Structure Parameter Optimized Kernel Based Online Prediction with a Generalized Optimization Strategy for Nonstationary Time Series

Jinhua Guo, *Student Member, IEEE*, Hao Chen, Jingxin Zhang, Sheng Chen, *Fellow, IEEE*

***Abstract*—In this paper, sparsification techniques aided online prediction algorithms in a reproducing kernel Hilbert space are studied for nonstationary time series. The online prediction algorithms as usual consist of the selection of kernel structure parameters and the kernel weight vector updating. For structure parameters, the kernel dictionary is selected by sparsification techniques with selective online modeling criteria, and the symmetric kernel covariance matrix is intermittently optimized with the covariance matrix adaptation evolution strategy (CMA-ES). This intermittent optimization can not only improve the kernel structure's flexibility by utilizing the cross relatedness of input variables, but also partly alleviate the prediction uncertainty arisen by the kernel dictionary selection for nonstationary time series. In order to sufficiently capture the underlying dynamic characteristics in prediction-error time series, a generalized optimization strategy is designed to sequentially construct the kernel dictionary selection and weight vector updating procedures in multiple kernel connection modes. The generalized optimization strategy is highly flexible and effective, and it is capable of enhancing the ability to adaptively track the changing dynamic characteristics due to nonstationarity. Finally, in the perspective of top-level design, we summarize the information interaction between the network topology in kernel regressors and the optimization of inner model parameters. Numerical simulations demonstrate that the proposed approach has superior prediction performance for nonstationary time series.**

***Index Terms*—Covariance matrix adaptation evolution strategy, kernel adaptive filter algorithm, nonstationary time series, online prediction, prediction-error time series, radial basis function neural network.**

I. Introduction

Online prediction of nonstationary time series is a particularly challenging and pervasive problem in diverse fields of signal processing, machine learning, process control, extreme climate analysis, and neuroscience. Various online learning scenarios with inlier noise, noisy inliers, outliers, chaos, unknown dynamics, etc. have placed challenging requirements on the sequential decision making strategy for nonstationary time series [1]. Conventional approaches and their parameters updating mechanisms may be insufficient to track the changing underlying dynamic characteristics due to nonstationarity [2]. Specifically, in order to guarantee good prediction performances, the online prediction model must be adaptively adjusted based on the underlying dynamic characteristics, so as to properly balance the local characteristics and global characteristics. Moreover, the updating procedure within the online prediction model must be well organized, and is capable of determining the types of prediction functions and sequentially updating specific inner model parameters [3].

In the past several decades, kernel based online approaches in a reproducing kernel Hilbert space have been extensively studied, since the problem may become easier to solve with the so-called "kernel trick" if the observed input data are mapped onto a high-dimensional Hilbert space [4]. According to the representer theorem [5], the optimal solution to minimize the regularized empirical risk can be expressed as weighted kernels that compose of all the available training samples. However, in an online learning scenario, the dimension of the estimated weight vector will be continuously increasing along with the sequentially arrived data, which not only brings intractable computational complexity issue but also may degrade the generalization ability and hence decrease prediction performance [6], [7]. Therefore, in kernel based online modeling, the sparsification procedure is essential.

The sparsification techniques for online kernel modeling can be implemented in a supervised or unsupervised way, to properly select kernel dictionary. The unsupervised sparsification techniques construct appropriate kernel dictionary by just using the observed input data based on some selective modeling criteria, such as the approximate linear dependency (ALD) [7], [8], the coherence criterion [9], the distance criterion [10], [11], and their combinations [1], [12]. The supervised sparsification techniques adopt a variety of criteria that also require the observed output data, such as the two-part novelty condition of novelty criterion [13], the subjective information measure of surprise criterion [14], the changed significance of loss function [15], and the orthogonal forward selection (OFS) procedure for online tunable gradient radial basis function (RBF) networks [16]. A particular sparsification approach is based on the subset selection of the training samples, which can also be viewed as an indirect approach

The authors would like to gratefully thank the anonymous reviewers for their careful reviews and remarks that improved the quality and clarity of this paper. This work was supported in part by the Fujian Science & Technology Innovation Laboratory for Optoelectronic Information of China under Grant 2021ZZ121, and in part by the National Natural Science Foundation of China under Grant 61873143. *(Corresponding author: Jinhua Guo.)*

J. Guo is with Fujian Provincial Key Laboratory of Intelligent Identification and Control of Complex Dynamic Systems, Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences, Fuzhou 350002, China (E-mail: 18202237256@163.com).

H. Chen is with Fujian Provincial Key Laboratory of Intelligent Identification and Control of Complex Dynamic Systems, Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences, Fuzhou 350002, China, and also with Fujian Science & Technology Innovation Laboratory for Optoelectronic Information of China, Fuzhou 350108, China (E-mail: chenhao@fjirsm.ac.cn).

J. Zhang is with Department of Automation, Tsinghua University, Beijing 100084, China (E-mail: zjx18@mails.tsinghua.edu.cn).

S. Chen is with School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (E-mail: sqc@ecs.soton.ac.uk).

to constructing the kernel dictionary, such as the information-theoretic learning (ITL) criterion [17], [18]. Moreover, some feature selection approaches in time series analysis also can be easily converted to sparsification techniques [19], including the emerging shapelet-based methods which provide a supplement to the kernel-based approaches due to their time series segments selection procedure [20], [21].

Within the aforementioned sparsification techniques, there are two representative approaches for online prediction. One is the kernel adaptive filter (KAF) approach [22], and the other one is the online tunable RBF approach with the fixed kernel dictionary size [23], [24]. We will review these two representative approaches, which motivates us to establish the new structure parameter optimization approach and generalized optimization strategy that can be directly implemented on or partly combined with the aforementioned online learning algorithms. Comparing with other online learning approaches, the sparsification techniques impose less computation complexity while improving generalization. For the comparison of RBF approaches with other neural networks, the reader is referred to [16], [23], [24]. For the comparison of sparsification techniques aided KAF approaches with other KAF variants, the reader is referred to [1], [12], [22].

In addition to the kernel dictionary, the other structure parameter of online kernel modeling is the kernel covariance matrix in each kernel, which takes effect in the metric space that is formed by the observed input data. Without loss of generality, the Gaussian kernel is used as the default one in this paper. To prevent from inducing intractable computational tasks, most Gaussian kernel based online modeling approaches choose to use restricted forms of kernel covariance matrix, including the isotropic matrix, which is proportional to the identity matrix, and the diagonal matrix, which can only capture axis-aligned dynamic characteristics [7], [16], [25]. Comparing to the restricted forms, the general symmetric covariance matrix considers the cross relatedness of input variables, and it greatly improves the kernel structure's flexibility as well as the online prediction modeling performance at the expense of increased computational complexity in optimizing the kernel covariance matrix. The general symmetric kernel covariance matrix can be interpreted as a coordinate transformation of shifting and rotating with respect to the original coordinates [26], [27].

The covariance matrix adaptation evolution strategy (CMA-ES) [27] can be adopted to optimize the kernel covariance matrix for online modeling in the following two ways. Firstly, the evolving mechanism for its covariance matrix of the Gaussian distribution in the CMA-ES can be adopted or extracted as the optimization or evolution mechanism for the kernel covariance matrix. However, the covariance matrix of the CMA-ES is very different in nature to the kernel covariance matrix in online prediction modeling [28], and therefore the control parameters of the selection and recombination operators in the CMA-ES must be carefully modified to match with the online prediction modeling procedure. The other approach directly uses a CMA-ES algorithm to optimize the kernel covariance matrix based on a prediction performance related objective function, which preserves the well-designed evolution strategy of the CMA-ES [29]. We also use the CMA-ES directly to optimize the symmetric kernel covariance matrix in this paper, and its effects on the prediction performance, in terms of both the sparsification techniques and weight vector updating approaches, are studied. The intermittent optimization of symmetric kernel covariance matrix using the CMA-ES not only improves the kernel structure's flexibility by utilizing the cross relatedness of input variables but also partly alleviates the prediction uncertainty that arisen by the kernel dictionary selection for nonstationary time series.

In order to sufficiently and timely capture the underlying time-varying dynamic characteristics in nonstationary time series, a variety of optimization strategies can be implemented to update both the structure parameters and weight vector. One representative approach to track the nonstationarity is organically combining the construction and elimination procedures of the kernel dictionary selection, which can be achieved by properly setting the sparsification criteria [30], [31]. Another representative approach is intermittently replacing the elements of existing kernel dictionary with new arrived input data, and hence it brings the beneficial property of relative-fixed kernel dictionary size [16], [24], [32]. For the weight vector adaptation procedure, forgetting mechanisms are typically introduced into the objective function to enhance the tracking ability, such as the sliding-window approach [33], the exponential state forgetting factor [34], and the multi-innovations based approaches [35], [36].

In this paper, inspired by the prediction-error compensation principle and the connectionist representational schemes [37], [38], we design a generalized optimization strategy to sequentially construct the kernel dictionary selection and weight vector updating procedures in multiple kernel connection modes. Specifically, three connection modes of kernel regressors are established to organically combine their complementary prediction abilities by using the time-varying prediction-error time series. This generalized optimization strategy is highly flexible and effective, thus enhances the ability to adaptively track the changing dynamic characteristics due to nonstationarity. We further discuss the information interaction between the network topology in kernel regressors and the optimization of inner model parameters, in the proposed design. In summary, the main contribution of this paper is to propose a structure parameter optimized kernel based online prediction approach with a generalized optimization strategy for nonstationary time series, which includes the following specific aspects.

1) By analyzing the existing sparsification techniques in sequential decision making, the uncertainty caused by nonstationarity in both kernel dictionary selection and weight vector updating is revealed.
2) In the light of CMA-ES, the intermittent optimization of the real symmetric form of kernel covariance matrix is realized, which not only improves the kernel structure's flexibility by utilizing the cross relatedness of input variables but also partly alleviates the prediction uncertainty that arisen by the kernel dictionary selection for nonstationary time series.
3) A generalized optimization strategy is designed to sequentially construct the kernel dictionary selection and weight vector updating procedures in multiple kernel

connection modes, which is highly flexible and effective, thus enhances the ability to adaptively track the changing dynamic characteristics due to nonstationarity.

4) An online algorithm with the generalized optimization strategy and the intermittent optimization of kernel covariance matrices is designed to improve information interaction of key kernel parameters, and its effectiveness in enhancing the prediction performances for nonstationary time series is validated in numerical simulations.

Section II analyzes existing online kernel algorithms, which also provides the motivation for our current work. In the context of CMA-ES, Section III realizes the intermittent optimization of the symmetric Gaussian kernel covariance matrix. The generalized optimization strategy with multiple kernel connection modes is presented in Section IV. Section V summarizes the proposed online modeling approach and discusses the potential benefits and some unsolved issues. Section VI examines the effectiveness of the proposed approach with numerical simulations. The paper concludes in Section VII.

II. Analysis of Sparsification Techniques Based Online Kernel Modeling

In this section, representative online prediction algorithms of sparsification techniques are reviewed. By analyzing these sparsification techniques in sequential decision making, the unified recursive operation of kernel dictionary selection procedures is realized, and the differences of weight vector updating procedures among the representative online prediction algorithms are compared. Most importantly, it reveals that in order to sufficiently capture the underlying dynamic characteristics in highly nonstationary environments, the uncertainties arisen in both kernel dictionary selection and recursive weight vector updating necessitate new generalized optimization strategy for kernel parameter optimization.

Assume that we are sequentially given a stream of input-output data pairs $\{\boldsymbol{x}_n, y_n\}_{n=1}^{N}$, with the p_x-dimensional input time series $\boldsymbol{x}_n \in \mathbb{R}^{p_x \times 1}$ and the corresponding output time series $y_n \in \mathbb{R}$. The input-output data can be denoted as $\boldsymbol{X} = \left[\boldsymbol{x}_1 \cdots \boldsymbol{x}_N\right]^{\mathrm{T}} \in \mathbb{R}^{N \times p_x}$ and $\boldsymbol{y} = \left[y_1 \cdots y_N\right]^{\mathrm{T}} \in \mathbb{R}^{N \times 1}$. We describe the mapping procedure as

$$\boldsymbol{\varphi}: \quad \mathcal{X} \to \mathcal{H}, \quad \boldsymbol{x} \to \boldsymbol{\varphi}(\boldsymbol{x}) \tag{1}$$

where the high- or infinite-dimensional Hilbert space $\mathcal{H} = \{\boldsymbol{\varphi}(\boldsymbol{x}) | \boldsymbol{x} \in \boldsymbol{X}\}$. A linear combination of the selected kernels at time step n can be obtained as the prediction function

$$f(\boldsymbol{x}) = \sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i k\big(\widetilde{\boldsymbol{x}}_i, \boldsymbol{x}\big) \tag{2}$$

where $D(n) = \left\{\widetilde{\boldsymbol{x}}_i\right\}_{i=1}^{m}$ denotes the selected kernel dictionary with the size $m \ll N$, the reproduced kernel function is defined as $k\big(\boldsymbol{x}_i, \boldsymbol{x}_j\big) = \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j)\rangle$, and the m-dimensional weight vector is given by $\widetilde{\boldsymbol{\alpha}} = \left[\widetilde{\alpha}_1 \cdots \widetilde{\alpha}_m\right]^{\mathrm{T}}$.

