IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 26, 2019, accepted August 10, 2019, date of publication August 20, 2019, date of current version September 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2936537

Robust Online Learning Method Based on Dynamical Linear Quadratic Regulator

HANWEN NING[1], JIAMING ZHANG[1], XINGJIAN JING[2], (Senior Member, IEEE), AND TIANHAI TIAN[3]

[1]School of Statistics and Mathematics, Zhongnan University of Economics and Law, Wuhan 430073, China
[2]Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong
[3]School of Mathematics, Monash University, Melbourne, VIC 3800, Australia

Corresponding author: Tianhai Tian (tianhai.tian@monash.edu)

This work was supported in part by the National Natural Science Foundation of China under Project 11301544, Project 61773401, and Project 11571368, in part by the National Social Science Foundation of China under Project 19BTJ025, in part by the China Scholarship Council under Project 201707085011, in part by the Research Grants Council, University Grants Committee, Hong Kong, through the General Research Fund under Project 15206717, and in part by the Internal Research Grants through The Hong Kong Polytechnic University.

ABSTRACT In this paper, a novel algorithm is proposed for inferring online learning tasks efficiently. By a carefully designed scheme, the online learning problem is first formulated as a state feedback control problem for a series of finite-dimensional systems. Then, the online linear quadratic regulator (OLQR) learning algorithm is developed to obtain the optimal parameter updating. Solid mathematical analysis on the convergence and rationality of our method is also provided. Compared with the conventional learning methods, our learning framework represents a completely different approach with optimal control techniques, but does not introduce any assumption on the characteristics of noise or learning rate. The proposed method not only guarantees the fast and robust convergence but also achieves better performance in learning efficiency and accuracy, especially for the data streams with complex noise disturbances. In addition, under the proposed framework, new robust algorithms can be potentially developed for various machine learning tasks by using the powerful optimal control techniques. Numerical results on benchmark datasets and practical applications confirm the advantages of our new method.

INDEX TERMS Online machine learning, optimal control, linear quadratic regulator, complex noise disturbances.

I. INTRODUCTION

As an important subtopic of machine learning, online learning has attracted increasing attention during the past decade due to its extensive applications to realistic modeling problems, for instance, online advertising, financial quantitative transaction, and mechanical damage detection [1]–[4]. In the online learning, data become available in a stream, and predictive models are required to be updated in a real time manner. Therefore, two key issues need to be addressed in developing new algorithms. The first issue is to absorb new information from the incoming data flow and incrementally update the predictive model. The other one is to remove the old and useless information to maintain the parsimony of the model and restrain the computation load within a certain lever.

A wide variety of online learning methods have been proposed during the last decade. A major approach is the gradient based method, in which the update is often given by solving an empirical error minimization problem [5]–[8]. The gradient descent principle is introduced to ensure that the computation load will not increase substantially when the data sequentially come into the learning. The convergence rates and approximation error have also been studied in literature [9], [10]. Another major approach is the online version of batch learning algorithms [11]–[13]. In these algorithms, the learning model is updated by repeatedly solving the corresponding regularized error minimization problem when the new instances are subsequently added. The sample selection techniques such as the moving window strategy,

The associate editor coordinating the review of this article and approving it for publication was Shagufta Henna.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

fast leave one out, and pruning error minimization, have been applied to reduce the computation complexity, meanwhile improve the sparsity and generalization performance of the learning model [14]–[19]. Among them, passive aggressive (PA) algorithms are a family of margin based online learning methods [20]–[25]. Instead of penalizing the complexity of the model, PA algorithms penalize the increments of learning model and update the model when the predictive error exceeds a predetermined threshold value. Compared with the gradient based methods, PA algorithms have shown advantages in robustness due to their less sensitivity to the parameter setting and adaptive learning rate.

Although the existing methods are very useful for many real-world applications, there still exist several challenges when these methods are applied to complex data streams. A major limitation is the deficiency of coping with noise effects. Regarding the batch learning methods, when complex random disturbances are encountered, the optimization in the least square form needs to be modified to improve the prediction efficiency. For example, for data with heterogeneous noise, the weighted least square regression is often utilized. However, in online learning cases, it is impossible to compute the weighted parameters or obtain a stable structure of noise effects. Thus, the sample selection techniques and modification strategies cannot be well incorporated into the learning simultaneously. In the gradient based methods and PA methods, the updates rely on the feedback of error signals. In many realistic applications, the error signals may be inevitably corrupted by noise and thus provide incorrect gradient information for updates. This problem will be even worse if the learning is conducted in the environments of high intensity noise. The updates will be continuously misguided, and the convergence rate will be seriously affected. Another limitation is that some key learning parameters, such as the length of the window in the moving window regression [26] and the learning rate in the gradient based methods [27], are difficult to be adjusted. It is noted that in online learning, the system under studies often experiences abrupt changes and the objective model is usually time-varying. Therefore, a satisfactory modeling performance cannot always be guaranteed with constant learning parameters. Unfortunately, in the majority of the literature, these parameters are predetermined based on heuristic knowledge, which also brings substantial challenges to the online learning problems.

In this paper, we introduce the state feedback control theory into the modeling of data streams and propose a novel online learning approach. By a carefully designed numerical scheme, the online learning problem is reasonably transformed into the state feedback control problems for a series of finite-dimensional, controllable, and completely observable systems. Dynamical linear quadratic regulator is introduced to solve the corresponding optimal control problem. Two algorithms, named as the online linear quadratic regulator learning algorithm (OLQR) and online kernel linear quadratic regulator learning algorithm (OKLQR) are developed for the learning problems in the linear space and reproducing kernel Hilbert space respectively. Since a completely different approach is explored in our framework, there is no need to introduce any online adjustment to the learning parameters, complex data window or pruning techniques, which enables our method to overcome the aforementioned limitations of the existing methods. Compared with the conventional methods, the proposed algorithms can achieve better performance in both learning efficiency and prediction accuracy regardless of the characteristics of noise disturbances. Although a number of pioneering approaches have been proposed for developing control-based learning algorithms [28]–[30], the learning problems were transformed into the output feedback control problems in these works rather than the state feedback control problems proposed in our method. Our approach not only bring better control performance for learning but also leads to strict mathematical analysis on the convergence, all of which rationalize our proposed method for establishing a solid learning framework.

Notation: The real field is denoted by R, while R^M denotes the set of real vectors of size M. The superscript $(\cdot)^T$ and $(\cdot)^{-1}$ denote the transpose and inverse of a matrix, respectively. $< \cdot, \cdot >$ and $|| \cdot ||$ denote the inner product and norm of a Hilbert space respectively. Vectors are denoted by bold letters and matrix by capital letters in Spencerian fonts i.e., $\boldsymbol{s}$ and $\mathcal{S}$ respectively. I denotes the identity matrix, and finally $0 \prec \mathcal{S} \prec I$ means that the symmetric matrix $\mathcal{S}$ and $I - \mathcal{S}$ are both positive definite.

II. BENCHMARK ONLINE LEARNING METHODS

We start with a brief review on some benchmark online learning methods. The issues discussed in this section for the linear system can be naturally extended to the nonlinear cases. The gradient based methods are the most popular algorithms in optimization and so far the most common approach in the online learning strategies. Assume that there is a sequence of random samples $(\boldsymbol{x}(n), y(n))$ $(n = 1, 2, \ldots)$ generated by the model

$$y(n) = f(\boldsymbol{x}(n)) + \varepsilon(n) = \boldsymbol{x}(n)\boldsymbol{\beta}^{\star} + \varepsilon(n), \tag{1}$$

where $\boldsymbol{x}(n) \in R^M$, $y(n) \in R$ and ε is the random term. Here $\boldsymbol{\beta}(n)$ denotes the estimate of $\boldsymbol{\beta}^{\star}$ at time slot n, and the prediction error is defined as $e(n) = y(n) - \boldsymbol{x}(n)\boldsymbol{\beta}(n)$. In the stochastic gradient methods [5], [6], the learning law is given by minimizing the following instantaneous risk

$$R_{inst}(\boldsymbol{\beta}, \boldsymbol{x}(n), y(n)) = \frac{1}{2}e(n)^2 + \frac{1}{2}\lambda||\boldsymbol{\beta}||^2. \tag{2}$$

The update is given by

$$\begin{aligned}\boldsymbol{\beta}(n+1) &= \boldsymbol{\beta}(n) - \eta\frac{\partial R_{inst}(\boldsymbol{\beta}(n), \boldsymbol{x}(n), y(n))}{\partial \boldsymbol{\beta}(n)} \\ &= (1 - \eta\lambda)\boldsymbol{\beta}(n) + \eta(y(n) - \boldsymbol{x}(n)\boldsymbol{\beta}(n))\boldsymbol{x}(n)^T \\ &= (1 - \eta\lambda)\boldsymbol{\beta}(n) + \eta e(n)\boldsymbol{x}(n)^T,\end{aligned} \tag{3}$$

where $\lambda > 0$ is the regularization parameter and η is the learning rate. Some gradient algorithms are also proposed

without an explicit regularization, i.e. $\lambda = 0$, given by

$$\boldsymbol{\beta}(n+1) = \boldsymbol{\beta}(n) + \eta e(n)\boldsymbol{x}(n)^T. \tag{4}$$

The learning algorithms displayed above are also referred as stochastic approximation methods or least mean square methods. A major limitation of the gradient based methods is the deficiency of coping with noise. The error signal $e(n)$ is used as the distance between $\boldsymbol{\beta}(n)$ and $\boldsymbol{\beta}^\star$. If $e(n) = 0$, the algorithm considers that $\boldsymbol{\beta}^\star$ is accurately estimated, and $\boldsymbol{\beta}(n)$ remains steady without any updates. Otherwise, $\boldsymbol{\beta}(n)$ will be updated. Noticed that

$$e(n) = y(n) - \boldsymbol{x}(n)\boldsymbol{\beta}(n) = \boldsymbol{x}(n)(\boldsymbol{\beta}^\star - \boldsymbol{\beta}(n)) + \varepsilon(n), \tag{5}$$

the update of $\boldsymbol{\beta}(n)$ becomes

$$\boldsymbol{\beta}(n+1) = \boldsymbol{\beta}(n) + \eta\boldsymbol{x}(n)(\boldsymbol{\beta}^\star - \boldsymbol{\beta}(n))\boldsymbol{x}(n)^T + \eta\varepsilon(n)\boldsymbol{x}(n)^T. \tag{6}$$

The error signal $e(n)$ is inevitably corrupted by noise $\varepsilon(n)$, and may provide false information for the online estimation. Another difficulty is the determination of learning rate η. A large learning rate may lead to quick convergence, but also raise the risk of instability of the algorithm. However, a small enough learning rate can guarantee the asymptotic convergence but often lead to slow learning and inefficiency for the fast-changing systems. Although the majority of the gradient based methods usually prefer a relatively small rate to ensure the convergence, the range of optimal learning rates is still distinct for online learning problems with time-varying objective functions. Although the adaptability of learning rates has been extensively studied [8]–[10], heuristic knowledge or additional cost is still needed.

In the PA learning methods [20], [23], suppose at time slot n, the algorithm receives instance $(\boldsymbol{x}(n), y(n))$ and makes a prediction $\hat{y}(n) = \boldsymbol{x}(n)\boldsymbol{\beta}(n)$. For the true target value y, the algorithm defines a loss function. For example, the υ-insensitive hinge loss function is defined as

$$\mathcal{L}_\upsilon(\boldsymbol{\beta}; \boldsymbol{x}, y) = \begin{cases} 0, & |\boldsymbol{x}\boldsymbol{\beta} - y| \le \upsilon, \\ |\boldsymbol{x}\boldsymbol{\beta} - y| - \upsilon, & \text{otherwise}, \end{cases} \tag{7}$$

where υ is a positive parameter which controls the sensitivity of the updates. The algorithm sets the law as the solution of following optimization

$$\begin{aligned} \boldsymbol{\beta}(n+1) = argmin_{\boldsymbol{\beta}\in R^M} \frac{1}{2}||\boldsymbol{\beta} - \boldsymbol{\beta}(n)||^2, \\ s.t.\ \mathcal{L}_\upsilon(\boldsymbol{\beta}; \boldsymbol{x}(n), y(n)) = 0. \end{aligned} \tag{8}$$

The update given in (8) has a closed form solution as

$$\boldsymbol{\beta}(n+1) = \boldsymbol{\beta}(n) + sign(y(n) - \hat{y}(n))\tau_n\boldsymbol{x}(n), \tag{9}$$

where $\tau_n = \mathcal{L}_\upsilon(\boldsymbol{\beta}(n); \boldsymbol{x}(n), y(n))$. From (8) and (9), it can be seen that the error signal $e(n)$ along with learning rate $\tau(n)$ are both corrupted by random disturbances, and the updates can be misguided for the similar reason that we have discussed for the gradient methods. The learning parameters are usually heuristically chosen to obtain a satisfactory modeling performance.

For online batch learning methods [11]–[13], suppose that at the beginning, we have data $(\boldsymbol{x}(n), y(n))$, $n = 1, 2, \ldots, N$, the optimization problem with regularization is given by

$$\min_\beta \sum_{n=1}^{N} (y(n) - \boldsymbol{x}(n)\boldsymbol{\beta})^2 + \gamma||\boldsymbol{\beta}||^2, \tag{10}$$

or by a more general formula, which can include the nonlinear case

$$\min_{\hat{f}} \sum_{n=1}^{N} (y(n) - \hat{f}(\boldsymbol{x}(n)))^2 + \gamma||\hat{f}||^2, \tag{11}$$

where $\hat{f}$ is a nonlinear approximator. This method cannot be directly applied to online cases since all the samples will be added into the learning. The estimated model will be a lack of sparsity and become computationally infeasible. Some sparsification techniques have been developed. For example, the moving window method was proposed in [26], which assumes that the earliest data in a window contains the least information. The pruning error minimization method was also developed in [14]. This method selects the sample that brings the smallest error after it has been pruned.

