Efficient Adaptive Deep Gradient RBF Network For Multi-output Nonlinear and Nonstationary Industrial Processes

Tong Liu[a], Sheng Chen[b,c], Po Yang[d], Yunpeng Zhu[e], Chris J. Harris[b]

[a]*Department of Chemical Engineering, Imperial College London, London SW7 2AZ, UK*
[b]*School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK*
[c]*Faculty of Information Science and Engineering, Ocean University of China, Qingdao, China*
[d]*Department of Computer Science, University of Sheffield, Sheffield S1 4DP, UK*
[e]*School of Engineering and Material Science, Queen Mary University of London, London E1 4NS, UK*

Abstract

Due to the complexity of process operation, industrial process data are often nonlinear and nonstationary, high dimensional, and multivariate with complex interactions between multiple outputs. To address all these issues, this paper proposes a novel industrial predictive model that integrates deep feature extraction and fast online adaptation, and can effectively deal with multiple process outputs. Specifically, a multi-output gradient radial basis function network (MGRBF) with excellent predictive capacity of nonstationary data is first used to provide preliminary prediction of target outputs. This prior quality information is combined with the original process input for deep feature learning and dimensional reduction. Through layer-wise feature extraction by the stacked autoencoder (SAE), deep quality-enhanced features can be obtained, which is further fed into a MGRBF tracker for online prediction. In order to timely capture the fast-changing process characteristics, the first two modules, namely, preliminary MGRBF predictor and SAE feature extractor are frozen after training, while the structure and parameters of the MGRBF tracker are

Email addresses: tliu.soton@gmail.com (Tong Liu), sqc@ecs.soton.ac.uk (Sheng Chen), po.yang@sheffield.ac.uk (Po Yang), yunpeng.zhu@qmul.ac.uk (Yunpeng Zhu), chrisharris57@msn.com (Chris J. Harris)

updated online in an efficient manner. Two industrial case studies demonstrate that the proposed adaptive deep MGRBF network outperforms existing state-of-the-art online modeling approaches as well as deep learning models, in terms of both multi-output modeling accuracy and online computational complexity.
Keywords: Multivariate nonlinear and nonstationary industrial process, multi-output gradient radial basis function network, stacked autoencoder, quality-enhanced feature extraction, online adaptive tracking

1. Introduction

To meet stringent requirements on safety, efficiency, and sustainability in modern process industry, process control and high-level decision making are urgently needed. These needs rely on accurately modeling of industrial plants from operational data, delivered by system identification and soft sensing [1]. The former aims to provide process dynamics for controller design, while the latter is to estimate key performance indicators based on easy-to-measure process variables. However, due to the complexity of process operation, industrial process data are usually nonlinear and nonstationary, high dimensional with strong correlations, and multivariate with complex interactions among multiple outputs [2, 3, 4, 5, 6, 7]. Although numerous studies have been devoted to address one or two of these issues, no research has addressed them all. For example, traditional deep learning models can extract useful features from high dimensional operation data, but it is difficult to adapt them online to track fast time-varying process dynamics due to their complex deep architecture [8, 9, 10]. This motivates our current work to develop an industrial predictive model that has both fast online adaptation and deep feature learning capacity as well as can effectively deal with multiple process outputs.

For nonlinear and nonstationary processes, multiple local model learning strategy has been widely used in adaptive soft sensor modeling [11, 12, 13]. The core idea is to partition the overall modeling space into multiple subspaces, and each subspace is considered to be stationary that can be coupled by a local

linear model. The growing and pruning selective ensemble regression (GAP-SER) grows local linear models online to automatically identify newly emerged process patters and combines the most up-to-data local models to make online prediction as well as prunes the unwanted out-of-data local models to reduce the online complexity [14, 15]. This GAP-SER is further extended to multi-output modeling, in which it adopts a novel adaptive local learning strategy based on multivariate statistic that enables growing and pruning multi-output local linear models [16]. Since the multi-output GAP-SER can exploit the complex interactions between multiple outputs, it attains better prediction accuracy than using multiple single-output GAP-SERs when modeling multivariate nonstationary industrial processes [16]. A potential drawback of the GAP-SER methods is that the size of the predictor changes from sample to sample, which makes it hard to act as an identifier in real-time process control.

As an alternative to the above methods, single nonlinear model learning that attempts to capture global nonlinear data characteristics, has also attracted considerable attention in processes modeling. One typical nonlinear model is radial basis function (RBF) network. By formulating the RBF network training as a subset selection problem, the well-known orthogonal least squares (OLS) is used to construct a parsimonious compact RBF model from the full model [17, 18, 19, 20]. The RBF network can be easily extended to multi-output modeling by adding multiple output neurons to the single-output network structure [21]. To provide the RBF model with adaptive capacity, the recursive least square (RLS) is usually employed to update the network weights online [22]. However, this is insufficient for highly nonstationary processes, where the process dynamics can change significantly and new process states may appear. In order to capture the newly emerged process state, the fast tunable RBF (TRBF) adjusts the model weights as well as RBF nodes online to adaptively modeling nonstationary data [23]. This fast TRBF algorithm can be naturally extended to multivariate processes modeling using the multi-output RBF network structure.

An extension to RBF network, called gradient RBF (GRBF) network, was proposed to deal with nonstationary time series [24]. This GRBF network

trained by the OLS algorithm is better at predicting nonstationary time series than the classic RBF network, because its hidden node can sense the gradient of time series rather than series itself [24]. By incorporating a highly efficient tunable node mechanism, an adaptive GRBF (AGRBF) was proposed for online time series prediction [25], which was further extended to online modeling of time-varying industrial processes [26]. The results of [26] show that this AGRBF is superior to the TRBF and GAP-SER, in terms of both online modeling accuracy and computational complexity. In order to deal with multivariate data, a novel multi-output GRBF (MGRBF) network structure was designed in [27], which is very different from the single-output GRBF network. Unlike most existing neural networks that produce single response of its hidden node, the MGRBF's hidden node produces multiple responses to the node's input, which correspond to multiple local predictors for different outputs. When equipping with an online adaptive mechanism, this MGRBF tracker outperforms the multi-output GAP-SER and multiple single-output AGRBFs for multivariate nonstationary processes modeling [27].

The aforementioned adaptive models, particularly the MGRBF tracker, impose very low computational complexity to perform online model adaptation, which meets the strict online computational constraint imposed by the system's sampling period. Hence, these approaches are efficient for online tracking applications. Despite the excellent adaptive capacity to nonstationary data, all the aforementioned methods have difficulty in dealing with high dimensionality that is commonly encountered in big process data. A simple way to reduce data dimension is to employ latent variable models, such as partial least square (PLS) that enables modeling data in the reduced-dimensional latent space [28, 29, 30, 31]. However, such models with shallow structure may fail to capture complex nonlinear features from process data. Another popular way is to use deep learning models, which has attracted growing attention recently in industrial processes modeling [9, 10, 32, 33, 34, 35, 36, 37, 38]. Deep neural networks with multilevel feature layers can effectively learn the compressed essential features from raw data and discover the intricate data patters. One

typical deep model used in soft sensor modeling is the stacked autoencoder (SAE). By stacking multiple autoencoders (AEs), hierarchical features can be successively learned from raw operational data, which are further used for the process output prediction. Several variants have been proposed to incorporate quality-relevant information into feature representation so as to improve the prediction accuracy [35, 36, 37, 38]. Another commonly used deep model is recurrent neural networks, such as long short-term memory (LSTM), which are good at extracting dynamic temporal information from time series data [39, 40]. These deep neural networks have deep nonlinear learning capability. Although these deep learning models achieve great success for dealing with large-scale big processes data, applying them in nonstationary industrial environments for real-time tracking remains largely understudied. This is because most deep models have a huge network architecture, and it is computationally prohibitive to optimize such large-size model structure online for timely tracking fast time-varying processes dynamics. Therefore, during online operation, these existing deep neural networks are fixed, and consequently their online prediction performance are significantly degraded. Additionally, most deep models are only used for single-output modeling, and applications to multivariate industrial processes have not been extensively investigated.

Motivated by the above background, this paper proposes a novel deep neural network for online modeling and identification of multi-output nonlinear and nonststaionary industrial processes, called the adaptive deep MGRBF network. The proposed framework integrates deep feature learning and fast online adaptation naturally, and it can effectively deal with multiple process outputs. Our novel contribution is three-fold:

1. In order to learn better and deeper features from high-dimensional industrial data, a quality-relevant feature extraction strategy is proposed by integrating MGRBF predictor and SAE feature extractor. To be specific, an MGRBF network is first employed to provide a preliminary prediction of the target outputs. This prior quality information is combined with the

original input data for feature learning. After layer-wise feature extraction by the SAE, deep quality-enhanced features obtained are used for the prediction of the process's multi-outputs by an MGRBF tracker.

2. In order to timely capture the fast time-varying process characteristics, an efficient online adaptation strategy is designed to tune the adaptive part of the deep model. During the online operation, the first two modules are unchanged, and they collaborate to extract the quality-related deep features, which are fed into the MGRBF tracker for online prediction and adaptive modeling. When the current model structure becomes insufficient for modeling the changing process dynamics, the worst node of the MGRBF tracker is replaced by a new node that automatically encodes the current process state.

3. Our proposed method is evaluated using two industrial case studies, soft sensing for penicillin fermentation process and online identification of a real-world industrial microwave heating process. Experimental results demonstrate that our method outperforms many state-of-the-art online modeling approaches as well as deep learning models, in terms of both multi-output prediction accuracy and online computational complexity.

