
Monograph:
Lee, K.L. and Billings, S.A. (2000) Time Series Prediction Using Support Vector Machines, the Orthogonal and the Regularised Orthogonal Least Squares Algorithms. Research Report. ACSE Research Report 780 . Department of Automatic Control and Systems Engineering

---

Time Series Prediction Using Support Vector Machines, the Orthogonal, and the Regularised Orthogonal Least Squares Algorithms

K.L. Lee S.A. Billings
Department of Automatic Control and Systems Engineering
University of Sheffield
Mappin Street, Sheffield S1 3JD
United Kingdom

Research Report No. 780

October 2000

University of Sheffield

Time Series Prediction Using Support Vector Machines, the Orthogonal, and the Regularised Orthogonal Least Squares Algorithms

K.L. Lee
Email: cop98kll@sheffield.ac.uk

S.A. Billings
Email: s.billings@sheffield.ac.uk

Department of Automatic Control and
Systems Engineering
University of Sheffield
Sheffield S1 3JD, UK

Abstract: Generalisation properties of support vector machines, orthogonal least squares and other variants of the orthogonal least squares algorithms are studied in this paper. In particular the zero-order regularised orthogonal least squares algorithm that has been proposed in (Chen et al. 1996) and the first-order regularised orthogonal least squares algorithm which can be obtained using the cost function from support vector machines will be discussed.
Simple noisy sine and sinx functions are used to show that overfitting in the orthogonal least squares algorithm can be greatly reduced if the free parameters of the algorithm are selected properly. Results on three chaotic time series show that the orthogonal least squares algorithm is slightly inferior compared to the other three algorithms. However the strength of the orthogonal least squares algorithm lies in the ability to obtain a very concise or parsimonious model and the algorithm has the fewest number of free parameters compared to the other algorithms.

1. Introduction

Recently, Support Vector Machines (SVM) have been employed for classification and prediction in pattern recognition and regression problems. Vapnik and other co-workers (Boser et al. 1992, Cortes and Vapnik 1995 and Scholkopf et al. 1995) developed the theory of support vector machines. The algorithm is based on statistical learning theory or VC (Vapnik-Chervonenkis) theory (Vapnik 1995), which was developed over the last three decades. Support vector machines are believed to be able to generalise well on unseen data. Many outstanding results (Muller et al. 1997, Drucker et al. 1997 and Mukherjee et al. 1997) have been reported which use support vector machines for time series prediction.

The idea of generalisation ability can be linked to the principle of parsimony. With a parsimonious structure, the problem of overfitting can be avoided and the constructed model can be expected to generalise well. The Orthogonal Least Squares (OLS) algorithm (Chen et al. 1989 and Chen et al. 1991) has been shown to be an efficient procedure for learning parsimonious structures. Recently it has been shown (Drezet and Harrison 1998) that support vector machines are not always able to construct

parsimonious structures in system identification. However the use of the parsimonious principle alone may not totally eliminate overfitting (Chen et al. 1996). Small models may still fit to the noise and the use of the zero-order Regularised Orthogonal Least Squares ($ROLS^0$) algorithm has been proposed. Besides using the zero-order regularised cost function to obtain the zero-order regularised orthogonal least squares algorithm, other cost functions can also be employed to obtain other variants of the orthogonal least squares algorithm. Using the cost function from support vector machines, the first-order Regularised Orthogonal Least Squares ($ROLS^1$) algorithm will be introduced in the present study. Simulation results on simple noisy sine and sinx functions and three well-known chaotic time series will be used to assess the performance of the different algorithms.

The paper is organised as follows. Section 2 briefly introduces support vector machines and the orthogonal least squares algorithm. The use of a zero-order regularised cost function to obtain the zero-order regularised orthogonal least squares algorithm is then discussed. Using the cost function from support vector machines, the first-order regularised orthogonal least squares algorithm is then derived. Empirical studies using noisy sine and sinx functions and three choatic time series, the Henon, Lorenz and Mackey-Glass time series are presented in section 3. Finally, conclusions are given in Section 4.

2. Support Vector Machines, the Orthogonal and the Regularised Orthogonal Least Squares Algorithms

In this section, support vector machines, the orthogonal, and the regularised orthogonal least squares algorithms are briefly discussed. Interested readers are referred to (Vapnik 1995, Burges 1998 and Smola and Scholkopf 1998) for support vector machines, (Korenberg et al. 1988 and Chen et al. 1989) for the orthogonal least squares algorithm and (Chen et al. 1996) for the zero-order regularised orthogonal least squares algorithm.

2.1 Support vector machines

Given a set of data $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2)...(\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_i \in \Re^m$, $y_i \in \Re$, the nonlinear Support Vector Machines (SVM) first map the input data $\{\mathbf{x}_1, \mathbf{x}_2...\mathbf{x}_N\}$ into a high dimensional feature space F by using a nonlinear mapping $\Phi(\cdot)$, and then perform a linear regression in this feature space so that

$$f(\mathbf{x}) = w \cdot \Phi(\mathbf{x}) + b \tag{1}$$

where $w \in F$, b is the threshold and $w \cdot \Phi(\mathbf{x})$ denotes dot product of w with $\Phi(\mathbf{x})$.

To determine the two unknowns (w,b) in equation (1) the following functional is minimised

$$\text{Minimise: } \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\left[\zeta(\xi_i) + \zeta(\xi_i^*)\right] \tag{2}$$

$$\text{subject to} \quad y_i - f(\mathbf{x}_i) \le \varepsilon + \xi_i \tag{3}$$

$$f(\mathbf{x}_i) - y_i \le \varepsilon + \xi_i^* \tag{4}$$

$$\xi_i, \xi_i^* \ge 0 \tag{5}$$

where $\|\cdot\|$ denotes the Euclidean norm, ε is the e-insensitive loss zone introduced by (Vapnik 1995), C is a positive constant, ξ_i, ξ_i^* are training errors and $\zeta(\xi)$ is the loss function. A squared loss function will be used in the present study

$$\zeta(\xi) = \frac{1}{2}\xi^2 \tag{6}$$

Note that with the use of the e-insensitive loss zone, errors $\left(\left|y_i - f(\mathbf{x}_i)\right|\right)$ which are less than ε will not contribute any cost to the optimisation function of equation (2) because the training errors are zero in this case, ξ_i or ξ_i^* is zero in equations (3) or (4).

