Available online at www.sciencedirect.com

SCIENCE DIRECT®

Neurocomputing 69 (2006) 559–585

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Local regularization assisted orthogonal least squares regression

S. Chen*

School of Electronics and Computer Science, University of Southampton, Highfield, Southampton SO17 1BJ, UK

Received 29 June 2004; received in revised form 24 November 2004; accepted 16 December 2004
Available online 1 June 2005
Communicated by J. Zhang

Abstract

A locally regularized orthogonal least squares (LROLS) algorithm is proposed for constructing parsimonious or sparse regression models that generalize well. By associating each orthogonal weight in the regression model with an individual regularization parameter, the ability for the orthogonal least squares model selection to produce a very sparse model with good generalization performance is greatly enhanced. Furthermore, with the assistance of local regularization, when to terminate the model selection procedure becomes much clearer. A comparison with a state-of-the-art method for constructing sparse regression models, known as the relevance vector machine, is given. The proposed LROLS algorithm is shown to possess considerable computational advantages, including well conditioned solution and faster convergence speed.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Orthogonal least squares algorithm; Regularization; Regression; Bayesian learning; Relevance vector machines; Evidence procedure

*Tel./fax: +44 23 8059 6660/4508.
E-mail address: sqc@ecs.soton.ac.uk.

0925-2312/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2004.12.011

1. Introduction

A basic principle in practical nonlinear data modeling is the parsimonious principle of ensuring the smallest possible model that explains the data. The orthogonal least squares (OLS) algorithm [9,11] is an efficient learning procedure for constructing sparse regression models. A key feature of the OLS algorithm is its ability to reveal the contribution of individual selected model regressor to modeling accuracy. This enables the selection of only those significant regressors and is responsible for producing parsimonious models (see [9,11]). A simple mechanism is automatically built into the OLS algorithm to avoid any ill-conditioning of learning problems. A data modeling problem becomes ill-conditioned if some of the eigenvalues of its associated design matrix are almost zero. For the OLS selection procedure, as a byproduct of orthogonalization, the eigenvalue or the energy of each candidate regressor is explicitly given. Thus if the energy of a candidate regressor is smaller than a threshold value, it will not be considered at all. Some other well-known construction algorithms based on the parsimonious principle can be found in [16,20,6,7]. The parsimonious principle alone however is not entirely immune to over-fitting. If data are highly noisy, small models constructed may still fit into noise. A useful technique for overcoming over-fitting is regularization [19,4,23]. By combining the parsimonious principle with a regularization method, a regularized OLS algorithm has been developed [10], which is capable of constructing sparse models with excellent generalization properties under severely noisy conditions. This regularized OLS algorithm employs a single common regularization parameter for every orthogonal weights in the model. For this reason, it is referred to as the uniformly regularized OLS (UROLS) algorithm.

It is worth pointing out an obvious but often overlooked property of the OLS algorithm. The subset model selection is carried out in the transformed orthogonal space, but the selected subset of the orthogonal regressors or bases corresponds precisely to a subset of the original model bases, that is, the algorithm actually selects a subset of the original model regressors. In other words, the sub-space spanned by the selected orthogonal bases is the same space spanned by the corresponding subset of the original model regressors. This is very different to the situation in many signal processing applications, where the objective is to transform the original signal space onto a new space and to tackle the problem on a transformed subspace. This can be achieved, for example, by using singular value decomposition (SVD) [17]. A subset of the orthonormal bases, which correspond to the largest eigenvalues, is selected to form the required subspace. Because each orthogonal base is a linear combination of all the original model regressors, it is not known which subset of the original model regressors can exactly represent the subspace spanned by the used orthogonal bases. When a subset model consisting of a subset of the original model regressors is required, the OLS algorithm has clear advantages. The one-to-one property of the OLS algorithm also ensures that introducing regularization in the orthogonal weight space is equivalent to introducing regularization in the original model weight space.

Recently, the support vector machine (SVM) method [27] has been gaining popularity and has been regarded as the state-of-the-art technique for regression and classification applications. It is believed that the formulation of SVM embodies the structural risk minimization principle, thus combining excellent generalization properties with a sparse model representation. Despite of these attractive features and many good empirical results obtained using the SVM method, data modeling practicians have begun to realize that the ability for the SVM method to produce sparse models has perhaps been overstated. For example, it has been shown that the standard SVM technique is not always able to construct parsimonious models in system identification [15]. A recent study [21] has compared the SVM and UROLS algorithms using time series prediction problems, and has found that the both methods have similar excellent generalization performance but the UROLS algorithm is able to produce much sparser models. In an application to communication multiuser detection [12], which is a classification problem, the SVM method performs well but the resulting detector model is not sparse enough. The fact that the SVM technique may not guarantee a sufficiently sparse model is the motivation for Tipping [26] to introduce the relevance vector machine (RVM) method.

The RVM method can be viewed from a Bayesian learning framework [23,24], and it has an identical functional form to the SVM. The results given in [26] have demonstrated that the RVM has a comparable generalization performance to the SVM but requires dramatically fewer kernel functions or model terms than the SVM. The introduction of an individual hyperparameter for every weight of the regression model is the key feature of the RVM method, and is ultimately responsible for the sparsity properties of the RVM method [26]. During the optimization process, many of these hyperparameters are driven to large values, so that the corresponding model weights are effectively forced to be zero. Thus the corresponding model terms are removed from the trained model. A drawback of the RVM method is a significant increase in computational complexity, compared with the SVM method. The iterative optimization process involved in the RVM method is inherently ill-conditioned, and computationally intensified and numerically robust methods, such as the SVD or other pseudo-inverse algorithms, often have to be used to solve for the corresponding optimization problem.

In this paper, an individually regularized approach is adopted to assist the OLS model selection. In this approach, each candidate regressor has an associated individual regularization parameter. From the Bayesian viewpoint, a regularization parameter is equivalent to the ratio of the related hyperparameter to the noise parameter [23]. In this sense, the proposed locally regularized orthogonal least squares (LROLS) algorithm bears some relationship to the RVM method. However, the LROLS algorithm has the ability to reveal the significance of individual model regressor and only selects those significant terms, just as the original OLS algorithm. Local regularization further help to enhance the sparsity of the selected model. The computation requirements are simple and the associated optimization process does not suffer from numerical ill-conditioning. As in [10], the regularization is introduced in the orthogonal regression weight space rather than the original regression weight space. Thus the Hessian matrix associated with the updating of regularization parameters is diagonal. This further simplifies the computational requirement of the iterative optimization process. In the original OLS model selection procedure, when to terminate the selection process is often not a clear cut. With the individually regularized approach, deciding when to terminate selection becomes much simpler. It should be emphasized that local regularization or smoothing has been considered before [25]. However, in [25] regularization is done in the original regression weight space. Thus updating of regularization parameters requires intensive computation and the problem is ill-conditioned. The algorithm of [25] does not select significant model terms and in this sense it is similar to the RVM method.

Before proceeding to the derivation of the LROLS algorithm, the choice of specific regularization scheme is commented. Different regularization schemes can be interpreted as different choices of prior in Bayesian learning. A commonly used Bayes prior is Gaussian prior [23,24]. Another choice is Laplacian prior [1,18], which is also known to produce very sparse linear regression models. A Gaussian prior is adopted in the derivation of the LROLS algorithm. The resulting regularizer is quadratic and fits naturally into the quadratic cost function for subset model selection. Furthermore, the evidence procedure [23] can readily be applied to optimize the regularization parameters. A general limitation inherent to the evidence framework is that the computation of the associated Hessian matrix is expensive and it is possible that this Hessian matrix is singular or near singular, and thus non-invertible. Some eigenvalues of the Hessian matrix may even become negative numerically [5], and thus cause numerical instability. It is worth re-emphasizing that this difficulty does not exist in the LROLS algorithm. The Hessian matrix required is readily provided by the OLS selection process, and it is well-conditioned and diagonal. The inverse of the Hessian matrix is thus trivial. This ensures that the iterative procedure for updating regularization parameters in the LROLS algorithm is well behaved and converges fast.

2. The regression model

Consider the general discrete-time nonlinear system represented by the nonlinear model [8]:

$$y(k) = f(y(k-1), \ldots, y(k-n_y), u(k-1), \ldots, u(k-n_u)) + e(k) = f(\mathbf{x}(k)) + e(k), \tag{1}$$

where $u(k)$ and $y(k)$ are the system input and output variables, respectively, n_u and n_y are positive integers representing the lags in $u(k)$ and $y(k)$, respectively, $e(k)$ is the system white noise, $\mathbf{x}(k) = [y(k-1) \; \cdots \; y(k-n_y) \; u(k-1) \; \cdots \; u(k-n_u)]^{\mathrm{T}}$ denotes the system "input" vector, and $f(\bullet)$ is the unknown system mapping. The system model (1) is to be identified from an N-sample observation data set $\{\mathbf{x}(k), y(k)\}_{k=1}^{N}$ using some suitable functional which can approximate $f(\bullet)$ with arbitrary accuracy. One class of such functionals is the kernel regression model of the form:

$$y(k) = \hat{y}(k) + e(k) = \sum_{i=1}^{n_{\mathrm{M}}} \theta_i \phi_i(k) + e(k), \quad 1 \leqslant k \leqslant N, \tag{2}$$

where $\hat{y}(k)$ denotes the model output, θ_i are the model weights, $\phi_i(k) = \phi_i(\mathbf{x}(k))$ are the regressors, and n_{M} is the total number of candidate regressors or model terms. The model (2) is very general and includes, for example, all the kernel based models, the polynomial-expansion model [9], and the general "linear-in-the-parameter" nonlinear model [2]. By defining

$$\mathbf{y} = [y(1) \ \cdots \ y(N)]^{\mathrm{T}}, \tag{3}$$

$$\mathbf{\Phi} = [\mathbf{\Phi}_1 \cdots \mathbf{\Phi}_{n_{\mathrm{M}}}], \tag{4}$$

