Regularised OLS Algorithm with Fast Implementation for Training Multi-Output Radial Basis Function Networks

S.Chen
University of Portsmouth, UK

Abstract

The paper presents an approach for training multi-output radial basis function (RBF) networks by combining subset selection with regularisation. A regularised orthogonal least squares (ROLS) algorithm is derived, which is capable of constructing parsimonious networks that generalise well. A fast implementation of the ROLS algorithm further reduces computational requirements significantly. System identification is used as an example to demonstrate the effectiveness of this training algorithm.

1 Introduction

In previous works [1],[2], we have presented an orthogonal least squares (OLS) algorithm for constructing parsimonious RBF networks based on an efficient forward subset selection approach. When training a large neural network, regularisation is often necessary in order to overcome overfitting problem [3],[4]. Combining the parsimonious principle with regularisation techniques is more attractive since the resulting learning algorithms are able to choose small networks that generalise well.

Two examples of adopting this combined approach are a first-order regularised step-wise selection of subset regression models [5] and a zero-order regularised forward selection for RBF networks [6]. A disadvantage of these regularised subset selection algorithms is that they require considerably more computation than the OLS algorithm. A recent study [7], however, has overcome this difficulty and derived a ROLS algorithm which requires the same amount of computation as the OLS algorithm for subset model selection. The present study extends this ROLS algorithm to multi-output RBF networks by adopting the same technique used in [2].

Although the OLS algorithm is a very efficient subset model selection scheme, its computational efficiency can further be improved by using a fast implementation version [8]. This fast implemetation is equally applicable to the ROLS algorithm and, when used for learning multi-output RBF networks, reduction in computational complexity is even more significant.

2 The ROLS algorithm

The task of network learning can be formulated as the following regression model [2]

$$\mathbf{D} = \mathbf{\Phi}\Theta + \mathbf{E} \tag{1}$$

Here

$$\mathbf{D} = [\mathbf{d}_1 \cdots \mathbf{d}_{n_o}] = \begin{bmatrix} d_1(1) & \cdots & d_{n_o}(1) \\ \vdots & \vdots & \vdots \\ d_1(N) & \cdots & d_{n_o}(N) \end{bmatrix} \tag{2}$$

is the desired output matrix, n_o is the number of network outputs and N is the number of training data;

$$\mathbf{E} = [\mathbf{e}_1 \cdots \mathbf{e}_{n_o}] \tag{3}$$

is the modelling error matrix;

$$\mathbf{\Phi} = [\Phi_1 \cdots \Phi_M] = \begin{bmatrix} \phi_1(1) & \cdots & \phi_M(1) \\ \vdots & \vdots & \vdots \\ \phi_1(N) & \cdots & \phi_M(N) \end{bmatrix} \tag{4}$$

is the response matrix of the hidden layer or regression matrix and, when every training inputs are used as centres, $M = N$;

$$\Theta = \begin{bmatrix} \theta_{1,1} & \cdots & \theta_{1,n_o} \\ \vdots & \vdots & \vdots \\ \theta_{M,1} & \cdots & \theta_{M,n_o} \end{bmatrix} \tag{5}$$

'Artificial Neural Networks', 26–28 June 1995, Conference Publication No. 409, © IEE, 1995

is the network weight matrix.

Let an orthogonal decomposition of $\mathbf{\Phi}$ be $\mathbf{\Phi} = \mathbf{WA}$. The model (1) can be rewritten as

$$\mathbf{D} = \mathbf{WG} + \mathbf{E} \tag{6}$$

where

$$\mathbf{G} = \begin{bmatrix} g_{1,1} & \cdots & g_{1,n_o} \\ \vdots & \vdots & \vdots \\ g_{M,1} & \cdots & g_{M,n_o} \end{bmatrix} = \mathbf{A}\Theta \tag{7}$$

Introduce the following zero-order regularised error criterion

$$J_R = \text{trace}[\mathbf{E}^T\mathbf{E} + \lambda\mathbf{G}^T\mathbf{G}] \tag{8}$$

where $\lambda \geq 0$ is a regularisation parameter. It can be shown that

$$\text{trace}[\mathbf{E}^T\mathbf{E} + \lambda\mathbf{G}^T\mathbf{G}] = \text{trace}[\mathbf{D}^T\mathbf{D}] - \sum_{j=1}^{M}\left(\sum_{i=1}^{n_o} g_{j,i}^2\right)(\mathbf{w}_j^T\mathbf{w}_j + \lambda) \tag{9}$$

where $\mathbf{w}_j$ are columns of $\mathbf{W}$. Similar to the case of the OLS algorithm [2], we can define the regularised error reduction ratio due to $\mathbf{w}_k$ as

$$[rerr]_k = \left(\sum_{i=1}^{n_o} g_{k,i}^2\right)(\mathbf{w}_k^T\mathbf{w}_k + \lambda)/\text{trace}[\mathbf{D}^T\mathbf{D}] \tag{10}$$

Based on this ratio, significant regressors can be selected in a forward-regression procedure exactly as in the case of the OLS algorithm [2].

