Monograph:
Billings, S.A. and Zheng, G.L. (1995) Radial Basis Function Network Configuration Using Mutual Information and the Orthogonal Least Squares Algorithm. UNSPECIFIED. ACSE Research Report 577 . Department of Automatic Control and Systems Engineering

Radial Basis Function Network Configuration Using Mutual Information and the Orthogonal Least Squares Algorithm

S A Billings and G L Zheng
Department of Automatic Control and Systems Engineering
University of Sheffield
Mappin Street
Sheffield
S1 3JD
UK

Research Report No 577

May 1995

Radial Basis Function Network Configuration Using Mutual Information and the Orthogonal Least Squares Algorithm

Guang. L. Zheng and Steve. A. Billings
Department of Automatic Control and Systems Engineering,
University of Sheffield, Mappin Street, Sheffield S1 4DU

March 1995

Abstract — Input nodes of neural networks are usually predetermined by using *a priori* knowledge or selected by trial and error. For example, in pattern recognition applications the input nodes are usually the given pattern features and in system identification applications the past input and output data are often used as inputs to the network. Some of the input variables may be irrelevant to the task in hand and therefore may cause a deterioration in network performance. Some may be redundant and may increase the complexity of the network and consume expensive computation time. In the present study, the mutual information between the input variables and the output of the network is used to select a suboptimal set of input variables for the network. The variables are selected according to the information content relevant to the output. Variables which have a higher mutual information with the output and lower dependence on other selected variables are used as network inputs. The algorithms are derived based on heuristics and performance is assessed by using radial basis function (RBF) networts trained with the orthogonal least squares algorithm (OLS), which selects the hidden layer nodes of the network according to the error reduction ratios on the network output. Both real and simulated data sets are used to demonstrate the effectiveness of the new algorithms.

Keywords — Radial Basis Function, Mutual Information, Input Node Selection, Hidden Node Selection, Network Structure, Pattern Recognition, System Identification.

1 Introduction

Radial basis function (**RBF**) neural networks have been studied by researchers in many diverse disciplines in recent years. It has been proved (Light, 1992; Powell, 1992) that a radial basis function network can approximate arbitrarily well any multivariate continuous function on a compact domain if a sufficient number of radial basis function units are given. A radial basis function network has a rather simple feedforward structure and network learning is very simple. The network can be configurated with one radial basis function

centre at each training data point. Thus, the complexity of the network is of the same order as the dimensionality of the training data. In practice, a network with a finite basis is often preferred. A variety of approaches for training radial basis function networks have been developed. Most of these can be divided into two stages (Moody and Darken, 1989; Chen, Billings and Grant, 1992; Kaviori and Venkata Subramanian, 1993; Xu, Krzyzak and Oja, 1993; Vogt, 1993; Zheng and Billings, 1994) *i).* learning the centres and widths in the hidden layer; *ii).* learning the connection weights from the hidden layer to the output layer. In these learning algorithms, various clustering algorithms have been used to partition the input space and the connection weights were computed based upon least squares or similar methods. The network structure however, including the input layer and the hidden layer nodes are predetermined. Learning algorithms which incorporate selection mechanisms for hidden layer nodes were developed by researchers (Chen, Billings, Cowan and Grant, 1990b; Holcomb and Morari, 1991; Lee and Rhee, 1991; Musavi, Ahmed, Chan, Faris and Hummels, 1992). In Chen's work (Chen et al., 1990b), the network was trained using an orthogonal least squares algorithm. Akaike's information criterion was used to determine the number of hidden layer nodes and an error reduction ratio was used to select the centres such that the approximation errors of the network are most effectively reduced at each selection step. The learning algorithm developed in (Holcomb and Morari, 1991) treats the radial basis functions associated with the hidden layer nodes as approximately orthogonal to each other. The network starts with one hidden layer node and additional nodes are added to the network when they are necessary. The locations of the hidden layer nodes are optimized using an optimization package. Training algorithms proposed by Lee (Lee and Rhee, 1991) and Musavi (Musavi et al., 1992) are based on supervised hierarchical clustering methods. In Lee's work (Lee and Rhee, 1991), the learning starts with one hidden layer node with a large width and creates additional nodes when they are desired. The associated widths and the locations are also changed. In Musavi's work (Musavi et al., 1992), the learning begins with a large number of nodes and merges them when possible. The associated widths and locations of the nodes are updated accordingly. All these algorithms can only select the hidden layer nodes, the input layer nodes were predetermined or selected by trial and error. In the present study, mutual information and the orthogonal least squeres algorithm are used to select both the input layer and the hidden layer nodes respectively. The algorithm selects the input layer nodes according to the information content relevant to the network output. The hidden layer nodes are then determined according to the error reduction ratios on the output.