$$\mathbf{\Phi}_i = [\phi_i(1) \ \cdots \ \phi_i(N)]^{\mathrm{T}}, \tag{5}$$

$$\boldsymbol{\theta} = [\theta_1 \ \cdots \ \theta_{n_{\mathrm{M}}}]^{\mathrm{T}}, \tag{6}$$

$$\mathbf{e} = [e(1) \ \cdots \ e(N)]^{\mathrm{T}}, \tag{7}$$

the regression model (2) can be rewritten in the matrix form

$$\mathbf{y} = \mathbf{\Phi}\boldsymbol{\theta} + \mathbf{e}. \tag{8}$$

Let an orthogonal decomposition of the regression matrix $\mathbf{\Phi}$ be

$$\mathbf{\Phi} = \mathbf{W}\mathbf{A}, \tag{9}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,n_{\mathrm{M}}} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n_{\mathrm{M}}-1,n_{\mathrm{M}}} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{10}$$

and

$$\mathbf{W} = [\mathbf{w}_1 \cdots \mathbf{w}_{n_{\mathrm{M}}}] \tag{11}$$

with orthogonal columns that satisfy $\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_j = 0$, if $i \neq j$. The regression model (8) can alternatively be expressed as

$$\mathbf{y} = \mathbf{W}\mathbf{g} + \mathbf{e}, \tag{12}$$

where the orthogonal regression weight vector $\mathbf{g} = [g_1 \ \cdots \ g_{n_{\mathrm{M}}}]^{\mathrm{T}}$ satisfy the triangular system

$$\mathbf{A}\boldsymbol{\theta} = \mathbf{g}. \tag{13}$$

Knowing $\mathbf{A}$ and $\mathbf{g}$, $\boldsymbol{\theta}$ can readily be solved from (13).

3. The locally regularized OLS algorithm

The LROLS algorithm is based on the following regularized error criterion:

$$J_{\mathrm{R}}(\mathbf{g}, \boldsymbol{\lambda}) = \mathbf{e}^{\mathrm{T}}\mathbf{e} + \sum_{i=1}^{n_{\mathrm{M}}} \lambda_i g_i^2 = \mathbf{e}^{\mathrm{T}}\mathbf{e} + \mathbf{g}^{\mathrm{T}}\boldsymbol{\Lambda}\mathbf{g}, \tag{14}$$

where $\boldsymbol{\lambda} = [\lambda_1 \cdots \lambda_{n_{\mathrm{M}}}]^{\mathrm{T}}$ is the regularization parameter vector, and $\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \ldots, \lambda_{n_{\mathrm{M}}}\}$. The UROLS algorithm [10] and the original OLS algorithm [9] can be viewed as the two special cases of this LROLS algorithm by setting $\lambda_i = \lambda$ and $\lambda_i = 0 \; \forall i$, respectively. After some simplification (see Appendix A), the criterion (14) can be expressed as

$$\mathbf{e}^{\mathrm{T}}\mathbf{e} + \mathbf{g}^{\mathrm{T}}\boldsymbol{\Lambda}\mathbf{g} = \mathbf{y}^{\mathrm{T}}\mathbf{y} - \sum_{i=1}^{n_{\mathrm{M}}} (\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_i + \lambda_i) g_i^2. \tag{15}$$

Normalizing (15) by $\mathbf{y}^{\mathrm{T}}\mathbf{y}$ yields

$$(\mathbf{e}^{\mathrm{T}}\mathbf{e} + \mathbf{g}^{\mathrm{T}}\boldsymbol{\Lambda}\mathbf{g})/\mathbf{y}^{\mathrm{T}}\mathbf{y} = 1 - \sum_{i=1}^{n_{\mathrm{M}}} (\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_i + \lambda_i) g_i^2/\mathbf{y}^{\mathrm{T}}\mathbf{y}. \tag{16}$$

As in the case of the OLS algorithm [9], the regularized error reduction ratio due to $\mathbf{w}_i$ is defined by

$$[\mathrm{rerr}]_i = (\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_i + \lambda_i) g_i^2/\mathbf{y}^{\mathrm{T}}\mathbf{y}. \tag{17}$$

Based on this ratio, significant regressors can be selected in a forward-regression procedure similar to the case of the OLS algorithm [9]. The selection is terminated at the n_{s}th stage when

$$1 - \sum_{l=1}^{n_{\mathrm{s}}} [\mathrm{rerr}]_l < \xi, \tag{18}$$

is satisfied, where $0 < \xi < 1$ is a chosen tolerance. This produces a sparse model containing n_{s} ($\ll n_{\mathrm{M}}$) significant regressors. The detailed algorithm selection procedure is given in Appendix B. Note that, in the selection procedure, if $\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_i$ is too small (near zero), this term will not be selected. Thus, any ill-conditioning or singular situations can automatically be avoided.

The Bayesian evidence procedure [23] can readily be used to "optimize" the regularization parameters. From the Bayesian viewpoint, the following error criterion is equivalent to the criterion (14):

$$J_{\mathrm{B}}(\mathbf{g}, \mathbf{h}, \beta) = \beta\mathbf{e}^{\mathrm{T}}\mathbf{e} + \sum_{i=1}^{n_{\mathrm{M}}} h_i g_i^2 = \beta\mathbf{e}^{\mathrm{T}}\mathbf{e} + \mathbf{g}^{\mathrm{T}}\mathbf{H}\mathbf{g}, \tag{19}$$

where β is the noise parameter (estimate of the inverse of noise variance), $\mathbf{h} = [h_1 \cdots h_{n_{\mathrm{M}}}]^{\mathrm{T}}$ is the hyperparameter vector, and $\mathbf{H} = \mathrm{diag}\{h_1, \ldots, h_{n_{\mathrm{M}}}\}$. The relationship between a regularization parameter and its corresponding hyperparameter is obviously given by

$$\lambda_i = \frac{h_i}{\beta}. \tag{20}$$

Following MacKay [23], it can be shown that the log evidence for $\mathbf{h}$ and β is

$$\begin{aligned}\log(\text{evidence}) = &\sum_{i=1}^{n_M} \frac{1}{2}\log(h_i) - \frac{n_M}{2}\log(\pi) - \frac{N}{2}\log(2\pi) + \frac{N}{2}\log(\beta) \\ &- \sum_{i=1}^{n_M}\frac{1}{2}h_i g_i^2 - \frac{1}{2}\beta \mathbf{e}^T\mathbf{e} - \frac{1}{2}\log(\det(\mathbf{B})) + \frac{n_M}{2}\log(2\pi).\end{aligned} \tag{21}$$

Because of the orthogonalization, the "Hessian" matrix $\mathbf{B}$ is diagonal and is given by

$$\mathbf{B} = \mathbf{H} + \beta\mathbf{W}^T\mathbf{W} = \text{diag}\{h_1 + \beta\mathbf{w}_1^T\mathbf{w}_1, \ldots, h_{n_M} + \beta\mathbf{w}_{n_M}^T\mathbf{w}_{n_M}\}. \tag{22}$$

Setting the derivatives of log(evidence) with respect to $\mathbf{h}$ and β to zeros yields the updating formulas for $\mathbf{h}$ and β, respectively. Substituting these updating formulas into (20) results in the updating formulas for the regularization parameters:

$$\lambda_i^{\text{new}} = \frac{\gamma_i^{\text{old}}}{N - \gamma^{\text{old}}}\frac{\mathbf{e}^T\mathbf{e}}{g_i^2}, \quad 1 \leqslant i \leqslant n_M, \tag{23}$$

where

$$\gamma_i = \frac{\mathbf{w}_i^T\mathbf{w}_i}{\lambda_i + \mathbf{w}_i^T\mathbf{w}_i} \tag{24}$$

and

$$\gamma = \sum_{i=1}^{n_M}\gamma_i. \tag{25}$$

As a special case, the single regularization parameter in the UROLS is updated using [10]

$$\lambda^{\text{new}} = \frac{\gamma^{\text{old}}}{N - \gamma^{\text{old}}}\frac{\mathbf{e}^T\mathbf{e}}{\mathbf{g}^T\mathbf{g}} \quad \text{with } \gamma = \sum_{i=1}^{n_M}\frac{\mathbf{w}_i^T\mathbf{w}_i}{\lambda + \mathbf{w}_i^T\mathbf{w}_i}. \tag{26}$$

The iterative model selection procedure can now be summarized:

Initialization. Set λ_i, $1 \leqslant i \leqslant n_M$ to the same small positive value (e.g. 0.001).

Step 1: Given the current $\boldsymbol{\lambda}$, use the procedure described in Appendix B to select a subset model with n_s terms.

Step 2: Update $\boldsymbol{\lambda}$ using (23)–(25) with $n_M = n_s$. If $\boldsymbol{\lambda}$ remains sufficiently unchanged in two successive iterations or a pre-set maximum iteration number is reached, stop; otherwise go to *Step* 1.

At the beginning of the iterative loop, the value of ξ for terminating subset model selection can deliberately be chosen to be smaller than really needed, so that Step 1 produces a n_s-term model which is larger than what is really needed. This ensures that no significant terms are lost when $\boldsymbol{\lambda}$ is far from its optimal value. When $\boldsymbol{\lambda}$ has converged (typically after 10 iterations), an appropriate value of ξ should then be used to produce a parsimonious final model.