However the optimisation problem is difficult to solve in the form of equation (2) and the problem can be reformulated in Lagrangian form (Fletcher 1987) as

$$L = \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_{i=1}^{N}(\xi_i^2 + \xi_i^{*2}) - \sum_{i=1}^{N}\alpha_i\left[\varepsilon + \xi_i - y_i + f(\mathbf{x}_i)\right]$$

$$- \sum_{i=1}^{N}\alpha_i^*\left[\varepsilon + \xi_i^* + y_i - f(\mathbf{x}_i)\right] - \sum_{i=1}^{N}\left(\eta_i\xi_i + \eta_i^*\xi_i^*\right) \tag{7}$$

where $\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \ge 0$.

The partial derivative of L with respect to w, b, ξ_i, ξ_i^* must vanish, hence substituting equation (1) into equation (7) and carrying out the necessary operations gives

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{N}\left(\alpha_i - \alpha_i^*\right)\Phi(\mathbf{x}_i) = 0 \tag{8}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right) = 0 \tag{9}$$

$$\frac{\partial L}{\partial \xi_i^{(*)}} = C\xi_i^{(*)} - \alpha_i^{(*)} - \eta_i^{(*)} = 0 \tag{10}$$

where equation (10) represents the short form of $\frac{\partial L}{\partial \xi_i} = C\xi_i - \alpha_i - \eta_i = 0$ and $\frac{\partial L}{\partial \xi_i^*} = C\xi_i^* - \alpha_i^* - \eta_i^* = 0$.

Using equations (8), (9) and (10), the dual formulation of equation (7) can be expressed as

$$Maximise: -\frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$$

$$-\varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*) - \frac{1}{2C}\sum_{i=1}^{N}((\alpha_i + \eta_i)^2 + (\alpha_i^* + \eta_i^*)^2) \tag{11}$$

subject to $$\sum_{i=1}^{N}(\alpha_i^* - \alpha_i) = 0 \tag{12}$$

$$\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \geq 0 \tag{13}$$

The dual variables (η_i, η_i^*) only appear in the last term of equation (11) and are subject to the constraint of equation (13), therefore the maximum of equation (11) can only be achieved if

$$\eta_i = \eta_i^* = 0 \tag{14}$$

Using both equations (10) and (14) gives

$$\xi_i^{(*)} = \frac{\alpha_i^{(*)}}{C} \tag{15}$$

By reformulating the problem into Lagrangian form the optimisation function only involves the dot product of the training data, $\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$ and this is a very important property. In general the mapping $\Phi(\cdot)$ may not be known explicitly but using Mercer kernels (Vapnik 1995) which satisfy

$$k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j) \tag{16}$$

the problem can be solved without ever mapping explicitly to the feature space F. The polynomial kernel $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^p$, where p is the degree of nonlinearity, and the Gaussian kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\| / \sigma^2\right)$, where σ is the width, are some typical kernels which satisfy the Mercer condition.

Finally, with equation (14) and using Mercer kernels, the optimisation function can be expressed as

$$Maximise: -\frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(\mathbf{x}_i, \mathbf{x}_j)$$

$$-\varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*) - \frac{1}{2C}\sum_{i=1}^{N}(\alpha_i^2 + \alpha_i^{*2}) \tag{17}$$

subject to $$\sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right) = 0 \tag{18}$$

$$\alpha_i, \alpha_i^* \geq 0 \tag{19}$$

A standard optimisation package can be used to solve the quadratic maximisation problem and the values of the Lagrange multipliers α_i and α_i^* can be found. With the use of the e-insensitive loss zone, only a number of coefficients $\alpha_i - \alpha_i^*$ will be different from zero and the data points $\mathbf{x}_i$ associated to them are called support vectors. The use of the interior point algorithms to solve the optimisation problem is preferred (Smola and Scholkopf 1998) because the algorithms solve both the primal and dual of equation (17) simultaneously. Note that the dual of equation (17) is the dual and dual of equation (2) and the value of the threshold term b can be obtained directly (Smola et al. 1998). The value of b can also be computed using the Karush-Kuhn Tucker (KKT) conditions (Fletcher 1987). Applying the KKT conditions give

$$\eta_i^{(*)}\xi_i^{(*)} = 0 \tag{20}$$

$$\alpha_i\left(\varepsilon + \xi_i - y_i + w\cdot\Phi(\mathbf{x}_i) + b\right) = 0 \tag{21}$$

$$\alpha_i^*\left(\varepsilon + \xi_i^* + y_i - w\cdot\Phi(\mathbf{x}_i) - b\right) = 0 \tag{22}$$

Using support vectors such that α_i or α_i^* is nonzero, the right hand bracket term of the equation (21) or (22) is zero. Using equation (15) the value of b can then be calculated as

$$b = y_i - \varepsilon - \frac{\alpha_i}{C} - \sum_{j=1}^{N}\left(\alpha_j - \alpha_j^*\right)k\left(\mathbf{x}_i, \mathbf{x}_j\right) \text{ or } b = y_i + \varepsilon + \frac{\alpha_i^*}{C} - \sum_{j=1}^{N}\left(\alpha_j - \alpha_j^*\right)k\left(\mathbf{x}_i, \mathbf{x}_j\right) \tag{23}$$

In this study the package LOQO (Vanderber 1994) which is based on the interior point algorithms was used.

2.1.1 Structural risk minimisation

If there was no control on the capacity/complexity of the machine (the term $\|w\|^2$ in equation (2)), a regression function w with zero empirical risk (error) can always be found by increasing the complexity of the machine such as using a higher number of centres for the Gaussian kernel. However zero empirical risk does not imply zero expected risk on unseen samples, and a bound on the expected risk has been derived in Vapnik (1995). The expected risk can be calculated with probability $(1-\upsilon)$ as

$$R \leq R_{emp} + \sqrt{\frac{h(\log_e \frac{2N}{h} + 1) - \log_e(\frac{\upsilon}{4})}{N}} \tag{24}$$

where R is the expected risk, R_{emp} is the empirical risk, N is the number of training samples and h is the Vapnik-Chervonenkis (VC) dimension. The VC-dimension depends on the chosen class of functions such as Gaussian or polynomial kernels, whereas the empirical risk and the expected risk depend on the particular function chosen by the training procedure. The idea of Structural Risk Minimisation (SRM) is to arrange the entire class of functions into nested subsets with each subset having a VC-dimension of h_i where $h_1 < h_2 < h_3 ...$ For each subset the goal of training is simply to minimise the empirical risk. Finally, the particular trained machine (function) with minimal expected risk as calculated from equation (24) is selected.

However the VC-dimension of a chosen class of functions is independent of A where $\|w\|^2 \leq A$, A is a positive constant and the VC-dimension can be infinite if the dimension of the feature space is infinite. Therefore an extension of the VC-dimension known as the V_γ-dimension has been developed (Alon et al. 1997 and Evgeniou et al. 1999) and the V_γ-dimension is dependent on A and is finite even if the feature space dimension is infinite. The extended structural risk minimisation principle (Evgeniou et al. 1999) can then be formulated as

Minimise: $$\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 \tag{25}$$

subject to $$\|w\|^2 \leq A_j \tag{26}$$

where A_j is a monotonically increasing sequence of positive constants.

