Communicated by John Platt

Regularization in the Selection of Radial Basis Function Centers

Mark J. L. Orr
Centre for Cognitive Science, University of Edinburgh,
2, Buccleuch Place, Edinburgh EH8 9LW, UK

Subset selection and regularization are two well-known techniques that can improve the generalization performance of nonparametric linear regression estimators, such as radial basis function networks. This paper examines regularized forward selection (RFS)—a combination of forward subset selection and zero-order regularization. An efficient implementation of RFS into which either delete-1 or generalized cross-validation can be incorporated and a reestimation formula for the regularization parameter are also discussed. Simulation studies are presented that demonstrate improved generalization performance due to regularization in the forward selection of radial basis function centers.

1 Introduction

In linear regression, subset selection is used to identify subsets of fixed functions of the independent variables (regressors), which can model most of the variation in the dependent variable. Finding the smallest subset that explains a given fraction of this variation is usually intractable, and suboptimal algorithms that do not search through all possible combinations of regressors are often used in practice. One of these, forward selection, starts with an empty model and then recursively adds the current most explanatory regressor to a growing subset until some criterion is met (Rawlings 1988).

Chen *et al.* (1991) used forward selection to choose the hidden units (centers) of radial basis function (RBF) networks to produce parsimonious networks. They also described an efficient implementation of forward selection, which they called orthogonal least squares (OLS). The criterion used to halt center selection was a simple threshold on the fraction of variance explained by the subset model.

If the threshold is chosen so that too much variance is explained by the chosen regressors, poor generalization performance (overfit) results. To avoid this, Orr (1993) introduced regularized forward selection (RFS) in which high hidden-to-output weights are penalized by using zero-order regularization (also known as ridge regression in statistics and weight-decay in neural networks). Subsequently, Chen *et al.* (1995), by penalizing

the orthogonalized weights, found an efficient implementation of RFS, which they called regularized orthogonal least squares (ROLS).

The purpose of the present paper is two-fold. Firstly, delete-1 or generalized cross-validation (GCV), both more effective ways of halting center selection than a fixed threshold on the explained variance, can easily be incorporated into OLS or ROLS. Second, if there are good a priori grounds for believing that the target function has some global smoothness property then the combination of regularization and cross-validated selection will give better generalization performance than cross-validated selection alone. In addition, a reestimation formula for the regularization parameter is derived that lets the data choose its value.

The combination of regularization and subset selection is rare but not unknown. Barron and Xiao (1991) use a derivative-based roughness penalty to avoid overfitting in the selection of subsets of polynomial regressors. Breiman (1992) used stacking to combine a number of different regression estimators of two types: one type used backward elimination (with various subset sizes) and the other type used ridge regression (with various values for the regularization parameter). The MARS algorithm (Friedman 1991) forwardly splits (then backwardly merges) spline basis functions using a GCV criterion and employs a kind of ridge regression. However, the regularization used in MARS merely fulfils a computational requirement—by fixing the regularization parameter just large enough to maintain numerical stability, the necessity for a numerically sensitive but computationally slow matrix inversion algorithm is avoided. Here, the combination of forward selection and zero-order regularization is explored with a view to improving the generalization performance of a single linear regression estimator—a radial basis function network.

Some alternative methods of building up radial basis function networks—resource-allocating networks (Platt 1991; Kadirkamanathan and Niranjan 1993) or growing cell structures (Fritzke 1994)—can be compared to RFS. In common with these methods, but unlike Moody and Darken (1989), RFS uses the output values as well as the input vectors of the training set to determine the center placement. However, in contrast, these other methods all involve adaptive centers (in position and size) and consequently some kind of gradient descent learning procedure and multiple passes through the data. In RFS the available centers are all fixed, but there is a process of selection to determine which ones are included in the network. The other methods search a continuous space (of weights, positions, and sizes) that grows in dimension as centers are added while RFS heuristically searches a discrete space of different combinations of fixed centers. Another difference is that the other approaches all involve several preset parameters and thresholds (used for adding new centers and performing gradient descent) that must be tuned to each new problem. RFS, applied to RBF networks, has only one preset parameter, the basis function width. One last difference is that the other methods are all naturally suited for on-line applications where the train-

ing data arrive sequentially in time. Although the RFS method could be adapted for that case, its fast implementation (ROLS, described below) depends, at least in its current form, on the data being available all at once.

The next section briefly reviews RBF networks, regularization, and forward selection and Section 3 reviews OLS and ROLS, algorithms that efficiently implement unregularized and regularized forward selection. Section 4 shows how cross-validation can be integrated into OLS and ROLS and Section 5 derives a reestimation formula for the regularization parameter. Section 6 reports the results of some simulation studies and the final section presents conclusions.

2 Regularization and Forward Selection

The general linear model has the form

$$y = f(\mathbf{x}) = \sum_{j=1}^{m} w_j h_j(\mathbf{x})$$

where the regressors, $\{h_j(\cdot)\}_1^m$, are fixed functions of the input, $\mathbf{x} \in \Re^n$, and only the coefficients, $\{w_j\}_1^m$, are unknown. The output, $y \in \Re$, is assumed to be scalar, for simplicity. To perform linear regression with this model on the training set $\{(\mathbf{x}_i, y_i)\}_1^p$ the system of equations

$$\mathbf{y} = \mathbf{H}\mathbf{w} + \mathbf{e}$$

is solved. The $p \times m$ elements of the design matrix $\mathbf{H}$ are the responses of the m regressors to the p inputs of the training set, $\mathbf{y} = [y_1 \ldots y_p]^\top$ is the p-dimensional vector of training set outputs, and the vector $\mathbf{e}$ contains p unknown errors between these (measured) outputs and their true values. The goal is to find the best linear combination of the columns of $\mathbf{H}$ (i.e., the best value for $\mathbf{w}$) to explain $\mathbf{y}$ according to some criterion. The normal criterion is minimization of the sum of squared errors,

$$E = \mathbf{e}^\top \mathbf{e}$$

in which case the solution is

$$\mathbf{w} = (\mathbf{H}^\top \mathbf{H})^{-1} \mathbf{H}^\top \mathbf{y} \tag{2.1}$$