It is noticed that realistic random disturbances are usually heterogeneous, temporal correlated and even non-Gaussian. Some modifications must be made to the optimization methods [31]–[33]. For the case of heterogeneous noise, the weighted least square optimization is usually employed

$$\min \sum \sigma_n e(n)^2 + \gamma||\hat{f}||^2, \quad s.t.\ e(n) = y(n) - \hat{f}(\boldsymbol{x}(n)), \tag{12}$$

where σ_n is the weight parameter. However, for the time varying data stream, it is extremely cumbersome or impossible to obtain stable and efficient estimates for σ_n in a real time manner.

III. A BRIEF INTRODUCTION OF OPTIMAL CONTROL

To demonstrate our motivations, we give a brief introduction of optimal control for discrete linear systems. Optimal control is a mature mathematical discipline with a wide range of applications in science and engineering. Consider the following linear control system

$$\boldsymbol{Z}(n+1) = \mathcal{A}\boldsymbol{Z}(n) + \mathcal{B}\boldsymbol{\theta}(n), \quad n \ge 0, \tag{13}$$

where $\mathcal{A}$ and $\mathcal{B}$ are coefficient matrices, $\boldsymbol{Z}(n)$ is the state vector series and $\boldsymbol{\theta}(n)$ is the control input vector. The objective of optimal control is to find a control policy that stabilizes systems, and optimizes a specific performance index for systems. With controller gain $\mathcal{F}$ and control input $\boldsymbol{\theta}(n) = \mathcal{F}\boldsymbol{Z}(n)$, the closed loop system $\boldsymbol{Z}(n+1) = (\mathcal{A}+\mathcal{B}\mathcal{F})\boldsymbol{Z}(n)$ is stable, i.e. $\mathcal{A}+\mathcal{B}\mathcal{F}$ is contractive. In optimal control, an efficient $\mathcal{F}$ can be obtained by minimizing following performance criterion

including a cost function that penalizing the state and control input simultaneously

$$\sum_{\tau=0}^{N-1}(\boldsymbol{Z}(\tau)^T\mathcal{Q}_0\boldsymbol{Z}(\tau)+\boldsymbol{\theta}(\tau)^T\mathcal{R}_0\boldsymbol{\theta}(\tau))+\boldsymbol{Z}(N)^T\mathcal{S}_0\boldsymbol{Z}(N),$$

where $\mathcal{S}_0$, $\mathcal{Q}_0$ and $\mathcal{R}_0$ are constant symmetric positive matrices, and N is a positive constant. Despite the accumulated optimal control models and methods, there are still limited approaches for addressing the online learning problems from the perspective of optimal control.

As discussed in the previous section, the prediction error is used in online learning to infer the distances between the model parameters and objective parameters, and the online updating of model parameters ($\Delta\boldsymbol{\beta}(n)=\boldsymbol{\beta}(n+1)-\boldsymbol{\beta}(n)$) is supposed to minimize the prediction error as much as possible. This process is similar to that in the stabilization problem of control theory, for example, obtaining a series of control inputs $\boldsymbol{\theta}(n)$s to stabilize (13) and making $\boldsymbol{Z}(n)$ converges to the original point, which sheds light on the motivations of the technical methods to be developed in this paper. In the following sections, we develop numerical schemes using the prediction errors as state variables of control system and the parameter updates as control input. Meanwhile, we also elaborate the connections between the optimal control and online learning.

IV. ONLINE LEARNING FRAMEWORK

In this section, we establish a novel online learning framework from the perspective of state feedback control. Consider an adaline machine with M input variables. Suppose there is a data stream $(\boldsymbol{x}(1),y(1)),\ldots,(\boldsymbol{x}(k),y(k)),\ldots$ generated by

$$y(k)=\sum_{j=1}^{M}\beta_j^\star x_j(k)+\varepsilon(k)=\boldsymbol{x}(k)\boldsymbol{\beta}^\star+\varepsilon(k),\tag{14}$$

where $\boldsymbol{\beta}^\star=(\beta_1^\star,\beta_2^\star,\ldots,\beta_M^\star)^T$ is the objective parameter vector. Let $\boldsymbol{\beta}(n)$ be the estimated parameter vector of $\boldsymbol{\beta}^\star$ at time slot n. Assume that at time slot n, we already have an estimated $\boldsymbol{\beta}(n)$, which will be updated by $\boldsymbol{\beta}(n+1)=\boldsymbol{\beta}(n)+\Delta\boldsymbol{\beta}(n)$. A projected estimation of $y(k)$ by $\boldsymbol{\beta}(n+1)$ is

$$\hat{y}(k)=\sum_{j=1}^{M}\beta_j(n+1)x_j(k)=\boldsymbol{x}(k)\boldsymbol{\beta}(n+1).\tag{15}$$

Let $\hat{e}(n-l)$ be the projected prediction error by $\boldsymbol{\beta}(n+1)$, i.e.

$$\begin{aligned}\hat{e}(n-l)&=\hat{y}(n-l)-y(n-l)\\&=\boldsymbol{x}(n-l)\boldsymbol{\beta}(n+1)-\boldsymbol{x}(n-l)\boldsymbol{\beta}^\star-\varepsilon(n-l)\\&=\boldsymbol{x}(n-l)(\boldsymbol{\beta}(n+1)-\boldsymbol{\beta}^\star)-\varepsilon(n-l),\end{aligned}\tag{16}$$

for $l=0,1,\ldots,M-1$. Let $e(n-l)$ be the prediction error by $\boldsymbol{\beta}(n)$,

$$\begin{aligned}e(n-l)&=\boldsymbol{x}(n-l)\boldsymbol{\beta}(n)-\boldsymbol{x}(n-l)\boldsymbol{\beta}^\star-\varepsilon(n-l)\\&=\boldsymbol{x}(n-l)(\boldsymbol{\beta}(n)-\boldsymbol{\beta}^\star)-\varepsilon(n-l),\end{aligned}\tag{17}$$

for $l=0,1,\ldots,M-1$. It follows that

$$\begin{aligned}&\hat{e}(n-l)\\&=\boldsymbol{x}(n-l)(\boldsymbol{\beta}(n)-\boldsymbol{\beta}^\star+\Delta\boldsymbol{\beta}(n))-\varepsilon(n-l)\\&=\boldsymbol{x}(n-l)(\boldsymbol{\beta}(n)-\boldsymbol{\beta}^\star)-\varepsilon(n-l)+\boldsymbol{x}(n-l)\Delta\boldsymbol{\beta}(n)\\&=e(n-l)+\boldsymbol{x}(n-l)\Delta\boldsymbol{\beta}(n),\end{aligned}\tag{18}$$

for $l=0,1,\ldots,M-1$. For all l, with

$$\begin{aligned}\hat{\boldsymbol{E}}(n)&\equiv[\hat{e}(n),\hat{e}(n-1),\ldots,\hat{e}(n-M+1)]^T,\\\boldsymbol{E}(n)&\equiv[e(n),e(n-1),\ldots,e(n-M+1)]^T,\\\Delta\boldsymbol{\beta}(n)&\equiv[\Delta\beta_1(n),\Delta\beta_2(n)\ldots,\Delta\beta_M(n)],\\\mathcal{B}(n)&\equiv\begin{bmatrix}x_1(n)&\cdots&x_M(n)\\x_1(n-1)&\cdots&x_M(n-1)\\\vdots&\cdots&\vdots\\x_1(n-M+1)&\cdots&x_M(n-M+1)\end{bmatrix},\end{aligned}$$

we rewrite (18) as

$$\hat{\boldsymbol{E}}(n)=\boldsymbol{E}(n)+\mathcal{B}(n)\Delta\boldsymbol{\beta}(n).\tag{19}$$

To investigate the learning problem from optimal control perspective, we further denote (19) as

$$\boldsymbol{E}(n+1)=\boldsymbol{E}(n)+\mathcal{B}(n)\boldsymbol{U}(n),\tag{20}$$

where $\boldsymbol{U}(n)=\Delta\boldsymbol{\beta}(n)$ is a feedback control input waited to be determined. This is a linear, discrete and finite dimensional error dynamical system with parameter vector $\mathcal{B}(n)$ and control input $\boldsymbol{U}(n)$. Based on (20), the update for the learning of linear model is transformed into a typical control problem of a discrete dynamical system with finite dimensional control input $\boldsymbol{U}(n)$ [34]. Generally, to realize online learning, $\boldsymbol{U}(n)$ is the feedback to the observations, i.e. $\boldsymbol{U}(n)=\mathcal{F}(n)\boldsymbol{E}(n)$, where $\mathcal{F}(n)$ is the controller gain to be determined from the optimal control problem. It is noted that $\boldsymbol{\beta}(n)$ is assumed to be known at time slot n. Therefore, $e(n-l)$ can be observed and the problem can be regarded as a state feedback control problem. The objective of the learning is to obtain an efficient $\mathcal{F}(n)$ to stabilize the error system (20) and make the state $\boldsymbol{E}(\cdot)$ be contractive to the origin point. This implies that after the update, the prediction error by $\boldsymbol{\beta}(n+1)$ will be smaller than that by $\boldsymbol{\beta}(n)$ on the given sample set, from the point view of algebraic mapping. As n grows, a series of time invariant control systems can also be formulated. If we sequentially stabilize these systems, a series of control inputs $\boldsymbol{U}(n+k)=\Delta\boldsymbol{\beta}(n+k)$ $(k=0,1,2,\ldots)$ can be obtained as well and

$$\boldsymbol{\beta}(n+k)=\boldsymbol{\beta}(n)+\sum\nolimits_{j=0}^{k-1}\Delta\boldsymbol{\beta}(n+j).\tag{21}$$

In the following, we show that the distance between $\boldsymbol{\beta}(n+k)$ and $\boldsymbol{\beta}^\star$ will be restricted in a compact set as $k\to\infty$. Moreover, in literatures, $\varepsilon(n)$ is usually set to be normal to facilitate the statistical analysis. However, random perturbations are always show the characteristics of boundness. For example, the return rate of the securities in stock market is usually assumed to be normal in econometrics models, but in fact it is bounded [35]. Therefore, $\varepsilon(n)$ is assumed to be bounded

by a constant D (i.e. $|\varepsilon(n)| < D$). Suppose the corresponding closed loop system for (20) is

$$\boldsymbol{E}(n+1) = [I + \mathcal{B}(n)\mathcal{F}(n)]\boldsymbol{E}(n), \tag{22}$$

where $I + \mathcal{B}(n)\mathcal{F}(n)$ is a state transition matrix for the control system. Denote $\widetilde{\boldsymbol{\beta}}(n) = (\beta_1(n) - \beta_1^\star, \beta_2(n) - \beta_2^\star, \ldots, \beta_M(n) - \beta_M^\star)^T$, and $\boldsymbol{d}(n) = (\varepsilon(n), \varepsilon(n-1), \ldots, \varepsilon(n-M+1))^T$. The basic results related to the stability of finite dimensional, discrete, linear system are employed to obtain the convergence of $\boldsymbol{\beta}(n)$ [48]. Thus we have the following theorem

Theorem 1: Assume that $\mathcal{B}(n)$ is invertible for all n. If the feedback control input $\boldsymbol{U}(n) = \mathcal{F}(n)\boldsymbol{E}(n)$ makes the transition matrix $\mathcal{T}(n) = I + \mathcal{B}(n)\mathcal{F}(n)$ to be positive and contractive, i.e. $0 \prec \mathcal{T}(n) \prec I$, then, for any initial value $\boldsymbol{\beta}(1)$ and $\boldsymbol{\beta}(n)$ obtained by $\boldsymbol{\beta}(n) = \boldsymbol{\beta}(1) + \sum_{j=0}^{n-1} \boldsymbol{U}(j)$, there exists a $D_e (= O(D))$ such that

$$\lim_{n\to\infty} ||\boldsymbol{\beta}(n) - \boldsymbol{\beta}^\star|| < D_e. \tag{23}$$

Proof: See the appendix A.

It is well known that in linear regression analysis, if the variables are perfectly multiple collinear, i.e. one independent variable is an exact linear combination of the others, there is no method to obtain a unique and promising learning model. Therefore, $\mathcal{B}(n)$ is assumed to be full rank and invertible. This also guarantees that system (20) is completely controllable, which greatly facilitates the development of our new method. Thus the estimation error of $\boldsymbol{\beta}(n)$ will be convergent to a compact set including the origin as $n \to \infty$ with a series of carefully designed control inputs $\boldsymbol{U}(n)$, which means it is possible to develop efficient learning algorithms from the perspective of state feedback control theory. When the intensity of $e(n)$ is zero ($D = 0$), the compact set can be extremely small in this noise-free case.

V. ROBUST ONLINE LEARNING METHOD BASED ON LQR

The theory of optimal control is concerned with operating a dynamic system at minimum cost. The case where the system dynamics are described by a set of linear equations and the cost is described by a quadratic function is called the linear quadratic (LQ) problem [34], [36], [37]. One of the main results for the LQ problem is the linear quadratic regulator, which is a well known method that provides optimally controlled feedback gains to enable the closed loop stable and high performance design of systems. For systems controlled by LQR, control inputs and plant responses are predicted using the system state space model and optimized over a family of piecewise constant intervals with respect to a cost function with weighting factors including penalties on the system states and control inputs. Once the optimization problem is solved, only the control input of the current time slot is implemented. This optimization procedure is then repeated in the next time slot to continuously generate a series of efficient control inputs.