2.1 Fast version

Significant saving in computation can be achieved by adopting a fast implementation of the OLS algorithm [8]. Define two matrices $\mathbf{B} = \mathbf{\Phi}^T[\mathbf{\Phi} \mid \mathbf{D}]$ and $\mathbf{C} = [\mathbf{A} \mid \mathbf{G}]$. The elements of $\mathbf{B}$ and $\mathbf{C}$ are denoted by $b_{i,j}$ and $c_{i,j}$ respectively. The kth stage of the selection procedure consists of:

(i) For $k \leq j \leq M$, compute

$$\left.\begin{aligned} & c_{k,M+i}^{(j)} = b_{j,M+i}/(b_{j,j} + \lambda),\ 1 \leq i \leq n_o \\ & [rerr]_k^{(j)} = \\ & \left(\sum_{i=1}^{n_o}(c_{k,M+i}^{(j)})^2\right)(b_{j,j} + \lambda)/\text{trace}[\mathbf{D}^T\mathbf{D}] \end{aligned}\right\}$$

(ii) Find

$$[rerr]_k = [rerr]_k^{(j_k)} = \max\{[rerr]_k^{(j)},\ k \leq j \leq M\}$$

The j_kth column of $\mathbf{B}$ is interchanged from the kth row upwards with the kth column of $\mathbf{B}$, and then the j_kth row of $\mathbf{B}$ is interchanged from the kth column upwards with the kth row of $\mathbf{B}$. The j_kth column of $\mathbf{C}$ is interchanged up to the $(k-1)$th row with the kth column of $\mathbf{C}$.

(iii) For $k+1 \leq j \leq M + n_o$, compute

$$c_{k,j} = b_{k,j}/b_{k,k}$$

For $k+1 \leq j \leq M$ and $j \leq l \leq M + n_o$, compute

$$\left.\begin{aligned} & b_{j,l} = b_{j,l} - c_{k,j}c_{k,l}b_{k,k} \\ & b_{l,j} = b_{j,l},\ \ l \leq M \end{aligned}\right\}$$

The selection is terminated at the M_s stage when a pre-set tolerance is satisfied

$$1 - \sum_{k=1}^{M_s}[rerr]_k < \xi \tag{11}$$

This produces a subset network containing M_s centres.

2.2 Complexity analysis

The number of multiplications required by this fast ROLS (FROLS) algorithm to select a subset network of size M_s from the matrix $\mathbf{\Phi}$ of size $N \times M$ with n_o outputs is

$$2(n_o+1)M_s + \frac{NM(M+1)}{2} + n_oN(M+1) + \sum_{k=1}^{M_s}(M-k)(M-k+4(n_o+1)) \tag{12}$$

If the ROLS is implemented using the original version [1],[2], the number of multiplications required to perform the same subset network selection is

$$(3n_oN + 2n_o + 2)M_s + n_oN + \sum_{k=1}^{M_s}(2(n_o+1)(N+1)+1)(M-k) \tag{13}$$

Figures 1 and 2 compare the complexity of these two versions for two different cases.

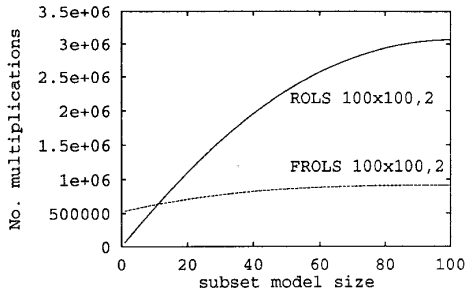

Figure 1: Computational requirement for $\mathbf{\Phi}$ matrix of size 100×100 with 2 outputs.

