Nonlinear Identification Using Orthogonal Forward Regression With Nested Optimal Regularization

Xia Hong, *Senior Member, IEEE*, Sheng Chen, *Fellow, IEEE*, Junbin Gao, and Chris J. Harris

***Abstract*—An efficient data based-modeling algorithm for nonlinear system identification is introduced for radial basis function (RBF) neural networks with the aim of maximizing generalization capability based on the concept of leave-one-out (LOO) cross validation. Each of the RBF kernels has its own kernel width parameter and the basic idea is to optimize the multiple pairs of regularization parameters and kernel widths, each of which is associated with a kernel, one at a time within the orthogonal forward regression (OFR) procedure. Thus, each OFR step consists of one model term selection based on the LOO mean square error (LOOMSE), followed by the optimization of the associated kernel width and regularization parameter, also based on the LOOMSE. Since like our previous state-of-the-art local regularization assisted orthogonal least squares (LROLS) algorithm, the same LOOMSE is adopted for model selection, our proposed new OFR algorithm is also capable of producing a very sparse RBF model with excellent generalization performance. Unlike our previous LROLS algorithm which requires an additional iterative loop to optimize the regularization parameters as well as an additional procedure to optimize the kernel width, the proposed new OFR algorithm optimizes both the kernel widths and regularization parameters within the single OFR procedure, and consequently the required computational complexity is dramatically reduced. Nonlinear system identification examples are included to demonstrate the effectiveness of this new approach in comparison to the well-known approaches of support vector machine and least absolute shrinkage and selection operator as well as the LROLS algorithm.**

***Index Terms*—Cross validation (CV), forward regression, identification, leave-one-out (LOO) errors, nonlinear system, regularization.**

I. Introduction

A LARGE class of nonlinear models including some types of neural networks can be classified as linear-in-the-parameters models. These models have provable learning and convergence conditions, are well suited for adaptive learning, and have clear engineering applications [1]–[3]. Note that in order to obtain a desired linear-in-the-parameters structure [4]–[7], however, nonlinear parameters in these models must be determined by other means as they cannot be estimated by a linear learning process. For example, the radial basis function (RBF) centers and kernel centers in the RBF neural network and the kernel model must be selected or be restricted to the training input data in order to apply the linear learning approaches of [4]–[7]. Furthermore, the RBF or kernel width parameter and other so-called hyperparameters, such as regularization parameters, also have to be learnt typically via cross validation (CV). One of the main aims of data-based modeling is good generalization, i.e., the model's capability to approximate accurately the system output for unseen data. The concept of CV is fundamental to the evaluation of model generalization capability [8], and it is widely embedded either in parameter estimation, e.g., tuning regularization parameter [9]–[11] and forming new parameter estimates [12], or in deriving model selection criteria based on information theoretic principles [13], which regularizes model structure in order to produce sparse models.

The orthogonal least squares (OLS) algorithm, which efficiently constructs sparse models in an orthogonal forward regression (OFR) procedure [14], [15], is a popular modeling tool in neural networks, such as RBF networks [4], [16], neurofuzzy systems [17], [18], and wavelet neural networks [19], [20]. The original OLS algorithm [14] selects regressors by virtue of their contribution to the maximization of the model error reduction ratio. One commonly used version of CV is the leave-one-out (LOO) CV. For linear-in-the-parameters models, the LOO errors can be calculated without actually splitting the training data set and estimating the associated models, by making use of the Sherman–Morrison–Woodbury theorem [21]. By incorporating the OFR framework with analytical expression of LOO errors, the LOO mean square error (LOOMSE) was proposed as a model term selection criterion, which can be sequentially optimized within the model construction process [22], [23], enabling the OFR model construction procedure to automatically terminate with a sparse model that generalizes well, without resorting to other stopping criterion. It is worth emphasizing that for nonlinear system identification at least, the objective is to obtain sparse models that generalize well. A system engineer is unlikely to accept a huge-size model with many model terms for controller design purpose, even the model is a faithful representation of the underlying nonlinear system that generates the data. Take the engine data identification of [24], which is also considered in this paper,

Manuscript received August 11, 2014; revised November 22, 2014; accepted January 6, 2015. Date of publication January 27, 2015; date of current version November 13, 2015. This paper was recommended by Associate Editor L. Rutkowski.

X. Hong is with the School of Systems Engineering, University of Reading, Reading RG6 6AY, U.K.

S. Chen is with Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K., and also with the Faculty of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia (e-mail: sqc@ecs.soton.ac.uk).

J. Gao is with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia.

C. J. Harris is with Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2015.2389524

no control engineer will use a model with two hundreds of model terms to design controller for the lorry engine, but a small model with 20 model terms is acceptable to a control engineer, provided that the model is accurate.

In [6], empirical evidence is provided to suggest that using the LOOMSE to optimize the kernel width and regularization parameter leads to overfitting for the least squares support vector machine (LSSVR), which uses all the training data as model base. From a model selection viewpoint based on bias and variance analysis, [6] shows that this overfitting phenomenon is due to the resultant LSSVR models having a large variance. We argue that it is very unlikely for the sparse models identified by our local regularization assisted OLS (LROLS) algorithm based on the LOOMSE [23] to overfit, because our model selection framework is very different from the one used in [6]. In fact, the nonsparse LSSVR model with all the training data as its kernels is inherently prone to overfitting. By contrast, our subset model selection based on the LOOMSE criterion constructs a sparse model, exactly aiming to avoid overfitting. While the results of [6] show that the LOOMSE continuously decreases as the iterations of the hyperparameter optimization increases, a classical sign of overfitting, it has been shown in [22] and [23] that in our subset model selection procedure, the LOOMSE reaches an optimal value at certain model size.

Generally, better model generalization can be achieved using parameter regularization, which penalizes the norm of parameters. The effect of parameter regularization is to add a small bias in order to gain advantage in combating the problem of large variance which is often associated with an oversized model structure. Although a very small positive regularization parameter based on the l^2-norm is often used to improve numerical condition, more advanced techniques aim at the regularization design with respect to given data, leading to sparser model structure. For example, sparse models can be constructed using the l^1-norm penalized cost function, e.g., the basis pursuit or least absolute shrinkage and selection operator (LASSO) [25]–[27]. However, the optimization of the l^1-norm regularizer with respect to model generalization analytically is less studied. Alternatively, the l^2-norm regularization technique has been incorporated into the OLS algorithm to produce a regularized OLS (ROLS) algorithm that carries out model term selection while reducing the variance of parameter estimates simultaneously [10], [11]. It has been shown [28], [29] that the l^2-norm parameter regularization is equivalent to a maximized *a posterior* probability estimate of parameters from Bayesian viewpoint by adopting a Gaussian prior for the parameters, leading to an iterative evidence procedure for solving the optimal regularization parameters [29], [30]. Further adopting the LOOMSE as the model selection criterion leads to our state-of-the-art LROLS algorithm [23], which is capable of producing a very sparse nonlinear model with excellent generalization performance.

Similar to all the other existing regularization assisted sparse model construction algorithms, the optimization of the regularization parameters in the LROLS algorithm [23] includes the OFR procedure as an inner loop, and therefore several iterations of the OFR procedure are needed. More specifically, with all the regularization parameters initially set to a very small positive value, the OFR procedure in the inner loop constructs a sparse model. The regularization parameters are then updated at the outer loop using for example the efficient evidence procedure of [30], and the inner-loop OFR algorithm is started again with the new regularization parameters. A few outer-loop iterations, typically no more than 10, will ensure that a set of near optimal regularization parameters are found. Therefore, the require computational complexity approximately equals to the complexity of single OFR procedure scaled up by the number of outer-loop iterations for computing near optimal regularization parameters. Moreover, other hyperparameters, such as the RBF widths, must also be optimized in order to ensure an excellent generalization capability. For the RBF or kernel model which employs a single common RBF width for all model regressors, a grid search procedure based on CV is typically used to find a near-optimal RBF width and this adds an extra third iterative loop to the model construction. Thus, the true total complexity is scaled up by the number of grid searches for the RBF width. For the RBF model where each model regressor has an individual RBF width, the nonlinear search space for all the RBF width parameters becomes too large for a grid-based procedure, and other nonlinear optimization means must be adopted [31]–[33]. It is, therefore, highly desirable that the hyperparameters, including nonlinear regularization parameters and RBF widths, can be analytically optimized within the single OFR model construction process. This motivates our current study.

In this paper, we propose a new OFR algorithm in which the optimization of kernel widths and regularization parameters is nested within the OFR procedure, so that only a single OFR iteration is needed for the estimation of these hyperparameters. Specifically, we use the Gaussian RBF model that has individually tunable kernel width for each kernel, and both the kernel width and associated regularization parameter are optimized one kernel at a time within the OFR procedure. An l^2-norm locally ROLS cost function [23], [28] is used for model parameter estimation, and the resultant LOOMSE measures the model generalization performance. By exploiting the analytical expression of LOO errors, we derive a new approach for successively estimating the two hyperparameters of RBF width and regularization parameter associated with each selected regressor using the LOOMSE criterion based on a simple line search and gradient descent algorithm, respectively, which is embedded naturally in each regressor selection step of the OFR procedure. Consequently, a computationally efficient and fully automated procedure can be achieved without resorting to any other validation data set for iterative model evaluations. Since the same LOOMSE criterion of the LROLS algorithm [23] is adopted, our proposed new OFR algorithm is also capable of producing a very sparse similar RBF model with excellent generalization performance. Unlike our previous LROLS algorithm [23], the new proposed algorithm constructs a sparse model within a single OFR procedure and consequently the required computational complexity is dramatically reduced.

This paper is organized as follows. Section II outlines the concept of the OFR using a regularized cost function.

In Section III, we present the nth stage of the OFR model representation and the concept of LOO CV for model term selection using the LOOMSE, followed by a line search for tuning the kernel width and a gradient descent algorithm to estimate the regularization parameter for the selected regressor. Both the line search and gradient descent algorithm are also based on minimizing the LOOMSE. Section IV presents the entire proposed OFR algorithm with the nested LOOMSE-based optimal regularization, referred to as the OFR + LOOMSE, for constructing sparse models. In Section V, the empirical results demonstrate the effectiveness of our proposed OFR + LOOMSE algorithm. The conclusion is given in Section VI.