A. Procedure of Kernel Dictionary Selection

1) KAF Sparsification Criteria: The ALD criterion [7] considers the linear dependency between the selected kernels in the form of $\boldsymbol{\varphi}(\mathbf{x})$ and it is defined as

$$\delta_1(n) = \min \left\| \sum\nolimits_{i=1}^{m} \alpha_i(n) \boldsymbol{\varphi}\big(\widetilde{\boldsymbol{x}}_i\big) - \boldsymbol{\varphi}\big(\boldsymbol{x}_n\big) \right\|^2 \le \nu_1 \tag{3}$$

where $\boldsymbol{\alpha}(n) = \left[\alpha_1(n) \cdots \alpha_m(n)\right]^{\mathrm{T}}$ is the coefficient vector to form a linear combination of the selected kernels $\left\{\boldsymbol{\varphi}\big(\widetilde{\boldsymbol{x}}_i\big)\right\}_{i=1}^{m}$, and ν_1 denotes the given threshold. Performing the minimization (3), we can check whether this condition is satisfied and obtain the optimal coefficient vector $\boldsymbol{\alpha}(n)$

$$\boldsymbol{\alpha}(n) = \widetilde{\boldsymbol{K}}^{-1}(n-1) \widetilde{\boldsymbol{k}}_{n-1}\big(\boldsymbol{x}_n\big) \tag{4}$$

$$\delta_1(n) = k_{n,n} - \widetilde{\boldsymbol{k}}_{n-1}^{\mathrm{T}}\big(\boldsymbol{x}_n\big) \boldsymbol{\alpha}(n) \tag{5}$$

where $\widetilde{\boldsymbol{K}}(n-1) \in \mathbb{R}^{m \times m}$ whose (i, j)-th element is $\widetilde{K}_{i,j}(n-1) = k\big(\widetilde{\boldsymbol{x}}_i, \widetilde{\boldsymbol{x}}_j\big)$ for $\widetilde{\boldsymbol{x}}_i, \widetilde{\boldsymbol{x}}_j \in D(n-1)$, $\widetilde{\boldsymbol{k}}_{n-1}\big(\boldsymbol{x}_n\big) \in \mathbb{R}^{m \times 1}$ whose i-th element is $k\big(\widetilde{\boldsymbol{x}}_i, \boldsymbol{x}_n\big)$, and $k_{n,n} = k\big(\boldsymbol{x}_n, \boldsymbol{x}_n\big)$. Consequently, for every n, we have

$$\boldsymbol{\varphi}\big(\boldsymbol{x}_n\big) = \sum\nolimits_{i=1}^{m(n)} \alpha_i(n) \boldsymbol{\varphi}\big(\widetilde{\boldsymbol{x}}_i\big) + \boldsymbol{\varphi}^{res}(n) \tag{6}$$

$$\left\|\boldsymbol{\varphi}^{res}(n)\right\|^2 \le \nu_1 \tag{7}$$

$$\boldsymbol{\Phi}_n = \widetilde{\boldsymbol{\Phi}}_n \boldsymbol{A}^{\mathrm{T}}(n) + \boldsymbol{\Phi}_n^{res} \tag{8}$$

where $\boldsymbol{\varphi}^{res}(n)$ is the residual vector, $\boldsymbol{\Phi}_n = \big[\boldsymbol{\varphi}(\boldsymbol{x}_1) \cdots \boldsymbol{\varphi}(\boldsymbol{x}_n)\big]$, $\widetilde{\boldsymbol{\Phi}}_n = \big[\boldsymbol{\varphi}(\widetilde{\boldsymbol{x}}_1) \cdots \boldsymbol{\varphi}(\widetilde{\boldsymbol{x}}_m)\big]$, and $\boldsymbol{\Phi}_n^{res} = \big[\boldsymbol{\varphi}^{res}(1) \cdots \boldsymbol{\varphi}^{res}(n)\big]$, while $\boldsymbol{A}(n) = \big[\boldsymbol{A}^{\mathrm{T}}(n-1) \ \boldsymbol{\alpha}(n)\big]^{\mathrm{T}} \in \mathbb{R}^{n \times m}$ is the coefficients matrix. Here we have explicitly indicate that m depends on n by using $m(n)$ in (6).

The quantized kernel recursive least squares (QKRLS) algorithm [10] uses the distance criterion, which is based on the principle that the principal neighborhoods of data-clustered regions can be approximately represented by the selected kernel dictionary. This criterion is defined as

$$\begin{aligned} \delta_2(n) &= \left\|\boldsymbol{x}_n - \widetilde{\boldsymbol{x}}_{j^\star}\right\|^2 \ \le \nu_2 \\ j^\star &= \arg\min_{1 \le j \le m} \left\|\boldsymbol{x}_n - \widetilde{\boldsymbol{x}}_j\right\|^2 \end{aligned} \tag{9}$$

where ν_2 denotes the given threshold.

In [15], the changed significance of the loss function is used to evaluate the significance of each observed input data as a kernel dictionary member. The criterion is defined as

$$\delta_3(n) = \frac{1}{2} \Delta\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}} \boldsymbol{H}_l(n) \Delta\widetilde{\boldsymbol{\alpha}} \ \le \nu_3 \tag{10}$$

where $\Delta\widetilde{\boldsymbol{\alpha}}$ is the change of the weight vector, $\boldsymbol{H}_l(n)$ is the Hessian of the loss function, and ν_3 is the given threshold.

2) Tunable RBF Sparsification Approaches: With a fixed kernel dictionary size, the online tunable RBF algorithms of [16], [23], [24] construct the initial kernel dictionary through two approaches. One approach depends on the distribution of the recent observed input data, e.g., the centers based on the nearest-neighbor. The other approach is the OFS procedure [16], which evaluates the significance of each training input data as a kernel dictionary member.

Given the training data $\{\boldsymbol{x}_t, y_t\}_{t=1}^{N_t}$, the full N_t-term relationship between the output of the predictor and the actual output $\boldsymbol{y}_{N_t} = \left[y_1 \cdots y_{N_t}\right]^{\mathrm{T}}$ can be expressed as

$$\boldsymbol{y}_{N_t} = \boldsymbol{K}_{N_t} \boldsymbol{\alpha}_{N_t} + \boldsymbol{e}_{N_t} \tag{11}$$

where $\boldsymbol{K}_{N_t} \in \mathbb{R}^{N_t \times N_t}$ is the full regression matrix whose (i,j)-th element is $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{\alpha}_{N_t}$ is the full weight vector, and $\boldsymbol{e}_{N_t}$ denotes the error vector. The orthogonal decomposition of $\boldsymbol{K}_{N_t}$ is given by $\boldsymbol{K}_{N_t} = \boldsymbol{W}_{N_t}\boldsymbol{A}_{N_t} = \big[\boldsymbol{w}_1 \cdots \boldsymbol{w}_{N_t}\big]\boldsymbol{A}_{N_t}$, where $\boldsymbol{w}_t$, $1 \le t \le N_t$, are the set of orthogonal bases and $\boldsymbol{A}_{N_t}$ is an unit upper triangular matrix [39], [40]. Then the expression (11) can be rewritten as

$$\boldsymbol{y}_{N_t} = \boldsymbol{W}_{N_t}\boldsymbol{g}_{N_t} + \boldsymbol{e}_{N_t} \tag{12}$$

where $\boldsymbol{g}_{N_t} = \boldsymbol{A}_{N_t}\boldsymbol{\alpha}_{N_t} = \big[g_1 \cdots g_{N_t}\big]^{\mathrm{T}}$ with $g_i = \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{y}_{N_t}/\big(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i\big)$ for $1 \le i \le N_t$. From (12), the sum of squares of the output $\boldsymbol{y}_{N_t}$ is given by

$$\boldsymbol{y}_{N_t}^{\mathrm{T}}\boldsymbol{y}_{N_t} = \sum\nolimits_{j=1}^{N_t} g_j^2 \boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{w}_j + \boldsymbol{e}_{N_t}^{\mathrm{T}}\boldsymbol{e}_{N_t} \tag{13}$$

Therefore, the error reduction ratio to evaluate the significance of each training input data is defined as

$$[\mathrm{err}]_j = g_j^2 \boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{w}_j / \boldsymbol{y}_{N_t}^{\mathrm{T}}\boldsymbol{y}_{N_t} \tag{14}$$

Remark 1. The kernel dictionary selections based on the criteria (3), (9), (10) and (14) can all be performed recursively, see [7], [10], [15], [40]. This is similar to the ALD kernel recursive least squares (ALD-KRLS) algorithm [32]. The other tunable RBF approaches that depend on the recent observed input data also can be performed in a recursive way [23], [24]. Therefore, the recursive kernel dictionary selection unifies the aforementioned online prediction algorithms.

3) Analysis of Kernel Dictionary Selection: Kernel dictionary selection is an objective-oriented problem, which leads to the diverse sparsification criteria and the organic combination of different sparsification approaches [1], [6], [12]. The optimal solution to minimize the regularized empirical risk can be expressed as weighted kernels that composed of all the available training samples [5]. Then the sparsification can be viewed as a procedure to deal with the differences between the two prediction spaces, one is formed by all the available training samples and the other is formed by the selected kernel dictionary. Since the high- or infinite-dimensional Hilbert space $\mathcal{H}$ may cause the excessive growth of kernel dictionary, sparsification approaches help to properly construct the kernel dictionary in order to obtain better generalization ability. From this perspective, sparsification in the kernel dictionary selection procedure becomes an effective measure of parameter regularization for superior prediction performance.

Once a sparsification approach is chosen, it is vital to properly set the threshold of the sparsification criterion and the termination condition of the kernel dictionary selection. The size of kernel dictionary is determined by the termination condition, which plays a crucial role in controlling the model generalization ability. As the sequential data arrive, the final selected kernel dictionary depends on both the first selected member $\widetilde{\boldsymbol{x}}_1$ and the given threshold, if other parameters in the kernel dictionary selection procedure have already been properly set. For the methods of deciding the thresholds, the reader is referred to the aforementioned references. It is worth recapping that the first selected member $\widetilde{\boldsymbol{x}}_1$ can make much difference to the final selected kernel dictionary, since the selected kernel dictionary is generated in a recursive way and all the subsequent selected members should fulfill the criteria that established by the already existed kernel dictionary. This is especially true for the nonstationary time series.

Remark 2. Given a nonstationary time series segment, the uncertainty caused by the first selected member $\widetilde{\boldsymbol{x}}_1$ is reflected on both the size of kernel dictionary and the actual selected kernel dictionary members. Although the issue of kernel dictionary size has been well-studied [1], [16], [24], the prediction uncertainty due to the actual selected kernel dictionary members has not been properly focused. How to handle this uncertainty in kernel dictionary selection caused by nonstationarity will be discussed in Sections III and IV.

B. Procedure of Weight Vector Updating

We now discuss the weight updating procedures in the representative online prediction algorithms, and further analyze the uncertainty arisen in the weight updating procedures.