In this paper, the infinite horizon LQR is utilized to obtain the optimal state feedback control inputs to establish our online learning method. For system (20), we construct a virtual time invariant dynamical control system as follows

$$\boldsymbol{E}_n(t+1) = \boldsymbol{E}_n(t) + \mathcal{B}_n\boldsymbol{U}_n(t), \quad t = 1, 2, \ldots, \tag{24}$$

where $\mathcal{B}_n = \mathcal{B}(n)$ with $\boldsymbol{E}_n(1) = \boldsymbol{E}(n)$. System (24) is regarded as a time invariant (static) linear system. It is easy to verify that (24) is a completely controllable and observable as long as $\mathcal{B}_n$ is full rank, which implies there exists an optimal control that can stabilize (24) according to the control theory [37], [38]. To develop the learning method by LQR, the infinite time horizon optimization problem is constructed as

$$\begin{aligned} V \;=\; & \min_{\boldsymbol{U}_n(1),\ldots,} \sum_{t=1}^{\infty} \boldsymbol{E}_n(t)^T \mathcal{Q}\boldsymbol{E}_n(t) + \boldsymbol{U}_n(t)^T \mathcal{R}\boldsymbol{U}_n(t), \\ & s.t.\ \boldsymbol{E}_n(t+1) = \boldsymbol{E}_n(t) + \mathcal{B}_n\boldsymbol{U}_n(t), \\ & \boldsymbol{U}_n(t) = \mathcal{F}_n\boldsymbol{E}_n(t), \end{aligned} \tag{25}$$

where $\mathcal{F}_n$ is the controller gain to be determined. $\mathcal{R}$ and $\mathcal{Q}$ are systematic and semi-definite matrices. These matrices can be chosen to obtain a desirable closed loop response. Here $\mathcal{Q}$ and $\mathcal{R}$ are set to unit matrices I and γI, respectively. The first term in V measures the output deviation, and the second term penalizes the intensity of control input. In addition, $\gamma > 0$ is a tradeoff parameter to weight the two goals of the optimization. Once the solution of (25) is obtained, the control input in (20) and parameter update for the learning model is given by applying only the first control input $\boldsymbol{U}_n(1)$ as $\Delta\boldsymbol{\beta}(n) = \boldsymbol{U}(n) = \boldsymbol{U}_n(1) = \mathcal{F}_n\boldsymbol{E}(n)$. At the next time slot $n+1$, $\mathcal{B}_n$ in (24) is updated to $\mathcal{B}_{n+1}$ accordingly. The corresponding LQR problem (25) is solved again to obtain the update law $\Delta\boldsymbol{\beta}(n+1)$. This procedure will be repeated to update the model in a real time manner as the observation data are continuously added into the learning process, which is also named as the dynamical LQR in this study. The techniques for obtaining the solution of linear quadratic optimization problem are also utilized to obtain the following schemes of our algorithm [37], [39].

Theorem 2: The solution of optimization problem (25) can be given by solving the following matrix equation for $\mathcal{P}_n$, given by

$$\mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n = \mathcal{Q}. \tag{26}$$

In addition, the optimal controller gain and parameter update are given as

$$\begin{aligned} \boldsymbol{\beta}(n+1) &= \boldsymbol{\beta}(n) + \mathcal{F}_n\boldsymbol{E}(n), \\ \mathcal{F}_n &= -(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n. \end{aligned} \tag{27}$$

The estimation error by applying the update law is convergent, i.e. there exists a constant $D_e = O(D)$ such that

$$\lim_{n\to\infty} ||\boldsymbol{\beta}(n) - \boldsymbol{\beta}^\star|| < D_e. \tag{28}$$

Proof: See the appendix B.

It can be seen from the proof of Theorem 2 that with the update law obtained by the infinite horizon LQR, the distance between $\boldsymbol{\beta}(n)$ and $\boldsymbol{\beta}^\star$ will be exponentially convergent to a

compact set including the origin as $n \to \infty$, which leads to an efficient and fast convergent online learning method named OLQR in this paper. In addition, the compact set and the estimation error can be extremely small in the noise free cases. Therefore, the dynamical LQR and the concept of state feedback control can be well introduced into the machine learning of data streams, which is one of the main contributions of this paper. Algorithm 1 shows the detailed steps of the proposed method for the learning of adaline system (14). Regarding the time horizon N in the index (25), from the perspective of optimal control, the solution obtained from LQR over a large time horizon can usually stabilize the closed loop system. However, the system may be still unstable with a time horizon that is not long enough. This problem can be entirely avoided if the controlled performance is evaluated over an infinite prediction horizon i.e. $N = \infty$, which not only ensure the convergence but also lead to an explicit, convenient and consistent computational scheme for the online learning [37], [39]. The algorithm is named as OLQR (for system (14)) and summarized as follows

1) Initialization: the number of variables M.
2) Let the system run for M steps, and $\boldsymbol{\beta}(M) = 0$.
3) $\boldsymbol{\beta}(n)$ is updated as $\boldsymbol{\beta}(n+1)$ according to (26) and (27).
4) If data flow ends, stop; otherwise n=n+1, go to step 3

VI. THE ONLINE LEARNING IN KERNEL SPACES

In this section, we consider the online learning problem in the reproducing kernel Hilbert space H_K associated with inner product $< \cdot, \cdot >_H$ and feature mapping $\boldsymbol{\phi}$. The kernel K for H_K has the reproducing property [8], [9], i.e. $< f, K(\boldsymbol{x}, \cdot) >_H = f(\boldsymbol{x})$ and $< \boldsymbol{\phi}(\boldsymbol{x}1), \boldsymbol{\phi}(\boldsymbol{x}2) >_H = < K(\boldsymbol{x}1, \cdot), K(\boldsymbol{x}2, \cdot) >_H = K(\boldsymbol{x}1, \boldsymbol{x}2)$ for $\forall \boldsymbol{x}, \boldsymbol{x}1, \boldsymbol{x}2 \in X$ and $f \in H_K$. Here H_K is the closure of the span of all $K(\boldsymbol{x}, \cdot)$, and X is a set in R^{M_1}. $K(\boldsymbol{x}1, \boldsymbol{x}2) = \exp(-||\boldsymbol{x}1 - \boldsymbol{x}2||^2/\sigma^2)$ is the Gaussian kernel, and σ is the bandwidth. In this study, it is assumed that X is a compact subset.

Suppose there exists a data stream $(\boldsymbol{x}(1), y(1)), \ldots, (\boldsymbol{x}(n), y(n)), \ldots$ generated by $y(n) = f(\boldsymbol{x}(n)) + \varepsilon(n)$, where $\boldsymbol{x}(\cdot) \in R^{M_1}$, f and ε are unknown nonlinear function and random disturbance respectively. Suppose there is an objective $\boldsymbol{w}^\star$ in H_K such that

$$y(n) = \boldsymbol{\phi}(\boldsymbol{x}(n))\boldsymbol{w}^\star + \varepsilon(n), \tag{29}$$

and we have a learning model in H_k

$$\hat{y}(n) = \boldsymbol{\phi}(\boldsymbol{x}(n))\boldsymbol{w}(n), \tag{30}$$

$\boldsymbol{x}(\cdot)$ is assumed to be in the compact set $X \subset R^{M_1}$. At time slot n, we have $\boldsymbol{w}(n)$, which will be updated to $\boldsymbol{w}(n+1)$ by $\Delta\boldsymbol{w}(n)$. The projected error by $\boldsymbol{w}(n+1)$ is set to be $\hat{e}(\cdot)$. By employing the same techniques developed in section IV,

$$\hat{e}(n-l) = \boldsymbol{\phi}(\boldsymbol{x}(n-l))(\boldsymbol{w}(n+1) - \boldsymbol{w}^\star) - \varepsilon(n-l), \tag{31}$$

for $l = 0, 1, 2 \ldots$. $e(n-l)$ is the prediction error by $\boldsymbol{w}(n)$.

$$e(n-l) = \boldsymbol{\phi}(\boldsymbol{x}(n-l))(\boldsymbol{w}(n) - \boldsymbol{w}^\star) - \varepsilon(n-l), \tag{32}$$

for $l = 0, 1, 2 \ldots$. It follows that

$$\hat{e}(n-l) = e(n-l) + \boldsymbol{\phi}(\boldsymbol{x}(n-l))\Delta\boldsymbol{w}(n). \tag{33}$$

Let $l = 0, 1, \ldots, M_0 - 1$, where M_0 will be specified later. Similar to (18) and (19), with

$$\boldsymbol{E}(n+1) \equiv [\hat{e}(n), \hat{e}(n-1), \ldots, \hat{e}(n-M_0+1)]^T,$$
$$\boldsymbol{E}(n) \equiv [e(n), e(n-1), \ldots, e(n-M_0+1)]^T,$$

Equation (33) can be written as

$$\boldsymbol{E}(n+1) = \boldsymbol{E}(n) + \boldsymbol{\Phi}(n)\Delta\boldsymbol{w}(n), \tag{34}$$

where $\boldsymbol{\Phi}(n) = (\boldsymbol{\phi}(\boldsymbol{x}(n))^T, \boldsymbol{\phi}(\boldsymbol{x}(n-1))^T, \ldots, \boldsymbol{\phi}(\boldsymbol{x}(n-M_0+1))^T)^T$. For any n, (34) can be considered as a control system with control $\Delta\boldsymbol{w}(n)$ defined in the kernel space. With the similar techniques developed in (24), for system (34), a virtual system can be designed as

$$\boldsymbol{E}_n(t+1) = \boldsymbol{E}_n(t) + \boldsymbol{\Phi}_n\Delta\boldsymbol{w}_n(t), \tag{35}$$

where $\boldsymbol{\Phi}_n = \boldsymbol{\Phi}(n)$, $\boldsymbol{w}_n(1) = \boldsymbol{w}(n)$ and $\boldsymbol{E}_n(1) = \boldsymbol{E}(n)$. The optimization of LQR with (35) in H_K is designed as

$$V = \min_{\Delta\boldsymbol{w}_n(1),\ldots,} \sum_{t=1}^{\infty} \boldsymbol{E}_n(t)^T\boldsymbol{E}_n(t) + \gamma||\Delta\boldsymbol{w}_n(t)||^2,$$
$$s.t.\ \boldsymbol{E}_n(t+1) = \boldsymbol{E}_n(t) + \boldsymbol{\Phi}_n\Delta\boldsymbol{w}_n(t). \tag{36}$$

It is noticed that $\boldsymbol{w}(n)$ in the kernel space may be infinite dimensional. To reasonably transform (36) into a finite dimensional one, we find optimal solution in a subspace H_K^s rather than in H_K itself [12], [13]. Let $\boldsymbol{u}_i$ $(i = 1, 2, \ldots, M)$ be different vectors in X. H_K^s is the linear subspace of H_K spanned by basis vectors $\boldsymbol{\phi}(\boldsymbol{u}_i)$, $i = 1, \ldots, M$. For $\forall n$, $\boldsymbol{w}(n)$ and objective vector $\boldsymbol{w}^\star$ are then represented in H_K^s as

$$\boldsymbol{w}^\star = \sum_{i=1}^{M} \alpha_i^\star\boldsymbol{\phi}(\boldsymbol{u}_i), \quad \boldsymbol{w}(n) = \sum_{i=1}^{M} \alpha_i(n)\boldsymbol{\phi}(\boldsymbol{u}_i). \tag{37}$$

Then, the prediction errors on the dataset $(\boldsymbol{x}(n-l), y(n-l))$, $l = 0, \ldots, M-1$ by $\boldsymbol{w}(n+1)$ (denoted as $\hat{e}(\cdot)$) and $\boldsymbol{w}(n)$ (denoted as $e(\cdot)$) can be reduced into difference equations with finite parameters

$$\begin{aligned}
&\hat{e}(n-l)\\
&= \boldsymbol{\phi}(\boldsymbol{x}(n-l))(\boldsymbol{w}(n+1) - \boldsymbol{w}^\star) - \varepsilon(n-l)\\
&= \sum_{i=1}^{M}(\alpha_i(n+1) - \alpha_i^\star)K(\boldsymbol{u}_i, \boldsymbol{x}(n-l))\\
&\quad - \varepsilon(n-l)e(n-l)\\
&= \boldsymbol{\phi}(\boldsymbol{x}(n-l))(\boldsymbol{w}(n) - \boldsymbol{w}^\star) - \varepsilon(n-l)\\
&= \sum_{i=1}^{M}(\alpha_i(n) - \alpha_i^\star)K(\boldsymbol{u}_i, \boldsymbol{x}(n-l)) - \varepsilon(n-l).
\end{aligned} \tag{38}$$

Denote $\boldsymbol{\alpha}(n) = (\alpha_1(n), \alpha_2(n), \ldots, \alpha_M(n))^T$, $\alpha_i(n+1) = \alpha_i(n) + \Delta\alpha_i(n)$, for $i = 1, 2, \ldots$. Then

$$\begin{aligned}
&\hat{e}(n-l)\\
&= \sum_{i=1}^{M}(\alpha_i(n+1) - \alpha_i^\star)K(\boldsymbol{u}_i, \boldsymbol{x}(n-l)) - \varepsilon(n-l)\\
&= \sum_{i=1}^{M}(\alpha_i(n) - \alpha_i^\star + \Delta\alpha_i(n))K(\boldsymbol{u}_i, \boldsymbol{x}(n-l)) - \varepsilon(n-l)\\
&= e(n-l) + \sum_{i=1}^{M}\Delta\alpha_i(n)K(\boldsymbol{u}_i, \boldsymbol{x}(n-l)),
\end{aligned} \tag{39}$$

for $l = 0, 1, \ldots, M-1$. With

$$\begin{aligned}
\boldsymbol{E}(n+1) &\equiv [\hat{e}(n), \hat{e}(n-1), \ldots, \hat{e}(n-M+1)]^T, \\
\boldsymbol{E}(n) &\equiv [e(n), e(n-1), \ldots, e(n-M+1)]^T, \\
\Delta\boldsymbol{\alpha}(n) &\equiv [\Delta\alpha_1(n), \Delta\alpha_2(n) \ldots, \Delta\alpha_M(n)], \\
\mathcal{K}_n &\equiv \begin{bmatrix} K(\boldsymbol{u}_1, \boldsymbol{x}(n)) & \cdots & K(\boldsymbol{u}_M, \boldsymbol{x}(n)) \\ K(\boldsymbol{u}_1, \boldsymbol{x}(n-1)) & \cdots & K(\boldsymbol{u}_M, \boldsymbol{x}(n-1)) \\ \vdots & \cdots & \vdots \\ K(\boldsymbol{u}_1, \boldsymbol{x}(n-M^-)) & \cdots & K(\boldsymbol{u}_M, \boldsymbol{x}(n-M^-)) \end{bmatrix}
\end{aligned}$$

where $M^- = M-1$, (39) is denoted as

$$\boldsymbol{E}(n+1) = \boldsymbol{E}(n) + \mathcal{K}_n \Delta\boldsymbol{\alpha}(n). \tag{40}$$

