(k, 1-\lambda^{k+1})$ and $(k, \lambda_{\min}(\Sigma_1^{k+1} - \lambda^{k+1}\Sigma_2^{k+1}))$ [Colour figure can be viewed at wileyonlinelibrary.com]

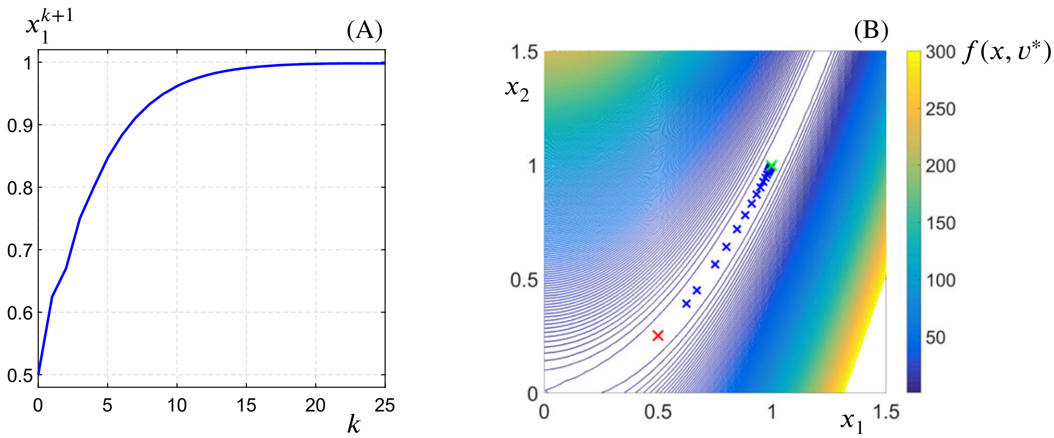

FIGURE 2 Rosenbrock function minimization. A, The plot of (k, x_1^{k+1}); B, The contour of $f(x, v^*)$ and the evolution of x^{k+1}. The red cross indicates $x^1 = (a^0, (a^0)^2)$, and the green cross indicates the solution after 25 iterations [Colour figure can be viewed at wileyonlinelibrary.com]

Suppose we do not know (a, b) and start from an initial guess $(a^0, b^0) = (0.5, 200)$. At the $(k+1)$st iteration, the minimizer is computed from $x^{k+1} = (x_1^{k+1}, x_2^{k+1}) = (a^k, (a^k)^2)$; the update step size is selected as a constant $\lambda^{k+1} = 0.25$ for all $k \in \mathbb{Z}_{\geq 0}$. The evolutions of x_1^{k+1} and $f(x^{k+1}, v^*)$ over the iterations are plotted in Figure 2A,B. It can be observed that x_1^{k+1} converges to the minimizer at $x_1^* = a = 1$ and $f(x^{k+1}, v^*)$ converges to $f(x^*, v^*) = 0$.

3 | FINITE-HORIZON LQ CONTROL OF UNKNOWN DISCRETE-TIME LINEAR SYSTEMS

In Section 2, we have introduced an optimization problem where the objective function f is of a known form but depends on an unknown parameter v^*. We have presented an iterative approach to address this problem and discussed its theoretical properties. In this section, we exploit the proposed approach to treat the finite-horizon LQ optimal control problem for unknown discrete-time systems. Our iterative approach requires multiple evaluations of the cost as a function of control; thus, it is applicable to situations where the cost can be evaluated through simulations subject to the same initial condition, or to batch processes where experiments with the same initial condition can be repeated.[37,38] Such a setting is the same as in the application of ES to the finite-horizon LQ control.[27,28] Therefore, we choose to compare the performance of our approach to that of the ES approach.

3.1 | Iterative approach to finite-horizon LQ control of unknown systems

A finite-horizon discrete-time optimal control problem may be stated as

$$\min_{u_i,\, i=0,\,\dots\,,N-1} f(u_0, u_1, \dots, u_{N-1}, y_1, y_2, \dots, y_N), \tag{56}$$

subject to the dynamic equation

$$y_{i+1} = \mathcal{T}(y_i, u_i) \tag{57}$$

and an initial condition y_0, where $u_i \in \mathbb{R}^{n_u}, i \in \{0, 1, \dots, N-1\}$, denotes the input sequence; $y_i \in \mathbb{R}^{n_y}, i \in \{0, 1, \dots, N\}$, denotes the state sequence; and N is the prediction horizon.

Given an initial condition y_0, the subsequent states, y_i, $i \in \{1, 2, \dots, N\}$, are determined by the input sequence $\{u_0, u_1, \dots, u_{N-1}\}$.

If the dynamic equation (57) is parameterizable, problem (56) may be converted into the following form:

$$\min_{x} f(x, v^*), \tag{58}$$

where $x = [u_0^\top, u_1^\top, \dots, u_{N-1}^\top]^\top$, and v^* represents the parameters of a model that reflects the dynamic coupling between $\{u_0, u_1, \dots, u_{N-1}\}$ and $\{y_0, y_1, \dots, y_N\}$ through (57). Then, the iterative approach presented in Section 2 may be used to solve problem (58).

Now, we discuss the approach to achieve the conversion from (56) and (57) into (58) for the finite-horizon LQ control of discrete-time linear systems.

A discrete-time finite-horizon LQ problem involves minimizing a cost function, ie,

$$\min_{u_i,\, i=0,\,\dots\,,N-1} f = \sum_{i=0}^{N-1} \left(y_{i+1}^\top Q y_{i+1} + u_i^\top R u_i \right), \tag{59}$$

subject to

$$y_{i+1} = A y_i + B u_i, \tag{60}$$

where $Q = Q^\top \geq 0$ and $R = R^\top \succ 0$.

The dynamic equation (60) yields

$$y_i = A^i y_0 + \sum_{j=0}^{i-1} A^{i-j-1} B u_j. \tag{61}$$

Substituting (61) into the objective function (59), we obtain

$$\begin{aligned} f &= \sum_{i=0}^{N-1} \left(\left(A^{i+1} y_0 + \sum_{j=0}^{i} A^{i-j} B u_j \right)^\top Q \left(A^{i+1} y_0 + \sum_{j=0}^{i} A^{i-j} B u_j \right) + u_i^\top R u_i \right) \\ &= \sum_{i=0}^{N-1} \left(y_0^\top \left(A^{i+1}\right)^\top Q A^{i+1} y_0 + 2 y_0^\top \left(A^{i+1}\right)^\top Q \left(\sum_{j=0}^{i} A^{i-j} B u_j \right) \right. \\ &\quad \left. + \left(\sum_{j=0}^{i} A^{i-j} B u_j \right)^\top Q \left(\sum_{j=0}^{i} A^{i-j} B u_j \right) + u_i^\top R u_i \right). \end{aligned} \tag{62}$$

Suppose the matrix pair (A, B) is unknown*; we can list the unknown entries in some order and define an unknown parameter vector, eg,

$$v^* = \left[\text{vec}^\top(A), \text{vec}^\top(B) \right]^\top = \left[a_{11}, a_{21}, \dots, a_{n_y n_y}, b_{11}, b_{21}, \dots, b_{n_y n_u} \right]^\top, \tag{63}$$

where a_{pq} denotes the (p, q)-entry (pth row, qth column) of matrix A and $b_{p'q'}$ denotes the (p', q')-entry of matrix B. Then, problem (59) and (60) is converted into the following form:

$$\min_{x} f(x, v^*), \tag{64}$$

where f is given by (62), $x = \left[u_0^\top, u_1^\top, \dots, u_{N-1}^\top \right]^\top$, and v^* is given by (63). Therefore, we can use the approach in Section 2 to update the estimate of v^* while improving the control sequence $x = \left[u_0^\top, u_1^\top, \dots, u_{N-1}^\top \right]^\top$.

*The case where only some entries of (A, B) are unknown can be treated similarly.

We note that the needed derivatives can be explicitly computed. Specifically, we have

$$\nabla_{u_m} f = 2\sum_{i=m}^{N-1}\left(y_0^\top \left(A^{i+1}\right)^\top Q\left(A^{i-m}B\right) + \left(\sum_{j=0}^{i} A^{i-j}Bu_j\right)^\top Q\left(A^{i-m}B\right)\right) + 2u_m^\top R, \tag{65}$$

$$\nabla_{u_m,u_n} f = \begin{cases} 2\sum_{i=m}^{N-1}\left(\left(A^{i-m}B\right)^\top Q\left(A^{i-n}B\right)\right), & \text{if } n < m, \\ 2\sum_{i=m}^{N-1}\left(\left(A^{i-m}B\right)^\top Q\left(A^{i-m}B\right)\right) + 2R, & \text{if } n = m, \\ 2\sum_{i=n}^{N-1}\left(\left(A^{i-m}B\right)^\top Q\left(A^{i-n}B\right)\right), & \text{if } n > m, \end{cases} \tag{66}$$

$$\begin{aligned} \nabla_{u_m,a_{pq}} f = 2\sum_{i=m}^{N-1}\Bigg(& \left(A^{i-m}B\right)^\top Q\bar{A}_{\{i+1,pq\}}y_0 + \left(\bar{A}_{\{i-m,pq\}}B\right)^\top QA^{i+1}y_0 \\ & + \left(A^{i-m}B\right)^\top Q\left(\sum_{j=0}^{i}\bar{A}_{\{i-j,pq\}}Bu_j\right) + \left(\bar{A}_{\{i-m,pq\}}B\right)^\top Q\left(\sum_{j=0}^{i}A^{i-j}Bu_j\right)\Bigg), \end{aligned} \tag{67}$$

where $\bar{A}_{\{\theta,pq\}}$ is a matrix given by

$$\bar{A}_{\{\theta,pq\}} = \sum_{r=0}^{\theta-1} A^r \left[1_{pq}\right]_A A^{\theta-r-1}, \tag{68}$$

in which $[1_{pq}]_A$ is a matrix of size(A), where the (p,q)-entry is 1 and all other entries are 0. We can obtain a similar expression for $\nabla_{u_m,b_{p'q'}} f$, ie,

$$\begin{aligned} \nabla_{u_m,b_{p'q'}} f = 2\sum_{i=m}^{N-1}\Bigg(& \left(A^{i-m}[1_{p'q'}]_B\right)^\top QA^{i+1}y_0 + \left(A^{i-m}[1_{p'q'}]_B\right)^\top Q\left(\sum_{j=0}^{i}A^{i-j}Bu_j\right) \\ & + \left(A^{i-m}B\right)^\top Q\left(\sum_{j=0}^{i}A^{i-j}[1_{p'q'}]_B u_j\right)\Bigg), \end{aligned} \tag{69}$$

in which $[1_{p'q'}]_B$ is a matrix of size(B), where the (p',q')-entry is 1 and all other entries are 0.