1) Weight Vector Updating in KAFs: Considering for example the ALD-KRLS algorithm [7], the sparsification procedure at time step n can be denoted as

$$f(\boldsymbol{x}) = \sum\nolimits_{i=1}^{n} \alpha_i(n) k\big(\boldsymbol{x}_i, \boldsymbol{x}\big) \to \sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i(n) k\big(\widetilde{\boldsymbol{x}}_i, \boldsymbol{x}\big) \tag{15}$$

With the selected kernel dictionary in (8), the loss function $\mathcal{L}(\boldsymbol{\omega}_n)$ can be defined as

$$\mathcal{L}(\boldsymbol{\omega}_n) = \Big\|\boldsymbol{y}_n - \boldsymbol{\Phi}_n\boldsymbol{\omega}_n\Big\|^2 \tag{16}$$

where $\boldsymbol{y}_n = \big[y_1 \cdots y_n\big]^{\mathrm{T}}$. The optimal solution of $\boldsymbol{\omega}_n$ that minimizes $\mathcal{L}(\boldsymbol{\omega}_n)$ can be expressed as

$$\boldsymbol{\omega}_n = \sum\nolimits_{i=1}^{n} \alpha_i(n)\boldsymbol{\varphi}\big(\boldsymbol{x}_i\big) = \boldsymbol{\Phi}_n\boldsymbol{\alpha}(n) \tag{17}$$

By omitting the residual component vector $\boldsymbol{\varphi}^{res}(n)$ in (8), we have the approximation $\boldsymbol{\omega}_n = \boldsymbol{\Phi}_n\boldsymbol{\alpha}(n) \approx \widetilde{\boldsymbol{\Phi}}_n\boldsymbol{A}^{\mathrm{T}}(n)\boldsymbol{\alpha}(n) = \widetilde{\boldsymbol{\Phi}}_n\widetilde{\boldsymbol{\alpha}}(n)$, where $\widetilde{\boldsymbol{\alpha}}(n) = \boldsymbol{A}^{\mathrm{T}}(n)\boldsymbol{\alpha}(n)$. Then the loss function $\mathcal{L}\big(\widetilde{\boldsymbol{\alpha}}(n)\big)$ can be defined as

$$\begin{aligned}\mathcal{L}\big(\widetilde{\boldsymbol{\alpha}}(n)\big) =& \Big\|\boldsymbol{y}_n - \boldsymbol{\Phi}_n^{\mathrm{T}}\widetilde{\boldsymbol{\Phi}}_n\widetilde{\boldsymbol{\alpha}}(n)\Big\|^2 \\ =& \Big\|\boldsymbol{y}_n - \boldsymbol{A}(n)\widetilde{\boldsymbol{K}}(n)\widetilde{\boldsymbol{\alpha}}(n)\Big\|^2\end{aligned} \tag{18}$$

The optimal weight vector $\widetilde{\boldsymbol{\alpha}}(n)$ can be directly obtained by the least squares algorithm as

$$\begin{aligned}\widetilde{\boldsymbol{\alpha}}(n) =& \Big(\boldsymbol{A}(n)\widetilde{\boldsymbol{K}}(n)\Big)^{\dagger}\boldsymbol{y}_n \\ =& \widetilde{\boldsymbol{K}}(n)^{-1}\left(\boldsymbol{A}^{\mathrm{T}}(n)\boldsymbol{A}(n)\right)^{-1}\boldsymbol{A}^{\mathrm{T}}(n)\boldsymbol{y}_n\end{aligned} \tag{19}$$

2) Weight Vector Updating in Tunable RBF Algorithms: In [16], [24], the weight vector $\widetilde{\boldsymbol{\alpha}}$ in (2) is updated with the multi-innovation recursive least squares (RLS) algorithm, which uses the latest p innovations to form the regression matrix. The cumulative loss function $\mathcal{L}\big(\widetilde{\boldsymbol{\alpha}}(n)\big)$ in this case is defined as

$$\mathcal{L}\big(\widetilde{\boldsymbol{\alpha}}(n)\big) = \frac{1}{2}\sum_{i=n-p+1}^{n} \beta^{n-i}\Big(y_i - \widetilde{\boldsymbol{k}}_n^{\mathrm{T}}(\boldsymbol{x}_i)\widetilde{\boldsymbol{\alpha}}(n)\Big)^2 \tag{20}$$

where β is the forgetting factor. The associated information matrix is $\widetilde{\boldsymbol{K}}_p = \big[\widetilde{\boldsymbol{k}}_n(\boldsymbol{x}_{n-p+1}) \cdots \widetilde{\boldsymbol{k}}_n(\boldsymbol{x}_n)\big]^{\mathrm{T}}$, and the optimal weight vector $\widetilde{\boldsymbol{\alpha}}(n)$ is obtained by the RLS algorithm:

$$\begin{cases} \boldsymbol{\Psi}_n &= \boldsymbol{P}_{n-1}\widetilde{\boldsymbol{K}}_p^{\mathrm{T}}\big(\beta \boldsymbol{I}_p + \widetilde{\boldsymbol{K}}_p \boldsymbol{P}_{n-1}\widetilde{\boldsymbol{K}}_p^{\mathrm{T}}\big)^{-1} \\ \boldsymbol{P}_n &= \big(\boldsymbol{P}_{n-1} - \boldsymbol{\Psi}_n \widetilde{\boldsymbol{K}}_p \boldsymbol{P}_{n-1}\big)\beta^{-1} \\ \widetilde{\boldsymbol{\alpha}}(n) &= \widetilde{\boldsymbol{\alpha}}(n-1) + \boldsymbol{\Psi}_n \boldsymbol{e}_p \end{cases} \tag{21}$$

where $\boldsymbol{\Psi}_n \in \mathbb{R}^{m\times p}$ is the Kalman gain matrix, $\boldsymbol{P}_n \in \mathbb{R}^{m\times m}$ is the inverse of the covariance matrix updated by the information matrix $\widetilde{\boldsymbol{K}}_p$, $\boldsymbol{I}_p$ denotes the $p\times p$ identity matrix, and $\boldsymbol{e}_p$ is the error vector of the latest p predictors.

3) Weight Vector Updating in Robust Online Kernel Learning: Instead of using the cumulative loss function $\mathcal{L}\big(\widetilde{\boldsymbol{\alpha}}(n)\big)$ (20), the robust recurrent kernel online learning (RRKOL) [41] utilizes the instantaneous prediction error e_n^2, together with the extra information provided by the past recurrent feedback signals $y_{n-1}, \cdots, y_{n-d}$ that are included in the input variable $\boldsymbol{x}_n$. The derivative of the instantaneous prediction error e_n^2 with respect to $\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}(n)$:

$$\left.\frac{\mathrm{d}e_n^2}{\mathrm{d}\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}(n)}\right|_{\boldsymbol{\Lambda}_n} = \frac{\partial\big(e_n^2\big)}{\partial\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}(n)} + \frac{\partial\big(e_n^2\big)}{\partial\widetilde{\boldsymbol{k}}_n}\frac{\partial\widetilde{\boldsymbol{k}}_n}{\partial\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}}(n)}\boldsymbol{\Lambda}_n \tag{22}$$

is used to update the weight vector $\widetilde{\boldsymbol{\alpha}}(n)$, where $\boldsymbol{\Lambda}_n$ is the hyperparameter matrix to weight the second term. The distinct attribute of the RRKOL algorithm is this second term in (22) which considers the past feedback signals within $\boldsymbol{x}_n$. It provides a distinct insight into the role played by the past recurrent feedback signals in the weight vector updating.

4) Analysis of Weight Vector Updating: The weight vector updating approaches of online kernel modeling can be classified into two categories, based on whether or not to explicitly consider the property of the mapping function $\boldsymbol{\varphi}(\boldsymbol{x})$ in (2). Most KAF algorithms [7], [10], [15] consider the property of $\boldsymbol{\varphi}(\mathbf{x})$ in the weight vector updating procedure and the solution of $\widetilde{\boldsymbol{\alpha}}$ can be acquired by minimizing the corresponding loss functions, such as (17) and (19). For the online tunable RBF algorithms [16], [24] and the robust online kernel learning algorithm [41], the solution of $\widetilde{\boldsymbol{\alpha}}$ is acquired with the information matrix of the kernel vector, and they do not explicitly consider the property of the mapping function $\boldsymbol{\varphi}(\boldsymbol{x})$, such as (21) and (22).

Remark 3. The weight vector updating is also an objective-oriented problem. Whether or not to consider the property of the mapping function $\boldsymbol{\varphi}(\boldsymbol{x})$ may bring the differences in handling weight vector updating and further leads some uncertainty about prediction performance. It is not advisable trying to use the property of the mapping function $\boldsymbol{\varphi}(\mathbf{x})$ if how the property impacts on the prediction performance is insufficiently understood. Other weight vector updating techniques, such as the forgetting mechanisms, the regularized $\mathcal{L}\big(\widetilde{\boldsymbol{\alpha}}\big)$ and the sliding window/multi-innovation, can also be properly chosen to implement specific online modeling targets. In order to sufficiently capture the underlying dynamic characteristics in nonstationary time series, therefore, it is necessary to carefully design an optimization strategy to achieve the effective combination of these weight vector updating techniques, which will be addressed in Section IV and realized in Section V.

III. Intermittent Optimization of Kernel Covariance Matrix in the light of CMA-ES

Assume that the type of kernel is chosen a priori. Without loss of generality we use the Gaussian kernel. With the aid of the well-studied Gaussian kernel distribution, the important role of the kernel covariance matrix is clearly understood, in terms of shaping the prediction model [26], [27]. The Gaussian kernel based prediction function (2) can be described as

$$\begin{aligned} f(\boldsymbol{x}) &= \sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i k\big(\widetilde{\boldsymbol{x}}_i, \boldsymbol{x}\big) \\ &= \sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i \exp\Big(\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)/h_0\Big) \end{aligned} \tag{23}$$

where $\boldsymbol{\Sigma}_i \in \mathbb{R}^{p_x\times p_x}$ is the kernel covariance matrix of the i-th Gaussian kernel member and h_0 is an order of magnitude factor. The quadratic term Δ_i^2 can be expressed as

$$\begin{aligned} \Delta_i^2 &= \big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big) \\ &= \Big(\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)^{\mathrm{T}}\boldsymbol{U}_i\Big)\Big(\boldsymbol{U}_i^{\mathrm{T}}\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)\Big) \end{aligned} \tag{24}$$

where $\boldsymbol{U}_i = \big[\boldsymbol{d}_{i,1}/\sqrt{\lambda_{i,1}} \cdots \boldsymbol{d}_{i,p_x}/\sqrt{\lambda_{i,p_x}}\big]$, with $\lambda_{i,j}$ and $\boldsymbol{d}_{i,j}$, $1 \le j \le p_x$, denoting the eigenvalues and the corresponding eigenvectors of $\boldsymbol{\Sigma}_i$, respectively. The eigenvectors can be chosen to form an orthonormal set, and the matrix $\boldsymbol{U}_i$ can be interpreted as a coordinate transformation of shifting and rotating with respect to the original coordinates [26]. This coordinate transformation directly shapes the surface of the prediction function (23) and further influences the prediction performance.

The following weighted selection mechanism estimates the empirical kernel covariance matrix $\boldsymbol{\Sigma}^{emp}$ using selected observed sequential data set $\big\{\widehat{\boldsymbol{x}}_j\big\}_{j=1}^{N_e}$

$$\boldsymbol{\Sigma}_i^{emp} = h_0 \sum\nolimits_{j=1}^{N_e} \widehat{\omega}_j \big(\widehat{\boldsymbol{x}}_j - \widetilde{\boldsymbol{x}}_i\big)\big(\widehat{\boldsymbol{x}}_j - \widetilde{\boldsymbol{x}}_i\big)^{\mathrm{T}} \tag{25}$$

where $\widehat{\boldsymbol{\omega}} = \big[\widehat{\omega}_1 \cdots \widehat{\omega}_{N_e}\big]^{\mathrm{T}}$ denotes the weight vector. The sequential data sets can be selected based on the distribution of the observed samples, such as the widely-adopted nearest neighbor method for RBF kernels [16], [24], [42]. This class of estimation approaches use the selected sequential data sets to approximately represent the intended population of each Gaussian kernel regressor in (23), which contributes limited predictive attributes due to its unsupervised nature.

To realize the intermittent optimization of the kernel covariance matrix in the context of CMA-ES, the correlations of the kernel covariance matrix with other parameters need to be considered, in terms of both the kernel dictionary selection and weight vector updating. For the kernel dictionary selection procedures discussed in Section II, the actual selected kernel dictionary can be very different from the suboptimal or optimal kernel dictionary, due to the uncertainty caused by the nonstationarity. Given the optimal or suboptimal sparsified kernel dictionary $\big\{\widetilde{\boldsymbol{x}}_i^{\star}\big\}_{i=1}^{m}$, with the corresponding estimated kernel covariance matrices $\boldsymbol{\Sigma}_i^{\star}$ and updated weights $\widetilde{\alpha}_i^{\star}$, the prediction function (23) becomes

$$\begin{aligned}f^{\star}(\boldsymbol{x}) =&\sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i^{\star} \exp\Big(\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i^{\star}\big)^{\mathrm{T}} \boldsymbol{\Sigma}_i^{\star-1}\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i^{\star}\big)/h_0\Big)\\ =&\sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i^{\star} \exp\Big(\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)^{\mathrm{T}} \boldsymbol{C}\boldsymbol{\Sigma}_i^{\star-1}\boldsymbol{C}^{\mathrm{T}}\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)/h_0\Big)\\ =&\sum\nolimits_{i=1}^{m} \widetilde{\alpha}_i^{\star} \exp\Big(\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)^{\mathrm{T}} \widetilde{\boldsymbol{\Sigma}}_i^{-1}\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)/h_0\Big) \quad (26)\end{aligned}$$

where $\boldsymbol{C}^{\mathrm{T}}$ is the transformation matrix from $\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i\big)$ to $\big(\boldsymbol{x}-\widetilde{\boldsymbol{x}}_i^{\star}\big)$. Comparing the prediction functions in (23) and (26), it can be seen that the variable to be optimized should be $\widetilde{\boldsymbol{\Sigma}}_i^{-1} = \boldsymbol{C}\boldsymbol{\Sigma}_i^{\star-1}\boldsymbol{C}^{\mathrm{T}}$, not $\big(\boldsymbol{\Sigma}_i^{emp}\big)^{-1}$ of (25), if the kernel dictionary selection procedure and the weight vector updating procedure are taken into consideration. The optimization of the real symmetric matrix $\widetilde{\boldsymbol{\Sigma}}_i^{-1}$ greatly improves the kernel structure's flexibility than the restricted form of $\boldsymbol{\Sigma}_i^{-1}$. Similar to the principal components in principal component analysis method or the orthogonal search paths in evolutionary algorithms [43], the enhanced kernel structure flexibility enhances the ability of each kernel regressor in (26) to capture the underlying dynamic characteristics in the neighborhood.