In radial basis function networks (Broomhead and Lowe 1988) the regressors are distinguished by a set of points (centers) $\{\mathbf{c}_j\}_1^m$ in the input space and a set of scale factors (radii) $\{r_j\}_1^m$ such that

$$h_j(\mathbf{x}) = \phi\left(\frac{\| \mathbf{x} - \mathbf{c}_j \|}{r_j}\right) \tag{2.2}$$

where $\phi(\cdot)$ is a nonlinear function that monotonically decreases as its argument increases from zero, for example the gaussian function $\phi(z) = \exp(-z^2)$. Each regressor is associated with a hidden unit in a feedforward architecture with a single hidden layer and the coefficients $\{w_j\}_1^m$ are the weights from the hidden units to the output unit. The radii can be kept constant ($r_j = r$, $1 \leq j \leq m$), since such networks, even thus restricted, are still universal approximators (Park and Sandberg 1991). The fixed radius r can be set by some heuristic (Moody and Darken 1989). About half the maximum distance separating pairs of input training points often gives good results, I find. The components of multidimensional ($n > 1$) inputs, which may have widely different variances in the training set, should be rescaled to all have the same (e.g., unit) variance or, equivalently, an appropriate non-Euclidean metric should be employed in 2.2.

If too many centers are used the large number of free parameters available in the regression will cause the network to be oversensitive to the details of the particular training set and result in poor generalization performance (overfit). An extreme case is if the set of centers is chosen to be the set of training inputs ($\mathbf{c}_i = \mathbf{x}_i$, $1 \leq i \leq p$) in which case $\mathbf{H}$ is square of dimension p (I will call this the full design matrix and denote it by $\mathbf{F}$). Then the normal equation 2.1 results in strict interpolation in which the training set is exactly reproduced by the network (Broomhead and Lowe 1988).

There are two main ways to avoid overfit. The first, regularization (Tikhonov and Arsenin 1977; Bishop 1991), reduces the "number of good parameter measurements" (MacKay 1992) in a large model (e.g., the full model) by adding a weight penalty term to the minimization criterion. For example, minimization of the energy

$$E = \mathbf{e}^\top \mathbf{e} + \lambda \mathbf{w}^\top \mathbf{w}$$

is zero-order regularization (Press *et al.* 1992), or ridge regression as it is known in statistics (Hocking 1983), and results in the solution

$$\mathbf{w} = (\mathbf{F}^\top \mathbf{F} + \lambda \mathbf{I}_p)^{-1} \mathbf{F}^\top \mathbf{y}$$

(where $\mathbf{I}_p$ is the $p \times p$ identity matrix). The regularization parameter, λ, has to be chosen a priori or estimated from the data (see Section 5).

The second way to avoid overfit is to explicitly limit the complexity of the network by allowing only a subset of the possible centers to participate. This method has the added advantage of producing parsimonious networks. Broomhead and Lowe (1988) suggested choosing such a subset randomly from the training inputs. However, a better approach is to choose the subset that best explains the variation in the dependent variable, and this is what the subset selection methods of regression analysis are for (Rawlings 1988). If forward selection is used, centers are picked one at a time from some large set (e.g., a regular array covering the sample space, or all the training set inputs) and added to an initially empty

subset model until some criterion is met. Regularized forward selection is formulated as follows (the unregularized formulation can be obtained by setting $\lambda = 0$ throughout).

At the mth step the old design matrix, $\mathbf{H}_{m-1}$, is augmented by a new column,

$$\mathbf{H}_m = [\mathbf{H}_{m-1}\ \mathbf{f}_i]$$

where $\mathbf{f}_i$ is chosen from the columns of the full design matrix $\mathbf{F}$. After including the new column in the subset and finding the regularized weight

$$\mathbf{w}_m = (\mathbf{H}_m^\top \mathbf{H}_m + \lambda \mathbf{I}_m)^{-1} \mathbf{H}_m^\top \mathbf{y}$$

the minimized energy is

$$\begin{aligned} E_m^{(i)} &= \mathbf{e}_m^\top \mathbf{e}_m + \lambda \mathbf{w}_m^\top \mathbf{w}_m \\ &= \mathbf{y}^\top \mathbf{P}_m \mathbf{y} \end{aligned}$$

where

$$\mathbf{P}_m = \mathbf{I}_p - \mathbf{H}_m (\mathbf{H}_m^\top \mathbf{H}_m + \lambda \mathbf{I}_m)^{-1} \mathbf{H}_m^\top$$

When $\lambda = 0$, $\mathbf{P}_m$ is a projection matrix projecting p-dimensional vectors perpendicular to the space spanned by the columns of $\mathbf{H}_m$. The criterion used to select the best column (center) from $\mathbf{F}$ is the constraint

$$E_m^{(i)} \le E_m^{(j)}, \qquad 1 \le j \le p$$

which is equivalent (Orr 1993) to selecting $\mathbf{f}_i$ to maximize

$$E_{m-1} - E_m^{(i)} = \frac{(\mathbf{y}^\top \mathbf{P}_{m-1} \mathbf{f}_i)^2}{\lambda + \mathbf{f}_i^\top \mathbf{P}_{m-1} \mathbf{f}_i} \tag{2.3}$$

Once the best column is chosen from among the $\{\mathbf{f}_i\}_1^p$ it is appended to the previously chosen columns to become $\mathbf{h}_m$, the last column of $\mathbf{H}_m$.