2. Multi-output GRBF Network

The task of online modeling of multi-output nonlinear and time-varying industrial process is to build a predictive model $\widehat{\boldsymbol{y}}_t = \boldsymbol{f}_{\text{sys}}(\boldsymbol{x}_t) \in \mathbb{R}^{n_o}$ to predict the multiple process outputs $\boldsymbol{y}_t \in \mathbb{R}^{n_o}$ given the input $\boldsymbol{x}_t \in \mathbb{R}^{n_i}$ at every sampling time t, where $\boldsymbol{y}_t = \left[y_{t,1} \cdots y_{t,n_o}\right]^{\mathrm{T}}$ is the n_o-dimensional process output vector, $\boldsymbol{x}_t = \left[x_{t,1} \cdots x_{t,n_i}\right]^{\mathrm{T}}$ is the n_i-dimensional system input vector, and $\boldsymbol{x}_t$ may contain past process outputs, past process inputs or both or even past process output gradients depending on the model structure design [3, 20, 26].

The MGRBF network can be adopted to perform this task. The structure of the MGRBF network is shown in Fig. 1. The input vector $\boldsymbol{x}_t$ is mapped onto

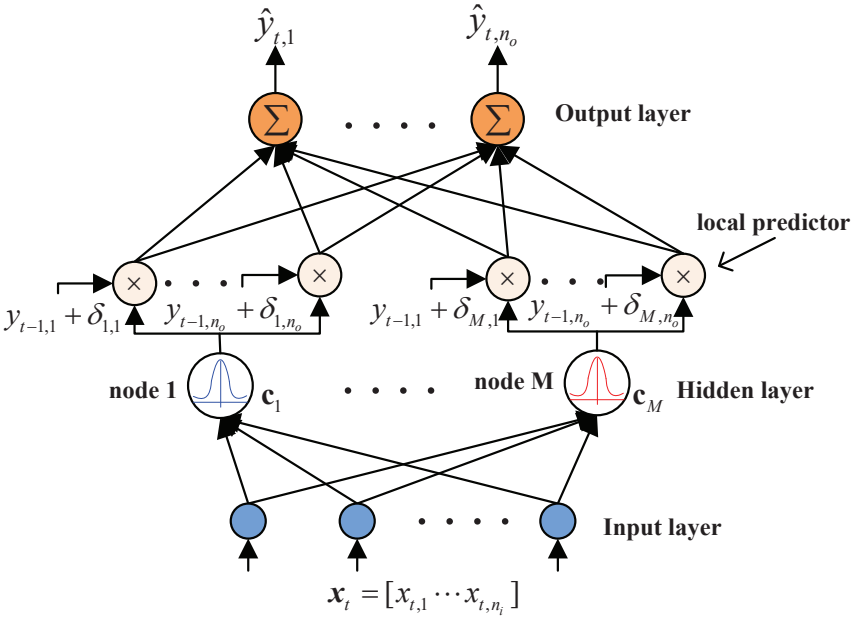

Figure 1: Structure of the MGRBF network.

the MGRBF's hidden layer. Observe that each MGRBF's hidden node produces the n_o local single-output predictors for the n_o process outputs, which is unlike any existing neural network whose hidden node only produces single response. Let M be the number of hidden nodes in the MGRBF network. The response of ith local predictor in the jth hidden node to the input vector $\boldsymbol{x}_t$ is given by

$$p_{j,i}(\boldsymbol{x}_t)=\exp\left(\frac{-\|\boldsymbol{x}_t-\boldsymbol{c}_j\|^2}{2\sigma^2}\right)\cdot\left(y_{t-1,i}+\delta_{j,i}\right), \tag{1}$$

for $1\le i\le n_o$ and $1\le j\le M$, where σ is the width of Gaussian kernel, which is set as the maximum Euclidean distance among the nodes [25], $\boldsymbol{c}_j\in\mathbb{R}^{n_i}$ is the node center, and $\delta_{j,i}$ is a scalar associated with the ith local predictor of the jth node. The term $\left(y_{t-1,i}+\delta_{j,i}\right)$ can be interpreted as a local one-step prediction of $y_{t,i}$ by the ith local predictor. Its physical interpretation is that if the input $\boldsymbol{x}_t$ is similar to the jth center $\boldsymbol{c}_j$, the value of the jth Gaussian function is close to 1 and all the local predictors $\left(y_{t-1,i}+\delta_{j,i}\right)$ for $1\le i\le n_o$ become fully active.

The MGRBF network produces the model output vector as $\widehat{\boldsymbol{y}}_t=\left[\widehat{y}_{t,1}\cdots\widehat{y}_{t,n_o}\right]^{\mathrm{T}}$ $\in\mathbb{R}^{n_o}$, which is the n_o linear combinations of the M hidden nodes' responses.

To be specific, the ith output is calculated as

$$\widehat{y}_{t,i} = \sum_{j=1}^{M} \boldsymbol{p}_{t,j}^{\mathrm{T}} \boldsymbol{\theta}_{i,j}, \tag{2}$$

where $\boldsymbol{p}_{t,j} = \left[p_{j,1}(\boldsymbol{x}_t) \cdots p_{j,n_o}(\boldsymbol{x}_t)\right]^{\mathrm{T}} \in \mathbb{R}^{n_o}$ is the response vector of the jth node, and $\boldsymbol{\theta}_{i,j} = \left[\theta_{i,j,1} \cdots \theta_{i,j,n_o}\right]^{\mathrm{T}}$ denotes the connection weights from the jth hidden node's response vector to the ith output node. More concisely, the overall output vector produced by the n_o output nodes is given by

$$\widehat{\boldsymbol{y}}_t = \boldsymbol{\Theta}_{\bar{M} \times n_o}^{\mathrm{T}} \overline{\boldsymbol{p}}_{\bar{M},t}, \tag{3}$$

where $\overline{\boldsymbol{p}}_{\bar{M},t} = \left[\boldsymbol{p}_{t,1}^{\mathrm{T}} \cdots \boldsymbol{p}_{t,M}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{\bar{M}}$ denotes the overall hidden layer's response vector with $\bar{M} = n_o M$, and $\boldsymbol{\Theta}_{\bar{M} \times n_o} \in \mathbb{R}^{\bar{M} \times n_o}$ is the overall output layer's connection matrix.

The training of the MGRBF network can be formulated as the problem of selecting an M-term subset model $\{\boldsymbol{c}_j, \boldsymbol{\delta}_j\}_{j=1}^{M}$ from the full N-term model $\{\boldsymbol{x}_t, \boldsymbol{d}_t\}_{t=1}^{N}$, where $\boldsymbol{\delta}_j = \left[\delta_{j,1} \cdots \delta_{j,n_o}\right]^{\mathrm{T}}$, N is the number of training samples, and $\boldsymbol{d}_t = \boldsymbol{y}_t - \boldsymbol{y}_{t-1}$ is the process output gradient. Because of the unique geometric property of the MGRBF hidden node, that is, each node provides multiple responses to the node input, the existing subset selection techniques cannot be directly applied to solve this problem. To address this difficulty, the work [27] proposed a two-step training procedure. First, the appropriate centers are selected from the training data set $\{\boldsymbol{x}_t; \boldsymbol{y}_t\}_{t=1}^{N}$ using the well-known OLS algorithm [17, 18]. With the selected centers $\{\boldsymbol{c}_j = \boldsymbol{x}_{t_j}\}_{j=1}^{M}$, their associated scalar vectors are then assigned to $\{\boldsymbol{\delta}_j = \boldsymbol{d}_{t_j}\}_{j=1}^{M}$ to complete the MGRBF's hidden layer. The output weight matrix $\boldsymbol{\Theta}_{\bar{M} \times n_o}$ of this constructed M-node MGRBF network is finally solved by the regularized least square (LS) estimation. The detailed MGRBF model construction procedure can be found in [27].

From this training procedure, an important physical property of the MGRBF hidden node can easily be inferred. Since in training, the jth hidden node's center is chosen as $\boldsymbol{c}_j = \boldsymbol{x}_{t_j}$ and its scalar vector is set to $\boldsymbol{\delta}_j = \boldsymbol{y}_{t_j} - \boldsymbol{y}_{t_j 1}$, it

is then obvious that the response of the jth hidden node to the input $\boldsymbol{x}_{t_j}$ is exactly $\boldsymbol{y}_{t_j}$. Furthermore, during the prediction operation, if the system input $\boldsymbol{x}_t$ is close to the jth hidden node's center $\boldsymbol{c}_j$, the response of the jth hidden node will be close to $\boldsymbol{y}_t$, i.e., an accurate prediction of the process output $\boldsymbol{y}_t$.

Owing to the capability of multiple local predictions for its hidden nodes, the MGRBF network is capable of modeling the multivariate nonstationary data well. In particular, online adaptation of the MGRBF network imposes very low computational complexity, and therefore it is highly efficient for online tracking applications [27]. In addition to nonstationary characteristics, industrial process data are also massive and high dimensional. The shallow network structure of MGRBF is less capable of capturing the complex dynamics from high-dimensional massive data, compared with deep neural networks. In what follows, we extend this shallow MGRBF network to a deep neural network, so as to provide it with deep feature learning and dimensional reduction capacity for dealing with high-dimensional massive process data.