It is also noted that in the subspace H_K^s

$$\begin{aligned}
||\Delta\boldsymbol{w}(n)||^2 &= < \sum_{i=1}^{M} \Delta\alpha_i(n)\boldsymbol{\phi}(\boldsymbol{u}_i), \sum_{i=1}^{M} \Delta\alpha_i(n)\boldsymbol{\phi}(\boldsymbol{u}_i) > \\
&= \Delta\boldsymbol{\alpha}(n)^T \mathcal{G} \Delta\boldsymbol{\alpha}(n),
\end{aligned} \tag{41}$$

where $\mathcal{G} = [K(\boldsymbol{u}_i, \boldsymbol{u}_j)]_{i,j=1,2,\ldots,M}$ is the kernel Gram matrix. Therefore, the index of LQR (35) can be reduced accordingly to a formula with finite control inputs

$$\begin{aligned}
V = \min_{\Delta\boldsymbol{w}_n(1),\ldots,} & \sum\nolimits_{t=1}^{\infty} \boldsymbol{E}_n(t)^T \boldsymbol{E}_n(t) + \gamma \Delta\boldsymbol{\alpha}_n(t)^T \mathcal{G} \Delta\boldsymbol{\alpha}_n(t), \\
s.t.\ & \boldsymbol{E}_n(t+1) = \boldsymbol{E}_n(t) + \mathcal{K}_n \Delta\boldsymbol{\alpha}_n(t).
\end{aligned} \tag{42}$$

By the same method presented in Theorem 2, the optimal update law can be obtained

$$\Delta\boldsymbol{\alpha}(n) = -(\gamma\mathcal{G} + \mathcal{K}_n^T \mathcal{P}_n \mathcal{K}_n)^{-1} \mathcal{K}_n^T \mathcal{P}_n E(n), \tag{43}$$

where $\mathcal{P}_n$ is given by the following matrix equation

$$\mathcal{P}_n \mathcal{K}_n (\gamma\mathcal{G} + \mathcal{K}_n^T \mathcal{P}_n \mathcal{K}_n)^{-1} \mathcal{K}_n^T \mathcal{P}_n = \boldsymbol{I}. \tag{44}$$

The computation for solving this matrix equation can be implemented by using the optimization toolbox in Matlab. For $\boldsymbol{\phi}$, we have the following results [15], [40].

Lemma 3: The feature mapping $\boldsymbol{\phi}$ is a compact mapping, i.e. $\boldsymbol{\phi}$ maps any bounded set onto a relatively compact set in H_K.

By the properties of relatively compact set [41], it can be concluded that there exists a finite open coverage for the range of $\boldsymbol{\phi}$ in H_K, which implies that for a given degree of accuracy, the range can be approximated by the linear combination of the vectors in H_K^s. Regarding the selection of basis vectors $B_S = \{\boldsymbol{\phi}(\boldsymbol{u}_1), \boldsymbol{\phi}(\boldsymbol{u}_2), \ldots, \boldsymbol{\phi}(\boldsymbol{u}_M)\}$, the approximate linear dependence (ALD) method is utilized here [15], [17]. Suppose at time n we have already identified $\boldsymbol{w}(n) = \sum_{i=1}^{M} \alpha_i(n)\boldsymbol{\phi}(\boldsymbol{u}_i)$. For the incoming data $(\boldsymbol{x}(n+1), y(n+1))$, let

$$\zeta(n) = \min_c || \sum\nolimits_{i=1}^{M} c(i)\boldsymbol{\phi}(\boldsymbol{u}_i) - \boldsymbol{\phi}(\boldsymbol{x}(n+1))||^2, \tag{45}$$

and $\zeta(n)$ can be easily expanded as $\zeta(n) = K(\boldsymbol{x}(n+1), \boldsymbol{x}(n+1)) - \boldsymbol{K}_M(\boldsymbol{x}(n+1))^T \mathcal{K}_M^{-1} \boldsymbol{K}_M(\boldsymbol{x}(n+1))$, where $\boldsymbol{K}_M(\boldsymbol{x}(n+1)) = (K(\boldsymbol{u}_1, \boldsymbol{x}(n+1)), \ldots, K(\boldsymbol{u}_M, \boldsymbol{x}(n+1)))^T$ and $\mathcal{K}_M^{-1} = [K(\boldsymbol{u}_i, \boldsymbol{u}_j)]_{1\le i,j\le M}^{-1}$.

If $\zeta(n)$ is large, $\boldsymbol{\phi}(\boldsymbol{x}(n+1))$ cannot be linearly represented very well. The approximation ability of the space spanned by $\{\boldsymbol{\phi}(\boldsymbol{u}_1), \boldsymbol{\phi}(\boldsymbol{u}_2), \ldots, \boldsymbol{\phi}(\boldsymbol{u}_M)\}$ is weaker than the space spanned by $\{\boldsymbol{\phi}(\boldsymbol{u}_1), \boldsymbol{\phi}(\boldsymbol{u}_2), \ldots, \boldsymbol{\phi}(\boldsymbol{u}_M)\} \cup \{\boldsymbol{\phi}(\boldsymbol{x}(n+1))\}$, which means the model obtained from the former space may be inadequate for the learning. If $\zeta(n)$ is very small, there is negligible difference between the spaces, and $\boldsymbol{\phi}(\boldsymbol{x}(n+1))$ can be redundant to be a basis vector. A predetermined constant ν can be chosen as a threshold for the update of B_S. The selection strategy is proposed as follows. If $\zeta(n) < \nu$, $\boldsymbol{\phi}(\boldsymbol{x}(n+1))$, B_S remains unchanged. If $\zeta(n) \ge \nu$, $\boldsymbol{x}(n+1)$ is added into B_S. Moreover, ν determines the number of the basis vectors and the sparsity of H_K^s. A small ν leads to a complicated model, in contrast, a relatively big ν may bring a parsimonious one, but with less approximation ability.

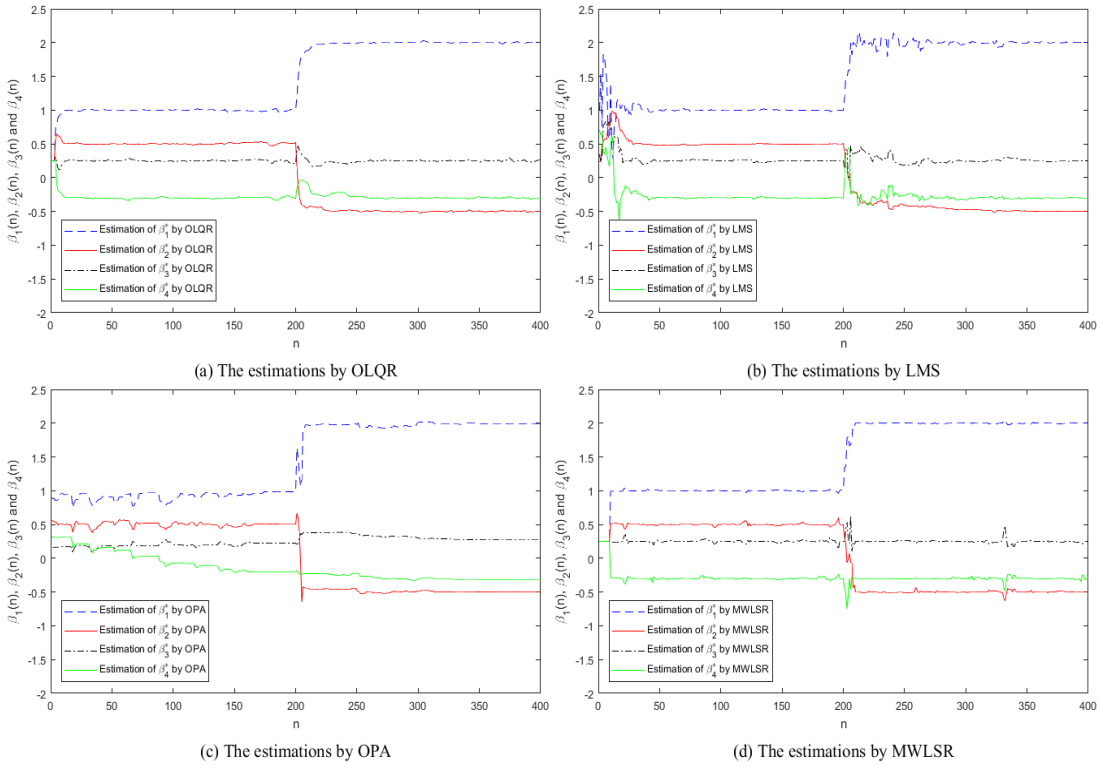
We propose that if $\boldsymbol{x}(n+1)$ is added into B_S, $\boldsymbol{w}(n) = \sum_{i=1}^{M} \alpha_i(n)\boldsymbol{\phi}(\boldsymbol{u}_i)$ is rewritten as $\boldsymbol{w}(n) = \sum_{i=1}^{M} \alpha_i(n)\boldsymbol{\phi}(\boldsymbol{u}_i) + 0\boldsymbol{\phi}(\boldsymbol{u}_{M+1})$, where $\boldsymbol{u}_{M+1} = \boldsymbol{x}(n+1)$, and $\mathcal{K}_n$ is updated as

$$\begin{bmatrix} K(\boldsymbol{u}_1, \boldsymbol{x}(n)) & \cdots & K(\boldsymbol{u}_{M+1}, \boldsymbol{x}(n)) \\ K(\boldsymbol{u}_1, \boldsymbol{x}(n-1)) & \cdots & K(\boldsymbol{u}_{M+1}, \boldsymbol{x}(n-1)) \\ \vdots & \cdots & \vdots \\ K(\boldsymbol{u}_1, \boldsymbol{x}(n-M+1)) & \cdots & K(\boldsymbol{u}_{M+1}, \boldsymbol{x}(n-M+1)) \end{bmatrix}. \tag{46}$$

$\boldsymbol{E}(n+1)$ and $\boldsymbol{E}(n)$ are expanded as $(\hat{e}(n), \hat{e}(n-1), \ldots, \hat{e}(n-M))^T$ and $(e(n), e(n-1), \ldots, e(n-M))^T$, respectively. $\boldsymbol{\alpha}(n)$ is updated as $(\alpha_1(n), \ldots, \alpha_M(n), 0)$. Then, the learning can be conducted by (43) and (44) accordingly. Our algorithm is summarized as online kernel linear quadratic regulator algorithm (OKLQR) and presented as follows

1) Initialization: the number of variables M, the kernel bandwidth σ, the threshold value ν, the basis vector set $B_S = \emptyset$. Let the system run for at least M steps. For $n = M$, $\boldsymbol{\beta}(n) = 0$.
2) Suppose at time n, $n \ge M$, $B_S = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_M\}$.
   If $\zeta(n) \le \nu$, B_S remains unchanged, and $\boldsymbol{\alpha}(n)$ is updated as $\boldsymbol{\alpha}(n+1)$ according to (43) and (44).
   Else, $\zeta(n) > \nu$, $B_S = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_M, \boldsymbol{\mu}_{M+1}\}$, $\boldsymbol{\alpha}(n)$ is updated as $\boldsymbol{\alpha}(n+1)$ by solving (42) based on (46), (43) and (44).
3) If data flow ends, stop; otherwise n=n+1, go to step 2.

Remark 4: Based on the theory of LQR, the updates of $\boldsymbol{\beta}(n)$ and $\boldsymbol{w}(n)$ are more likely to be restricted by a large γ [38], [39]. This may increase the robustness of the method but also make the learning model adapt the new dynamics at a relatively low speed. On the contrary, a small γ brings relatively fast learning and makes the model trance the changes of the data more efficiently due to the fewer restrictions on the update size, but may also result in the over-fitting problem. Therefore, the second terms in (25) and (42) can be treated as regularization for the learning. By borrowing the concepts of "passive" and "aggressive" that defined in [20], [23], the method can be described to be more passive with a



(a) The estimations by OLQR (b) The estimations by LMS

(c) The estimations by OPA (d) The estimations by MWLSR

FIGURE 1. The estimations by OLQR, LMS, OPA and MWLS for system (47).

larger γ, but more aggressive with a smaller γ. Therefore, a moderate γ is apt to be chosen for most of the learning tasks. We also remark that in gradient based method, the learning parameters such as learning rate should be carefully chosen to avoid divergence. Unlike the gradient based methods, γ is the only parameter that needs to be adjusted in OLQR and OKLQR. More importantly, no matter what γ is finally selected, the methods are still exponential convergence and stable as long as $\gamma > 0$. This is the main advantage of our method. Also, the new method can also be extended for the leaning problems with polynomial kernels.