II. Preliminaries

Consider the general nonlinear system represented by the nonlinear model [34]

$$y(k) = f(y(k-1), \ldots, y(k-n_y), u(k-1), \ldots, u(k-n_u)) + v(k) = f(\boldsymbol{x}(k)) + v(k) \quad (1)$$

where $y(k)$ and $u(k)$ are the system output and control input with the lags n_y and n_u, respectively, at sample time index k, and $v(k)$ denotes the system white noise, while $m = n_y + n_u$, $\boldsymbol{x}(k) = [y(k-1) \cdots y(k-n_y)\ u(k-1) \cdots u(k-n_u)]^{\mathrm{T}} = [x_1(k)\ x_2(k) \cdots x_m(k)]^{\mathrm{T}} \in \mathbb{R}^m$ denotes the m-dimensional system input vector, and $f(\bullet)$ is the unknown system mapping. The unknown nonlinear system (1) is to be identified based on an observation data set $D_N = \{\boldsymbol{x}(k), y(k)\}_{k=1}^N$ using some suitable functional which can approximate $f(\bullet)$ with arbitrary accuracy. Without loss of generality, in this paper, we use the data set D_N to construct a RBF network model of the form

$$\widehat{y}^{(M)}(k) = f^{(M)}(\boldsymbol{x}(k)) = \sum_{i=1}^{M} \theta_i \phi_i(\boldsymbol{x}(k)) \quad (2)$$

where $\widehat{y}^{(M)}(k)$ is the model prediction output for the input vector $\boldsymbol{x}(k)$ based on the M-term RBF model, M is the total number of regressors or model terms, and θ_i are the model weights, while the regressor $\phi_i(\boldsymbol{x})$ takes the form of Gaussian basis function given by

$$\phi_i(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_i\|^2}{2\tau_i^2}\right) \quad (3)$$

in which $\boldsymbol{c}_i = [c_{1,i}\ c_{2,i} \cdots c_{m,i}]^{\mathrm{T}}$ is the center vector of the ith RBF unit and $\tau_i > 0$ is the ith RBF unit's width parameter. We assume that each RBF kernel is placed on a training data, namely, all the RBF center vectors $\{\boldsymbol{c}_i\}_{i=1}^M$ are selected from the training data $\{\boldsymbol{x}(k)\}_{k=1}^N$.

Denote $e^{(M)}(k) = y(k) - \widehat{y}^{(M)}(k)$ as the M-term modeling error for the input data point $\boldsymbol{x}(k)$. Over the training data set D_N, further denote $\boldsymbol{y} = [y(1)\ y(2) \cdots y(N)]^{\mathrm{T}}$, $\boldsymbol{e}^{(M)} = [e^{(M)}(1)\ e^{(M)}(2) \cdots e^{(M)}(N)]^{\mathrm{T}}$, and $\boldsymbol{\Phi}_M = [\boldsymbol{\phi}_1\ \boldsymbol{\phi}_2 \cdots \boldsymbol{\phi}_M]$ with $\boldsymbol{\phi}_n = [\phi_n(\boldsymbol{x}(1))\ \phi_n(\boldsymbol{x}(2)) \cdots \phi_n(\boldsymbol{x}(N))]^{\mathrm{T}}$, $1 \le n \le M$. We have the M-term model in the matrix form of

$$\boldsymbol{y} = \boldsymbol{\Phi}_M \boldsymbol{\theta}_M + \boldsymbol{e}^{(M)} \quad (4)$$

where $\boldsymbol{\theta}_M = [\theta_1\ \theta_2 \cdots \theta_M]^{\mathrm{T}}$. Let an orthogonal decomposition of the regression matrix $\boldsymbol{\Phi}_M$ be $\boldsymbol{\Phi}_M = \boldsymbol{W}_M \boldsymbol{A}_M$, where

$$\boldsymbol{A}_M = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,M} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{M-1,M} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \quad (5)$$

and

$$\boldsymbol{W}_M = [\boldsymbol{w}_1\ \boldsymbol{w}_2 \cdots \boldsymbol{w}_M] \quad (6)$$

with $\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_j = 0$, if $i \neq j$. The regression model (4) can alternatively be expressed as

$$\boldsymbol{y} = \boldsymbol{W}_M \boldsymbol{g}_M + \boldsymbol{e}^{(M)} \quad (7)$$

where $\boldsymbol{g}_M = [g_1\ g_2 \cdots g_M]^{\mathrm{T}}$ satisfies the triangular system $\boldsymbol{A}_M \boldsymbol{\theta}_M = \boldsymbol{g}_M$, which can be used to determine $\boldsymbol{\theta}_M$, given $\boldsymbol{A}_M$ and $\boldsymbol{g}_M$.

Further consider the following regularized cost function:

$$L_e\left(\boldsymbol{\Lambda}_M, \boldsymbol{g}_M\right) = \left\|\boldsymbol{y} - \boldsymbol{W}_M \boldsymbol{g}_M\right\|^2 + \boldsymbol{g}_M^{\mathrm{T}} \boldsymbol{\Lambda}_M \boldsymbol{g}_M \quad (8)$$

where $\boldsymbol{\Lambda}_M = \mathrm{diag}\{\lambda_1, \lambda_2, \ldots, \lambda_M\}$, which contains the local regularization parameters $\lambda_i > 0$, for $1 \le i \le M$. For a given $\boldsymbol{\Lambda}_M$, the solution for $\boldsymbol{g}_M$ can be obtained by setting the derivative vector of L_e to zero, i.e., $(\partial L_e / \partial \boldsymbol{g}_M) = \boldsymbol{0}$, yielding

$$g_i^{(\mathrm{R})} = \frac{\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{y}}{\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_i + \lambda_i} \quad (9)$$

for $1 \le i \le M$. Our objective $L_e(\boldsymbol{\Lambda}_M, \boldsymbol{g}_M)$ is constructed on the orthogonal space and the l^2-norm parameter constraints are associated with the orthogonal bases $\boldsymbol{w}_i$, $1 \le i \le M$.

III. Simultaneous Regularization Parameter Optimization and Model Construction Using LOOMSE

For each stage of the OFR, we select the regressor with the smallest LOOMSE amongst all the candidate regressors using a common preset kernel width and a very small preset regularization parameter, followed by determining the optimal kernel width and then the optimal regularization parameter associated with the selected regressor.

A. nth Stage OFR Model Representation and LOOMSE

At the nth OFR stage, the nth model term is selected from a candidate pool that is formed according to (3) using the whole set or a subset of training data as candidate centers and a common preset kernel width $\tau_n = \tau_0$ and a preset regularization parameter $\lambda_n = \lambda_0$ initially. Consider the OFR modeling process that has produced the n-term model. Let us denote the constructed n columns of regressors as $\boldsymbol{W}_n = [\boldsymbol{w}_1\ \boldsymbol{w}_2 \cdots \boldsymbol{w}_n]$, with $\boldsymbol{w}_n = [w_n(1)\ w_n(2) \cdots w_n(N)]^{\mathrm{T}}$. The model output vector of this n-term model is denoted as

$$\widehat{\boldsymbol{y}}^{(n)} = \sum_{i=1}^{n} g_i^{(\mathrm{R})} \boldsymbol{w}_i = \boldsymbol{W}_n \boldsymbol{g}_n^{(\mathrm{R})} \quad (10)$$

where $\boldsymbol{g}_n^{(\mathrm{R})} = [g_1^{(\mathrm{R})}\ g_2^{(\mathrm{R})} \cdots g_n^{(\mathrm{R})}]^{\mathrm{T}}$, and the corresponding modeling error vector is given by $\boldsymbol{e}^{(n)} = \boldsymbol{y} - \widehat{\boldsymbol{y}}^{(n)}$. Clearly, the nth OFR stage can be represented by

$$\boldsymbol{e}^{(n-1)} = g_n \boldsymbol{w}_n + \boldsymbol{e}^{(n)}. \tag{11}$$

Equation (11) illustrates the fact that the nth OFR stage is simply to fit a one-variable model using the current model residual produced after the $(n-1)$th stage as the desired system output.

The selection of one regressor from the $(M-n+1)$ candidate regressors involves generating the $(M-n+1)$ candidates for $\boldsymbol{w}_n$, i.e., by making each of the $(M-n+1)$ candidate regressors orthogonal to the $(n-1)$ orthogonal basis vectors $\boldsymbol{w}_i$ for $1 \le i \le n-1$, already selected in the previous $(n-1)$ OFR stages. Then the contributions of the candidate regressors are evaluated based on the idea of the LOO CV outlined below.

Consider the model fitting (11) on D_N, where the desired output vector is $\boldsymbol{e}^{(n-1)}$ with its elements given by $e^{(n-1)}(k)$, $1 \le k \le N$. For notational convenience, we also denote a candidate model identified using all the N data points as $e^{(n-1)}(k, \lambda_n, \tau_n) = g_n^{(\mathrm{R})} w_n(k)$ at the nth model fitting stage. Suppose that we now sequentially set aside each data point in the estimation set D_N in turn and estimate a model using the remaining $(N-1)$ data points. The prediction error is calculated on the data point that was removed from the identification. That is, for $k = 1, 2, \ldots, N$, the model is estimated by removing the kth data point from D_N, and the output of the model, identified based on $D_N \setminus (\boldsymbol{x}(k), y(k))$, is computed for the kth data unused in the identification and is denoted by $e^{(n-1,-k)}(k, \lambda_n, \tau_n)$. Then, the LOO prediction error is calculated as

$$e^{(n,-k)}(k, \lambda_n, \tau_n) = e^{(n-1)}(k) - e^{(n-1,-k)}(k, \lambda_n, \tau_n). \tag{12}$$

Direct evaluation of $e^{(n,-k)}(k, \lambda_n, \tau_n)$ by splitting the data set requires extensive computational efforts. However, as we have shown in [23], they can be exactly calculated without actually sequentially splitting the estimation data set. The LOOMSE is then defined as the average of the squared LOO errors, given by $J(\lambda_n, \tau_n) = 1/N \sum_{k=1}^{N} (e^{(n,-k)}(k, \lambda_n, \tau_n))^2$.