3. Proposed Deep MGRBF Network

A prominent feature of deep learning models is to learn hierarchical feature representations from raw high-dimensional data. This is often achieved by stacking multilevel feature extraction layers, such as the SAE [8]. From an industrial process modeling perspective, it is vital to learn quality-relevant features from the raw process measurements with the guidance of quality data. Note that in our industrial process modeling, the quality data are the current process output measurement. This idea has attracted a lot of attention in both statistical machine learning and deep learning methods for process data analytics and modeling. For example, the PLS is such a counterpart of principle component analysis for quality-related feature learning [28]. The works of [35, 36, 37, 38] integrate quality information into the SAE to largely improve the predictive performance of soft sensor modeling. However, all the these methods are unable to track the changing process dynamics in nonstationary industrial environments.

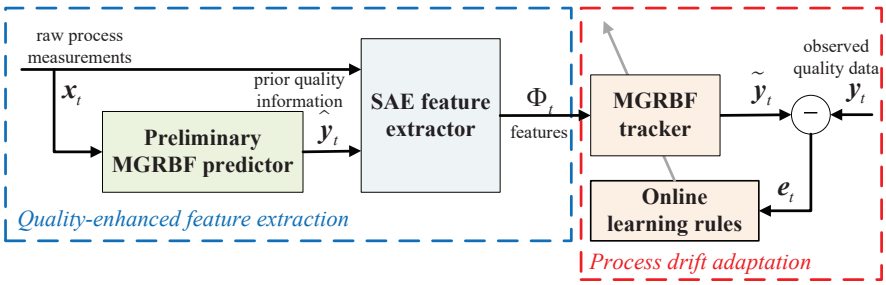

Figure 2: Schematic diagram of adaptive deep MGRBF network.

3.1. Architecture of Deep MGRBF Network

To better adapt our model to industrial processes, we propose to integrate MGRBF network with SAE to form a novel deep learning model with both quality-related feature extraction and online adaptation capacity. Specifically, a MGRBF predictor is first employed to provide a preliminary prediction of the target outputs. This prior quality information is combined with the original input data to form a new input vector to the SAE. Through layer-wise feature extraction, the SAE extracts the reduced-dimension quality-related deep features, which is then fed into a MGRBF tracker to adaptively track nonstationary process dynamics. Therefore, this novel deep neural network consists of three modules: a preliminary MGRBF predictor, a SAE feature extractor, and an adaptive MGRBF tracker, which are connected in series as shown in Fig. 2. After training, the parameters and structures of the first two modules are fixed, and during online operation we only adapt the third module, namely, the MGRBF tracker, online for tracking purpose. This ensures that our model with very deep architecture costs very little online computation for model adaptation. From the online learning perspective, it is computationally too expensive to update the whole network structure, and the MGRBF tracker itself is sufficient to handle the process drifts.

With applications to nonlinear and nonstationary industrial process modeling, therefore, our proposed adaptive deep MGRBF network operates in two phases, namely, initial training and online adaptive modeling.

1. During initial training, we have a set of historical process input and output measurements to form the training set. Given the training data, the three

modules of the deep MGRBF network are trained in sequence. First, the preliminary MGRBF predictor is constructed. Then the SAE feature extractor is trained. Finally, the adaptive MGRBF tracker is constructed.

2. During online adaptive modeling, the preliminary MGRBF predictor and the SAE feature extractor are fixed and they are used to provide deep quality-related features, which are fed into the adaptive MGRBF tracker to produce the final prediction of the process output. Then the weights and structure of the adaptive MGRBF tracker are updated according to the current process dynamics as measured by its prediction performance.

The following subsections detail these two phases of operations.

3.2. Construction of Deep MGRBF Network

We now discuss how to construct the three modules of the proposed adaptive deep MGRBF network during training.

3.2.1. Construction of Preliminary MGRBF Predictor

During training, a compact M-term MGRBF network is first constructed from the training set $\left\{\boldsymbol{x}_t; \boldsymbol{d}_t, \boldsymbol{y}_t\right\}_{t=1}^{N_{\rm tr}}$ using the two-step training procedure [27], where $N_{\rm tr}$ is the number of training samples. The trained MGRBF predictor produces the preliminary prediction $\left\{\widehat{\boldsymbol{y}}_t\right\}_{t=1}^{N_{\rm tr}}$ of the process outputs $\left\{\boldsymbol{y}_t\right\}_{t=1}^{N_{\rm tr}}$.

3.3. Construction of SAE

In order to obtain quality-related features, the SAE needs the target process output vector $\boldsymbol{y}_t$ as part of its input but this current process output is unavailable. The preliminary MGRBF predictor is used to provide a preliminary process output prediction $\widehat{\boldsymbol{y}}_t$ for the SAE as a substitute to this quality data. Specifically, the preliminary prediction $\widehat{\boldsymbol{y}}_t$ is combined with the original input data to form a new input vector

$$\boldsymbol{x}_t' = \left[\widehat{\boldsymbol{y}}_t^{\rm T} \ \boldsymbol{x}_t^{\rm T}\right]^{\rm T} \in \mathbb{R}^{n_o+n_i}. \tag{4}$$

The new training set $\{\boldsymbol{x}'_t\}_{t=1}^{N_{\mathrm{tr}}}$ that contains important quality information is fed into the SAE so as to learn the corresponding quality-related features. The SAE consists of the hierarchically stacked multiple AEs. Each AE is an unsupervised self-learning network with encoder and decoder.

The input vector $\boldsymbol{x}'_t$ (4) is projected onto the first AE's hidden layer $\boldsymbol{\phi}_t = \left[\phi_1(\boldsymbol{x}'_t) \cdots \phi_s(\boldsymbol{x}'_t)\right]^{\mathrm{T}}$ by the nonlinear mapping $\boldsymbol{f}$ as

$$\boldsymbol{\phi}_t = \boldsymbol{f}\left(\boldsymbol{W}_1 \boldsymbol{x}'_t + \boldsymbol{b}_1\right) \tag{5}$$

where s is the size of hidden layer, $\boldsymbol{W}_1$ and $\boldsymbol{b}_1$ are the weight matrix and bias vector, respectively, from the input layer to the hidden layer. The decoder reconstructs the input vector $\boldsymbol{x}'_t$ by mapping $\boldsymbol{\phi}_t$ onto the output layer as

$$\widetilde{\boldsymbol{x}}'_t = \widetilde{\boldsymbol{f}}\left(\widetilde{\boldsymbol{W}_1}\boldsymbol{\phi}_t + \widetilde{\boldsymbol{b}_1}\right), \tag{6}$$

where $\widetilde{\boldsymbol{f}}$ is the output layer's nonlinear mapping, $\widetilde{\boldsymbol{W}_1}$ and $\widetilde{\boldsymbol{b}_1}$ are the connecting weight matrix and bias vector, respectively, from the hidden layer to the output layer. The AE aims to learn a mapping $\boldsymbol{F}(\boldsymbol{x}'_t) = \widetilde{\boldsymbol{f}}\left(\boldsymbol{f}(\boldsymbol{x}'_t)\right) \approx \boldsymbol{x}'_t$ that makes the reconstruction error between $\widetilde{\boldsymbol{x}}'_t$ and $\boldsymbol{x}'_t$ as small as possible. This can be formulated as an optimization problem to minimize the following mean squared reconstructed error

$$J_{\mathrm{unsup}}\left(\boldsymbol{W}_1, \widetilde{\boldsymbol{W}_1}, \boldsymbol{b}_1, \widetilde{\boldsymbol{b}_1}\right) = \frac{1}{2N_{\mathrm{tr}}} \sum_{t=1}^{N_{\mathrm{tr}}} \left\|\widetilde{\boldsymbol{x}}'_t - \boldsymbol{x}'_t\right\|^2. \tag{7}$$

This optimization can be solved by a gradient descend algorithm, yielding the first AE's hidden layer features $\boldsymbol{\phi}_{\mathrm{AE},1}$ as well as the encoder's weights and bias $\{\boldsymbol{W}_1, \boldsymbol{b}_1\}$. After the first AE is trained, its hidden layer parameters $\{\boldsymbol{W}_1, \boldsymbol{b}_1\}$ are fixed, and the obtained hidden layer features $\boldsymbol{\phi}_{\mathrm{AE},1}$ serve as the input to the second AE. Then, the second AE is trained to obtain its hidden layer parameters $\{\boldsymbol{W}_2, \boldsymbol{b}_2\}$ and the associated features $\boldsymbol{\phi}_{\mathrm{AE},2}$. In a progressive way, the whole SAE is pre-trained layer by layer until the last (nth) AE is obtained.

After the above unsupervised pre-training, supervised fine-tuning is carried out. A linear regression layer with n_o output neurons having the weight matrix $\boldsymbol{W}_{\rm o}$ and bias vector $\boldsymbol{b}_{\rm o}$ is added on the top of the SAE to produce the prediction $\widetilde{\boldsymbol{y}}_t$ of the process output vector $\boldsymbol{y}_t$. The entire network is fine-tuned by the back propagation with the training data $\{\boldsymbol{x}'_t; \boldsymbol{y}_t\}_{t=1}^{N_{\rm tr}}$ based on the cost function

$$J_{\rm sup}\big(\boldsymbol{W}_{\rm o}, \boldsymbol{b}_{\rm o}, \boldsymbol{W}_i, \boldsymbol{b}_i, 1 \le i \le n\big) = \frac{1}{2N_{\rm tr}} \sum_{t=1}^{N_{\rm tr}} \|\widetilde{\boldsymbol{y}}_t - \boldsymbol{y}_t\|^2, \tag{8}$$

with the pre-trained SAE's parameters used to initialize the hidden layers $\{\boldsymbol{W}_i, \boldsymbol{b}_i\}_{i=1}^{n}$ of the supervised SAE. After the SAE is trained, the extracted quality-related features $\{\boldsymbol{\Phi}_t\}_{t=1}^{N_{\rm tr}}$ can be obtained from the last AE. Then, the regression output layer is removed, and the last AE is connected to the MGRBF tracker.