3 Fast Regularized Forward Selection

The equations above are amenable to an orthogonal implementation that speeds up the computations (Chen *et al.* 1991; 1995). The design matrix is factored into

$$\mathbf{H}_m = \tilde{\mathbf{H}}_m \mathbf{U}_m$$

where the $p \times m$ matrix $\tilde{\mathbf{H}}_m$ has orthogonal columns, $\{\tilde{\mathbf{h}}_j\}_1^m$, and the square m-dimensional matrix $\mathbf{U}_m$ is upper triangular. Then the regression problem is in the form

$$\mathbf{y} = \tilde{\mathbf{H}}_m \tilde{\mathbf{w}}_m + \mathbf{e}_m$$

where

$$\tilde{\mathbf{w}}_m = \mathbf{U}_m \mathbf{w}_m \tag{3.1}$$

At the mth step $\tilde{\mathbf{H}}_{m-1}$ is augmented by a new column that is orthogonal to each of its $m-1$ existing and already orthogonal columns,

$$\tilde{\mathbf{H}}_m = [\tilde{\mathbf{H}}_{m-1}\ \tilde{\mathbf{f}}_i]$$

where

$$\tilde{\mathbf{f}}_i = \mathbf{f}_i - \sum_{j=1}^{m-1} \frac{\tilde{\mathbf{f}}_i^\top \tilde{\mathbf{h}}_j}{\tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j} \tilde{\mathbf{h}}_j \tag{3.2}$$

and $\mathbf{f}_i$ is selected from the columns of $\mathbf{F}$. In the case of a regularized network ($\lambda > 0$) orthogonalization is possible only if the roughness penalty term depends on the orthogonalized weights, $\tilde{\mathbf{w}}_m$, and not the ordinary weights, $\mathbf{w}_m$ (Chen *et al.* 1995). Then the minimized energy is

$$\begin{aligned} \tilde{E}_m^{(i)} &= \mathbf{e}_m^\top \mathbf{e}_m + \lambda \tilde{\mathbf{w}}_m^\top \tilde{\mathbf{w}}_m \\ &= \mathbf{y}^\top \tilde{\mathbf{P}}_m \mathbf{y} \end{aligned}$$

where

$$\begin{aligned} \tilde{\mathbf{P}}_m &= \mathbf{I}_p - \tilde{\mathbf{H}}_m (\tilde{\mathbf{H}}_m^\top \tilde{\mathbf{H}}_m + \lambda \mathbf{I}_m)^{-1} \tilde{\mathbf{H}}_m^\top \\ &= \mathbf{I}_p - \sum_{j=1}^{m-1} \frac{\tilde{\mathbf{h}}_j \tilde{\mathbf{h}}_j^\top}{\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j} - \frac{\tilde{\mathbf{f}}_i \tilde{\mathbf{f}}_i^\top}{\lambda + \tilde{\mathbf{f}}_i^\top \tilde{\mathbf{f}}_i} \\ &= \tilde{\mathbf{P}}_{m-1} - \frac{\tilde{\mathbf{f}}_i \tilde{\mathbf{f}}_i^\top}{\lambda + \tilde{\mathbf{f}}_i^\top \tilde{\mathbf{f}}_i} \end{aligned}$$

The selected $\tilde{\mathbf{f}}_i$ is the one that maximizes

$$\tilde{E}_{m-1} - \tilde{E}_m^{(i)} = \frac{(\mathbf{y}^\top \tilde{\mathbf{f}}_i)^2}{\lambda + \tilde{\mathbf{f}}_i^\top \tilde{\mathbf{f}}_i} \tag{3.3}$$

and it becomes $\tilde{\mathbf{h}}_m$, the last column of $\tilde{\mathbf{H}}_m$.

A more efficient alternative to orthogonalizing each $\mathbf{f}_i$, $1 \le i \le p$, at each step using 3.2 is to recursively compute the matrix

$$\tilde{\mathbf{F}}_m = \tilde{\mathbf{F}}_{m-1} - \frac{\tilde{\mathbf{h}}_m \tilde{\mathbf{h}}_m^\top \tilde{\mathbf{F}}_{m-1}}{\tilde{\mathbf{h}}_m^\top \tilde{\mathbf{h}}_m} \tag{3.4}$$

(with $\tilde{\mathbf{F}}_0 = \mathbf{F}$ initially) whose columns are precisely the $\{\tilde{\mathbf{f}}_i\}_1^p$ from which the selection at step $m+1$ is made.

To recover the unregularized weight vector $\mathbf{w}_m$ at the end note that the components of the regularized weight vector $\tilde{\mathbf{w}}_m$ are given by

$$(\tilde{\mathbf{w}}_m)_j = \frac{\mathbf{y}^\top \tilde{\mathbf{h}}_j}{\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j}, \qquad 1 \le j \le m$$

Then 3.1 can be used to obtain

$$\mathbf{w}_m = \mathbf{U}_m^{-1}\tilde{\mathbf{w}}_m$$

an easy inversion since $\mathbf{U}_m$ is triangular. $\mathbf{U}_m$ can be recursively computed as

$$\mathbf{U}_m = \begin{bmatrix} \mathbf{U}_{m-1} & (\tilde{\mathbf{H}}_{m-1}^\top \tilde{\mathbf{H}}_{m-1})^{-1}\tilde{\mathbf{H}}_{m-1}^\top \mathbf{f}_i \\ \mathbf{0}_{m-1}^\top & 1 \end{bmatrix} \tag{3.5}$$

(with $\mathbf{U}_1 = 1$ initially). Note that the matrix $\tilde{\mathbf{H}}_{m-1}^\top \tilde{\mathbf{H}}_{m-1}$ is diagonal.

The efficiency of the orthogonalization scheme derives from the relative ease of computing 3.3 instead of 2.3, even with the overheads of 3.4 and 3.5. The computational cost (number of floating point operations) required to select one center from a pool of size M with p patterns in the training set is, to first order, proportional to Mp with orthogonalization. Without orthogonalization, the cost is roughly proportional to Mp^2. If the input training points are used as the pool of selectable centers then $M = p$ and the corresponding costs are p^2 and p^3, respectively.

4 Cross-Validation

The previous two sections described an algorithm for making selections but without mentioning criteria for halting the selection process. In Chen *et al.* (1991) a simple fixed threshold on the fraction of unexplained variance was used so that the last center was selected as soon as the condition

$$E_m < \xi\, \mathbf{y}^\top \mathbf{y}, \qquad 0 < \xi < 1 \tag{4.1}$$

became true for some prechosen threshold ξ. In the $\lambda = 0$ case, a possible choice for ξ is

$$\xi = \frac{p\,\sigma^2}{\mathbf{y}^\top \mathbf{y}} \tag{4.2}$$

where σ^2 is the noise variance on the training outputs. However, such a fixed threshold is liable to result in overfit. This is illustrated in Figure 1, which shows fits to some noisy data from a sine wave, the same example as used in Chen *et al.* (1995). The training data are shown in Figure 1a and the forward selection fit using a fixed threshold is in Figure 1b.