Then, we can formulate the matrices in (9) as

$$\nabla_{x,x} f\left(x^{k+1}, v^k\right) = \left[\nabla_{u_m,u_n} f\right] = \begin{bmatrix} \nabla_{u_0,u_0} f & \nabla_{u_0,u_1} f & \cdots & \nabla_{u_0,u_{N-1}} f \\ \nabla_{u_1,u_0} f & \nabla_{u_1,u_1} f & \cdots & \nabla_{u_1,u_{N-1}} f \\ \vdots & \vdots & \ddots & \vdots \\ \nabla_{u_{N-1},u_0} f & \nabla_{u_{N-1},u_1} f & \cdots & \nabla_{u_{N-1},u_{N-1}} f \end{bmatrix}, \tag{70}$$

$$\nabla_{x,v} f\left(x^{k+1}, v^k\right) = \left[\nabla_{u_m,a_{pq}} f, \nabla_{u_m,b_{p'q'}} f\right] = \begin{bmatrix} \nabla_{u_0,a_{11}} f & \nabla_{u_0,a_{21}} f & \cdots & \nabla_{u_0,a_{n_yn_y}} f & \nabla_{u_0,b_{11}} f & \cdots & \nabla_{u_0,b_{n_yn_u}} f \\ \vdots & \vdots & & \ddots & \ddots & \ddots & \vdots \\ \nabla_{u_{N-1},a_{11}} f & \nabla_{u_{N-1},a_{21}} f & \cdots & \nabla_{u_{N-1},a_{n_yn_y}} f & \nabla_{u_{N-1},b_{11}} f & \cdots & \nabla_{u_{N-1},b_{n_yn_u}} f \end{bmatrix}.$$

The minimizer $x^{k+1} = [(u_0^{k+1})^\top, (u_1^{k+1})^\top, \dots, (u_{N-1}^{k+1})^\top]^\top$ at the $(k+1)$st iteration can be analytically computed by, at first, performing the backward-in-time calculations, ie,

$$\begin{aligned} F_{i-1}^{\{k+1\}} &= -\left(R + (B^{\{k\}})^\top S_i^{\{k+1\}} B^{\{k\}}\right)^{-1} (B^{\{k\}})^\top S_i^{\{k+1\}} A^{\{k\}}, \\ S_{i-1}^{\{k+1\}} &= Q + (A^{\{k\}})^\top S_i^{\{k+1\}} A^{\{k\}} + (A^{\{k\}})^\top S_i^{\{k+1\}} B^{\{k\}} F_{i-1}^{\{k+1\}}, \end{aligned} \tag{71}$$

where $S_N^{\{k+1\}} = Q$ and the matrix pair $(A^{\{k\}}, B^{\{k\}})$ is constructed from v^k, and, then, performing the forward-in-time calculations, ie,

$$\begin{aligned} u_i^{k+1} &= F_i^{\{k+1\}} y_i^{k+1}, \\ y_{i+1}^{k+1} &= A^{\{k\}} y_i^{k+1} + B^{\{k\}} u_i^{k+1}, \end{aligned} \tag{72}$$

where $y_0^{k+1} = y_0$. Note that in the above expressions, the superscripts $k+1$ and $\{k+1\}$ indicate the iteration index, not the matrix power.

We remark that we treat the finite-horizon LQ control as an open-loop control problem, that is, at the $(k+1)$st iteration, we use the computed input sequence $\{u_i^{k+1}\}$ to control the system.

Our iterative algorithm for the finite-horizon LQ control of unknown discrete-time linear systems is formally presented as Algorithm 1.

Algorithm 1. Iterative algorithm for finite-horizon LQ control

Initialize the v^*-estimate $v^0 = \text{vec}\left([A^0, B^0]\right) \in \mathbb{R}^{n_y^2+n_y n_u}$;
Initialize the RLS parameter $\Pi^0 \in \mathbb{R}^{(n_y^2+n_y n_u)\times(n_y^2+n_y n_u)}$;
$[A^0, B^0] = \text{vec}^{-1}(v^0)$;
Solve $\left\{F_i^{\{1\}}\right\}_{i=0}^{N-1}$ using (71); Solve $x^1 = \text{vec}\left([u_0^1, \dots, u_{N-1}^1]\right)$ using (72);
Obtain $x^0 = \text{vec}\left([u_0^0, \dots, u_{N-1}^0]\right)$ by $u_i^0 = u_i^1 + e_i$, $i = 0, \dots, N-1$;
$f(x^0, v^*) = \text{Cost}(x^0)$;
for $k = 0 : k_{\max} - 1$ **do**
  $f(x^{k+1}, v^*) = \text{Cost}(x^{k+1})$;
  Compute ϕ^{k+1} using (11); Compute ζ^{k+1} using (12), (13), and (14);
  Update Π^{k+1} using (17); Update v^{k+1} using (18);
  $[A^{k+1}, B^{k+1}] = \text{vec}^{-1}(v^{k+1})$;
  Solve $\left\{F_i^{\{k+2\}}\right\}_{i=0}^{N-1}$ using (71); Solve $x^{k+2} = \text{vec}\left([u_0^{k+2}, \dots, u_{N-1}^{k+2}]\right)$ using (72);
end for

In the algorithm, $e_i \in \mathbb{R}^{n_u}$, $i = 0, \dots, N-1$, are small perturbations.

By Assumption 2, the value of $f(x^k, v^*)$ can be measured for each $k \in \mathbb{Z}_{\geq 0}$. In the case of state measurement, the cost evaluation function Cost($\cdot$) takes the form of Algorithm 2.

Algorithm 2. Cost evaluation for finite-horizon LQ control

Function $\text{Cost}(x^k)$
Input $x^k = \text{vec}\left([u_0^k, \dots, u_{N-1}^k]\right)$
Output $f(x^k, v^*)$
$[u_0^k, \dots, u_{N-1}^k] = \text{vec}^{-1}(x^k)$;
$y_0^k = y_0$;
for $i = 0 : N-1$ **do**
  $y_{i+1}^k = \mathcal{T}(y_i^k, u_i^k, v^*)$;
end for
$f(x^k, v^*) = \sum_{i=0}^{N-1} \left((y_{i+1}^k)^\top Q\, y_{i+1}^k + (u_i^k)^\top R\, u_i^k\right)$.

The $\mathcal{T}(\cdot, \cdot, v^*)$ represents the true system (60).

3.2 | Finite-horizon LQ control via ES

ES is a non–model-based method for real-time optimization.[26] In the work of Frihauf et al,[27] an algorithm based on multiparameter ES for the finite-horizon LQ control of unknown discrete-time linear systems is proposed. The approach is extended to handle measurement noise in the work of Liu et al.[28] To estimate gradients of the cost function, sinusoidal probing signals are added to the nominal control inputs. In this section, we review the algorithm and will later compare the performance of our iterative algorithm to that of ES.

The control input sequence $x = [u_0^\top, u_1^\top, \dots, u_{N-1}^\top]^\top$ is updated according to Algorithm 3.

Algorithm 3. ES algorithm for finite-horizon LQ control

Initialize the nominal control sequence $\hat{x}^0 = \text{vec}\left([\hat{u}_0^0, \dots, \hat{u}_{N-1}^0]\right) \in \mathbb{R}^{n_u N}$;
Initialize the parameter $\xi^0 \in \mathbb{R}$;
for $k = 0 : k_{\max} - 1$ **do**
  $x^k = \hat{x}^k + \alpha S^k$;
  $f(x^k, v^*) = \text{Cost}(x^k)$;
  $\hat{x}^{k+1} = \hat{x}^k - \epsilon K M^{k+1}\left(f(x^k, v^*) - \xi^k\right)$;
  $\xi^{k+1} = (1 - \epsilon h)\,\xi^k + \epsilon h f(x^k, v^*)$;
end for

In the algorithm, $\epsilon > 0$ is a small parameter, K is a positive diagonal matrix, $h > 0$ and $\alpha > 0$ are design parameters, and

$$M^k = \begin{bmatrix} \cos(k\omega_1 - \psi_1) \\ \vdots \\ \cos(k\omega_{n_uN} - \psi_{n_uN}) \end{bmatrix}, \qquad S^k = \begin{bmatrix} \cos(k\omega_1) \\ \vdots \\ \cos(k\omega_{n_uN}) \end{bmatrix}, \tag{73}$$

where $\omega_i = b_i\pi$; b_i is a rational number such that $\omega_i \neq \omega_j$ for all distinct $i, j \in \{1, 2, \dots, n_uN\}$; and $\psi_i = -\omega_i$.

The ES algorithm uses the sinusoidal perturbations M^k and S^k to estimate, and drive to zero, the gradient of the cost function $f(\cdot, v^*)$. Thus, if x^k converges to x^∞, x^∞ satisfies the first-order necessary condition for optimality, ie,

$$\nabla_x f(x^\infty, v^*) = 0. \tag{74}$$

The convergence of x^k to x^∞ is discussed in the works of Frihauf et al[27] and Liu et al.[28] If f is (locally) convex, x^∞ is a (local) minimizer.

4 | INFINITE-HORIZON LQ CONTROL OF UNKNOWN DISCRETE-TIME LINEAR SYSTEMS

In this section, we describe how to exploit our proposed approach to treat the infinite-horizon LQ optimal control problem for unknown discrete-time systems. Similar to the finite-horizon case, our iterative approach is applicable to situations where the cost can be evaluated through simulations, or to batch processes.

4.1 | Iterative approach to infinite-horizon LQ control of unknown systems

A discrete-time infinite-horizon LQ problem involves minimizing a cost function, ie,

$$\min_{u_i,\, i=0,1,\dots} f = \sum_{i=0}^{\infty} \left(y_i^\top Q y_i + u_i^\top R u_i \right), \tag{75}$$

subject to

$$y_{i+1} = A y_i + B u_i, \tag{76}$$

where $Q = Q^\top \succeq 0$ and $R = R^\top \succ 0$.

It is known that, under the standard stabilizability and detectability assumptions,[17] the optimal solution to the above problem is a time-invariant feedback law, ie,

$$u_i = F y_i. \tag{77}$$

Substituting (77) into (75) and (76), we obtain an "equivalent" problem as follows:

$$\min_F f = \sum_{i=0}^{\infty} y_i^\top (Q + F^\top R F) y_i, \tag{78}$$

subject to

$$y_{i+1} = (A + BF) y_i. \tag{79}$$

Note that the "equivalence" here is in a sense that the optimal solution to (75) and (76) takes the form of $u_i = Fy_i$, where F is the global optimal solution to (78) and (79).

