

Adaptive Multioutput Gradient RBF Tracker for Nonlinear and Nonstationary Regression

Tong Liu¹, Member, IEEE, Sheng Chen², Fellow, IEEE,
Kang Li³, Senior Member, IEEE, Shaojun Gan⁴, and Chris J. Harris⁵

Abstract—Multioutput regression of nonlinear and nonstationary data is largely understudied in both machine learning and control communities. This article develops an adaptive multioutput gradient radial basis function (MGRBF) tracker for online modeling of multioutput nonlinear and nonstationary processes. Specifically, a compact MGRBF network is first constructed with a new two-step training procedure to produce excellent predictive capacity. To improve its tracking ability in fast time-varying scenarios, an adaptive MGRBF (AMGRBF) tracker is proposed, which updates the MGRBF network structure online by replacing the worst performing node with a new node that automatically encodes the newly emerging system state and acts as a perfect local multioutput predictor for the current system state. Extensive experimental results confirm that the proposed AMGRBF tracker significantly outperforms existing state-of-the-art online multioutput regression methods as well as deep-learning-based models, in terms of adaptive modeling accuracy and online computational complexity.

Index Terms—Multioutput gradient radial basis function (MGRBF) network, multivariate nonlinear and nonstationary regression, online adaptive tracking, two-step training.

I. INTRODUCTION

MANY real-world systems are nonstationary, and they operate in real time to continuously produce tremendous amount of data in the form of fast arriving data streams [1], [2], [3]. A regression model must be capable of adapting to the newly emerging system state within a strictly limited processing time. Recently, there is an increasing trend of applying deep learning to nonlinear regression problem.

Manuscript received 24 October 2022; revised 14 December 2022; accepted 3 January 2023. Date of publication 2 February 2023; date of current version 22 November 2023. This work was supported by the National Natural Science Foundation of China under Grant 62003011. This article was recommended by Associate Editor P. Shi. (Corresponding author: Shaojun Gan.)

Tong Liu is with the Department of Chemical Engineering, Imperial College London, SW7 2AZ London, U.K. (e-mail: tliu.soton@gmail.com).

Sheng Chen is with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K., and also with the Faculty of Information Science and Engineering, Ocean University of China, Qingdao 266005, China (e-mail: sqc@ecs.soton.ac.uk).

Kang Li is with the School of Electronic and Electrical Engineering, University of Leeds, LS2 9JT Leeds, U.K. (e-mail: k.li1@leeds.ac.uk).

Shaojun Gan is with the Beijing Key Laboratory of Traffic Engineering, College of Metropolitan Transportation, Beijing University of Technology, Beijing 100124, China (e-mail: s.gan@bjut.edu.cn).

Chris J. Harris is with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K. (e-mail: chrisharris57@msn.com).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TCYB.2023.3235155>.

Digital Object Identifier 10.1109/TCYB.2023.3235155

Typically, deep networks with multilevel feature extraction layers are adopted, such as stacked autoencoder (SAE) [4], [5] or long short-term memory (LSTM) recurrent neural network (RNN) [6], [7]. However, the online adaptation of these deep nonlinear models is very challenging if not impossible, since it is prohibitive to adapt large deep networks within a small sampling period in order to track fast time-varying system dynamics.

With the increasing demand for today's complex decision making, the traditional single-output modeling is not coping well with multitask predictions. Unlike single-output modeling, multioutput modeling takes into account the complex interactions and compound dependencies among the multiple outputs [8], [9]. Multioutput modeling becomes even more challenging under nonstationary environment. Multioutput data gathered from heterogeneous sources come in high speed and evolve over time with unforeseen drifts. This motivates our current work to develop an effective multioutput online predictive model to deal with fast time-varying characteristics in multivariate data. Existing methods in the literature for online regression can be classified into two groups: 1) multiple local model learning and 2) single nonlinear model learning.

The multiple local model learning strategy has been successfully applied to industrial adaptive soft sensor design in the presence of frequent operating condition deviations [10], [11], [12]. The essence of this local learning strategy is to partition a nonlinear and nonstationary process into multiple local regions with a moving window. Each region is considered to be stationary and is modeled by a local linear model. The selective ensemble-based multiple local model (SEMLM) learning grows local linear models online to automatically identify newly emerged process states and combines the most up-to-date local models to make an accurate selective ensemble regression (SER)-based prediction [13]. The growing and pruning SER (GAP-SER) can further prune the past accumulated local models that are no longer relevant, thus, significantly reduce the online computation burden in making prediction [14]. Although the GAP-SER can achieve excellent online prediction performance for modeling nonstationary data, while imposing much less averaged computation time per sample (ACTpS) than the SEMLM, it is still restricted to single-output modeling. The more recent multioutput GAP-SER (MGAP-SER) adopts a novel adaptive local learning strategy based on multivariate statistic that enables growing and pruning multioutput local linear models [15]. Unlike the

single-output GAP-SER, the MGAP-SER exploits the complex interactions between multiple output variables, and it attains better prediction accuracy than using multiple single-output GAP-SERs when modeling multioutput processes. A potential drawback common to all multiple local model learning methods is that the size of the SER prediction model constructed changes from sample to sample. It is hard to implement an adaptive controller based on such a variable-size predictor.

The single nonlinear model learning strategy attempts to capture the global nonlinear characteristics from the data using a nonlinear model, such as a neural network. Among various nonlinear models, the radial basis function (RBF) network has become very popular, due to its elegant mathematical formulation, ease of model optimization, and strong modeling performance [16]. By formulating the RBF model training as a subset selection problem, the orthogonal least squares (OLSs) is used to identify appropriate RBF centers from training input data, and to estimate output weights simultaneously in an efficient manner [17], [18], [19], [20], [21]. This procedure can be easily extended to multioutput modeling, where the subset selection is directly based on the trace of the error covariance matrix [22], [23], [24], [25]. Some other hybrid training algorithms have also been proposed for multioutput RBF (MRBF) model construction [26], [27]. The RBF model typically tracks the changing process dynamics by updating its weight vector using the recursive least square (RLS) algorithm [28], [29], [30]. However, in a highly time-varying environment, the process dynamics can vary dramatically and weight updating alone is insufficient. Hence, online model structure adaptation is necessary. A typical way of online adapting the RBF structure is to adaptively grow or prune RBF nodes based on their significance, such as resource allocating network (RAN) [31] and growing-and-pruning RBF (GAP-RBF) [32]. Such an approach, however, produces a variable-size nonlinear model, which is not helpful for implementing an adaptive controller. To keep a fixed compact-size RBF model, the fast tunable RBF (TRBF) of [33] adjusts the individually tunable nodes to track the time-varying process dynamics. Experimental results of [33] show that this TRBF outperforms the RAN and GAP-RBF for online modeling of nonstationary systems.

A novel extension to the RBF network, known as the gradient RBF (GRBF) network, was proposed to deal with time series exhibiting homogeneous nonstationary characteristics [34]. Instead of sensing the trajectory of the series itself, the hidden nodes of the GRBF network react to the gradient of time series. Not surprisingly, this GRBF network trained by the OLS algorithm outperforms the classic RBF network in nonstationary time-series prediction [34]. To enhance its adaptive prediction capability for highly nonstationary time series, a fast adaptive GRBF (AGRBF) was proposed for online time-series prediction [35], which was further extended to online modeling and identification of nonlinear and time-varying processes in [36]. Similar to the TRBF, during online operation when the current modeling error becomes unacceptable, the AGRBF adapts the model structure by replacing an insignificant node with a new node that automatically encodes the current data. Owing to the local prediction property of the

GRBF node, the new node optimization is much more efficient than the TRBF, which imposes little online computation. Extensive experiments of [36] demonstrate that this AGRBF is superior to various existing state-of-the-art adaptive nonlinear models, including the TRBF and GAP-SER, in terms of both online modeling accuracy and computational efficiency. Also, the fixed compact size of AGRBF makes it more applicable in an adaptive control scheme than the GAP-SER.

Although the AGRBF achieves great success in the online regression of nonstationary data, it is suitable only for single-output modeling. Due to the particular geometry structure of the GRBF network, we cannot extend it to multioutput modeling by simply adding multiple output neurons to the single-output-neuron GRBF structure, because this does not realize its full predictive capability. Note that this is unlike the MRBF network, which can be obtained by adding the multiple output neurons to the single-output-neuron RBF structure. Also, the online adaptive strategies of the AGRBF [36], TRBF [33], RAN [31], and GAP-RBF [32] are all restricted to single-output modeling and they are not applicable to online multioutput modeling. Although we may apply the multiple single-output AGRBF models to identify multioutput systems, this will not only increase the modeling effort considerably but also degrades the achievable online modeling accuracy. Therefore, it is necessary to develop the new online adaptive multioutput model by extending the highly desirable GRBF network to multioutput regression. This article proposes an adaptive multioutput GRBF (AMGRBF) network for tracking nonlinear and nonstationary processes. Our novel contributions are summarized as follows.

- 1) We propose a new MGRBF network with excellent predictive capacity for multioutput modeling. Due to the multioutput nature of this new MGRBF structure, the OLS learning cannot be directly applied to model construction. Hence, we further propose a new two-step training method to construct a compact MGRBF model.
- 2) To improve its tracking ability in highly nonstationary environments, we derive an adaptive mechanism to efficiently adjust the MGRBF model online. During online operation, the MGRBF network adapts its structure by replacing an insignificant node with a new node that automatically encodes the newly emerging system state. This proposed online tracker fully exploits the geometric structure of the MGRBF network, and it is capable of timely capturing the fast-changing process dynamics while maintaining a very low online complexity.
- 3) Extensive applications of real-world multivariate nonlinear and nonstationary regression demonstrate that the proposed AMGRBF significantly outperforms the existing state-of-the-art online adaptive multioutput models, including the MGAP-SER and the new multioutput TRBF (MTRBF), as well as deep-learning-based models.

II. PROPOSED MGRBF NETWORK

We consider the generic multi-input–multi-output nonlinear time-varying dynamic system that can be represented by

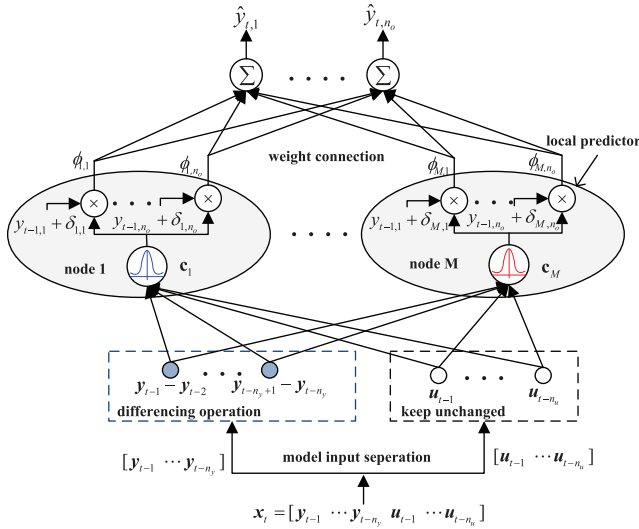


Fig. 1. Proposed MGRBF network structure.

the following multivariate nonlinear autoregressive moving average (NARMA) model [25]:

$$\mathbf{y}_{t+(K-1)} = \mathbf{f}_{\text{sys}}(\mathbf{x}_t; t) \quad (1)$$

where $K \geq 1$ is the prediction step, $\mathbf{y}_t = [y_{t,1} \dots y_{t,n_o}]^T \in \mathbb{R}^{n_o}$ is the n_o -dimensional system output vector, $\mathbf{f}_{\text{sys}}(\bullet; t)$ is the n_o -dimensional unknown and time-varying system mapping, and the overall system input $\mathbf{x}_t \in \mathbb{R}^{(n_o n_y + n_i n_u)}$ is given by

$$\mathbf{x}_t = [\mathbf{y}_{t-1}^T \dots \mathbf{y}_{t-n_y}^T \mathbf{u}_{t-1}^T \dots \mathbf{u}_{t-n_u}^T]^T \quad (2)$$

in which $\mathbf{u}_t = [u_{t,1} \dots u_{t,n_i}]^T \in \mathbb{R}^{n_i}$ is the n_i -dimensional system input vector, and n_y and n_u are the system output and input lags, respectively. Without loss of generality, we focus on one-step-ahead prediction modeling, that is, $K = 1$. All the results are, however, equally applicable to the case of $K > 1$.