One way to avoid overfit is to regularize, as in RFS (Orr 1993). Alternatively, criteria other than 4.1 can be used that are designed to halt center selection before the onset of overfit. For example, the fact that mean square residual error,

$$\mathrm{MSRE}_m = \frac{E_m}{p - m}$$

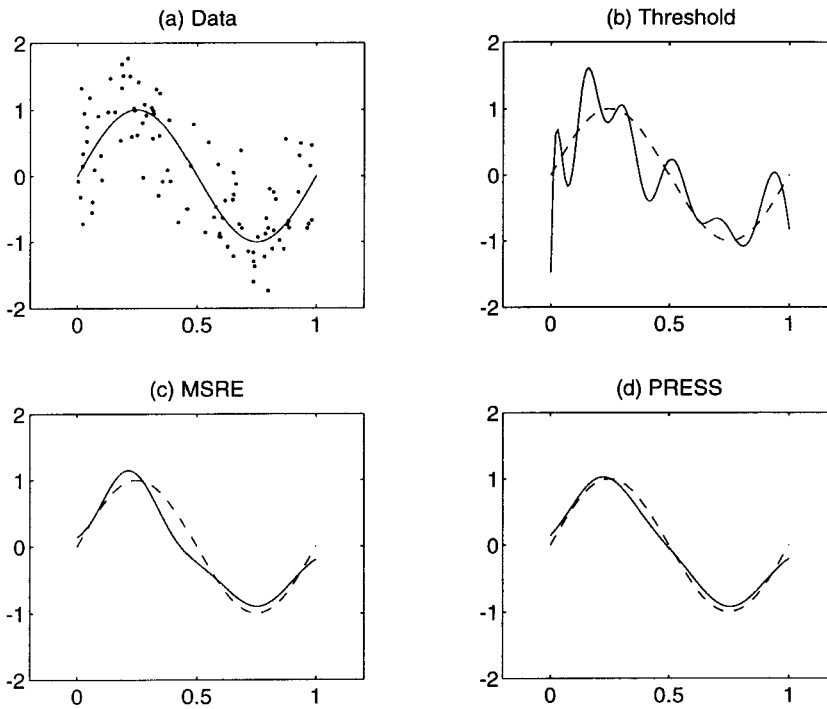

Figure 1: Forward selection fits to data taken from a sine wave. (a) The training data consist of $p = 100$ points randomly sampled from part of a sine wave (solid curve) in $[0\ 1]$ and corrupted with noise of standard deviation $\sigma = 0.5$. An RBF network with gaussian basis functions of width $r = 0.2$ centered on the training inputs is grown by OLS forward selection using various halting criteria. (b) Fifteen centers were selected before the unexplained variance reduced below the fixed threshold (4.2) and the result is overfit (solid line). (c) MSRE stopped decreasing after only five centers and the fit shows much better generalization (solid curve) but still slight signs of overfit. (d) PRESS (and also GCV) stopped selection after only three centers and produced the best fit.

is an estimate of the noise variance when $\lambda = 0$ and the correct subset size has been reached (Rawlings 1988) can be exploited. The obvious choice is the moving threshold $\xi = (p-m)\,\sigma^2/\mathbf{y}^\top\mathbf{y}$ but the mean square residual error may never reduce below the noise variance and this threshold may never be crossed. It is better to detect when MSRE stops decreasing, and this does not need prior knowledge of σ^2. Actually this method is not much of an improvement on using a fixed threshold (see Section 6)

although it happened to perform better on the example in Figure 1 (see Fig. 1c).

For the case $\lambda > 0$, E_m is more than just the residual square error since it contains a weight penalty component. If MSRE is used to halt center selection in RFS the correct formula to use is

$$\text{MSRE}_m = \frac{\mathbf{y}^\top \tilde{\mathbf{P}}_m^2 \mathbf{y}}{p - m}$$

where

$$\tilde{\mathbf{P}}_m = \mathbf{I}_p - \sum_{j=1}^{m} \frac{\tilde{\mathbf{h}}_j \tilde{\mathbf{h}}_j^\top}{\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j} \tag{4.3}$$

Other criteria for choosing subset size are described in Hocking (1983). One general method is cross-validation, which has a number of variations, two of which are delete-1 cross-validation (Allen 1974; Stone 1974) and generalized cross-validation (Golub *et al.* 1979). In delete-1 cross-validation (or predicted sum of squares, PRESS) generalization performance is measured by the average (over all training examples) of the squared prediction error when the network is tested on one example and trained on the remainder. If $f_m^{(i)}(\cdot)$ is the network output (at selection step m) when trained on all but the ith training example, the average predicted sum of squares is

$$\text{PRESS}_m = \frac{1}{p} \sum_{i=1}^{p} [f_m^{(i)}(\mathbf{x}_i) - y_i]^2 \tag{4.4}$$

Good generalization performance is associated with low values of PRESS so in forward selection the subset size is determined by the point at which this measure reaches a minimum. For nonlinear regression problems with long training times, e.g., multilayer perceptrons trained with back-propagation, delete-1 cross-validation is too expensive to compute, but for linear systems, such as RBF networks, it can be derived analytically (Golub *et al.* 1979) as

$$\text{PRESS}_m = \frac{1}{p} \parallel [\text{diag}(\tilde{\mathbf{P}}_m)]^{-1} \tilde{\mathbf{P}}_m \mathbf{y} \parallel^2$$

where $\text{diag}(\cdot)$ denotes the matrix obtained by zeroing all off-diagonal terms. In ROLS the expansion of $\tilde{\mathbf{P}}_m$ in terms of orthogonal vectors 4.3 allows PRESS to be computed very efficiently. $\tilde{\mathbf{P}}_m \mathbf{y}$ and $\text{diag}(\tilde{\mathbf{P}}_m)$ can be recursively updated at each step and the product of $[\text{diag}(\tilde{\mathbf{P}}_m)]^{-1}$ and $\tilde{\mathbf{P}}_m \mathbf{y}$ is equivalent to a mere element-by-element division of two p-dimensional vectors.