3.3.1. Construction of MGRBF Tracker

Given the training data $\big\{\boldsymbol{\Phi}_t; \boldsymbol{d}_t, \boldsymbol{y}_t\big\}_{t=1}^{N_{\rm tr}}$, where $\boldsymbol{\Phi}_t$ is the extracted features by the SAE, a compact M-term MGRBF model is constructed as the MGRBF tracker using the two-step training procedure [27]. The trained MGRBF Tracker produces the final prediction $\big\{\widetilde{\boldsymbol{y}}_t\big\}_{t=1}^{N_{\rm tr}}$ of the process outputs $\big\{\boldsymbol{y}_t\big\}_{t=1}^{N_{\rm tr}}$.

Remark 1. *In traditional SAE, a simple linear regression layer is added on the top of the SAE for online prediction and adaptive modeling, in which the output regression layer weights are either fixed or simply updated by the RLS algorithm. Since the fixed SAE latent space is capable of extracting the compressed nonlinear features from raw data, an adaptive linear regression layer using the RLS algorithm is sufficient to track the slowly time-varying processes. However, when the process exhibits severe nonstationarity and has multiple outputs, a simple linear layer is unable to track the fast time-varying process dynamics and modeling the coupling effects of multiple outputs well. Hence, we replace the linear output layer with a stronger MGRBF tracker to deal with this problem.*

3.4. Online Tracking of Changing Process Dynamics

After training, the parameters and structures of the MGRBF preliminary predictor and the SAE feature extractor are fixed during online operation, while the MGRBF tracker is adapted to track the fast time-varying dynamics between the extracted features and the process outputs.

3.4.1. Online Prediction

During online operation, the newly observed process input measurement vector $\boldsymbol{x}_{p_t}$ at sampling time t is inputed into the preliminary MGRBF predictor to obtain a preliminary prediction $\widehat{\boldsymbol{y}}_{p_t}$ of the process output $\boldsymbol{y}_{p_t}$. This preliminary prediction $\widehat{\boldsymbol{y}}_{p_t}$ is combined with the raw input observation $\boldsymbol{x}_{p_t}$ to form a new input vector $\boldsymbol{x}'_{p_t} = \left[\widehat{\boldsymbol{y}}_{p_t}^{\mathrm{T}}\ \boldsymbol{x}_{p_t}^{\mathrm{T}}\right]$ to the SAE. Through forward propagation from the first feature layer to the last one, the deep quality-related features are extracted at the last layer as $\boldsymbol{\Phi}_{p_t}$, which is used as the input to the MGRBF tracker for it to produce the final prediction $\widetilde{\boldsymbol{y}}_{p_t}$ of the process output $\boldsymbol{y}_{p_t}$.

Specifically, the quality-related feature $\boldsymbol{\Phi}_{p_t}$ at sample t serves as the input to the MGRBF tracker, whose hidden layer response vector $\overline{\boldsymbol{p}}_{\bar{M},p_t}$ is calculated by (1). Then the MGRBF tracker produces the prediction according to (3) as

$$\widetilde{\boldsymbol{y}}_{p_t} = \boldsymbol{\Theta}_{\bar{M}\times n_o,t-1}^{\mathrm{T}}\overline{\boldsymbol{p}}_{\bar{M},p_t}, \tag{9}$$

where $\boldsymbol{\Theta}_{\bar{M}\times n_o,t-1}$ is the weight matrix obtained at sampling time $t-1$, $\widetilde{\boldsymbol{y}}_{p_t}$ is the final prediction of the process output vector $\boldsymbol{y}_{p_t}$.

3.4.2. Online Adaptation

After performing the online prediction, the structure and parameters of the MGRBF tracker is updated according to its prediction performance. To be specific, when the measurement of the true process output vector $\boldsymbol{y}_{p_t}$ becomes available, we measure the entire deep MGRBF network's prediction performance by the normalized prediction output error as

$$\widetilde{e}_{p_t} = \|\boldsymbol{y}_{p_t} - \widetilde{\boldsymbol{y}}_{p_t}\|^2 / \|\boldsymbol{y}_{p_t}\|^2. \tag{10}$$

Based on this metric, we update the MGRBF tracker according to the following criterion

$$\begin{cases} \text{if } \widetilde{e}_{p_t} < \varepsilon: \text{ weight adaptation only,} \\ \text{if } \widetilde{e}_{p_t} \geq \varepsilon: \text{ tunable node adaptation,} \end{cases} \tag{11}$$

where ε is a pre-set threshold, which determines the frequency of node replacement. The two adaptation modes are elaborated below.

Weight Adaptation Only: When $\widetilde{e}_{p_t} < \varepsilon$, the process varies slowly and the current model structure is still sufficient to capture the underlying dynamics. Hence, we simply update the weight matrix of the MGRBF tracker using the RLS algorithm

$$\begin{cases} \boldsymbol{g}_t = \boldsymbol{\Gamma}_{t-1}\overline{\boldsymbol{p}}_{\bar{M},p_t}\left(\gamma + \overline{\boldsymbol{p}}_{\bar{M},p_t}^{\mathrm{T}}\boldsymbol{\Gamma}_{t-1}\overline{\boldsymbol{p}}_{\bar{M},p_t}\right)^{-1}, \\ \boldsymbol{\Gamma}_t = \left(\boldsymbol{\Gamma}_{t-1} - \boldsymbol{g}_t\overline{\boldsymbol{p}}_{\bar{M},p_t}^{\mathrm{T}}\boldsymbol{\Gamma}_{t-1}\right)\gamma^{-1}, \\ \boldsymbol{\Theta}_{\bar{M}\times n_o,t} = \boldsymbol{\Theta}_{\bar{M}\times n_o,t-1} + \boldsymbol{g}_t\boldsymbol{e}_{p_t}^{\mathrm{T}}, \end{cases} \tag{12}$$

where $\boldsymbol{e}_{p_t} = \boldsymbol{y}_{p_t} - \widetilde{\boldsymbol{y}}_{p_t}$ is the prediction error, $\boldsymbol{g}_t \in \mathbb{R}^{\bar{M}}$ is the Kalman gain vector, $0.9 \leq \gamma < 1$ is the forgetting factor, and $\boldsymbol{\Gamma}_t \in \mathbb{R}^{\bar{M}\times\bar{M}}$ is the inverse of the covariance matrix which is usually initialized to $\boldsymbol{\Gamma}_0 = \vartheta\boldsymbol{I}_{\bar{M}}$ with ϑ being a large positive constant and $\boldsymbol{I}_{\bar{M}}$ being the $\bar{M} \times \bar{M}$ identity matrix.

Tunable Node Adaptation: When $\widetilde{e}_{p_t} \geq \varepsilon$, the MGRBF tracker performs poorly and the RLS weight adaptation itself is insufficient for tracking fast time-varying process characteristics. Thus the current model structure is updated. To be specific, the worst node with the least contribution to the overall performance is replaced with a new one. The contribution of a node is measured by its sum of squared weighted local predictor response, which is defined by

$$contri_j = \sum_{i=1}^{n_o}\left(\boldsymbol{p}_{t,j}^{\mathrm{T}}\boldsymbol{\theta}_{i,j}^{t-1}\right)^2, \ 1 \leq j \leq M, \tag{13}$$

where $\boldsymbol{p}_{t,j}$ is the jth hidden node's response vector to the input $\boldsymbol{\Phi}_{p_t}$, and $\boldsymbol{\theta}_{i,j}^{t-1}$ is the connection weight vector from the jth hidden node to the ith output node,

obtained at $t-1$. We find the node with the smallest *contri*

$$m = \arg \min_{1 \le j \le M} contri_j, \tag{14}$$

and replace it by a new node, whose center $\boldsymbol{c}_m$ and scalars $\boldsymbol{\delta}_m$ can be determined by exploiting the geometric property of MGRBF hidden node. Specifically, we set $\boldsymbol{c}_m = \boldsymbol{\Phi}_{p_t}$ and $\boldsymbol{\delta}_m = \boldsymbol{y}_{p_t} - \boldsymbol{y}_{p_t-1}$ to ensure that the new replacement node m encodes the newest feature state and is a perfect local multi-output predictor of $\boldsymbol{y}_{p_t}$. Since the set of centers contain a new one, the Gaussian width σ is updated based on the new maximum Euclidean distance among the centers.

After the new node is determined, the weight matrix of the updated MGRBF tracker is recalculated by the regularized LS method based on the q latest data points $\{\boldsymbol{\Phi}_{p_t-i}; \boldsymbol{y}_{p_t-i}\}_{i=0}^{q-1}$ as

$$\boldsymbol{\Theta}_{\bar{M} \times n_o, t} = \left(\boldsymbol{P}_{q \times \bar{M}, t}^{\mathrm{T}} \boldsymbol{P}_{q \times \bar{M}, t} + \lambda \boldsymbol{I}_{\bar{M}}\right)^{-1} \boldsymbol{P}_{q \times \bar{M}, t}^{\mathrm{T}} \boldsymbol{Y}_{q \times n_o, t}, \tag{15}$$

where λ is a very small positive regularization parameter, $\boldsymbol{Y}_{q \times n_o, t} = \left[\boldsymbol{y}_{p_t}\ \boldsymbol{y}_{p_t-1} \cdots \boldsymbol{y}_{p_t-q+1}\right]^{\mathrm{T}}$, and $\boldsymbol{P}_{q \times \bar{M}, t} = \left[\overline{\boldsymbol{p}}_{\bar{M}, p_t}\ \overline{\boldsymbol{p}}_{\bar{M}, p_t-1} \cdots \overline{\boldsymbol{p}}_{\bar{M}, p_t-q+1}\right]^{\mathrm{T}}$. The number of the latest data q trades off estimation accuracy and tracking performance. To guarantee a smooth transition from tunable node adaptation to weight adaptation only, the inverse covariance matrix in the RLS algorithm is reinitialized to

$$\boldsymbol{\Gamma}_t = \left(\boldsymbol{P}_{q \times \bar{M}, t}^{\mathrm{T}} \boldsymbol{P}_{q \times \bar{M}, t} + \lambda \boldsymbol{I}_{\bar{M}}\right)^{-1}. \tag{16}$$

Note that in order to track fast time-varying characteristics, q should be very small and typically we have $q \ll \bar{M}$. Therefore, the regularization is necessary.