Unfortunately the above formulation is difficult to solve as this involves a constrained minimisation with nonlinear constraints. The Lagrangian form of equation (25) is

$$L = \sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 - C_j(A_j - \|w\|^2) \tag{27}$$

where $C_j \geq 0$

If the optimal value of the Lagrange multiplier is known and is noted as C_j^* then equation (27) can be simplified as

Minimise: $$\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 + C_j^*\|w\|^2 \tag{28}$$

with respect to w. This is in the same form as the formulation for support vector machines. Therefore support vector machines implement the structural risk minimisation principle and hence many excellent

results have been reported using support vector machines. In practical situations, the optimal value of C_j^* is rarely known and this has to be estimated using techniques such as cross-validation.

2.2 Orthogonal least squares type algorithms

Assume that a training set of N samples $\{(\mathbf{x}_1, y_1),...(\mathbf{x}_N, y_N)\}$ where $\mathbf{x}_i \in \Re^m$ and $y_i \in \Re$ is available and a regression model can be obtained as

$$f(\mathbf{x}) = \sum_{i=1}^{N} \theta_i \phi_i(\mathbf{x}) \tag{29}$$

where θ_i is the unknown parameters and $\phi_i(\mathbf{x})$ is assumed to be the nonlinear Mercer kernels in this study, $\phi_i(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}_i)$, but other nonlinear expansions such as a polynomial expansions are also possible (Chen and Billings 1989). Therefore the output y_i can be expressed as

$$y_i = f(\mathbf{x}_i) + \xi_i = \sum_{j=1}^{N} \theta_j k(\mathbf{x}_i, \mathbf{x}_j) + \xi_i \tag{30}$$

where ξ_i is again the training error. The model equation (29) can be referred to as the 'full' model since all the training samples are used to construct the model. The use of the 'full' model is highly undesirable due to the overfitting problem and a longer time will be needed to train and to test large models. An efficient subset selection procedure has been derived based on the orthogonal least squares method (Chen et al. 1989). Equation (30) can be rewritten in matrix form as

$$Y = P\Theta + \Xi \tag{31}$$

where $$P = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \tag{32}$$

An orthogonal decomposition of P is given as

$$P = TQ \tag{33}$$

where Q is an N x N upper unit triangular matrix.

$$Q = \begin{bmatrix} 1 & \lambda_{12} & \cdots & \lambda_{1N} \\ & 1 & \cdots & \lambda_{2N} \\ & & \ddots & \vdots \\ 0 & & & 1 \end{bmatrix} \tag{34}$$

and T is a N x N matrix with orthogonal columns satisfying

$$T^T T = diag\{K_1, K_2, ...K_N\}, \quad K_i = t_i^T t_i \tag{35}$$

Rearranging equation (31) yields

$$\mathrm{Y} = (PQ^{-1})(Q\Theta) + \Xi = Tg + \Xi \tag{36}$$

The total squared error cost function

$$J = \Xi^T \Xi \tag{37}$$

can be used to estimate the unknown parameters g_i as

$$g_i = \frac{t_i^T \mathrm{Y}}{t_i^T t_i} \tag{38}$$

With $Q\theta = g$ and knowing Q and g, θ can be determined through back substitution.

Using equations (36) & (38), the cost function can alternatively be expressed as

$$J = Y^T Y - \sum_{i=1}^{N} g_i^2 (t_i^T t_i) \tag{39}$$

The Error Reduction Ratio (ERR) (Korenberg et al. 1988) due to P_i can be expressed as

$$ERR_i = \frac{g_i^2 t_i^T t_i}{\mathrm{Y}^T \mathrm{Y}} \tag{40}$$

Using the ERR ratio, significant model terms can be selected in a forward-selection procedure. At the i iteration, the term which gives the largest value of ERR_i is selected and added to the previously selected $(i-1)$ terms model. The selection procedure can be terminated using cross-validation. Full details of the forward regression version of this algorithm are available in (Chen et al 1989). Empirical results have shown that parsimonious models can be obtained using the orthogonal least squares algorithm.

2.2.1 Zero-order regularised orthogonal least squares algorithm

If the samples are highly noisy, small models constructed using the orthogonal least squares algorithm may still fit to the noise and the use of the zero-order Regularised Orthogonal Least Squares ($ROLS^0$) algorithm has been proposed (Chen et al 1996).

The zero-order regularised cost function used for the $ROLS^0$ algorithm is

$$J = \Xi^T \Xi + C g^T g \tag{41}$$

where C is a positive constant.

The corresponding expressions for g_i and the zero-order Regularised Error Reduction Ratio ($RERR_i^0$) can be derived as

$$g_i = \frac{t_i^T Y}{t_i^T t_i + C} \tag{42}$$

$$RERR_i^0 = \frac{(t_i^T t_i + C) g_i^2}{Y^T Y} \tag{43}$$

Similarly, the term which gives the largest $RERR_i^0$ at each iteration i is selected and cross-validation can be used to terminate the selection procedure.

The constant C can also be lumped together with the training error term in order to be consistent with the cost function used in support vector machines and hence the subsequent cost function is

$$J = C\left(\Xi^T \Xi\right) + g^T g \tag{44}$$

and the resulting g_i and $RERR_i^0$ are

$$g_i = \frac{t_i^T Y}{t_i^T t_i + 1/C} \tag{45}$$

$$RERR_i^0 = \frac{(t_i^T t_i + 1/C) g_i^2}{Y^T Y} \tag{46}$$

2.2.2 First-order regularised orthogonal least squares algorithm

As well as considering the total squared error (equation (37)) and zero-order regularised (equation (41)) cost functions, other cost functions may also be employed. In particular the structural risk minimisation principle is well founded in statistical learning theory and the cost function used in support vector machines may be useful. This is investigated below.

The cost function is

$$J = C(\Xi^T \Xi) + w^T w \tag{47}$$

where $w = \sum_{i=1}^{N} \theta_i \Phi(\mathbf{x}_i)$

Let $\Lambda_N(\mathbf{x}) = \left[\Phi(\mathbf{x}_1) \quad \cdots \quad \Phi(\mathbf{x}_N)\right]^T$ and hence

$$w = \Lambda_N(\mathbf{x})^T \Theta \tag{48}$$

and P in equation (32) can be written as

$$P = \Lambda_N(\mathbf{x}) \Lambda_N(\mathbf{x})^T \tag{49}$$

Consider the expression

$$\begin{aligned} \Theta^T P^T P \Theta &= \Theta^T \left(\Lambda_N(\mathbf{x}) \Lambda_N(\mathbf{x})^T\right)^T \left(\Lambda_N(\mathbf{x}) \Lambda_N(\mathbf{x})^T\right) \Theta \\ &= \Theta^T \Lambda_N(\mathbf{x}) D \Lambda_N(\mathbf{x})^T \Theta \\ &= D w^T w \end{aligned} \tag{50}$$

where $D = \Lambda_N(\mathbf{x})^T \Lambda_N(\mathbf{x})$ is a positive constant.