The dynamic equation (79) yields

$$y_i = (A + BF)^i y_0. \tag{80}$$

Substituting (80) into (78), we get

$$f = y_0^\top \sum_{i=0}^{\infty} \left(\left((A + BF)^i\right)^\top \left(Q + F^\top R F\right) \left((A + BF)^i\right) \right) y_0. \tag{81}$$

If we list the entries of F to optimize in some order and define a vector, eg,

$$x = \text{vec}(F) = \left[\mathfrak{f}_{11}, \mathfrak{f}_{21}, \ldots, \mathfrak{f}_{n_u n_y}\right]^\top, \tag{82}$$

where $\mathfrak{f}_{mn}$ denotes the (m, n)-entry of matrix F, and list the unknown entries of (A, B) in some order and define an unknown parameter vector, eg,

$$v^* = \left[\text{vec}^\top(A), \text{vec}^\top(B)\right]^\top = \left[a_{11}, a_{21}, \ldots, a_{n_y n_y}, b_{11}, b_{21}, \ldots, b_{n_y n_u}\right]^\top, \tag{83}$$

then problem (78) and (79) is converted into the following form:

$$\min_x f(x, v^*), \tag{84}$$

where f is given by (81), x is given by (82), and v^* is given by (83). Therefore, we can use the approach in Section 2 to update the estimate of v^* while improving the feedback gain $F = \text{vec}^{-1}(x)$.

Suppose the closed-loop system (79) is asymptotically stable. Then, the series

$$T := \sum_{i=0}^{\infty} \left(\left((A + BF)^i\right)^\top \left(Q + F^\top RF\right) \left((A + BF)^i\right) \right) \tag{85}$$

converges, and T satisfies

$$(Q + F^\top RF) + (A + BF)^\top T(A + BF) = T, \tag{86}$$

which is in the form of a discrete-time Lyapunov equation.

The analytic solution to (86) is given by

$$\text{vec}(T) = \Lambda^{-1}\text{vec}(Q + F^\top RF), \tag{87}$$

where

$$\Lambda := I_{n_y^2} - (A + BF)^\top \otimes (A + BF)^\top. \tag{88}$$

The objective function (81) can be written as

$$f = \text{vec}(f) = \text{vec}\left(y_0^\top T y_0\right) = (y_0 \otimes y_0)^\top \text{vec}(T). \tag{89}$$

The first-order partial derivatives have the following form:

$$\begin{aligned}\nabla_{\mathfrak{f}_{mn}} f &= (y_0 \otimes y_0)^\top \left(-\Lambda^{-1} \left(\nabla_{\mathfrak{f}_{mn}} \Lambda\right) \Lambda^{-1}\text{vec}(Q + F^\top RF) + \Lambda^{-1} \nabla_{\mathfrak{f}_{mn}} \left(\text{vec}(Q + F^\top RF)\right)\right) \\ &= (y_0 \otimes y_0)^\top \left(\Lambda^{-1}\Omega_{mn}\Lambda^{-1}\text{vec}(Q + F^\top RF) + \Lambda^{-1}\text{vec}(\Xi_{mn})\right),\end{aligned} \tag{90}$$

where

$$\Omega_{mn} := -\left(\nabla_{\mathfrak{f}_{mn}} \Lambda\right) = (B[1_{mn}]_F)^\top \otimes (A + BF)^\top + (A + BF)^\top \otimes (B[1_{mn}]_F)^\top,$$

$$\Xi_{mn} := \nabla_{\mathfrak{f}_{mn}}(Q + F^\top RF) = [1_{mn}]_F^\top RF + F^\top R[1_{mn}]_F,$$

and $[1_{mn}]_F$ is a matrix of size(F), where the (m, n)-entry is 1 and all other entries are 0.

The second-order partial derivatives have the following forms:

$$\begin{aligned}\nabla_{\mathfrak{f}_{mn},\mathfrak{f}_{m'n'}} f &= (y_0 \otimes y_0)^\top \left(-\Lambda^{-1} \left(\nabla_{\mathfrak{f}_{m'n'}} \Lambda\right) \Lambda^{-1}\Omega_{mn}\Lambda^{-1}\text{vec}(Q + F^\top RF) + \Lambda^{-1} \left(\nabla_{\mathfrak{f}_{m'n'}} \Omega_{mn}\right) \Lambda^{-1}\text{vec}(Q + F^\top RF)\right. \\ &\quad - \Lambda^{-1}\Omega_{mn}\Lambda^{-1} \left(\nabla_{\mathfrak{f}_{m'n'}} \Lambda\right) \Lambda^{-1}\text{vec}(Q + F^\top RF) + \Lambda^{-1}\Omega_{mn}\Lambda^{-1}\nabla_{\mathfrak{f}_{m'n'}} \left(\text{vec}(Q + F^\top RF)\right) \\ &\quad \left. - \Lambda^{-1} \left(\nabla_{\mathfrak{f}_{m'n'}} \Lambda\right) \Lambda^{-1}\text{vec}(\Xi_{mn}) + \Lambda^{-1}\nabla_{\mathfrak{f}_{m'n'}} (\text{vec}(\Xi_{mn}))\right) \\ &= (y_0 \otimes y_0)^\top \left(\Lambda^{-1} \left(\Omega_{m'n'}\Lambda^{-1}\Omega_{mn} + \nabla_{\mathfrak{f}_{m'n'}}\Omega_{mn} + \Omega_{mn}\Lambda^{-1}\Omega_{m'n'}\right) \Lambda^{-1}\text{vec}(Q + F^\top RF)\right. \\ &\quad \left. + \Lambda^{-1}\Omega_{mn}\Lambda^{-1}\text{vec}(\Xi_{m'n'}) + \Lambda^{-1}\Omega_{m'n'}\Lambda^{-1}\text{vec}(\Xi_{mn}) + \Lambda^{-1}\text{vec}\left(\nabla_{\mathfrak{f}_{m'n'}}\Xi_{mn}\right)\right),\end{aligned} \tag{91}$$

where

$$\nabla_{\mathfrak{f}_{m'n'}} \Omega_{mn} = (B[1_{mn}]_F)^\top \otimes (B[1_{m'n'}]_F)^\top + (B[1_{m'n'}]_F)^\top \otimes (B[1_{mn}]_F)^\top,$$
$$\nabla_{\mathfrak{f}_{m'n'}} \Xi_{mn} = [1_{mn}]_F^\top R[1_{m'n'}]_F + [1_{m'n'}]_F^\top R[1_{mn}]_F;$$

$$\nabla_{\mathfrak{f}_{mn}, a_{pq}} f = (y_0 \otimes y_0)^\top \left(-\Lambda^{-1} \left((\nabla_{a_{pq}} \Lambda)\Lambda^{-1} \Omega_{mn} - \nabla_{a_{pq}} \Omega_{mn} + \Omega_{mn} \Lambda^{-1} (\nabla_{a_{pq}} \Lambda) \right) \right.$$
$$\left. \Lambda^{-1} \mathrm{vec}(Q + F^\top RF) - \Lambda^{-1} (\nabla_{a_{pq}} \Lambda) \Lambda^{-1} \mathrm{vec}(\Xi_{mn}) \right), \tag{92}$$

where

$$\nabla_{a_{pq}} \Lambda = -\left([1_{pq}]_A\right)^\top \otimes (A + BF)^\top - (A + BF)^\top \otimes \left([1_{pq}]_A\right)^\top,$$
$$\nabla_{a_{pq}} \Omega_{mn} = (B[1_{mn}]_F)^\top \otimes \left([1_{pq}]_A\right)^\top + \left([1_{pq}]_A\right)^\top \otimes (B[1_{mn}]_F)^\top;$$

and

$$\nabla_{\mathfrak{f}_{mn}, b_{p'q'}} f = (y_0 \otimes y_0)^\top \left(-\Lambda^{-1} \left((\nabla_{b_{p'q'}} \Lambda)\Lambda^{-1} \Omega_{mn} - \nabla_{b_{p'q'}} \Omega_{mn} + \Omega_{mn} \Lambda^{-1} (\nabla_{b_{p'q'}} \Lambda) \right) \right.$$
$$\left. \Lambda^{-1} \mathrm{vec}(Q + F^\top RF) - \Lambda^{-1} (\nabla_{b_{p'q'}} \Lambda) \Lambda^{-1} \mathrm{vec}(\Xi_{mn}) \right), \tag{93}$$

where

$$\nabla_{b_{p'q'}} \Lambda = -\left([1_{p'q'}]_B F\right)^\top \otimes (A + BF)^\top - (A + BF)^\top \otimes \left([1_{p'q'}]_B F\right)^\top,$$
$$\nabla_{b_{p'q'}} \Omega_{mn} = (B[1_{mn}]_F)^\top \otimes \left([1_{p'q'}]_B F\right)^\top + \left([1_{p'q'}]_B F\right)^\top \otimes (B[1_{mn}]_F)^\top$$
$$+ \left([1_{p'q'}]_B [1_{mn}]_F\right)^\top \otimes (A + BF)^\top + (A + BF)^\top \otimes \left([1_{p'q'}]_B [1_{mn}]_F\right)^\top.$$

Then, we can formulate the matrices in (9) as

$$\nabla_{x,x} f\left(x^{k+1}, v^k\right) = \left[\nabla_{\mathfrak{f}_{mn}, \mathfrak{f}_{m'n'}} f \right] = \begin{bmatrix} \nabla_{\mathfrak{f}_{11}, \mathfrak{f}_{11}} f & \nabla_{\mathfrak{f}_{11}, \mathfrak{f}_{21}} f & \cdots & \nabla_{\mathfrak{f}_{11}, \mathfrak{f}_{n_u n_y}} f \\ \nabla_{\mathfrak{f}_{21}, \mathfrak{f}_{11}} f & \nabla_{\mathfrak{f}_{21}, \mathfrak{f}_{21}} f & \cdots & \nabla_{\mathfrak{f}_{21}, \mathfrak{f}_{n_u n_y}} f \\ \vdots & \vdots & \ddots & \vdots \\ \nabla_{\mathfrak{f}_{n_u n_y}, \mathfrak{f}_{11}} f & \nabla_{\mathfrak{f}_{n_u n_y}, \mathfrak{f}_{21}} f & \cdots & \nabla_{\mathfrak{f}_{n_u n_y}, \mathfrak{f}_{n_u n_y}} f \end{bmatrix}, \tag{94}$$

$$\nabla_{x,v} f\left(x^{k+1}, v^k\right) = \left[\nabla_{\mathfrak{f}_{mn}, a_{pq}} f, \nabla_{\mathfrak{f}_{mn}, b_{p'q'}} f \right] = \begin{bmatrix} \nabla_{\mathfrak{f}_{11}, a_{11}} f & \nabla_{\mathfrak{f}_{11}, a_{21}} f & \cdots & \nabla_{\mathfrak{f}_{11}, a_{n_y n_y}} f & \nabla_{\mathfrak{f}_{11}, b_{11}} f & \cdots & \nabla_{\mathfrak{f}_{11}, b_{n_y n_u}} f \\ \vdots & \vdots & & \ddots & \ddots & \ddots & \vdots \\ \nabla_{\mathfrak{f}_{n_u n_y}, a_{11}} f & \nabla_{\mathfrak{f}_{n_u n_y}, a_{21}} f & \cdots & \nabla_{\mathfrak{f}_{n_u n_y}, a_{n_y n_y}} f & \nabla_{\mathfrak{f}_{n_u n_y}, b_{11}} f & \cdots & \nabla_{\mathfrak{f}_{n_u n_y}, b_{n_y n_u}} f \end{bmatrix}.$$