The model generalization contribution from the nth stage of OFR depends on the selection of a regressor from the $(M-n+1)$ candidate regressors and further tuning of kernel width as well as regularization parameter optimization based on the updated regressor. We propose that the LOOMSE is used initially for regressor selection and then used for kernel width as well as regularization parameter estimation. To be more specific, firstly we use a very small regularization parameter λ_0, e.g., $\lambda_0 = 10^{-8}$, for all the $(M-n+1)$ candidate regressors, and the associated LOOMSE values are calculated and ranked. Then the regressor with the smallest LOOMSE is selected as the nth regressor, denoted as $\boldsymbol{\phi}_n(\tau_0)$. Secondly, for this selected regressor, we first maximize its potential model generalization performance by successively updating $\boldsymbol{w}_n$ via τ_n when λ_0 is fixed, then followed by the associated regularization parameter optimization by minimizing the LOOMSE with respect to λ_n when τ_n is fixed, so that:

$$\lambda_n^{\mathrm{opt}} = \arg\min_{\lambda_n} \left\{ \min_{\tau_n} \left\{ \frac{1}{N} \sum_{k=1}^{N} \left(e^{(n,-k)}(k, \lambda_n, \tau_n) \right)^2 \right\} \right\}. \tag{13}$$

An optimization procedure is introduced in Section III-B for this two-variables optimization problem.

B. Successive Kernel Width and Regularization Parameter Estimation via Minimizing LOOMSE

Because the joint optimization of the LOOMSE with respect to both the kernel width and regularization parameter is very complex, we opt for the following successive optimization procedure: 1) we first hold the regularization parameter as λ_0 and tune the kernel width τ_n by directly evaluating the LOOMSE using a line search algorithm and 2) with the obtained optimal τ_n, we then optimize λ_n based on the LOOMSE using a gradient descent algorithm.

The model parameter estimator $g_n^{(\mathrm{R})}(\lambda_n, \tau_n)$ can be written as

$$g_n^{(\mathrm{R})}(\lambda_n, \tau_n) = \boldsymbol{w}_n^{\mathrm{T}} \boldsymbol{e}^{(n-1)} \big/ \alpha_n \tag{14}$$

where

$$\alpha_n = \boldsymbol{w}_n^{\mathrm{T}} \boldsymbol{w}_n + \lambda_n. \tag{15}$$

The model residual is then given by

$$e^{(n)}(k, \lambda_n, \tau_n) = e^{(n-1)}(k) - w_n(k) \boldsymbol{w}_n^{\mathrm{T}} \boldsymbol{e}^{(n-1)} \big/ \alpha_n \tag{16}$$

and the LOOMSE can be calculated as [23]

$$J(\lambda_n, \tau_n) = \tfrac{1}{N} \textstyle\sum_{k=1}^{N} \left(\frac{e^{(n)}(k, \lambda_n, \tau_n)}{\beta_n(k)} \right)^2 \tag{17}$$

where

$$\beta_n(k) = 1 - \sum_{i=1}^{n} \frac{(w_i(k))^2}{\alpha_i} = \beta_{n-1}(k) - \frac{(w_n(k))^2}{\alpha_n}. \tag{18}$$

Note that $\beta_{n-1}(k)$ has already been determined in the previous regression step, and $\beta_0(k) = 1$.

Further, note that, $\boldsymbol{\phi}_n = \boldsymbol{\phi}_n(\tau_n)$ is a functional of τ_n and

$$\boldsymbol{w}_n = \boldsymbol{\phi}_n(\tau_n) - \sum_{i=1}^{n-1} a_{i,n} \boldsymbol{w}_i \tag{19}$$

with $a_{i,n} = \boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{\phi}_n(\tau_n) / \boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_i$, $1 \le i \le n-1$. Therefore, we have

$$\boldsymbol{w}_n = \boldsymbol{B}_n \boldsymbol{\phi}_n(\tau_n) \tag{20}$$

where $\boldsymbol{B}_n = \boldsymbol{B}_{n-1} - \boldsymbol{w}_{n-1} \boldsymbol{w}_{n-1}^{\mathrm{T}} / \boldsymbol{w}_{n-1}^{\mathrm{T}} \boldsymbol{w}_{n-1}$, and $\boldsymbol{B}_1 = \boldsymbol{I}$ is the $N \times N$ identity matrix. Clearly the gradient $\partial J(\lambda_n, \tau_n)/\partial \tau_n$ is very complex. However, if we fix $\lambda_n = \lambda_0$, then LOOMSE becomes a one-variable function of τ_n which either increases or decreases as τ_n increases until it reaches a local minimum. Thus, we can simply use a 1-D line search to solve for τ_n, as shown in Table I.

Remark 1: This 1-D line search algorithm is guaranteed to find a near optimal kernel width $\tau_n \in [\varsigma^{\mathrm{It}_1} \tau_0,\ \varsigma^{-\mathrm{It}_1} \tau_0]$ with no more than It_1 iterations given the learning rate $0 < \varsigma < 1$. Provided that the initial kernel width τ_0 is not too far away

TABLE I
Kernel Width Parameter Update For Fixed λ_0

```
Set number of iterations $It_1$ to a small value, e.g. 10.
Preset learning rate $\varsigma$ as $0.95 < \varsigma < 1$, e.g. $0.98$.
Set $\tau_n^{\rm old} = \tau_0$, $\tau_n^{\rm old,1} = \tau_0/\varsigma$ and $\tau_n^{\rm old,2} = \tau_0 * \varsigma$.
Calculate $J\big(\lambda_0, \tau_n^{\rm old}\big)$, $J\big(\lambda_0, \tau_n^{\rm old,1}\big)$ and $J\big(\lambda_0, \tau_n^{\rm old,2}\big)$ according to (20) and (15)-(18).
Set $\tau_n^{\rm new} = \arg \min_{\tau_n \in \{\tau_n^{\rm old}, \tau_n^{\rm old,1}, \tau_n^{\rm old,2}\}} J\big(\lambda_0, \tau_n\big)$.
if $\tau_n^{\rm new} = \tau_n^{\rm old,1}$ then
    $\tau_n^{\rm old} \leftarrow \tau_n^{\rm old,1}$, $\tau_n^{\rm old,1} \leftarrow \tau_n^{\rm old,1}/\varsigma$.
    Calculate $J\big(\lambda_0, \tau_n^{\rm old,1}\big)$ according to (20) and (15)-(18).
    $\iota = 1$.
    while $\iota < It_1$ and $J\big(\lambda_0, \tau_n^{\rm old,1}\big) < J\big(\lambda_0, \tau_n^{\rm old}\big)$ do
        $\tau_n^{\rm old} \leftarrow \tau_n^{\rm old,1}$, $\tau_n^{\rm old,1} \leftarrow \tau_n^{\rm old,1}/\varsigma$.
        Calculate $J\big(\lambda_0, \tau_n^{\rm old,1}\big)$ according to (20) and (15)-(18).
        $\iota = \iota + 1$.
    end while
    $\tau_n = \tau_n^{\rm old}$.
else
    if $\tau_n^{\rm new} = \tau_n^{\rm old,2}$ then
        $\tau_n^{\rm old} \leftarrow \tau_n^{\rm old,2}$, $\tau_n^{\rm old,2} \leftarrow \tau_n^{\rm old,2} * \varsigma$.
        Calculate $J\big(\lambda_0, \tau_n^{\rm old,2}\big)$ according to (20) and (15)-(18).
        $\iota = 1$.
        while $\iota < It_1$ and $J\big(\lambda_0, \tau_n^{\rm old,2}\big) < J\big(\lambda_0, \tau_n^{\rm old}\big)$ do
            $\tau_n^{\rm old} \leftarrow \tau_n^{\rm old,2}$, $\tau_n^{\rm old,2} \leftarrow \tau_n^{\rm old,2} * \varsigma$.
            Calculate $J\big(\lambda_0, \tau_n^{\rm old,2}\big)$ according to (20) and (15)-(18).
            $\iota = \iota + 1$.
        end while
        $\tau_n = \tau_n^{\rm old}$.
    else
        if $\tau_n^{\rm new} = \tau_n^{\rm old}$ then
            $\tau_n = \tau_n^{\rm new}$.
        end if
    end if
end if
Return $\tau_n$.
```

from a local minimum, the preset number of iterations It_1 can be set to a small value, e.g., 10. In this way, the computational cost of this 1-D line search in each OFR step can be controlled to be negligibly small, compared to the complexity required by selecting a model term in each OFR step. More specifically, it is well known that the complexity of selecting a model term in each OFR step is approximately on the order of N^2, denoted as $\mathcal{O}(N^2)$ [23], [33]. It is straightforward to verify that complexity of computing the LOOMSE according to (20) and (15)–(18) is on the order of $\mathcal{O}(N)$, and the total complexity of this line search is no more than $\text{It}_1 \cdot \mathcal{O}(N)$. Since $\text{It}_1 \ll N$, this is still on the order of $\mathcal{O}(N)$, which is much smaller than $\mathcal{O}(N^2)$.