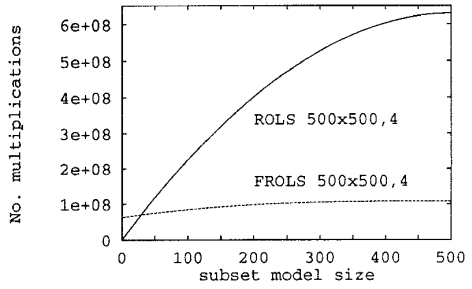

Figure 2: Computational requirement for $\mathbf{\Phi}$ matrix of size 500×500 with 4 outputs.

2.3 Determining value of λ

The appropriate value of λ depends on the underlying system that generates the training data and the choice of basis function. Previous study [4],[7] has suggested that the performance of the RBF network may be fairly insensitive to the precise value of λ. An elegant approach to the selection of regularisation parameter is to adopt a Bayesian interpretation and to calculate the best value of λ using the evidence procedure [9]. This leads to the re-estimation formula for λ

$$\lambda = \frac{\gamma}{N - \gamma} \frac{\text{trace}[\mathbf{E}^T\mathbf{E}]}{\text{trace}[\mathbf{G}^T\mathbf{G}]} \tag{14}$$

where

$$\begin{aligned}\gamma &= M_s - \lambda \,\text{trace}[(\mathbf{W}^T\mathbf{W} + \lambda\mathbf{I})^{-1}] \\ &= \sum_{i=1}^{M_s} \frac{\mathbf{w}_i^T\mathbf{w}_i}{\mathbf{w}_i^T\mathbf{w}_i + \lambda}\end{aligned} \tag{15}$$

is known as the number of good parameter measurements.

3 Example

A two-input two-output data set collected from a 50MW turboalternator is given in [10]. This data set was fitted with a RBF network using the OLS algorithm [2]. The data set is very short, containing only 100 points, and therefore cannot be partitioned into a training set and a validating set. Also the measured system outputs are clean. In order to demonstrate how overfitting can occur in noisy conditions, we artificially create a set of noisy system outputs by adding Gaussian white noise to the clean system outputs

$$y_{n,i}(k) = y_{s,i}(k) + e_i(k), \quad 1 \le k \le 100 \tag{16}$$

where $y_{s,i}(k)$ are the clean system outputs, $e_i(k)$ are Gaussian noises, each having zero mean and variance 0.04, and $e_1(k)$ and $e_2(k)$ are uncorrelated. The system inputs $u_j(k)$, $j = 1, 2$, and $y_{n,i}(k)$ form the training set.

The RBF network with thin-plate-spline nodes is used. The network training input is defined as

$$\begin{aligned}\mathbf{x}(k) \;=\; & [y_{n,1}(k-1)\cdots y_{n,1}(k-3)\; y_{n,2}(k-1) \\ & \cdots y_{n,2}(k-3)\; u_1(k-1)\; u_1(k-2) \\ & u_2(k-1)\; u_2(k-2)]^T\end{aligned} \tag{17}$$

The network prediction errors over the training set are defined as

$$\epsilon_{n,i}(k) = y_{n,i}(k) - \hat{y}_{n,i}(k) \tag{18}$$

where $\hat{y}_{n,i}(k)$ are the network predictions given the input (17). The iterative network output errors are defined by

$$\epsilon_{d,i}(k) = y_{s,i}(k) - \hat{y}_{d,i}(k) \tag{19}$$

where $\hat{y}_{d,i}(k)$ are iterative network outputs given the input

$$\begin{aligned}\hat{\mathbf{x}}(k) \;=\; & [\hat{y}_{d,1}(k-1)\cdots\hat{y}_{d,1}(k-3)\; \hat{y}_{d,2}(k-1) \\ & \cdots\hat{y}_{d,2}(k-3)\; u_1(k-1)\; u_1(k-2) \\ & u_2(k-1)\; u_2(k-2)]^T\end{aligned} \tag{20}$$

The iterative network outputs can be used to evaluate validation perforamance.

Similar to [2], a 45-centre RBF network was identified using the OLS algorithm. Table 1 summarizes the covariances of the network prediction errors and iterative network output errors respectively. The iterative network outputs $\hat{y}_{d,i}(k)$ are superimposed on the clean system

outputs $y_{s,i}(k)$ in Figure 3. Evidence of overfitting can be seen from the covariance of the network prediction errors, which is much smaller than the noise covariance even after taking into account the small number of samples used in calculating the covariance.

covariance of network prediction errors	0.0075 0.0019	0.0019 0.0091
covariance of iterative network output errors	0.0194 -0.0036	-0.0036 0.0162

Table 1: Modelling accuracy by the OLS algorithm.

covariance of network prediction errors	0.0194 -0.0003	-0.0003 0.0199
covariance of iterative network output errors	0.0119 0.0003	0.0003 0.0142

Table 2: Modelling accuracy by the ROLS algorithm.