It is worth pointing out, however, that the choice of ξ is less critical than the original OLS algorithm. In the original OLS selection procedure, when data is very noisy, it is possible that the normalized (training) mean square error (MSE) $1 - \sum[\mathrm{err}]_l$ continuously decreases as more terms are added, as these unnecessarily added regressors may simply fit into the noise in the training data. This may lead to over-fitting. Often a cross-validation using a separate testing data set is required to learn an appropriate value for ξ. For the LROLS algorithm, multiple regularizers enforce sparsity, and $1 - \sum[\mathrm{rerr}]_l$ will not continuously decreases as more terms are added. This is because those unnecessarily added terms will have very large λ_l associated with them, effectively forcing their weights to be zero. Thus, when to terminate model selection or how many regressors to include in the final model becomes much clearer. This will be illustrated in the simulation examples. It should also be pointed out that the above iterative procedure can generally find a local optimal $\boldsymbol{\lambda}$. As in [14], a genetic algorithm can be combined with the LROLS method to find a global optimal $\boldsymbol{\lambda}$. This is however achieved at the cost of a considerable increase in complexity, and therefore this strategy is not adopted here.

4. Comparison with the relevance vector machine

Adopting the equivalent regularization formula, the RVM for regression [26] involves an iterative loop of the model parameter estimation and regularization parameter updating. With given $\boldsymbol{\lambda}$, the model parameter estimate is the usual regularized least squares solution:

$$\boldsymbol{\theta} = \tilde{\mathbf{B}}^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{y}, \tag{27}$$

where the Hessian matrix $\tilde{\mathbf{B}}$ is given by

$$\tilde{\mathbf{B}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} + \boldsymbol{\Lambda}. \tag{28}$$

The regularization parameters are updated using

$$\lambda_i^{\mathrm{new}} = \frac{\tilde{\gamma}_i^{\mathrm{old}}}{N - \tilde{\gamma}^{\mathrm{old}}} \frac{\mathbf{e}^{\mathrm{T}}\mathbf{e}}{\theta_i^2}, \quad 1 \leqslant i \leqslant n_{\mathrm{M}}, \tag{29}$$

where

$$\tilde{\gamma} = \sum_{i=1}^{n_{\mathrm{M}}} \tilde{\gamma}_i \quad \text{with } \tilde{\gamma}_i = 1 - \lambda_i \bar{b}_{i,i} \tag{30}$$

and $\bar{b}_{i,i}$ denotes the ith diagonal element of $\tilde{\mathbf{B}}^{-1}$.

It is clear that the RVM starts with the full model set $\boldsymbol{\Phi}$ and removes those model regressors that have large values in their associated regularization parameters. In other words, it is based on the backward elimination principle. This is in contrast to the forward selection principle adopted by the LROLS algorithm. It is well known and obvious that forward selection is computationally more attractive compared with backward elimination. More importantly, in the LROLS algorithm, only a subset matrix of the Hessian matrix $\beta^{-1}\mathbf{B}$ (see (22)) is used in updating the regularization parameters. This subset Hessian matrix is diagonal and well-conditioned with a small eigenvalue spread. Therefore, the inverse of the Hessian is trivial, the regularization parameter updating is exact and simple, and the iterative procedure converges fast. For the RVM, the Hessian matrix may be ill-conditioned or even singular, the regularization parameter updating is much more expensive, and the iterative procedure generally converges with slower rate and may suffer from numerical instability.

To summarize, the generalization capabilities and levels of sparsity produced for both the RVM and LROLS are expected to be similar, since they both use an approach of multiple regularizers to enforce sparsity and adopts a similar evidence procedure for updating regularization parameters. However, the RLOLS algorithm has considerable computational advantages, it can operate robustly in difficult modeling conditions, and its iterative loop generally converges faster compared with the RVM. The RVM may suffer from numerical instability or even fails to work if the conditional number or eigenvalue spread of the underlying modeling problem is extremely large.

5. Simulation results

Three examples were used in simulation to illustrate the LROLS algorithm discussed in Section 3 and to compare its performance with the OLS and UROLS algorithms, and the RVM method.

Example 1. This was the simple example of modeling the scalar function

$$f(x) = \sin(2\pi x), \quad 0 \leqslant x \leqslant 1, \tag{31}$$

by a Gaussian radial basis function (RBF) network. The Gaussian kernel function used had a variance of 0.04. One hundred training data were generated from $y = f(x) + \varepsilon$, where x was taken from the uniform distribution in $(0, 1)$ and the noise ε had a Gaussian distribution with zero mean and variance 0.16. The noisy training points y and the underlying function $f(x)$ are plotted in Fig. 1. As each training data x was considered as a candidate RBF center, there were $n_{\mathrm{M}} = 100$ regressors in the model (2). Note that the training data were very noisy, and this learning problem was ill-conditioned as x was drawn randomly from $(0, 1)$. For this simple example, many sets of different noisy training data were generated, and the modeling results were consistent and similar to the results shown below, which were typical.

It is informative to examine the selection process of the OLS algorithm, listed in Table 1. Notice that the normalized MSE continuously decreased as more terms were added. The procedure stopped at the 16th stage, when it detected that adding

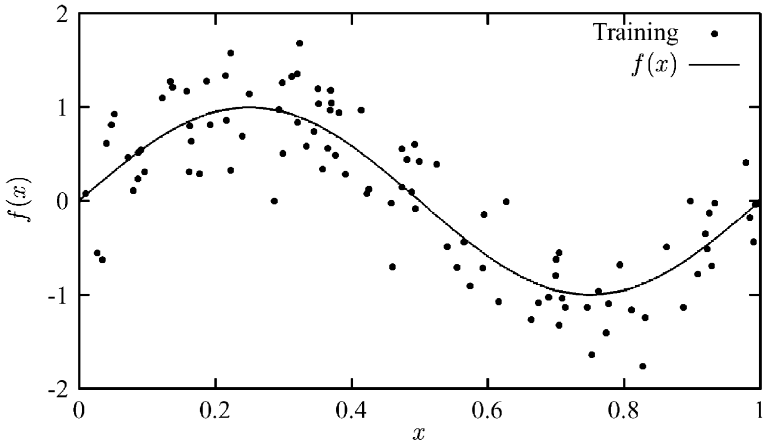

Fig. 1. A typical set of noisy training data y (dots) and underlying function $f(x)$ (curve) for the simple scalar function modeling problem.

Table 1
OLS selection procedure for the simple scalar function modeling problem

Stage l	Accuracy $1-\sum[\mathrm{err}]_l$	Weight θ_l
1	0.6461718264	2.60935e + 06
2	0.2840641827	−2.28370e + 06
3	0.2416057207	−1.29831e + 08
4	0.2260673781	−2.21722e + 09
5	0.2189319619	3.63027e + 08
6	0.2179112365	1.66438e + 09
7	0.2169210404	−3.19282e + 09
8	0.2156145110	1.70011e + 09
9	0.2135190658	4.06932e + 09
10	0.2113153903	−1.94658e + 09
11	0.2108713704	−2.72236e + 08
12	0.2095033180	−4.28658e + 07
13	0.2093349973	5.60372e + 06
14	0.2091282455	−1.59224e + 06
15	0.2068241235	3.83400e + 05

Stop due to no term selected at 16 stage
MSE over noisy training set: 0.147430

one more term would cause the problem to be singular or very ill-conditioned. This produced a 15-term model. The model weights had very large value, as can be seen in Table 1. This was a typical sign of over-fitting. The MSE over the training set was smaller than the noise variance, indicating that the model was fitted into the noise. Over-fitting can also be seen clearly by the model map given in Fig. 2. Notice that

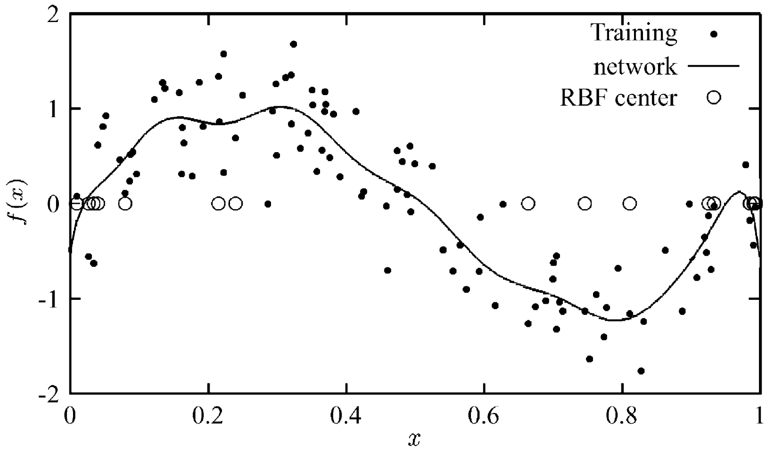

Fig. 2. Model mapping (curve) produced by the OLS algorithm for the simple scalar function modeling problem. Dots indicate noisy training data y and circles the RBF centers.

Table 2
UROLS selection procedure for the simple scalar function modeling problem after λ has converged (10 iterations)

Stage l	Accuracy $1-\sum[\text{rerr}]_l$	Weight θ_l
1	0.6490143575	1.62388e + 00
2	0.2908595802	−2.28935e + 00
3	0.2508542689	−8.48791e − 01
4	0.2361130705	8.22056e − 01
5	0.2322792890	1.03731e + 00
6	0.2312755537	−3.73154e − 01
7	0.2312749762	3.01529e − 02
8	0.2312737869	−1.51268e − 02
9	0.2312736479	−5.40054e − 03
10	0.2312736475	3.76698e − 04
11	0.2312736474	9.55162e − 05
12	0.2312736474	−1.27653e − 05
13	0.2312736474	−2.25256e − 07

Stop due to no term selected at 14 stage
MSE over noisy training set: 0.156678
Regularization parameter λ: 3.09037e − 01

even smaller models were used, say 12-term or 10-term ones, similar over-fitted results were produced.