Once τ_n is found we seek λ_n using a gradient descent algorithm. From (16), we have

$$\frac{\partial e^{(n)}(k, \lambda_n, \tau_n)}{\partial \lambda_n} = w_n(k)\boldsymbol{w}_n^{\rm T}\boldsymbol{e}^{(n-1)}\big/\alpha_n^2 = \frac{w_n(k) g_n^{(\rm R)}(\lambda_n, \tau_n)}{\alpha_n} \tag{21}$$

and the gradient of the LOOMSE with respect to λ_n is given by

$$\frac{\partial J(\lambda_n, \tau_n)}{\partial \lambda_n} = \frac{2}{N}\sum_{k=1}^{N} \frac{e^{(n)}(k, \lambda_n)}{\beta_n(k)} \frac{\partial e^{(n,-k)}(k, \lambda_n)}{\partial \lambda_n} \tag{22}$$

where

$$\begin{aligned} \frac{\partial e^{(n,-k)}(k, \lambda_n)}{\partial \lambda_n} &= \frac{\partial e^{(n)}(k, \lambda_n)\big/\partial \lambda_n}{\beta_n(k)} - \frac{e^{(n)}(k, \lambda_n)}{(\beta_n(k))^2} \frac{(w_n(k))^2}{\alpha_n^2} \\ &= g_n^{(\rm R)}(\lambda_n)\gamma_n(k) - e^{(n)}(k, \lambda_n)(\gamma_n(k))^2 \end{aligned} \tag{23}$$

and

$$\gamma_n(k) = \frac{w_n(k)}{\alpha_n \beta_n(k)}. \tag{24}$$

With $\lambda_n^{\rm old} = \lambda_0$ and λ_0 a preset very small positive value, the gradient descent algorithm for minimizing the LOOMSE of (17) is applied as follows:

$$\begin{cases} \lambda_n^{\rm new} = \max\left\{\lambda_0, \lambda_n^{\rm old} - \eta \cdot \text{sign}\left(\frac{\partial J}{\partial \lambda_n}\Big|_{\lambda_n = \lambda_n^{\rm old}}\right)\right\} \\ \lambda_n^{\rm old} = \lambda_n^{\rm new} \end{cases} \tag{25}$$

for a predetermined number of iterations It_2, e.g., $\text{It}_2 = 20$, where $\eta > 0$ is a very small positive learning rate. Note that $\text{sign}(\partial J(\lambda_n, \tau_n)/\partial \lambda_n)$ is used in (25), indicating that this is a normalized version of gradient descent algorithm and a small learning rate η will scale well with the search space of λ_n, irrespective of the actual size of $(\partial J(\lambda_n, \tau_n)/\partial \lambda_n)$.

Remark 2: This 1-D gradient descent algorithm is guaranteed to converge to a near optimal solution λ_n, provided that the initial regularization parameter λ_0 is not too far away from a local minimum. The computational cost of this gradient descent algorithm is low due to the neat expression of (25). Furthermore, the recursion formula for $\beta_n(k)$ given in (18) significantly reduces the cost of the derivative evaluation. Thus, the computational complexity of the above optimal regularization parameter estimator is on the order of $\mathcal{O}(N)$, scaled by the number of iterations It_2 which is usually much smaller than N. Therefore, the computational complexity of this gradient descent algorithm is on the order of $\mathcal{O}(N)$, which is negligibly small compared to the complexity $\mathcal{O}(N^2)$ required by selecting a model term in each OFR step.

This near optimal one regularization parameter estimator together with kernel parameter tuning for the selected regressor offers the benefit of further reduction in the LOOMSE for the selected regressor and hence improves the model generalization for the same model terms selected sequentially by the OFR procedure.

IV. Proposed OFR With Nested Optimal Regularization Using LOOMSE

The proposed OFR + LOOMSE algorithm is presented below integrating: 1) the model regressor selection based on minimizing the LOOMSE using a preset regularization parameter from the candidate set; 2) successively updating the kernel width and optimizing regularization parameter also based on minimizing the LOOMSE for the selected model regressor; and 3) the modified Gram-Schmidt orthogonalization procedure [14]. For notational convenience, define

$$\boldsymbol{\Phi}^{(n-1)} = \left[\boldsymbol{w}_1 \cdots \boldsymbol{w}_{n-1}\ \boldsymbol{\phi}_n^{(n-1)} \cdots \boldsymbol{\phi}_M^{(n-1)}\right] \in \mathbb{R}^{N\times M} \tag{26}$$

with $\boldsymbol{\Phi}^{(0)} = \boldsymbol{\Phi}_M$. If some of the columns in $\boldsymbol{\Phi}^{(n-1)}$ are interchanged, it is still referred to as $\boldsymbol{\Phi}^{(n-1)}$ for notational simplicity. The initial conditions are set as follows: $\boldsymbol{e}^{(0)} = \boldsymbol{y}$, $\beta_0(k) = 1$ for $1 \le k \le N$, and the learning rate η is a given small positive number, e.g., $\eta = 0.01$. Further denote the kth element of $\boldsymbol{\phi}_j^{(n-1)}$ as $\phi_j^{(n-1)}(k)$.

With the initialization of $\lambda_n^{\text{old}} = \lambda_0$ and $\tau_n = \tau_0$, the nth stage of the OFR procedure is given as follows.

Step 1: Model Term Selection

1) For $n \le j \le M$, calculate

$$\alpha_n^{(j)} = \left(\boldsymbol{\phi}_j^{(n-1)}\right)^{\mathrm{T}} \boldsymbol{\phi}_j^{(n-1)} + \lambda_n^{\text{old}} \tag{27}$$

$$\beta_n^{(j)}(k) = \beta_{n-1}(k) - \left(\phi_j^{(n-1)}(k)\right)^2 \Big/ \alpha_n^{(j)},\ \ 1 \le k \le N \tag{28}$$

$$g_n^{(\mathrm{R},j)} = \frac{\left(\boldsymbol{\phi}_j^{(n-1)}\right)^{\mathrm{T}} \boldsymbol{e}^{(n-1)}}{\alpha_n^{(j)}} \tag{29}$$

$$\boldsymbol{e}^{(n,j)} = \boldsymbol{e}^{(n-1)} - g_n^{(\mathrm{R},j)} \boldsymbol{\phi}_j^{(n-1)} \tag{30}$$

$$J_n^{(j)} = \frac{1}{N}\sum_{k=1}^{N}\left(e^{(n,j)}(k)/\beta_n^{(j)}(k)\right)^2. \tag{31}$$

2) Find

$$J_n = J_n^{(j_n)} = \min\left\{J_n^{(j)},\ n \le j \le M\right\}. \tag{32}$$

Then the j_nth and the nth columns of $\boldsymbol{\Phi}^{(n-1)}$ are interchanged. The j_nth column and the nth column of $\boldsymbol{A}_M$ are interchanged up to the $(n-1)$th row. This effectively selects the nth regressor in the subset model.

Step 2: Kernel Width Optimization

Apply the algorithm in Table I to find the optimal kernel width parameter τ_n. Form $\boldsymbol{\phi}_n(\tau_n)$ using (3) and update the nth column of $\boldsymbol{A}_M$ up to the $(n-1)$th row as

$$a_{i,n} = \frac{\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{\phi}_n(\tau_n)}{\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i},\ \ 1 \le i \le n-1. \tag{33}$$

The modified Gram-Schmidt orthogonalization procedure [14] then calculates the nth row of $\boldsymbol{A}_M$ and transfers $\boldsymbol{\Phi}^{(n-1)}$ into $\boldsymbol{\Phi}^{(n)}$ as follows:

$$\left.\begin{aligned} \boldsymbol{w}_n &= \boldsymbol{\phi}_n(\tau_n) - \sum_{i=1}^{n-1} a_{i,n}\boldsymbol{w}_i \\ a_{n,j} &= \boldsymbol{w}_n^{\mathrm{T}}\boldsymbol{\phi}_j^{(n-1)}\Big/\boldsymbol{w}_n^{\mathrm{T}}\boldsymbol{w}_n,\ \ n+1 \le j \le M \\ \boldsymbol{\phi}_j^{(n)} &= \boldsymbol{\phi}_j^{(n-1)} - a_{n,j}\boldsymbol{w}_n,\ \ n+1 \le j \le M. \end{aligned}\right\} \tag{34}$$

Step 3: Regularization Parameter Optimization

1) For the derived $\boldsymbol{w}_n$, iterate the following steps It_2 times:

$$\alpha_n = \boldsymbol{w}_n^{\mathrm{T}}\boldsymbol{w}_n + \lambda_n^{\text{old}} \tag{35}$$

$$\beta_n(k) = \beta_{n-1}(k) - (w_n(k))^2 \big/ \alpha_n,\ \ 1 \le k \le N \tag{36}$$

$$\gamma_n(k) = \frac{w_n(k)}{\alpha_n\beta_n(k)},\ \ 1 \le k \le N \tag{37}$$

$$g_n^{(\mathrm{R})} = \boldsymbol{w}_n^{\mathrm{T}}\boldsymbol{e}^{(n-1)}\big/\alpha_n \tag{38}$$

$$\boldsymbol{e}^{(n)} = \boldsymbol{e}^{(n-1)} - g_n^{(\mathrm{R})}\boldsymbol{w}_n \tag{39}$$

$$\lambda_n^{\text{new}} = \max\left\{\lambda_0, \lambda_n^{\text{old}} - \eta\cdot\operatorname{sign}\left(\frac{\partial J\left(\lambda_n^{\text{old}}\right)}{\partial\lambda_n}\right)\right\} \tag{40}$$

$$\lambda_n^{\text{old}} = \lambda_n^{\text{new}} \tag{41}$$

where $(\partial J(\lambda_n)/\partial\lambda_n) = (2/N)\sum_{k=1}^{N}(e^{(n)}(k)/\beta_n(k))(g_n^{(\mathrm{R})}\gamma_n(k) - e^{(n)}(k)(\gamma_n(k))^2)$.

2) Update the LOOMSE

$$J_n = \frac{1}{N}\sum_{k=1}^{N}\left(e^{(n)}(k)/\beta_n(k)\right)^2. \tag{42}$$

Termination

The OFR procedure is terminated at the (n_s+1)th stage automatically when the condition $J_{n_s+1} \ge J_{n_s}$ is detected, yielding a subset model with the n_s significant regressors.

Remark 3: The existence of this desired model size n_s is guaranteed. This is because initially the LOOMSE decreases as the model size increases, reaching a minimum value at certain model size, and then LOOMSE starts increasing when the model size increases further [22], [23], [30]. Thus, the OFR model selection procedure based on the LOOMSE is guaranteed to automatically "converge" or to terminate at a minimum LOOMSE solution with n_s significant model terms.

The computational complexity of out proposed OFR + LOOMSE algorithm can easily be shown to be

$$\begin{aligned} C_{\text{OFR + LOOMSE}}^{\text{total}} &\approx (n_s+1)\cdot\mathcal{O}\left(N^2 + (\mathrm{It}_1 + \mathrm{It}_2)N\right) \\ &\approx (n_s+1)\cdot\mathcal{O}\left(N^2\right). \end{aligned} \tag{43}$$

Note that with a fixed common kernel width and given all the fixed regularization parameters, our previous LROLS algorithm has the computational complexity given by [23], [32], and [33]

$$C_{\text{LROLS}}^{\text{single-OFR}} \approx (n_s+1)\cdot\mathcal{O}\left(N^2\right) \tag{44}$$

assuming that the OFR procedure also produces a n_s-term model. This confirms that the new OFR + LOOMSE algorithm only requires negligible extra computational cost in comparison to the OFR algorithm with a fixed kernel width and fixed regularization parameters. Further assuming that the iterative loop for updating the regularization parameters requires I_{rps} iterations and the grid search for tuning the single common kernel width employs I_{skw} points, then the total complexity of the LROLS algorithm is approximately

$$C_{\text{LROLS}}^{\text{total}} \approx I_{\text{skw}}\cdot I_{\text{rps}}\cdot(n_s+1)\cdot\mathcal{O}\left(N^2\right). \tag{45}$$

Therefore, an approximate complexity reduction factor of $I_{\text{skw}} \cdot I_{\text{rps}}$ is achieved by the proposed new OFR-LOOMSE algorithm, over our previous LROLS algorithm.