Generalized cross-validation, given by

$$\text{GCV}_m = \frac{1}{p} \frac{\parallel \tilde{\mathbf{P}}_m \mathbf{y} \parallel^2}{\left[(1/p)\,\text{trace}(\tilde{\mathbf{P}}_m)\right]^2} \tag{4.5}$$

is similar to the delete-1 form but the average over the diagonal elements of $\tilde{\mathbf{P}}_m$ makes it even easier to compute than PRESS since the scalar quantities $\| \tilde{\mathbf{P}}_m \mathbf{y} \|^2$ and $\text{trace}(\tilde{\mathbf{P}}_m)$ can both be computed recursively. PRESS and GCV tend to choose similar subset sizes and are both much better at avoiding overfit than a fixed threshold or MSRE (as is shown in Section 6). Figure 1d shows the fit to the example training set using PRESS to halt the selection of radial basis function centers.

5 Automatic Estimation of λ

As is shown in the next section, while cross-validation scores such as PRESS and GCV are certainly good criteria for avoiding overfit, using regularization as well further decreases the likelihood of overfit. First, however, a simple reestimation formula is derived that can be integrated into ROLS for letting the data choose a value for the regularization parameter, λ. The formula is based on GCV minimization, like Gu and Wahba (1991), except they used the Newton method. An alternative reestimation formula results from maximizing Bayesian evidence (MacKay 1992).

Differentiating 4.5 with respect to λ and setting the result to zero gives a minimum when

$$\mathbf{y}^\top \tilde{\mathbf{P}}_m \frac{\partial \tilde{\mathbf{P}}_m \mathbf{y}}{\partial \lambda} \text{trace}(\tilde{\mathbf{P}}_m) = \mathbf{y}^\top \tilde{\mathbf{P}}_m^2 \mathbf{y} \frac{\partial \text{trace}(\tilde{\mathbf{P}}_m)}{\partial \lambda} \tag{5.1}$$

However, from 4.3 it can be shown that

$$\mathbf{y}^\top \tilde{\mathbf{P}}_m \frac{\partial \tilde{\mathbf{P}}_m \mathbf{y}}{\partial \lambda} = \lambda \tilde{\mathbf{w}}_m^\top (\tilde{\mathbf{H}}_m^\top \tilde{\mathbf{H}}_m + \lambda \mathbf{I}_m)^{-1} \tilde{\mathbf{w}}_m$$

This result allows 5.1 to be rearranged into the reestimation formula

$$\lambda := \frac{[\partial \text{trace}(\tilde{\mathbf{P}}_m)/\partial \lambda] \, \mathbf{y}^\top \tilde{\mathbf{P}}_m^2 \mathbf{y}}{\text{trace}(\tilde{\mathbf{P}}_m) \, \tilde{\mathbf{w}}_m^\top (\tilde{\mathbf{H}}_m^\top \tilde{\mathbf{H}}_m + \lambda \mathbf{I}_m)^{-1} \tilde{\mathbf{w}}_m} \tag{5.2}$$

where

$$\frac{\partial \text{trace}(\tilde{\mathbf{P}}_m)}{\partial \lambda} = \sum_{j=1}^{m} \frac{\tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j}{(\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j)^2} \tag{5.3}$$

$$\mathbf{y}^\top \tilde{\mathbf{P}}_m^2 \mathbf{y} = \mathbf{y}^\top \mathbf{y} - \sum_{j=1}^{m} \frac{(2\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j)(\mathbf{y}^\top \tilde{\mathbf{h}}_j)^2}{(\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j)^2} \tag{5.4}$$

$$\text{trace}(\tilde{\mathbf{P}}_m) = p - \sum_{j=1}^{m} \frac{\tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j}{\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j} \tag{5.5}$$

$$\tilde{\mathbf{w}}_m^\top (\tilde{\mathbf{H}}_m^\top \tilde{\mathbf{H}}_m + \lambda \mathbf{I})^{-1} \tilde{\mathbf{w}}_m = \sum_{j=1}^{m} \frac{(\mathbf{y}^\top \tilde{\mathbf{h}}_j)^2}{(\lambda + \tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j)^3} \tag{5.6}$$

A new value of λ can be reestimated after each forward selection step by using the previous value in the right-hand side of 5.2, initializing $\lambda = 0$ prior to the first step. Equations 5.3–5.6 are not fully recursive since the value of λ changes after each step. In other words, each term in each summation must be recomputed at each step (instead of just the last term if λ had been fixed). However, the extra computation this involves is proportional only to m (assuming the results of $\{\tilde{\mathbf{h}}_j^\top \tilde{\mathbf{h}}_j\}_1^m$ and $\{(\mathbf{y}^\top \tilde{\mathbf{h}}_j)^2\}_1^m$ are cached) and is thus negligible compared to the complexity of the whole algorithm (which is proportional to Mp operations per selected center where $M, p > m$—see Section 3).

6 Simulation Studies

In this section RFS is applied to two simulated learning problems. The first involves a one-dimensional Hermite polynomial and is used to compare ordinary forward selection (using the OLS algorithm and various halting criteria) to RFS (using ROLS, a GCV criterion and λ reestimation) and then to compare RFS to an alternative method of building RBF networks, the RAN-EKF algorithm (Kadirkamanathan and Niranjan 1993). The second is a multivariate problem with data from a simulated alternating current series circuit and is used to compare RFS with the MARS algorithm (Friedman 1991), which is based on recursive splitting of spline basis functions.

In the first problem the target function is the Hermite polynomial,

$$f(x) = 1.1\,(1 - x + 2\,x^2)\,\exp\left(-\frac{x^2}{2}\right)$$

from which are taken noisy samples. Figure 2 shows a typical data set and some fits. To properly assess the performance of the different selection methods 1000 training sets were generated each with different inputs $\{x_i\}_1^p$ (sampled uniformly from the range $[-4, 4]$) and errors $\{e_i\}_1^p$ (sampled from the same normal distribution). Each algorithm used Cauchy basis functions $[\phi(z) = 1/(1+z^2)]$ with a radius of $r = 1.5$ and drew centers from a set of 100 equally spaced points in the interval $[-5, 5]$. Ordinary forward selection (implemented by OLS) with three different halting criteria, (1) a fixed threshold on the unexplained variance, (2) minimization of MSRE, (3) minimization of PRESS, were compared with (4) regularized forward selection, implemented by ROLS, using a GCV criterion and λ reestimation.