For the UROLS algorithm, it was seen that 10 iterations was sufficient to ensure the convergence of the iterative procedure. The model selection procedure, after the single λ had converged, is listed in Table 2. The selection stopped at the 14th stage, as there was no more candidate which would not cause an ill-conditioning or singular problem. The modeling accuracy $1 - \sum[\text{rerr}]_l$ remained unchanged after the 11th stage. Examining the weight of the 13th regressor, which was effectively zero. This indicated a 12-term model. The model map produced by this 12-term model is depicted in Fig. 3, where it is clearly seen that over-fitting did not occur. From

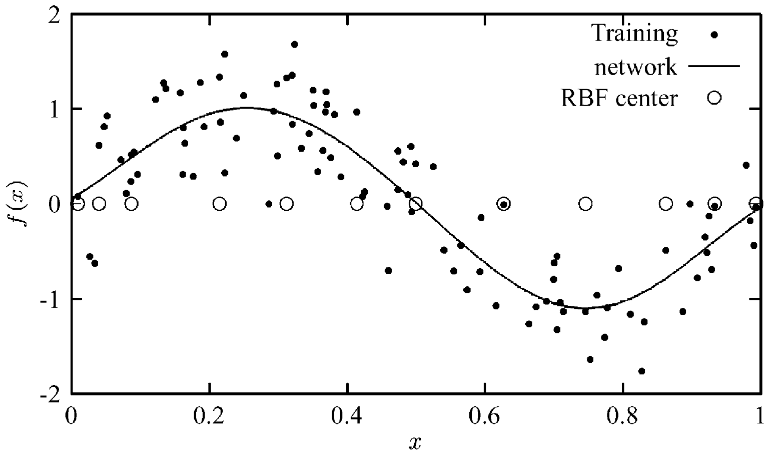

Fig. 3. Model mapping (curve) produced by the UROLS algorithm for the simple scalar function modeling problem. Dots indicate noisy training data y and circles the RBF centers.

Table 3
LROLS selection procedure for the simple scalar function modeling problem after λ has converged (10 iterations)

Stage l	Accuracy $1 - \sum[\text{rerr}]_l$	Weight θ_l	Regularizerλ_l
1	0.6485054202	1.87494e + 00	2.53227e − 01
2	0.2887313702	−1.70014e + 00	1.81540e − 01
3	0.2500895914	−1.00970e + 00	2.01490e − 01
4	0.2349327688	5.67310e − 01	8.64601e − 01
5	0.2336724743	4.17979e − 01	1.36357e + 00
6	0.2332827490	−1.51352e − 01	6.93984e − 01
7	0.2332827490	−9.49873e − 10	5.67623e + 07
8	0.2332827490	−2.79967e − 10	1.11770e + 08
9	0.2332827490	7.14157e − 11	1.03860e + 07
10	0.2332827490	−2.05313e − 12	1.92708e + 08
11	0.2332827490	−1.32386e − 13	7.85977e + 08
12	0.2332827490	2.29641e − 14	4.09979e + 08
13	0.2332827490	−2.53260e − 38	1.15132e + 32

Stop due to no term selected at 14 stage
MSE over noisy training set: 0.159167

Table 2, it can be seen that the 12th and 11th terms also had very small weights, indicating a 10-term model was also feasible. This 10-term model had an identical performance to that of the 12-term model. The generalization performance of the model produced by the UROLS algorithm was obviously very good.

The LROLS selection procedure, after $\boldsymbol{\lambda}$ had converged (10 iterations), is listed in Table 3. The selection stopped at the 14th stage, as there was no more candidate which would not cause an ill-conditioning or singular problem. The modeling accuracy $1 - \sum[\mathrm{rerr}]_l$ however remained unchanged after the 6th stage. The regularization parameters related with the 7–13th terms were all very large, and

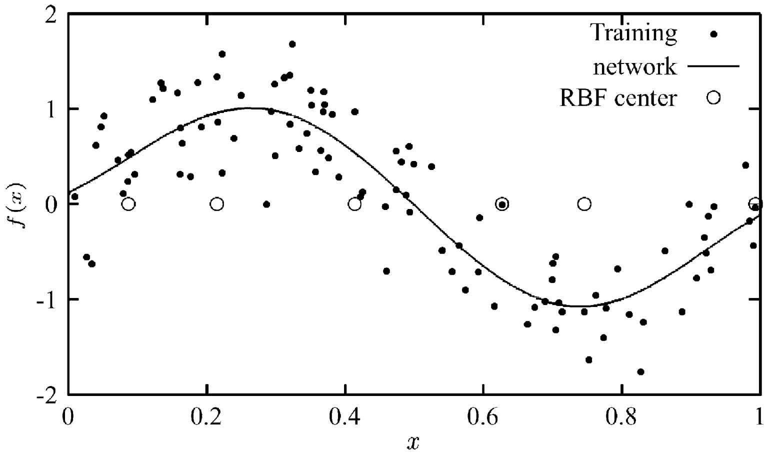

Fig. 4. Model mapping (curve) produced by the LROLS algorithm for the simple scalar function modeling problem. Dots indicate noisy training data y and circles the RBF centers.

Table 4
Model obtained by the RVM for the simple scalar function modeling problem after $\boldsymbol{\lambda}$ has converged (40 iterations)

Weight θ_l	Regularizer λ_l
1.27556e − 01	1.12245e + 00
1.66234e − 01	8.62460e − 01
7.68171e − 03	1.78695e + 01
−1.72753e − 01	6.15214e − 01
−1.57279e + 00	5.90294e − 02
5.29620e − 03	2.09115e + 01
5.26387e − 03	2.10376e + 01
−4.34350e − 01	7.69138e − 01
1.08845e + 00	1.21877e − 01
8.99832e − 01	1.63270e − 01

MSE over noisy training set: 0.157819

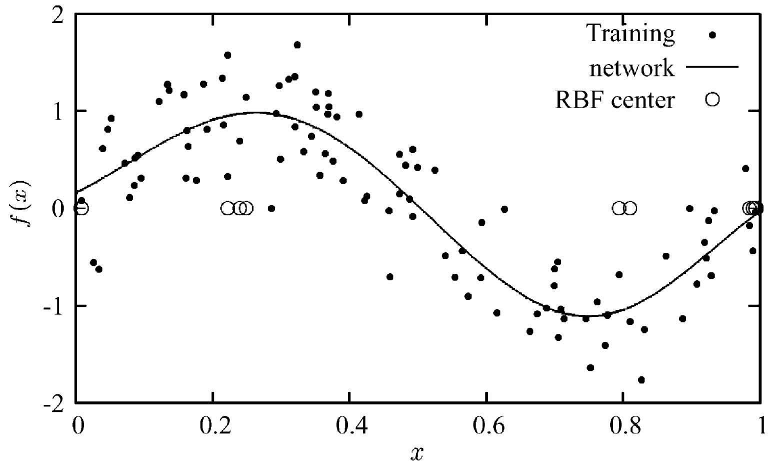

Fig. 5. Model mapping (curve) produced by the RVM algorithm for the simple scalar function modeling problem. Dots indicate noisy training data y and circles the RBF centers.

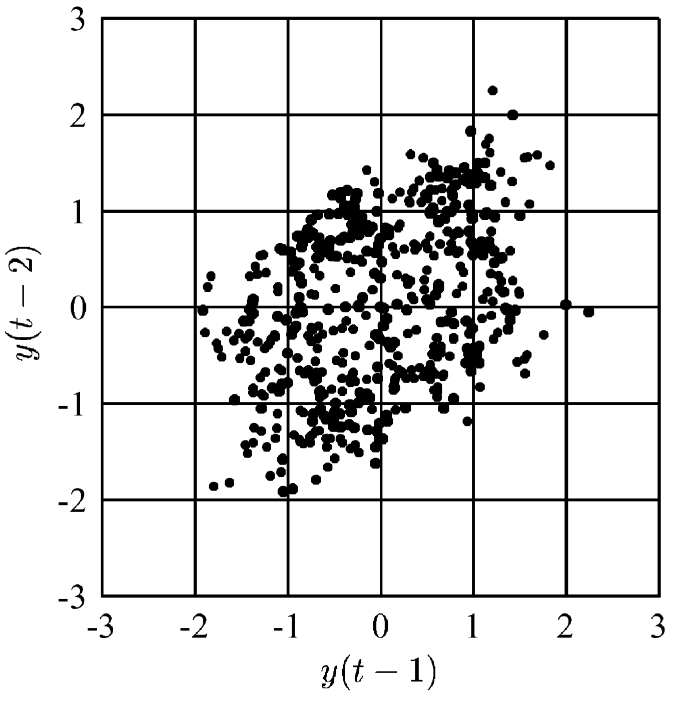

Fig. 6. Phase plot of a typical set of noisy training data ($y(0) = y(-1) = 0.0$) for the two-dimensional time series modeling problem.

the associated model weights were effectively zero. This clearly indicated a 6-term model. The model map produced by this 6-term model is depicted in Fig. 4, where it can be seen that the generalization performance of this 6-term model was similar to that of the 12-term model produced by the UROLS algorithm. The LROLS

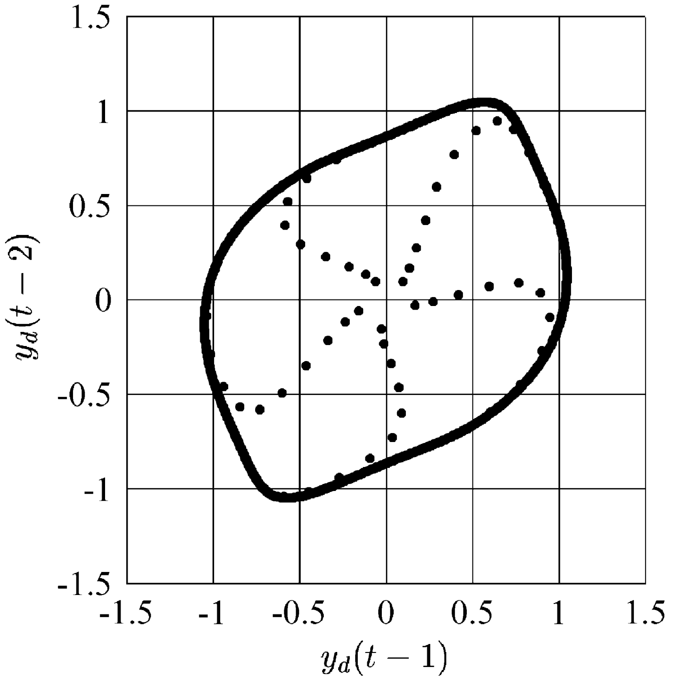

Fig. 7. Phase plot of the noise-free two-dimensional time series ($y_d(0) = y_d(-1) = 0.1$).