In the following experimental study section, we will demonstrate that the new OFR-LOOMSE algorithm is capable of obtaining very similar sparse models with very similar excellent generalization performance, as the state-of-the-art LROLS algorithm does.

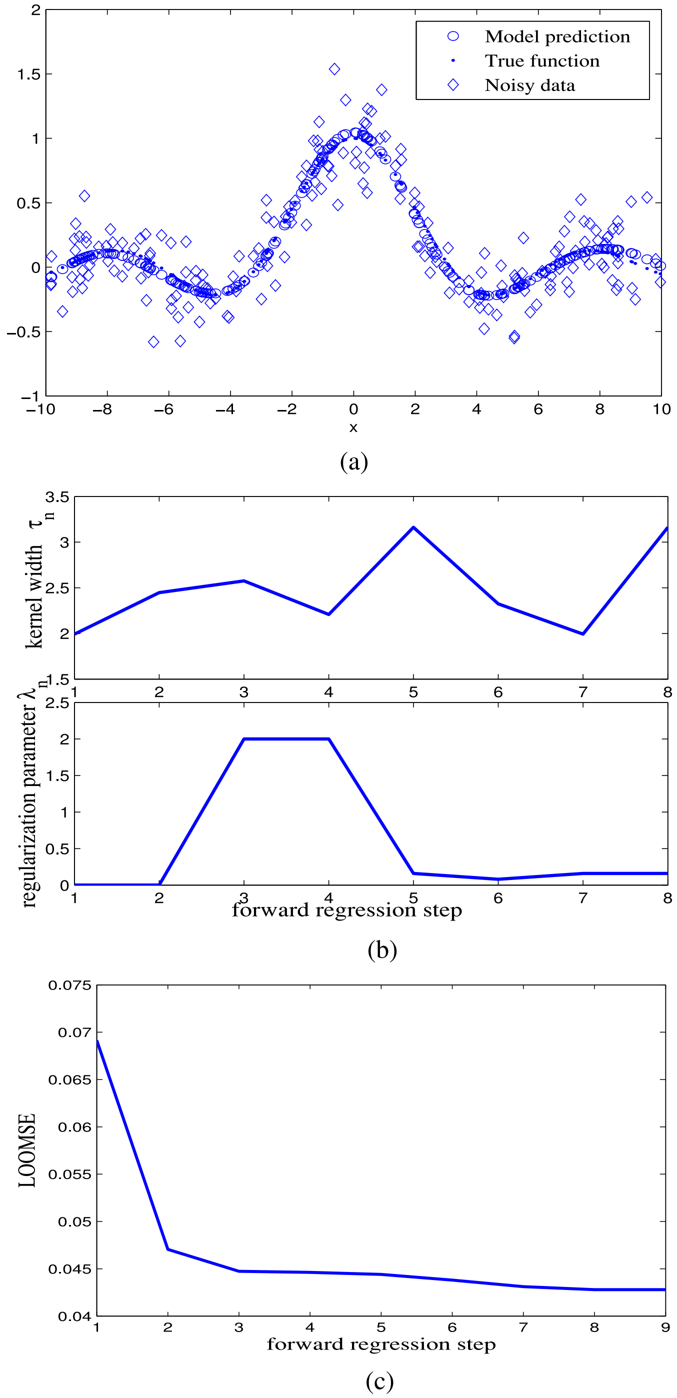
V. Simulation Study

Example 1: Consider using an RBF network to approximate the unknown scalar function

$$f(x) = \frac{\sin(x)}{x}. \tag{46}$$

A data set of two hundred points was generated from $y(x) = f(x) + v$, where the input x was uniformly distributed in the range $[-10,\ 10]$ and the noise v was Gaussian with zero mean and standard deviation 0.2. The noisy data $y(x)$ and the underlying function $f(x)$ are illustrated in Fig. 1(a). The Gaussian function $\phi_i(x) = \exp(-(x - c_i)^2/2\tau_i^2)$ was used as the basis function to construct an RBF model. All the two hundred data points $\{x\}$ were used as the candidate RBF center set for $\{c_i\}$. For the proposed OFR + LOOMSE algorithm, at each OFR step, the initial kernel width was set to $\tau_0 = \sqrt{10}$, and the initial regularization parameter was set to $\lambda_0 = 10^{-8}$. For the line search algorithm of Table I, the iteration number $\text{It}_1 = 10$ and the learning rate $\varsigma = 0.95$. For the 1-D gradient descent algorithm, we set the learning rate to $\eta = 0.02$ and the iteration number to $\text{It}_2 = 30$. At each OFR step, the near optimal kernel width and regularization parameter found for the selected regressor are shown in Fig. 1(b), while Fig. 1(c), indicates that the proposed OFR + LOOMSE algorithm automatically selects a sparse model of size $n_s = 8$ when J_n reaches the minimum at $n = n_s$. The model predictions $\{\widehat{y}(x)\}$ of the resultant eight-term model are also depicted in Fig. 1(a).

For comparison, the ε-SVM algorithm [5], the LASSO using the MATLAB function *lasso.m* with tenfold CV being used to select the associated regularization parameter, and our previous LROLS algorithm [23] were experimented to construct models based on the same kernel function set but with a common kernel width τ for every kernel. The MATLAB function *quadprog.m* was used with the algorithm option set as "interior-point-convex" for the ε-SVM algorithm. The tuning parameters in the ε-SVM algorithm, such as soft margin parameter C [5], were set empirically so that the best possible result is obtained after several trials. The best kernel width value τ for each of these three algorithms was also empirically set after several trials. The mean square errors (MSEs) of the four resulting models over the noisy data set, defined by $E[(\widehat{y}(x) - y(x))^2]$, and the MSEs of the these four models over the true function, defined as $E[(\widehat{y}(x) - f(x))^2]$, are recorded in Table II, where the expectation $E[\,\cdot\,]$ indicates the averaging over the data set $\{x\}$. The results of Table II show that the proposed OFR + LOOMSE algorithm achieves similar excellent performance as our previous state-of-the-art LROLS algorithm, in terms of both model sparsity and model generalization. As mentioned in the previous section, the proposed method significantly reduces the total computational cost in constructing a sparse model.



(a)

(b)

(c)

Fig. 1. Scalar function modeling example. (a) Noisy data, model prediction, and true function. (b) Values of the kernel width parameters and regularization parameters. (c) Evolution of the LOOMSE.

Example 2: The engine data set [24] consists of the 410 data points of the fuel rack position (input $u(k)$) and the engine speed (output $y(k)$), collected from a Leyland TL11 turbocharged, direct injection diesel engine when operated at a low engine speed. Fig. 2(a) depicts the input $u(k)$ and output $y(k)$ of this engine data set. The first 210 data

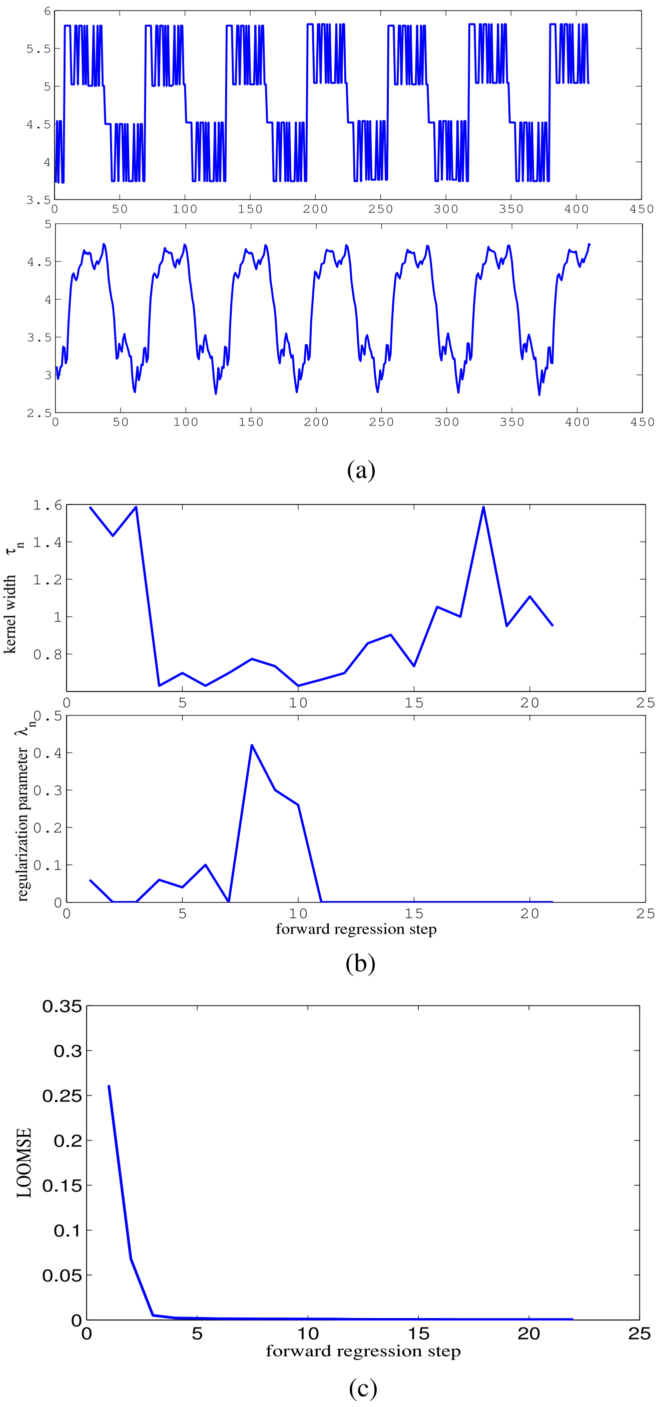
TABLE II
COMPARISON OF THE MODELING PERFORMANCE FOR THE UNKNOWN SCALAR FUNCTION

Algorithm	MSE over the noisy output $y(x)$	MSE over the true function $f(x)$	Model Size
ε-SVM [5]	0.0395	0.0010	200
LASSO	0.0443	0.0052	20
LROLS [23]	0.0389	0.0009	7
The proposed	0.0398	0.0010	8