The ROLS algorithm with $\lambda = 0.7$ was also used to identify a 45-centre RBF network. The modelling accuracy is given in Table 2, and Figure 4 compares the iterative network outputs with the clean system outputs. The results clearly show that the network identified by the ROLS algorithm suffers less from overfitting and captures the underlying system dynamics better than the network obtained without regularisation. Re-estimation formula for λ was also tested. Starting with $\lambda = 0.0$ and after a few repeated runs, λ converged to 0.63 and the resulting network was similar to that obtained with $\lambda = 0.7$.

4 Conclusions

A regularised orthogonal least squares algorithm has been extended for constructing multi-output radial basis function networks. A fast implementation of this learning method has been presented, which offers significant reduction in computational complexity for subset network selection. An example has been included to demonstrate the advantages of combining regularisation with the orthogonal least squares learning.

5 References

[1] S. Chen, C.F.N. Cowan and P.M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks*, Vol.2, No.2, pp.302–309, 1991.

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

[3] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, Vol.78, No.9, pp.1481–1497, 1990.

[4] C. Bishop, "Improving the generalization properties of radial basis function neural networks," *Neural Computation*, Vol.3, No.4, pp.579–588, 1991.

[5] A.R. Barron and X. Xiao, "Discussion of multivariate adaptive regression splines," *Annals of Statistics*, Vol.19, pp.67–82, 1991.

[6] M.J.L. Orr, "Regularised centre recruitment in radial basis function networks," Centre for Cognitive Science, University of Edinburgh, *Research Paper 59*, 1993.

[7] S. Chen, E.S. Chng and K. Alkadhimi, "Regularised orthogonal least squares algorithm for constructing radial basis function networks," submitted to *Int. J. Control*, 1994.

[8] S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset model selection," accepted for publication in *IEEE Trans. Signal Processing*, 1995.

[9] D.J.C. MacKay, "Bayesian interpolation," *Neural Computation*, Vol.4, No.3, pp.415–447, 1992.

[10] G.M. Jenkins and D.G. Watts, *Spectral Analysis and Its Applications*. San Francisco: Holden-Day, 1968.

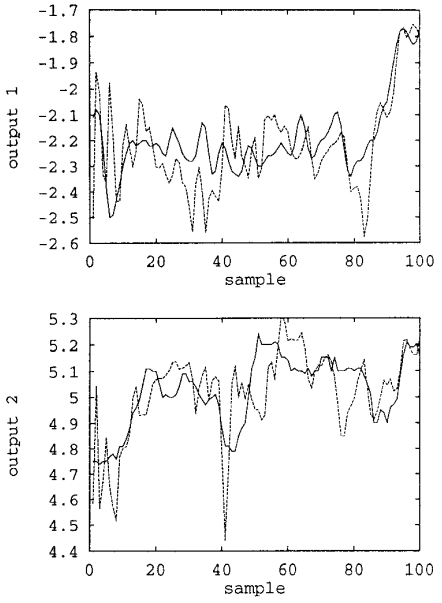

Figure 3: Iterative network outputs (dashed) superimposed on clean system outputs (solid). The network was obtained by the OLS algorithm.

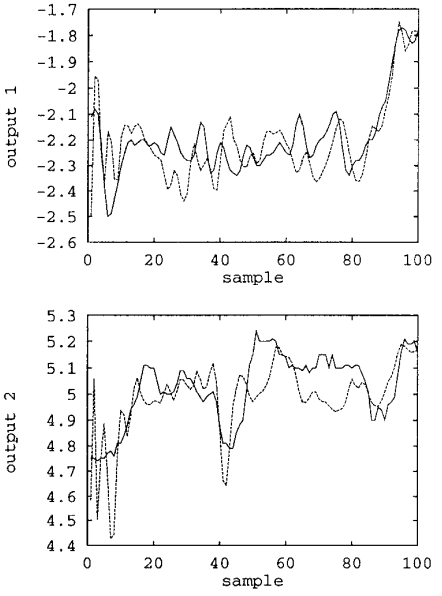

Figure 4: Iterative network outputs (dashed) superimposed on clean system outputs (solid). The network was obtained by the ROLS algorithm.