Table 5
OLS selection procedure for the two-dimensional time series problem based on cross-validation

Stage l	MSE over training set	MSE over testing set
34	0.088584	0.097726
35	0.088541	0.097367
36	0.088423	0.097092
37	0.088099	0.096657
38	0.088027	0.097092
39	0.087922	0.097411
40	0.087875	0.097136
41	0.087745	0.097570
42	0.087561	0.098307
43	0.087426	0.099624
44	0.087106	0.101361

algorithm had advantages of producing a sparser model with a clear-cut decision on how many terms to include in the final model.

For the RVM, the iterative loop was observed to converge slower compared with the LROLS algorithm. In fact, with 20 iterations, the resulting model contained 20 terms, and it took 40 iterations to produced the final sparse model of 10 terms, listed in Table 4. The model map generated by this 10-term model is shown in Fig. 5. As expected, the generalization capability of this constructed model is similar to those produced by the UROLS and LROLS algorithms.

Example 2. This was a two-dimensional simulated nonlinear time series given by

$$\begin{aligned} y(k) = {} & \big(0.8 - 0.5\exp(-y^2(k-1))\big)y(k-1) - \big(0.3 + 0.9\exp(-y^2(k-1))\big)y(k-2) \\ & + 0.1\sin(\pi y(k-1)) + \varepsilon(k), \qquad (32) \end{aligned}$$

where the noise $\varepsilon(k)$ was Gaussian with zero mean and variance 0.09. One thousand noisy samples were generated given $y(0) = y(-1) = 0.0$. The first 500 data points, plotted in Fig. 6, were used for training, and the other 500 samples were used for possible cross-validation. The underlying noise-free system

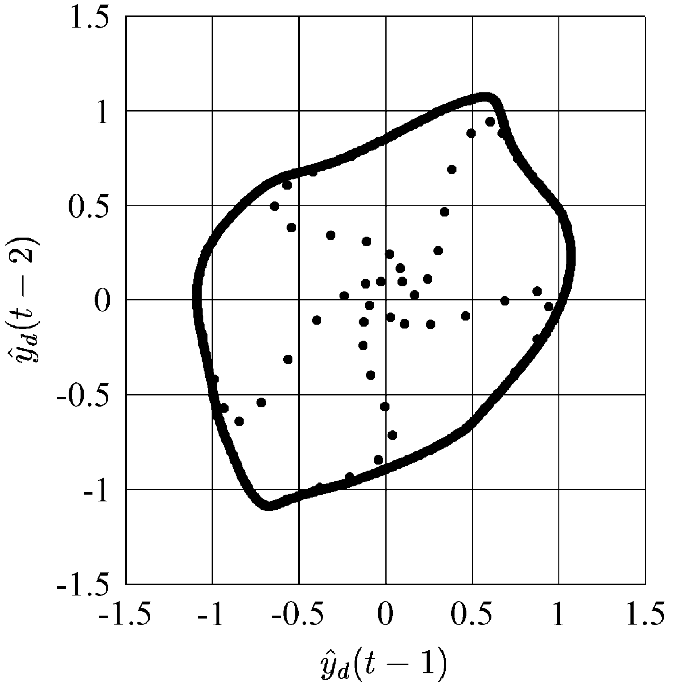

Fig. 8. Phase plot of the iterative RBF model output ($\hat{y}_d(0) = \hat{y}_d(-1) = 0.1$). The 37-term model was selected by the OLS algorithm.

$$
\begin{aligned}
y_{\rm d}(k) = & \big(0.8 - 0.5\exp(-y_{\rm d}^2(k-1))\big)y_{\rm d}(k-1) \\
& - \big(0.3 + 0.9\exp(-y_{\rm d}^2(k-1))\big)y_{\rm d}(k-2) \\
& + 0.1\sin(\pi y_{\rm d}(k-1))
\end{aligned}
\tag{33}
$$

was specified by a limit circle, as shown by the one thousand samples given in Fig. 7 with $y_{\rm d}(0) = y_{\rm d}(-1) = 0.1$. A Gaussian RBF model of the form

$$\hat{y}(k) = f_{RBF}(\mathbf{x}(k)), \tag{34}$$

with $\mathbf{x}(k) = [y(k-1)\ y(k-2)]^{\rm T}$, was constructed using the noisy training data. The Gaussian kernel function was chosen to have a variance of 0.81. As each data point $\mathbf{x}(k)$ was considered as a candidate RBF center, $n_{\rm M} = 500$.

When using the OLS algorithm to select a subset model, the MSE continuously decreased as more terms were added. The choice of ξ should ensure that the selection procedure is terminated at an appropriate model size. Basically, a trade-off between training performance and model complexity is required, and there is no simple uniform rule for deciding the desired value for ξ—it often has to be learnt by interacting with the selection procedure. An alternation termination criterion is based on the Akaike information criterion [22]

$$\mathrm{AIC}(\chi) = N\log(\mathbf{e}^{\rm T}\mathbf{e}/N) + n_{\rm s}\chi, \tag{35}$$

Table 6
UROLS selection procedure for the two-dimensional time series problem after λ has converged (10 iterations)

Stage l	Accuracy $1-\sum[\mathrm{rerr}]_l$	Weight θ_l
25	0.1248104407	$-2.75251\mathrm{e}-01$
26	0.1248096321	$-3.42929\mathrm{e}-01$
27	0.1248084895	$-6.96862\mathrm{e}-02$
28	0.1248074233	$7.82537\mathrm{e}-02$
29	0.1248067749	$1.26036\mathrm{e}-01$
30	0.1248061653	$1.45025\mathrm{e}-01$
31	0.1248057240	$8.11109\mathrm{e}-02$
32	0.1248051109	$-4.57533\mathrm{e}-02$
33	0.1248048539	$1.29904\mathrm{e}-01$
34	0.1248038870	$-8.58027\mathrm{e}-02$
35	0.1248037989	$2.16182\mathrm{e}-02$
36	0.1248037697	$1.23090\mathrm{e}-02$
37	0.1248037567	$1.01068\mathrm{e}-02$
38	0.1248037541	$7.58840\mathrm{e}-03$
39	0.1248037474	$1.18317\mathrm{e}-02$
40	0.1248037434	$6.19265\mathrm{e}-03$

MSE of 30-term model over training set: 0.091087
MSE of 30-term model over testing set: 0.095067
Regularization parameter λ: $8.56570\mathrm{e}-02$

where χ is the critical value of the chi-squared distribution with one degree of freedom and for a given level of significance. The regressors are selected by the OLS algorithm and the selection is terminated when the AIC reaches the minimum. With this approach, an appropriate value for χ has to be chosen. A practical strategy to avoid an oversized model is to have a separate validation data set at a cost of increasing complexity. The model selection is carried out on the training set, and the MSE of the selected model over the validation set is monitored. When this testing accuracy ceases to improve, the selection procedure is terminated. Using this strategy, the OLS model selection procedure is listed in Table 5, which indicated a 37-term model. The selected 37-term model was used to iteratively generate the time series according to

$$\hat{y}_{\mathrm{d}}(k) = f_{RBF}(\hat{\mathbf{x}}_{\mathrm{d}}(k)), \tag{36}$$

with $\hat{\mathbf{x}}_{\mathrm{d}}(k) = [\hat{y}_{\mathrm{d}}(k-1)\ \hat{y}_{\mathrm{d}}(k-2)]^{\mathrm{T}}$ and given the initial condition $\hat{y}_{\mathrm{d}}(0) = \hat{y}_{\mathrm{d}}(-1) = 0.1$. The resulting phase plot is shown in Fig. 8.

The UROLS selection procedure took 10 iterations to converge and the selection procedure, after the single λ had converged, is given in Table 6. Using the training set alone and without specifying an appropriate stopping threshold ξ, there was some difficulty to determine when to stop the selection but the situation was clearer than

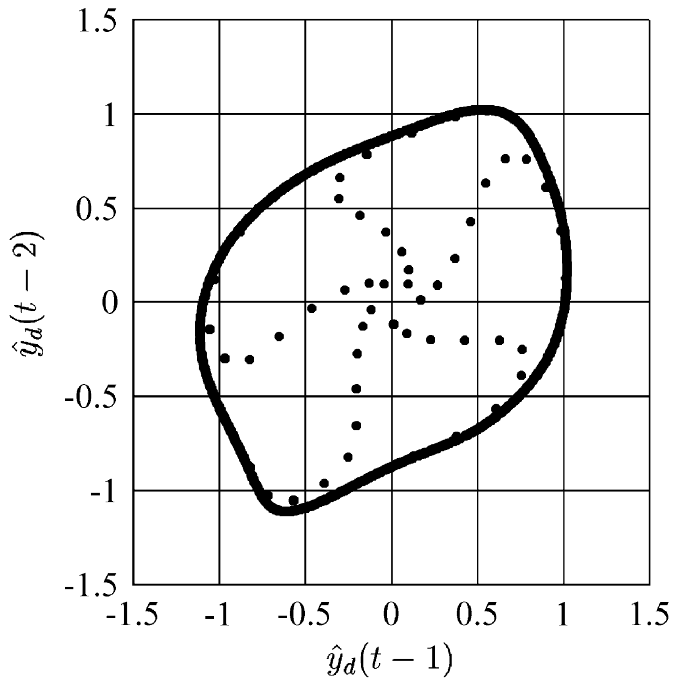

Fig. 9. Phase plot of the iterative RBF model output ($\hat{y}_{\mathrm{d}}(0) = \hat{y}_{\mathrm{d}}(-1) = 0.1$). The 30-term model was selected by the UROLS algorithm.