The minimizer $x^{k+1} = \mathrm{vec}(F^{\{k+1\}})$ at the $(k + 1)$st iteration can be analytically computed by solving the discrete-time algebraic Riccati equation, ie,

$$T^{\{k+1\}} = Q + (A^{\{k\}})^\top T^{\{k+1\}} A^{\{k\}} - (A^{\{k\}})^\top T^{\{k+1\}} B^{\{k\}} \left(R + (B^{\{k\}})^\top T^{\{k+1\}} B^{\{k\}}\right)^{-1} (B^{\{k\}})^\top T^{\{k+1\}} A^{\{k\}},$$
$$F^{\{k+1\}} = -\left(R + (B^{\{k\}})^\top T^{\{k+1\}} B^{\{k\}}\right)^{-1} (B^{\{k\}})^\top T^{\{k+1\}} A^{\{k\}}. \tag{95}$$

Note that the superscript $\{k + 1\}$ indicates the iteration index, not the matrix power.

We remark that in the infinite-horizon LQ control case, we apply our methodology to search for an optimal feedback matrix F, whereas in the finite-horizon LQ control case considered in Section 3, we apply our methodology to search for an optimal open-loop control sequence.

Our iterative algorithm for the infinite-horizon LQ control of unknown discrete-time linear systems is formally presented as Algorithm 4.

Algorithm 4. Iterative algorithm for infinite-horizon LQ control

Initialize the v^*-estimate $v^0 = \mathrm{vec}\big([A^0, B^0]\big) \in \mathbb{R}^{n_y^2+n_yn_u}$;
Initialize the RLS parameter $\Pi^0 \in \mathbb{R}^{(n_y^2+n_yn_u)\times(n_y^2+n_yn_u)}$;
$[A^0, B^0] = \mathrm{vec}^{-1}(v^0)$;
Solve $x^1 = \mathrm{vec}(F^{\{1\}})$ using (95), $x^0 = x^1 + e^{\{0\}}$;
$f(x^0, v^*) = \overline{\mathrm{Cost}}(x^0)$;
for $k = 0 : k_{\max} - 1$ **do**
  $f(x^{k+1}, v^*) = \overline{\mathrm{Cost}}(x^{k+1})$;
  Compute ϕ^{k+1} using (11); Compute ζ^{k+1} using (12), (13), and (14);
  Update Π^{k+1} using (17); Update v^{k+1} using (18);
  $[A^{k+1}, B^{k+1}] = \mathrm{vec}^{-1}(v^{k+1})$;
  Solve $x^{k+2} = \mathrm{vec}(F^{\{k+2\}})$ using (95);
  if *Perturbation* = *True* **then**
    $x^{k+1} \leftarrow x^{k+1} + e^{\{k+1\}}$;
    $f(x^{k+1}, v^*) = \overline{\mathrm{Cost}}(x^{k+1})$;
  end if
end for

In the algorithm, $e^{\{k\}} \in \mathbb{R}^{n_un_y}$, with $e^{\{k\}} \to 0$ as $k \to \infty$, are small perturbations. When *Perturbation* = *True*, perturbations are added, playing the role of probing signals, to generate persistent excitations to the RLS algorithm. It will be shown in the example in Section 5 that adding perturbations in the algorithm can effectively help the algorithm escape local minima, at the cost of slower convergence.

By Assumption 2, the value of $f(x^k, v^*)$ can be measured for each $k \in \mathbb{Z}_{\geq 0}$. In the case of state measurement, the cost evaluation function $\overline{\mathrm{Cost}}(\cdot)$ takes the form of Algorithm 5, with $i_{\max} \in \mathbb{Z}_{\geq 0}$ being sufficiently large.

Algorithm 5. Cost evaluation for infinite-horizon LQ control

Function $\overline{\mathrm{Cost}}(x^k)$
Input $x^k = \mathrm{vec}(F^{\{k\}})$
Output $f(x^k, v^*)$
$F^{\{k\}} = \mathrm{vec}^{-1}(x^k)$;
$y_0^k = y_0$;
for $i = 0 : i_{\max} - 1$ **do**
  $y_{i+1}^k = \mathcal{T}(y_i^k, F^{\{k\}}y_i^k, v^*)$;
end for
$f(x^k, v^*) = \sum_{i=0}^{i_{\max}} (y_i^k)^\top \big(Q + (F^{\{k\}})^\top R F^{\{k\}}\big) y_i^k$.

4.2 | Infinite-horizon LQ control via RL

As discussed in Section 1, approaches based on RL have also been proposed for the infinite-horizon LQ control problem of unknown dynamic systems. In particular, an RL algorithm that exploits Q-learning can be used, which is now reviewed. We will then compare the performance of our iterative algorithm to that of RL.

The Q-learning algorithm presented here is based on the works of Lewis and Vrabie[11] and Bradtke.[18]

The Bellman value function[13] of the infinite-horizon LQ control problem (75) and (76) is known to be $\mathcal{V}(y) = y^\top T y$, where T is the unique stabilizing solution to the discrete-time algebraic Riccati equation, ie,

$$T = Q + A^\top TA - A^\top TB(R + B^\top TB)^{-1}B^\top TA. \tag{96}$$

The Q-value of the state-control pair (y, u) is defined as

$$\mathcal{Q}(y,u) = y^\top Q y + u^\top R u + \mathcal{V}(Ay+Bu) = \begin{bmatrix} y \\ u \end{bmatrix}^\top \begin{bmatrix} H_{yy} & H_{yu} \\ H_{yu}^\top & H_{uu} \end{bmatrix} \begin{bmatrix} y \\ u \end{bmatrix}, \tag{97}$$

where

$$H_{yy} = Q + A^\top TA, \quad H_{yu} = A^\top TB, \quad H_{uu} = R + B^\top TB. \tag{98}$$

On the basis of the Bellman optimality condition,[13] the optimal control at state y is

$$u^*(y) = \operatorname*{argmin}_u \mathcal{Q}(y,u). \tag{99}$$

Since $\mathcal{Q}(y,u)$ is convex and differentiable in u, $u^*(y)$ can be solved using the first-order necessary condition

$$\nabla_u \mathcal{Q}(y,u) = 2y^\top H_{yu} + 2u^*(y)^\top H_{uu} = 0, \quad u^*(y) = -(H_{uu})^{-1} H_{yu}^\top y. \tag{100}$$

That is, the optimal feedback matrix, ie, the optimal solution to (78) and (79), is

$$F = -(H_{uu})^{-1} H_{yu}^\top. \tag{101}$$

Equation (97) can be written as

$$\mathcal{Q}(y,u) = v^\top \overline{H}, \tag{102}$$

where

$$v = \begin{bmatrix} y \\ u \end{bmatrix} \otimes \begin{bmatrix} y \\ u \end{bmatrix}, \quad \overline{H} = \operatorname{vec}\left(\begin{bmatrix} H_{yy} & H_{yu} \\ H_{yu}^\top & H_{uu} \end{bmatrix} \right). \tag{103}$$

Suppose k samples of $(y^i, u^i, \mathcal{Q}(y^i, u^i))$, $i = 1, \dots, k$, are collected, the least squares estimate of $\overline{H}$ is

$$\overline{H}^k = \left((\Upsilon^k)^\top \Upsilon^k \right)^{-1} (\Upsilon^k)^\top \mathbf{Q}^k, \tag{104}$$

where

$$\Upsilon^k = \left[v^1, \ \dots, \ v^k \right]^\top, \quad \mathbf{Q}^k = \left[\mathcal{Q}(y^1, u^1), \ \dots, \ \mathcal{Q}(y^k, u^k) \right]^\top. \tag{105}$$

The true Q-values are unknown but can be estimated, where the estimate is updated after each update of the $\overline{H}$ estimation, based on

$$\begin{aligned} \mathcal{Q}^{k+1}(y^k, u^k) = (1-\lambda^{k+1})\mathcal{Q}^k(y^k, u^k) + \lambda^{k+1} \Bigg(& (y^k)^\top Q y^k + (u^k)^\top R u^k \\ & + \begin{bmatrix} y^{k+1} \\ -(H_{uu}^k)^{-1} (H_{yu}^k)^\top y^{k+1} \end{bmatrix}^\top \begin{bmatrix} H_{yy}^k & H_{yu}^k \\ (H_{yu}^k)^\top & H_{uu}^k \end{bmatrix} \begin{bmatrix} y^{k+1} \\ -(H_{uu}^k)^{-1} (H_{yu}^k)^\top y^{k+1} \end{bmatrix} \Bigg), \end{aligned} \tag{106}$$

where $y^{k+1} = Ay^k + Bu^k$ is the next state, obtained via a simulation/experiment on the true system (76) with (y^k, u^k) as the current state-control pair when (A, B) are unknown, $\lambda^{k+1} \in [0,1]$ is a learning rate, and H_{yy}^k, H_{yu}^k, and H_{uu}^k are reconstructed using $\overline{H}^k$.

The convergence of Q^{k+1} to Q and the convergence of $\overline{H}^k$ to $\overline{H}$ are discussed in the work of Lewis and Vrabie.[11] The Q-learning algorithm used in this paper is formally presented as Algorithm 6.