The results are shown in the four plots, one for each algorithm, of Figure 3. The plots display data error (horizontal axis) against fit error (vertical axis) and there is a point in each plot for every training set. The data error is the root mean square error between the data and the true function (i.e., $\sqrt{(1/p)\sum_1^p e_i^2}$) and is concentrated around the value $\sigma = 0.5$—the size of normal distribution from which the errors were

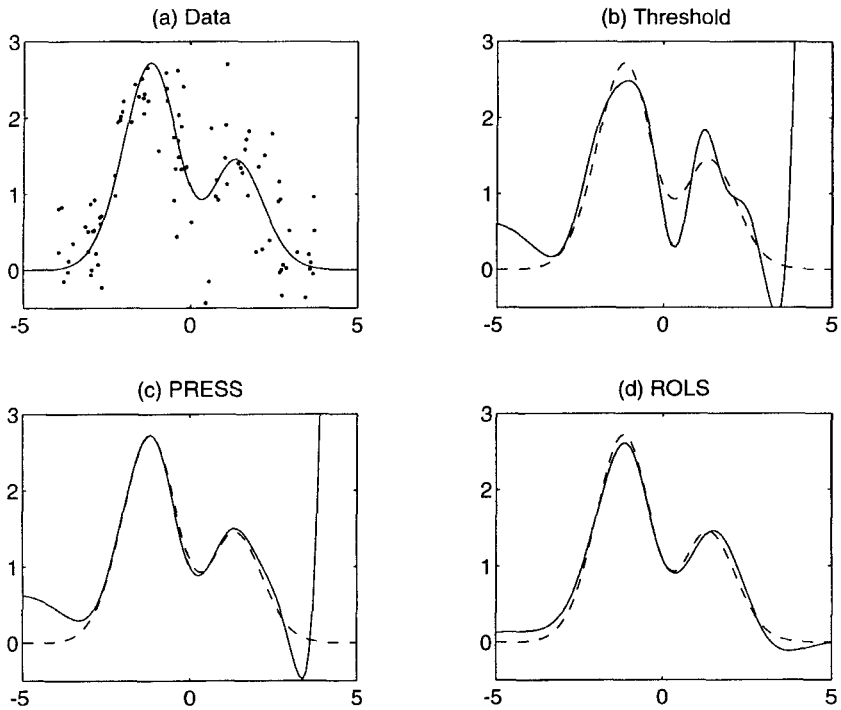

Figure 2: One of the Hermite polynomial training sets. (a) The true function (solid curve) is sampled at $p = 100$ random positions in the range $[-4, 4]$ and gaussian noise of standard deviation $\sigma = 0.5$ is added. OLS-PRESS performed worse on this training set, as measured by fit error, than on 999 other similar sets (see Fig. 3). (b) The poor fit produced by OLS with a fixed threshold on variance. (c) The OLS-PRESS fit with overfitting near the edges of the sample space. (d) The RFS fit achieved by the ROLS algorithm with a GCV criterion and λ reestimation.

sampled. The fit error is the root mean square error between the fit and the true function over a set of 100 equally spaced points in the same range from which the training inputs were sampled. It objectively measures how well a particular algorithm generalizes from a particular training set, but of course, like the data error, is realizable only in synthetic examples such as this where the true target function is known.

As can be seen from Figure 3a, unregularized forward selection with a fixed threshold can lead to extremely bad generalization (note the logarithmic scale of the vertical axis) if the training set contains an above

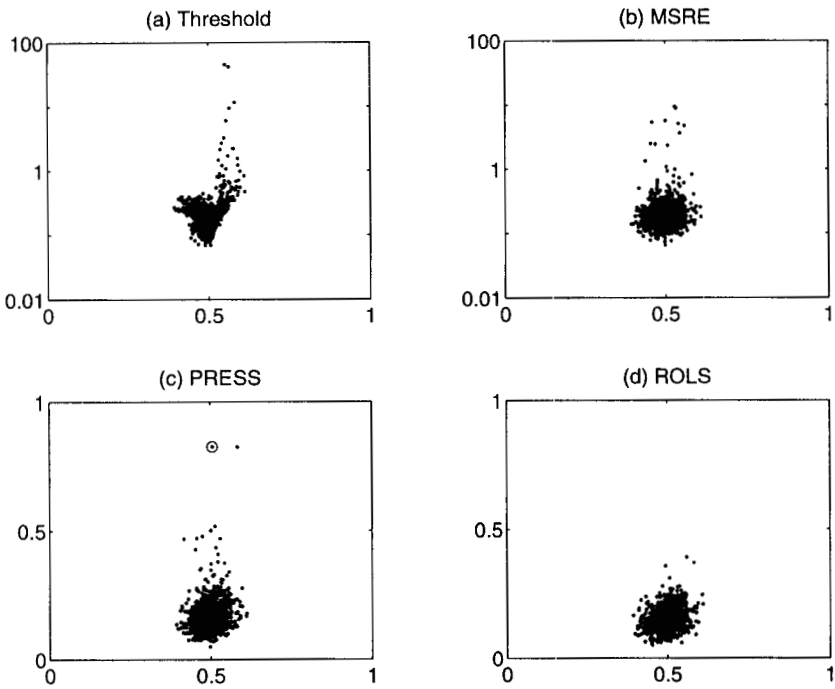

Figure 3: Plots of data error (horizontal axis) versus fit error (vertical axis) for 1000 training sets (similar to the one shown in Fig. 2a) and four fitting algorithms: (a) OLS with a fixed threshold on variance, (b) OLS-MSRE, (c) OLS-PRESS, and (d) ROLS-GCV with λ reestimation. Logarithmic scales have been used on the vertical axes in (a) and (b) to embrace the larger dynamic range of the OLS-threshold and OLS-MSRE fit errors. The ringed point in (c), the worst OLS-PRESS fit, corresponds to the training data used in Figure 2.

average data error. The algorithm accommodates the extra noise in the training set by selecting extra centers, which cause overfit. Halting the selection of centers after MSRE has stopped decreasing (Fig. 3b) is slightly better and appears to be relatively indifferent to the size of the data error but, like the fixed threshold, produces many very bad fits. In contrast, using PRESS gives much improved performance (Fig. 3c). Similar results are obtained with GCV. Finally, using regularized forward selection and GCV for both halting selection and λ reestimation (Fig. 3d) shows still further improvement over the unregularized algorithm.