Table 7
LROLS selection procedure for the two-dimensional time series problem after $\boldsymbol{\lambda}$ has converged (10 iterations)

Stage l	Accuracy $1-\sum[\text{rerr}]_l$	Weight θ_l	Regularizer λ_l
14	0.1261656992	$-1.88490e+00$	$2.97139e+00$
15	0.1261217915	$2.91399e-01$	$1.44933e+00$
16	0.1261071047	$2.63144e+00$	$2.50988e-01$
17	0.1257874268	$1.84482e+00$	$1.95521e-02$
18	0.1256611544	$-9.92855e-02$	$7.22097e-01$
19	0.1256611424	$5.20550e-05$	$4.99250e+03$
20	0.1256611385	$1.37736e-04$	$1.88246e+02$
21	0.1256611372	$2.00067e-05$	$5.23736e+04$
22	0.1256611369	$-3.17587e-05$	$4.16648e+03$
23	0.1256611366	$1.76802e-05$	$8.33651e+02$
24	0.1256611364	$1.53437e-05$	$7.81282e+02$
25	0.1256611363	$-5.29943e-06$	$8.97688e+03$
26	0.1256611362	$-6.65540e-06$	$5.65923e+03$
27	0.1256611361	$-1.12777e-05$	$6.57092e+02$
28	0.1256611360	$-1.16153e-05$	$1.50267e+02$
29	0.1256611360	$4.51493e-07$	$6.74909e+04$
30	0.1256611360	$1.01617e-07$	$3.54793e+05$

MSE of 18-term model over training set: 0.092637
MSE of 18-term model over testing set: 0.096775

the case of the OLS selection. For example, after the 30th stage, the modeling accuracy $1-\sum[\text{rerr}]_l$ was hardly changing. Therefore, it was decided that the final model had 30 terms. The resulting phase plot of this 30-term model generated iteratively is depicted in Fig. 9. The LROLS selection procedure, after $\boldsymbol{\lambda}$ had converged (10 iterations), is listed in Table 7. Note that how many terms to include in the final model is a clear decision based only on the training set. As any terms added after the stage 18 all had very large regularization parameters and their weights were effectively zero, the selected model contained 18 terms. The MSE values of this 18-term model over the training and testing sets were only slightly worse than those of the 30-term model selected by the UROLS algorithm. The resulting phase plot generated iteratively by this 18-term model is shown in Fig. 10. It can be seen that it had a similar generalization performance as the model produced by the UROLS algorithm.

The result obtained using the RVM algorithm is almost identical to that produced by the LROLS algorithm, as can be seen clearly in the final model listed in Table 8 and the phase plot shown in Fig. 11. The only difference in this case appears to be that the RVM method is computationally more costly due to its backward elimination nature and the slower convergence of its iterative loop. Again many different sets of noisy training data were used, and the modeling results obtained were consistent with the typical results shown here.

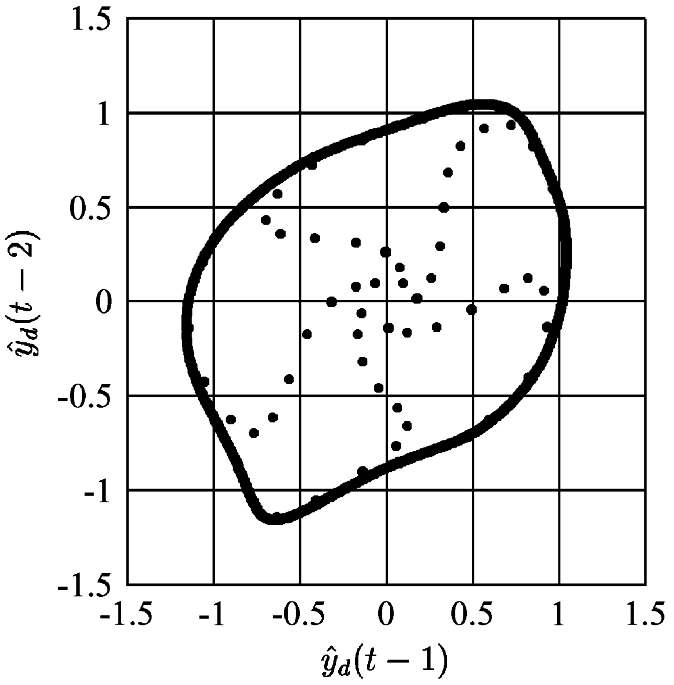

Fig. 10. Phase plot of the iterative RBF model output ($\hat{y}_d(0) = \hat{y}_d(-1) = 0.1$). The 18-term model was selected by the LROLS algorithm.

Example 3. This example constructed a model representing the relationship between the fuel rack position (input) and the engine speed (output) for a Leyland TL11 turbocharged, direct injection diesel engine operated at low engine speed. It is known that at low engine speed, the relationship between the input and output is nonlinear [3]. Detailed system description and experimental setup can be found in [3]. The data set, depicted in Fig. 12, contained 410 samples. The first 210 data points were used in modeling and the last 200 points in model validation. In the previous investigation [3], it was found that the appropriate system "input" vector was $\mathbf{x}(k) = [y(k-1)\ u(k-1)\ u(k-2)]^{\mathrm{T}}$ and, therefore, a RBF model of the form:

$$\hat{y}(k) = f_{RBF}(\mathbf{x}(k)) \tag{37}$$

was used to model the data. As each data vector $\mathbf{x}(k)$ was considered as a candidate RBF center, there were $n_{\mathrm{M}} = 210$ regressors in the regression model (2). The variance of the RBF kernel function was chosen to be 1.69. Note that a strong periodic component is presented in the data, as can be seen clearly from Fig. 12.

Using the selection strategy identical to that used in Example 2, the OLS algorithm selected a 60-term model, the UROLS constructed a 46-term model, and

Table 8
Model obtained by the RVM for the two-dimensional time series problem after $\boldsymbol{\lambda}$ has converged (40 iterations)

Weight θ_l	Regularizer λ_l
2.45654e − 02	3.59452e + 00
−2.40398e + 00	1.57590e − 02
−5.31978e − 02	1.48458e + 00
−1.56233e + 00	3.83902e − 02
1.25172e − 02	7.27529e + 00
1.76934e − 03	4.01853e + 01
3.01592e − 01	7.26365e − 01
8.22632e − 03	2.92744e + 01
−1.60659e + 00	3.58431e − 02
1.57424e + 00	3.48003e − 02
1.76799e + 00	2.81382e − 02
1.73747e + 00	3.07907e − 02
1.85297e − 01	1.28503e + 00
−9.23914e − 01	1.03374e − 01
2.16866e − 01	3.55010e − 01
−9.70271e − 01	8.24441e − 02
8.29455e − 01	9.85890e − 02
6.81251e − 04	9.18239e + 01
1.39631e + 00	4.70555e − 02

MSE over training set: 0.092194
MSE over testing set: 0.096757

the LROLS algorithm constructed a 34-term model. The MSE values over the training and testing sets for these three models are given in Table 9, where it can be seen that the model produced by the LROLS had the best generalization performance. The constructed RBF model was used to generate the one-step prediction $\hat{y}(k)$ of the system output according to (37). The iterative model output $\hat{y}_{\mathrm{d}}(k)$ was also produced using

$$\hat{y}_{\mathrm{d}}(k) = f_{RBF}(\hat{\mathbf{x}}_{\mathrm{d}}(k)) \tag{38}$$

with $\hat{\mathbf{x}}_{\mathrm{d}}(k) = [\hat{y}_{\mathrm{d}}(k-1)\ u(k-1)\ u(k-2)]^{\mathrm{T}}$. The one-step model prediction and iterative model output for the 34-term model selected by the LROLS algorithm are shown in Fig. 13, in comparison with the system output.

For this example, the RVM algorithm as implemented in the form given in Section 4 failed to work due to numerical instability of the iterative loop for updating regularization parameters. Various initial values for $\boldsymbol{\lambda}$ were tried and a more stable updating formula for $\boldsymbol{\lambda}$

$$\lambda_i^{\mathrm{new}} = (1-\eta)\lambda_i^{\mathrm{old}} + \eta\,\frac{\tilde{\gamma}_i^{\mathrm{old}}}{N - \tilde{\gamma}^{\mathrm{old}}}\,\frac{\mathbf{e}^{\mathrm{T}}\mathbf{e}}{\theta_i^2}, \quad 1 \leqslant i \leqslant n_{\mathrm{M}}, \tag{39}$$

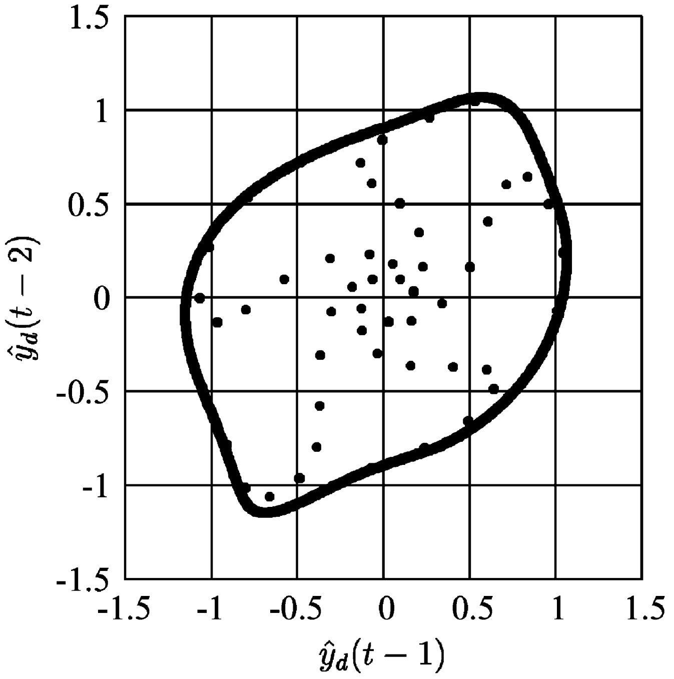

Fig. 11. Phase plot of the iterative RBF model output ($\hat{y}_d(0) = \hat{y}_d(-1) = 0.1$). The 19-term model was constructed by the RVM algorithm.

was also used, but the iterative loop for updating λ was unstable. This numerical instability caused the algorithm to force every regularization parameters to take very large values, which was the root of failure. It is conceivable that the RVM method implemented with some other more robust form may still work well in this situation. However, the results shown here serve to highlight a potentially inherent instability of the RVM method, which can affect the algorithm's performance in adverse modeling environments.