For notational simplification, we drop the subindex i in the sequel. The real symmetric matrix $\widetilde{\boldsymbol{\Sigma}}^{-1}$ can be optimized in the following "rank-one updating" form

$$\widetilde{\boldsymbol{\Sigma}}^{-1}(n) = (1-c_0)\widetilde{\boldsymbol{\Sigma}}^{-1}(n-1) \pm \boldsymbol{p}_{\sigma}(n)\boldsymbol{p}_{\sigma}^{\mathrm{T}}(n) \qquad (27)$$

where c_0 is the learning rate which can be calculated as $c_0 = 2/p_x^2$ [27], and $\boldsymbol{p}_{\sigma}$ is the target vector to optimize the kernel covariance matrix. In order to give more predictive attributes to the intermittent optimization of $\widetilde{\boldsymbol{\Sigma}}^{-1}$, the objective function to be minimized is set to

$$\mathcal{L}_{\sigma} = \sum_{\boldsymbol{x}_j \in D_{\sigma}} \omega_{\sigma_j}\big(y_j - f^{\star}(\boldsymbol{x}_j)\big)^2 \qquad (28)$$

where D_{σ} is the set of the selected samples, and ω_{σ_j} are the weights of the loss to tradeoff the local and global characteristics. To better prevent the occurrence of some irregularities, such as outliers, during the online prediction procedure, the selected samples D_{σ} in (28) and the order of magnitude factor h_0 in (26) can be used to deal with these irregularities. The appropriate sample set D_{σ} actually provides an alternative interface for eliminating the irregular samples through corresponding modeling approaches, such as the outliers which can be identified via the robust signal decomposition [36], [44], and the unused training samples which can be determined by the subset selection approaches [17], [18]. An appropriate h_0 in (26) can help control the learning path of $\widetilde{\boldsymbol{\Sigma}}^{-1}$, especially when the intermittent optimization of $\widetilde{\boldsymbol{\Sigma}}^{-1}$ outputs specific numeric attributes, such as the positive integers.

According to the above analysis, we focus on directly using the CMA-ES to optimize the target vector $\boldsymbol{p}_{\sigma}$ in (27) based on the objective function $\mathcal{L}_{\sigma}$ of (28). We introduce the so-called "pure CMA-ES" algorithm to better describe the intermittent optimization of $\boldsymbol{p}_{\sigma}$. This pure CMA-ES algorithm is the fundamental and simplest version among many variants of the CMA-ES algorithms. As an evolutionary algorithm, it consists of three steps, the initialization step, the repeating evolution step with four operators of mutation, evaluation, selection and recombination, and the termination step [45].

Let the population size be λ_c. A population of new individuals $\boldsymbol{x}_k^{(g+1)}$ at the $(g+1)$-th generation are generated by the mutation operator, which samples a multivariate normal distribution $\mathcal{N}\big(\boldsymbol{0}, \boldsymbol{C}^{(g)}\big)$ with the zero mean vector $\boldsymbol{0}$ and the covariance matrix $\boldsymbol{C}^{(g)}$. The sampling operation, from the g-th generation to the next generation, can be described as

$$\boldsymbol{x}_k^{(g+1)} \sim \boldsymbol{m}^{(g)} + \sigma_c^{(g)}\mathcal{N}\big(\boldsymbol{0}, \boldsymbol{C}^{(g)}\big) \qquad (29)$$

where $\boldsymbol{m}^{(g)}$ denotes the mean vector of the g-th population individuals, and $\sigma_c^{(g)}$ is the mutation step-size. Then, evaluation of these mutated individuals on the objective function $\mathcal{L}_{\sigma}$ (28) is implemented, and the top μ_c ranked individuals, $\boldsymbol{x}_{k(1)}^{(g+1)}, \boldsymbol{x}_{k(2)}^{(g+1)}, \cdots, \boldsymbol{x}_{k(\mu_c)}^{(g+1)}$, are selected, where $\mu_c < \lambda_c$. Finally, the weighted recombination of the best μ_c ranked individuals is reflected in the following updating equations for the parameters of the mutation operator (29)

$$\boldsymbol{m}^{(g+1)} = \boldsymbol{m}^{(g)} + c_m \sum\nolimits_{i=1}^{\mu_c} \omega_m(i)\Big(\boldsymbol{x}_{k(i)}^{(g+1)} - \boldsymbol{m}^{(g)}\Big) \qquad (30)$$

$$\begin{aligned}\boldsymbol{C}^{(g+1)} =& \left(1 - c_1 - c_{\mu}\sum_{i=1}^{\lambda_c}\omega_c(i)\right)\boldsymbol{C}^{(g)} + c_1 \underbrace{\boldsymbol{p}_{c1}^{(g+1)}\big(\boldsymbol{p}_{c1}^{(g+1)}\big)^{\mathrm{T}}}_{\text{rank-one update}}\\ &+ c_{\mu}\underbrace{\sum\nolimits_{i=1}^{\lambda_c}\omega_c(i)\boldsymbol{p}_{c\mu(i)}^{(g+1)}\big(\boldsymbol{p}_{c\mu(i)}^{(g+1)}\big)^{\mathrm{T}}}_{\text{rank-}\mu_c\text{ update}} \qquad (31)\end{aligned}$$

$$\sigma_c^{(g+1)} = \sigma_c^{(g)} \exp\left(\frac{c_{\sigma_c}}{d_{\sigma_c}}\left(\frac{\|\boldsymbol{p}_{\sigma_c}{}^{(g+1)}\|}{\widehat{\chi}_{p_x}} - 1\right)\right) \qquad (32)$$

where $\boldsymbol{p}_{c1}$ and $\big\{\boldsymbol{p}_{c\mu(1)}, \cdots, \boldsymbol{p}_{c\mu(\lambda_c)}\big\}$ are respectively the well designed evolution paths of the "rank-one update" and "rank-μ_c update", $\boldsymbol{p}_{\sigma_c}$ is the cumulative step-size control evolution path for the mutation step-size, $\boldsymbol{\omega}_c = \big[\omega_c(1)\cdots\omega_c(\lambda_c)\big]^{\mathrm{T}}$ and $\boldsymbol{\omega}_m = \big[\omega_m(1)\cdots\omega_m(\lambda_c)\big]^{\mathrm{T}}$ are the respective coefficient vectors, while c_m, c_1, c_{μ} and $\frac{c_{\sigma_c}}{d_{\sigma_c}}$ are the learning rates of the respective evolution paths, and $\widehat{\chi}_{p_x}$ is the expected length of a random variable distributed according to $\mathcal{N}\big(\boldsymbol{0}, \boldsymbol{I}_{p_x}\big)$ which helps to normalize the length of $\boldsymbol{p}_{\sigma_c}$. The detailed parameter settings can be found in [27].

As a randomized search algorithm, the fundamental design principles of the CMA-ES are the invariance, which enables an identical behavior on a class of objective functions, and the unbiasedness, which partly prevents the risk of divergence or premature convergence [27]. These properties provide algorithmic setting options to be chosen for online modeling in practical applications. The termination criterion in the CMA-ES should be carefully set according to specific online prediction problems. Our algorithm implementations of using the CMA-ES to optimize the kernel covariance matrix $\widetilde{\boldsymbol{\Sigma}}^{-1}$ (27) can be classified into two cases, based on whether the kernel dictionary selective criterion is independent of the calculation of the updated kernel covariance matrix $\widetilde{\boldsymbol{\Sigma}}^{-1}$ in (26). As summarized in Algorithm 1, the ALD criterion of (3) represents the case that the kernel dictionary selective criteria in the online prediction algorithms are dependent on the updated kernel covariance matrix. For Algorithm 2, the distance criterion of (9) represents the case that the kernel dictionary selective criteria are independent of the updated kernel covariance matrix. Algorithm 1 or 2 can be used

Algorithm 1 Intermittent Optimization of Kernel Covariance Matrix Using CMA-ES for ALD-KRLS

```
Input: $\{\boldsymbol{x}_n, y_n\}$, $\boldsymbol{x}_n \in D_\sigma$.
Initialization
  Initialize strategy parameters of CMA-ES [27].
  Initialize $\widetilde{\boldsymbol{\Sigma}}^{-1}(0)$ and $c_0$ in (27).
Repeat
  Mutation as in (29).
  Evaluation
    For each mutation individual
      Set each mutation individual as $\boldsymbol{p}_\sigma$.
      Calculate $\widetilde{\boldsymbol{\Sigma}}^{-1}$ as (27).
      Initialize $h_0$ in (26).
      Initialize the parameters as in ALD-KRLS [7].
      For each selected sample in $D_\sigma$
        Operate kernel dictionary selection procedure.
        Operate weight vector updating procedure.
        Calculate prediction performance.
      End for
      Return value of objective function $\mathcal{L}_\sigma$.
    End for
  Selection
    Return best ranked individuals $\boldsymbol{x}_{k(1)}^{(g+1)}, \cdots, \boldsymbol{x}_{k(\mu_c)}^{(g+1)}$.
  Recombination as (30) to (32).
Until termination criterion is fulfilled
Calculate $\widetilde{\boldsymbol{\Sigma}}^{-1}$ as (27) with $\boldsymbol{p}_\sigma = \boldsymbol{x}_{k(1)}^{(g+1)}$.
Return: final $\widetilde{\boldsymbol{\Sigma}}^{-1}$.
End
```

Algorithm 2 Intermittent Optimization of Kernel Covariance Matrix Using CMA-ES for QKRLS

```
Input: $\{\boldsymbol{x}_n, y_n\}$, $\boldsymbol{x}_n \in D_\sigma$.
Initialization
  Initialize the parameters as in QKRLS [10].
  Initialize $\widetilde{\boldsymbol{\Sigma}}^{-1}(0)$ and $c_0$ in (27).
  Initialize $h_0$ in (26).
  Operate kernel dictionary selection procedure.
  Initialize strategy parameters as in CMA-ES [27].
Repeat
  Mutation as in (29).
  Evaluation
    For each mutation individual
      Set each mutation individual as $\boldsymbol{p}_\sigma$.
      Calculate $\widetilde{\boldsymbol{\Sigma}}^{-1}$ as (27).
      For each selected sample in $D_\sigma$
        Operate weight vector updating procedure.
        Calculate prediction performance.
      End for
      Return value of objective function $\mathcal{L}_\sigma$.
    End for
  Selection
    Return best ranked individuals $\boldsymbol{x}_{k(1)}^{(g+1)}, \cdots, \boldsymbol{x}_{k(\mu_c)}^{(g+1)}$.
  Recombination as (30) to (32).
Until termination criterion is fulfilled
Calculate $\widetilde{\boldsymbol{\Sigma}}^{-1}$ as in (27) with $\boldsymbol{p}_\sigma = \boldsymbol{x}_{k(1)}^{(g+1)}$.
Return: final $\widetilde{\boldsymbol{\Sigma}}^{-1}$.
End
```

to optimize the symmetric kernel covariances matrices for the training procedure and online prediction procedure of the proposed algorithmic framework, namely, Algorithm 3 presented in Section V.

IV. Generalized Optimization Strategy in Kernel Connection Mode

In this section, we propose a generalized optimization strategy to sequentially construct the kernel dictionary selection and weight vector updating procedures in multiple kernel connection modes. Generally, the prediction-error time series is a good metric, providing a very useful clue to improve the prediction performance of online sequential data [37], [46]. In the kernel dictionary selection procedure, it is well known that the kernel dictionary size m is of vital importance to the generalization ability of the prediction function (2), and there usually exists a proper kernel dictionary size for specific observed sequential data. It implies that any part of the whole kernel regressors can be viewed as an error compensator for the other part of the kernel regressors, i.e., the compensator can help to further capture the underlying dynamics in the prediction-error time series that generated by the already existing part. We simply call this the principle of prediction-error compensation. Essentially, the prediction-error compensation principle is a product of the relationship between the whole and the parts of the kernel regressors.