samples were used in training and the last 200 data samples for model testing. The system input vector was given by $\boldsymbol{x}(k) = [y(k-1)\ u(k-1)\ u(k-2)]^{\mathrm{T}}$. We used the Gaussian RBF kernel (3), and the candidate set for $\{\boldsymbol{c}_i\}$ was formed using all the training data samples. For the proposed OFR + LOOMSE algorithm, at each OFR step, the initial kernel width was chosen to be $\tau_0 = 1$ and the initial regularization parameter was set to $\lambda_0 = 10^{-8}$. Furthermore, the line search algorithmic parameters was set to $\mathrm{It}_1 = 10$ and $\varsigma = 0.95$, while the 1-D gradient descent algorithmic parameters was given by $\eta = 0.02$ and $\mathrm{It}_2 = 30$. At each OFR step, the near optimal kernel width and regularization parameter found for the selected regressor are shown in Fig. 2(b), while Fig. 2(c) indicates that the proposed OFR + LOOMSE algorithm automatically selects a sparse model of size $n_s = 21$ when J_n reaches the minimum at $n = n_s$.

The ε-SVM algorithm [5], the LASSO and our previous LROLS algorithm [23] were used for comparison, based on the same Gaussian kernel set with a common kernel width τ for every kernel. Again, for the ε-SVM, the MATLAB function *quadprog.m* was used with the option set as "interior-point-convex," while the soft margin parameter and other tuning parameters were empirically chosen after several trials. For the LASSO, the MATLAB function *lasso.m* was used. For both the ε-SVM and LASSO, we list the results obtained for a range of kernel width τ values in Table III, in comparison to the results obtained by the proposed OFR + LOOMSE algorithm as well as the LROLS algorithm. The results of Table III show that the ε-SVM can achieve similar test MSE as the proposed algorithm but requires a very large model, and the LASSO algorithm is unable to yield a comparable performance to our algorithm. Moreover, the OFR + LOOMSE algorithm achieves similar excellent performance as the LROLS algorithm, in terms of both model sparsity and model generalization, while imposing a lower computational cost than the latter, as mentioned in the previous section. It is also worth pointing out that the computational cost of the proposed OFR + LOOMSE algorithm is significantly lower than the ε-SVM algorithm and the LASSO algorithm, even when these two algorithms was given a fixed kernel width τ. For example, using MATLAB on a computer with Intel Core i7-3770K CPU, the recorded running time for the proposed algorithm is 0.972248 s, but the LASSO with $\tau = 0.1$ took 26.24080 s.

Example 3: The liquid level data set was collected from a nonlinear liquid level plant, which consisted of a dc water pump feeding a conical flask which in turn fed a square tank. The system input $u(k)$ was the voltage to the pump motor and the system output $y(k)$ was the water level in the conical flask.



(a)

(b)

(c)

Fig. 2. Engine data set modeling example. (a) System input $u(k)$ and output $y(k)$. (b) Values of the kernel width parameters and regularization parameters. (c) Evolution of the LOOMSE.

A description of this nonlinear process can be found in [35] and Fig. 3(a) shows the 1000 data samples of the data set used in this experiment. From the data set, 1000 data points $\{\boldsymbol{x}(k), y(k)\}$ were constructed with

$$\boldsymbol{x}(k) = [y(k-1)\ y(k-2)\ y(k-3)\ u(k-1)\\ u(k-2)\ u(k-3)\ u(k-4)]^{\mathrm{T}}. \quad (47)$$

The first 500 pairs of the data were used for training and the remaining 500 pairs for testing the constructed model.

TABLE III
COMPARISON OF MODELING PERFORMANCE FOR THE ENGINE DATA SET

Algorithm	MSE over training data set	MSE over test data set	Model size
ε-SVM (τ=3)	0.000502	0.000482	208
ε-SVM (τ=2.5)	0.000480	0.000475	208
ε-SVM (τ=2)	0.000461	0.000486	208
ε-SVM (τ=1.5)	0.000415	0.000579	208
ε-SVM (τ=1)	0.000370	0.000794	208
LASSO (τ=1.5)	0.000923	0.001010	70
LASSO (τ=1)	0.000708	0.000748	44
LASSO (τ=0.5)	0.000706	0.000842	54
LASSO (τ=0.2)	0.000565	0.000800	81
LASSO (τ=0.1)	0.000644	0.001907	76
LROLS [23]	0.000453	0.000490	22
The proposed	0.000477	0.000484	21

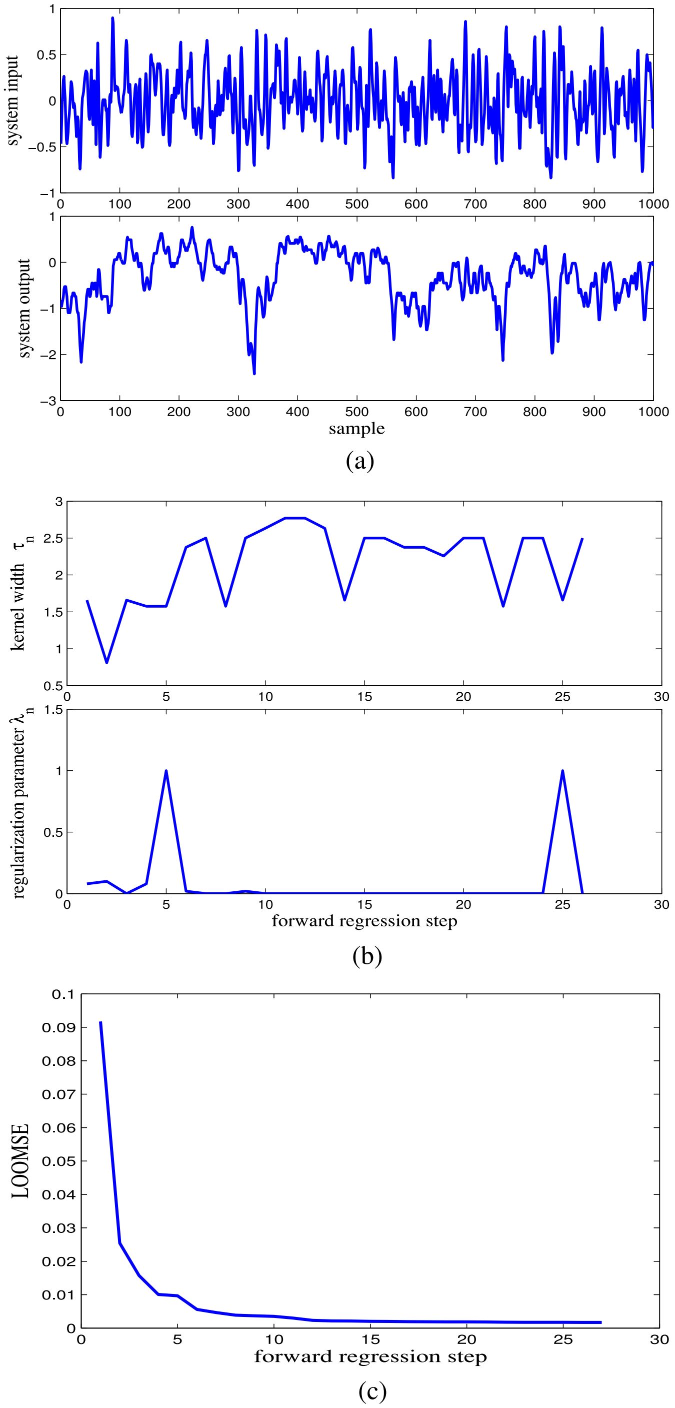
The Gaussian RBF kernel (3) was employed, and the candidate set for $\{\boldsymbol{c}_i\}$ was formed using all the training data points. The initial kernel width was chosen as $\tau_0 = 2.5$ and the initial regularization parameter was set to $\lambda_0 = 10^{-8}$. For the line search algorithm of Table III, the iteration number $\mathrm{It}_1 = 10$ and the learning rate $\varsigma = 0.95$, while for the 1-D gradient descent algorithm, we set the learning rate to $\eta = 0.02$ and the iteration number to $\mathrm{It}_2 = 30$. At each OFR step, the near optimal kernel width and regularization parameter found for the selected regressor are depicted in Fig. 3(b), while Fig. 3(c) indicates that the OFR + LOOMSE algorithm automatically selects a sparse model of size $n_s = 26$ when J_n reaches the minimum at $n = n_s$.

For comparison, the ε-SVM algorithm [5], the LASSO algorithm and the LROLS algorithm [23] were also applied based on the same Gaussian kernel set but with a common kernel width τ for every kernel. The ε-SVM was implemented using the MATLAB function *quadprog.m* with the algorithm option set as interior-point-convex, and the soft margin and other tuning parameters were empirically chosen after several trials. For the LASSO, the MATLAB function *lasso.m* was used with tenfold CV for selecting the associated regularization parameter. For both the ε-SVM and LASSO algorithms, we list the results obtained for a range of kernel width τ values in Table IV, in comparison to the results obtained by our proposed OFR + LOOMSE algorithm and our previous LROLS algorithm. The modeling results of Table IV again demonstrate that the proposed OFR + LOOMSE algorithm has comparable performance to the LROLS algorithm [23], in terms of both model generalization and model sparsity. The ε-SVM algorithm is able to achieve a similar test MSE as the proposed OFR + LOOMSE algorithm but requires a very large model. The LASSO algorithm is able to obtain a sparser model but yields a poorer test MSE. Note that the proposed OFR + LOOMSE algorithm is computationally much more efficient than the ε-SVM and LASSO algorithms. Using MATLAB on a computer with Intel Core i7-3770K CPU, the recorded running time for the proposed algorithm, which includes optimizing the kernel width τ_n at each OFR step, is 3.361728 s, compared to 26.685302 s required by the LASSO with the kernel width fixed to $\tau = 3$.