6. Conclusions

A locally regularized OLS algorithm has been developed for constructing parsimonious regression models. The proposed algorithm combines both the advantages of OLS forward model selection, which has ability to select only those significant regressors to explain training data, and local regularization, which enforces sparsity of models. Thus this LROLS algorithm is capable of producing very sparse regression models that generalize well. A further advantage of this algorithm is that when to terminate the model selection procedure can be made easily

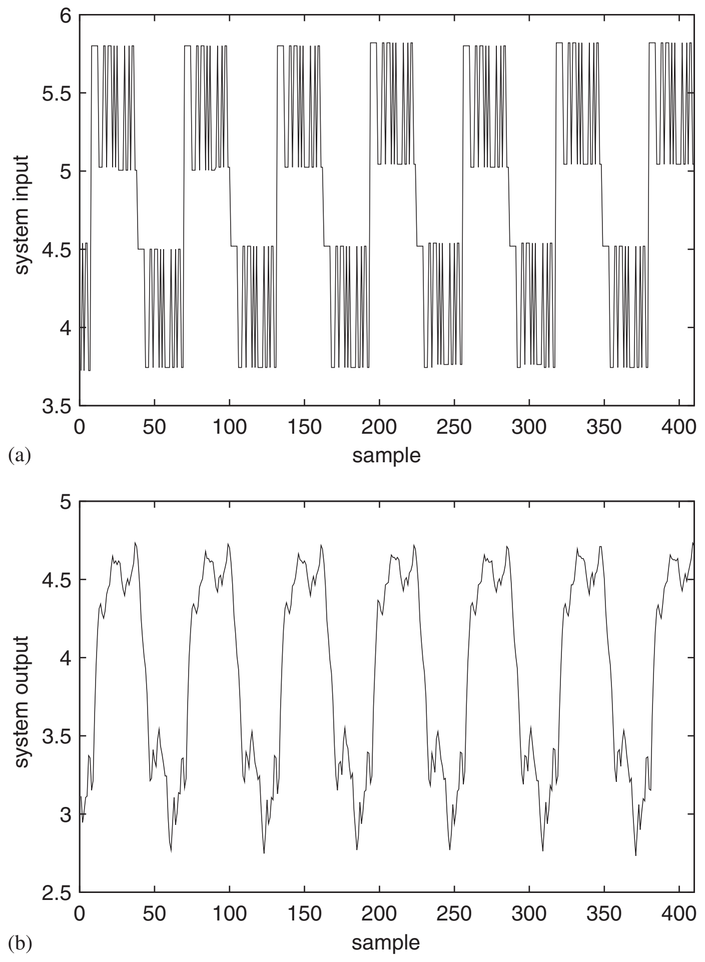

Fig. 12. Engine data set (a) input $u(k)$ and (b) output $y(k)$.

Table 9
Comparison of modeling accuracy for the engine data example

Model	MSE for training	MSE for testing
60-term (OLS)	0.000336	0.000872
46-term (UROLS)	0.000427	0.000532
34-term (LROLS)	0.000435	0.000487

based only on the training data, thus avoiding the need of costly cross-validation using a separate testing data set. As regularization is introduced in the orthogonal weight space, computational requirements of the iterative model selection procedure

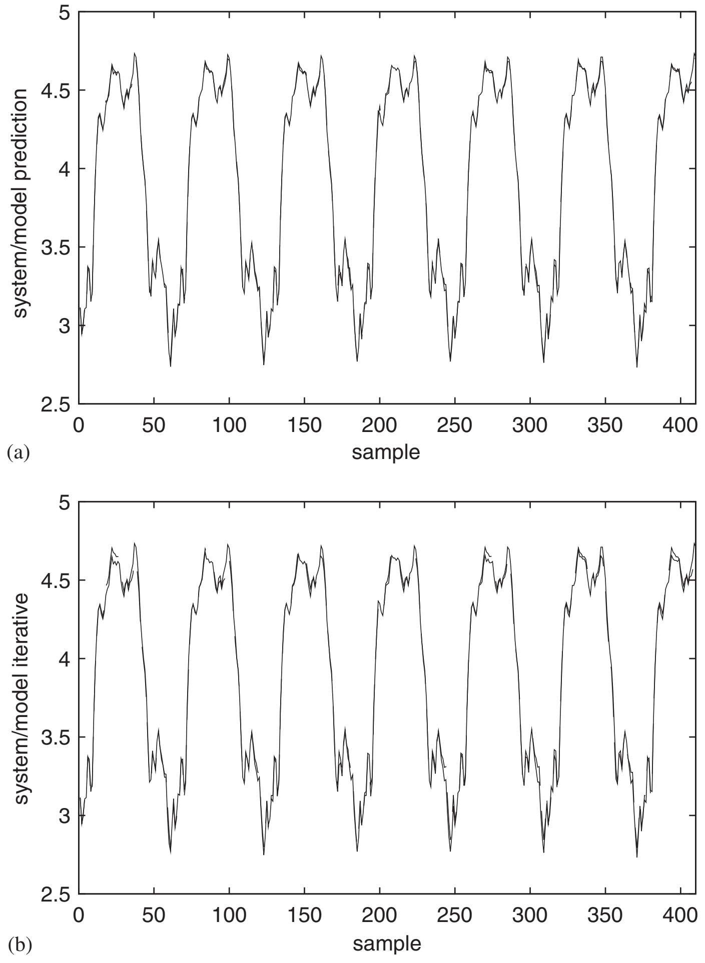

Fig. 13. System output $y(k)$ (solid) of the engine data set superimposed on (a) model one-step prediction $\hat{y}(k)$ (dashed) and (b) model iterative output $\hat{y}_d(k)$ (dashed). The model was selected by the LROLS.

are simple and straightforward. Any numerical ill-conditioning problems can automatically be avoided.

A comparison has been given with a state-of-the-art sparse modeling method known as the RVM. The two algorithms share many common features, and they both adopt a same approach of using multiple regularizers to enforce sparsity and use a similar evidence procedure to update regularization parameters or hyperparameters. It can be seen that the both methods possess similar generalization capabilities and degrees of sparsity. However, the proposed LROLS algorithm has clearly computational advantages, is numerically stable and robust, and is capable of performing well in adverse modeling environments.

Appendix A

The least squares solution for $\mathbf{g}$ is obtained by setting $\partial J_{\mathrm{R}}/\partial \mathbf{g} = \mathbf{0}$, that is,

$$\mathbf{W}^{\mathrm{T}}\mathbf{y} = (\mathbf{W}^{\mathrm{T}}\mathbf{W} + \mathbf{\Lambda})\mathbf{g}. \tag{40}$$

Now

$$\begin{aligned}\mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} &= (\mathbf{W}\mathbf{g} + \mathbf{e})^{\mathrm{T}}(\mathbf{W}\mathbf{g} + \mathbf{e}) - 2\mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} \\ &= \mathbf{g}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{W}\mathbf{g} + \mathbf{e}^{\mathrm{T}}\mathbf{e} + \mathbf{g}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{e} + \mathbf{e}^{\mathrm{T}}\mathbf{W}\mathbf{g} - 2\mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g}.\end{aligned} \tag{41}$$

Noting (40),

$$\mathbf{g}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{e} - \mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} = \mathbf{g}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}(\mathbf{y} - \mathbf{W}\mathbf{g}) - \mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} = \mathbf{g}^{\mathrm{T}}(\mathbf{W}^{\mathrm{T}}\mathbf{y} - \mathbf{W}^{\mathrm{T}}\mathbf{W}\mathbf{g} - \mathbf{\Lambda}\mathbf{g}) = \mathbf{0}. \tag{42}$$

Similarly,

$$\mathbf{e}^{\mathrm{T}}\mathbf{W}\mathbf{g} - \mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} = \mathbf{0}. \tag{43}$$

Thus

$$\mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} = \mathbf{g}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{W}\mathbf{g} + \mathbf{e}^{\mathrm{T}}\mathbf{e} \tag{44}$$

or

$$\mathbf{e}^{\mathrm{T}}\mathbf{e} + \mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g} = \mathbf{y}^{\mathrm{T}}\mathbf{y} - \mathbf{g}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{W}\mathbf{g} - \mathbf{g}^{\mathrm{T}}\mathbf{\Lambda}\mathbf{g}. \tag{45}$$