Algorithm 6. Q-learning algorithm for infinite-horizon LQ control

Initialize the Q-function parameter $\overline{H}^0 \in \mathbb{R}^{(n_y+n_u)^2}$;
Initialize the RLS parameter $\Theta^0 \in \mathbb{R}^{(n_y+n_u)^2\times(n_y+n_u)^2}$;
for $k = 0 : k_{\max} - 1$ **do**
$\begin{bmatrix} H^k_{yy} & H^k_{yu} \\ (H^k_{yu})^\top & H^k_{uu} \end{bmatrix} = \left(\mathrm{vec}^{-1}(\overline{H}^k) + \mathrm{vec}^{-1}(\overline{H}^k)^\top\right)/2;$
$y^{k+1}_0 = y_0;$
$\hat{u}^{k+1}_0 = -(H^k_{uu})^{-1}(H^k_{yu})^\top\, y^{k+1}_0;$
$Y^{k+1} = \text{null}, \mathbf{Q}^{k+1} = \text{null};$
for $i = 0 : i_{\max} - 1$ **do**
$u^{k+1}_i = \hat{u}^{k+1}_i + e^{\{k+1\}}_i;$
$y^{k+1}_{i+1} = \mathcal{T}(y^{k+1}_i, u^{k+1}_i, v^*);$
$\hat{u}^{k+1}_{i+1} = -(H^k_{uu})^{-1}(H^k_{yu})^\top\, y^{k+1}_{i+1};$
$v^{k+1}_i = \begin{bmatrix} y^{k+1}_i \\ u^{k+1}_i \end{bmatrix} \otimes \begin{bmatrix} y^{k+1}_i \\ u^{k+1}_i \end{bmatrix}, \hat{v}^{k+1}_{i+1} = \begin{bmatrix} y^{k+1}_{i+1} \\ \hat{u}^{k+1}_{i+1} \end{bmatrix} \otimes \begin{bmatrix} y^{k+1}_{i+1} \\ \hat{u}^{k+1}_{i+1} \end{bmatrix};$
$Q^{k+1}(y^{k+1}_i, u^{k+1}_i) = (y^{k+1}_i)^\top Q\, y^{k+1}_i + (u^{k+1}_i)^\top R\, u^{k+1}_i + (\hat{v}^{k+1}_{i+1})^\top \overline{H}^k;$
$(Y^{k+1})^\top = \left[(Y^{k+1})^\top \; v^{k+1}_i\right], (\mathbf{Q}^{k+1})^\top = \left[(\mathbf{Q}^{k+1})^\top \; Q^{k+1}(y^{k+1}_i, u^{k+1}_i)\right];$
end for
$\Theta^{k+1} = \Theta^k - \Theta^k (Y^{k+1})^\top \left(I_{i_{\max}} + Y^{k+1}\Theta^k (Y^{k+1})^\top\right)^{-1} Y^{k+1}\Theta^k;$
$\overline{H}^{k+1} = \overline{H}^k + \lambda^{k+1}\Theta^{k+1}(Y^{k+1})^\top\left(\mathbf{Q}^{k+1} - Y^{k+1}\overline{H}^k\right);$
end for

Algorithm 6 corresponds to updating the estimate of $\overline{H}$ every $i_{\max}$ samples using a mini-batch RLS algorithm. The $e^{\{k\}}_i \in \mathbb{R}^{n_u}$, with $e^{\{k\}}_i \to 0$ as $k \to \infty$, are probing signals that have to be added to let the Q-function estimate converge.

4.3 | Infinite-horizon LQ control via ES

Multiparameter ES[26] can also be used for the infinite-horizon LQ control of unknown discrete-time linear systems, specifically, to solve optimization problem (78) and (79).

The feedback gain parameters $x = \mathrm{vec}(F)$ are updated according to Algorithm 7.

Algorithm 7. ES algorithm for infinite-horizon LQ control

Initialize the nominal feedback gain parameters $\hat{x}^0 = \mathrm{vec}(\hat{F}^{\{0\}}) \in \mathbb{R}^{n_u n_y}$;
Initialize the parameter $\xi^0 \in \mathbb{R}$;
for $k = 0 : k_{\max} - 1$ **do**
$x^k = \hat{x}^k + \alpha S^k;$
$f(x^k, v^*) = \overline{\mathrm{Cost}}(x^k);$
$\hat{x}^{k+1} = \hat{x}^k - \epsilon K M^{k+1}\left(f(x^k, v^*) - \xi^k\right);$
$\xi^{k+1} = (1 - \epsilon h)\, \xi^k + \epsilon h f(x^k, v^*);$
end for

In the algorithm, the parameters ϵ, h, and α and the matrices K, M^k, and S^k are defined similarly to those in the ES algorithm for finite-horizon LQ control in Section 3.2.

5 | PITCH ATTITUDE FLIGHT CONTROL

In this section, we use a dynamic model representing the short-period pitch attitude dynamics of an AFTI/F-16 aircraft to illustrate our approach discussed in Sections 3 and 4 (see Figure 3).

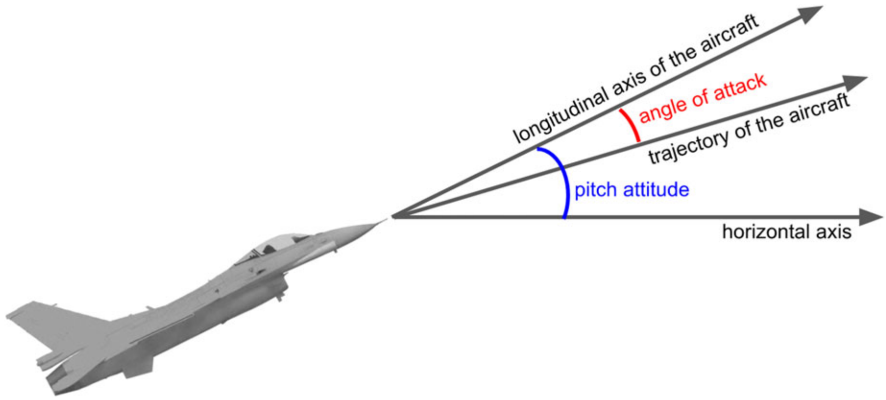

FIGURE 3 Longitudinal kinematics of an aircraft [Colour figure can be viewed at wileyonlinelibrary.com]

Conventional approaches to developing flight control algorithms rely on first identifying aircraft linear time-invariant (LTI) models corresponding to different aircraft internal states and external operating conditions† (off-line), designing feedback control laws at these states (off-line), and then implementing a control corresponding to the current state based on gain scheduling/gain interpolation (online).[39-41] Our iterative approach, in contrast, optimizes the control corresponding to the current state without relying on pre-identified LTI models. This could be an advantage when aircraft parameters change, eg, due to aging, when storing large gain tables is impractical, which could be the case in unmanned aerial vehicle–type aircraft, or when the aircraft operates at unanticipated conditions, eg, aircraft icing or in loss-of-control situations, where we may not have enough time to identify a new LTI model first, but need to control the aircraft immediately.

In the implementation, the cost $f(x^{k+1}, v^*)$ at each iteration $k+1$ can be evaluated via a simulation using an onboard-stored high-fidelity flight dynamics model, such as the F-16 model,[42] the National Aeronautics and Space Administration generic models,[43-45] and the flexible aircraft model,[46] subject to the current state as the initial condition. In this paper, we do not use a high-fidelity model for cost evaluation but assume an underlying LTI model that represents the true aircraft pitch attitude dynamics of the current operating condition, which is unknown to the algorithms. The continuous-time model is taken from the work of Sobel and Shapiro,[39] corresponding to an altitude of 3000 ft and a Mach number of 0.6. We discretize the continuous-time model with sampling time $\Delta t = 0.1$ s and use the MATLAB *c2d* function[47] to obtain the discrete-time model, ie,

$$y_{i+1} = A y_i + B u_i, \tag{107}$$

where

$$y = \begin{bmatrix} \theta \\ q \\ \alpha \\ \delta_e \\ \delta_f \end{bmatrix} \begin{array}{l} \text{– pitch attitude (deg),} \\ \text{– pitch rate (deg/s),} \\ \text{– angle of attack (deg),} \\ \text{– elevator deflection (deg),} \\ \text{– flaperon deflection (deg),} \end{array} \qquad u = \begin{bmatrix} \delta_{ec} \\ \delta_{fc} \end{bmatrix} \begin{array}{l} \text{– elevator deflection command (deg),} \\ \text{– flaperon deflection command (deg)} \end{array} \tag{108}$$

and

$$A = \begin{bmatrix} 1.0000 & 0.1025 & 0.2080 & -0.0502 & -0.0057 \\ 0 & 1.1175 & 4.1534 & -0.8000 & -0.1010 \\ 0 & 0.0955 & 1.0722 & -0.0541 & -0.0153 \\ 0 & 0 & 0 & 0.1353 & 0 \\ 0 & 0 & 0 & 0 & 0.1353 \end{bmatrix}, \quad B = \begin{bmatrix} -0.0377 & -0.0040 \\ -1.0042 & -0.1131 \\ -0.0453 & -0.0175 \\ 0.8647 & 0 \\ 0 & 0.8647 \end{bmatrix}. \tag{109}$$

†These include aircraft weight, trim condition, etc, and are uniformly referred to as the aircraft "state."

5.1 | Finite-horizon LQ

Assume that the true model (109) is unknown. Instead, we have initial estimates of the matrices (A, B) that have some errors. At first, we consider the initial estimate to be (*Case 1*)

$$A^0 = \begin{bmatrix} 1.0000 & 0.0992 & 0.1918 & -0.0512 & -0.0055 \\ 0 & 1.1090 & 3.9431 & -0.8356 & -0.1025 \\ 0 & 0.0978 & 1.0704 & -0.0575 & -0.0168 \\ 0 & 0 & 0 & 0.1319 & 0 \\ 0 & 0 & 0 & 0 & 0.1523 \end{bmatrix}, \quad B^0 = \begin{bmatrix} -0.0357 & -0.0038 \\ -0.9759 & -0.1112 \\ -0.0441 & -0.0188 \\ 0.7927 & 0 \\ 0 & 0.8852 \end{bmatrix}. \tag{110}$$

We create this initial estimate by first adding $[-10\%, 10\%]$ randomly generated multiplicative error to each entry of the true continuous-time model and then using *c2d* to obtain the corresponding discrete-time model.

Secondly, we consider the initial estimate to be (*Case 2*)

$$A^0 = \begin{bmatrix} 1.0000 & 0.0901 & 0.2195 & -0.0559 & -0.0055 \\ 0 & 1.1271 & 5.0308 & -1.0556 & -0.1172 \\ 0 & 0.0836 & 1.1022 & -0.0588 & -0.0168 \\ 0 & 0 & 0 & 0.1954 & 0 \\ 0 & 0 & 0 & 0 & 0.1721 \end{bmatrix}, \quad B^0 = \begin{bmatrix} -0.0395 & -0.0035 \\ -1.2248 & -0.1142 \\ -0.0454 & -0.0172 \\ 0.9464 & 0 \\ 0 & 0.8565 \end{bmatrix}. \tag{111}$$

We create this estimate by first adding $[-20\%, 20\%]$ randomly generated multiplicative error to each entry of the true continuous-time model and then using *c2d* to obtain the corresponding discrete-time model. We test the robustness of our proposed approach to this inaccurate initial estimation.