Two special cases of this NARMA model are the multivariate nonlinear autoregressive (NAR) model or time-series prediction in which $\mathbf{x}_t = [\mathbf{y}_{t-1}^T \dots \mathbf{y}_{t-n_y}^T]^T$, and the multivariate nonlinear moving average (NMA) model in which $\mathbf{x}_t = [\mathbf{u}_{t-1}^T \dots \mathbf{u}_{t-n_u}^T]^T$.

A. MGRBF Network Structure

The modeling task is to build a prediction model $\hat{\mathbf{y}}_t = \mathbf{f}_{\text{mod}}(\mathbf{x}_t; \Theta)$ to predict \mathbf{y}_t given the input \mathbf{x}_t of (2), where Θ denotes the model parameter matrix. We propose a novel MGRBF network, as illustrated in Fig. 1, to perform this task. This MGRBF network is very different from the MRBF network of [22], [23], and [24]. It also has a significant difference with the single-output GRBF network [36].

First, unlike the MRBF network, the input layer of the MGRBF network automatically differences the past outputs in \mathbf{x}_t to yield the actual network input vector $\mathbf{x}'_t \in \mathbb{R}^{n_c}$ as

$$\mathbf{x}'_t = [(y_{t-1} - y_{t-2})^T \dots (y_{t-n_y+1} - y_{t-n_y})^T \mathbf{u}_{t-1}^T \dots \mathbf{u}_{t-n_u}^T]^T \quad (3)$$

where $n_c = (n_o(n_y - 1) + n_i n_u)$. Differencing helps to eliminate trend and seasonality and makes the series less time-dependent [37]. For the NAR regression task, the input vector

\mathbf{x}_t does not contain the past system outputs, and the MGRBF network will not implement this difference operation.

Also, an MGRBF hidden node is very different from that of the single-output GRBF network. Specifically, unlike the single-output GRBF network, where each Gaussian node's response is modified by a single-output local predictor, the n_o local predictors are required in each MGRBF hidden node. Let M be the number of hidden nodes in the MGRBF network. Observe from Fig. 1 that each MGRBF hidden node contains the n_o local one-step predictors that modify the Gaussian node's response. The response of the i th local predictor in the j th MGRBF node to the input vector \mathbf{x}'_t is given by

$$\phi_{j,i}(\mathbf{x}'_t) = \exp\left(\frac{-\|\mathbf{x}'_t - \mathbf{c}_j\|^2}{2\sigma^2}\right) \times (y_{t-1,i} + \delta_{j,i}) \quad \text{for } i = 1, \dots, n_o, j = 1, \dots, M \quad (4)$$

where σ is the width of the Gaussian kernel, which is typically set as the maximum Euclidean distance among nodes [38], $\mathbf{c}_j \in \mathbb{R}^{n_c}$ is the node center, and $\delta_{j,i}$ is a scalar associated with the i th local predictor of the j th node. It can be observed from (4) that the multiple local predictors in the hidden node share the same Gaussian center \mathbf{c}_j , but with different multiplication terms $(y_{t-1,i} + \delta_{j,i})$, $1 \leq i \leq n_o$. The term $(y_{t-1,i} + \delta_{j,i})$ can be interpreted as a local one-step prediction of $y_{t,i}$ by the i th local predictor. The essence of the MGRBF node is that if the input vector \mathbf{x}'_t is very similar to the j th center \mathbf{c}_j , the value of the j th Gaussian function is close to 1 and all the local predictors $(y_{t-1,i} + \delta_{j,i})$ for $1 \leq i \leq n_o$ become fully active.

The output layer of the MGRBF network consists of the n_o output nodes, which form the n_o linear combiners of the M hidden layer responses to produce the model output vector $\hat{\mathbf{y}}_t = [\hat{y}_{t,1} \dots \hat{y}_{t,n_o}]^T \in \mathbb{R}^{n_o}$. Let $\phi_{t,j} \in \mathbb{R}^{n_o}$ be the response vector of the j th node to \mathbf{x}'_t , that is

$$\phi_{t,j} = [\phi_{j,1}(\mathbf{x}'_t) \dots \phi_{j,n_o}(\mathbf{x}'_t)]^T \quad (5)$$

and denote the connection weight vector from the j th hidden node's response vector to the i th output node as

$$\theta_{i,j} = [\theta_{i,j,1} \dots \theta_{i,j,n_o}]^T. \quad (6)$$

Then, the i th output of the MGRBF network is given by

$$\hat{y}_{t,i} = \sum_{j=1}^M \phi_{t,j}^T \theta_{i,j}. \quad (7)$$

Further define the overall response vector $\phi_{\bar{M},t} \in \mathbb{R}^{\bar{M}}$ of the hidden layer to the input \mathbf{x}'_t as

$$\phi_{\bar{M},t}^T = [\phi_{t,1}^T \dots \phi_{t,M}^T] \quad (8)$$

where $\bar{M} = n_o M$, as well as the overall output layer connection matrix $\Theta_{\bar{M} \times n_o} \in \mathbb{R}^{\bar{M} \times n_o}$

$$\Theta_{\bar{M} \times n_o} = \begin{bmatrix} \theta_{1,1} & \theta_{2,1} & \dots & \theta_{n_o,1} \\ \theta_{1,2} & \theta_{2,2} & \dots & \theta_{n_o,2} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{1,M} & \theta_{2,M} & \dots & \theta_{n_o,M} \end{bmatrix}. \quad (9)$$

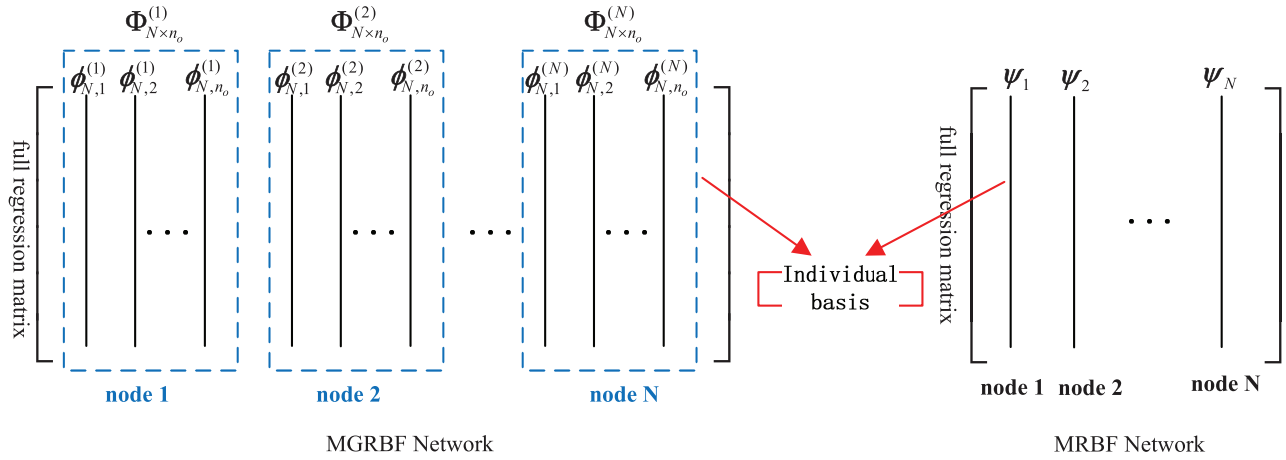


Fig. 2. Difference of the regression matrices between the MGRBF network and the RBF/MRBF/GRBF network.

The output vector of the MGRBF network to \mathbf{x}'_t is given by

$$\hat{\mathbf{y}}_t^T = \boldsymbol{\phi}_{M,t}^T \boldsymbol{\Theta}_{M \times n_o}. \quad (10)$$

Remark 1: The hidden neuron node in all the existing neural network architectures produces a single response to the input. In contrast, the hidden neuron node in our MGRBF network produces multiple response to the input. This represents an innovative idea in the artificial neural network design.

B. Two-Step Training of MGRBF

The task of building an MGRBF model is to choose appropriate centers \mathbf{c}_j , and scalars $\boldsymbol{\delta}_j = [\delta_{j,1} \cdots \delta_{j,n_o}]^T$ based on the given set of training inputs and outputs $\{\mathbf{x}'_t; \mathbf{d}_t, \mathbf{y}_t\}_{t=1}^N$, where N is the total number of training samples, and

$$\mathbf{d}_t = \mathbf{y}_t - \mathbf{y}_{t-1}. \quad (11)$$

Specifically, we choose the centers \mathbf{c}_j , $1 \leq j \leq M$, from the training input data $\{\mathbf{x}'_t\}_{t=1}^N$. In particular, if \mathbf{x}'_t is selected as the j th center \mathbf{c}_j , we set $\boldsymbol{\delta}_j = \mathbf{d}_t$ to ensure that the j th hidden node response $\boldsymbol{\phi}_{t,j}$ is a perfect local predictor of \mathbf{y}_t . Thus, by considering every data point $(\mathbf{x}'_t, \mathbf{d}_t)$ as a candidate hidden node, the problem of constructing the MGRBF network is equivalent to the task of selecting an M -term subset model $\{\mathbf{c}_j, \boldsymbol{\delta}_j\}_{j=1}^M$ from the full N -term model $\{\mathbf{x}'_t, \mathbf{d}_t\}_{t=1}^N$.