In order to better realize the prediction-error compensation principle in the kernel dictionary selection procedure, a generalized optimization strategy is designed to construct the kernel connections in multiple kernel connection modes. Inspired by the connectionist representational schemes [38], three basic connection modes of kernel regressors are established to organically combine their complementary prediction abilities by using the time-varying prediction-error time series. Similar to the connections in the circuit topology [47], two or more kernel regressors are in *series connections* if their kernel dictionary elements are selected from the same time series (input time series or prediction-error time series) and the weights of these kernel regressors form a weight vector to be updated synchronously. Two or more groups of series-connected kernel regressors are in *parallel connections* if their kernel dictionary members are selected from the same time series, and in an order, the weight vector of the next parallel-connected group is to be updated according to the prediction-error time series that generated by the previous parallel-connected group. A cascade-connected group is a head-to-tail arrangement of two or more parallel-connected groups. In *cascade connections*, the kernel dictionary members of the next cascade-connected group are selected from another constructed time series, and the prediction-error time series that generated by the previous cascade-connected group is used to update the weight vectors of the next cascade-connected group. As illustrated in the connection diagram of Fig. 1, the kernel dictionaries of the cascade-connected groups R_a and R_b are selected from two diverse time series, and the time series TS_{a0} and TS_{b0} are

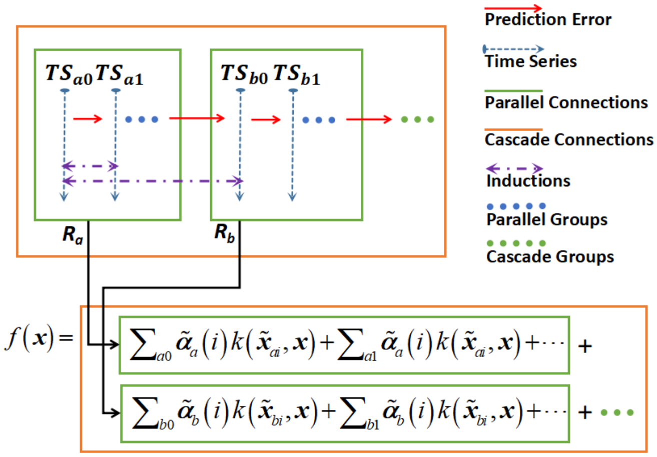

TS_{a0} TS_{a1}
TS_{b0} TS_{b1}
R_a
R_b
Prediction Error
Time Series
Parallel Connections
Cascade Connections
Inductions
Parallel Groups
Cascade Groups
$f(x) =$
$\sum_{a0} \tilde{\alpha}_a(i) k(\tilde{x}_{ai}, x) + \sum_{a1} \tilde{\alpha}_a(i) k(\tilde{x}_{ai}, x) + \cdots$ +
$\sum_{b0} \tilde{\alpha}_b(i) k(\tilde{x}_{bi}, x) + \sum_{b1} \tilde{\alpha}_b(i) k(\tilde{x}_{bi}, x) + \cdots$ +

Fig. 1. Illustration of the generalized connection modes.

used to update the weight vectors in R_a and R_b, respectively. The three basic connection modes can be distinguished in terms of how to realize the prediction-error compensation principle in the kernel dictionary selection and weight vector updating procedures.

After the initial construction of the three basic connection modes, specific relationships between different time series can be set, and some of the selected kernel dictionary members also can be transferred into other groups based on the adopted information criteria. In the three basic connection modes, some connections can be left out if not needed. All the adopted optimization strategies in the aforementioned kernel online algorithms can reconstruct their corresponding models, in terms of the three basic connection modes, which motivates us to call the framework of three basic connection modes a generalized optimization strategy. A specific algorithmic implementation of these basic connection modes is integrated within the training procedure and online prediction procedure of Algorithm 3 presented in Section V.

The generalized optimization strategy is particularly useful for online prediction of nonstationary time series. This is because it provides a more self-contained way to construct the entire kernel connections and thus better explores the complementary prediction abilities in the time-varying prediction-error time series, which enhances the ability to track the changing dynamic characteristics. The three basic connection modes divide the whole kernel regressors into different groups, which actually provides a perspective to handle with the relationship of the whole and the parts of kernel regressors. In the procedure of the generalized optimization strategy, the modeling of the next specific connection group can make a change accordingly if the previously determined connection groups have already provided useful information. More useful information can be acquired by monitoring the prediction performance of all the divided groups than by just monitoring the whole kernel regressors.

V. Summary of Proposed Approach

This section summarizes the top-level design of our proposed online algorithm, which integrates the information interaction between the network topology in kernel regressors to sequentially construct the kernel dictionary selection and weight vector updating procedures (of Section IV) and the optimization of inner model parameters (of Section III). Both the intermittent optimization of the kernel covariance matrix and the generalized optimization strategy in three basic kernel connection modes can improve the information interaction in both the kernel dictionary selection procedure and the weight vector updating procedure. The improved information interaction not only enhances the ability to deal with various online modeling problems but also provides a more flexible kernel structure and a more compatible way of kernel connections. It can be further combined with other existing online modeling techniques. An implementation or realization of this improved information interaction relies on gradually growing the additive kernel regressors to continuously produce the new prediction-error time series for the newly-added specific connection group, and then sequentially constructing their respective kernel dictionary. Given a nonstationary time series TS_{a0} of Fig. 1 to be predicted, the operations of this implementation or realization are as follows.

First the kernel dictionary selection procedure in the first cascade-connected group R_a can be naturally carried out by any online prediction algorithm of Section II. Specifically, the selected kernel dictionary members within R_a are divided into the different parallel-connected groups $(R_{a0}, R_{a1}, \cdots)$ in an orderly fashion. The online predictor within the first parallel-connected group R_{a0} operates its weight vector updating procedure, and its prediction-error time series TS_{a1} is continuously recorded. For the second parallel-connected group R_{a1}, the online predictor within R_{a1} tracks the dynamic characteristics underlying the prediction-error time series TS_{a1}, and it makes prediction for the forthcoming prediction errors in the prediction-error times series TS_{a1}. The next online predictor R_{a2} operates on the prediction-error time series TS_{a2} that are recorded by the previous online predictor within R_{a1}, and so on. In particular, the online predictor within R_{a1} is acquired by analyzing the effectiveness of different online algorithm candidates in the training procedure, and some of its parameters can be adjusted online with the sequentially arrived data. If the user decides to intermittently optimize the kernel covariance matrices, Algorithm 1 or 2 is applied depending on the type of online kernel modeling technique used. The aforementioned design principle for the online predictor within R_{a1} is repeated sequentially for the online predictors of the following parallel-connected groups within R_a.

For the second cascade-connected group R_b, the online predictors within R_b track the dynamic characteristics underlying the prediction-error time series TS_{b0} recorded by its previous cascade-connected group R_a, and they make prediction for the forthcoming prediction errors in the prediction-error times series TS_{b0}. For the online predictors within R_b, the elements of the input vector may come from the recorded prediction-error time series TS_{b0} or other variables provided by the previous cascade-connected groups, which leads to different kernel dictionary selections, compared with R_a. Also, the online predictors and parallel connections within R_b are acquired by analyzing the effectiveness of different online algorithm candidates in the training procedure. In practice, the

Algorithm 3 Illustrative Online Algorithm for Improved Information Interaction with Generalized Optimization Strategy and Intermittent Optimization of Kernel Covariance Matrices

```
Input: $\boldsymbol{X}_a$, $\boldsymbol{X}_b$; $\boldsymbol{y} = \big[y_1\ y_2 \cdots y_N\big]^{\mathrm{T}}$, i.e., $TS_{a0}$ of Fig. 1.
Training Procedure
Initialization
    Initialize thresholds $\nu_2^a$ in (9), $\nu_1^b$ in (3) to determine
    parallel connections in $R_a$ and $R_b$, respectively.
    Initialize thresholds $l_a$, $l_b$ for loss functions $\mathcal{L}_a$, $\mathcal{L}_b$.
    Initialize $(\widetilde{\boldsymbol{\Sigma}}_{a0}^{-1}, \widetilde{\boldsymbol{\Sigma}}_{a1}^{-1}, \cdots)$ and $(\widetilde{\boldsymbol{\Sigma}}_{b0}^{-1}, \widetilde{\boldsymbol{\Sigma}}_{b1}^{-1}, \cdots)$.
    Initialize $i = -1$, $j = -1$.
Initialize algorithmic parameters of ALD-KRLS [7] for $R_a$.
Select the kernel dictionary $D_a$ via (3) with $\boldsymbol{X}_a$.
Determine $(D_{a0}, D_{a1}, \cdots, D_{aN_a})$ via (9).
Repeat $i = i + 1$
    Calculate $\widetilde{\alpha}_{ai}$ via (19) with $TS_{ai}$.
    Optimize $\widetilde{\boldsymbol{\Sigma}}_{ai}^{-1}$ using Algorithm 1.
    Compute $\mathcal{L}_a$ and form $TS_{a(i+1)}$.
Until $i \geq N_a$ or $\mathcal{L}_a \leq l_a$
Send $TS_{a(i+1)}$ to $TS_{b0}$.
Initialize algorithmic parameters of QKRLS [10] for $R_b$.
Select the kernel dictionary $D_b$ via (9) with $\boldsymbol{X}_b$.
Determine $(D_{b0}, D_{b1}, \cdots, D_{bN_b})$ via (3).
Repeat $j = j + 1$
    Calculate $\widetilde{\alpha}_{bj}$ as in QKRLS with $TS_{bj}$.
    Optimize $\widetilde{\boldsymbol{\Sigma}}_{bj}^{-1}$ using Algorithm 2.
    Compute $\mathcal{L}_b$ and form $TS_{b(j+1)}$.
Until $j \geq N_b$ or $\mathcal{L}_b \leq l_b$
Return: $f(\boldsymbol{x})$ and $(D_{a0}, \cdots, D_{aN_a})$, $(D_{b0}, \cdots, D_{bN_b})$.
Online Prediction Procedure
For each new arriving sample $(x_n, y_n)$
    Compute MAE/MSE metrics of $(R_{a0}, R_{a1}, \cdots)$ and
    $(R_{b0}, R_{b1}, \cdots)$, respectively.
    Update $(TS_{a0}, TS_{a1}, \cdots)$ and $(TS_{b0}, TS_{b1}, \cdots)$.
    Calculate $(\widetilde{\alpha}_{a0}, \widetilde{\alpha}_{a1}, \cdots)$ and $(\widetilde{\alpha}_{b1}, \widetilde{\alpha}_{b2}, \cdots)$, sequentially.
    Adjust $(D_{a0}, \cdots, D_{aN_a})$, $(D_{b0}, \cdots, D_{bN_b})$ via (3), (9).
    Optimize $(\widetilde{\boldsymbol{\Sigma}}_{a0}^{-1}, \widetilde{\boldsymbol{\Sigma}}_{a1}^{-1}, \cdots)$ using Algorithm 1, and
    $(\widetilde{\boldsymbol{\Sigma}}_{b0}^{-1}, \widetilde{\boldsymbol{\Sigma}}_{b1}^{-1}, \cdots)$ using Algorithm 2, if necessary.
    Return: updated $f(\boldsymbol{x})$ and its updated connections.
End for
```

online predictors should be carefully constructed to achieve good prediction performance according to the effectiveness tests in the training procedure. The aforementioned design principle for the online predictor within R_b is repeated on the online predictors in the following cascade-connected groups $(R_c, \cdots)$. As shown in Fig. 1, the sum of all the generated online predictors, from the first to a certain following number of cascade-connected groups, constitutes the additive kernel model for the certain cascade-connected group. Most importantly, the adopted prediction function at a given time sample can be chosen from all the generated additive kernel models by monitoring their prediction performances online.

As the organic combination of the ALD and distance criteria has been well demonstrated to be effective [1], [12], we use these two selective criteria to illustrate how to perform the above mentioned information interaction specifically. The illustrated algorithm consists of both the training and online prediction procedures, and its generated online predictor is composed of two cascade groups R_a and R_b. To be specific, R_a adopts the ALD-KRLS algorithm [7] as the basic approach to operate the kernel dictionary selection and weight vector updating, and uses the distance criterion to determine the parallel connections in R_a. By contrast, R_b adopts the QKRLS algorithm [10] as the basic approach, and uses the ALD criterion to determine its parallel connections. The operations of this illustrative implementation are summarized in Algorithm 3, where the group indexes a and b are added to distinguish the corresponding variables and quantities in the two groups, e.g., the threshold ν_2 for the group R_a becomes ν_2^a and the threshold ν_1 for the group R_b becomes ν_1^b, D_a is the kernel dictionary for R_a and D_b is the kernel dictionary for R_b, etc. Compared with the existing algorithms proposed in [1] and [12], this new algorithm integrates the kernel modeling techniques more naturally, and reveals a deeper hierarchical network topology in kernel regressors. The beneficial properties of our approach and how specific modeling techniques used in Algorithm 3 affects its online prediction performance will be further studied in the next section.