Remark 5: As discussed in section 4 and 5, in the gradient based learning methods, $\boldsymbol{\beta}(n)$ and $\boldsymbol{w}(n)$ are supposed to be updated on feature plane (Euclid plane for linear model, hyperplane for kernel model) by decreasing the quadratic risk index. In the best cases, we wish that $\boldsymbol{\beta}(n)$ and $\boldsymbol{w}(n)$ are updated in the ''perfectly correct'' direction, i.e. $\boldsymbol{\beta}(n)$ and $\boldsymbol{w}(n)$ directly converge to $\boldsymbol{\beta}^\star$ and $\boldsymbol{w}^\star$. However, ''perfectly correct'' cannot be achieved since the feature plane will be no longer smooth due to the noise effects. This situation can be even worse if the noise disturbances are complex. The derivative in the gradient algorithms may indicate incorrect information for the distance between $\boldsymbol{\beta}(n)$ and $\boldsymbol{\beta}^\star$, and provide a wrong update direction. In our method, the learning problem is solved through a completely different approach with optimal control techniques. By employing the control input obtained by LQR, the prediction errors and distance between $\boldsymbol{\beta}(n)$ and $\boldsymbol{\beta}^\star$ will always be exponentially convergent to the neighborhood of origin point regardless of the characteristics of the noise effects. It means that despite ''perfectly correct'' cannot be achieved due to the noise effects, $\boldsymbol{\beta}(n)$ and $\boldsymbol{w}(n)$ are still updated in a ''generally correct'' direction. Therefore, the new method is able to provide more robust modeling performance on both convergence speed and prediction accuracy in the case of complex noise disturbances. This is another main advantage of our method.

VII. NUMERICAL EXAMPLES

This section provides numerical results obtained by our method using simulation data and realistic data. We also compare our results with those from the existing benchmark online methods.

A. ONLINE REGRESSION ANALYSIS

Consider following discrete, adaline system

$$\begin{cases} y(n) = z(n-1) + 0.5z(n-2) + 0.25z(n-1)z(n-2) \\ \qquad - 0.3z^3(n-1) + \varepsilon(n), \quad 1 \le n \le 200, \\ y(n) = 2z(n-1) - 0.5z(n-2) + 0.25z(n-1)z(n-2) \\ \qquad - 0.3z^3(n-1) + \varepsilon(n), \quad 201 \le n \le 400. \end{cases} \tag{47}$$

The input vector for this system is $(z(n-1), z(n-2), z(n-1)z(n-2), z(n-1)^3)$. Let $\mathcal{B}$ be the back-shift operator. The random term $\varepsilon(n) = (0.5 + \varsigma_3(n))(1 - 0.5\mathcal{B})^{-1}(\varsigma_1(n) + \varsigma_2(n))$, is temporal correlated and heterogenous, where $\varsigma_1(n)$

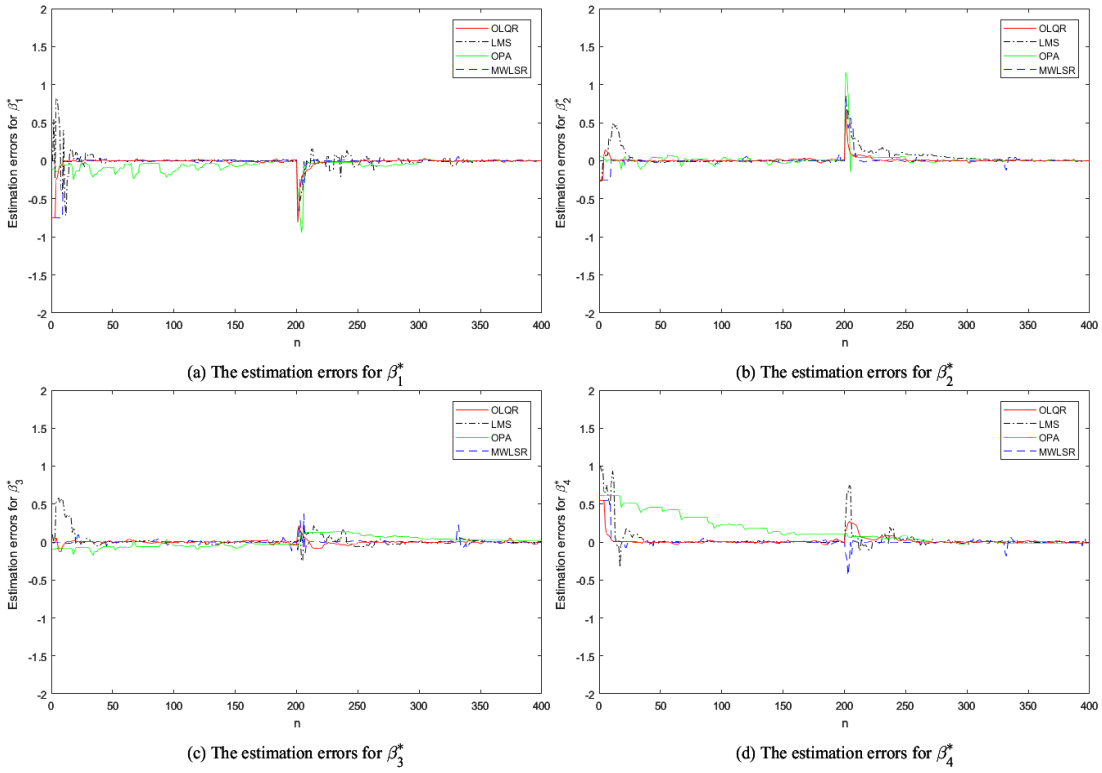


FIGURE 2. The estimation errors by different online learning methods for system (47).

is uniformly distributed on $[-0.5, 0.5]$, $\varsigma_2(n) \sim N(0, 1)$, and $\varsigma_3(t)$ is uniformly distributed on $[-1, 1]$. This system is perturbed by high intensity, temporal correlated, and non-stationary random disturbances, which means the learning is conducted in a complex noise environment. Learning results of four methods are presented in Fig. 1 and Fig. 2. The hyper-parameters of each method are chosen to give the optimal performance. In OLQR, γ is set as 1.5. The learning rate η is 0.2 in LMS. In OPA, υ is chosen to be 0.05, and the window length is 8 in WMLSR.

Consider following linear system with classical online learning setting

$$\begin{cases} y(n) = z_1(n) - z_2(n) + \varepsilon(n), & 1 \le n \le 200, \\ y(n) = -1.5z_1(n) + 2.5z_2(n) + \varepsilon(n), & 201 \le n \le 400. \end{cases} \tag{48}$$

For system (48), $z_1(n)$ is uniformly distributed on $[-1, 1]$, $z_2(n)$ is generated by $N(0, 1)$, $\varepsilon(n)$ is a Gaussian noise with variance of 0.05. Fig. 3 gives the online estimation results obtained by OLQR, least mean square algorithm (LMS) [6], online passive aggressive algorithm (OPA) [20] and least square regression with moving window algorithm (MWLSR) [26]. The estimation errors are compared in Fig. 4. In MWLSR, the window length is chosen to be 5. In OPA, the value of υ for hinge loss function is 0.01. The learning rate of LMS is 0.15. The regularization γ in OLQR is set to be 5. The instances and variables are set to be independent in (48). This also illustrates that although this paper focuses on the data streams with complex noise disturbances, the proposed method can also be well applied to the learning problems with the classical online learning setting.

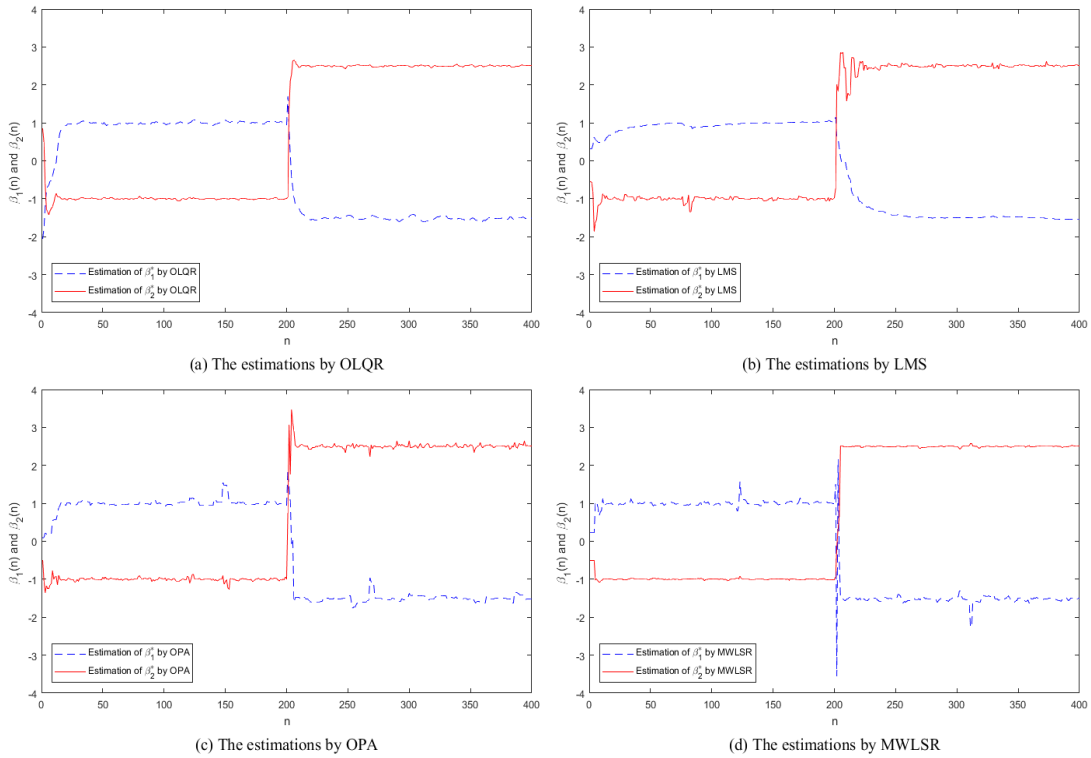
Although the system undergoes significant changes during the learning process, i.e. the objective parameter vector $(\beta_1^\star, \beta_2^\star, \beta_3^\star, \beta_4^\star)$ in (47) shifts from $(1, 0.5, 0.25, -0.3)$ to $(2, -0.5, 0.25, -0.3)$, and $(\beta_1^\star, \beta_2^\star)$ shifts from $(1, -1)$ to $(-1.5, 2.5)$ in (48), the changes of the systems are immediately detected and traced, and the objective parameters are online estimated very well. The estimations by OLQR converge quickly to the true values as shown in figures, suggesting that the proposed new method provides better performance in terms of both learning accuracy and convergence rate for this test system.

B. NONLINEAR SYSTEM IDENTIFICATION

In this example, we compare the proposed method with some benchmark algorithms based on their performances on nonlinear system identification. The nonlinear system with heterogenous noise term is given as follows

$$y(n) = 4\sin(z(n)) + 2\cos(z(n)) + \varepsilon(n), \quad 1 \le n \le 1000, \tag{49}$$

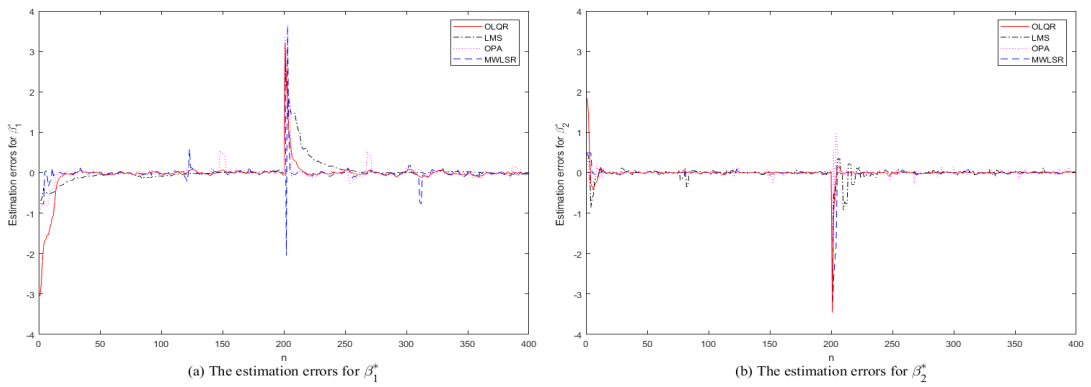
where $z(n) \sim N(0, 2)$. This system is perturbed by white noise or heterogenous noise. For the former case $\varepsilon(n) = 0.05\varsigma_1(n)$, where $\varsigma_1(n) \sim N(0, 1)$. Two series of datasets are generated. For the latter case



(a) The estimations by OLQR (b) The estimations by LMS

(c) The estimations by OPA (d) The estimations by MWLSR

FIGURE 3. The estimations by OLQR, LMS, OPA and MWLS for system (48).



(a) The estimation errors for β_1^* (b) The estimation errors for β_2^*

FIGURE 4. The estimation errors by different online learning methods for system (48).