Formally, by using every $(\mathbf{x}'_t, \mathbf{d}_t)$ as a candidate hidden node, we obtain the full N -hidden-node MGRBF network with the overall output layer connection matrix $\boldsymbol{\Theta}_{\bar{N} \times n_o} \in \mathbb{R}^{\bar{N} \times n_o}$, where $\bar{N} = n_o N$, as well as the overall response matrix $\boldsymbol{\Phi}_{N \times \bar{N}} \in \mathbb{R}^{N \times \bar{N}}$ of the hidden layer to the inputs $\{\mathbf{x}'_t\}_{t=1}^N$

$$\boldsymbol{\Phi}_{N \times \bar{N}} = \begin{bmatrix} \boldsymbol{\phi}_{N,1}^T \\ \boldsymbol{\phi}_{N,2}^T \\ \vdots \\ \boldsymbol{\phi}_{N,N}^T \end{bmatrix} = \left[\boldsymbol{\Phi}_{N \times n_o}^{(1)} \quad \boldsymbol{\Phi}_{N \times n_o}^{(2)} \quad \cdots \quad \boldsymbol{\Phi}_{N \times n_o}^{(N)} \right] \quad (12)$$

where $\boldsymbol{\Phi}_{N \times n_o}^{(l)} \in \mathbb{R}^{N \times n_o}$ is the response matrix of the l th candidate node whose center vector is $\mathbf{c}_l = \mathbf{x}'_t$ and scalar vector

is $\boldsymbol{\delta}_l = \mathbf{d}_l$. Since each hidden node contains n_o local predictors and N hidden nodes are employed to form the full MGRBF model, the size of the regression matrix is $N \times \bar{N}$. For $1 \leq l \leq N$, $\boldsymbol{\Phi}_{N \times n_o}^{(l)}$ can be expressed as

$$\boldsymbol{\Phi}_{N \times n_o}^{(l)} = \left[\boldsymbol{\phi}_{N,1}^{(l)} \quad \boldsymbol{\phi}_{N,2}^{(l)} \quad \cdots \quad \boldsymbol{\phi}_{N,n_o}^{(l)} \right] \quad (13)$$

with $\boldsymbol{\phi}_{N,i}^{(l)} = [\phi_{l,i}(\mathbf{x}'_1) \cdots \phi_{l,i}(\mathbf{x}'_N)]^T$ being the response of the i th local predictor of the l th node to $\{\mathbf{x}'_t\}_{t=1}^N$. Constructing the M -term MGRBF network $\{\mathbf{c}_j, \boldsymbol{\delta}_j\}_{j=1}^M$ for $M \ll N$ becomes a subset selection problem of selecting the M matrix bases $\{\boldsymbol{\Phi}_{N \times n_o}^{(j)}\}_{j=1}^M$ from the full regression matrix $\boldsymbol{\Phi}_{N \times \bar{N}}$.

Remark 2: Note that this subset selection problem is very different from constructing an M -term MRBF network. The latter task is to select a subset of the M basis vectors from the full set of the N basis vectors, and the multioutput OLS algorithm [22], [23], [24] can readily be applied to efficiently solve this subset selection problem. In contrast, our current subset selection problem is to select a subset of M basis matrices from the full set of the N basis matrices, and it is mathematically infeasible to directly employ the OLS to select these basis matrices from the candidate pool. Fig. 2 highlights this main difference of the two subset selection tasks by displaying the full regression matrices or candidate pools for these two tasks. Also, the hybrid constructive algorithms of [26] and [27] cannot be used to train the MGRBF network. To the authors' best knowledge, we are not aware any existing subset selection algorithm which can select subset matrix bases.

We propose to solve the problem of constructing the MGRBF network by separating it into two parts. First, we select appropriate centers from the training dataset $\{\mathbf{x}'_t; \mathbf{y}_t\}_{t=1}^N$. This subset selection problem can be formulated as the problem of constructing an "equivalent" M -term MRBF network with the OLS algorithm. Second, with the selected centers $\{\mathbf{c}_j = \mathbf{x}'_{t_j}\}_{j=1}^M$ from the constructed MRBF model, their associated scalar vectors are assigned to $\{\boldsymbol{\delta}_j = \mathbf{d}_{t_j}\}_{j=1}^M$ so as to complete the MGRBF hidden layer. The output weight matrix of this constructed M -node MGRBF network is finally solved by the regularized least square (LS) estimation method. This two-step construction procedure is detailed as follows.

OLS-Based Center Selection: The aim is to select the M center vectors of the MGRBF network from the training dataset. This is equivalent to construct an M -node subset MRBF network with the OLS algorithm. By using every input data \mathbf{x}'_i as a candidate center vector, we can model the training dataset $\{\mathbf{x}'_i; \mathbf{y}_i\}_{i=1}^N$ with the full N -node MRBF network as

$$\mathbf{Y}_{N \times n_o} = \Psi_{N \times N} \Theta_{N \times n_o} + \mathbf{E}_{N \times n_o} \quad (14)$$

where $\Theta_{N \times n_o} \in \mathbb{R}^{N \times n_o}$ is the output-layer connection weight matrix of the full N -node MRBF network, and $\Psi_{N \times N} \in \mathbb{R}^{N \times N}$ is the corresponding full regression matrix given by

$$\Psi_{N \times N} = [\psi_1 \ \psi_2 \ \cdots \ \psi_N] = \begin{bmatrix} \psi_{1,1} & \psi_{1,2} & \cdots & \psi_{1,N} \\ \psi_{2,1} & \psi_{2,2} & \cdots & \psi_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \psi_{N,1} & \psi_{N,2} & \cdots & \psi_{N,N} \end{bmatrix} \quad (15)$$

in which $\psi_{t,j}$ denotes the response of the j th node to \mathbf{x}'_t , while

$$\mathbf{Y}_{N \times n_o} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix} \in \mathbb{R}^{N \times n_o} \quad (16)$$

and $\mathbf{E}_{N \times n_o} \in \mathbb{R}^{N \times n_o}$ is the modeling residual matrix.

Let the orthogonal decomposition of $\Psi_{N \times N}$ be

$$\Psi_{N \times N} = \mathbf{W}_{N \times N} \mathbf{A}_{N \times N} = [\mathbf{w}_1 \ \mathbf{w}_2 \ \cdots \ \mathbf{w}_N] \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,N} \\ 0 & 1 & \cdots & a_{2,N} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{bmatrix} \quad (17)$$

with orthogonal columns that satisfy $\mathbf{w}_i^T \mathbf{w}_j = 0$, for $i \neq j$. The space spanned by the set of bases $\{\mathbf{w}_j\}$ is the same space spanned by the bases $\{\psi_j\}$, and (14) can be rewritten as

$$\mathbf{Y}_{N \times n_o} = \mathbf{W}_{N \times N} \mathbf{G}_{N \times n_o} + \mathbf{E}_{N \times n_o} \quad (18)$$

where the weight matrix $\mathbf{G}_{N \times n_o}$ for the space spanned by the columns of $\mathbf{W}_{N \times N}$, namely

$$\mathbf{G}_{N \times n_o} = \begin{bmatrix} g_{1,1} & \cdots & g_{1,n_o} \\ \vdots & \cdots & \vdots \\ g_{N,1} & \cdots & g_{N,n_o} \end{bmatrix} \quad (19)$$

is linked to the weight matrix $\Theta_{N \times n_o}$ by the triangular system $\mathbf{A}_{N \times N} \Theta_{N \times n_o} = \mathbf{G}_{N \times n_o}$. The classic Gram-Schmidt method can be used to perform this orthogonal decomposition [22].

For multioutput case, the contribution of a candidate basis to the trace of the desired output covariance matrix is used to define how significant this basis is, and the trace of the covariance of $\mathbf{Y}_{N \times n_o}$ can be expressed as [22], [23], [24]

$$\begin{aligned} \text{tr}(\mathbf{Y}_{N \times n_o}^T \mathbf{Y}_{N \times n_o} / N) &= \sum_{j=1}^N \left(\sum_{i=1}^{n_o} g_{j,i}^2 \right) \mathbf{w}_j^T \mathbf{w}_j / N \\ &\quad + \text{tr}(\mathbf{E}_{N \times n_o}^T \mathbf{E}_{N \times n_o} / N). \end{aligned} \quad (20)$$

The error reduction ratio due to \mathbf{w}_k can be defined as

$$[\text{err}]_k = \frac{\left(\sum_{i=1}^{n_o} g_{k,i}^2 \right) \mathbf{w}_k^T \mathbf{w}_k}{\text{tr}(\mathbf{Y}_{N \times n_o}^T \mathbf{Y}_{N \times n_o})}, \quad 1 \leq k \leq N. \quad (21)$$

Based on this ratio, significant nodes can be selected in a forward regression procedure. At the k th step of the selection procedure, a candidate node with the largest value of $[\text{err}]_k$ is selected, from among the rest of $N - k + 1$ candidates. The selection procedure is terminated when $1 - \sum_{j=1}^M [\text{err}]_j$ is smaller than a preset threshold, and this yields a regression model with M hidden nodes or centers $\{\mathbf{c}_j = \mathbf{x}'_{t_j}\}_{j=1}^M$.

MGRBF Construction and Weight Estimation: After the significant centers $\{\mathbf{c}_j = \mathbf{x}'_{t_j}\}_{j=1}^M$ have been obtained, they are used as the centers of the M -node MGRBF network, and the associated scalar vectors are assigned as $\{\delta_j = \mathbf{d}_{t_j}\}_{j=1}^M$. This completes the construction of the hidden layer of the MGRBF network. To determine the output layer connection matrix $\Theta_{\bar{M} \times n_o}$ of the constructed MGRBF network, express the network output matrix to the inputs $\{\mathbf{x}'_i\}_{i=1}^N$ by

$$\hat{\mathbf{Y}}_{N \times n_o} = \Phi_{N \times \bar{M}} \Theta_{\bar{M} \times n_o} \quad (22)$$

where $\Phi_{N \times \bar{M}}$ is the hidden layer response matrix. We can calculate the weight matrix $\Theta_{\bar{M} \times n_o}$ by minimizing the regularized LS cost function

$$J = \|\mathbf{Y}_{N \times n_o} - \hat{\mathbf{Y}}_{N \times n_o}\|^2 + \lambda \|\Theta_{\bar{M} \times n_o}\|^2 \quad (23)$$

where $\lambda \geq 0$ is the regularization parameter. Typically, a small positive λ is used, and $\lambda = 0$ corresponds to no regularization. The closed-form regularized LS solution of (23) is given by

$$\Theta_{\bar{M} \times n_o} = (\Phi_{N \times \bar{M}}^T \Phi_{N \times \bar{M}} + \lambda \mathbf{I}_{\bar{M}})^{-1} \Phi_{N \times \bar{M}}^T \mathbf{Y}_{N \times n_o} \quad (24)$$

where $\mathbf{I}_{\bar{M}}$ denotes the $\bar{M} \times \bar{M}$ identity matrix.

III. MGRBF TRACKER

With the above two-step training procedure, we can construct a compact MGRBF network by including the most significant training data states to accurately model the training data. Each selected hidden-node center encodes a significant system state from the training data and the node response vector is a perfect local predictor for the system output related to this system state. In a time-varying scenario, however, the process dynamics can vary dramatically and new data states may emerge, making some of the past data dynamics encoded in the hidden nodes of the MGRBF network obsolete. Therefore, it is vital for the MGRBF network to forget the most out-of-date old system states encoded in the hidden layer so as to free up space for capturing the newly emerged data dynamics as fast as these new states appear.

The single-output AGRBF [36] adopts an effective online adaptive strategy to address the above-mentioned problem. Specifically, it carries out weight updating at every sampling time as usual. If the prediction performance of the GRBF network becomes unacceptable, it replaces the hidden node that contributes the least with a new node to capture the newly emerging data dynamics. We modify this effective adaptive strategy to be suitable for the MGRBF network and propose the MGRBF *tracker*, which contains the *weight adaptation* and *tunable node adaptation* components.