In the online kernel modeling of nonstationary time series, intuitively the nonstationarity may be classified into two categories according to the prediction performance of the currently-adopted kernel model. One is the nonstationary trend, which can be captured by just optimizing the modeling parameters in the selected kernel model, and the other category is higher-order nonstationarity, which may need to be learned by changing the type of kernel function or even changing to other online prediction algorithms. How to change the Gaussian kernel function (23) to other type of kernel function during online operation is beyond the scope of this paper. However, the improved information interactions can enhance the currently-adopted kernel model's tracking ability as much as possible before the underlying nonstationarity is assigned to the second case, which may be beneficial to the selection procedure of online kernel models. Sometimes, if the unpredictable part of the nonstationarity has a great influence, the prediction performance can be a misleading indicator to determine whether the selected online kernel model should be changed. In this case, analyzing all the generated prediction-error time series can provide some useful clues to identify whether unpredictable nonstationarity accounts for a large part.

VI. Numerical Simulations

Three nonstationary time series, a chaotic time series, the capacitor-current time series in the second-order circuit with the unstable equilibrium point, and the real-world sunspot time series, are chosen in the experiments to investigate the effectiveness of the proposed approach. For the KAF algorithms, the ALD-KRLS algorithm [7], the improved KRLS (IKRLS) algorithm [12] [1] and the RRKOL algorithm [41] are

[1] The correct weight-vector updating formula (11) in the reference [12] is $\widetilde{\boldsymbol{\alpha}}(n) = \widetilde{\boldsymbol{\alpha}}(n-1) + \boldsymbol{Q}(n-1)\widetilde{\boldsymbol{k}}_{n-1}\big(y(n) - \widetilde{\boldsymbol{k}}_{n-1}^{\mathrm{T}}\widetilde{\boldsymbol{\alpha}}(n-1)\big)/\big(\lambda + \widetilde{\boldsymbol{k}}_{n-1}^{\mathrm{T}}\boldsymbol{Q}(n-1)\widetilde{\boldsymbol{k}}_{n-1}\big)$.

adopted as the representative algorithms to be studied. For the online tunable RBF algorithms, the fast tunable RBF (FT-RBF) algorithm [24] and the fast tunable gradient RBF (FT-GRBF) algorithm [16] are adopted as the representative algorithms.

A. Chaotic Time Series Online Prediction

Derived from a finite mode truncation of the partial differential equations, the Lorenz time series [1], [48] is given by three Lorenz differential equations

$$\begin{aligned}\frac{\mathrm{d}z_1(t)}{\mathrm{d}t} &= \sigma_1\big(z_2(t)-z_1(t)\big),\\ \frac{\mathrm{d}z_2(t)}{\mathrm{d}t} &= -z_1(t)z_3(t)+rz_1(t)-z_2(t),\\ \frac{\mathrm{d}z_3(t)}{\mathrm{d}t} &= z_1(t)z_2(t)-bz_3(t),\end{aligned} \tag{33}$$

where σ_1, r and b are the parameters that control the behavior of the Lorenz system. The Lorenz time series samples are generated with the step size 0.01 and the starting point (0,1,0). The first 3000 samples are set as the training dataset to obtain the optimal or appropriate parameters, and the 3000 $\sim$ 8000 samples are set as the testing dataset to examine the effectiveness of the studied algorithms. During the training procedure, the optimal or near-optimal parameter settings are acquired by validating the sequential prediction performance on the 2500 $\sim$ 3000 samples. For the online prediction of the dynamic input-output relationship in (33), we set the input vector as $\boldsymbol{x}_n = [z_1(n)\ z_2(n)\ z_3(n)]^{\mathrm{T}}$ to predict $y_n = z_2(n{+}5)$, with the time-varying control parameters $\sigma_1 = 10, b = \frac{1}{3}(4+3(1+\sin(0.1t))), r = 25+3\big(1+\cos\big(2^{0.001\mathrm{t}}\big)\big)$ [16], [23]. Since the input vector $\boldsymbol{x}_n$ here does not explicitly contain the past output y_n, the differencing operation for the input of the FT-GRBF algorithm and the second recurrent term of (22) in the RRKOL algorithm are not applied in this simulation [16], [41].

TABLE I
OVERVIEW OF REPRESENTATIVE ALGORITHMS FROM GENERALIZED OPTIMIZATION VIEWPOINT

Algorithms	$\widetilde{\boldsymbol{\Sigma}}^{-1}$	Parallel Connections	Cascade Connections
ALD-KRLS	√	√	√
IKRLS	√	√	√
RRKOL	–	–	√
FT-RBF	–	–	√
FT-GRBF	–	–	√

An overview of the five representative algorithms from the enhanced information interaction viewpoint is summarized in Table I. Owing to the tunable structures of the online tunable RBF/GRBF algorithms and the lack of kernel dictionary selection procedure in the RRKOL algorithm, only the ALD-KRLS and IKRLS algorithms are adopted to fully examine the effectiveness of the improved information interactions in this simulation. However, all the five algorithms with cascade connections are compared to reveal the importance of the clues in underlying prediction-error time series. We start the selection procedure for the isotropic form of kernel covariance matrix in (26), namely, in the form of kernel bandwidth, and the following experiments are with the acquired isotropic matrices as the initial kernel covariance matrices.

For the IKRLS algorithm, the prediction performance indicators for the test dataset of 3000 $\sim$ 8000 samples are shown in Table II. In the first cascade-connected group R_a, the selected kernel dictionary D_m consists of $m = 6$ members. The D_m can be divided into different groups in parallel connections, such as (3,3) which denotes that the first parallel-connected group includes 3 kernel regressors and the second parallel-connected group also includes 3 kernel regressors, and (1×6) which denotes that there are a total of 6 parallel-connected groups and each parallel-connected group includes one kernel regressor. The optimized V_1 is the acquired isotropic kernel covariance matrix, V_2 is the optimized symmetric kernel covariance matrix with no parallel connections, while Vp_1 is the set of optimized symmetric kernel covariance matrices that come from their respective parallel-connected groups, and so is Vp_2. The kernel dictionary selection criteria in the IKRLS algorithm include both the ALD criterion and the distance criterion [12], and thus Algorithm 1 is adopted to optimize $\widetilde{\boldsymbol{\Sigma}}^{-1}$. $R_a, R_b, \cdots, R_g$ denote the cascade-connected groups as illustrated in Fig. 1.

As described in the training procedure of Algorithm 3, the kernel dictionary selection and parallel connections for the first cascade-connected group R_a has already been constructed. The respective key parameters are described in the left side of the vertical line in Table II, and the corresponding mean absolute error (MAE) and mean square error (MSE) metrics for the online prediction procedure are listed under R_a.

For the second cascade-connected group R_b, if the input vector is explicitly composed of the past signals in the prediction-error time series recorded by R_a, then the simplest online predictor is using the last datum in the recorded prediction-error time series to predict the current forthcoming prediction error, i.e., for the previous cascade-connected group R_a, setting the latest prediction error as the current error compensation for its online predictor. If the simplest online predictors are applied to R_b and all the following cascade-connected groups, then to a certain extent, these online predictors actually capture the high-order gradients in the prediction-error time series recorded by R_a. For more information about the high-order gradients, the reader is referred to a typical RBF model in [49]. In this simulation, after analyzing the prediction-error time series that is recorded by R_a in the training procedure, we find that capturing the high-order gradients in the prediction-error time series is more efficient than continuously using the Gaussian kernel function modeling. Thus in the following cascade-connected groups, we use the simplest online predictors to show the effectiveness of the proposed information interactions, where the kernel dictionary selection procedures are not performed. Note that some improved online predictors can be applied to obtain better prediction performance in practice, such as the linear RLS algorithm and online kernel algorithms. As shown in Table II, the corresponding MAE/MSE metrics for each cascade-connected group in the online prediction procedure are listed under $R_b, \cdots, R_g$, respectively.

How the connection modes and key parameters in the first cascade-connected group affect the prediction performances in the following cascade-connected groups can be revealed. For example, it can be seen that the best prediction performance is attained at the R_d/R_e cascade-connected group with the (3,3) parallel-connected first cascade-connected group. The results of Table II also indicate that the cascade connections can help to capture the underlying dynamic characteristics in the prediction-error time series. By comparing the MAE/MSE metrics of the first two IKRLS predictors with those of the last three IKRLS predictors in Table II, we can clearly observe the important role of the improved information interactions in the key-parameters selection for the IKRLS algorithm. From Table II, it can be seen that the test MSE and test MAE attend their minimum values either at the same cascade-connected group or at the neighboring cascade-connected groups. Note that the MSE and MAE are two different prediction performance metrics. For example, the MSE indicator is more sensitive than the MAE indicator to these prediction errors of large absolute values. Therefore, it is not necessary that the test MSE and test MAE should both attend their minimum values at the same cascade-connected group.

TABLE III
OPTIMIZATION OF $\widetilde{\mathbf{\Sigma}}^{-1}$ WITH DIFFERENT FORMS FOR IKRLS

Algorithms	m	$\widetilde{\mathbf{\Sigma}}^{-1}$	MAE	MSE
IKRLS	6	V_1	0.092	0.460
IKRLS	6	V_{21}	0.065	0.354
IKRLS	6	V_{22}	0.055	0.330
IKRLS	6	V_2	0.054	0.328
IKRLS	6	Vd_{21}	0.163	2.334
IKRLS	6	Vd_{22}	0.060	0.155
IKRLS	6	Vd_{23}	0.572	16.945
IKRLS	6	Vd_2	0.074	0.638

In Table III, the isotropic kernel covariance matrix V_1 is iteratively optimized to the diagonal form Vd_2 and to the symmetric form V_2, using Algorithm 1. As expected, the online prediction performance is enhanced with the symmetric general kernel covariance matrix. More specifically, the respective evolution paths are $V_1 \rightarrow V_{21} \rightarrow V_{22} \rightarrow V_2$ and $V_1 \rightarrow Vd_{21} \rightarrow Vd_{22} \rightarrow Vd_{23} \rightarrow Vd_2$, and the progressively enhanced prediction performances in each evolution path are validated in the training procedure. Comparing the MAE/MSE metrics of each evolution path in the online prediction procedure given in Table III, we can observe that optimizing the isotropic kernel covariance matrix with (27) behaves stably and robustly, which implies that the improved flexibility in kernel structure helps to enhance the generalization ability.

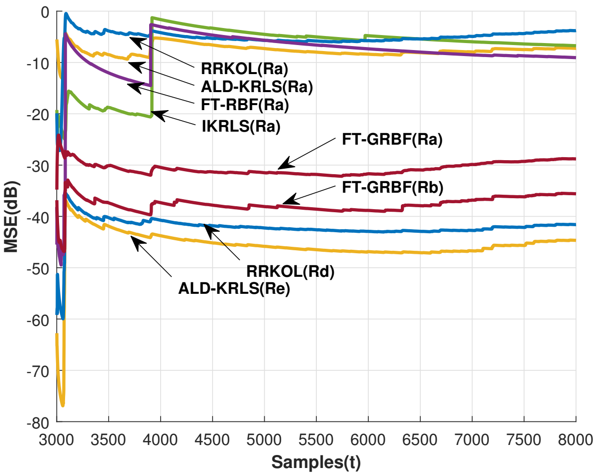


Fig. 2. Prediction performance comparison of the Lorenz time series with cascade connections for various algorithms.

In Table IV, the ALD-KRLS algorithm is operated with the kernel dictionary D_m, the ALD criterion based kernel dictionary D_{ald} with 13 members, and the ALD criterion based kernel dictionary D'_{ald} with 6 members, respectively. It can be observed that the last ALD-KRLS with the optimized symmetric $\widetilde{\mathbf{\Sigma}}^{-1}$ achieves the same prediction performance with much smaller kernel dictionary size than the first ALD-KRLS with the isotropic kernel matrices. The last ALD-KRLS also obtains a better prediction performance than the second and third ALD-KRLS algorithms. This implies that the optimized symmetric form of $\widetilde{\mathbf{\Sigma}}^{-1}$ can alleviate the prediction uncertainty caused by the selected kernel dictionaries.

Table V and Fig. 2 compare the prediction performances of all the five algorithms with cascade connections. Observe that except for the IKRLS and FT-RBF, the optimal prediction performances (marked in bold) are significantly better than the performances of the first cascade-connected groups R_a, which indicates that the improved information interaction helps to select superior online kernel models.