Remark 2. *Adapting the proposed deep MGRBF network online is achieved by adapting its MGRBF tracker online as given in Subsection 3.4.2. In the weight adaptation only, the computational complexity comes from the RLS algorithm (12), which is on the order of* $O(\bar{M}^2)$*, while in the tunable node adaptation, the computational complexity is dominated by the regularized LS estimator (15),*

which is on the order of $O(\bar{M}^3)$. *Therefore, the complexity per sample of the proposed adaptive deep MGRBF network is no more than* $O(\bar{M}^3)$. *This is clearly affordable since* $\bar{M} = n_o M$ *is typically small, where* n_o *is the number of the process outputs and* M *is the number of hidden nodes in the MGRBF tracker.*

It can also be inferred that the real online computational complexity of the proposed adaptive deep MGRBF network is less than that of the very efficient shallow adaptive MGRBF network of [27]. This can be explained as follows. Online adaptation operation of the adaptive MGRBF network [27] involves a MGRBF network of M *hidden nodes (assuming the same as the MGRBF tracker in the deep MGRBF network) with the input* $\boldsymbol{x}_{p_t}$, *while online adaptation operation of the proposed deep MGRBF network involves a MGRBF network of* M *hidden nodes with the input* $\boldsymbol{\Phi}_{p_t}$. *Owing to the excellent nonlinear dimensional reduction capability of the SAE, the dimension of the feature vector* $\boldsymbol{\Phi}_{p_t}$ *is much smaller than the dimension of the original input vector* $\boldsymbol{x}_{p_t}$. *Based on this fact, it is not difficult to draw the above conclusion. We will confirm this analysis with the industrial applications of the next section.*

Remark 3. *As with all the known sample-by-sample adaptive strategies or models, given the process input* $\boldsymbol{x}_{p_t}$ *at each sampling instance* p_t, *our adaptive deep MGRBF network first produces the prediction* $\widetilde{\boldsymbol{y}}_{p_t}$ *of the process output* $\boldsymbol{y}_{p_t}$. *Later, after the arrival of the true process output measurement* $\boldsymbol{y}_{p_t}$, *the adaptation of the deep MGRBF network takes place. This implies that the acquisition of the process output measurement must be timely in order to take the full advantage of this sample-by-sample adaptation strategy. For some industrial processes, however, acquisition of the process output measurement can be seriously delayed. In such scenarios, all the known sample-by-sample adaptive models, including our adaptive deep MGRBF network, will be unable to perform adaptation timely after the main prediction operation at each sampling instance. Without timely adaptation to track the time-varying characteristics of the underlying process, prediction performance will degrade considerably during the online operation course of the process.*

An alternative adaptation paradigm that does not rely on supervised learning strategy is called for nonlinear and nonstationary industrial processes with delayed process output measurement. Specifically, it is highly desirable to investigate whether some kind of unsupervised model adaptation is possible based only on the process input data. Presently, we do not know how to achieve this extremely challenging adaptive strategy. Clearly it is beyond the scope of this paper and much research is warranted to investigate this potential direction.

4. Industrial Applications

Two industrial case studies, soft sensing for penicillin fermentation process and online identification of a real-world industrial microwave heating process, are carried out to verify the effectiveness of our adaptive deep MGRBF network. Three metrics, the mean square error (MSE), the determinant of the error covariance $\log(\det(\mathrm{Cov}(\boldsymbol{E})))$ [21] and the coefficient of determination (R^2), are utilized to evaluate each output and multi-output modeling performance. Since each output has an R^2 value, the averaged R^2 over all the outputs is used to evaluate the multi-output modeling performance. The online computation complexity is quantified by the averaged computation time per sample (ACTpS).

The proposed method is compared with the state-of-the-art multi-output modeling approaches, including the PLS [28], the multi-output RBF network [21], the multi-output GRBF network of Section 2, the multi-output TRBF network, which is a multi-output extension to [23], the multi-output AGRBF network [27], and the multi-output GAP-SER algorithm [16]. In addition, two deep learning models, the SAE [35, 36, 37, 38] and the LSTM [39, 40], are also used for comparison. Note that the original SAE is a nonadaptive model. To provide it with some adaptability, we adapt the weights of its output regression layer online using the RLS algorithm. This adaptive SAE, denoted as $\mathrm{SAE}_{\mathrm{RLS}}$, is used as the third deep learning model benchmark. The PLS, RBF network and GRBF network are fixed during online operation. All the benchmark methods are multi-output modeling methods.

For all the adaptive models, the forgetting factor of the RLS algorithm is set to $\gamma = 0.98$. The regularization parameter is set to $\lambda = 0.001$. For all the deep learning models, the learning rate and the number of training epochs are set empirically to 0.01 and 200, respectively. Other hyperparameters are chosen carefully and empirically, as detailed in the two case studies.

4.1. Soft Sensing for Penicillin Fermentation Process

The penicillin fermentation process is an industrial biochemical fed-batch process, which has been widely adopted for performance assessment of adaptive soft sensors [12, 13]. During different stages of fermentation, there are hyphae growth, reproduction, aging, synthesis, and hydrolysis of penicillin, making the process nonlinear and nonstationary. For our soft sensor modeling, the penicillin concentration, biomass concentration and substrate concentration are selected as the process outputs, while the other 10 process variables are used as the process inputs. The detailed process data description can be found in [13]. In our experiments, four batches of process data were generated using the PenSim tool [41] with different operating conditions. Each batch is composed of 400 samples, and the entire dataset has 1600 samples. The first two batches are used for initial training, and the rest two batches are for online adaptive modeling.

The structures of all the models are carefully chosen by trial and error. Specifically, the number of latent variables for PLS is set to 4 to attain its best performance. The size of the RBF, GRBF, TRBF and AGRBF networks are all set to 10, as suggested in [27]. For our method, we set the sizes of the two MGRBF predictors both to 10, while its SAE unit contains three layers having the numbers of neurons $\{8, 6, 4\}$, respectively. We set the node replacement threshold $\varepsilon = 0.1$ and the bandwidth $q = 1$ for the MGRBF tracker. For the TRBF and AGRBF benchmarks, the node replacement thresholds are chosen to be 0.1 and 0.01, respectively. For the GAP-SER, we set the window size $W = 150$, bandwidth $q = 20$ and threshold $\xi = 0.9$ to best trade off the prediction accuracy and online complexity [15, 16]. The number of hidden nodes for the LSTM is set to 128, which is obtained by a grid-search, while the structures of

Table 1: Test performance comparison of various methods for soft sensing of penicillin fermentation process.

Methods	Model type	MSE (dB)			log(det(Cov($\boldsymbol{E}$))) (dB)	averaged R^2	ACTpS (ms)
		y_1	y_2	y_3			
PLS	Fixed	-20.4131	-32.7461	-18.2955	-8.8180	0.9292	NA
RBF	Fixed	-24.6497	-14.3837	-19.7832	-7.0354	0.8360	NA
TRBF	Adaptive	-28.6030	-36.7634	**-35.1201**	-11.1485	0.9943	0.0780
GRBF	Fixed	**-37.4990**	-29.1443	-30.2648	-11.0040	0.9927	NA
AGRBF	Adaptive	**-39.8615**	**-37.9934**	**-35.2746**	**-12.2161**	**0.9983**	**0.0296**
GAP-SER	Adaptive	-28.7491	**-81.2151**	-30.7240	**-15.3111**	**0.9936**	4.3732
LSTM	Fixed	-29.3219±3.0994	-27.2901±2.7164	-22.2428±1.8730	-9.3079±0.2651	0.9696±0.0169	NA
SAE	Fixed	-28.2688±9.0967	-28.0394±5.6297	-21.6508±5.8354	-9.4044±1.3017	0.9296±0.0812	NA
SAE_{RLS}	Adaptive	-28.0778±11.4203	-32.3667±7.6443	-30.4218±7.4376	-10.6432±1.4741	0.9359±0.1174	**0.0036**
Proposed	Adaptive	**-46.9541±3.5820**	**-49.9950±4.6632**	**-53.0888±7.7268**	**-17.1598±0.8739**	**0.9998±0.0002**	**0.0221**

the SAE and adaptive SAE are identical to the SAE unit in our method.