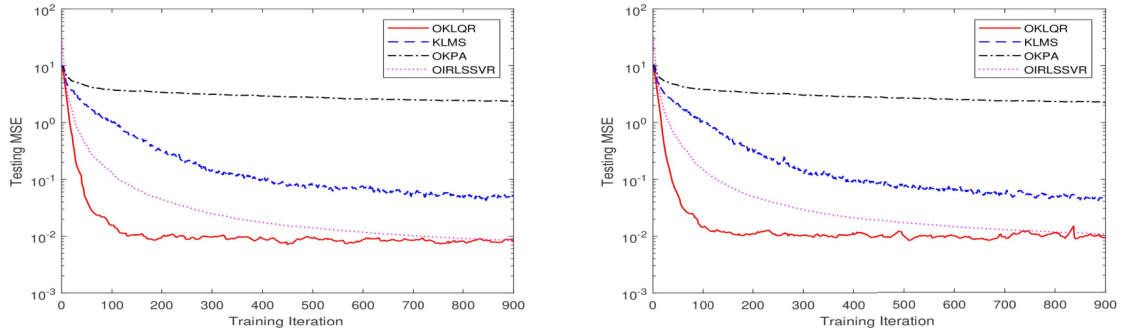
$\varepsilon(n) = 0.1\varsigma_1(n)\varsigma_2(n)$, where $\varsigma_2(n)$ is uniformly distributed on interval [0, 1]. In both cases, the data at the first 900 points are chosen as the training data and the subsequent 100 ones are used for evaluation. The testing mean square error (Testing MSE) is defined as

$$MSE(n) = \frac{1}{100}\sum_{j=901}^{1000}\left(\hat{y_n}(j) - y(j)\right)^2,$$

where $n = 1, \ldots, 900$, and $\hat{y_n}(j)$ is the prediction output of the learning model for $y(j)$ at time slot n.

The comparison results are shown by comparing the proposed algorithm with some existing online kernel learning algorithms including the kernel least mean square algorithm (KLMS) [6], online kernel passive aggressive algorithm (OKPA) [20], and online independent reduced least square support vector regression algorithm (OIRLSSVR) [13]. The Gaussian kernel is applied in this example and the optimal kernel bandwidth is chosen as 1 for all the algorithms. In both OKPA and OIRLSSVR, υ is chosen to be 0.001. The learning rate of KLMS is 0.9. The regularization parameter γ in OLQR is 1.

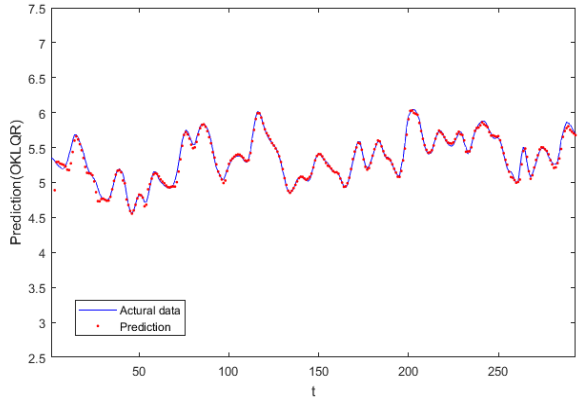
Fig. 5a and 5b demonstrate the testing MSE of the 900 training steps for homogeneous and heterogenous cases, respectively. The results are the average performance calculated from 50 independent Monte Carlo simulations. It can be



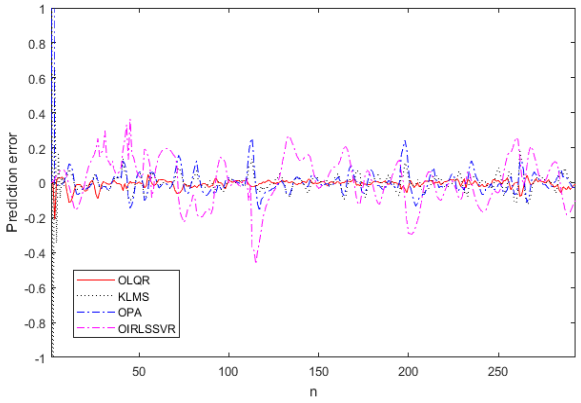
(a) The testing MSEs for system (50) with white noise.

(b) The testing MSEs for system (50) with heterogenous noise.

FIGURE 5. The testing MSEs for system (49) with white noise and heterogenous noise.



(a) The data of CO_2 concentration and predictions by OKLQR.



(b) The prediction errors by different online learning methods.

FIGURE 6. The prediction output by OKLQR and the prediction errors by different online learning methods.

seen that all MSEs obtained by different algorithms converge to a steady state after a number of learning iterations. OLQR outperforms the other algorithms for its smaller testing error and fast convergence rate in both cases.

C. NONLINEAR TIME SERIES ANALYSIS

In this section, a widely used benchmark system [43], is considered to examine the efficiency of our new method. The system is given by

$$\begin{cases} y(t) \\ = \dfrac{y(t-1)y(t-2)y(t-3)(y(t-3)-1)u(t-1)+u(t)}{1+y(t-2)^2+y(t-3)^2} \\ \qquad +\varepsilon(t), \quad 4 \le t \le 1000, \\ y(t' \\ = \dfrac{y(t-1)y(t-2)y(t-3)(y(t-2)-2)u(t-1)+u(t)}{1+5y(t-3)^2} \\ \qquad +\varepsilon(t), \quad 1001 \le t. \end{cases} \tag{50}$$

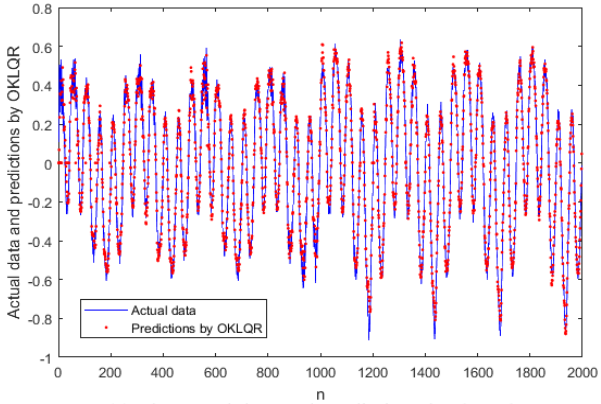
where $u(t) = 0.4\sin(\pi t/125) + 0.6\sin(\pi t/25)$. The initial values are $y(1) = y(2) = y(3) = 1$. The noise term $\varepsilon(t)$ is temporal correlated and governed by $\varepsilon(t) = 0.05(1 - 0.8\mathcal{B})^{-1}\varsigma(t)$, where $\varsigma(t) \sim N(0, 1)$. The input vector of the learning model consists of $u(t)$ and $y(t)$ as well as their delayed terms, i.e. $(y(t-1), y(t-2), y(t-3), u(t))$, and various kernel methods are applied for the learning. The system undergoes significant changes at point 1000 and a sequence of 2000 samples is generated.

The actual data of the system and prediction outputs by OKLQR with bandwidth $\sigma = 1$, penalty $\gamma = 10$ and ALD threshold parameter $\upsilon = 0.5$ are shown in Fig.7a. The comparison results with the prediction errors obtained by OKPA, KLMS, and OIRLSSVR are presented in Fig.7b. To achieve the optimal performance, the regularization parameter γ in OLQR is set to be 1, the learning rate of KLMS is 0.9, For both OKPA and OIRLSSVR, υ is 0.05.

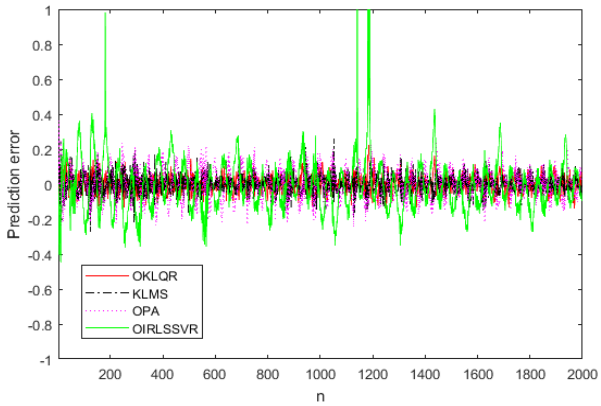
It shows that after a short period of transient effects, the system dynamics can be well predicted by OKLQR in one step ahead manner. In this example, the mean square error is defined as

$$MSE = \frac{1}{1960}\sum_{t=41}^{2000}\left(\hat{y}(t) - y(t)\right)^2,$$

by removing the initial transient response effects, where $\hat{y(t)}$ is the predicted output for $y(t)$. The average MSEs obtained by different algorithms calculated from 100 independent Monte Carlo simulations are shown in Table 1 to further



(a) The actual data and predictions by OKLQR



(b) The prediction errors by different online learning methods

FIGURE 7. The actural data, predictions by OKLQR and prediction errors by different learning methods.

TABLE 1. Learning results for nonlinear time series prediction.

Method	σ	γ	ν	η	ξ	MSE
OKLQR	1	2	0.05			0.0731
KLMS	1			1		0.0826
OKPA	1		0.05		0.01	0.1197
OIRLSSVR	1		0.05		0.05	0.1823

illustrate the advantages of our method. It can be seen that OKLQR can achieve better learning results with faster convergence and smaller prediction error than the other learning algorithms.

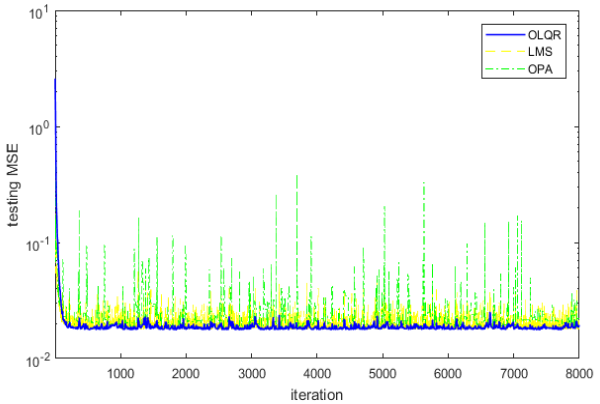
D. REALISTIC DATA: BOX-JENKINS GAS FURNACE PROBLEM

In this section, we demonstrate our method on a real world data set. The Box-Jenkins gas furnace problem is a common benchmark to test learning methods [43], [44]. The data consists of 296 pair of samples and are measured from a gas furnace with the CO_2 concentration $y(t)$ and the gas flow rate $u(t)$. The input vector is chosen to be $(y(t-1), y(t-2), y(t-3), u(t-1), u(t-2), u(t-3))$.

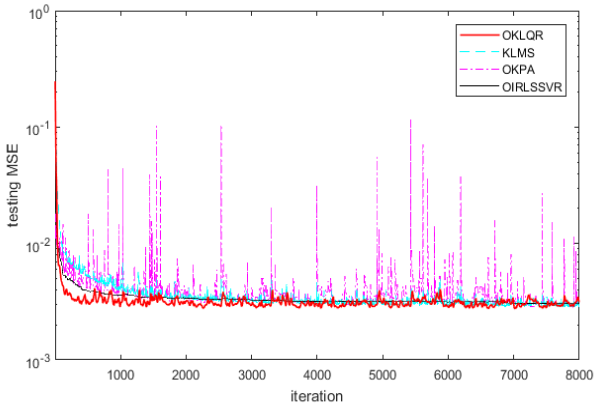
The prediction outputs by OKLQR with bandwidth $\sigma = 10$, penalty $\gamma = 0.2$ and ALD threshold parameter $\nu = 0.001$ are shown in Fig.6a. The comparison results with the prediction errors obtained by OKPA, KLMS, and

TABLE 2. Learning results for real world data prediction.

Method	σ	γ	ν	η	ξ	MSE
OKLQR	10	0.2	0.0001			0.0013
KLMS	10			1.5		0.0085
OKPA	10		0.0001		0.0001	0.0159
OIRLSSVR	10		0.0001		0.02	0.0771



(a) The prediction errors of linear models.



(b) The prediction errors of nonlinear models.

FIGURE 8. The estimation errors by different online learning methods.

OIRLSSVR are shown in Fig.6b. To achieve the optimal performance, the regularization parameter γ in OLQR is set to be 1. The learning rate of KLMS is set as 1.5. For both OKPA and OIRLSSVR, υ is 0.01. The MSE for this example is defined as

$$MSE = \frac{1}{293}\sum_{t=4}^{296}\left(\hat{y}(t) - y(t)\right)^2,$$

where $\hat{y(t)}$ is the predicted output for $y(t)$. The MSEs obtained by different algorithms are demonstrated in Table 2. It can be seen that the OKLQR can achieve satisfactory and better performance than the other learning methods in this example.

E. PRACTICAL APPLICATION: FULL LOAD ELECTRICAL POWER OUTPUT PREDICTION

In this application, we apply our new algorithms to a real-world online predicting problem. Predicting electrical power output is of great importance for the efficiency and

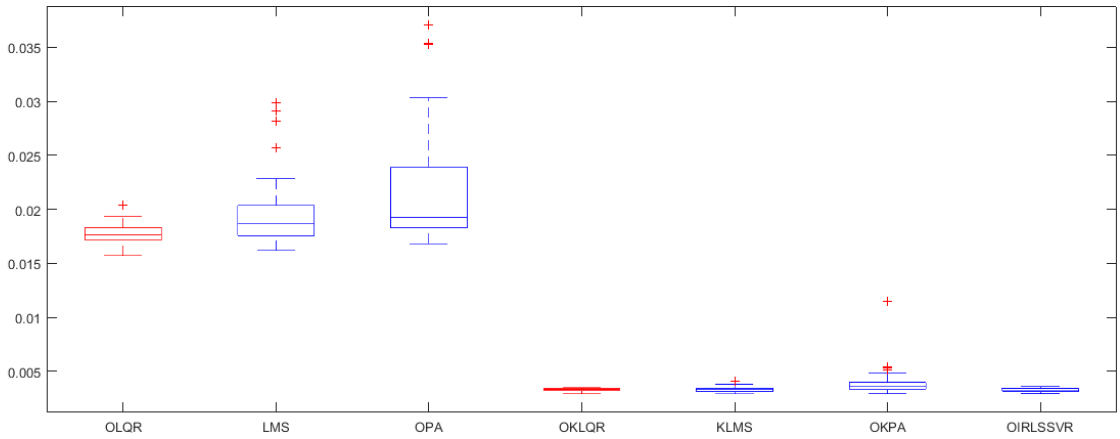


FIGURE 9. Evaluation of final testing MSEs in terms of boxplot by Monte Carlo simulations.

economic operation of a power plant. Electrical power output can be expressed as either linear or nonlinear function of environmental variables such as temperature and humidity recorded by sensors. This problem has been studied in many literature using other machine learning tools [45]–[47].