We consider the initial condition $y_0 = [2, 0, 1, 0, 0]^\top$ and the objective function (59) with

$$Q = \mathrm{diag}(1, 1, 1, 1, 1), \qquad R = \mathrm{diag}(1, 1), \qquad N = 10. \tag{112}$$

We implement our Algorithm 1 for both *Case 1* and *Case 2*, ie, initializing $v^0 = \mathrm{vec}([A^0, B^0])$ using, respectively, (110) and (111). We initialize $\Pi^0 = 10^2 I_{n_y^2 + n_y n_u}$ and set $\lambda^{k+1} = 1, k \in \mathbb{Z}_{\geq 0}$, in both cases. We also implement the ES algorithm, Algorithm 3, for both cases to compare the performance. To have a relatively fair comparison, in Algorithm 3, we initialize $\hat{x}^0 = \mathrm{vec}([\hat{u}_0^0, \dots, \hat{u}_{N-1}^0])$ as the solution to (71) and (72), where (A^0, B^0) are using (110) in *Case 1* and (111) in *Case 2*, and initialize $\xi^0 = 0$ in both cases. Moreover, we set $\epsilon = 10^{-5}, h = 10^{-1}, \alpha = 10^{-3}, K = I_{n_u N}$, and $b_i = i/(n_u N + 1)$, $i = 1, \dots, n_u N$. These parameters have been tuned to generate the best results that we can get.

In Figures 4A and 5A, we plot the (θ, q) trajectories. The black dashed curve represents the trajectory if one applies the LQ control sequence obtained based on the model (110) or (111) to the true system (109); the red dash-dotted curve represents the trajectory when using the true system (109) to compute the LQ control (referred to as "LQ-true"). We can observe that there is a significant mismatch between the black curve and the red curve—the initial control sequence computed using (110) or (111) fails to satisfactorily control the system. The blue solid curve represents the trajectory obtained

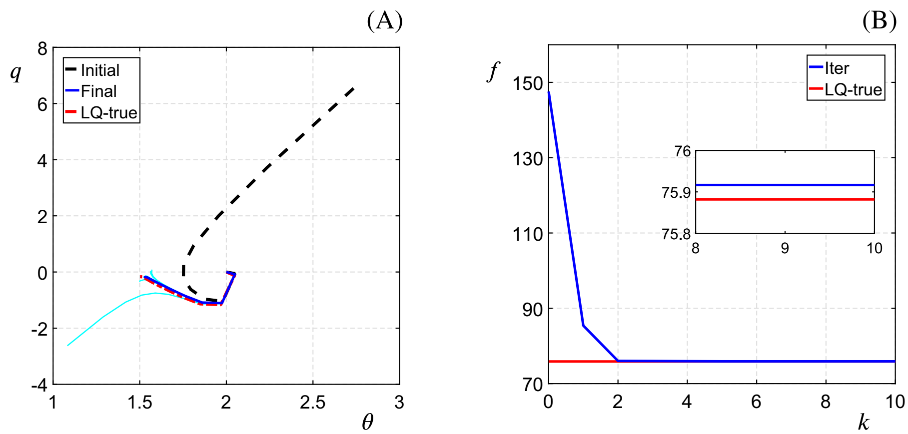

FIGURE 4 Finite-horizon pitch attitude control *Case 1*. A, The initial (black dashed), final (blue solid), and intermediate (cyan) trajectories on the (θ, q)-plane of Algorithm 1 iterations versus the linear quadratic (LQ)-true trajectory (red dash-dotted); B, The cost evolution over Algorithm 1 iterations $(k, f(x^{k+1}, v^*))$ (blue) versus the LQ-true cost (red) [Colour figure can be viewed at wileyonlinelibrary.com]

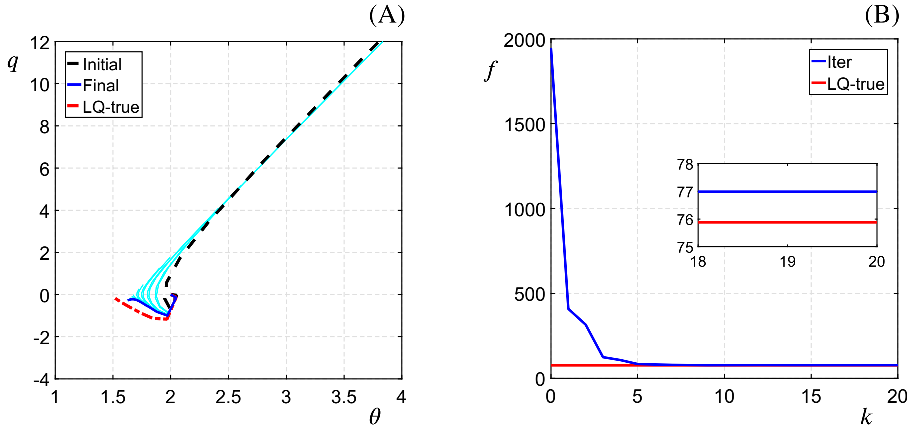

FIGURE 5 Finite-horizon pitch attitude control *Case 2*. A, The initial (black dashed), final (blue solid), and intermediate (cyan) trajectories on the (θ, q)-plane of Algorithm 1 iterations versus the linear quadratic (LQ)-true trajectory (red dash-dotted); B, The cost evolution over Algorithm 1 iterations $(k, f(x^{k+1}, v^*))$ (blue) versus the LQ-true cost (red) [Colour figure can be viewed at wileyonlinelibrary.com]

using our proposed approach after convergence. It matches the red curve well. The cyan curves show the trajectory evolution over the iterations. Figures 4B and 5B show the cost evolution over the iterations. We can observe that (i) the initial control sequence obtained based on the model (110) or (111) leads to a large cost, (ii) the final control sequence obtained using our proposed approach has a cost very close to the optimal value, and (iii) the cost decreases and converges very fast—it gets close to the optimal value after only two iterations in *Case 1* and after only five iterations in *Case 2*.

On the other hand, in Figure 6A,B, we plot the cost evolution over the iterations using the ES algorithm. It can be observed that although the costs converge to the optimal value in both *Case 1* and *Case 2*, it takes thousands of iterations to achieve this, which, in real implementation, corresponds to thousands of high-fidelity simulations or experiments. We note that our implementation of ES is comparable to the implementation in the works of Frihauf et al[27] and Liu et al,[28] which also takes thousands of iterations to converge in a lower-dimensional system.

Note that our Algorithm 1 is robust to step size selection, as shown in Figure 7A, and, importantly, does not need to use probing signals, which are necessary in the ES algorithm. To show the advantage of not using probing signals, we analyze the sensitivity of Algorithm 3 to probing signals. In particular, we plot the cost evolution over the iterations with different ϵ selections, which reflects the probing signal magnitude, while keeping the other parameter values unchanged, in Figure 7B. It can be observed that when the magnitude of probing signals is too small, $\epsilon = 0.2 \times 10^{-5}$, the convergence of the cost significantly slows down because of insufficient excitation, and when the magnitude of probing signals is too large, $\epsilon = 5 \times 10^{-5}$, although the cost decreases faster at the beginning, it increases instead of decreasing over the later iterations, which implies a failure in convergence. Furthermore, when the magnitude of probing signals is larger, the

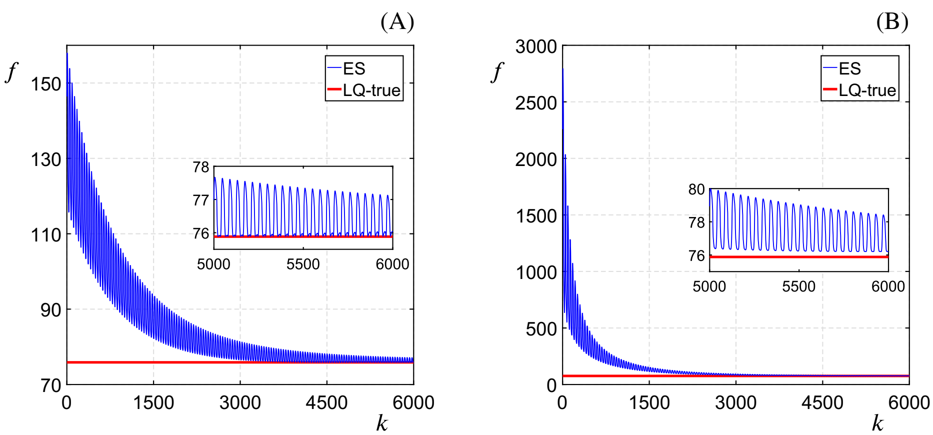

FIGURE 6 Finite-horizon pitch attitude control *Case 1* and *Case 2* using Algorithm 3. A, The cost evolution over Algorithm 3 iterations $(k, f(x^{k+1}, v^*))$ of *Case 1* (blue) versus the linear quadratic (LQ)-true cost (red); B, The cost evolution over Algorithm 3 iterations $(k, f(x^{k+1}, v^*))$ of *Case 2* (blue) versus the LQ-true cost (red). ES, extremum seeking [Colour figure can be viewed at wileyonlinelibrary.com]

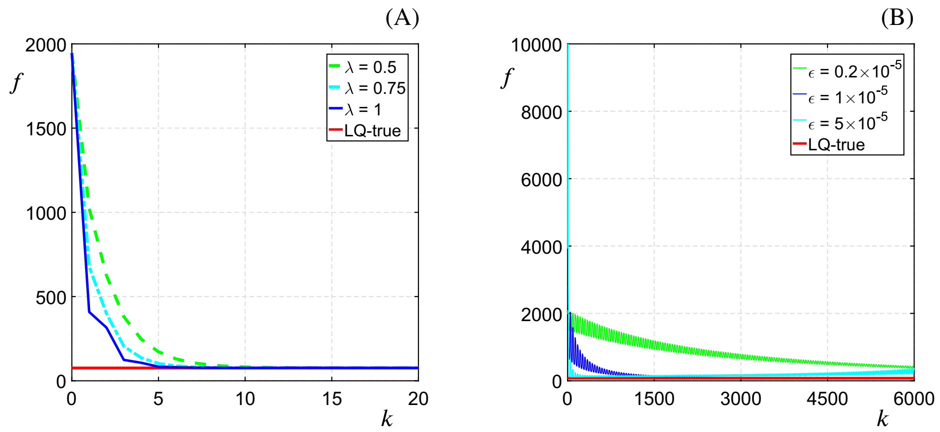

FIGURE 7 Sensitivity to parameters. A, The cost evolutions over Algorithm 1 iterations $(k, f(x^{k+1}, v^*))$ of *Case 2* corresponding to different step size selections versus the linear quadratic (LQ)-true cost (red); B, The cost evolutions over Algorithm 3 iterations $(k, f(x^{k+1}, v^*))$ of *Case 2* corresponding to different ϵ selections versus the LQ-true cost (red) [Colour figure can be viewed at wileyonlinelibrary.com]

overshoot of the cost becomes larger (from 3000 for $\varepsilon = 1 \times 10^{-5}$ to 10 000 for $\varepsilon = 5 \times 10^{-5}$ at the beginning of the iterations). This significant overshoot may imply a danger of destabilization or a damage to the system, when the iterations are performed through hardware experiments. As a result, a careful design of the probing signals may be needed, which may not be straightforward due to the lack of physical interpretations of probing signal parameters. On the other hand, our Algorithm 1 does not have this problem since probing signals are not necessary in our approach. Without using any probing signals, the cost can converge to a satisfactory value, as shown in both *Case 1* and *Case 2*.