Therefore minimising $w^T w$ is equivalent to minimising $\Theta^T P^T P\Theta$ and knowing that $P\Theta = gT$, the cost function equation (47) can be re-expressed as

$$J = C(\Xi^T \Xi) + \Theta^T P^T P\Theta = C(\Xi^T \Xi) + g^T T^T Tg \tag{51}$$

The matrix $T^T T$ is diagonal and the corresponding g_i and first-order Regularised Error Reduction Ratio ($RERR_i^1$) can easily be derived as

$$g_i = \left(\frac{C}{C+1}\right)\frac{t_i^T Y}{t_i^T t_i} \tag{52}$$

$$RERR_i^1 = \left(\frac{C+1}{C}\right)\frac{t_i^T t_i g_i^2}{Y^T Y} \tag{53}$$

This approach will be referred to as the first-order Regularised Orthogonal Least Squares (ROLS[1]) algorithm and a similar selection procedure of choosing the largest value of $RERR_i^1$ at each iteration can be performed. Regardless of the value chosen for the constant C the same terms as the original orthogonal least squares algorithm will be selected with a scale value of the parameters g_i (compare equations (38) and (40) with equations (52) and (53) respectively). Therefore the performance of the first-order regularised orthogonal least squares algorithm should be similar to the orthogonal least squares algorithm. To increase the performance and the flexibility of the first-order regularised orthogonal least squares algorithm, an optimisation procedure will be performed after M important terms have been selected. At this optimisation step, the direct support vector machines cost function with the e-insensitive loss zone will be used and the optimisation operation will then be performed to obtain a global solution of the parameters θ_i.

After the M important terms have been selected, the regression model can be expressed as

$$f(\mathbf{x}) = \sum_{i=1}^{M} \theta_i k(\mathbf{x}, \mathbf{x}'_i) \tag{54}$$

where $\mathbf{x}'_i, i = 1...M$ are the selected terms from $\mathbf{x}_i, i = 1...N$ using the orthogonal least squares algorithm.

The optimisation cost function will be

$$\text{Minimise: } \frac{1}{2}\sum_{i,j=1}^{M} \theta_i k(\mathbf{x}'_i, \mathbf{x}'_j)\theta_j + \frac{C}{2}\sum_{i=1}^{N}(\xi_i^2 + \xi_i^{*2}) \tag{55}$$

subject to $y_i - \sum_{j=1}^{M} \theta_j k(\mathbf{x}_i, \mathbf{x}'_j) \leq \varepsilon + \xi_i$ (56)

$$\sum_{j=1}^{M} \theta_j k(\mathbf{x}_i, \mathbf{x}'_j) - y_i \leq \varepsilon + \xi_i^* \quad (57)$$

and $\xi_i, \xi_i^* \geq 0$.

Standard optimisation packages such as LOQO (Vanderber 1994) can be used to obtain the values of θ. With the introduction of the extra parameter ε (the e-Insensitive loss zone), the resulting first-order regularised orthogonal least squares algorithm is more flexible but with more parameters to be determined during cross-validation.

3 Experiments

In the first experiment, similar examples to those used in Chen et al. (1996) were conducted. In Chen et al. (1996), the first example was used to show that the orthogonal least squares algorithm can fit to the noise whereas using the regularised orthogonal least squares algorithm the overfitting problem is less likely to occur. An RBF network with a gaussian kernel and a width of $\sigma = 0.2$ was used to approximate the scalar function

$$f_{true}(x_i) = \sin(2\pi x_i), \; 0 \leq x_i \leq 1 \quad (58)$$

One hundred training samples were generated where x_i was taken from the uniform distribution [0 1] and zero mean white gaussian noise e_i with standard deviation 0.4 was added to the function output to obtain the output y_i as

$$y_i = f_{true}(x_i) + e_i \quad (59)$$

The noisy data and the true noise-free data are plotted together in Figure 1. Following Chen et al. (1996), the constant C was chosen to be 1, and 15 centres were selected from the training samples for both the OLS and zero-order ROLS^0 algorithms (using equations (42) and (43)). The resulting approximation functions obtained from both algorithms are shown in Figures 2a and 2b. Clearly, overfitting can be seen in Figure 2a using the OLS algorithm whereas in Figure 2b using the ROLS^0 algorithm, visually the overfitting problem is much reduced. However a simple check with the total sum of the error reduction ratio showed that $\left(\sum_{i=1}^{15} ERR_i \right) = 0.84$ whereas $\left(\sum_{i=1}^{15} RERR_i^0 \right) = 0.76$. A higher value of the total sum of the error reduction ratio implies higher chances of overfitting. Since this was a rather simple example two centres were sufficient to approximate the sine function. The resulting approximation functions obtained by OLS and ROLS^0 using two centres are shown in Figures

3a and 3b. Visually, no overfitting has occurred for both approximation functions. The values of $\left(\sum_{i=1}^{k} ERR_i\right)$ and $\left(\sum_{i=1}^{k} RERR_i^0\right)$ are plotted in Figure 4 for the number of centres (k) from 2 to 15.

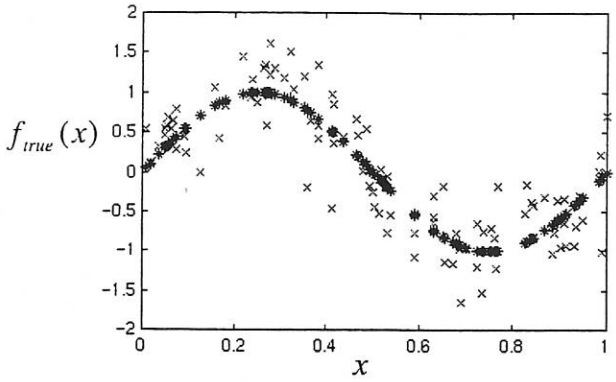

Figure (1) Noisy training data (crosses) and the true noise-free data (asterisks).