Consider a combined cycle power plant (CCPP) with two gas turbines, one steam turbine, and two heat recovery systems [45], [47], those types of equipment are sensitive to four main variables: ambient temperature (AT), atmospheric pressure (AP), relative humidity (RH) and exhaust steam pressure (or vacuum, V). Thus, the full load electrical power output (P_E) is affected by the above four variables. The dataset in [45] is utilized for modeling, where the four variables and one target variable are recorded by different sensors hourly over six years (2006-2011). Thus 9568 data points are included. The proposed algorithms (OLQR and OKLQR) and five benchmark online learning algorithms are employed to predict in the context of linear and kernel regression model. In the experiments, after shuffling the whole dataset, first 8000 samples are used as training set and the rest 1568 are for testing. For this example, the MSE at learning step n is defined as follows

$$MSE(n) = \frac{1}{1568}\sum_{j=8001}^{9568}\left(\hat{y_n}(j) - y(j)\right)^2,$$

where $\hat{y_n}(j)$ is the predicting output of jth test sample. To remove the transient effects, the prediction results obtained from the first 400 samples are abandoned and average MSE (AMSE) aiming at measuring the average predicting error is defined as

$$AMSE = \frac{1}{7600}\sum_{n=401}^{8000} MSE(n).$$

Using the linear model, the testing MSEs obtained by OLQR, LMS and OPA are shown in Fig. 8(a). We can observe that LMS and PA methods are seriously affected by chaotic noise perturbing, and OLQR show advantage on prediction accuracy with fewer peaks in the figure. Using kernel models, the testing MSEs obtained by OKLQR, KLMS, OKPA and OIRLSSVR are presented in Fig. 8(b). Our control-based methods also show advantages in robustness, convergence speed, and average predicting accuracy. The learning parameters of each algorithm are chosen for the best learning performance. Details of the hyperparameter setting and AMSE are given in Table 3.

TABLE 3. Learning results for real world data prediction.

Method	σ	γ	ν	η	ξ	AMSE
OLQR		1000				0.0186
LMS				0.2		0.0202
OPA					0.25	0.0236
OKLQR	1	1000	0.01			0.0031
KLMS	1			0.1		0.0034
OKPA	1				0.1	0.0039
OIRLSSVR	1		0.1		0.1	0.0033

To further illustrate our methods, the whole dataset is shuffled 50 times, the training and testing sets are constructed using the same strategy for 50 times. The Mento Carlo simulation results using different algorithms are shown in terms of boxplot in Fig. 9. OLQR and OKLQR algorithms can achieve better predicting accuracy with less fluctuation.

VIII. CONCLUSION AND DISCUSSION

In this paper, we have proposed a new learning method named the "online linear quadratic regulator" learning algorithm. By using a carefully designed scheme, the online learning problem is transformed into a state feedback control problem of a group of controllable, observable and time-varying systems. Two dynamical linear quadratic regulator based numerical algorithms are developed to give the model update law for the online learning of adaline models and reproducing kernel Hilbert space models. This method provides a novel online learning approach from the perspective of state feedback optimal control with solid theoretical basis. Compared with the existing online learning methods, the proposed method has better convergence rate and prediction accuracy for data streams with complex noise. Some key limitations

of the existing methods such as robustness to noise effects, restrictions of learning parameters et.al. can be therefore overcome. The effectiveness and efficiency of our theory are also demonstrated by the encouraging experimental results. Our future work will focus on the further extensions and potential applications of the novel optimal control based online learning algorithms.

APPENDIX A

From (17) and (18), we have $\boldsymbol{E}(n+1) = \mathcal{B}(n)\widetilde{\boldsymbol{\beta}(n+1)} - \boldsymbol{d}(n)$, and $\boldsymbol{E}(n) = \mathcal{B}(n)\widetilde{\boldsymbol{\beta}(n)} - \boldsymbol{d}(n)$. Substitute the above equation into (22),

$$\begin{aligned}\mathcal{B}(n)\widetilde{\boldsymbol{\beta}(n+1)} - \boldsymbol{d}(n) &= \mathcal{T}(n)(\mathcal{B}(n)\widetilde{\boldsymbol{\beta}(n)} - \boldsymbol{d}(n)) \\ &= \mathcal{T}(n)\mathcal{B}(n)\widetilde{\boldsymbol{\beta}(n)} - \mathcal{T}(n)\boldsymbol{d}(n). \quad (51)\end{aligned}$$

Let $\mathcal{H}(n) = \mathcal{B}(n)^{-1}\mathcal{T}(n)\mathcal{B}(n)$ and then

$$\begin{aligned}\widetilde{\boldsymbol{\beta}(n+1)} &= \mathcal{B}(n)^{-1}\mathcal{T}(n)\mathcal{B}(n)\widetilde{\boldsymbol{\beta}(n)} + \mathcal{B}(n)^{-1}(I - \mathcal{T}(n))\boldsymbol{d}(n) \\ &= \mathcal{H}(n)\widetilde{\boldsymbol{\beta}(n)} + \mathcal{B}(n)^{-1}(I - \mathcal{T}(n))\boldsymbol{d}(n). \quad (52)\end{aligned}$$

By iteration, we have

$$\begin{aligned}&\widetilde{\boldsymbol{\beta}(n+k)} \\ &= \prod_{i=0}^{k-1}\mathcal{H}(n+i)\widetilde{\boldsymbol{\beta}(n)} + \mathcal{B}(n)^{-1}(I - \mathcal{T}(n))\boldsymbol{d}(n) \\ &\quad + \sum_{j=1}^{k-1}\Big(\prod_{i=j}^{k-1}\mathcal{H}(n+i)\Big)\mathcal{B}(n+k-1-i)^{-1} \\ &\quad \times (I - \mathcal{T}(n+k-1-i))\boldsymbol{d}(n+k-1-i). \quad (53)\end{aligned}$$

If $\mathcal{H}(n)$ is contractive and positive, $\mathcal{H}(n) = \mathcal{B}(n)^{-1}\mathcal{T}(n)\mathcal{B}(n)$ is also positive and contractive. We can conclude from the assumptions that, for any i, $\mathcal{B}(n+i)$ is bounded and invertible, and $I - \mathcal{T}(n+i)$ is positive and contractive, which implies $\mathcal{B}(n+i)^{-1}$ and $I - \mathcal{T}(n+i)$ are bounded linear operators [48]. Thus, for the Euclid norm of $\widetilde{\boldsymbol{\beta}(n+k)}$, there exist positive constants $C_h < 1$ and $C_\star < 1$, such that

$$\begin{aligned}&||\widetilde{\boldsymbol{\beta}(n+k)}|| \\ &\leq \prod_{i=0}^{k}\mathcal{H}(n+i)\widetilde{\boldsymbol{\beta}(n)}|| + ||\mathcal{B}(n)^{-1}(I - \mathcal{T}(n))\boldsymbol{d}(n)|| \\ &\quad + ||\sum_{j=1}^{k-1}\Big(\prod_{i=0}^{j}\mathcal{H}(n+i)\Big)\mathcal{B}(n+i-j)^{-1} \\ &\quad (I - \mathcal{T}(n+i-j))\boldsymbol{d}(n+j-1)|| \\ &\leq C_h^k||\widetilde{\boldsymbol{\beta}(n)}|| + \sum_{j=0}^{k-1}C_\star^j D \\ &= C_h^k||\widetilde{\boldsymbol{\beta}(n)}|| + \frac{1 - C_\star^k}{1 - C_\star}D. \quad (54)\end{aligned}$$

Let $D_e = \frac{C_\star}{1-C_\star}\rho D$ and $k \to \infty$, the theorem is proved. ■

APPENDIX B

For given n, V is quadratic i.e., there exists a positive definite and systematical matrix $\mathcal{P}_n$ such that $V(\boldsymbol{E}(n)) = \boldsymbol{E}(n)^T\mathcal{P}_n\boldsymbol{E}(n)$. A Hamilton-Jacobi equation [39] is given as

$$\begin{aligned}V(\boldsymbol{E}(n)) = \min_{U_n(1)}(&\boldsymbol{E}_n(1)^T\mathcal{Q}\boldsymbol{E}_n(1) + \boldsymbol{U}_n(1)^T\mathcal{R}\boldsymbol{U}_n(1) \\ &+ V(\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n(1))). \quad (55)\end{aligned}$$

It follows

$$\begin{aligned}&V(\boldsymbol{E}(n)) \\ &= \min_{\boldsymbol{U}_n(1)}(\boldsymbol{E}_n(1)^T\mathcal{Q}\boldsymbol{E}_n(1) + \boldsymbol{U}_n(1)^T\mathcal{R}\boldsymbol{U}_n(1) \\ &\quad + (\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n(1))^T\mathcal{P}_n(\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n(1))). \quad (56)\end{aligned}$$

To minimize V, we set the partial derivative with respect to $\boldsymbol{U}_n(1)$ to zero,

$$2\boldsymbol{U}_n(1)^T\mathcal{R} + 2(\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n(1))^T\mathcal{P}_n\mathcal{B}_n = 0. \quad (57)$$

Then, the solution is given as

$$\boldsymbol{U}_n^\star(1) = -(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n\boldsymbol{E}_n(1). \quad (58)$$

Therefore, we have Hamilton-Jacobi equation as follows

$$\begin{aligned}&\boldsymbol{E}_n(1)^T\mathcal{P}\boldsymbol{E}_n(1) \\ &= \boldsymbol{E}_n(1)^T\mathcal{Q}\boldsymbol{E}_n(1) + \boldsymbol{U}_n^\star(1)^T\mathcal{R}\boldsymbol{U}_n^\star(1) \\ &\quad + (\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n^\star(1))^T\mathcal{P}_n(\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n^\star(1)), \quad (59)\end{aligned}$$

and

$$\begin{aligned}&\boldsymbol{E}_n(1)^T\mathcal{P}_n\boldsymbol{E}_n(1) \\ &= \boldsymbol{E}_n(1)^T\mathcal{Q}\boldsymbol{E}_n(1) + \boldsymbol{U}_n^\star(1)^T\mathcal{R}\boldsymbol{U}_n^\star(1) \\ &\quad + (\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n^\star(1))^T\mathcal{P}_n(\boldsymbol{E}_n(1) + \mathcal{B}_n\boldsymbol{U}_n^\star(1)) \\ &= \boldsymbol{E}_n(1)^T\mathcal{Q}\boldsymbol{E}_n(1) + \boldsymbol{E}_n(1)^T\mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{R} \\ &\quad \times (\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n\boldsymbol{E}_n(1) \\ &\quad + \boldsymbol{E}_n(1)^T\mathcal{P}_n\boldsymbol{E}_n(1) + \boldsymbol{E}_n(1)^T\mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1} \\ &\quad \times \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n\boldsymbol{E}_n(1) \\ &\quad - 2\boldsymbol{E}_n(1)^T\mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n\boldsymbol{E}_n(1) \\ &= E_n(1)^T\mathcal{Q}\boldsymbol{E}_n(1) + \boldsymbol{E}_n(1)^T\mathcal{P}_n\boldsymbol{E}_n(1) \\ &\quad - \boldsymbol{E}_n(1)^T\mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n\boldsymbol{E}_n(1). \quad (60)\end{aligned}$$

Since this must hold for all $\boldsymbol{E}_n(1)$, we have following algebraic Riccati equation [37]

$$\mathcal{P}_n = \mathcal{Q} + \mathcal{P}_n - \mathcal{P}_n\mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n. \quad (61)$$

The update law of the online learning is given by the optimal input law $\boldsymbol{U}_n(1) = \mathcal{F}_n\boldsymbol{E}_n(1) = \mathcal{F}_n\boldsymbol{E}(n)$, where $\mathcal{F}_n = -(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n$. Let $\mathcal{Q} = I$ and $\mathcal{R} = \gamma I$. We have

$$\mathcal{P}_n = I + \mathcal{P}_n - \mathcal{P}_n\mathcal{B}_n(\gamma I + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n. \quad (62)$$

Thus, the algebraic Riccati equation for $\mathcal{P}_n$ is simplified as

$$\mathcal{P}_n\mathcal{B}_n(\gamma I + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n = I. \quad (63)$$

Let $\mathcal{A}_n^\star$ denote the transit matrix of closed loop system. With the optimal control input $\boldsymbol{U}_n(1) = -(\gamma I + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n\boldsymbol{E}_n(1)$, we have

$$\begin{aligned}
\boldsymbol{E}(n+1) &= \mathcal{A}_n^\star\boldsymbol{E}(n)\\
&= (I + \mathcal{B}_n\mathcal{F}_n)\boldsymbol{E}(n)\\
&= (I - \mathcal{B}_n(\mathcal{R} + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n)\boldsymbol{E}(n)\\
&= (I - \mathcal{B}_n(\gamma I + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n)\boldsymbol{E}(n)\\
&= (I - (\gamma(\mathcal{B}_n\mathcal{B}_n^T\mathcal{P}_n)^{-1} + I)^{-1})\boldsymbol{E}(n)\\
&= \gamma(\mathcal{B}_n\mathcal{B}_n^T\mathcal{P}_n)^{-1}(\gamma(\mathcal{B}_n\mathcal{B}_n^T\mathcal{P}_n)^{-1} + I)^{-1}\boldsymbol{E}(n)\\
\boldsymbol{E}(n+1) &= \mathcal{A}_n^\star\boldsymbol{E}(n)\\
&= (I - \mathcal{B}_n(R + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n)\boldsymbol{E}(n)\\
&= (I - \mathcal{B}_n(\gamma I + \mathcal{B}_n^T\mathcal{P}_n\mathcal{B}_n)^{-1}\mathcal{B}_n^T\mathcal{P}_n)\boldsymbol{E}(n)\\
&= (I - \mathcal{P}_n^{-1})\boldsymbol{E}(n). \qquad (64)
\end{aligned}$$