However, examination of the handful of training sets with fit errors above about 0.4 in Figure 3c revealed that these, the poorest fits, had

resulted from overfitting close to the edges of the area of input space from which the training inputs were drawn (the sample space). The training set used in Figure 2, which is the one upon which OLS-PRESS performed least well and corresponds to the ringed point in Figure 3c, illustrates this. Absence of training points or chance alignments between training points near the extremes of the sample space can cause local overfitting (Fig. 2a and c) since cross-validation is dependent on the presence of data to constrain the fit. As is well known from the Bayesian interpretation of regularization (MacKay 1992), regularization provides an extra a priori constraint—the fitted function should be smooth—which allows the fit to extrapolate gracefully across the edges of the sample space (Fig. 2d). Missing data in interior regions of the sample space would also produce opportunities for local overfitting that regularization could ameliorate (Orr 1993).

RFS results on the Hermite polynomial function were compared to those of the RAN-EKF algorithm (Kadirkamanathan and Niranjan 1993), which adapts the positions and sizes of existing basis functions as well as adding new ones. The same number of training examples ($p = 40$), the same test set (200 uniformly spaced noiseless samples in the range $[-4, 4]$), and the same type of radial basis functions (gaussians) were used as in their study. The results were averaged over 100 runs for each of several different noise levels in the training data. The randomly chosen 40 training set inputs in each run were used as the centers of the basis functions (of radius $r = 1.5$) from which each network was built. Figure 4 shows how the average over 100 runs of the number of selected centers and the root mean square error (of the fit over the test set) varies with noise variance. The RAN-EKF data have been read off Figure 4 of Kadirkamanathan and Niranjan (1993). The number of centers chosen by RFS tends to drop as the noise level increases (the opposite trend to RAN-EKF, see Fig. 4a), and the RFS fit error is consistently smaller than that of RAN-EKF (Fig. 4b). These results suggest that RFS is more accurate than RAN-EKF and better at producing parsimonious networks.

RFS was also applied to a problem from Friedman (1991) involving data from a simulated alternating current series circuit where the input vectors come from a four-dimensional space,

$$\mathbf{x} = [R \ \omega \ L \ C]^\top$$

with resistance (R ohms), angular frequency (ω radians per second), inductance (L henries), and capacitance (C farads) in the ranges

$$\begin{aligned} 0 &\leq R \leq 100 \\ 40\pi &\leq \omega \leq 560\pi \\ 0 &\leq L \leq 1 \\ 1 \times 10^{-6} &\leq C \leq 11 \times 10^{-6} \end{aligned}$$

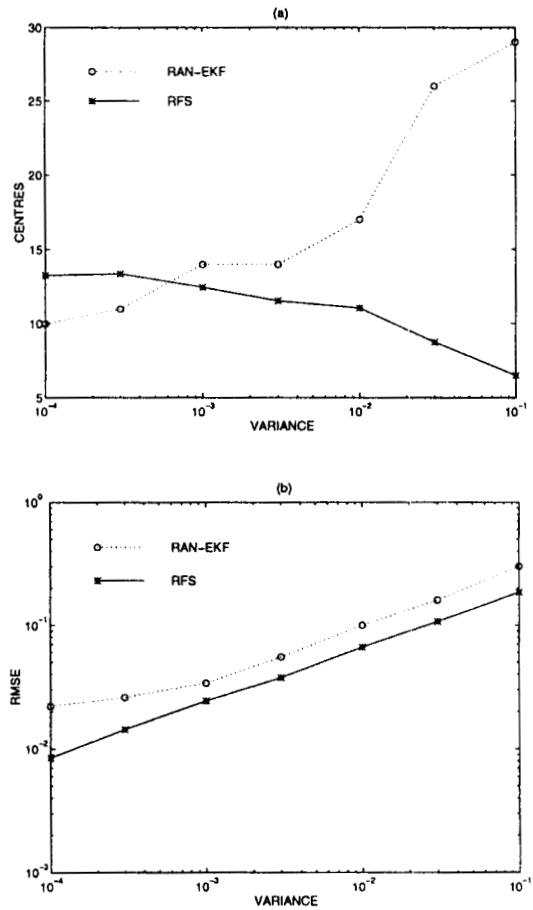

Figure 4: (a) The number of selected centers and (b) the fit error as a function of noise level for RFS (averaged over 100 runs) and the RAN-EKF algorithm.

The two dependent variables are impedance (Z ohms) and phase (ϕ radians), given by

$$Z(\mathbf{x}) = \sqrt{R^2 + (\omega L - 1/\omega C)^2} \tag{6.1}$$

$$\phi(\mathbf{x}) = \tan^{-1}\left(\frac{\omega L - 1/\omega C}{R}\right) \tag{6.2}$$

Following the procedure in Friedman (1991) as closely as possible, training sets of various sizes ($p = 100, 200, 400$) were replicated 100 times each.

Table 1: The Average (over 100 Runs) Scaled Mean Square Error of the RFS and MARS Fits to the Impedance (Z) and Phase (ϕ) Data for Different Training Set Sizes (p).