Example 4: The gas furnace data set (the time series J in [36]) contained 296 pairs of input-output points as depicted in Fig. 4(a), where the input $u(k)$ was the coded input gas feed rate and the output $y(k)$ represented the CO$_2$ concentration from the gas furnace. The Gaussian RBF kernel (3) was employed with the system input vector given by

$$\boldsymbol{x}(k) = [y(k-1)\ y(k-2)\ y(k-3)\\ u(k-1)\ u(k-2)\ u(k-3)]^{\mathrm{T}}. \quad (48)$$

From Fig. 4(a), it can be seen that the second half of the data set is significantly different from the first half. Therefore,



(a)

(b)

(c)

Fig. 3. Liquid level data set modeling example. (a) System input $u(k)$ and output $y(k)$. (b) Values of the kernel width parameters and regularization parameters. (c) Evolution of the LOOMSE.

TABLE IV
COMPARISON OF MODELING PERFORMANCE FOR THE LIQUID LEVEL DATA SET

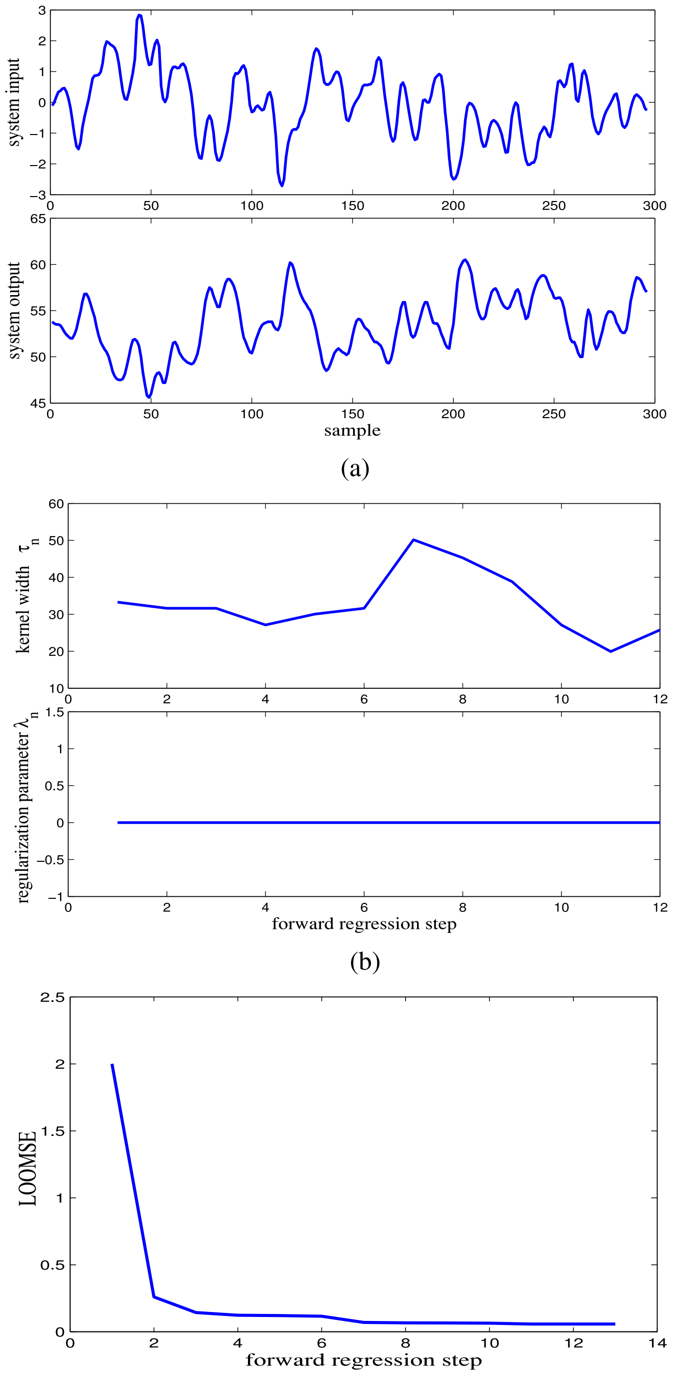
Algorithm	MSE over training data set	MSE over test data set	Model size
ε-SVM (τ=6)	0.00154	0.00229	488
ε-SVM (τ=5)	0.00139	0.00257	493
ε-SVM (τ=4)	0.00131	0.00289	487
ε-SVM (τ=3)	0.00113	0.00499	490
LASSO (τ=3)	0.00386	0.00432	14
LASSO (τ=2.5)	0.00334	0.00377	13
LASSO (τ=2)	0.00251	0.00370	42
LASSO (τ=1.5)	0.00224	0.00429	28
LROLS [23]	0.00140	0.00253	30
The proposed	0.00154	0.00245	26

we used the even-number pairs of $\{\boldsymbol{x}(k)y(k)\}$ for training and the odd-number pairs for testing. The candidate RBF center set $\{\boldsymbol{c}_i\}$ was formed using all the training data samples. The initial kernel width was given by $\tau_0 = \sqrt{1000}$ and the initial regularization parameter was set to $\lambda_0 = 10^{-8}$. For the line search algorithm of Table I, the iteration number $\text{It}_1 = 10$ and the learning rate $\varsigma = 0.95$, while the 1-D gradient descent algorithm used the learning rate of $\eta = 0.02$ and the iteration number of $\text{It}_2 = 30$. At each OFR step, the optimal kernel width and regularization parameter found for the selected regressor is shown in Fig. 4(b), while Fig. 4(c), indicates that the proposed OFR + LOOMSE algorithm automatically selects a sparse model of size $n_s = 12$ when J_n reaches the minimum at $n = n_s$.

The ε-SVM algorithm [5], the LASSO algorithm and the LROLS algorithm [23] were used based on the same Gaussian kernel set but with a common kernel width τ for every kernel, for a comparison. The results of the ε-SVM algorithm and LASSO algorithm based on the common kernel variance of $\tau^2 = 500$, 1000, and 2000, respectively, are compared with the results obtained by the OFR + LOOMS and LROLS algorithms in Table V. The modeling results of Table V demonstrate that for this example the proposed OFR + LOOMS algorithm attains the best test MSE performance.

Example 5: The Boston housing data set, available at the UCI repository [37], comprised 506 data points with 14 variables. We performed the task of predicting the median house value from the remaining 13 attributes. We randomly selected 456 data points from the data set for training and used the remaining 50 data points to form the test set. Average results were given over 100 realizations. For each realization, 13 input attributes were normalized so that each attribute has zero mean and standard deviation of one. The Gaussian RBF model was constructed from the 456 candidate regressors of each realization. For the proposed OFR + LOOMSE algorithm, at each OFR step, the initial kernel width was set to $\tau_0 = 3$ and the initial regularization parameter was set to $\lambda_0 = 10^{-8}$, while the learning rates $\eta = 0.002$ and $\varsigma = 0.95$ as well as the iteration numbers $\text{It}_1 = 10$ and $\text{It}_2 = 50$ were used.

In Table VI, the modeling performance of the proposed OFR + LOOMSE algorithm is compared with the results of the ε-SVM [5] and the LROLS [23], which are quoted from [32]. We also experimented with the LASSO algorithm supplied by MATLAB *lasso.m* with option set as tenfold CV to select the associated regularization parameter. For the LASSO, a common kernel width τ was set for constructing the kernel model from the 456 candidate regressors of each realization, and a range of τ values were experimented. As shown in Table VI, the proposed OFR + LOOMSE algorithm is very competitive in terms of model size and model generalization capability, compared with the LROLS algorithm and the LASSO algorithm. As analyzed in Section IV,



(a)

(b)

(c)

Fig. 4. Gas furnace data set example. (a) System input $u(k)$ and output $y(k)$. (b) Values of the kernel width parameters and regularization parameters. (c) Evolution of the LOOMSE.

TABLE V
COMPARISON OF MODELING PERFORMANCE FOR THE GAS FURNACE DATA SET

Algorithm	MSE over training data set	MSE over test data set	Model size
ε-SVM (τ^2=500)	0.0470	0.0822	142
ε-SVM (τ^2=1000)	0.0493	0.0855	145
ε-SVM (τ^2=2000)	0.0499	0.0751	146
LASSO (τ^2=500)	0.0964	0.0988	10
LASSO (τ^2=1000)	0.0942	0.0938	7
LASSO (τ^2=2000)	0.0994	0.0957	9
LROLS [23]	0.0474	0.0805	12
The proposed	0.0533	0.0730	12

TABLE VI
COMPARISON OF MODELING PERFORMANCE FOR THE BOSTON HOUSE DATA SET. THE RESULTS WERE AVERAGED OVER 100 REALIZATIONS AND GIVEN AS MEAN ± STANDARD DEVIATION, AND THE RESULTS FOR THE ε-SVM AND THE LROLS ARE QUOTED FROM [32]

Algorithm	MSE over training data set	MSE over test data set	Model size
ε-SVM [5]	6.80 ± 0.44	23.18 ± 9.05	243 ± 5.3
LASSO (τ=2)	8.52 ± 3.57	14.37 ± 8.15	76.8 ± 39.7
LASSO (τ=3)	8.55 ± 1.07	13.31 ± 6.65	68.6 ± 29.3
LASSO (τ=5)	10.45 ± 1.07	15.05 ± 8.37	85.9 ± 19.7
LASSO (τ=10)	16.42 ± 1.78	19.39 ± 8.31	29.9 ± 21.3
LROLS [23]	12.97 ± 2.67	17.42 ± 4.67	58.6 ± 11.3
The proposed	8.56 ± 1.50	13.54 ± 6.80	33.7 ± 10.5

the computational cost of the OFR + LOOMSE algorithm is lower than that of the LROLS algorithm. Also, the running time of the OFR + LOOMSE algorithm over 100 realizations was recorded as 126.02 s, much faster than the LASSO algorithm with $\tau = 3$ which took 1259.9 s.