On the basis of the above results, our iterative approach is superior to the ES approach, in terms of speed of convergence and implementation complexity, to the reported finite-horizon LQ control problem of the F-16 aircraft pitch attitude dynamics. We note that one of the major advantages of the ES approach is that it is a non–model-based method and does not even need to know the order of the system, which our iterative approach needs to know.

5.2 | Infinite-horizon LQ

We consider the same model (109) and the same weights (112), but now, the cost function is defined over an infinite prediction horizon. Instead of searching for an optimal input sequence $\{u_0, u_1, \ldots, u_{N-1}\}$, we search for an optimal feedback matrix F and feedback control $u_i = Fy_i$.

Because feedback control usually has better robustness against model uncertainties compared to open-loop control, we may be able to handle even more inaccurate initial estimates: we consider the initial estimate of (A, B) to be (*Case 3*)

$$A^0 = \begin{bmatrix} 1.0000 & 0.1131 & 0.2447 & -0.0670 & -0.0067 \\ 0 & 1.1987 & 4.6131 & -1.0086 & -0.1112 \\ 0 & 0.1217 & 1.1459 & -0.0804 & -0.0160 \\ 0 & 0 & 0 & 0.1251 & 0 \\ 0 & 0 & 0 & 0 & 0.1214 \end{bmatrix}, \quad B^0 = \begin{bmatrix} -0.0626 & -0.0060 \\ -1.5574 & -0.1549 \\ -0.0817 & -0.0214 \\ 1.0496 & 0 \\ 0 & 1.0294 \end{bmatrix}. \tag{113}$$

We create this estimate by first adding $[-25\%, 25\%]$ randomly generated multiplicative error to each entry of the true continuous-time model and then using *c2d* to obtain the corresponding discrete-time model. We check that the gain matrix F^1 computed based on the pair (A^0, B^0) can stabilize the true open-loop system (109) (although it is not optimal), which is a necessary condition for operating our proposed approach.

We implement our Algorithm 4 for *Case 3* by initializing $v^0 = \text{vec}([A^0, B^0])$ using (113) and $\Pi^0 = 10^2 I_{n_y^2 + n_y n_u}$ and setting $\lambda^{k+1} = 1$, $k \in \mathbb{Z}_{\geq 0}$, and $i_{\max} = 100$ in the cost evaluation function, Algorithm 5. We also implement the RL algorithm, Algorithm 6, and the ES algorithm, Algorithm 7, to compare the performance. To have a relatively fair comparison, in Algorithm 6, we initialize $\overline{H}^0 = \text{vec}\left(\begin{bmatrix} H_{yy}^0 & H_{yu}^0 \\ (H_{yu}^0)^\top & H_{uu}^0 \end{bmatrix}\right)$ based on (96) and (98), where (A^0, B^0) are using (113), initialize $\Theta^0 = 10^2 I_{(n_y+n_u)^2}$, set $\lambda^{k+1} = 1$, $k \in \mathbb{Z}_{\geq 0}$, and set $i_{\max} = 100$ in the inner for-loop; in Algorithm 7, we initialize $\hat{x}^0 = \text{vec}(\hat{F}^{\{0\}})$ as the solution to (95), where (A^0, B^0) are using (113), and initialize $\xi^0 = 0$. Furthermore, in

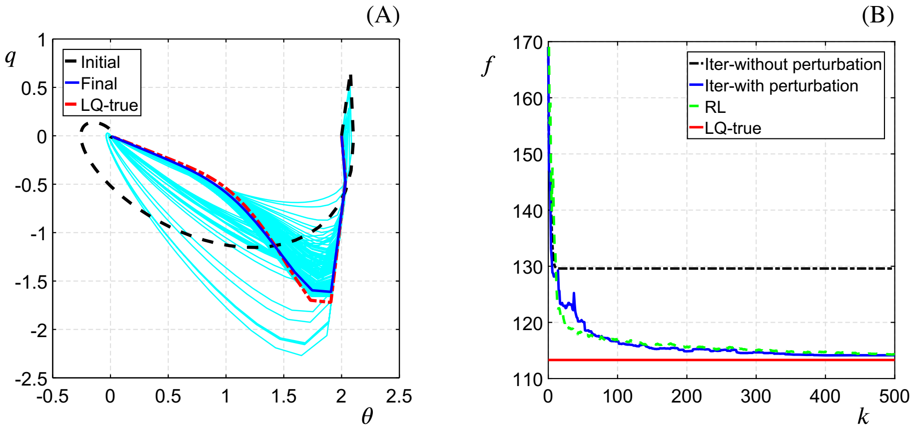

FIGURE 8 Infinite-horizon pitch attitude control *Case 3*. A, The initial (black dashed), final (blue solid), and intermediate (cyan) trajectories on the (θ, q)-plane of Algorithm 4 iterations with perturbations versus the linear quadratic (LQ)-true trajectory (red dash-dotted); B, The cost evolutions over Algorithm 4 iterations without perturbations (black dash-dotted), over Algorithm 4 iterations with perturbations (blue solid), and over Algorithm 6 iterations (green dashed) versus the LQ-true cost (red). RL, reinforcement learning [Colour figure can be viewed at wileyonlinelibrary.com]

Algorithm 4, we set $e^{\{k\}} = 0.99^k \hat{e}^{\{k\}}$, where $\hat{e}^k \sim \mathcal{U}[-10^{-2}, 10^{-2}]^{n_u n_y}$ ‡ ; in Algorithm 6, we set $e_i^{\{k\}} = 0.99^k \tilde{e}_i^{\{k\}}$, where $\tilde{e}_i^{\{k\}} \sim \mathcal{U}[-5 \times 10^{-2}, 5 \times 10^{-2}]^{n_u}$; and in Algorithm 7, we set $\varepsilon = 10^{-4}$, $h = 10^{-1}$, $\alpha = 10^{-4}$, $K = I_{n_u n_y}$, and $b_i = i/(n_u n_y + 1)$, $i = 1, \dots, n_u n_y$. These parameters have been tuned to generate the best results that we can get.

In Figure 8A, we plot the (θ, q) trajectories. The black dashed curve represents the trajectory if one uses model (113) to compute the feedback gain and applies it to the true system (109); the red dash-dotted curve represents the trajectory if one uses the true system (109) to compute the feedback gain (referred to as “LQ-true”). We can observe that there is a significant mismatch between the black curve and the red curve—the performance of the feedback gain computed using (113) may not be satisfactory as the LQ-true trajectory represents the user-desired response. The blue solid curve represents the trajectory obtained using our proposed approach with perturbations added after convergence. It matches the red curve well. The cyan curves show the trajectory evolution over the iterations. In Figure 8B, we plot the cost evolution over the iterations corresponding to different implementations—the black dash-dotted curve corresponds to the implementation of Algorithm 4 without adding perturbations; the blue solid curve corresponds to the implementation of Algorithm 4 with perturbations added; the green dashed curve corresponds to the implementation of the RL algorithm, Algorithm 6. It can be observed that (i) although the feedback gain obtained based on (113) also stabilizes the system, its performance is not satisfactory in terms of the cost, which represents a measure of the control performance defined by the user. This is verified by the value of the initial point of the cost evolution curves. (ii) The implementation of our algorithm with perturbations added exhibits a similar speed of cost decrease and convergence to that of our implementation of the RL algorithm. (iii) Our algorithm converges and achieves a significant amount of cost decrease within the first 10 iterations, even without adding perturbations. On the other hand, probing signals are necessary to the RL algorithm—without adding probing signals, ie, setting $e_i^{\{k\}} = 0$ in Algorithm 6, the iterations diverge. Its plot is omitted as it provides no additional information. (iv) Without adding perturbations, our algorithm converges to a local minimum; the addition of perturbations in the algorithm can effectively help the algorithm escape local minima and ultimately converge to a solution very close to the optimal, at the cost of slower convergence.

In Figure 9, we plot the cost evolution over the iterations using the ES algorithm, Algorithm 7. We can observe that it takes thousands of iterations for the cost to converge to a local minimum. We remark that (i) probing signals are also necessary to the ES algorithm and that (ii) the ES algorithm is non–model based and does not need to know the order of the system, which our iterative algorithm and the RL algorithm both need to know.

On the basis of the above results, our iterative approach, with perturbations added, is competitive with the RL approach and is superior to the ES approach, in terms of speed of convergence, to the reported infinite-horizon LQ control problem of the F-16 aircraft pitch attitude dynamics. Furthermore, our approach can achieve convergence and cost decrease even without using perturbations, which neither the RL approach nor the ES approach can.

‡ $e \sim \mathcal{U}[\sigma_1, \sigma_2]^n$ represents that e is randomly created according to a uniform distribution over the box $[\sigma_1, \sigma_2]^n$.