The test performance attained by 10 models are compared in Table 1. Because all the deep models, the LSTM, the SAE, the SAE_{RLS} and the SAE unit in our proposed method, involve random initialization of network weights, we run 10 independent experiments, and report their means and standard deviations of modeling accuracy and computation time. It can be seen that the adaptive models TRBF, AGRBF and SAE_{RLS} outperform their nonadaptive versions RBF, GRBF and SAE, which indicates that the process is time-varying. The three deep learning models, the LSTM, SAE and SAE_{RLS}, are inferior to the shallow GRBF network, because the GRBF has better predictive capacity for nonstationary data, while these deep models lack such capacity. Our method achieves the best online modeling accuracy, as evidenced by the smallest log(det(Cov($\boldsymbol{E}$))) and average R^2. The GAP-SER attains the second-best performance as measured by log(det(Cov($\boldsymbol{E}$))) and it achieves the highest prediction accuracy of y_2. However, it is computationally expensive and imposes the largest ACTpS. The AGRBF with no feature extraction capacity attains the third best prediction accuracy, but its prediction performance is 5 dB worst than our method as measured by log(det(Cov($\boldsymbol{E}$))). Notably our adaptive deep MGRBF with deep architecture imposes a lower ACTpS than the shallow AGRBF network, which demonstrates its high efficiency (See *Remark 2*).

Define the error covariance at test sample t as

$$\text{Cov}(\boldsymbol{E}_t) = \frac{1}{t-1}\sum_{i=1}^{t}\left(\boldsymbol{e}_i - \bar{\boldsymbol{e}}\right)\left(\boldsymbol{e}_i - \bar{\boldsymbol{e}}\right)^{\text{T}}, \tag{17}$$

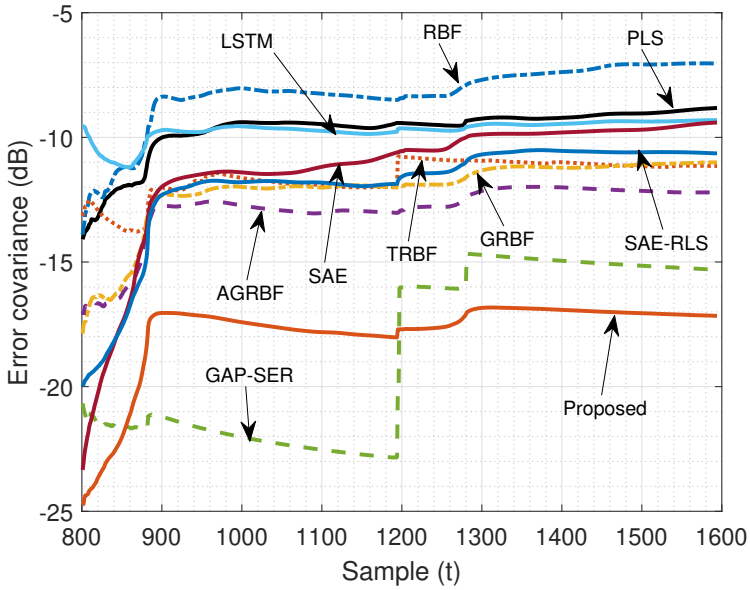

Figure 3: Comparison of test $\log(\det(\mathrm{Cov}(\boldsymbol{E}_t)))$ learning curves of various models for soft sensing of penicillin fermentation process.

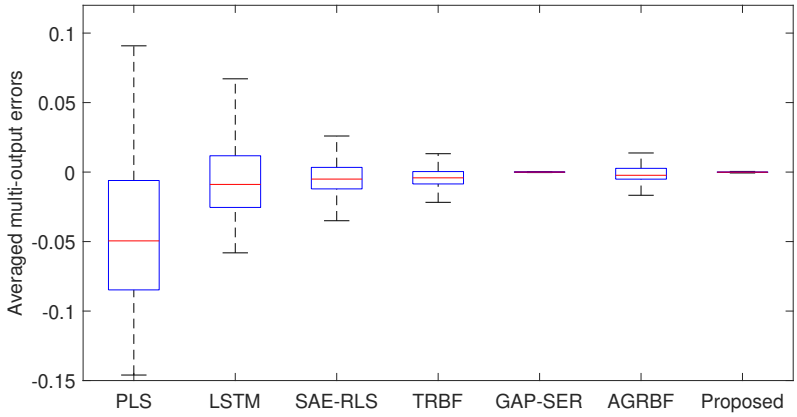

Figure 4: Box plots of the averaged multi-output prediction errors for penicillin fermentation process by 7 methods.

where $\boldsymbol{e}_i = \boldsymbol{y}_i - \widetilde{\boldsymbol{y}}_i$ is the prediction error at test sample i and $\bar{\boldsymbol{e}}$ denotes the sample average of the prediction errors. The online $\log(\det(\mathrm{Cov}(\boldsymbol{E}_t)))$ learning curves for various models are presented in Fig. 3. As can be seen that although the GAP-SER attains smaller $\log(\det(\mathrm{Cov}(\boldsymbol{E})))$ than our method at the beginning of online operation, its performance degrades significantly at 1200 samples. This is due to the change of the process batch. By contrast, our method is immune to this change. Fig. 4 further shows the box plots of the averaged

multi-output prediction errors by the 7 models (the nonadaptive RBF, GRBF and SAE models are omitted for clear comparison). As can be seen that the GAP-SER and our proposed method have the tightest error distribution around zero among the 7 methods.

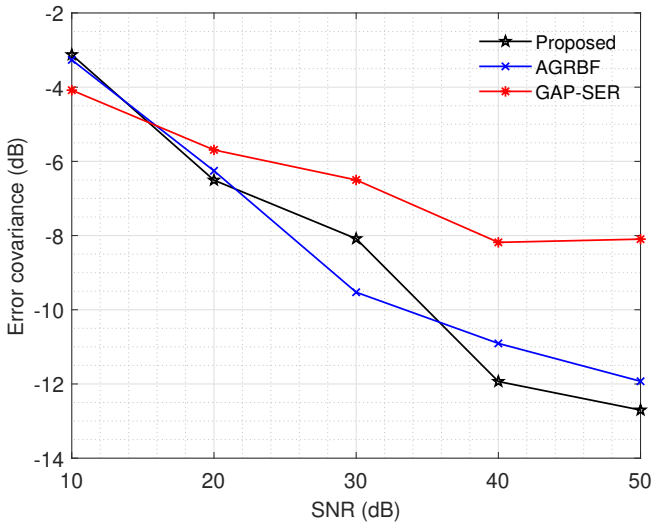

Figure 5: The effect of process output measurement noise on achievable test error covariance performance of GAP-SER, AGRBF and proposed method for penicillin fermentation process.

Next, we investigate the effect of the process output measurement noise on the online prediction performance for the three best approaches (GAP-SER, AGRBF and proposed method), which is shown in Fig. 5. As expected, the achievable online prediction performance degrade as the signal-to-noise ratio (SNR) decreases. From Fig. 5, it can be seen that our deep adaptive MGRBF network method attains the better performance than the other two adaptive methods in most cases. The results of Fig. 5 also suggest that adaptive GRBF based methods (the proposed method and AGRBF) generally outperforms the GAP-SER in noisy nonstationary environments.

4.2. Online Identification of Microwave Heating Process

Microwave heating process (MHP) is a complex nonlinear and nonstationry thermal process [42]. Unlike conventional heating method, the energy transformation in high-frequency electric field brings instantaneous temperature rise of

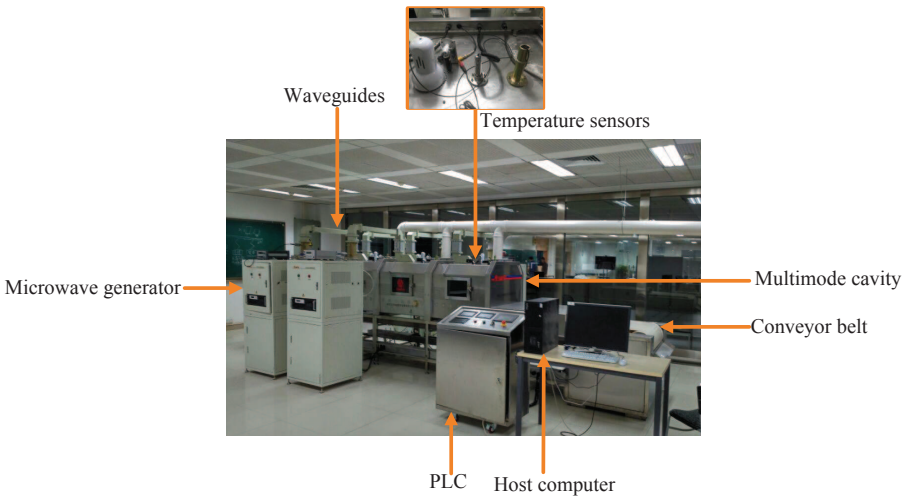

Figure 6: An real-world industrial-scale microwave heating system.

heated materials. Despite its advantageous heating efficiency, overheating and thermal runaway often occur in MHP and lead to destructive results. In order to detect thermal runaway in advance and maintain heating safety, real-time temperature estimation is essential for MHP [43, 44].