The layout of the paper is organized as follows. Section two briefly describes the radial basis function network and its structure. Section three introduces the basic concept of mutual information and its application in input layer nodes selection. Two input nodes selection algorithms are presented in this section. Section four presents an orthogonal least squares algorithm for hidden layer node selection. Experimental results are given in section five, which demonstrates the effectiveness of the algorithms. Finally, section six is devoted to conclusions and disscusions on future research directions.

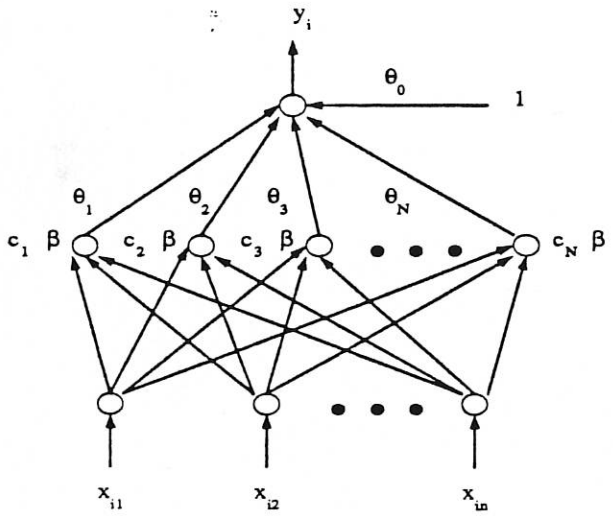


Figure 1: Radial Basis Function Network Architecture

2 Radial Basis Function Networks

A basic radial basis function (**RBF**) network may be depicted as shown in **Fig 1**. Without loss of generality, the number of outputs in the network will be assumed to be one, but the architecture can be readily extended to cope with multi-output problems. The architecture consists of an input layer, a hidden layer and an output layer. The input vector to the network is passed to the hidden layer nodes via unit connection weights. The hidden layer consists of a set of radial basis functions. Associated with each hidden layer node is a parameter vector $\mathbf{c}_i$ called a centre. The hidden layer node calculates the Euclidean distance between the centre and the network input vector and then passes the result to a radial basis function. All the radial basis functions in the hidden layer nodes are usually of the same type. Typical choices of the radial basis functions are

i). the thin-plate-spline function:

$$\phi(\mathbf{v}) = \mathbf{v}^2 \times \log(\mathbf{v}) \tag{1}$$

ii). the Gaussian function:

$$\phi(\mathbf{v}) = e^{-\left(\frac{\mathbf{v}^2}{\beta^2}\right)} \tag{2}$$

iii). the multiquadric function:

$$\phi(\mathbf{v}) = (\mathbf{v}^2 + \beta^2)^{\frac{1}{2}} \tag{3}$$

vi). the inverse multiquadric function:

$$\phi(\mathbf{v}) = \frac{1}{(\mathbf{v}^2 + \beta^2)^{\frac{1}{2}}} \tag{4}$$

where $\mathbf{v}$ is a non-negative number and is the distance from the input vector $\mathbf{x}$ to the radial basis function centre $\mathbf{c}$, and β is the width of the radial basis functions. In radial basis

function networks, the thin-plate-spline function has been used in (Chen et al., 1992; Chen et al., 1990b), and the Gaussian and multiquadric functions have been used by Moody (Moody and Darken, 1989), Broomhead (Broomhead and Lowe, 1988) and Poggio (Poggio and Girosi, 1990). In the present work, the thin-plate-spline function will be implemented in the network. However, other functions listed above can be readily included by using a constant β parameter.

Note that the network structure is determined by the input layer and the hidden layer. The input layer structure depends on the input vectors applied to the network, which are usually formed using past input output data in system identification applications or pattern vectors in pattern recognition applications. In the present work, we investigate the use of mutual information between the input variables and the output variables to select a subset of all the input variables available as the input of the network. The subset is selected according to the information content relevant to the output. The principle of the selection procedure is discussed in section three. In the remainder of this section however the input variables to the network are assumed to be determined by the selection scheme.