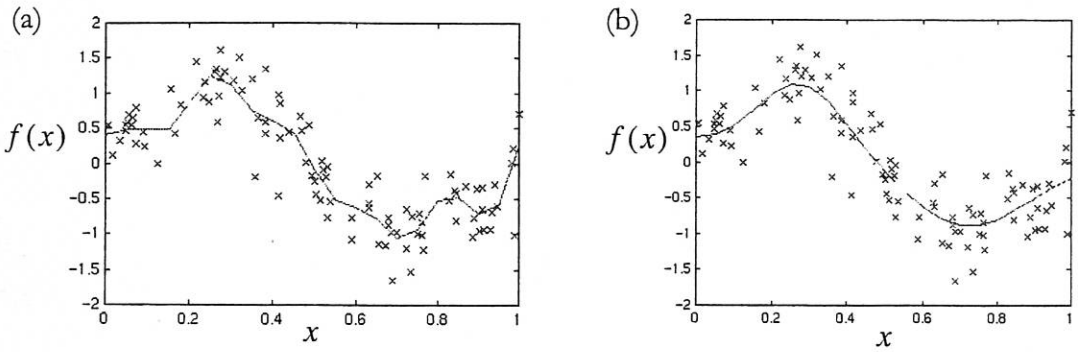

Figure (2) Approximation functions constructed by (a) orthogonal least squares and (b) zero-order regularised orthogonal least squares algorithms using a Gaussian kernel with 15 centres and a width of 0.2.

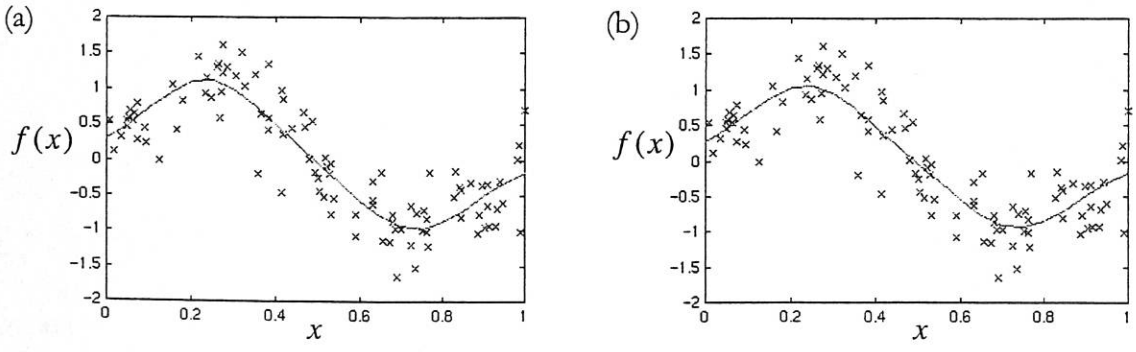

Figure (3) Approximation functions constructed by (a) orthogonal least squares and (b) zero-order regularised orthogonal least squares algorithms using a Gaussian kernel with 2 centres and a width of 0.2.

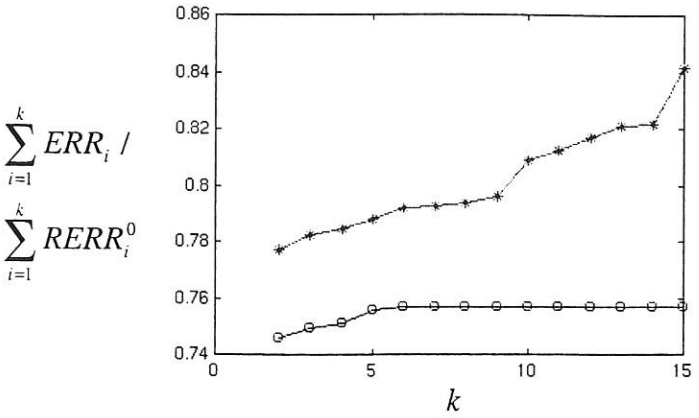

Figure (4) The total sum of the error reduction ratio (asterisks) and the zero-order regularised error reduction ratio (circles) for the number of centres (k) from 2 to 15.

Another problem associated with the use of the OLS algorithm is that missing data in interoir regions of the sample space could produce opportunities for local overfitting as shown in (Orr 1993 and Orr 1995). Consider the following scalar sinx function,

$$f_{true} = \frac{\sin x}{x} \tag{60}$$

Sixty four samples from four different regions of x were drawn and corrupted by Gaussian noise of zero mean and standard deviation 0.05. A Guassian kernel of width 2 was used and the OLS algorithm was employed to select a subset of 10 centres from a total set of 201 points equally spaced between (-30,50). The function obtained by OLS is shown in figure 5.

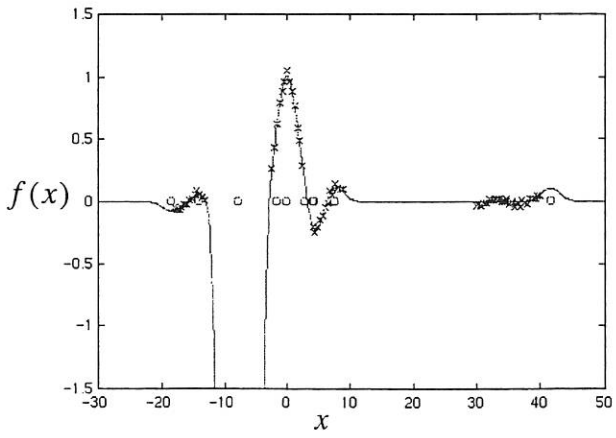

Figure (5) Approximation function constructed by OLS using a Gaussian kernel with 10 centres and a width of 2. The centres are chosen from a set of 201 points equally spaced between (-30,50). The centres chosen are shown in circles and the data points are plotted in crosses.

It can clearly be seen that the approximated function obtained by OLS algorithm can perform poorly in areas where there are no data samples. This is because the OLS algorithm may select centres in these regions where there are no data and for such centres to influence the reduction in data error, large weights will result. As shown in Orr (1993), this problem can be solved if regularization is used because regularization will avoid choosing centres with large weights. However a much simpler solution is to create the potential set of centres solely from the data, hence giving no chance for the OLS algorithm to select centres from a region where there are no data. With the centres chosen solely from the data points, the resulting function obtained by the OLS algorithm is shown in Figure 6. It can clearly be seen that the local overfitting problem is now avoided. This is because the Gaussian kernel belongs to the set of local basis functions. Therefore each centre will have decreasing influence as the distance gets further away from the centre. Since no centres are selected in the empty data region, the influence of the nearby data point centres selected by OLS should diminish in this region and hence local overfitting is avoided.