Appendix B

The modified Gram–Schmidt orthogonalization procedure calculates the $\mathbf{A}$ matrix row by row and orthogonalizes $\mathbf{\Phi}$ as follows: at the lth stage make the columns $\mathbf{\Phi}_j$, $l+1 \leqslant j \leqslant n_{\mathrm{M}}$, orthogonal to the lth column and repeat the operation for $1 \leqslant l \leqslant n_{\mathrm{M}} - 1$. Specifically, denoting $\mathbf{\Phi}_j^{(0)} = \mathbf{\Phi}_j$, $1 \leqslant j \leqslant n_{\mathrm{M}}$, then

$$\left.\begin{aligned} &\mathbf{w}_l = \mathbf{\Phi}_l^{(l-1)}, \\ &a_{l,j} = \mathbf{w}_l^{\mathrm{T}}\mathbf{\Phi}_j^{(l-1)}/(\mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l),\ l+1 \leqslant j \leqslant n_{\mathrm{M}}, \\ &\mathbf{\Phi}_j^{(l)} = \mathbf{\Phi}_j^{(l-1)} - a_{l,j}\mathbf{w}_l,\ l+1 \leqslant j \leqslant n_{\mathrm{M}}, \end{aligned}\right\} \quad l = 1, 2, \ldots, n_{\mathrm{M}} - 1. \tag{46}$$

The last stage of the procedure is simply $\mathbf{w}_{n_{\mathrm{M}}} = \mathbf{\Phi}_{n_{\mathrm{M}}}^{(n_{\mathrm{M}}-1)}$. The elements of $\mathbf{g}$ are computed by transforming $\mathbf{y}^{(0)} = \mathbf{y}$ in a similar way:

$$\left.\begin{aligned} &g_l = \mathbf{w}_l^{\mathrm{T}}\mathbf{y}^{(l-1)}/(\mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l + \lambda_l), \\ &\mathbf{y}^{(l)} = \mathbf{y}^{(l-1)} - g_l\mathbf{w}_l, \end{aligned}\right\} \quad 1 \leqslant l \leqslant n_{\mathrm{M}}. \tag{47}$$

This orthogonalization scheme can be used to derive a simple and efficient algorithm for selecting subset models in a forward-regression manner. First define

$$\mathbf{\Phi}^{(l-1)} = [\mathbf{w}_1 \cdots \mathbf{w}_{l-1}\mathbf{\Phi}_l^{(l-1)} \cdots \mathbf{\Phi}_{n_{\mathrm{M}}}^{(l-1)}]. \tag{48}$$

If some of the columns $\mathbf{\Phi}_l^{(l-1)}, \ldots, \mathbf{\Phi}_{n_{\mathrm{M}}}^{(l-1)}$ in $\mathbf{\Phi}^{(l-1)}$ have been interchanged, this will still be referred to as $\mathbf{\Phi}^{(l-1)}$ for notational convenience. Let a very small positive number T_z be given, which specifies the zero threshold and is used to automatically avoiding any ill-conditioning or singular problem. The lth stage of the selection procedure is given as follows.

Step 1: For $l \leqslant j \leqslant n_{\mathrm{M}}$:

Test—Conditioning number check. If $(\mathbf{\Phi}_j^{(l-1)})^{\mathrm{T}}\mathbf{\Phi}_j^{(l-1)} < T_z$, the jth candidate is not considered.

Compute

$$g_l^{(j)} = \left(\mathbf{\Phi}_j^{(l-1)}\right)^{\mathrm{T}} \mathbf{y}^{(l-1)} / \left((\mathbf{\Phi}_j^{(l-1)})^{\mathrm{T}}\mathbf{\Phi}_j^{(l-1)} + \lambda_j\right),$$
$$[\mathrm{rerr}]_l^{(j)} = (g_l^{(j)})^2 \left((\mathbf{\Phi}_j^{(l-1)})^{\mathrm{T}}\mathbf{\Phi}_j^{(l-1)} + \lambda_j\right) / (\mathbf{y}^{\mathrm{T}}\mathbf{y}).$$

Step 2: Find

$$[\mathrm{rerr}]_l = [\mathrm{rerr}]_l^{j_l} = \max\{[\mathrm{rerr}]_l^{(j)},\ l \leqslant j \leqslant n_{\mathrm{M}} \text{ and } j \text{ passes Test}\}.$$

Then the j_lth column of $\mathbf{\Phi}^{(l-1)}$ is interchanged with the lth column of $\mathbf{\Phi}^{(l-1)}$, the j_lth column of $\mathbf{A}$ is interchanged up to the $(l-1)$th row with the lth column of $\mathbf{A}$, and the j_lth element of $\boldsymbol{\lambda}$ is interchanged with the lth element of $\boldsymbol{\lambda}$. This effectively selects the j_lth candidate as the lth regressor in the subset model.

Step 3: Perform the orthogonalization as indicated in (46) to derive the lth row of $\mathbf{A}$ and to transform $\mathbf{\Phi}^{(l-1)}$ into $\mathbf{\Phi}^{(l)}$. Calculate g_l and update $\mathbf{y}^{(l-1)}$ into $\mathbf{y}^{(l)}$ in the way shown in (47).

The selection is terminated at the n_{s} stage when the criterion (18) is satisfied and this produces a subset model containing n_{s} significant regressors. The algorithm described here is in its standard form. A fast implementation can be adopted, as shown in [13], to reduce complexity.

References

[1] K.P. Bennett, O.L. Mangasarian, Robust linear programming discrimination of two linearly inseparable sets, Optim. Methods Software 1 (1992) 22–34.

[2] S.A. Billings, S. Chen, Extended model set, global data and threshold model identification of severely non-linear systems, Internat. J. Control 50 (5) (1989) 1897–1923.

[3] S.A. Billings, S. Chen, R.J. Backhouse, The identification of linear and non-linear models of a turbocharged automotive diesel engine, Mech. Systems Signal Process. 3 (2) (1989) 123–142.

[4] C.M. Bishop, Improving the generalisation properties of radial basis function neural networks, Neural Comput. 3 (4) (1991) 579–588.

[5] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, UK, 1995.

[6] M. Brown, C.J. Harris, Neurofuzzy Adaptive Modeling and Control, Prentice-Hall, Englewood Cliffs, NJ, 1994.

[7] S. Chen, Basis pursuit, Ph.D. Thesis, Department of Statistics, Stanford University, 1995.

[8] S. Chen, S.A. Billings, Representation of non-linear systems: the NARMAX model, Internat. J. Control 49 (3) (1989) 1013–1032.

[9] S. Chen, S.A. Billings, W. Luo, Orthogonal least squares methods and their application to non-linear system identification, Internat. J. Control 50 (5) (1989) 1873–1896.

[10] S. Chen, E.S. Chng, K. Alkadhimi, Regularised orthogonal least squares algorithm for constructing radial basis function networks, Internat. J. Control 64 (5) (1996) 829–837.
[11] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for radial basis function networks, IEEE Trans. Neural Networks 2 (2) (1991) 302–309.
[12] S. Chen, A.K. Samingan, L. Hanzo, Support vector machine multiuser receiver for DS-CDMA signals in multipath channels, IEEE Trans. Neural Networks 12 (3) (2001) 604–611.
[13] S. Chen, J. Wigger, Fast orthogonal least squares algorithm for efficient subset model selection, IEEE Trans. Signal Process. 43 (7) (1995) 1713–1715.
[14] S. Chen, Y. Wu, B.L. Luk, Combined genetic algorithm optimisation and regularised orthogonal least squares learning for radial basis function networks, IEEE Trans. Neural Networks 10 (5) (1999) 1239–1243.
[15] P.M.L. Drezet, R.F. Harrison, Support vector machines for system identification, in: Proceedings of the UKACC International Conference on Control'98, Swansea, UK, September 1–4, 1998, pp. 688–692.
[16] J.H. Friedman, Multivariate adaptive regression splines, Ann. Statist. 19 (1) (1991) 1–141.
[17] G.H. Golub, C.F. Van Loan, Matrix Computations, John Hopkins University Press, Baltimore, MD, 1989.
[18] T. Graepel, R. Herbrich, B. Schölkopf, A. Smola, P. Bartlett, K.R. Müller, K. Obermayer, R. Williamson, Classification on proximity data with LP-machines, in: Proceedings of ICANN-99, 1999, pp. 304–309.
[19] A.E. Hoerl, R.W. Kennard, Ridge regression: biased estimation for non-orthogonal problems, Technometrics 12 (1970) 55–67.
[20] T. Kavli, ASMOD: an algorithm for adaptive spline modeling of observation data, Internat. J. Control 58 (4) (1993) 947–968.
[21] K.L. Lee, S.A. Billings, Time series prediction using support vector machines, the orthogonal and the regularized orthogonal least-squares algorithms, Internat. J. Systems Sci. 33 (10) (2002) 811–821.
[22] I.J. Leontaritis, S.A. Billings, Model selection and validation methods for non-linear systems, Internat. J. Control 45 (1) (1987) 311–341.
[23] D.J.C. MacKay, Bayesian interpolation, Neural Comput. 4 (3) (1992) 415–447.
[24] D.J.C. MacKay, The evidence framework applied to classification networks, Neural Comput. 4 (1992) 720–736.
[25] M.J.L. Orr, Local smoothing of radial basis function networks, in: Proceedings of the International Symposium on Artificial Neural Networks, Hsinchu, Taiwan, 1995.
[26] M.E. Tipping, Sparse Bayesian learning and the relevance vector machine, J. Mach. Learning Res. 1 (2001) 211–244.
[27] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.

Sheng Chen received the B.Eng. degree in control engineering from the East China Petroleum Institute, Dongying, China, in 1982 and the Ph.D. degree in control engineering from the City University, London, UK, in 1986.

He joined the School of Electronics and Computer Science, University of Southampton, Southampton, UK, in September 1999. He previously held research and academic appointments at the University of Sheffield, Sheffield, UK, the University of Edinburgh, Edinburgh, UK, and University of Portsmouth, Portsmouth, UK. His recent research works include adaptive nonlinear signal processing, wireless communications, modeling and identification of nonlinear systems, neural networks and machine learning, finite-precision digital controller design, evolutionary computation methods, and optimization. He has published over 240 research papers.