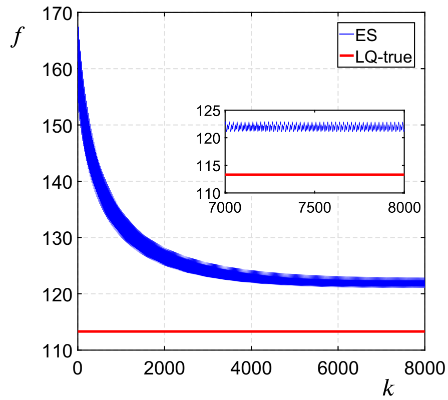

FIGURE 9 Infinite-horizon pitch attitude control *Case 3*. The cost evolution over Algorithm 7 iterations $(k, f(x^{k+1}, v^*))$ (blue) versus the linear quadratic (LQ)-true cost (red). ES, extremum seeking [Colour figure can be viewed at wileyonlinelibrary.com]

Furthermore, we remark that our iterations are cheap in terms of computational time: the average CPU time for one iteration is 2.4 ms, performed on the MATLAB R2016a platform using an Intel Core i7-4790 3.60-GHz PC with Windows 10 and 16.0 GB of RAM, calculated by using the MATLAB tic-toc command, which is also competitive with the RL approach (8.2 ms per iteration).

6 | CONCLUSION

In this paper, we have proposed a novel iterative approach to the finite-horizon and infinite-horizon LQ optimal control of unknown discrete-time linear systems. The iterative approach was applicable to situations where the cost could be evaluated through simulations, or to batch processes, as it required multiple evaluations of the cost as a function of the control subject to the same initial condition.

We compared the performance of the proposed approach to the application of ES in the case of finite-horizon LQ control and to the applications of RL and ES in the case of infinite-horizon LQ control, by considering an example of the pitch attitude control of an AFTI/F-16 aircraft. It was shown that (i) our approach was superior to the ES approach in terms of speed of convergence and implementation complexity in the finite-horizon case and that (ii) our approach was competitive with the RL approach in terms of speed of convergence and computational complexity and was superior to the ES approach in terms of speed of convergence, in the infinite-horizon case.

The cornerstone of our approach to treat optimal control problems is in parameterizing the dynamics to obtain a static parameter-dependent function to minimize. In this paper, we have achieved such a parameterization for a discrete-time linear system by taking advantage of the superposition property. In a future work, the approach may be extended to nonlinear systems exploiting other parameterization techniques. Furthermore, extensions to incorporate state and/or control constraints and to the receding-horizon optimal control for "black box"-type systems will be explored.

ACKNOWLEDGEMENT

Nan Li and Ilya Kolmanovsky acknowledge support from the National Science Foundation through Award CNS 1544844.

ORCID

Nan Li https://orcid.org/0000-0001-7928-8796

REFERENCES

1. Bryson AE, Ho Y-C. *Applied Optimal Control: Optimization, Estimation and Control*. New York, NY: Routledge; 2018.
2. Ljung L. *System Identification: Theory for the User*. Upper Saddle River, NJ: Prentice-hall; 1987.
3. Ioannou PA, Sun J. *Robust Adaptive Control*. Upper Saddle River, NJ: Prentice-Hall; 1996.
4. Boltyansky VG. Robust maximum principle in minimax control. *Int J Control*. 1999;72(4):305-314.

5. Poznyak AS, Duncan TE, Pasik-Duncan B, Boltyanski VG. Robust maximum principle for multi-model LQ-problem. *Int J Control*. 2002;75(15):1170-1177.
6. Bemporad A, Borrelli F, Morari M. Min-max control of constrained uncertain discrete-time linear systems. *IEEE Trans Autom Control*. 2003;48(9):1600-1606.
7. Walton C, Phelps C, Gong Q, Kaminer I. A numerical algorithm for optimal control of systems with parameter uncertainty. *IFAC-Pap*. 2016;49(18):468-475.
8. Dierks T, Thumati BT, Jagannathan TS. Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. *Neural Netw*. 2009;22(5-6):851-860.
9. Lewis FL, Vamvoudakis KG. Reinforcement learning for partially observable dynamic processes: adaptive dynamic programming using measured output data. *IEEE Trans Syst Man Cybern B Cybern*. 2011;41(1):14-25.
10. Wang D, Liu D, Wei Q, Zhao D, Jin N. Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. *Automatica*. 2012;48(8):1825-1832.
11. Lewis FL, Vrabie D. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits Syst Mag*. 2009;9(3):32-50.
12. Recht B. A tour of reinforcement learning: the view from continuous control. arXiv preprint arXiv:1806.09460. 2018.
13. Bellman R. *Dynamic Programming*. Chelmsford, MA: Courier Corporation; 2013.
14. Hudson J, Gupta R, Li N, Kolmanovsky I. Iterative model and trajectory refinement for orbital trajectory optimization. *Optim Control Appl Methods*. 2017;38(6):1132-1147.
15. Li N, Kolmanovsky I, Girard A. Model-free optimal control based automotive control system falsification. Paper presented at: 2017 American Control Conference (ACC); 2017; Seattle, WA.
16. Li N, Girard A, Kolmanovsky I. Optimal control based falsification of unknown systems with time delays: a gasoline engine A/F ratio control case study. Paper presented at: 5th IFAC Conference on Engine and Powertrain Control, Simulation and Modeling (E-CoSM); 2018; Changchun, China.
17. Anderson BDO, Moore JB. *Optimal Control: Linear Quadratic Methods*. Chelmsford, MA: Courier Corporation; 2007.
18. Bradtke SJ. Reinforcement learning applied to linear quadratic regulation. In: *Advances in Neural Information Processing Systems 5*. San Francisco, CA: Morgan Kaufmann Publishers; 1993:295-302.
19. Kawamura Y, Nakano M, Yamamoto H. Model-free recursive LQ controller design (learning LQ control). *Int J Adapt Control Signal Process*. 2004;18(7):551-570.
20. Jiang Y, Jiang Z-P. Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica*. 2012;48(10):2699-2704.
21. Ouyang Y, Gagrani M, Jain R. Learning-based control of unknown linear systems with Thompson sampling. arXiv preprint arXiv:1709.04047. 2017.
22. Abbasi-Yadkori Y, Lazic N, Szepesvari C. The return of ϵ-greedy: sublinear regret for model-free linear quadratic control. arXiv preprint arXiv:1804.06021. 2018.
23. Fazel M, Ge R, Kakade S, Mesbahi M. Global convergence of policy gradient methods for the linear quadratic regulator. In: Proceedings of the 35th International Conference on Machine Learning (ICML); 2018; Stockholm, Sweden.
24. Zhao Q, Xu H, Sarangapani J. Finite-horizon near optimal adaptive control of uncertain linear discrete-time systems. *Optim Control Appl Methods*. 2015;36(6):853-872.
25. Fong J, Tan Y, Crocher V, Oetomo D, Mareels I. Dual-loop iterative optimal control for the finite horizon LQR problem with unknown dynamics. *Syst Control Lett*. 2018;111:49-57.
26. Ariyur KB, Krstić M. *Real-Time Optimization by Extremum-Seeking Control*. Hoboken, NJ: John Wiley & Sons; 2003.
27. Frihauf P, Krstić M, Başar T. Finite-horizon LQ control for unknown discrete-time linear systems via extremum seeking. *Eur J Control*. 2013;19(5):399-407.
28. Liu S-J, Krstić M, Başar T. Batch-to-batch finite-horizon LQ control for unknown discrete-time linear systems via stochastic extremum seeking. *IEEE Trans Autom Control*. 2017;62(8):4116-4123.
29. Dean S, Mania H, Matni N, Recht B, Tu S. On the sample complexity of the linear quadratic regulator. arXiv preprint arXiv:1710.01688. 2017.
30. Dean S, Mania H, Matni N, Recht B, Tu S. Regret bounds for robust adaptive control of the linear quadratic regulator. arXiv preprint arXiv:1805.09388. 2018.
31. Bernstein DS. *Matrix Mathematics: Theory, Facts, and Formulas with Application to Linear Systems Theory*. Princeton, NJ: Princeton University Press; 2005.
32. Boyd S, El Ghaoui L, Feron E, Balakrishnan V. *Linear Matrix Inequalities in System and Control Theory*. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1994.
33. Anderson BDO, Johnson CR. Exponential convergence of adaptive identification and control algorithms. *Automatica*. 1982;18(1):1-13.
34. Drygas H. Weak and strong consistency of the least squares estimators in regression models. *Zeitschrift Wahrscheinlichkeitstheorie Verwandte Gebiete*. 1976;34(2):119-127.
35. Anderson B. Exponential stability of linear equations arising in adaptive identification. *IEEE Trans Autom Control*. 1977;22(1):83-88.
36. Rosenbrock HH. An automatic method for finding the greatest or least value of a function. *Comput J*. 1960;3(3):175-184.
37. Bonvin D, Srinivasan B, Hunkeler D. Control and optimization of batch processes. *IEEE Control Syst Mag*. 2006;26(6):34-45.

38. Hawkins WM, Fisher TG. *Batch Control Systems: Design, Application, and Implementation*. Research Triangle Park, NC: ISA; 2006.
39. Sobel KM, Shapiro EY. A design methodology for pitch pointing flight control systems. *J Guid Control Dyn*. 1985;8(2):181-187.
40. McDonough K, Kolmanovsky I. Fast computable recoverable sets their use for aircraft loss-of-control handling. *J Guid Control Dyn*. 2017;40(4):934-947.
41. Andrade JPP, Campos VAF. Robust control of a dynamic model of an F-16 aircraft with improved damping through linear matrix inequalities. *Int J Comput Electr Autom Control Inf Eng*. 2017;11(2):230-236.
42. Russell RS. *Non-linear F-16 Simulation Using Simulink and Matlab*. Technical Paper. Minneapolis, MN: University of Minnesota; 2003.
43. Garza FR, Morelli EA. *A Collection of Nonlinear Aircraft Simulations in Matlab*. Technical Report. Hampton, VA: NASA Langley Research Center; 2003.
44. Hueschen RM. *Development of the Transport Class Model (TCM) Aircraft Simulation From a Sub-Scale Generic Transport Model (GTM) Simulation*. Technical Report. Hampton, VA: NASA Langley Research Center; 2011.
45. Grauer JA, Morelli EA. Generic global aerodynamic model for aircraft. *J Aircr*. 2015;52(1):13-20.
46. Dussart G, Portapas V, Pontillo A, Lone M. Flight dynamic modelling and simulation of large flexible aircraft. In: *Flight Physics: Models, Techniques and Technologies*. London, UK: IntechOpen Limited; 2018:247.
47. MathWorks Inc. c2d: convert model from continuous to discrete time. https://www.mathworks.com/help/control/ref/c2d.html. Accessed July 25, 2018.

How to cite this article: Li N, Kolmanovsky I, Girard A. LQ control of unknown discrete-time linear systems—A novel approach and a comparison study. *Optim Control Appl Meth*. 2019;40:265–291. https://doi.org/10.1002/oca.2477