A. Weight Adaptation

The weight matrix $\Theta_{\bar{M} \times n_o}$ of the MGRBF network can be simply updated by the RLS algorithm to track smooth data variations. At sample t , given \mathbf{x}'_t , the MGRBF network produces the output according to (10) as

$$\hat{\mathbf{y}}_t = \Theta_{\bar{M} \times n_o, t-1}^T \boldsymbol{\varphi}_{\bar{M}, t} \quad (25)$$

where $\Theta_{\bar{M} \times n_o, t-1}$ is the weight matrix obtained at sample $t-1$. Then, the prediction error $\mathbf{e}_t \in \mathbb{R}^{n_o}$ is given as

$$\mathbf{e}_t = \mathbf{y}_t - \Theta_{\bar{M} \times n_o, t-1}^T \boldsymbol{\varphi}_{\bar{M}, t} \quad (26)$$

The RLS algorithm adapts the weight matrix according to

$$\begin{cases} \mathbf{g}_t = \Gamma_{t-1} \boldsymbol{\varphi}_{\bar{M}, t} (\gamma + \boldsymbol{\varphi}_{\bar{M}, t}^T \Gamma_{t-1} \boldsymbol{\varphi}_{\bar{M}, t})^{-1} \\ \Gamma_t = (\Gamma_{t-1} - \mathbf{g}_t \boldsymbol{\varphi}_{\bar{M}, t}^T \Gamma_{t-1}) \gamma^{-1} \\ \Theta_{\bar{M} \times n_o, t} = \Theta_{\bar{M} \times n_o, t-1} + \mathbf{g}_t \mathbf{e}_t^T \end{cases} \quad (27)$$

where $\mathbf{g}_t \in \mathbb{R}^{\bar{M}}$ is the Kalman gain vector, $0.9 \leq \gamma < 1$ is the forgetting factor, and $\Gamma_t \in \mathbb{R}^{\bar{M} \times \bar{M}}$ is the inverse of the covariance matrix which is usually initialized to $\Gamma_0 = \vartheta \mathbf{I}_{\bar{M}}$ with ϑ being a large positive constant.

B. Tunable Node Adaptation

The RLS weight adaptation itself is insufficient under a highly nonstationary environment. When the MGRBF network performs poorly, the current MGRBF structure will need updating. The normalized average output error is used to measure the MGRBF performance at every sampling time

$$\tilde{\epsilon}_t = \|\mathbf{e}_t\|^2 / \|\mathbf{y}_t\|^2. \quad (28)$$

Based on this metric, we have the following criterion:

$$\begin{cases} \text{if } \tilde{\epsilon}_t < \varepsilon: \text{MGRBF structure unchanged} \\ \text{if } \tilde{\epsilon}_t \geq \varepsilon: \text{worst node replaced by new node} \end{cases} \quad (29)$$

where ε is a preset threshold. In general, the smaller ε is, the better modeling accuracy can be achieved, but the more frequent node replacement may occur.

When $\tilde{\epsilon}_t \geq \varepsilon$, the worst node with the least contribution to the overall performance is replaced with a new one. The contribution of a node is revealed by its sum of squared weighted local predictor outputs, which is defined by

$$\text{contri}_j = \sum_{i=1}^{n_o} (\boldsymbol{\phi}_{i,j}^T \boldsymbol{\theta}_{i,j})^2, \quad 1 \leq j \leq M. \quad (30)$$

We find the node with the smallest contri

$$m = \arg \min_{1 \leq j \leq M} \text{contri}_j \quad (31)$$

and replaced it by a new node. Since each MGRBF hidden node contains a center and n_o local predictor scalars, they need to be replaced together.

The center \mathbf{c}_m and scalars $\boldsymbol{\delta}_m$ of the new replacement node can be determined or “optimized” easily by exploiting the geometric property of MGRBF hidden node, similar to the AGRBF [35], [36]. Specifically, we simply set $\mathbf{c}_m = \mathbf{x}'_t$ and $\boldsymbol{\delta}_m = \mathbf{d}_t$ to ensure that the new replacement node m encodes the newest data state and is a perfect local multioutput predictor of \mathbf{y}_t . Since the set of centers now contains a new one,

the Gaussian width σ needs to be updated based on the new maximum Euclidean distance among the centers.

After the new node is determined, the weight matrix of the updated network needs to be recalculated. We use the p latest data $\{\mathbf{x}_{t-i}, \mathbf{y}_{t-i}\}_{i=0}^{p-1}$ to compute the regularized LS estimate

$$\Theta_{\bar{M} \times n_o, t} = (\Phi_{p \times \bar{M}, t}^T \Phi_{p \times \bar{M}, t} + \lambda \mathbf{I}_{\bar{M}})^{-1} \Phi_{p \times \bar{M}, t}^T \mathbf{Y}_{p \times n_o, t} \quad (32)$$

where the desired output matrix $\mathbf{Y}_{p \times n_o, t} \in \mathbb{R}^{p \times n_o}$ and the regression matrix $\Phi_{p \times \bar{M}, t} \in \mathbb{R}^{p \times \bar{M}}$ are given, respectively, by

$$\mathbf{Y}_{p \times n_o, t} = \begin{bmatrix} \mathbf{y}_t^T \\ \mathbf{y}_{t-1}^T \\ \vdots \\ \mathbf{y}_{t-p+1}^T \end{bmatrix}, \quad \Phi_{p \times \bar{M}, t} = \begin{bmatrix} \boldsymbol{\varphi}_{\bar{M}, t}^T \\ \boldsymbol{\varphi}_{\bar{M}, t-1}^T \\ \vdots \\ \boldsymbol{\varphi}_{\bar{M}, t-p+1}^T \end{bmatrix}. \quad (33)$$

In general, the number of the latest data p trades off estimation accuracy and tracking performance. For severely drifting or nonstationary data streams, a small p is preferred.

If the tunable node adaptation takes place at sample t , the RLS weight updating (27) does not take place. Instead, the weight matrix is updated with the regularized LS estimate (32), and we need to initialize the inverse covariance matrix to

$$\Gamma_t = (\Phi_{p \times \bar{M}, t}^T \Phi_{p \times \bar{M}, t} + \lambda \mathbf{I}_{\bar{M}})^{-1}. \quad (34)$$

This ensures a smooth transition from one adaptation mode to another at the next sample.

C. Algorithm Summary

The proposed MGRBF tracker is summarized in Algorithm 1. During the online operation, our MGRBF tracker adapts the model according to the current environment. If the process undergoes smooth variation, it operates only by the weight adaptation with the RLS algorithm. When abrupt changes occur in the system, it updates the model structure by replacing the worst performing node with the new one that automatically encodes the newly emerged data state and acts as the perfect local predictor of the current system output.

From a learning perspective, our MGRBF tracker achieves a balanced tradeoff between the “stability” and “plasticity” [35]. Specifically, it retains the acquired knowledge in the hidden layer of the MGRBF network (stability), with each node encoding an independent local data state learned from the history. At the same time, it adapts to new knowledge with a fast recovery (plasticity)—every time when a new system state emerges, it “frees” the memory by “forgetting” the most out-of-date knowledge to encode the newly emerged knowledge. It can be seen that our MGRBF tracker is designed to have fast tracking capability for accurately capturing the underlying characteristics of nonstationary data while maintaining low online computation complexity.

IV. NUMERICAL EXPERIMENTS

We perform extensive experiments, including modeling of real-world river flow time series as well as two industrial soft sensor applications, to evaluate the proposed MGRBF

Algorithm 1 AMGRBF Tracker

- 1: **Two-step training:** Construct M -node MGRBF network from N -sample training set with centers and local predictors $\{c_j, \delta_j\}_{j=1}^M$ and weight matrix $\Theta_{\tilde{M} \times n_o}$.
- 2: **Hyperparameters:** Node replacement threshold ε , bandwidth p , regularization parameter λ , forgetting factor γ .
- 3: **Initialization:** Set sample index $t = 1$, $\Gamma_{t-1} = \vartheta I_{\tilde{M}}$, $\Theta_{\tilde{M} \times n_o, t-1} = \Theta_{\tilde{M} \times n_o}$.
- 4: **Online prediction:** Given x'_t , compute MGRBF network prediction \hat{y}_t with (25).
- 5: **Online adaptation:** When y_t is available, compute \tilde{e}_t with (26) and (28).
- 6: **IF** $\tilde{e}_t < \varepsilon$: *Weight adaptation* mode
- 7: Update weight matrix to $\Theta_{\tilde{M} \times n_o, t}$ with RLS (27).
- 8: **ELSE IF** $\tilde{e}_t \geq \varepsilon$: *Tunable node adaptation* mode
- 9: Compute contributions for all M nodes with (30).
- 10: Find worst node m with (31), and replace it with new node by setting $c_m = x'_t$ and $\delta_m = y_t - y_{t-1}$.
- 11: Compute new Gaussian width σ with new maximum Euclidean distance among centers.
- 12: Use p latest data $\{x_{t-i}, y_{t-i}\}_{i=0}^{p-1}$ to compute weight matrix $\Theta_{\tilde{M} \times n_o, t}$ with regularized LS estimate (32).
- 13: Calculate Γ_t according to (34).
- 14: **END IF**
- 15: Set $t = t + 1$ and go to step 4.

network. Two metrics, the mean-square error (MSE) and the determinant of the error covariance $\log(\det(\text{Cov}(\mathbf{E})))$ over the test data, are utilized to evaluate each single-output and multioutputs online modeling performance, respectively. Additionally, the coefficient of determination (R^2) is also utilized. Since each output has an R^2 value, the averaged R^2 over all the outputs is used to evaluate the multioutputs modeling performance. The online computation complexity of an adaptive model is quantified by ACTpS. The computer for carrying out the experiments has the following configuration: Windows 10, 16 GB of RAM, CPU i7-9750 (2.60 GHz), and MATLAB version R2018b.

A. State-of-the-Art Benchmarks

We compare with the following state-of-the-art benchmarks.

- 1) MGAP-SER [15], consisting of multioutput local model growing and pruning as well SER-based online multioutput predictor, is a powerful multiple local model learning approach for multioutput nonstationary data modeling. The window size W , bandwidth p , and threshold ξ are the three algorithmic parameters, and their influence on the modeling performance can be found in [13], [14], and [15].
- 2) MTRBF is an extension of the single-output TRBF [33] to adaptive multioutput modeling. During online operation, it adopts a similar adaptive mechanism of [33] to replace the worst performing node with a new node. This new node replacement is achieved by iterative gradient descent optimization to determine the new center. Hence,

the MTRBF imposes much higher online complexity than our MGRBF tracker. We empirically set the step size and the number of iterations for gradient descent to 0.1 and 5, respectively. The node replacement threshold ε and bandwidth p are the two algorithmic parameters to be determined.

- 3) *Multiple AGRBFs:* We also employ the multiple (n_o) AGRBFs, one for each output of the multioutput systems in our experiments. The node replacement threshold ε and bandwidth p are the two algorithmic parameters, and their sensitivity analysis can be found in [35] and [36].
- 4) SAE is a deep network that learns hierarchical feature representations from data with multilevel feature layers [4], [5], [39], [40]. First, multiple autoencoders (AEs) are pretrained in an unsupervised way layer by layer. After pretraining, a linear regression layer with multioutput neurons is added on the top of the stacked AEs, and the whole SAE is trained by the stochastic gradient descent algorithm. The trained SAE is fixed during online operation, since it is impractical to adapt it in real time.
- 5) LSTM is a variant of RNN that is designed to extract dynamic temporal information from data [6], [7], [41], [42], [43]. For the LSTM with a single hidden layer and multioutput neurons, the Adam optimizer [44] is used to train the model based on the MSE cost. The trained LSTM is fixed during online operation, as it is impossible to adapt it in real time.
- 6) MRBF is a nonadaptive version of MTRBF. A compact MRBF network is constructed by the multioutput OLS algorithm [22], [23], [24] during the training, and the trained model is fixed during online prediction.
- 7) *Multiple LSSVMs:* A least-squares support-vector machine (LSSVM) is a least-squares version of the SVM [45], [46], [47], [48]. The trained multiple LSSVMs are fixed during online prediction, as it is difficult to adapt them online.
- 8) *Multiple RVMs:* The relevance vector machine (RVM) has a similar form to the SVM but requires fewer kernel functions and provides higher sparseness [49], [50], [51], [52], which is computationally more efficient than the SVM for modeling. The trained RVMs are fixed during online prediction, as it is difficult to adapt them in real time.