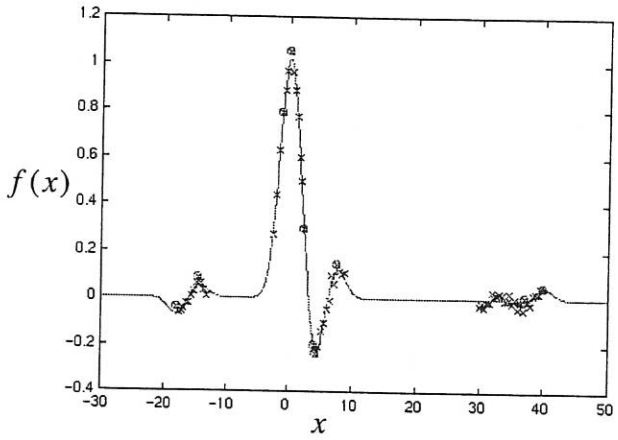

Figure (6) Approximation function constructed by OLS using a Gaussian kernel with 10 centres and a width of 2. The centres are chosen from the data points which are shown in circles and the data points are plotted in crosses.

Another factor which plays an important role in this situation is the width of the Guassian kernel. If too large a width is chosen, the influence of the selected centres will not diminish rapidly and a local overfitting problem may still occur as shown in Figure 7.

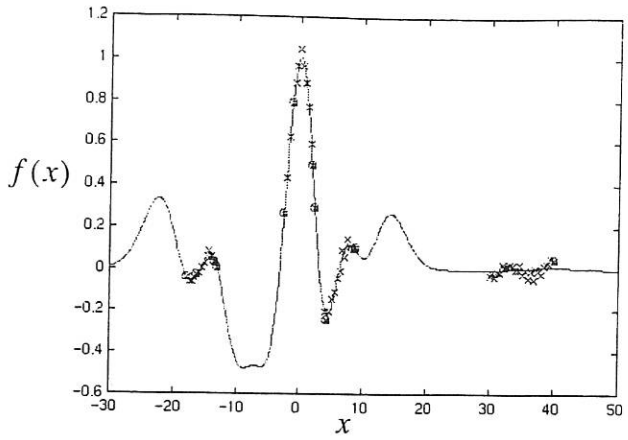

Figure (7) Approximation function constructed by OLS using a Gaussian kernel with 12 centres and a width of 5. The centres are chosen from the data points which are shown in circles and the data points are plotted in crosses.

Therefore in order to avoid the problem of local overfitting, the OLS algorithm can still be used but proper selection of the width of the Gaussian kernel is required and the set of potential centres for selection has to be generated from the data points. The problem of local overfitting is likely to occur if a global basis is employed such as a polynomial function or a thin-plate spline kernel, so that the use of regularization may be preferable in these cases.

Several observations can be drawn from these two simple examples.

i) The comparison would favour one algorithm rather than another if the centres were fixed at a certain number. Instead the number of centres should vary and the different optimal number of centres for each different algorithm should be properly selected. The comparison will then be more meaningful. This applies not only to the number of centres but also to all the free parameters associated with the algorithms such as the constant C in ROLS^0.

ii) The approximation functions constructed by ROLS^0 are less sensitive to the number of centres selected after a minimal number of centres have been included in the model and $\left(\sum_{i=1}^{k} RERR_i^0 \right)$ reaches a plateau for large values of k as seen from Figure 4. Hence ROLS^0 is less prone to overfitting. On the other hand, if the required number of centres of the OLS algorithm is properly selected, the overfitting problem may also be avoidable.

iii) Although the computational effort is the same for both OLS and ROLS^0 algorithms if the constant C is known, in practical situations the optimal value of C is rarely known and this has to be estimated by techniques such as cross-validation. Therefore the advantage gained by ROLS^0 in not

requiring to select the precise optimal number of centres is offset by the requirement to determine the optimal value of C.

iv) The problem of local overfitting by the OLS algorithm can be avoided if local basis functions are used with the width of the basis properly selected and the centres are chosen from the data.

The problems associated with the use of the OLS algorithm can be greatly reduced if the algorithm is used properly. However it is expected that the performance (one step ahead prediction error on unseen data) of the OLS algorithm may be worse than ROLS^0. This is because the OLS algorithm is a subset of the ROLS^0 algorithm. By setting the value of C in equation (41) equal to zero in the ROLS^0 algorithm, the OLS algorithm is obtained. The predictive capability of the four algorithms (support vector machines, orthogonal least squares, zero-order regularised orthogonal least squares and first-order regularised orthogonal least squares algorithms) on three well-know chaotic time series benchmarks, Henon, Lorenz and Mackey-Glass time series are examined next.

The Henon equation is

$$x_{i+1} = 1.4 - x_i^2 + 0.3x_{i-1} \tag{62}$$

A total of 1500 points of x_i were generated and white Gaussian noise e_i with zero mean and a noise level of 10% was added to the values of x_i to obtain a noisy y_i as

$$y_i = x_i + e_i \tag{63}$$

The noise level is defined as the ratio of the standard deviation of the noise with respect to the standard deviation of the clean noise-free signal in percentages. The first 500 noisy data points of y_i served as the training data set and the next 1000 noisy points were used as the testing set. The embedding dimension m used was 4, that is $f(\mathbf{x}_i) = f(y_i, y_{i-1}, ..., y_{i-m+1})$, with $m = 4$.

The Lorenz equations are

$$\begin{aligned} \dot{x}_T &= 10(y_T - x_T) \\ \dot{y}_T &= 28x_T - y_T - x_T z_T \\ \dot{z}_T &= -\frac{8}{3} z_T + x_T y_T \end{aligned} \tag{64}$$

A fourth order Runge-Kutta algorithm was used to integrate the equations with a step size of 0.01. A total of 1500 points of x_i were collected with a sampling rate of 0.05. Zero mean white Gaussian noise of 13% noise level was added to obtain the noise corrupted outputs y_i. Similarly, the first 500 noisy

data were used as the training set with the remaining 1000 data points as the testing set. The embedding dimension used was $m = 6$ (Casdagli 1989 and Mukherjee et al. 1997).

The Mackey-Glass equation is

$$\dot{x}_T = -0.1x_T + \frac{0.2x_{T-\Delta}}{1 + x_{T-\Delta}^{10}} \tag{65}$$

with $\Delta = 17$.

A fourth order Runge-Kutta algorithm was used again to integrate the equation with a step size of 0.01 and 1500 data points were collected with a sampling rate of 6. Zero mean white Gaussian noise was added to the data with a 23% noise level. The first 400 noisy points were used as the training set and the next 1100 noisy points were used as the testing set. The embedding dimension used was $m = 4$ (Casdagli 1989 and Mukherjee et al. 1997).