B. Online Prediction of the Capacitor-Current Time Series

For this second-order *RLC* circuit, the relationship between the capacitor-current $x_{c1}(t)$ and the capacitor-voltage $x_{c2}(t)$ is nonstationary with the unstable equilibrium point [47], which can be described as,

$$\begin{aligned} \frac{\mathrm{d}x_{c1}(t)}{\mathrm{d}t} &= x_{c2}(t), \\ \frac{\mathrm{d}x_{c2}(t)}{\mathrm{d}t} &= -\omega_c^2(t)x_{c1}(t) - 2\delta_t x_{c2}(t), \end{aligned} \tag{34}$$

TABLE II
PREDICTION PERFORMANCE OF THE LORENZ TIME SERIES WITH OPTIMIZED $\widetilde{\mathbf{\Sigma}}^{-1}$ AND GENERALIZED CONNECTIONS FOR IKRLS ALGORITHM

Algorithms	m	$\widetilde{\mathbf{\Sigma}}^{-1}$	Indicators	R_a	R_b	R_c	R_d	R_e	R_f	R_g
IKRLS	6	V_1	MAE	0.092	**0.048**	0.066	0.118	0.229	0.449	0.890
			MSE	**0.460**	0.626	1.081	1.972	3.686	6.990	13.380
IKRLS	6	V_2	MAE	0.054	**0.026**	0.030	0.050	0.093	0.180	0.353
			MSE	**0.328**	0.416	0.680	1.211	2.236	4.212	8.027
IKRLS	3,3	V_2	MAE	0.270	0.046	0.011	0.005	**0.003**	0.005	0.008
			MSE	0.381	0.072	0.022	0.010	**0.009**	0.014	0.025
IKRLS	3,3	Vp_1	MAE	0.084	0.022	0.010	**0.006**	0.006	0.008	0.014
			MSE	0.131	0.042	0.023	**0.018**	0.019	0.028	0.047
IKRLS	1×6	Vp_2	MAE	0.442	0.124	0.068	**0.053**	0.063	0.090	0.141
			MSE	0.653	0.306	**0.238**	0.254	0.342	0.528	0.876

TABLE IV
PREDICTION PERFORMANCE OF THE LORENZ TIME SERIES WITH OPTIMIZED $\widetilde{\boldsymbol{\Sigma}}^{-1}$ AND GENERALIZED CONNECTIONS FOR ALD-KRLS ALGORITHM

Algorithms	m	D	$\widetilde{\boldsymbol{\Sigma}}^{-1}$	Indicators	R_a	R_b	R_c	R_d	R_e	R_f	R_g
ALD-KRLS	13	D_{ald}	–	MAE	0.268	0.029	0.007	0.003	**0.002**	0.003	0.006
				MSE	0.435	0.060	0.018	0.007	**0.006**	0.009	0.017
ALD-KRLS	6	D'_{ald}	–	MAE	0.660	0.062	0.013	0.004	**0.003**	0.003	0.006
				MSE	0.999	0.107	0.026	0.010	**0.007**	0.010	0.017
ALD-KRLS	6	D_m	–	MAE	0.649	0.061	0.012	0.004	**0.003**	0.003	0.006
				MSE	1.001	0.107	0.025	0.010	**0.007**	0.010	0.017
ALD-KRLS	6	D_m	√	MAE	0.313	0.030	0.007	0.003	**0.002**	0.003	0.005
				MSE	0.414	0.049	0.015	0.007	**0.006**	0.009	0.017

TABLE V
PREDICTION PERFORMANCE OF THE LORENZ TIME SERIES WITH CASCADE CONNECTIONS FOR VARIOUS ALGORITHMS

Algorithms	m	β	Indicators	R_a	R_b	R_c	R_d	R_e	R_f	R_g
ALD-KRLS	13	–	MAE	0.268	0.029	0.007	0.003	**0.002**	0.003	0.006
			MSE	0.435	0.060	0.018	0.007	**0.006**	0.009	0.017
IKRLS	6	0.90	MAE	0.092	**0.048**	0.066	0.118	0.229	0.449	0.890
			MSE	**0.460**	0.626	1.081	1.972	3.686	6.990	13.380
RRKOL	6	–	MAE	0.440	0.056	0.011	**0.004**	0.004	0.008	0.015
			MSE	0.647	0.096	0.021	**0.008**	0.011	0.021	0.041
FT-RBF	10	0.70	MAE	**0.046**	0.048	0.081	0.154	0.297	0.583	1.145
			MSE	**0.352**	0.497	0.865	1.586	2.974	5.655	10.843
FT-GRBF	6	0.90	MAE	0.023	0.007	**0.006**	0.008	0.015	0.030	0.058
			MSE	0.036	**0.017**	0.022	0.038	0.068	0.127	0.241

where the control parameters are set as $\omega_c(t) = 5\cos(0.05t)$ and $\delta_t = -\frac{1}{2}$ to enhance the nonstationarity. The simulated samples are generated with the step size of 0.008 and the starting point (0,0.30). The first 500 samples are used as the training dataset, and the $500 \sim 2500$ samples are set as the testing dataset. In the training procedure, the optimal or appropriate parameter settings can be acquired by validating the sequential prediction performance on the first $300 \sim 500$ samples. We set the input vector as $\boldsymbol{x}_n = \left[x_{c1}(n)\ x_{c2}(n)\right]^{\mathrm{T}}$ to predict $y_n = x_{c2}(n+1)$, i.e., this is a one-step-ahead prediction. Again, the differencing operation for the input of the FT-GRBF algorithm is not applied, but the second recurrent term of (22) in the RRKOL algorithm is applied since the input $\boldsymbol{x}_n$ contains the past output $y_{n-1} = x_{c2}(n)$. The adopted optimization strategy in this simulation is the same as in Subsection VI-A.

Table VI and Fig. 3 compare the prediction performances of the representative algorithms with both parallel connections and cascade connections. The cascade connections in each algorithm (except for the FT-GRBF algorithm) obtain better prediction performances in the following cascade-connected groups, which indicates that the cascade connections can help to capture the underlying dynamic characteristics in the prediction-error time series. For the ALD-KRLS and RROKL algorithms, their respective parallel-connected cases obtain better prediction performances, which indicates that the parallel connections also can help to capture the underlying dynamic characteristics in the prediction-error time series. Therefore, the effectiveness of the generalized kernel connections is demonstrated in this simulation. The online tunable RBF algorithms can obtain better prediction performances than the KAF algorithms in the first cascade-connected group, but this is not the case for the optimal prediction performances among cascade-connected groups.

TABLE VI
PREDICTION PERFORMANCE OF THE CAPACITOR-CURRENT TIME SERIES WITH GENERALIZED CONNECTIONS FOR VARIOUS ALGORITHMS

Algorithms	m	h_0	β	Indicators	R_a	R_b	R_c	R_d	R_e	R_f	R_g
ALD-KRLS	7	$1e10$	–	MAE	1.314	0.029	**0.002**	0.002	0.004	0.007	0.014
				MSE	3.021	0.061	**0.004**	0.006	0.012	0.022	0.041
ALD-KRLS	4,3	$2e10$	–	MAE	0.093	0.005	**0.001**	0.002	0.004	0.007	0.014
				MSE	0.316	0.021	**0.004**	0.007	0.012	0.022	0.042
IKRLS	6	$2e10$	0.005	MAE	2.454	0.093	**0.060**	0.118	0.236	0.473	0.945
				MSE	4.925	**0.468**	0.782	1.427	2.669	5.063	9.693
RRKOL	7	$2e10$	–	MAE	10.388	0.266	0.007	**0.002**	0.003	0.006	0.013
				MSE	21.128	0.504	0.012	**0.006**	0.011	0.020	0.039
RRKOL	4,3	$2e10$	–	MAE	0.678	0.017	**0.002**	0.002	0.005	0.009	0.018
				MSE	1.287	0.029	**0.004**	0.007	0.013	0.025	0.048
FT-RBF	10	$1e9$	0.05	MAE	1.095	**0.220**	0.362	0.658	1.239	2.349	4.509
				MSE	1.757	**0.596**	1.010	1.842	3.449	6.550	12.552
FT-GRBF	7	$1e10$	0.01	MAE	**0.043**	0.054	0.095	0.176	0.333	0.640	1.236
				MSE	**0.100**	0.143	0.252	0.464	0.872	1.663	3.197

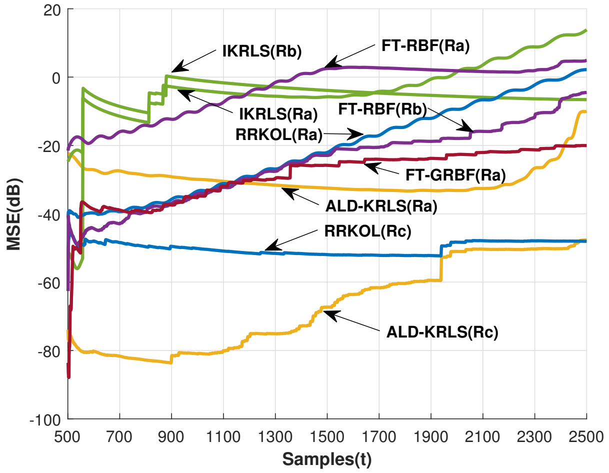


Fig. 3. Prediction performance comparison of the capacitor-current time series with generalized connections for various algorithms.

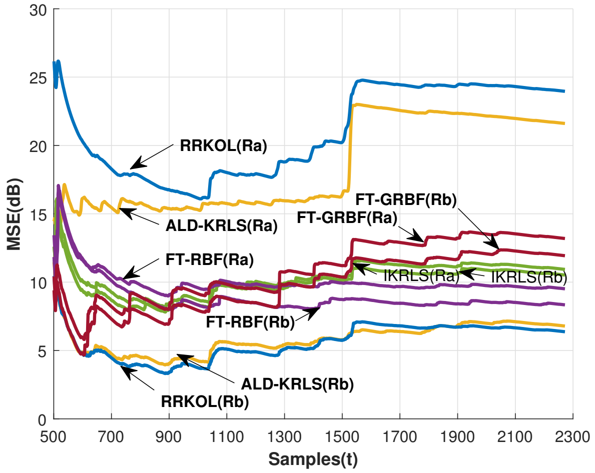


Fig. 4. Prediction performance of the sunspot time series with cascade connections for various algorithms.

C. Sunspot Time Series Online Prediction

The sunspot time series is composed of annual averaged numbers of observed sunspots, which is a widely used benchmark that contains nonstationarity [16]. The one-step ahead prediction of the monthly recorded sunspot time series $\{x_s(n)\}$ from 1830 to 2019 is considered. The input vector of the predictor is set to $\boldsymbol{x}_n = [x_s(n-1)\ x_s(n-2)\ x_s(n-3)\ x_s(n-4)]^{\mathrm{T}}$ and the desired output is $y_n = x_s(n)$ in this simulation. The first 500 samples are used as the training dataset, and the $500 \sim 2280$ samples are set as the testing dataset to examine the effectiveness of the studied algorithms. In the training procedure, the optimal or appropriate parameter settings can be acquired by validating the sequential prediction performance on the first $300 \sim 500$ samples. Since the input vector is composed of the past output signals of y_n, both the differencing procedure in the FT-GRBF algorithm and the second recurrent term of (22) in the RRKOL algorithm are applied in this simulation. As the sunspot time series is monthly recorded and annually averaged, there exists strong linearity among the neighboring time series samples. The cascade connections can provide a structure to combine the kernel based online modeling approaches with linear RLS algorithm in this simulation, i.e., the first cascade-connected group adopts the five representative algorithms, and the second cascade-connected group adopts the linear RLS algorithm.

Table VII and Fig. 4 compare the prediction performances of the representative algorithms with cascade connections, where p_r and β_2 respectively denote the dimension of the input vector and the exponential forgetting factor, in the linear RLS algorithm. Each representative algorithm obtains better prediction performance in the second cascade-connected group, especially for the ALD-KRLS and RRKOL algorithms. This demonstrates the effectiveness of the cascade connections in terms of providing a structure to combine complementary algorithms.

TABLE VII
PREDICTION PERFORMANCE OF THE SUNSPOT TIME SERIES WITH CASCADE CONNECTIONS FOR VARIOUS ALGORITHMS

Algorithms	m	p_r	β_2	Indicators	R_a	R_b
ALD-KRLS	7	1	0.96	MAE MSE	6.451 12.036	1.525 2.185
IKRLS	5	11	0.98	MAE MSE	2.204 3.520	2.097 3.356
RRKOL	7	17	0.98	MAE MSE	9.607 15.776	1.422 2.082
FT-RBF	23	33	0.92	MAE MSE	2.143 2.990	1.730 2.612
FT-GRBF	12	11	0.98	MAE MSE	2.386 4.566	2.119 3.950

VII. CONCLUSIONS

In this paper, we have proposed a structure parameter optimized kernel based online prediction approach with a generalized optimization strategy for nonstationary time series. The intermittent optimization of the real symmetric kernel covariance matrix has been realized to improve the kernel structure's flexibility and alleviate the prediction uncertainty caused by the kernel dictionary selection procedure for nonstationary data. A generalized optimization strategy with multiple kernel connection modes has been designed to provide a self-contained way for constructing the entire connections of kernel regressors, with the enhanced ability to track the changing dynamic characteristics. The improved information interaction not only enhances the ability to deal with various online modeling problems but also provides a more flexible kernel structure and a more compatible way of kernel connections that can be combined with other existing online modeling techniques.