The response of the output layer node may be considered as a map f: $\mathbf{R}^n \rightarrow \mathbf{R}$, that is

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{N} \theta_i \phi(\| \mathbf{x} - \mathbf{c}_i \|) + \theta_0 \tag{5}$$

where N is the number of training data and $\| \bullet \|$ denotes the Euclidean norm. $\mathbf{c}_i$ (i=1, 2, ..., N) is the i^{th} centre and is the i^{th} data sample in this particular network structure. $\mathbf{x}, \mathbf{c}_i \in \mathbf{R}^n$, θ_i $(i = 1, 2, ..., N)$ are the weights associated with the i^{th} radial basis function centre. θ_0 is a constant term which acts as a shift in the output level. When the input layer nodes are determined, the training of the network may be seen as an interpolation problem and the solution may be obtained by solving a set of constrained linear equations. The complexity increases with the number of training data, which may make the implementation of the network above unrealistic. In practical applications, it is often desirable to use a network with a finite number of basis functions. A natural approximated solution would be

$$\mathbf{f}^*(\mathbf{x}) = \sum_{j=1}^{n_c} \theta_j \phi(\| \mathbf{x} - \mathbf{c}_j \|) + \theta_0 \tag{6}$$

where n_c is the number of radial basis function centres, $\mathbf{c}_j$ is the j^{th} centre which can be selected from the data samples. Given a set of data $(\mathbf{x}_i, \mathbf{y}_i)$, $(i = 1, 2, ..., N)$ $\mathbf{x}_i \in \mathbf{R}^n$, $y_i \in \mathbf{R}$, $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{in})^T$, the connection weights, centres and widths may be obtained by minimizing the following objective function

$$\mathbf{J}_1(\theta, \mathbf{c}) = \sum_{i=1}^{N} (y_i - \mathbf{f}^*)^T (y_i - \mathbf{f}^*) \tag{7}$$

where $\theta = (\theta_0, \theta_1, \theta_2, ..., \theta_{n_c})^T$, $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_{n_c})^T$. The above minimization problem may be solved using a nonlinear optimization or gradient decent algorithm. Note that the number of hidden layer nodes is predetermined in this network. The number of the hidden layer nodes may be selected using *a priori* knowledge. However, an appropriate number can only be determined by trial and error. It is therefore desirable to optimize the number

of hidden nodes, the locations or centres and the connection weights simultaneously. The objective function could be chosen as

$$\mathbf{J}_2(n_c, \theta, \mathbf{c}) = \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{n_c} \theta_j \phi(\| \mathbf{x}_i - \mathbf{c}_j \|) - \theta_0 \right)^T \left(y_i - \sum_{j=1}^{n_c} \theta_j \phi(\| \mathbf{x}_i - \mathbf{c}_j \|) - \theta_0 \right) \quad (8)$$

Note that the best structure which minimizes the objective function $\mathbf{J}_2$ has N hidden layer nodes and the centres tend to the data samples such that the network reverts to the one given in formula (5). The resulting network can only interpolate the particular data set and will fail to capture the underlying functional relation in the data, which will certainly lead to overfitting. To provide a compromise between network performance and network complexity, Akaike's information criterion (**AIC**) may be used and the objective function to be minimized can be amended to

$$\mathbf{J}_3(n_c, \theta, \mathbf{c}) = N \times log \left(\frac{1}{N} \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{n_c} \theta_j \phi(\| \mathbf{x}_i - \mathbf{c}_j \|) - \theta_0 \right)^T \left(y_i - \sum_{j=1}^{n_c} \theta_j \phi(\| \mathbf{x}_i - \mathbf{c}_j \|) - \theta_0 \right) \right) + 4 \times n_c + 4 \quad (9)$$

In previous paragraphs, it is assumed that the number of input nodes is known and the input layer nodes are predetermined. Therefore, the minimization of the objective function in (9) would determine both the network structure and the parameters simultaneously. In section four it will be shown how this objective can be achieved using the orthogonal least squares algorithm. The input nodes selection will be discussed in the next section.