For the LSSVM and RVM, the Gaussian function is adopted as their nonlinear kernels. For the adaptive models, the forgetting factor of the RLS algorithm is set to $\gamma = 0.98$, and the regularization parameter is set to $\lambda = 0.001$. Other important hyperparameters are chosen carefully and empirically, as detailed in the following case studies.

B. River Network Flow Time Series

The river flow domain is a temporal prediction task designed to test predictions on the flows in a river network for 48 h in the future at specific locations [53]. The datasets were obtained from the U.S. National Weather Service and include hourly flow observations for eight sites in the Mississippi

TABLE I
TEST PERFORMANCE COMPARISON OF MRBF, MULTIPLE LSSVMs, MULTIPLE RVMS, LSTM, SAE, MGAP-SER,
MTRBF, MULTIPLE AGRBFs, AND PROPOSED METHOD FOR RIVER FLOW PREDICTION

Methods	Nodes/models	MSE (dB)				$\log(\det(\text{Cov}(\mathbf{E})))$	averaged R^2	ACTpS (ms)
		site 1	site 2	site 3	site 4			
MRBF	10	-16.0189	-11.2589	-12.8737	-18.7931	-8.0962	-1.4443	NA
multiple (4) LSSVMs	NA	-6.9514	-15.6354	-15.8206	-18.9504	-7.7754	-2.0772	NA
multiple (4) RVMS	NA	-10.7052	-10.5302	-10.5785	-12.9145	-7.3356	-4.5478	NA
LSTM	32	-11.8314±1.4558	-10.4837±1.4551	-13.8334±0.5648	-17.2837±2.2197	-7.6419±0.1807	-2.2374±0.7302	NA
SAE	[9, 6, 3]	-6.3909±2.1459	-7.9033±4.0252	-15.1941±2.0442	-9.7459±5.6262	-6.8673±0.5025	-14.1398±3.6912	NA
MGAP-SER	56 to 57	-41.0089	-34.3093	-40.1263	-36.9421	-15.2414	0.9841	2.6661
MTRBF	10	-40.6812	-34.9584	-40.4439	-36.4690	-15.6787	0.9836	0.8272
multiple AGRBFs	10×4	-41.5132	-36.5974	-41.4592	-37.1300	-15.6840	0.9866	0.5284
Proposed	10	-41.7907	-36.2358	-42.6381	-37.1156	-15.8511	0.9867	0.1159

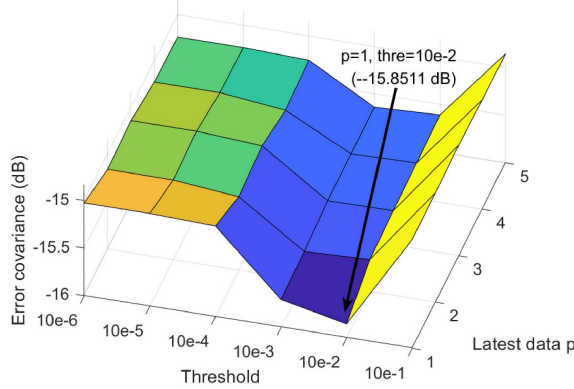


Fig. 3. Impact of node replacement threshold ε and bandwidth p on modeling performance of AMGRBF for river flow prediction.

River network in the United States from September 2011 to September 2012. This domain is a natural candidate for multi-target regression because there are clear physical relationships between readings in the contiguous river network [53]. We select the four sites in the river network with time-lagged observations from 6, 18, and 36 h in the past as the model inputs

$$\mathbf{x}_t = [y_{t-6,1} \ y_{t-18,1} \ y_{t-36,1} \ y_{t-6,2} \ y_{t-18,2} \ y_{t-36,2} \ \dots \ y_{t-6,4} \ y_{t-18,4} \ y_{t-36,4}]^T \in \mathbb{R}^{12} \quad (35)$$

to predict the river flows at the four sites 48 h in the future

$$\mathbf{y}_{t+48} = [y_{t+48,1} \ y_{t+48,2} \ y_{t+48,3} \ y_{t+48,4}]^T \in \mathbb{R}^4. \quad (36)$$

From the four sites, 2000 samples are collected with the first 500 samples as the training set and the remaining 1500 samples for online prediction and adaptive modeling.

The sizes of the MTRBF and each AGRBF network are empirically chosen to be $M = 10$, as suggested in [35] and [36]. Hence, the $n_o = 3$ AGRBFs have a total of 40 hidden nodes. For a fair comparison, the MRBF and our proposed method also have a network size of $M = 10$. The node replacement threshold ε and bandwidth p are two key algorithmic parameters for the proposed method, and we conduct a grid search, yielding the results depicted in Fig. 3. According to Fig. 3, we set $\varepsilon = 10e-2$ and $p = 1$ for our method. The node replacement thresholds are empirically chosen to be $10e-2$ and $10e-3$ for the AGRBFs and MTRBF, respectively, while their bandwidths are both set to be 1. We set the window size $W = 50$, bandwidth $p = 5$, and threshold $\xi = 0.4$ for the

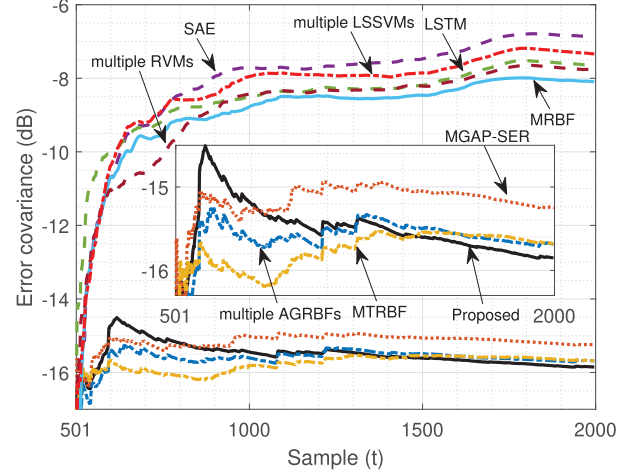


Fig. 4. Comparison of test $\log(\det(\text{Cov}(\mathbf{E})))$ learning curves of various models for river flow prediction.

MGAP-SER empirically. The LSTM network has 32 hidden nodes, and the structure of the SAE model is [9, 6, 3]. The learning rate and training epochs for both the SAE and LSTM are 0.001 and 200, respectively. The Gaussian kernel widths for the LSSVM and RVM are set to 1 and 10, respectively.

The online prediction performance by the nine models is tabulated in Table I, and their online error covariance learning curves are compared in Fig. 4. It is well known that the performance of deep neural networks, such as SAE and LSTM, depends on initialization. Therefore, the average MSEs and $\log(\det(\text{Cov}(\mathbf{E})))$ over 20 independent experiments together with the corresponding standard deviations are listed in Table I for the SAE and LSTM. Observe that the nonadaptive methods, including the MRBF, multiple LSSVMs, multiple RVMS, LSTM, and SAE, are the worst models, as evidenced by their large test $\log(\det(\text{Cov}(\mathbf{E})))$. In Table I, the black boldface value indicates the best performance and the blue boldface value indicates the second best one. Observe from Fig. 4 that all the adaptive models significantly outperform the nonadaptive models, while our method exhibits the fastest reduction in the error covariance among the adaptive models. The test prediction accuracies achieved by the four adaptive models are quite close, but our method is undoubtedly the winner, in terms of both online prediction accuracy and computational complexity. In particular, its ACTpS is only 0.1159 ms, which is nearly five times smaller than the second best multiple AGRBFs. Fig. 5 depicts the river flow prediction results by the

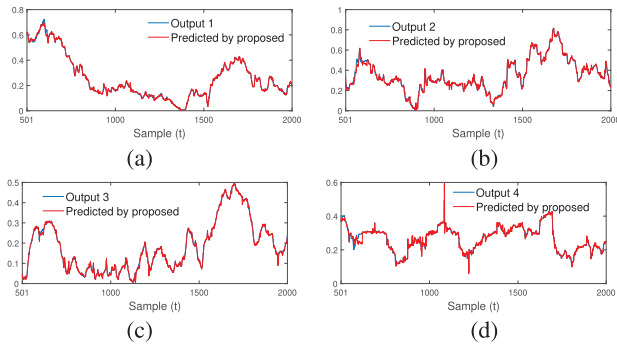


Fig. 5. Online prediction of river network flows by proposed AMGRBF: (a) site 1, (b) site 2, (c) site 3, and (d) site 4.

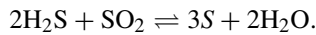
TABLE II
VARIABLE DESCRIPTION IN SRU PROCESS

Input and output variables	Description
$u_{t,1}$	MEA gas flow
$u_{t,2}$	First air flow
$u_{t,3}$	Second air flow
$u_{t,4}$	Gas flow in SWS zone
$u_{t,5}$	Air flow in SWS zone
$y_{t,1}$	Concentration of H_2S
$y_{t,2}$	Concentration of SO_2

proposed method, which clearly demonstrates that our method is capable of effectively capturing the real-time time-varying dynamics of the river flows in the four different sites.

C. Sulfur Recovery Unit Process

The sulfur recovery unit (SRU) [54] is used to remove environment pollutions from the acid gas streams. Specifically, it is widely used in petrochemical industry to recover H_2S as elemental sulfur through the Claus reaction



The sulfur recovery rate of the Claus process is about 95%–97%. The tail gas normally contains unrecovered sulfur H_2S and SO_2 , which are harmful to human health. Therefore, their concentrations must be properly monitored before released to atmosphere. However, these two kinds of acid gas can seriously damage hardware sensors by corrosion and consequently the hardware instruments are frequently removed for maintenance. To avoid this costly maintenance and to provide continuously monitoring, soft sensors are often employed to estimate the concentrations of H_2S and SO_2 online. Five process variables and the concentrations of H_2S and SO_2 are considered as the process inputs and outputs for the soft sensor, which are tabulated in Table II.

Based on expert knowledge and physical insight, the input vector to the SRU process can be expressed as [54]

$$\mathbf{x}_t = [u_{t,1} \ u_{t-5,1} \ u_{t-7,1} \ u_{t-9,1} \ u_{t,2} \ \cdots \ u_{t-9,4} \ u_{t,5} \ u_{t-5,5} \ u_{t-7,5} \ u_{t-9,5}]^T \in \mathbb{R}^{20} \quad (37)$$

and the process output vector is $\mathbf{y}_t = [y_{t,1} \ y_{t,2}]^T \in \mathbb{R}^2$. Because \mathbf{x}_t does not contain the past system output, we have $\mathbf{x}'_t = \mathbf{x}_t$.

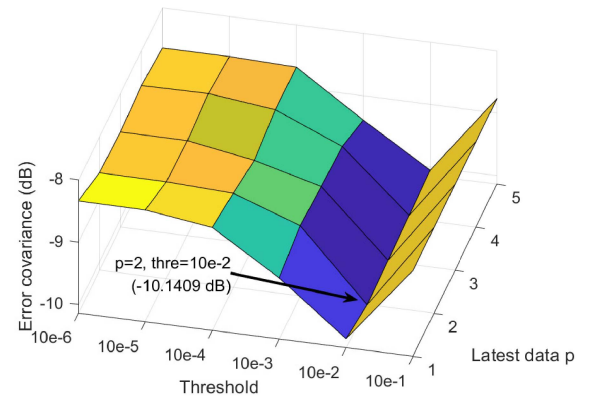


Fig. 6. Impact of node replacement threshold ε and bandwidth p on modeling performance of AMGRBF for online modeling of SRU process.