VI. CONCLUSION

We have shown how to optimize the pairs of kernel width and regularization parameters one pair at a time within the OFR procedure with the aim of maximizing the model generalization capability by proposing a new OFR + LOOMSE algorithm based on the Gaussian RBF model with tunable kernel widths for nonlinear system identification application. During each stage of the OFR, the LOOMSE is used as the selection criterion to select one regressor from the candidate set, while the kernel width of the selected regressor is tuned followed by optimizing the associated regularization parameter for the tuned regressor, both also based on the LOOMSE. Since the kernel width parameters as well as regularization parameters are optimized within the OFR algorithm, there is no need to iteratively run the OFR for choosing these parameters, as many other existing schemes do, including our previous state-of-the-art LROLS algorithm. Computational analysis has shown that the total computational cost of the proposed OFR + LOOMSE algorithm is lower than that of the LROLS algorithm which is well-known to be a highly efficient method for constructing sparse models that generalize well. Five examples, including three real-world nonlinear system identification applications, have been included to demonstrate the effectiveness this new data modeling approach, in comparison to several existing approaches, in terms of model generalization performance and model sparsity. Although the proposed algorithm is presented based on the Gaussian RBF model, it can be applied to other types of linear-in-the-parameter models.

REFERENCES

[1] A. E. Ruano, Ed., *Intelligent Control Systems Using Computational Intelligence Techniques*. London, U.K.: IEE Publishing, 2005.

[2] R. Murray-Smith and T. A. Johansen, *Multiple Model Approaches to Modeling and Control*. Bristol, PA, USA: Taylor and Francis, 1997.

[3] S. G. Fabri and V. Kadirkamanathan, *Functional Adaptive Control: An Intelligent Systems Approach*. London, U.K.: Springer, 2001.

[4] 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.

[5] S. R. Gun, "Support vector machines for classification and regression," Dept. Electron. Comput. Sci., ISIS Group, University of Southampton, Southampton, U.K., May 1998.

[6] G. C. Cawley and N. L. C. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *J. Mach. Learn. Res.*, vol. 11, pp. 2079–2107, Jul. 2010.

[7] 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.

[8] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *J. Royal Stat. Soc. B*, vol. 36, no. 2, pp. 111–147, 1974.

[9] G. H. Golub, M. Heath, and G. Wahba, "Generalized cross-validation as a method for choosing good ridge parameter," *Technometrics*, vol. 21, no. 2, pp. 215–223, 1979.

[10] S. Chen, Y. Wu, and B. L. Luk, "Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1239–1243, Sep. 1999.

[11] M. J. L. Orr, "Regularisation in the selection of radial basis function centers," *Neural Comput.*, vol. 7, no. 3, pp. 606–623, 1995.

[12] X. Hong and S. A. Billings, "Parameter estimation based on stacked regression and evolutionary algorithms," *IEE Proc. Control Theory Appl.*, vol. 146, no. 5, pp. 406–414, Sep. 1999.

[13] L. Ljung and T. Glad, *Modeling of Dynamic Systems*. Englewood Cliffs, NJ, USA: Prentice Hall, 1994.

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

[15] M. J. Korenberg, "Identifying nonlinear difference equation and functional expansion representations: The fast orthogonal algorithm," *Ann. Biomed. Eng.*, vol. 16, no. 1, pp. 123–142, 1988.

[16] D. Du, X. Li, M. Fei, and G. W. Irwin, "A novel locally regularized automatic construction method for RBF neural models," *Neurocomputing*, vol. 98, pp. 4–11, Dec. 2012.

[17] L. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 807–814, Sep. 1992.

[18] X. Hong and C. J. Harris, "Neurofuzzy design and model construction of nonlinear dynamical processes from data," *IEE Proc. Control Theory Appl.*, vol. 148, no. 6, pp. 530–538, Nov. 2001.

[19] Q. Zhang, "Using wavelets network in nonparametric estimation," *IEEE Trans. Neural Netw.*, vol. 8, no. 2, pp. 227–236, Mar. 1993.

[20] S. A. Billings and H. L. Wei, "The wavelet-NARMAX representation: A hybrid model structure combining polynomial models with multiresolution wavelet decompositions," *Int. J. Syst. Sci.*, vol. 36, no. 3, pp. 137–152, 2005.

[21] R. H. Myers, *Classical and Modern Regression with Applications*, 2nd Ed. Boston, MA, USA: PWS-KENT, 1990.

[22] X. Hong, P. M. Sharkey, and K. Warwick, "Automatic nonlinear predictive model construction using forward regression and the PRESS statistic," *IEE Proc. Control Theory Appl.*, vol. 150, no. 3, pp. 245–254, May 2003.

[23] S. Chen, X. Hong, C. J. Harris, and P. M. Sharkey, "Sparse modeling using orthogonal forward regression with PRESS statistic and regularization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 898–911, Apr. 2004.

[24] S. A. Billings, S. Chen, and R. Backhouse, "The identification of linear and nonlinear models of a turbocharged automotive diesel engine," *Mech. Syst. Signal Process.*, vol. 3, no. 2, pp. 123–142, 1989.

[25] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[26] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *J. Royal Stat. Soc. B*, vol. 58, no. 1, pp. 267–288, 1996.

[27] B. Efron, I. Johnstone, T. Hastie, and R. Tibshirani, “Least angle regression,” *Ann. Stat.*, vol. 32, no. 2, pp. 407–451, 2004.

[28] S. Chen, “Locally regularized orthogonal least squares algorithm for the construction of sparse kernel regression models,” in *Proc. 6th Int. Conf. Signal Process.*, Beijing, China, 2002, pp. 1229–1232.

[29] D. J. C. MacKay, “Bayesian methods for adaptive models,” Ph.D. dissertation, Dept. Comput. Neural Syst., California Inst. Technol., Pasadena, CA, USA, 1992.

[30] S. Chen, X. Hong, and C. J. Harris, “Sparse kernel regression modeling using combined locally regularized orthogonal least squares and D-optimality experimental design,” *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 1029–1036, Jun. 2003.

[31] S. Chen, X. Hong, and C. J. Harris, “Non-linear system identification using particle swarm optimization tuned radial basis function models,” *Int. J. Bio-Ins. Comput.*, vol. 1, no. 4, pp. 246–258, 2009.

[32] S. Chen, X. Hong, and C. J. Harris, “Construction of tunable radial basis function networks using orthogonal forward selection,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 457–466, Apr. 2009.

[33] S. Chen, X. Hong, and C. J. Harris, “Particle swarm optimization aided orthogonal forward regression for unified data modeling,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 4, pp. 477–499, Aug. 2010.

[34] S. Chen and S. A. Billings, “Representation of nonlinear systems: The NARMAX model,” *Int. J. Control*, vol. 49, no. 3, pp. 1013–1032, 1989.

[35] S. A. Billings and W. Voon, “A prediction-error and stepwise-regression estimation algorithm for nonlinear systems,” *Int. J. Control*, vol. 44, no. 3, pp. 803–822, 1986.

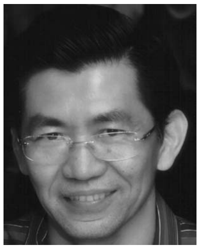
[36] G. E. P. Box and G. M. Jenkins, *Time Series Analysis, Forecasting and Control*. San Francisco, CA, USA: Holden-Day, 1976.

[37] A. Frank and A. Asuncion. (2010). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml

Xia Hong (SM’02) received the B.Sc. and M.Sc. degrees from the National University of Defense Technology, Changsha, China, in 1984 and 1987, respectively, and the Ph.D. degree from the University of Sheffield, Sheffield, U.K., in 1998, all in automatic control.

She was a Research Assistant at the Beijing Institute of Systems Engineering, Beijing, China, from 1987 to 1993. She was a Research Fellow at the Department of Electronics and Computer Science, University of Southampton, Southampton, U.K., from 1997 to 2001. She is currently a Professor with the School of Systems Engineering, University of Reading, Reading, U.K. She is actively engaged in research of nonlinear systems identification, data modeling, estimation and intelligent control, neural networks, pattern recognition, learning theory, and their applications. She has published over 200 research papers, and co-authored a research book.

Prof. Hong was the recipient of the Donald Julius Groen Prize by IMechE in 1999.



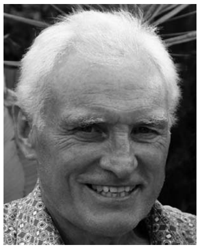
Sheng Chen (M’90–SM’97–F’08) received the B.Eng. degree from the East China Petroleum Institute, Dongying, China, the Ph.D. degree from the City University London, London, U.K., in 1982 and 1986, respectively, both in control engineering, and the D.Sc. degree from the University of Southampton, Southampton, U.K., in 2005.

From 1986 to 1999, he held research and academic appointments at 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 Department of Electronics and Computer Science, University of Southampton, where he is currently a Professor of Intelligent Systems and Signal Processing. His current research interests include adaptive signal processing, wireless communications, modeling and identification of nonlinear systems, neural network and machine learning, intelligent control system design, evolutionary computation methods, and optimization. He has published over 500 research papers. He was an ISI highly cited researcher in engineering in 2004.

Dr. Chen is a Distinguished Adjunct Professor at King Abdulaziz University, Jeddah, Saudi Arabia. He is a fellow of the IET. He was elected as a fellow of the U.K. Royal Academy of Engineering in 2014.

Junbin Gao received the B.Sc. degree in computational mathematics from the Huazhong University of Science and Technology (HUST), Wuhan, China, and the Ph.D. degree from the Dalian University of Technology, Dalian, China, in 1982 and 1991, respectively.

He is a Professor of Computing Science with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW, Australia. He was a Senior Lecturer and a Lecturer of Computer Science at the University of New England, Armidale, NSW, Australia, from 2001 to 2005. From 1982 to 2001, he was an Associate Lecturer, a Lecturer, an Associate Professor, and a Professor at the Department of Mathematics, HUST. His current research interests include machine learning, data mining, Bayesian learning and inference, and image analysis.



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

He is a Emeritus Research Professor with the University of Southampton. He held senior academic appointments at Imperial College, London, U.K., the University of Oxford, and the University of Manchester, Manchester, U.K. He was a Deputy Chief Scientist for the U.K. Government. He has co-authored over 450 scientific research papers.

Prof. Harris was the recipient of 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.