For the four different algorithms, the Gaussian kernel was used on all three time series. The free parameters of each algorithm were varied and the (near) optimal values which gave the minimum one step ahead mean squared error on the test set were selected. For example, the free parameters of the support vector machines are σ (Gaussian kernel width), ε (e-insensitive loss zone) and C (the positive constant). The smallest step size chosen for ε was 0.05 and $C \in \{10^{-1}, 10^{0}, ..., 10^{5}\}$. Note that the equations employed for the zero-order regularised orthogonal least squares algorithm were equation (45) and equation (46). The results are presented in Tables 1, 2 and 3.

	σ	ε	C	No of centres	Mean squared error on test set
SVM	1	0.10	10	297	0.0388
OLS	1	---	---	33	0.0399
ROLS^0	1	---	10	30	0.0392
ROLS^1	1	0.10	10	47	0.0387

Table 1 Comparison of the minimum one step ahead mean squared prediction error on the test set of the Henon time series between Support Vector Machines (SVM), Orthogonal Least Squares (OLS), zero-order Regularised Orthogonal Least Squares (ROLS^0) and first-order Regularised Orthogonal Least Squares (ROLS^1) algorithms. Note that ---- implies not applicable.

	σ	ε	C	No of centres	Mean squared error on test set
SVM	23	0.90	100	254	2.067
OLS	46	---	---	16	2.157
ROLS^0	24	---	1000	33	2.059
ROLS^1	28	0.40	100	48	2.071

Table 2 Comparison of the minimum one step ahead mean squared prediction error on the test set of the Lorenz time series between Support Vector Machines (SVM), Orthogonal Least Squares (OLS), zero-order Regularised Orthogonal Least Squares (ROLS^0) and first-order Regularised Orthogonal Least Squares (ROLS^1) algorithms. Note that ---- implies not applicable.

	σ	ε	C	No of centres	Mean squared error on test set
SVM	0.5	0.0	10	400	0.005087
OLS	0.5	---	---	14	0.005119
ROLS^0	0.5	---	100	25	0.005101
ROLS^1	0.5	0.0	1000	28	0.005087

Table 3 Comparison of the minimum one step ahead mean squared prediction error on the test set of the Mackey-Glass time series between Support Vector Machines (SVM), Orthogonal Least Squares (OLS), zero-order Regularised Orthogonal Least Squares (ROLS^0) and first-order Regularised Orthogonal Least Squares (ROLS^1) algorithms. Note that ---- implies not applicable.

From the above three Tables 1, 2 and 3, the performance of the one step ahead prediction error on the test set of the orthogonal least squares algorithm was slightly inferior when compared to the other algorithms for the three chaotic time series. The best algorithm in terms of the smallest prediction error in each case was less than 5% better than the orthogonal least squares algorithm. However the orthogonal least squares algorithm has the fewest number of free parameters which implies fast training times and the constructed models are usually much smaller (having fewer number of centres) than the other algorithms which in turn produces fast testing times. In support vector machines, although the number of centres is found automatically, this number is usually much larger than the other algorithms. This shows that orthogonal least squares type algorithms can construct parsimonious models. Note that the performance of the first-order regularised orthogonal least squares algorithm based on the one step ahead prediction errors was very similar to support vector machines for all three cases but more concise models were obtained.

The iterated multistep mean squared prediction errors on the test set for the three time series using the four different algorithms were computed and the results are shown in Figures 8, 9 and 10. As an example, the two step ahead prediction can be calculated as

$$\hat{y}_{i+2} = f(\hat{y}_{i+1}, y_i, \ldots, y_{i-m+2}) \tag{66}$$

where $\hat{y}_{i+1}$ is the one step ahead prediction.

From Figures 8, 9 and 10, the performance of the four different algorithms were comparatively similar, especially for small step ahead predictions.

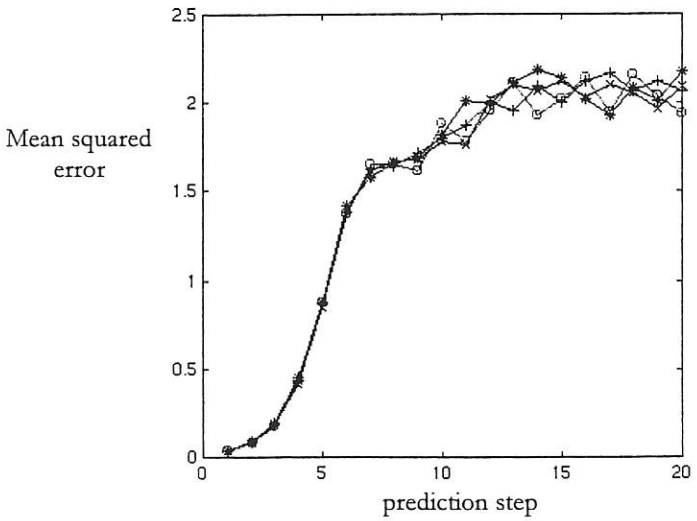

Figure (8) The mean squared multistep ahead prediction error for the test set of the Henon time series using support vector machines (crosses x), orthogonal least squares (asterisks), zero order regularised orthogonal least squares (circles) and first-order regularised least squares (crosses +) algorithms.

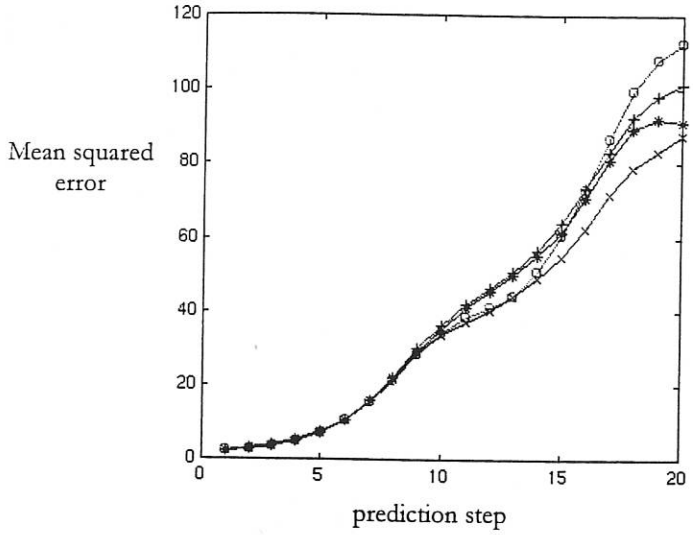

Figure (9) The mean squared multistep ahead prediction error for the test set of the Lorenz time series using support vector machines (crosses x), orthogonal least squares (asterisks), zero order regularised orthogonal least squares (circles) and first-order regularised least squares (crosses +) algorithms.