It is noted that $\gamma(\mathcal{B}_n\mathcal{B}_n^T\mathcal{P}_n)^{-1}$ and $\mathcal{P}_n^{-1}$ are positive definite, therefore $\gamma(\mathcal{B}_n\mathcal{B}_n^T P_n)^{-1}(\gamma(\mathcal{B}_n\mathcal{B}_n^T\mathcal{P}_n)^{-1} + I)^{-1} \succ 0$ is also positive definite and $I - \mathcal{P}_n^{-1} \prec I$. It can be obtained that $0 \prec \mathcal{A}_n^\star \prec I$, and $\mathcal{A}_n^\star$ is contractive. Denote $\widetilde{\boldsymbol{\beta}(n)} = (\beta_1(n) - \beta_1^\star, \beta_2(n) - \beta_2^\star, \ldots, \beta_M(n) - \beta_M^\star)^T$. Noticed that $\boldsymbol{E}(n+1) = \mathcal{B}_n\widetilde{\boldsymbol{\beta}(n+1)} - \boldsymbol{d}(n)$ and $\boldsymbol{E}(n) = \mathcal{B}_n\widetilde{\boldsymbol{\beta}(n)} - \boldsymbol{d}(n)$, we have

$$\mathcal{B}_n\widetilde{\boldsymbol{\beta}(n+1)} - \boldsymbol{d}(n) = \mathcal{A}_n^\star(\mathcal{B}_n\widetilde{\boldsymbol{\beta}(n)} - \boldsymbol{d}(n)). \qquad (65)$$

It follows

$$\widetilde{\boldsymbol{\beta}(n+1)} = \mathcal{B}_n^{-1}\mathcal{A}_n^\star\mathcal{B}_n\widetilde{\boldsymbol{\beta}(n)} + \mathcal{B}_n^{-1}(I - \mathcal{A}_n^\star)\boldsymbol{d}(n). \qquad (66)$$

It is obvious that the eigenvalues of $\mathcal{B}_n^{-1}\mathcal{A}_n^\star\mathcal{B}_n$ and $\mathcal{A}_n^\star$ are the same, which implies that $\mathcal{B}_n^{-1}\mathcal{A}_n^\star\mathcal{B}_n$ is also contractive. By the same techniques presented in (52), (53) and (54), the theorem can be completed. ■

REFERENCES

[1] T. Anderson, *The Theory and Practice of Online Learning*. Athabasca, AB, Canada: Athabasca Univ. Press, 2008.

[2] J. Lu, S. C. H. Hoi, J. Wang, P. Zhao, and Z.-Y. Liu, "Large scale online kernel learning," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1613–1655, Jan. 2016.

[3] S. Yin, X. Li, H. Gao, and O. Kaynak, "Data-based techniques focused on modern industry: An overview," *IEEE Trans. Ind. Electron.*, vol. 62, no. 1, pp. 657–667, Jan. 2015.

[4] H. Ning, G. Qing, and X. Jing, "Identification of nonlinear spatiotemporal dynamical systems with nonuniform observations using reproducing-kernel-based integral least square regulation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2399–2412, Nov. 2016.

[5] S. Haykin and B. Widrow, *Least-Mean-Square Adaptive Filters*, vol. 31. Hoboken, NJ, USA: Wiley, 2003.

[6] W. Liu, P. P. Pokharel, and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 2, pp. 543–554, Feb. 2008.

[7] S. Shalev-Shwartz and Y. Singer, "Online learning: Theory, algorithms, and applications," Hebrew Univ., Jerusalem, Israel, Tech. Rep. 7, 2007.

[8] W. Liu, J. C. Principe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*, vol. 57. Hoboken, NJ, USA: Wiley, 2011.

[9] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[10] S. Smale and Y. Yao, "Online learning algorithms," *Found. Comput. Math.*, vol. 6, no. 2, pp. 145–170, Apr. 2006.

[11] B. Chen, S. Zhao, P. Zhu, and J. C. Príncipe, "Quantized kernel recursive least squares algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 9, pp. 1484–1491, Sep. 2013.

[12] Y.-J. Lee and S.-Y. Huang, "Reduced support vector machines: A statistical theory," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 1–13, Jan. 2007.

[13] Y.-P. Zhao, J.-G. Sun, Z.-H. Du, Z.-A. Zhang, and Y.-B. Li, "Online independent reduced least squares support vector regression," *Inf. Sci.*, vol. 201, pp. 37–52, Oct. 2012.

[14] B. J. de Kruif and T. J. A. de Vries, "Pruning error minimization in least squares support vector machines," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 696–702, May 2003.

[15] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[16] H. Ning, X. Jing, and L. Cheng, "Online identification of nonlinear spatiotemporal systems using kernel learning approach," *IEEE Trans. Neural Netw.*, vol. 22, no. 9, pp. 1381–1394, Sep. 2011.

[17] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1058–1067, Mar. 2009.

[18] E. Kayacan, O. Cigdem, and O. Kaynak, "Sliding mode control approach for online learning as applied to type-2 fuzzy neural networks and its experimental evaluation," *IEEE Trans. Ind. Electron.*, vol. 59, no. 9, pp. 3510–3520, Sep. 2012.

[19] R. Razavi-Far, E. Hallaji, M. Saif, and G. Ditzler, "A novelty detector and extreme verification latency model for nonstationary environments," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 561–570, Jan. 2019.

[20] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 551–585, Dec. 2006.

[21] C. Liu, S. C. Hoi, P. Zhao, J. Sun, and E.-P. Lim, "Online adaptive passive-aggressive methods for non-negative matrix factorization and its applications," in *Proc. ACM 25th Int. Conf. Inf. Knowl. Manage.*, 2016, pp. 1161–1170.

[22] J. Jorge and R. Paredes, "Passive-Aggressive online learning with nonlinear embeddings," *Pattern Recognit.*, vol. 79, pp. 162–171, Jul. 2018.

[23] J. Lu, D. Sahoo, P. Zhao, and S. C. Hoi, "Sparse passive-aggressive learning for bounded online kernel methods," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 4, 2018, Art. no. 45.

[24] Z. Wang and S. Vucetic, "Online passive-aggressive algorithms on a budget," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 908–915.

[25] J. Lu, P. Zhao, and S. C. H. Hoi, "Online sparse passive aggressive learning with kernels," in *Proc. SIAM Int. Conf. Data Mining*, 2016, pp. 675–683.

[26] H.-S. Tang, S.-T. Xue, R. Chen, and T. Sato, "Online weighted LS-SVM for hysteretic structural system identification," *Eng. Struct.*, vol. 28, no. 12, pp. 1728–1735, 2006.

[27] Q. Wu, Y. Ying, and D.-X. Zhou, "Learning rates of least-square regularized regression," *Found. Comput. Math.*, vol. 6, no. 2, pp. 171–192, 2006.

[28] X. Jing and L. Cheng, "An optimal PID control algorithm for training feedforward neural networks," *IEEE Trans. Ind. Electron.*, vol. 60, no. 6, pp. 2273–2283, Jun. 2013.

[29] X. Jing, "An H_∞ control approach to robust learning of feedforward neural networks," *Neural Netw.*, vol. 24, no. 7, pp. 759–766, 2011.

[30] X. Jing, "Robust adaptive learning of feedforward neural networks via LMI optimizations," *Neural Netw.*, vol. 31, no. 1, pp. 33–45, 2012.

[31] K. D. Brabanter, J. D. Brabanter, J. A. K. Suykens, and B. L. R. D. Moor, "Kernel regression in the presence of correlated errors," *J. Mach. Learn. Res.*, vol. 12, pp. 1955–1976, Jun. 2011.

[32] M. Espinoza, J. A. K. Suykens, and B. De Moor, "LS-SVM regression with autocorrelated errors," in *Proc. 14th IFAC Symp. Syst. Identificat. (SYSID)*, 2006, pp. 582–587.

[33] A. C. Cameron and P. K. Trivedi, *Microeconometrics: Methods and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[34] L. D. Berkovitz, *Optimal Control Theory*, vol. 12. New York, NY, USA: Springer, 2013.

[35] J. Y. Campbell, A. W. Lo, and A. C. MacKinlay, *The Econometrics of Financial Markets*, vol. 2. Princeton, NJ, USA: Princeton Univ. Press, 1997.

[36] P. O. M. Scokaert and J. B. Rawlings, "Constrained linear quadratic regulation," *IEEE Trans. Autom. Control*, vol. 43, no. 8, pp. 1163–1169, Aug. 1998.

[37] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*. Hoboken, NJ, USA: Wiley, 2012.

[38] J. Yu, P. Shi, W. Dong, and H. Yu, "Observer and command-filter-based adaptive fuzzy output feedback control of uncertain nonlinear systems," *IEEE Trans. Ind. Electron.*, vol. 62, no. 9, pp. 5962–5970, Sep. 2015.

[39] B. D. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*. North Chelmsford, MA, USA: Courier, 2007.
[40] H. Ning, G. Qing, T. Tian, and X. Jing, "Online identification of nonlinear stochastic spatiotemporal system with multiplicative noise by robust optimal control-based kernel learning method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 389–404, Feb. 2019.
[41] J. T. Oden and L. Demkowicz, *Applied Functional Analysis*. London, U.K.: Chapman & Hall, 2017.
[42] H.-L. Wei and S. A. Billings, "Model structure selection using an integrated forward orthogonal search algorithm assisted by squared correlation and mutual information," *Int. J. Model., Identificat. Control*, vol. 3, no. 4, pp. 341–356, 2008.
[43] Y. Liu, H. Wang, J. Yu, and P. Li, "Selective recursive kernel learning for online identification of nonlinear systems with NARX form," *J. Process Control*, vol. 20, no. 2, pp. 181–194, 2010.
[44] D. Chen, J. Wang, F. Zou, W. Yuan, and W. Hou, "Time series prediction with improved neuro-endocrine model," *Neural Comput. Appl.*, vol. 24, no. 6, pp. 1465–1475, 2014.
[45] P. Tüfekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *Int. J. Elect. Power Energy Syst.*, vol. 60, pp. 126–140, Sep. 2014.
[46] G. K. F. Tso and K. K. W. Yau, "Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks," *Energy*, vol. 32, no. 9, pp. 1761–1768, Sep. 2007.
[47] J. Che, J. Wang, and G. Wang, "An adaptive fuzzy combination model based on self-organizing map and support vector regression for electric load forecasting," *Energy*, vol. 37, no. 1, pp. 657–664, 2012.
[48] E. D. Kirk, *Optimal Control Theory: An Introduction*. North Chelmsford, MA, USA: Courier, 2012.

HANWEN NING received the B.S. degree in applied mathematics and the Ph.D. degree in mathematical statistics from the Huazhong University of Science and Technology, Wuhan, China, in 2005 and 2010, respectively. He was a Research Associate with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong. He has been an Associate Professor with the Department of Statistics, Zhongnan University of Economics and Law, Wuhan, since 2011. His current research interests include nonlinear system identification, machine learning, and analysis of partial differential equations.

JIAMING ZHANG received the B.S. degree in applied statistics from the Zhongnan University of Economics and Law, Wuhan, China, in 2016, where she is currently pursuing the Ph.D. degree in applied statistics. Her current research interests include machine learning, deep learning, and reinforcement learning.

XINGJIAN JING (M'13–SM'17) received the B.S. degree from Zhejiang University, China, in 1998, the M.S. and Ph.D. degrees in robotics from the Shenyang Institute of Automation, Chinese Academy of Sciences, in 2001 and 2005, respectively, and the Ph.D. degree in nonlinear systems and signal processing from the University of Sheffield, U.K., in 2008. He is currently an Associate Professor with the Department of Mechanical Engineering, The Hong Kong Polytechnic University (PolyU), since July 2015. Before joining in PolyU as an Assistant Professor, in November 2009, he was a Research Fellow with the Institute of Sound and Vibration Research, University of Southampton. He has published more than 100 referred papers and holds about 10 patents filed in China and USA. His current research interests include nonlinear dynamics, vibration, and control. He was a recipient of a series of academic and professional awards, including the 2016 IEEE SMC Andrew P. Sage Best Transactions Paper Award, 2017 TechConnect World Innovation Award, USA, 2017 EASD Senior Research Prize, Europe, and 2017 HK Construction Industry Council Innovation Award (the First Prize). He currently serves as a Technical Editor of IEEE/ASME TRANSACTION ON MECHATRONICS and an Associate Editor of *Mechanical Systems and Signal Processing*.

TIANHAI TIAN received the B.S. and M.S. degrees in computational mathematics from the Huazhong University of Science and Technology, Wuhan, China, in 1982 and 1988, respectively, and the Ph.D. degree in mathematics from the University of Queensland, Brisbane, QLD, Australia, in 2001. He was an Australian Research Council (ARC) Fellow with the University of Queensland, the Lord Kelvin Fellow with the University of Glasgow, Glasgow, U.K., and the ARC Future Fellow with Monash University, Melbourne, Australia, where he is currently an Associate Professor. His current research interests include stochastic modeling of molecular regulatory networks, simulation of stochastic models, and reverse engineering and parameter inference for complex networks in biology, finance, and social science. He received various research grants from ARC, the U.K. Biotechnology and Biological Science Research Council, the National Natural Science Foundation of China, the Royal Society, and the Australia Academy of Science.