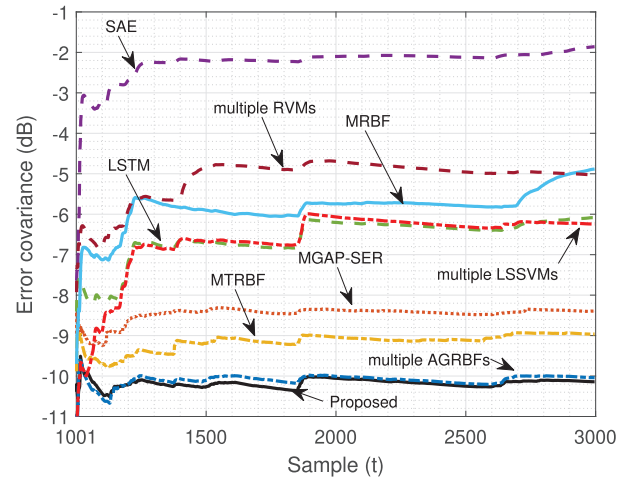


Fig. 7. Comparison of test $\log(\det(\text{Cov}(\mathbf{E})))$ learning curves of various models for online modeling of SRU process.

Total of 3000 samples are collected from the SRU dataset, among which the first 1000 samples are used for training and the rest of 2000 samples for online prediction.

Similarly, we set node replacement threshold $\varepsilon = 10e - 2$ and bandwidth $p = 2$ for the proposed method according to Fig. 6. The node replacement thresholds are set to be $10e - 2$ and $10e - 3$ for the AGRBFs and MTRBF, respectively, while their bandwidths are both set to be 2. We set window size $W = 90$, bandwidth $p = 4$, and threshold $\xi = 0.9$ for the MGAP-SER by trail and error. The LSTM network has 128 hidden nodes, and the structure of the SAE model is [15, 10, 7]. The structure parameters for the MRBF, LSSVM, RVM, LTSM, and SAE are identical with the previous simulation.

Table III lists the online prediction results attained by the nine models, and Fig. 7 compares their online learning curves. The average R^2 of the SAE is extremely poor, as it can hardly predict this process. Our method attains slightly better prediction accuracy than the second best AGRBFs while imposing an ACTpS half of the later. The MGAP-SER and MTRBF achieve comparable performance, but the former costs much higher computation time. Again, the nonadaptive models are inferior to the adaptive models. The online prediction

TABLE III
TEST PERFORMANCE COMPARISON OF MRBF, MULTIPLE LSSVMS, MULTIPLE RVMS, LSTM, SAE, MGAP-SER, MTRBF, MULTIPLE AGRBFs, AND PROPOSED METHOD FOR ONLINE MODELING OF SRU PROCESS

Methods	Nodes/models	MSE (dB)		log(det(Cov(E))) (dB)	averaged R ²	ACTpS (ms)
		H ₂ S	SO ₂			
MRBF	10	-22.6132	-21.9699	-4.8827	-4.5430	NA
multiple (2) LSSVMs	NA	-24.5532	-22.9210	-5.0276	-2.8162	NA
multiple (2) RVMS	NA	-31.5518	-27.1409	-6.2441	0.0009	NA
LSTM	128	-29.9898±0.7944	-26.7115±1.6079	-6.0864±0.1645	-0.3132±0.2427	NA
SAE	[15, 10, 7]	-9.2133±5.1917	-1.1580±3.9066	-1.8591±0.9642	-465.627±459.5	NA
MGAP-SER	97 to 99	-43.6743	-39.9716	-8.3931	0.9434	2.5112
MTRBF	10	-46.5586	-42.4370	-8.9619	0.9695	0.6439
multiple AGRBFs	10×2	-51.0393	-48.7857	-10.0409	0.9910	0.0683
Proposed	10	-51.4999	-48.7655	-10.1409	0.9915	0.0344

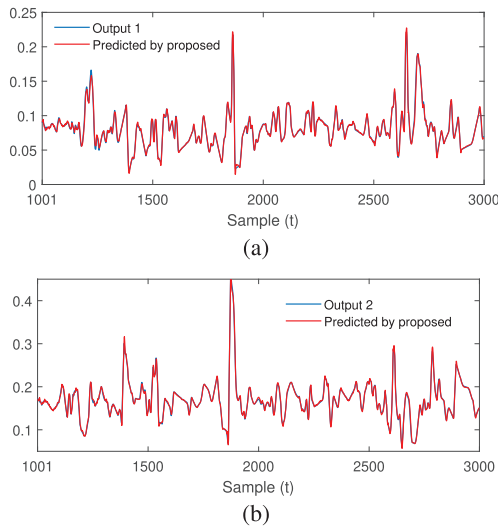


Fig. 8. Online modeling of SRU process by AMGRBF: (a) H₂S and (b) SO₂.

results of H₂S and SO₂ by our method are shown in Fig. 8, which again demonstrates its excellent tracking capacity.

The aforementioned results are the one-step-head predictive models' performance. Our proposed method is equally applicable to the multistep prediction, similar to the other multioutput adaptive models. We further compare the multistep ahead prediction performance of three multioutput adaptive models, namely, the MGAP-SER, MTRBF, and our proposed method in Fig. 9. It is evident that the proposed method consistently outperforms the other two methods at any prediction step. In terms of online computational complexity, a multistep-ahead predictor has similar ACTpS as its one-step-ahead counterpart. The best multistep-ahead prediction accuracy and the lowest online computational complexity together with a fixed compact network size make our AMGRBF network ideal for adaptive controller implementation.

D. Wastewater Treatment Plant

The wastewater treatment plant (WWTP) [55] is a large nonlinear and time-varying system subject to large perturbations in influent flow rate and pollutant load, together with uncertainties concerning the composition of the incoming wastewater. As shown in Fig. 10, the benchmark plant contains a five-compartment activated sludge reactor consisting of two anoxic

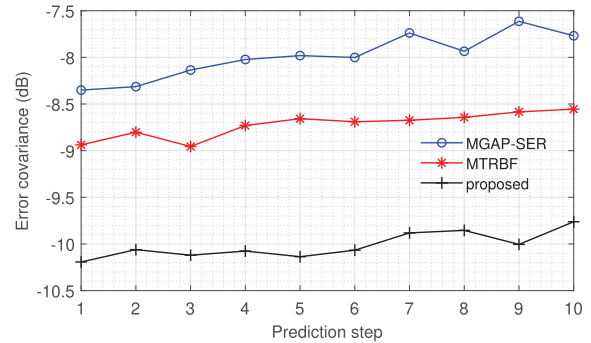


Fig. 9. Comparison of multistep-ahead prediction performance for SRU process by MGAP-SER, MTRBF, and proposed method.

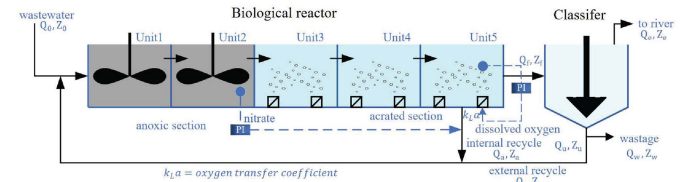


Fig. 10. Schematic overview of WWTP.

TABLE IV
VARIABLE DESCRIPTION IN THE WWTP

Input and output variables	Description
$u_{t,1}$	Readily biodegradable substrate
$u_{t,2}$	Particulate inert organic matter
$u_{t,3}$	Slowly biodegradable substrate
$u_{t,4}$	Active heterotrophic biomass
$u_{t,5}$	NH ₄ ⁺ + NH ₃ nitrogen
$y_{t,1}$	S _{ND}
$y_{t,2}$	X _{ND}
$y_{t,3}$	Flow rate

tanks followed by three aerobic tanks. The activated sludge reactor is followed by a secondary clarifier. The aim of this process is to remove organic matter and perform nitrification and denitrification. The purpose of this study is to establish an adaptive multioutput soft sensor to simultaneously estimate the soluble biodegradable organic nitrogen S_{ND}, particulate biodegradable organic nitrogen X_{ND}, and flow rate, which are three key variables. The process inputs and outputs are listed in Table IV. Hence, the process input vector is

TABLE V
TEST PERFORMANCE COMPARISON OF MRBF, MULTIPLE LSSVMs, MULTIPLE RVMS, LSTM, SAE, MGAP-SER,
MTRBF, MULTIPLE AGRBFs, AND PROPOSED METHOD FOR ONLINE MODELING OF WWTP

Methods	Nodes/models	MSE (dB)			$\log(\det(\text{Cov}(\mathbf{E})))$ (dB)	averaged R^2	ACTpS (ms)
		S_{ND}	X_{ND}	Flow rate			
MRBF	10	-31.9835	-36.7155	-12.0979	-8.8920	0.6646	NA
multiple (3) LSSVMs	NA	-25.9981	-27.8330	-12.0723	-8.6290	0.6392	NA
multiple (3) RVMS	NA	-31.5492	-35.9096	-11.6451	-8.8968	0.6276	NA
LSTM	64	-29.5306 ± 0.7073	-29.8652 ± 1.7440	-10.4257 ± 0.1561	-8.1325 ± 0.1170	0.5000 ± 0.0167	NA
SAE	[5, 4, 3]	-29.1925 ± 6.7376	-36.7223 ± 5.7536	-9.8310 ± 0.7487	-9.9372 ± 0.8495	0.4233 ± 0.0832	NA
MGAP-SER	209 to 210	-67.3247	-49.5914	17.4971	-9.9418	-299.0	4.8829
MTRBF	10	-34.2170	-35.8245	-30.6560	-10.0960	0.9909	0.0907
multiple AGRBFs	10×3	-31.8713	-35.3647	-26.3281	-9.4047	0.9809	0.3084
Proposed	10	-34.7267	-38.0835	-31.5260	-10.5854	0.9927	0.0600

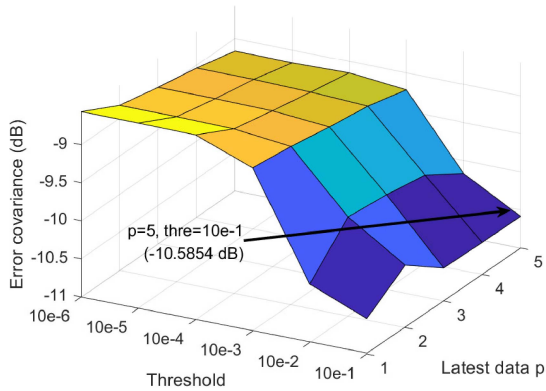


Fig. 11. Impact of node replacement threshold ε and bandwidth p on modeling performance of AMGRBF for online modeling of WWTP.

$$\mathbf{x}_t = [u_{t,1} \ u_{t,2} \ u_{t,3} \ u_{t,4} \ u_{t,5}]^T \in \mathbb{R}^5 \quad (38)$$

and the process output vector is

$$\mathbf{y}_t = [y_{t,1} \ y_{t,2} \ y_{t,3}]^T \in \mathbb{R}^3. \quad (39)$$

The influent data are collected under severe weather condition (a combination of dry weather and a long rain period), which makes the underlying process dynamics strongly nonstationary and imposes an enormous challenge on the performance of soft sensor [56]. We collect 1300 samples from the WWTP dataset, among which the first 500 samples are used for training, while the rest of them are for online prediction.