3 Mutual Information and Input Nodes Selection

Mutual information is one of the most fundamental information measures in information theory. It is usually discussed in conjunction with efficient coding of sources and their reliable transmission over noisy channels. But it has also been applied in other disciplines. For example, Fraser and Swinney (Fraser and Swinney, 1986) used mutual information to select an optimal time delay for phase-portrait reconstruction from chaotic time series data. A slightly different formula was then used in (Ashrafi, Conway, Rokni, Sperling, Roszman and Cooley, 1993) for the same purpose. Although it was found that (Martinrie, Albano, Mees and Rapp, 1992) the method was not consistantly successful in identifying the optimal window due to the flatness of the mutual information with different window lengths for some time series. It was emphasized (Martinrie et al., 1992) that the importance of mutual information in estimating metric entropy remains undisputed and that the mutual information can be used for pattern recognition of noisy time series. In image processing, Noonan and Marcus (Noonan and Marcus, 1990) used the concept of mutual information for image restoration. The input image was restored by minimizing the mutual information between the input image and the output image while keeping the mean squared error

associated with the estimate of the input image equal to the known noise power. In neural networks, Linsker (Linsker, 1989) derived a learning rule by maximizing the mutual information between input and output signals to generate ordered maps. Several processes including a Hebb like modification, cooperation and competition among processing nodes emerged as components of the learning rule. The mutual information between the inputs and the outputs of the hidden neurons was introduced (Deco, Finnoff and Zimmermann, 1995) into the backpropagation training algorithm as a penalty term, which penalizes an excessive extraction of information by the hidden neuron and prevents overfitting. Conese and Maselli (Conese and Maselli, 1993) applied mutual information to select optimum band subsets from remotely sensed scenes for visual interpretation. Recently, Battiti (Battiti, 1994) applied it to select pattern features for pattern recognition. In the present work, we investigate the use of mutual information in selecting input nodes for radial basis function networks for pattern recognition applications. We also investigate the feasibility of selecting a suboptimal set of input variables for system identification applications. Before formulating the selection algorithm, we review the concepts of mutual information below.

Let $\mathbf{Y}$ be a random variable with a set of N_Y outcomes $\mathbf{Y}_1$, $\mathbf{Y}_2$, ..., $\mathbf{Y}_{N_Y}$ with probabilities $P(\mathbf{Y}_j) = p_j$, $(j = 1, 2, \ldots, N_Y)$, a measure of informatuion for $\mathbf{Y}$ is the entropy defined as

$$H(\mathbf{Y}) = -\sum_{j=1}^{N_Y} P(\mathbf{Y}_j)\ log\ P(\mathbf{Y}_j) = -\sum_{j=1}^{N_Y} p_j\ log\ p_j \tag{10}$$

When p_j is zero, the value zero is signed to $p_j\ log\ p_j$. H($\mathbf{Y}$) measures the uncertainty in the variable $\mathbf{Y}$. It vanishes if, and only if there is complete certainty in the variable $\mathbf{Y}$.

Let $\mathbf{U}$ be a random variable with a set of N_U outcomes $\mathbf{U}_1$, $\mathbf{U}_2$, ..., $\mathbf{U}_{N_U}$ with probabilities $P(\mathbf{U}_k) = q_k, (k = 1, 2, \ldots, N_U)$. The connection between the two variables is obtained by specifying their joint probabilities $P(\mathbf{Y}_j \cap \mathbf{U}_k) = p_{jk}, (j = 1, 2, \ldots, N_Y, k = 1, 2, \ldots, N_U)$. The conditional probabilities are

$$P(\mathbf{Y}_j|\mathbf{U}_k) = \frac{p_{jk}}{q_k}, \quad P(\mathbf{U}_k|\mathbf{Y}_j) = \frac{p_{jk}}{p_j} \tag{11}$$

The conditional entropy is defined as

$$H(\mathbf{Y}|\mathbf{U}) = -\sum_{k=1}^{N_U} P(\mathbf{U}_k) \sum_{j=1}^{N_Y} P(\mathbf{Y}_j|\mathbf{U}_k)\ log\ P(\mathbf{Y}_j|\mathbf{U}_k) = -\sum_{k=1}^{N_U}\sum_{j=1}^{N_Y} p_{jk}\ log\ \left(\frac{p_{jk}}{q_k}\right) \tag{12}$$