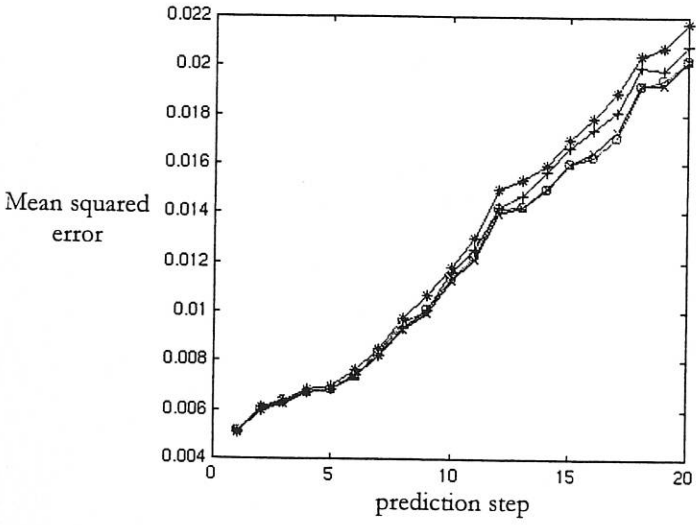

Figure (10) The mean squared multistep ahead prediction error for the test set of the Mackey-Glass time series using support vector machines (crosses +), orthogonal least squares (asterisks), zero-order regularised orthogonal least squares (circles) and first-order regularised least squares (crosses x) algorithms.

4 Conclusions

Using a simple noisy sine function example, it was shown that if the free parameters of the orthogonal least squares and the zero-order regularised orthogonal least squares algorithms are fixed at a pre-

specified value, the results obtained tend to favour one algorithm rather than another. To get a useful comparison, the free parameters of each different algorithm should be properly selected. Proper selection of the free parameters also reduces the overfitting problems associated with the orthogonal least squares algorithm.

Support vector machines achieve good generalisation properties using the structural risk minimisation principle. The orthogonal least squares algorithm uses the principle of parsimony to construct small models and hence the constructed models are less likely to fit to the noise and may result in better predictions. The zero or first-order regularised orthogonal least squares algorithms use the same parsimonious principle but with slight modifications to the cost function. Both the square of the errors and the square of the weights or the norm of the regression function are minimised. The performance of the four algorithms was compared based on three well-known chaotic time series with moderate noise levels (10~20%) and the results have shown that the prediction errors of the four algorithms were relatively similar. If the objective is to obtain minimum prediction error on a test set, then support vector machines or regularised orthogonal least squares algorithms should be employed. Otherwise the use of the orthogonal least squares algorithm is preferred since there are less free parameters to determine and a very concise model is usually obtained.

References

Alon, N., Ben-David, S., Cesa-Bianchi, N., and Haussler, D., 1997, Scale-insensitive dimensions, uniform convergence, and learnability. Journal of the ACM, **44**, (4), 616-631.

Boser, B., Guyon, I., and Vapnik, V., 1992, A training algorithm for optimal margin classifiers. Firth Annual Workshop on Computational Learning Theory. ACM Press, Pittsburgh.

Burges, C.J.C, 1998, A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, **2**, (2), 121-167.

Casdagli, M., 1989, *Nonlinear* Prediction of chaotic time series. Physica D, **35**, (3), 335-356.

Chen, S., and Billings, S.A., 1989, Modelling and Analysis of Non-linear Time Series. International Journal of Control, **50**, (6), 2151-2171.

Chen, S., Billings, S.A., and Luo, W., 1989, Orthogonal Least Squares methods and their application to non-linear system identification. International Journal of Control, **50**, (5), 1873-96.

Chen, S., Chng, E.S., and Alkadhimi, K., 1996, Regularized orthogonal least squares algorithm for constructing radial basis function networks. International Journal of Control, **64**, (5), 829-37.

Chen, S., Cowan, C.F.N., and Grant, P.M., 1991, Orthogonal least squares learning algorithm for radial basis function networks. IEEE Transactions on Neural Networks, **2**, (2), 302-309.

Cortes, C., and Vapnik, V., 1995. Support Vector Networks. Machine Learning, **20**, 1-25.

Drezet, P.M.L., and Harrison, R.F., 1998, Support vector machines for system identification. UKACC International Conference on Control '98, 2 vol xiviii+1727, vol 1, pp 688-92.

Drucker H. Burges, C. Kaufman, L. Smola, A. and Vapnik, V. 1997. Support vector regression machines. Advances in Neural Information Processing Systems, 9, pp 155-161.

Evgeniou, T., Pontil, M., and Poggio, T., 1999, A unified framework for regularisation networks and support vector machines. Technical Report AI Memo No 1654, MIT Artificial Intelligence Laboratory.

Fletcher, R., 1987, Practical methods of optimisation, 2nd edition, (John Wiley and Sons).

Korenberg, M. Billings, S.A. Liu, Y.P. Mcllroy, P.J. 1988. Orthogonal parameter estimation algorithm for non-linear stochastic systems. International Journal of Control. **48**, (1), 193-210.

Mukherjee, S. Osuna, E. and Girosi, F. 1997. Nonlinear prediction of chaotic time series using support vector machines. Neural networks for signal processing VII, Proceedings of the 1997 IEEE signal processing society workshop, xiii+667, pp 511-520.

Muller K.R., Smola, A., Ratsch, G., Scholkopf, B., Kohlmorgen, J., and Vapnik, V., 1997, Predicting time series with support vector machines. Artificial neural networks-Proceedings ICANN'97, pp999-1004.

Orr, M.J.L., 1993, Regularised centre recruitment in radial basis function networks. Research report 59, Centre for Cognitive Science, Edinburgh University.

Orr, M.J.L., 1995, Regularization in the selection of radial basis function centers. Neural computation, 7, 606-623.

Scholkopf, B., Burges, C., and Vapnik, V., 1995, Extracting support data for a given task. In U.M. Fayyad and R. Uthurusamy (eds.), Proceedings, First International Conference on Knowledge Discovery & Data Mining, AAAI Press, Menlo Park, CA, USA, pp 252-257.

Smola, A.J., and Scholkopf, B., 1998, A tutorial on support vector regression. Neuro Colt Technical Report TR-1998-031, Royal Holloway College.

Smola, A., Scholkopf, B., and Muller, K.R., 1998, General Cost Functions for Support Vector Regression. ACNN'98, Proceeding of the ninth Australian Conference on neural networks, University Queensland, Brisbane, Qld, Australia, pp 72-78.

Vanderber R.J., 1994, LOQO: An interior point code for quadratic programming. Technical Report SOR-94-15, Statistics and Operations Research, Princeton University. N.J.

Vapnik, V, 1995, The nature of Statistical Learning Theory. (Springer Verlag).