Again, we carefully set all algorithmic parameters of the nine models by trial and error. From Fig. 11, the node replacement threshold and bandwidth for the proposed method are set to $\varepsilon = 10e - 2$ and $p = 2$, respectively. The MTRBF has the identical algorithmic parameter settings with the proposed method, while the AGRBFs choose $\varepsilon = 10e - 1$ and $p = 1$ to achieve the best performance. For the MGAP-SER, we set window size $W = 15$, bandwidth $p = 1$, and threshold $\xi = 0.9$. The LSTM network has 64 hidden nodes, and the structure of the SAE model is [5, 4, 3]. The MRBF, LSSVM, and RVM have the identical structure settings with the previous simulation.

Table V presents the online modeling performance achieved by the nine models, and Fig. 12 compares their online error covariance learning curves. The results again show that our proposed method achieves the best online estimation accuracy while imposing the lowest computational complexity. It

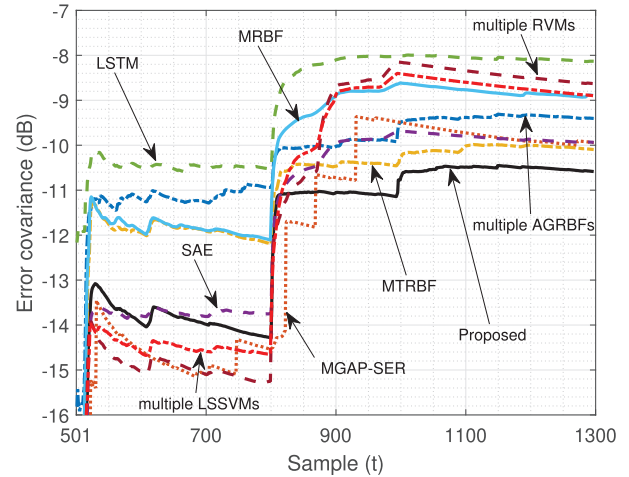


Fig. 12. Comparison of test $\log(\det(\text{Cov}(\mathbf{E})))$ learning curves of various models for online modeling of WWTP.

is interesting to observe that although the MGAP-SER attains the smallest MSEs for estimating S_{ND} and X_{ND} , its flow rate prediction is the worst among all the models, which is also reflected in its extremely poor average R^2 . This may be due to that the local multioutput linear models of the MGAP-SER do not handle the severe nonlinear relationship between the multiple output variables. The SAE achieves a comparable performance with the MGAP-SER and is slightly better than the multiple AGRBFs. However, its performance has a large fluctuation. Another interesting phenomenon can be seen from Fig. 12 is that the performance of all the nine models degrade significantly at nearly 800 samples. This can be explained from Fig. 13, which depicts the three process outputs. It can be clearly seen that all the three process outputs experience abrupt changes around $t = 800$, particularly for the flow rate. As mentioned before, this dataset is collected under severe weather conditions, and this abrupt drift can be contributed to the abrupt weather changes. The online prediction results of S_{ND} , X_{ND} , and flow rate by our method are depicted in Fig. 13, in comparison with the three true outputs of the WWTP. The results of Fig. 13 clearly demonstrate the excellent tracking capacity of the proposed MGRBF tracker.

Since multistep ahead process control is essential for WWTP, we also compare the multistep ahead prediction performance of the three adaptive multioutput models, and the

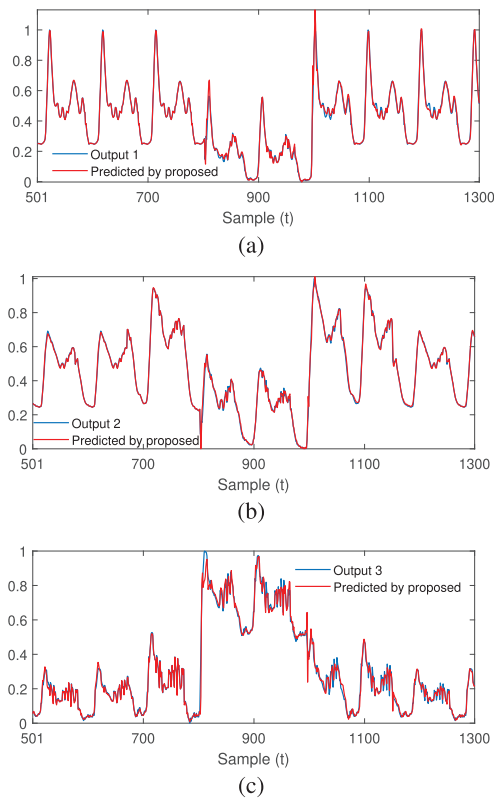


Fig. 13. Online modeling of WWTP by AMGRBF: (a) S_{ND} , (b) X_{ND} , and (c) flow rate.

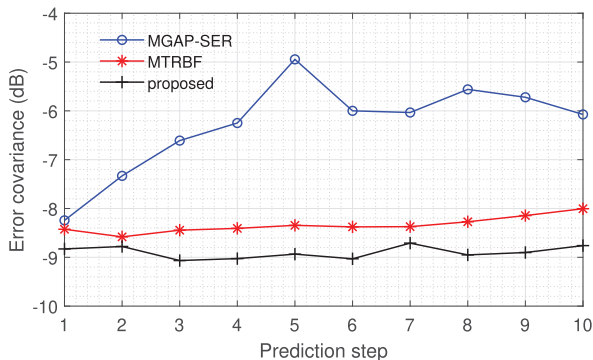


Fig. 14. Comparison of multistep-ahead prediction performance for WWTP by the MGAP-SER, MTRBF, and proposed method.

results are presented in Fig. 14. Again, our method consistently outperforms the other two methods. Our method has the additional advantage of imposing the lowest online computational burden.

Experimental results involving three real-world multioutput nonlinear and nonstationary processes demonstrate that the proposed AMGRBF tracker achieves the state-of-the-art adaptive modeling performance. Our method not only consistently attains the best tracking accuracy but also imposes very low computational complexity in online adaptation. Unlike adopting multiple single-output AGRBFs, our method is capable of capturing the complex interactions among the multiple process outputs and simultaneously tracking fast time-varying dynamics of the different process outputs. Moreover, the proposed

method has excellent multistep-ahead prediction capability, which is highly desired for nonlinear model predictive control design.

V. CONCLUSION

We have proposed a novel AMGRBF network for online modeling of multioutput nonlinear and time-varying processes. First, we have designed a new MGRBF network structure with strong multioutput predictive modeling capacity. A two-step training procedure has been proposed to construct a compact MGRBF network, with each hidden node encoding an independent data state and acting as a perfect local predictor of the system output vector corresponding to this system state. Second, we have designed a new adaptive tracking mechanism to efficiently adapt the MGRBF network in real time. Specifically, during online operation, the MGRBF tracker replaces the worst performing node with a new one that automatically captures the newly emerged process state. Extensive experiments involving real-world river network flow time-series prediction and two industrial soft sensor applications have demonstrated that our proposed AMGRBF model outperforms the existing state-of-the-art online multioutput modeling methods, in terms of both online prediction accuracy and computation complexity. Our results have also confirmed that state-of-the-art online adaptive models are superior over non-adaptive deep neural networks for online adaptive modeling of multivariate nonlinear and fast time-varying data. With the advantages of excellent adaptive modeling capability and very low online computational complexity together with a small fixed-size network structure, our proposed AMGRBF network offers an ideal platform for implementing an adaptive nonlinear control scheme.

In addition to nonlinear and nonstationary characteristics, many real-world systems are highly dimensional with strong correlations among variables. A popular way to deal with high dimensionality is to employ deep-learning models to extract deep latent features from raw data. Due to the complex architecture of deep neural networks, it is difficult to adapt them online to track fast time-varying data dynamics. Hence, the integration of deep-learning models with the proposed AMGRBF offers an important future research direction for handling high-dimensional nonlinear and nonstationary data tracking.

REFERENCES

- [1] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: A survey," *IEEE Comput. Intell. Mag.*, vol. 10, no. 4, pp. 12–25, Nov. 2015.
- [2] I. Skrljanc et al., "Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey," *Inf. Sci.*, vol. 490, pp. 344–368, Jul. 2019.
- [3] J. L. Lobo et al., "Evolving spiking neural networks for online learning over drifting data streams," *Neural Netw.*, vol. 108, pp. 1–19, Dec. 2018.
- [4] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.
- [5] W. Yan, D. Tang, and Y. Lin "A data-driven soft sensor modeling method based on deep learning and its application," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4237–4245, May 2017.