$H(\mathbf{Y}|\mathbf{U})$ is a measure of the uncertainty in variable $\mathbf{Y}$ after knowing the variable $\mathbf{U}$. The combined uncertainty of variable $\mathbf{Y}$ and $\mathbf{U}$ is then given by the joint entropy

$$H(\mathbf{Y} \cap \mathbf{U}) = -\sum_{j=1}^{N_Y}\sum_{k=1}^{N_U} P(\mathbf{Y}_j \cap \mathbf{U}_k)\ log\ P(\mathbf{Y}_j \cap \mathbf{U}_k) = -\sum_{j=1}^{N_Y}\sum_{k=1}^{N_U} p_{jk}\ log\ p_{jk} \tag{13}$$

From (12) and (13), it may be shown that the joint entropy $H(\mathbf{Y} \cap \mathbf{U})$ is the sum of the conditional entropy $H(\mathbf{Y}|\mathbf{U})$ and the uncertainty $H(\mathbf{U})$, that is

$$H(\mathbf{Y} \cap \mathbf{U}) = H(\mathbf{Y}|\mathbf{U}) + H(\mathbf{U}) \tag{14}$$

In general, the uncertainty in variable $\mathbf{Y}$ will be reduced after knowing the variable $\mathbf{U}$ if they are statistically related. The amount of reduction in uncertainty is

$$I(\mathbf{Y},\mathbf{U}) = H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{U}) = H(\mathbf{Y}) + H(\mathbf{U}) - H(\mathbf{Y} \cap \mathbf{U}) = \sum_{j=1}^{N_Y} \sum_{k=1}^{N_U} p_{jk} \; log \left(\frac{p_{jk}}{p_j q_k} \right) \quad (15)$$

where I($\mathbf{Y}$, $\mathbf{U}$) is the mutual information. It was shown (Jones, 1979) that the conditional entropy $H(\mathbf{Y}|\mathbf{U})$ can not exceed the uncertainty in variable $\mathbf{Y}$, *i.e.*

$$H(\mathbf{Y}|\mathbf{U}) \leq H(\mathbf{Y}) \quad (16)$$

The equality holds only when the two variables $\mathbf{Y}$ and $\mathbf{U}$ are statistically independent. When the two variables $\mathbf{Y}$ and $\mathbf{U}$ are perfectly associated to each other in the sence that $P(\mathbf{Y}_j|\mathbf{U}_j) = 1$ for all j and $P(\mathbf{Y}_j|\mathbf{U}_k) = 0$ for $k \neq j$, then

$$H(\mathbf{Y}|\mathbf{U}) = 0 \quad (17)$$

Substituting (17) into equation (15) yields

$$I(\mathbf{Y},\mathbf{U}) = H(\mathbf{Y}) = H(\mathbf{U}) \quad (18)$$

This is to be expected since the reduction of uncertainty in variable $\mathbf{Y}$ after knowing variable $\mathbf{U}$ can not be larger than the original uncertainty. Since the entropy can not be negative, it follows from (15) and (16) that

$$0 \leq I(\mathbf{Y},\mathbf{U}) \leq H(\mathbf{Y}) \quad (19)$$

The equality on the left holds only when the two variables are statistically independent and the one on the right holds only when the two variables are perfectly associated. The formulae given above are also correct when $\mathbf{U}$ is replaced by a vector.

Mutual information is generally considered as a measure of the dependence between two variables. If the two variables are independent, the mutual information between them is zero, if the two variables are strongly dependent, *e.g.* one is the function of another, the mutual information between them is large. Alternatively, the mutual information can be considered as a measure of the stored information in one variable about another, or a measure of the degree of the predictability of the output variable by knowing the input variable. Note that the mutual information measures the general dependence including nonlinear relations between variables. It may therefore be used to select input nodes for neural networks. The training of neural networks may be seen as a process of information extraction. The underlying functional relationship between the input and output variables is learnt during the training process. It is clear that input variables which are irrelevant to the output or which carry low information about the output may increase the computation time unnecessarily and even cause a deterioration in network performance. It may therefore be advantageous to eliminate these input variables before training.