REFERENCES

[1] M. Han, *et al.*, "Multivariate chaotic time series online prediction based on improved kernel recursive least squares algorithm," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1160–1172, Apr. 2019.

[2] X. Hong, *et al.*, "Model selection approaches for non-linear system identification: A review," *Int. J. Syst. Sci.*, vol. 39, no. 10, pp. 925–946, Oct. 2008.

[3] F. Tan and X. Guan, "Research progress on intelligent system's learning, optimization, and control–Part II: Online sparse kernel adaptive algorithm," *IEEE Trans. Syst., Man, and Cybern.: Syst.*, vol. 50, no. 12, pp. 5369–5385, Dec. 2020.

[4] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag: New York 1995.

[5] B. Schölkopf, R. Herbrich, and A. J. Smola, “A generalized representer theorem,” in *Proc. COLT 2001* (Amsterdam, Netherlands), Jul. 16-19, 2001, pp. 416–426.

[6] P. Honeine, “Approximation errors of online sparsification criteria,” *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4700–4709, Sep. 2015.

[7] 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.

[8] J. D. A. Santos and G. A. Barreto, “A regularized estimation framework for online sparse LSSVR models,” *Neurocomputing*, vol. 238, pp. 114–125, Jan. 2017.

[9] 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.

[10] 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.

[11] S. Wang, W. Wang, and S. Duan, “A class of weighted quantized kernel recursive least squares algorithms,” *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 64, no. 6, pp. 730–734, Jun. 2017.

[12] J. Guo, H. Chen, and S. Chen, “Improved kernel recursive least squares algorithm based online prediction for nonstationary time series,” *IEEE Signal Process. Lett.*, vol. 27, pp. 1365–1369, Jul. 2020.

[13] J. Platt, “A resource-allocating network for function interpolation,” *Neural Comput.*, vol. 3, no. 2, pp. 213–225, Jun. 1991.

[14] W. Liu, I. Park, and J. C. Príncipe, “An information theoretic approach of designing sparse kernel adaptive filters,” *IEEE Trans. Neural Netw.*, vol. 20, no. 12, pp. 1950–1961, Dec. 2009.

[15] H. Fan and Q. Song, “A sparse kernel algorithm for online time series data prediction,” *Expert Syst. Appl.*, vol. 40, no. 6, pp. 2174–2181, May 2013.

[16] T. Liu, *et al.*, “Fast adaptive gradient RBF networks for online learning of nonstationary time series,” *IEEE Trans. Signal Process.*, vol. 68, pp. 2015–2030, Mar. 2020.

[17] A. R. C. Paiva, “Information-theoretic dataset selection for fast kernel learning,” in *Proc. IJCNN 2017* (Anchorage, AK, USA), May 4-19, 2017, pp. 2088–2095.

[18] H. Fan, Q. Song, and Z. Xu, “An information theoretic sparse kernel algorithm for online learning,” *Expert Syst. Appl.*, vol. 41, no. 9, pp. 4349–4359, 2014.

[19] X. Wu, *et al.*, “Online feature selection with streaming features,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 5, pp. 1178–1192, May 2013.

[20] Q. Zhang, *et al.*, “Salient subsequence learning for time series clustering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2193–2207, Sep. 2019.

[21] H. Wang, *et al.*, “Time series feature learning with labeled and unlabeled data,” *Pattern Recognit.*, vol. 89, pp. 55–66, May 2019.

[22] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*. Hoboken, NJ, USA: Wiley 2010.

[23] H. Chen, Y. Gong, and X. Hong, “Online modeling with tunable RBF network,” *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 935–947, Jun. 2013.

[24] H. Chen, Y. Gong, X. Hong, and S. Chen, “A fast adaptive tunable RBF network for nonstationary systems,” *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2683–2692, Dec. 2016.

[25] B. Chen, J. Liang, N. Zheng, and J. C. Príncipe, “Kernel least mean square with adaptive kernel size,” *Neurocomputing*, vol. 191, pp. 95–106, May 2016.

[26] C. Bishop, *Pattern Recognition and Machine Learning*. Springer: New York, 2006.

[27] N. Hansen, “The CMA evolution strategy: A tutorial,” *arXiv: 1604.00772*, 2016. [Online]. Available: http://arxiv.org/abs/1604.00772.

[28] N. Hansen and A. Auger, “Principled design of continuous stochastic search: From theory to practice,” Chapter in: Y. Borenstein and A. Moraglio (eds.), *Theory and Principled Methods for the Design of Metaheuristics*. Springer-Verlag Berlin, 2014, pp. 145–180.

[29] D. Vermetten, S. van Rijn, T. Bäck, and C. Doerr, “Online selection of CMA-ES variants,” in *Proc. GECCO 2019* (Prague, CZ), Jul. 13-17, 2019, pp. 951–959.

[30] G. B. Huang, P. Saratchandran, and N. Sundararajan, “An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks,” *IEEE Trans. Syst., Man, and Cybern., B(Cybern.)*, vol. 34, no. 6, pp. 2284–2292, Dec. 2004.

[31] S. Van Vaerenbergh, M. Lázaro-Gredilla, and I. Santamaría, “Kernel recursive least-squares tracker for time-varying regression,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1313–1326, Aug. 2012.

[32] S. Van Vaerenbergh, I. Santamaría, W. Liu, and J. C. Príncipe, “Fixed-budget kernel recursive least-squares,” in *Proc. ICASSP 2010* (Dallas, TX, USA), Mar. 14-19, 2010, pp. 1882–1885.

[33] S. Van Vaerenbergh, J. Via, and I. Santamaría, “A sliding-window kernel RLS algorithm and its application to nonlinear channel identification,” in *Proc. ICASSP 2006* (Toulouse, France), May 14-19, 2006, pp. V-789–V-792.

[34] W. Liu, I. Park, Y. Wang, and J. C. Príncipe, “Extended kernel recursive least squares algorithm,” *IEEE Trans. Signal Process.*, vol. 57, no. 10, pp. 3801–3814, Oct. 2009.

[35] F. Ding, P. X. Liu, and G. Liu, “Multiinnovation least-squares identification for system modeling,” *IEEE Trans Syst., Man, and Cybern., B (Cybern.)*, vol. 40, no. 3, pp. 767–778, Jun. 2010.

[36] H. Xu, C. Caramanis, and S. Sanghavi, “Robust PCA via outlier pursuit,” *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 3047–3064, May 2012.

[37] J. L. Elman, “Finding structure in time,” *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, 1990.

[38] C. Buckner and J. Garson, “Connectionism,” in *The Stanford Encyclopedia of Philosophy*, (ed.) E. N. Zalta, fall 2019 ed., Metaphysics Research Lab, Stanford University, 2019.

[39] S. Chen, X. Hong, B. L. Luk, and C. J. Harris, “Orthogonal-least-squares regression: A unified approach for data modelling,” *Neurocomputing*, vol. 72, nos. 10-12, pp. 2670–2681, 2009.

[40] S. Chen, S. A. Billings, and W. Luo, “Orthogonal least squares methods and their application to non-linear system identification,” *Int. J. Control*, vol. 50, no. 5, pp. 1873–1896, 1989.

[41] Q. Song, X. Zhao, H. Fan, and D. Wang, “Robust recurrent kernel online learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1068–1081, May 2017.

[42] S. Kitayama and K. Yamazaki, “Simple estimate of the width in Gaussian kernel with adaptive scaling technique,” *Appl. Soft Comput.*, vol. 11, no. 8, pp. 4726–4737, Dec. 2011.

[43] B. Schölkopf, A. Smola, and K. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[44] G. Papageorgiou, P. Bouboulis and S. Theodoridis, “Robust nonlinear regression: A greedy approach employing kernels with application to image denoising,” *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4309–4323, Aug. 2017.

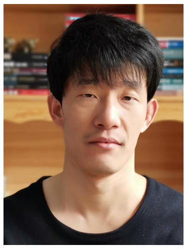
[45] T. Bäck, C. Foussette, and P. Krause, *Contemporary Evolution Strategies*, Springer-Verlag: Berlin Heidelberg, 2013.

[46] R. S. Tsay, “Regression models with time series errors,” *J. Am. Stat. Assoc.*, vol. 79, no. 385, pp. 118–124, Mar. 1984.

[47] C. Alexander and M. N. O. Sadiku, *Fundamentals of Electric Circuits*. McGraw-Hill, 1999.

[48] E. N. Lorenz, “Deterministic nonperiodic flow,” *J. Atmos. Sci.*, vol. 20, no. 2, pp. 130–141, 1963.

[49] E. S. Chng, S. Chen, and B. Mulgrew, “Gradient radial basis function networks for nonlinear and nonstationary time series prediction,” *IEEE Trans. Neural Netw.*, vol. 7, no. 1, pp. 190–194, Jan. 1996.



Jinhua Guo (Student Member, IEEE) received his B.Eng. degree in Automation from China University of Petroleum, Qingdao and Dongying, China, in 2013, and his M.Eng. degree in Control Engineering from a joint education program in 2020, at Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences, Fuzhou, China, and North University of China, Taiyuan, China. From 2013 to 2015, he worked as an electrical engineer in weak and strong electricity at China Oilfield Services Limited (COSL). From 2020 to 2021, he was a research assistant at the Fujian Provincial Key Laboratory of Intelligent Identification and Control of Complex Dynamic Systems, Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences. His current research interests include kernel based online prediction for nonstationary time series, model selection, topology inference, machine learning with graph theory, graph signal processing, topological data analysis, information theory, statistical learning, data and network sciences. He is a Graduate Student member of IEEE Signal Processing Society and IEEE Information Theory Society.

Hao Chen (Senior Member, IEEE) received his B.Eng. degree in automatic control from the National University of Defense Technology, Changsha, China, in 2006, the M.Sc. degree (with Distinction) in control systems from the University of Sheffield, Sheffield, U.K., in 2009, and the Ph.D. degree in cybernetics from the School of Systems Engineering, University of Reading, Reading, U.K., in 2014, sponsored by the Engineering and Physical Sciences Research Council and Defence Science and Technology Laboratory from the British Government. From 2014 to 2015, he was a Post-Doctoral Research Fellow with the Department of Chemical and Materials Engineering, University of Alberta, Edmonton, AB, Canada, and Syncrude Canada, Ltd., Fort McMurray, AB, Canada. He is currently a Professor and the Head of Fujian Provincial Key Laboratory of Intelligent Identification and Control of Complex Dynamic Systems, Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences, Fuzhou 350002, China. His current research interests include online learning, soft sensors, system identification, neural networks, data-based diagnosis, and data analysis and their applications in the industrial process.

Jingxin Zhang (Student Member, IEEE) received her B.Eng. degree from School of Electrical Engineering and Automation, Harbin Engineering University, Harbin, China, and her M.Eng. degree in Control Science and Engineering from Harbin Institute of Technology, Harbin, China, in 2014 and 2016, respectively. From 2016 to 2018, she was an assistant engineer at Department of Quanzhou Institute of Equipment Manufacturing, Haixi Institutes, Chinese Academy of Sciences. Since 2018, she has been pursuing the Ph.D. degree with the Department of Automation, Tsinghua University, Beijing, China. Her research interests are data-driven fault detection and diagnosis, performance monitoring and their applications in the industrial process.

Sheng Chen (Fellow, IEEE) received his B.Eng. degree from the East China Petroleum Institute, Dongying, China, in 1982, and his Ph.D. degree from the City University, London, in 1986, both in control engineering. In 2005, he was awarded the higher doctoral degree, Doctor of Sciences (D.Sc.), from the University of Southampton, Southampton, U.K. From 1986 to 1999, He held research and academic appointments at the Universities of Sheffield, Edinburgh and Portsmouth, all in U.K. Since 1999, he has been with the School of Electronics and Computer Science, the University of Southampton, U.K., where he holds the post of Professor in Intelligent Systems and Signal Processing. Dr. Chen's research interests include adaptive signal processing, wireless communications, modeling and identification of nonlinear systems, neural network and machine learning, evolutionary computation methods and optimization. He has published over 650 research papers. Professor Chen has 17,400+ Web of Science citations with h-index 57 and 34,300+ Google Scholar citations with h-index 79. Dr. Chen is a Fellow of the United Kingdom Royal Academy of Engineering, a Fellow of Asia-Pacific Artificial Intelligence Association and a Fellow of IET. He is one of the original ISI highly cited researcher in engineering (March 2004).