A industrial-scale microwave heating system has established in the previous work [45], which is shown in Fig. 6. The system consists of five microwave power sources with each microwave source having a maximal power of 3 kW (total 15 kW). Microwave energy is transmitted through the waveguide, fed into the cavity to heat material. The material is transported through cavity by the conveyor belt, whose speed can be adjusted. Three fiber optical sensors (FOSs), denoted as FOS1 to FOS3, are placed at three different locations to record multiple-points of temperature online. During real-time process operation, the control center receives the measured temperatures from FOSs, and sends control commends, including five microwave powers $u_{p_i}(t)$, $1 \leq i \leq 5$ and the conveyor speed $v(t)$. The control inputs to the MHP are given by

$$\boldsymbol{u}_t = \left[u_{p_1}(t)\ u_{p_2}(t)\ u_{p_3}(t)\ u_{p_4}(t)\ u_{p_5}(t)\ v(t)\right]^{\mathrm{T}}. \tag{18}$$

Each FOS measures the temperature at its location, which is the MHP's output

$y_i(t)$, $1 \leq i \leq 3$. Because of near instantaneous response of MHP, the process's temperatures $\boldsymbol{y}_t = \left[y_1(t)\ y_2(t)\ y_3(t)\right]^{\mathrm{T}}$ can be adequately modeled as

$$\boldsymbol{y}_t = \boldsymbol{f}_{\mathrm{sys}}(\boldsymbol{x}_t; t), \tag{19}$$

where $\boldsymbol{f}_{\mathrm{sys}}(\cdot; t)$ represents the unknown nonlinear time-varying system mapping with the input vector given by

$$\boldsymbol{x}_t = \left[\boldsymbol{y}_{t-1}^{\mathrm{T}}\ \boldsymbol{u}_{t-1}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^9. \tag{20}$$

3,500 process data have been collected from this MHP. The process data is first normalized before predictive modeling. We use the first 1000 samples for training, and the rest 2,500 samples for online adaptive modeling.

Again, the structures and heperparameters of all the models are chosen empirically. For our method, the size of the SAE unite is $\{7, 5, 3\}$. The threshold and bandwidth for the adaptive MGRBF tracker are set to 0.001 and 2, respectively. The AGRBF and TRBF have the same node replacement threshold. The latent variables for the PLS is set to 4. The parameters for the GAP-SER are set to $W = 110$, $q = 30$, and $\xi = 0.9$. The SAE and adaptive SAE have the same structure of $\{7, 5, 3\}$, while the LSTM has 64 hidden nodes.

The test performance comparison of various models are summarized in Table 2, and the online $\log(\det(\mathrm{Cov}(\boldsymbol{E}_t)))$ learning curves are compared in Fig. 7. In terms of prediction performance, our proposed deep model attains the best accuracy, and the AGRBF is the closed second, while the GAP-SER achieves the third best performance. In terms of online computational complexity, our

Table 2: Test performance comparison of various methods for online identification of microwave heating process.

Methods	Model type	MSE (dB)			log(det(Cov($\boldsymbol{E}$))) (dB)	averaged R^2	ACTpS (ms)
		y_1	y_2	y_3			
PLS	Fixed	-36.3661	-30.5462	-31.2921	-11.8121	0.8592	NA
RBF	Fixed	-28.6941	-29.9555	-33.7634	-10.8746	0.7978	NA
TRBF	Adaptive	-42.0098	-41.9637	-43.0211	-12.9293	0.9871	**0.0434**
GRBF	Fixed	-39.9221	-39.7723	-33.1966	-12.2785	0.9529	NA
AGRBF	Adaptive	**-45.9210**	**-46.0491**	**-46.4951**	**-13.8801**	**0.9947**	0.0501
GAP-SER	Adaptive	**-46.5108**	**-46.2364**	**-42.2596**	**-13.5019**	**0.9926**	1.8770
LSTM	Fixed	-25.4328±4.3065	-25.6605±3.1740	-24.2661±3.3113	-9.0478±0.5745	0.1200±0.6496	NA
SAE	Fixed	-28.4760±7.3149	-27.8770±5.6352	-27.6597±6.2336	-10.7618±1.1757	0.2633±0.9046	NA
$\mathrm{SAE}_{\mathrm{RLS}}$	Adaptive	-31.5634±5.4558	-33.0014±4.1909	-32.6310±3.1645	-10.8150±1.0406	0.8278±0.1077	**0.0035**
Proposed	Adaptive	**-46.1024±0.4867**	**-46.7387±0.3587**	**-46.8103±0.4730**	**-13.9698±0.0737**	**0.9952±0.0005**	**0.0284**

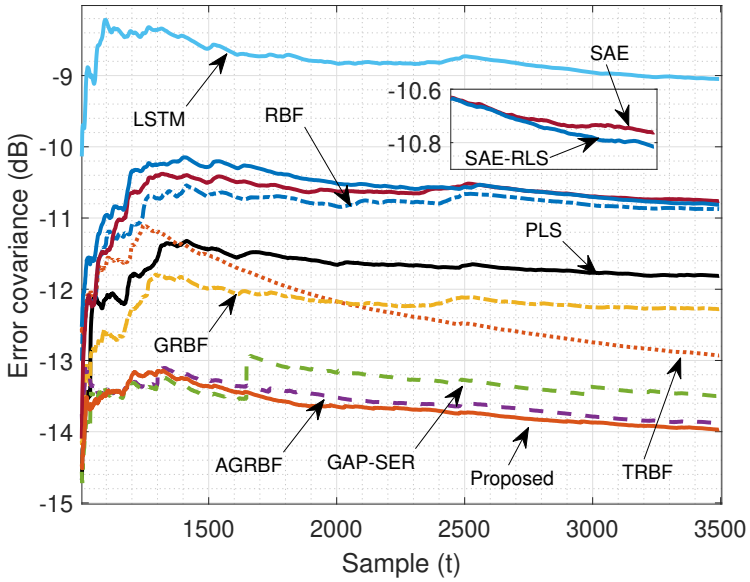

Figure 7: Comparison of test $\log(\det(\mathrm{Cov}(\boldsymbol{E}_t)))$ learning curves of various models for online identification of microwave heating process.

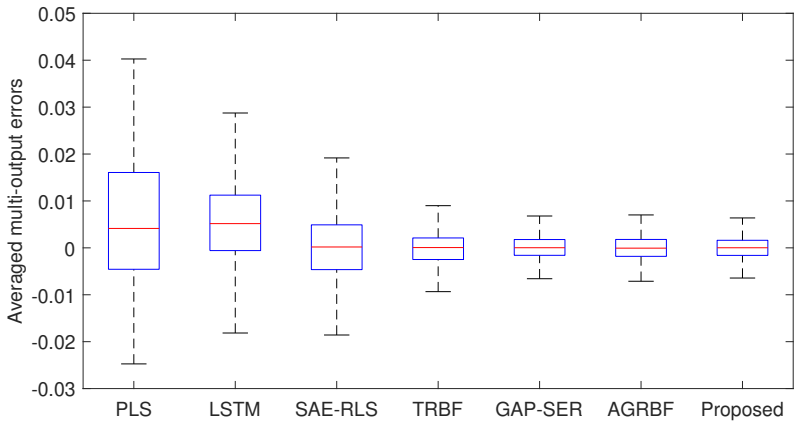

Figure 8: Box plots of the averaged multi-output prediction errors for microwave heating process by 7 methods.

method imposes a smaller ACTpS than the AGRBF, while the GAP-SER consumes the highest ACTpS. The three deep models, the LSTM, SAE and $\mathrm{SAE}_{\mathrm{RLS}}$, perform poorly, and even the PLS achieves a smaller prediction error than them. The box plots of average multi-output prediction errors by the 7 models are shown in Fig. 8. It can be seen that the proposed method, AGRBF

and GAP-SER attain the best performance among the 7 models, since their prediction errors are more densely distributed around 0.

4.3. Discussion

The experimental results of these two industrial case studies clear demonstrate that our proposed multi-output adaptive deep GRBF network has significant advantages over the existing state-of-the-art online modeling approaches as well as traditional deep learning models, in terms of both multi-output prediction accuracy and online computational complexity. Specifically, our deep model consistently achieves the highest prediction accuracy, while imposing a very small ACTpS, lower even than the shallow adaptive models.

Remark 4. *The proposed adaptive deep MGRBF network and all the benchmark modeling methods used in the above two industrial case studies are purely data-driven, and no knowledge of the underlying process is required in modeling. In the terminology of learning, this type of data-driven learning is referred to as black-box modeling. A fundamental principle in data modeling is to incorporate available a priori information regarding the underlying data generating mechanism into the data modeling process, which is known as grey-box modeling in the terminology of learning. Grey-box modeling capable of incorporating prior knowledge typically outperforms black-box modeling [46, 47, 48]. There is a scope of investigating how to incorporate prior knowledge of the underlying process into the adaptive deep MGRBF network. Since prior knowledge is process specific, such a grey-box modeling is also specific to the particular process to be modeled. However, there exist some general grey-box modeling approach for data modeling by the RBF network [46, 47], which may also be helpful for developing grey-box modeling using the GRBF network.*

In real-time process operation optimization and control applications, without an up-to-data and accurate surrogate model, there is a high risk that an industrial plant is operated in a suboptimal manner due to plant-model mismatch and unknown uncertainties [49, 50, 51]. The chief advantage of our proposed

adaptive deep MGRBF network is that it provides high prediction accuracy and fast adaptation capability as well as low online computational complexity, which makes it particularly desirable for real-time optimization/control of nonlinear and time-varying processes. Specifically similar to [52], the unknown multivariate high-dimensional nonlinear model of the underlying industrial plant can be approximated by the proposed deep MGRBF network. During real-time optimization/control, when the process operation varies dramatically, our proposed online adaptation rule enables effective tracking of the changing process characteristics, thereby reducing the plant-model mismatch.

5. Conclusions

This paper has proposed a novel adaptive deep MGRBF network for online modeling and identification of multivariate nonlinear and nonstationary industrial processes. Specifically, a MGRBF network is first utilized to provide a preliminary prediction of the target outputs, which is combined with the original input data to define the attribute input vector to a SAE. Deep quality-enhanced features are than extracted progressively by the SAE, to provide the inputs to a MGRBF tracker for online prediction and adaptive modeling of the process output. The proposed framework has integrated the feature learning capability of the SAE with the adaptive capability of the MGRBF network, and it can perform model adaptation very efficiently for real-time tracking of fast time-varying process dynamics. Applications to two industrial processes have demonstrated the superiority of our proposed method over the existing state-of-the-art online adaptive models and deep learning models, in terms of both multi-output prediction accuracy and online computational complexity.