	Z		ϕ	
p	RFS	MARS	RFS	MARS
100	0.45	0.28	0.26	0.24
200	0.26	0.12	0.20	0.16
400	0.14	0.07	0.16	0.12

The p random input vectors of each set were drawn randomly from the above ranges and gaussian errors of size $\sigma_Z = 175$ and $\sigma_\phi = 0.44$ (to give 3/1 signal-to-noise ratios) were added to the corresponding p impedance and phase values. All four components of the input vectors were standardized (to have zero mean and unit variance) before RFS was applied. The pool of selectable centers was set to be the p standardized inputs of the training set and gaussian basis functions were used. The fixed radius was set at $r = 3.5$, which is about half the maximum distance between any two standardized input points in the four-dimensional space ($\approx 2\sqrt{3}$). The two sets of data (impedance and phase) were processed separately and the quality of each fit determined by mean square error (MSE) scaled by the variance of the function,

$$\text{MSE} = \frac{\sum_{k=1}^{N} \left[f(\mathbf{x}_k) - \hat{f}(\tilde{\mathbf{x}}_k)\right]^2}{\sum_{k=1}^{N} \left[f(\mathbf{x}_k) - \bar{f}\right]^2} \tag{6.3}$$

where $f(\cdot)$ is the true function [either $Z(\cdot)$ or $\phi(\cdot)$] with mean $\bar{f}$ over the randomly chosen test inputs, $\{\mathbf{x}_k\}_1^N$ (with $N = 5000$), $\hat{f}(\cdot)$ is the RFS fit trained on data with standardized inputs and $\{\tilde{\mathbf{x}}_k\}_1^N$ are standardized versions of the test inputs.

Table 1 shows average (over 100 replications) MSE values for the different training set sizes with corresponding figures for the MARS algorithm. The latter were read from Tables 9 and 11 of Friedman (1991) from the rows pertaining to $mi = 2$ (the best value, for this problem, of the MARS interaction parameter) and the columns labelled ISE.[1] As can be seen from the table, MARS is much more accurate than RFS for the impedance data (by about a factor of 2 in MSE) and slightly better for the phase data. Further investigations are required to fully explain the difference between the two methods.

[1]Friedman (1991) claimed to have calculated a Monte Carlo approximation to (scaled) integrated square error (ISE), which is given by 6.3 times a factor V (the volume of the unscaled sample space—in this case 1.63). However, as communicated to me privately by Friedman, the factor V was omitted, which means he was really calculating (scaled) mean square error (MSE) as given by 6.3.

7 Conclusions

Zero-order regularization (with automatic estimation of the regularization parameter) along with either delete-1 or generalized cross-validation can be incorporated into an efficient algorithm for performing regularized forward selection (RFS) of linear regressors, such as the centers of RBF networks. While cross-validation alone is an effective method for limiting the number of selected centers to avoid overfit, the experimental evidence supports the additional use of regularization to further reduce overfit. The extra information about the target function implicit in regularization (namely, that it has some degree of smoothness) improves generalization performance, particularly in areas of the sample space, such as the edges, where training data are sparse.

In tests, RFS performed better (in terms of accuracy and network size) than RAN-EKF (an alternative technique for constructing RBF networks) on a simple one-dimensional problem but proved less accurate than MARS (a recursive splitting algorithm using splines) on a more complex multivariate problem.

Acknowledgments

I thank Sheng Chen, Roy Hougen, Warren Sarle, and two anonymous referees for useful comments and references. This work was supported by Grant RR21748 from the U.K. Joint Councils Initiative in Human Computer Interaction and Cognitive Science.

References

Allen, D. M. 1974. The relationship between variable selection and data augmentation and a method for prediction. *Technometrics* **16**(1), 125–127.

Barron, A. R., and Xiao, X. 1991. Discussion of "Multivariate adaptive regression splines" by J. H. Friedman. *Ann. Stat.* **19**, 67–82.

Bishop, C. 1991. Improving the generalization properties of radial basis function neural networks. *Neural Comp.* **3**(4), 579–588.

Breiman, L. 1992. *Stacked Regression.* Tech. Rep. TR-367, Department of Statistics, University of California, Berkeley.

Broomhead, D. S., and Lowe, D. 1988. Multivariate functional interpolation and adaptive networks. *Complex Syst.* **2**, 321–355.

Chen, S., Chng, E. S., and Alkadhimi, K. 1995. Regularised orthogonal least squares algorithm for constructing radial basis function networks. *International Journal of Control*, submitted.

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

Friedman, J. H. 1991. Multivariate adaptive regression splines (with discussion). *Ann. Stat.* **19**, 1–141.

Fritzke, B. 1994. Supervised learning with growing cell structures. In *Advances in Neural Information Processing Systems 6*, J. D. Cowan, G. Tesauro, and J. Alspector, eds., pp. 255–262. Morgan Kaufmann, San Mateo, CA.

Golub, G. H., Heath, M., and Wahba, G. 1979. Generalised cross-validation as a method for choosing a good ridge parameter. *Technometrics* **21**(2), 215–223.

Gu, C., and Wahba, G. 1991. Minimising GCV/GML scores with multiple smoothing parameters via the Newton method. *SIAM J. Sci. Stat. Comp.* **12**(2), 383–398.

Hocking, R. R. 1983. Developments in linear regression methodology: 1959–1982 (with discussion). *Technometrics* **25**, 219–249.

Hoerl, A. E., and Kennard, R. W. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* **12**(3), 55–67.

Kadirkamanathan, V., and Niranjan, M. 1993. A function estimation approach to sequential learning with neural networks. *Neural Comp.* **5**(6), 954–975.

MacKay, D. J. C. 1992. Bayesian interpolation. *Neural Comp.* **4**(3), 415–447.

Moody, J., and Darken, C. J. 1989. Fast learning in units of locally-tuned processing units. *Neural Comp.* **1**(2), 281–294.

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

Park, J., and Sandberg, I. W. 1991. Universal approximation using radial-basis-function networks. *Neural Comp.* **3**(2), 246–257.

Platt, J. 1991. A resource-allocating network for function interpolation. *Neural Comp.* **3**(2), 213–225.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. 1992. *Numerical Recipes in C*, 2nd ed. Cambridge University Press, Cambridge, UK.

Rawlings, J. O. 1988. *Applied Regression Analysis.* Wadsworth & Brooks/Cole, Pacific Grove, CA.

Stone, M. 1974. Cross-validation choice and the assessment of statistical predictions. *J. R. Stat. Soc. (B)* **36**, 111–147.

Tikhonov, A. N., and Arsenin, V. Y. 1977. *Solutions of Ill-Posed Problems.* Winston, Washington.

Received March 14, 1994; accepted September 13, 1994.