- [6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
- [7] X. Yuan, L. Li, and Y. Wang, "Nonlinear dynamic soft sensor modeling with supervised long short-term memory network," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3168–3175, May 2020.
- [8] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2409–2429, Jul. 2020.
- [9] H. Borchani, G. Varando, C. Bielza, and P. Larranaga, "A survey on multi-output regression," *Data Min. Knowl. Disc.*, vol. 5, no. 5, pp. 216–233, 2015.
- [10] P. Kadlec and B. Gabrys, "Local learning-based adaptive soft sensor for catalyst activation prediction," *AIChE J.*, vol. 57, no. 5, pp. 1288–1301, May 2011.
- [11] W. Shao, X. Tian, P. Wang, X. Deng, and S. Chen, "Online soft sensor design using local partial least squares models with adaptive process state partition," *Chemometr. Intell. Lab. Syst.*, vol. 144, pp. 108–121, May 2015.
- [12] W. Shao, S. Chen, and C. J. Harris, "Adaptive soft sensor development for multi-output industrial processes based on selective ensemble learning," *IEEE Access*, vol. 6, pp. 55628–55642, 2018.
- [13] T. Liu, S. Chen, S. Liang, and C. J. Harris, "Selective ensemble of multiple local model learning for nonlinear and nonstationary systems," *Neurocomputing*, vol. 378, pp. 98–111, Feb. 2020.
- [14] T. Liu, S. Chen, S. Liang, and C. J. Harris, "Growing and pruning selective ensemble regression for nonlinear and nonstationary systems," *IEEE Access*, vol. 8, pp. 73278–73292, 2020.
- [15] T. Liu, S. Chen, S. Liang, S. Gan, and C. J. Harris, "Multi-output selective ensemble identification of nonlinear and nonstationary industrial processes," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 5, pp. 1867–1880, May 2022.
- [16] Q. Que and M. Belkin, "Back to the future: Radial basis function network revisited," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 1856–1867, Aug. 2020.
- [17] 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.
- [18] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.
- [19] X. Hong, R. J. Mitchell, S. Chen, C. J. Harris, K. Li, and G. W. Irwin, "Model selection approaches for non-linear system identification: A review," *Int. J. Syst. Sci.*, vol. 39, no. 10, pp. 925–946, Oct. 2008.
- [20] S. Chen, X. Hong, and C. J. Harris, "Particle swarm optimization aided orthogonal forward regression for unified data modelling," *IEEE Trans. Evol. Comput.*, vol. 14, no. 4, pp. 477–499, Aug. 2010.
- [21] L. Zhang, K. Li, E. Bai, and G. W. Irwin, "Two-stage orthogonal least squares methods for neural network construction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 8, pp. 1608–1621, Aug. 2015.
- [22] S. Chen, P. M. Grant, and C. F. N. Cowan, "Orthogonal least squares algorithm for training multi-output radial basis function networks," *IEE Proc. F, Radar Signal Process.*, vol. 139, no. 6, pp. 378–384, Dec. 1992.
- [23] S. Chen, "Multi-output regression using a locally regularised orthogonal least-squares algorithm," *IEE Proc. Vis. Image Signal Process.*, vol. 149, no. 4, pp. 185–195, Aug. 2002.
- [24] S. Chen, X. Hong, and C. J. Harris, "Sparse multioutput radial basis function network construction using combined locally regularised orthogonal least square and D-optimality experimental design," *IEE Proc. Control Theory Appl.*, vol. 150, no. 2, pp. 139–146, Mar. 2003.
- [25] S. A. Billings and S. Chen, "The determination of multivariable nonlinear models for dynamic systems," in *Control and Dynamic Systems: Neural Network Systems Techniques and Applications*, vol. 7, C. L. Leondes, Ed. San Diego, CA, USA: Academic, 1998, pp. 231–278.
- [26] D. Du, K. Li, X. Li, M. Fei, and H. Wang, "A multi-output two-stage locally regularized model construction method using the extreme learning machine," *Neurocomputing*, vol. 128, pp. 104–112, Mar. 2014.
- [27] X. Qian, H. Huang, X. Chen, and T. Huang, "Generalized hybrid constructive learning algorithm for multioutput RBF networks," *IEEE Trans. Cybern.*, vol. 47, no. 11, pp. 3634–3648, Nov. 2017.
- [28] S. Chen and S. Billings, "Recursive prediction error parameter estimator for non-linear models," *Int. J. Control*, vol. 49, no. 2, pp. 569–594, 1989.
- [29] S. Chen, "Nonlinear time series modelling and prediction using Gaussian RBF networks with enhanced clustering and RLS learning," *Electron. Lett.*, vol. 31, no. 2, pp. 117–118, Jan. 1995.
- [30] N. Liang, G. B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.
- [31] J. Platt, "A resource-allocating network for function interpolation," *Neural Comput.*, vol. 3, no. 2, pp. 213–225, Jun. 1991.
- [32] 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, Cybern. B, Cybern.*, vol. 34, no. 6, pp. 2284–2292, Dec. 2004.
- [33] 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.
- [34] 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.
- [35] T. Liu, S. Chen, S. Liang, S. Gan, and C. J. Harris, "Fast adaptive gradient RBF networks for online learning of nonstationary time series," *IEEE Trans. Signal Process.*, vol. 68, pp. 2015–2030, Mar. 2020.
- [36] T. Liu, S. Chen, S. Liang, D. Du, and C. J. Harris, "Fast tunable gradient RBF networks for online modeling of nonlinear and nonstationary dynamic processes," *J. Process Control*, vol. 93, pp. 53–65, Sep. 2020.
- [37] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken, NJ, USA: Wiley, 2015.
- [38] D. Lowe, "Adaptive radial basis function nonlinearities, and the problem of generalisation," in *Proc. 1st IEE Int. Conf. Artif. Neural Netw.*, London, U.K., Oct. 1989, pp. 171–175.
- [39] L. Wang, Z.-H. You, J.-Q. Li, and Y.-A. Huang, "IMS-CDA: Prediction of CircRNA-disease associations from the integration of multisource similarity information with deep stacked autoencoder model," *IEEE Trans. Cybern.*, vol. 51, no. 11, pp. 5522–5531, Nov. 2021.
- [40] Q. Sun and Z. Ge, "Gated stacked target-related autoencoder: A novel deep feature extraction and layerwise ensemble method for industrial soft sensor application," *IEEE Trans. Cybern.*, vol. 52, no. 5, pp. 3457–3468, May 2022.
- [41] N. Zheng, "Predicting COVID-19 in China using hybrid AI model," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 2891–2904, Jul. 2020.
- [42] J. Chen et al., "E-LSTM-D: A deep learning framework for dynamic network link prediction," *IEEE Trans. Cybern.*, vol. 51, no. 6, pp. 3699–3712, Jun. 2021.
- [43] S. M. Zandavi, T. H. Rashidi, and F. Vafaee, "Dynamic hybrid model to forecast the spread of COVID-19 using LSTM and behavioral models under uncertainty," *IEEE Trans. Cybern.*, vol. 52, no. 11, pp. 11977–11989, Nov. 2022.
- [44] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, [arXiv:1412.6980](https://arxiv.org/abs/1412.6980).
- [45] J. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, Jun. 1999.
- [46] X. Lu, L. Ming, T. Hu, and B. Fan, "Collaborative learning-based clustered support vector machine for modeling of nonlinear processes subject to noise," *IEEE Trans. Cybern.*, vol. 50, no. 12, pp. 5162–5171, Dec. 2020.
- [47] Q. Ge et al., "Industrial power load forecasting method based on reinforcement learning and PSO-LSSVM," *IEEE Trans. Cybern.*, vol. 52, no. 2, pp. 1112–1124, Feb. 2022.
- [48] X. Lu and Y. Bai, "Robust least-squares support vector machine using probabilistic inference," *IEEE Trans. Cybern.*, vol. 52, no. 6, pp. 4391–4399, Jun. 2022.
- [49] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Jun. 2001.
- [50] S. Chen, S. R. Gunn, and C. J. Harris, "The relevance vector machine technique for channel equalization application," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1529–1532, Nov. 2001.
- [51] L. Chen, L. Wang, J. Zhao, and W. Wang, "Relevance vector machines-based time series prediction for incomplete training dataset: Two comparative approaches," *IEEE Trans. Cybern.*, vol. 51, no. 8, pp. 4298–4311, Aug. 2021.
- [52] X. Wang, B. Jiang, S. X. Ding, N. Lu, and Y. Li, "Extended relevance vector machine-based remaining useful life prediction for DC-link capacitor in high-speed train," *IEEE Trans. Cybern.*, vol. 52, no. 9, pp. 9746–9755, Sep. 2022.
- [53] E. Spyromitros-Xioufifis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-label classification methods for multi-target regression," 2014, [arXiv:1211.6581v4](https://arxiv.org/abs/1211.6581v4).

- [54] L. Fortuna, S. Graziani, A. Rizzo, and M. G. Xibilia, *Soft Sensors for Monitoring and Control of Industrial Processes*. London, U.K.: Springer-Verlag, 2007.
- [55] U. Jeppsson et al., "Towards a benchmark simulation model for plant-wide control strategy performance evaluation of WWTPs," *Water Sci. Technol.*, vol. 53, no. 1, pp. 287–295, Jan. 2006.
- [56] Y. Liu, B. Liu, X. Zhao, and M. Xie, "Development of RVM-based multiple-output soft sensors with serial and parallel stacking strategies," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 6, pp. 2727–2734, Nov. 2019.



Tong Liu (Member, IEEE) received the B.S. degree in automation and the Ph.D. degree in control theory and engineering from the College of Automation, Chongqing University, Chongqing, China, in 2016 and 2021, respectively.

From 2018 to 2019, he was a joint training Ph.D. student with the Next Generation Wireless Group, School of Electronics and Computer Science, University of Southampton, Southampton, U.K. From 2021 to 2022, he worked as a Research Associate with the Department of Computer Science,

The University of Sheffield, Sheffield, U.K. He is currently a Research Associate with the Department of Chemical Engineering, Imperial College London, London, U.K. His research interests include machine learning and data analytics, and data-driven modeling, optimization, and control of complex industrial systems.



Sheng Chen (Fellow, IEEE) received the B.Eng. degree in control engineering from East China Petroleum Institute, Dongying, China, in 1982, the Ph.D. degree in control engineering from the City, University of London, London, U.K., in 1986, and the Doctor of Science degree from the University of Southampton, Southampton, U.K., in 2005.

From 1986 to 1999, he held research and academic appointments with the University of Sheffield, Sheffield, U.K.; the University of Edinburgh, Edinburgh, U.K., and the University of Portsmouth,

Portsmouth, U.K. Since 1999, he has been with the School of Electronics and Computer Science, University of Southampton, where he is a Professor of Intelligent Systems and Signal Processing. He has published over 650 research papers. He has 18 300+ Web of Science citations with H-index 59 and 35 800+ Google Scholar citations with H-index 81. His research interests include adaptive signal processing, wireless communications, modeling and identification of nonlinear systems, neural network and machine learning, evolutionary computation methods, and optimization.

Prof. Chen is a Fellow of the United Kingdom Royal Academy of Engineering, Asia-Pacific Artificial Intelligence Association, and IET. He is one of the original ISI highly cited researchers in engineering in March 2004.



Kang Li (Senior Member, IEEE) received the B.Sc. degree in industrial automation from Xiangtan University, Xiangtan, Hunan, China, in 1989, the M.Sc. degree in control theory and applications from the Harbin Institute of Technology, Harbin, China, in 1992, the Ph.D. degree in control theory and applications from Shanghai Jiao Tong University, Shanghai, China, in 1995, and the D.Sc. degree in engineering from Queen's University Belfast, Belfast, U.K., in 2015.

From 1995 to 2002, he worked with Shanghai Jiao Tong University; the Delft University of Technology, Delft, The Netherlands; and Queen's University Belfast as a Research Fellow. From 2002 to 2018, he was a Lecturer in 2002, a Senior Lecturer in 2007, a Reader in 2009, and a Professor in 2011 with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast. He currently holds the Chair of Smart Energy Systems and leads the Institute of Communication and Power Networks, University of Leeds, Leeds, U.K. He is also a Visiting Professor with Shanghai Jiao Tong University; Southeast University, Nanjing, China; the Shenzhen Advanced Institute of Technology, Chinese Academy of Sciences, Shenzhen, China; Shanghai University, Shanghai; Xiangtan University; and the Qingdao University of Science and Technology, Qingdao, China. He has authored/coauthored over 200 journal publications and edited/co-edited 18 conference proceedings, winning over 20 prizes and awards. His research interests cover nonlinear system modeling, identification, and control, and machine learning, with substantial applications to energy and power systems, transport decarbonization, and energy management in energy intensive manufacturing processes.

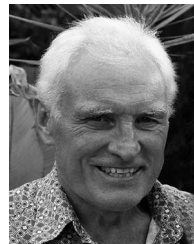
Dr. Li was the Chair of the IEEE UKRI Control and Communication Ireland Chapter and the Secretary of the IEEE U.K. and Ireland Section.



Shaojun Gan received the M.Sc. degree in control engineering and the Ph.D. degree in control theory and control engineering from Chongqing University, Chongqing, China, in 2013 and 2017, respectively.

He was an exchange Ph.D. student with Queen's University Belfast, Belfast, U.K., from 2014 to 2016. He became a Lecturer with the Beijing University of Technology, Beijing, China, in 2019 and got promoted to an Associate Professor in 2022. His research interests include nonlinear system modeling and identification with recent applications on intel-

ligent transportation systems.



Chris J. Harris received the B.Sc. degree from the University of Leicester, Leicester, U.K., in 1967, the M.A. degree from the University of Oxford, Oxford, U.K., in 1976, and the Ph.D. and Doctor of Science degrees from the University of Southampton, Southampton, U.K., in 1972 and 2001, respectively.

He is an Emeritus Research Professor with the University of Southampton, having previously held senior academic appointments with Imperial College London, London, U.K., the University of Oxford, and The University of Manchester, Manchester, U.K., as well as the Deputy Chief Scientist for the U.K. Government. He is the coauthor of over 550 scientific research papers during a 55-year research career.

Prof. Harris was awarded the IEE Senior Achievement Medal for data fusion research and the IEE Faraday Medal for distinguished international research in machine learning. He was elected as a Fellow of the U.K. Royal Academy of Engineering in 1996.