Owing to the excellent adaptive modeling and dimensional reduction capacity of the proposed deep MGRBF network, it can serve as a powerful function approximator for deep reinforcement learning in industrial process control. Through real-time interaction with the industrial plant, online measurement data can be used to update the proposed network to learn the value function

and control policy. This is an important application of the proposed method in process control, where the plant-model mismatch due to time-varying process characteristics needs to be considered, and it will be our future study.

References

[1] X. Zhang and Z. Ge, "Automatic deep extraction of robust dynamic features for industrial big data modeling and soft sensor application," *IEEE Trans. Ind. Informat.*, vol. 16, no. 7, pp. 4456–4467, Jul. 2020.

[2] C. Shang and F. You, "Data analytics and machine learning for smart process manufacturing: Recent advances and perspectives in the big data era," *Engineering*, vol. 5, no. 6, pp. 1010–1016, Dec. 2019.

[3] S A. Billings and S. Chen, "The determination of multivariable nonlinear models for dynamic systems," in *Control and Dynamic Systems , Volume 7 of Neural Network Systems Techniques and Applications*, (ed. C.L. Leondes, San Diego: Academic Press), 1998, pp. 231–278.

[4] X. Zhang, C. Song, J. Zhao, and D. Xia, "Gaussian mixture continuously adaptive regression for multimode processes soft sensing under time-varying virtual drift," *J. Process Control*, vol. 124, pp. 1–13, Apr. 2023.

[5] M. Jia, *et al.*, "Graph convolutional network soft sensor for process quality prediction," *J. Process Control*, vol. 123, pp. 12–25, Mar. 2023.

[6] T. Zhang, *et al.*, "Dynamic transfer soft sensor for concept drift adaptation," *J. Process Control*, vol. 123, pp. 50–63, March 2023.

[7] S. Liu and S. Li, "A semi-supervised soft sensor method based on vine copula regression and tri-training algorithm for complex chemical processes," *J. Process Control*, vol. 120, pp. 115–128, Dec. 2022.

[8] 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.

[9] K. Wang, R. B. Gopaluni, J. Chen, and Z. Song, "Deep learning of complex batch process data and its application on quality prediction," *IEEE Trans. Ind. Informat.*, vol. 16, no. 12, pp. 7233–7242, Dec. 2020.

[10] Q. Sun and Z. Ge, "A survey on deep learning for data-driven soft sensors," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 5853–5866, Sep. 2021.

[11] W. Shao, *et al.*, "Online soft sensor design using local partial least squares models with adaptive process state partition," *Chemometrics Intell. Lab. Syst.*, vol. 144, pp. 108–121, May 2015.

[12] H. Jin, *et al.*, "Dual learning-based online ensemble regression approach for adaptive soft sensor modeling of nonlinear time-varying processes," *Chemometrics Intell. Lab. Syst.*, vol. 151, pp. 228–244, Feb. 2016.

[13] 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, Oct. 2018.

[14] 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.

[15] 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, no. 1 pp. 73278–73292, Apr. 2020.

[16] T. Liu, *et al.*, "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.

[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, *et al.*, "Model selection approaches for non-linear system identification: A review," *Int. J. Systems Science*, vol. 39, no. 10, pp. 925–946, Oct. 2008.

[20] S. Chen, X. X. Wang, and C. J. Harris, "NARX-based nonlinear system identification using orthogonal least squares basis hunting," *IEEE Trans. Control Systems Technology*, vol. 18, no. 1, pp. 78–84, Jan. 2008.

[21] S. Chen, P. M. Grant, and C. F. N. Cowan, "Orthogonal least squares algorithm for training multi-output radial basis function networks," *IEE Proc. Part F*, vol. 139, no. 6, pp. 378–384, Dec. 1992.

[22] S. Chen, "Nonlinear time series modelling and prediction using Gaussian RBF networks with enhanced clustering and RLS learning," *Electronics Letters*, vol. 31, no. 2, pp. 117–118, Jan. 1995.

[23] 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.

[24] 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.

[25] 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.

[26] T. Liu, *et al.*, "Fast tunable gradient RBF networks for online modeling of nonlinear and nonstationary dynamic processes," *J. Process Control*, vol. 93, pp. 53–65, Sep. 2020.

[27] T. Liu, *et al.*, "Adaptive multi-output gradient RBF tracker for nonlinear and nonstationary regression," *IEEE Trans. Cybern.* (Early Access), pp. 1–14, Feb. 2023. DOI:10.1109/TCYB.2023.3235155

[28] S. J Qin, *et al.*, "Bridging systems theory and data science: A unifying review of dynamic latent variable analytics and process monitoring," *Annual Reviews in Control*, vol. 50, pp. 29–48, Oct. 2020.

[29] Y. Dong, Y. Liu, and S. J. Qin, "Efficient dynamic latent variable analysis for high-dimensional time series data," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4068–4076, Jun. 2020.

[30] Y. Dong and S. J. Qin, "Regression on dynamic PLS structures for supervised learning of dynamic data," *J. Process Control*, vol. 68, pp. 64–72, Aug. 2018.

[31] G. Li and S. J. Qin, "Comparative study on monitoring schemes for non-Gaussian distributed processes," *J. Process Control*, vol. 67, pp. 69–82, Jul. 2018.

[32] R. Xie, *et al.*, "Supervised variational autoencoders for soft sensor modeling with missing data," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2820–2828, Apr. 2020.

[33] C. Liu, K. Wang, Y. Wang, and X. Yuan, 'Learning deep multi-manifold structure feature representation for quality prediction with an industrial application," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 5849–5858, Sep. 2022.

[34] B. Shen and Z. Ge, "Weighted nonlinear dynamic system for deep extraction of nonlinear dynamic latent variables and industrial application," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3090–3098, May 2021.

[35] X. Yuan, *et al.*, "Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3235–3243, Jul. 2018.

[36] C. Ou, *et al.*, "Quality-driven regularization for deep learning networks and its application to industrial soft sensors," *IEEE Trans. Neural Netw. Learn. Syst.* (Early Access), pp. 1–11, 2022, doi:10.1109/TNNLS.2022.3144162.

[37] X. Yuan, *et al.*, 'Hierarchical quality-relevant feature representation for soft sensor modeling: A novel deep learning strategy," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 3721–3730, Jun. 2020.

[38] X. Yuan, *et al.*, "A deep supervised learning framework for data-driven soft sensor modeling of industrial processes," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4737–4746, Nov. 2020.

[39] R. Xie, *et al.*, "Data-driven modeling based on two-stream λ gated recurrent unit network with soft sensor application," *IEEE Trans. Ind. Electron.*, vol. 67, no. 8, pp. 7034–7043, Aug. 2020.

[40] 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.

[41] *A Web Based Program for Dynamic Simulation of Fed-Batch Penicillin Production*. Accessed: Jun. 2017. [Online]. Available: http://simulator.iit.edu/web/pensim/simul.htm

[42] T. Liu, S. Liang, Q. Xiong, and K. Wang, "Data-based online optimal temperature tracking control in continuous microwave heating system by adaptive dynamic programming," *Neural Processing Letter*, vol. 51, no. 1, pp. 167–191, Feb. 2020.

[43] K. Wang, *et al.*, "Learning to detect local overheating of the high-power microwave heating process with deep learning," *IEEE Access*, vol. 6, pp. 10288–10296, Feb. 2018.

[44] T. Liu, S. Liang, Q. Xiong, and K. Wang, "Integrated CS optimization and OLS for recurrent neural network in modeling microwave thermal process,"

Neural Computing & Applications, vol. 32, no. 16, pp. 12267–12280, Aug. 2020.

[45] T. Liu, S. Liang, and J. L. Hu, "Expert control system based hierarchical control strategy for tunnel microwave rice drying," *Proc. ECC 2019* (Naples, Italy), Jun. 25-28, 2019, pp. 3619–3624.

[46] X. Hong and S. Chen, "A new RBF neural network with boundary value constraints," *IEEE Trans. Systems, Man, Cybern., Part B*, vol. 39, no. 1, pp. 298–303, Feb. 2009.

[47] S. Chen, X. Hong, and C. J. Harris, "Grey-box radial basis function modelling," *Neurocomputing*, vol. 4, no. 10, pp. 1564–1571, 2011.

[48] Y. Meng, *et al.*, "Physics-guided generative adversarial networks for sea subsurface temperature prediction," *IEEE Trans. Neural Netw. Learn. Syst.* (Early Access), pp. 1–14, Nov. 2021, DOI:10.1109/TNNLS.2021.3123968.

[49] M. Zagorowska, *et al.*, "Online feedback optimization of compressor stations with model adaptation using Gaussian process regression," *J. Process Control*, vol. 121, pp. 119–133, Jan. 2023.

[50] L. MacKinnon, *et al.*, "Dynamic real-time optimization for nonlinear systems with Lyapunov stabilizing MPC," *J. Process Control*, vol. 114, pp. 1–15, Jun. 2022.

[51] V. Ramasamy, *et al.*, "A comprehensive review on advanced process control of cement kiln process with the focus on MPC tuning strategies," *J. Process Control*, vol. 121, pp. 85–102, Jan. 2023.

[52] J. Qiu, M Ma, T. Wang, and H. Gao, "Gradient descent based adaptive learning control for autonomous underwater vehicles with unknown uncertainties," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5266–5273, Dec. 2021.