Assume that there is a set S_m with m input variables. The task is to find a subset S_n with $n < m$ input variables which are maximaly informative about the output $\mathbf{Y}$. In terms of information theory, this may be seen as a problem of finding a subset S_n from S_m such

that the mutual information between the output $\mathbf{Y}$ and the subset S_n of input variables is a maximum, *i.e.*

$$\bar{S}_n = max_{S_n \subset S_m} I(\mathbf{Y}, S_n) \tag{20}$$

The number of all possible subsets of S_n with n input variables amounts to $\begin{pmatrix} m \\ n \end{pmatrix}$. To consider all these subsets may be computationally expensive when n is large. In addition, the number of data samples required to compute the mutual information $I(\mathbf{Y}, S_n)$ may be in the order of millions. In real applications however, the number of data samples is usually in the order of thounds or even hundreds. An alternative way therefore is to seek a suboptimal solution by computing a lower dimension mutual information and to build up the subset step by step. For an input variable to be selected, it seems natural that it should satisfy the following heuristic criteria.

- It should be comparatively informative about the output.
- It should not be strongly dependent on other variables selected.

If a variable is strongly dependent on other variables in the subset S_n, it adds little information about the output and may be considered as redundant. Note that the two selection criteria may conflict with each other. The concept of Pareto-optimality may be applied to appropriately compare two variables using these two criteria. In the following analysis $\mathbf{U}$ will be replaced by $\mathbf{X}$ and denoted by subscripts. Assume that $I(\mathbf{Y}, \mathbf{X}_k)$, $I(\mathbf{Y}, \mathbf{X}_i)$ and $I(\mathbf{Y}, \mathbf{X}_j)$ are known, where $\mathbf{X}_k \in S_n$, $\mathbf{X}_i, \mathbf{X}_j \in S_m, i \neq j$. For any $j \neq i$, if the following two relations are not satisfied simultaneously

$$I(\mathbf{Y}, \mathbf{X}_j) \geq I(\mathbf{Y}, \mathbf{X}_i) \tag{21}$$

$$\sum_k I(\mathbf{X}_k, \mathbf{X}_j) \leq \sum_k I(\mathbf{X}_k, \mathbf{X}_i) \tag{22}$$

Then it may be said that the input $\mathbf{X}_i$ is not dominated by $\mathbf{X}_j$. If $\mathbf{X}_i$ is not dominated by all $\mathbf{X}_j, j \neq i$, $\mathbf{X}_i$ is nondominated or noninferior. A set of noninferior input variables may be called a Pareto-optimal set. At each selection step, the input variables in the Pareto-optimal set are candidates to be included in S_n. Since the Pareto-optimal set may contain more than one variable, the above selection method may lead to ambiguity. In practice however, it is usually easy to judge which is the one to be selected. For example, variables which have low mutual information with both the output variable and those variables already selected may be excluded according to the heuristics. This procedure can be done interactively by plotting the mutual information $I(\mathbf{Y}, \mathbf{X}_i)$ against $\sum_k I(\mathbf{X}_k, \mathbf{X}_i)$ for every $\mathbf{X}_i$. Alternatively, it can be automated by weighting the two criteria of those variables in the Pareto-optimal set. In either case our experience shows that the final set S_n are usually very similar. It was also found that the noninferior variables at an early stage often turn out to be noninferior later. An alternative would be to simply select the most informative variable at each step. This strategy was used in references (Conese and Maselli, 1993) and (Battiti, 1994). This is clearly a special case of our selection procedure. Another alternative is to maximize a weighted difference $I(\mathbf{Y}, \mathbf{X}_j) - \alpha \sum_k I(\mathbf{X}_k, \mathbf{X}_j)$. If $I(\mathbf{Y}, \mathbf{X}_i) - \alpha \sum_k I(\mathbf{X}_k, \mathbf{X}_i) > I(\mathbf{Y}, \mathbf{X}_j) - \alpha \sum_k I(\mathbf{X}_k, \mathbf{X}_j)$, for $j \neq i$, then the variable $\mathbf{X}_i$ is selected. Best results were obtained when the parameter α was in the range of 0.5 to 1.0 (Battiti, 1994). It may be

seen that this procedure may select a variable which is dominated by others. It is also possible to select a variable which is weakly dependent on other variables in S_n but carries little information about the output as well. This is clearly inconsistent with the heuristics given above. Since the methods search for the set S_n in a suboptimal way, it is conceivable that no one selection procedure mentioned here can achieve consistantly better performance than the others. The selection scheme may therefore be summarized as follows:

Input node selection algorithm A

1. Set S_n as an empty set and S_m to contain all the input variables available. Set the desired number of input nodes K.
2. For each $\mathbf{X}_i$ in S_m, compute $I(\mathbf{Y}, \mathbf{X}_i)$.
3. Select $\mathbf{X}_i$ such that $I(\mathbf{Y}, \mathbf{X}_i) \geq I(Y, \mathbf{X}_j)$, for $j \neq i$, move $\mathbf{X}_i$ from set S_m into set S_n. Set counter k=1.
4. Compute $I(\mathbf{X}_k^{S_n}, \mathbf{X}_i^{S_m}), \mathbf{X}_k^{S_n} \in S_n, \mathbf{X}_i^{S_m} \in S_m$. Plot $I(\mathbf{Y}, \mathbf{X}_i^{S_m})$ versus $\sum_k I(\mathbf{X}_k^{S_n}, \mathbf{X}_i^{S_m})$. Select one $\mathbf{X}_i^{S_m}$ from the Pareto-optimal set. Move $\mathbf{X}_i^{S_m}$ from set S_m into set S_n. Set counter k = k + 1.
5. If k = K, output set S_n, otherwise, go to step 4.

The main task in computing the mutual information is estimating the probabilities P(**Y**), $P(\mathbf{X}_i)$ and P($\mathbf{Y} \cap \mathbf{X}_i$). In the present work, these probabilities are approximated by histograms. The range of variables **Y** and $\mathbf{X}_i$ are divided into equal sized intervals. The number of data samples falling into the intervals are counted and the probabilities are obtained by dividing these numbers by the total number of data samples. Due to the finite size effect of the data samples, estimation error exists in the mutual information. The method always overestimates the mutual information (Li, 1990; Treves and Panzeri, 1995). In the following, we denote the estimation of the probabilities over a finite data length N as $P_N(\mathbf{Y})$, $P_N(\mathbf{X}_i)$ and $P_N(\mathbf{Y} \cap \mathbf{X}_i)$ respectively. Assume that the partition of the output variable **Y** satisfies an independence condition *i.e.*, the number of times a given interval is occupied should depend only on the underlying probability P(**Y**) and that $P_N(\mathbf{X}_i)$ is given by a binomial distribution of mean value $P(\mathbf{X}_i)$. Average over all the possible N outcomes of the variable **Y**, the difference between the average of the estimation $< I_N >$ and the true mutual information is (Treves and Panzeri, 1995)

$$< I_N >= I + \sum_{m=1}^{\infty} C_m \tag{23}$$

where C_m $(i = 1, 2, \ldots)$ are the correction terms. The first term C_1 is

$$C_1 = \frac{1}{2N}(N_{X_i} - 1)(N_Y - 1) \tag{24}$$

where N_Y and N_{X_i} are the number of intervals of **Y** and $\mathbf{X}_i$ respectively. This term is invariant with respect to the probability distributions of **Y** and $\mathbf{X}_i$. The other terms depend

explicitly on averages of inverse powers of $P_N(\mathbf{X}_i)$, $P(\mathbf{Y}|\mathbf{X}_i)$ and $P(\mathbf{Y})$. This dependence can be very strong and produce strong fluctuations in the corresponding correction terms as the underlying probability distribution varies by tiny amounts. It was shown in (Treves and Panzeri, 1995) however by simulated results that C_1 was a good approximation of $< I_N > -I$. A similar result was given in reference (Li, 1990) under the assumption that the ratio of $P(\mathbf{Y} \cap \mathbf{X}_i)/P(\mathbf{Y})P(\mathbf{X}_i)$ does not change with the values of $\mathbf{Y}$ and $\mathbf{X}_i$ very much and that the fluctuation of the variables is of the magnitude of the square root of the variable values, *i.e.* $\delta P_N(\mathbf{Y} \cap \mathbf{X}_i) \propto \sqrt{P_N(\mathbf{Y} \cap \mathbf{X}_i)}$ and $\delta P_N(\mathbf{X}_i) \propto \sqrt{P_N(\mathbf{X}_i)}$. The underlying probabilities are usually unknown in practical applications, the estimation of the mutual information therefore must be treated with care. As a guildline, one should ensure that the independence condition holds and $C_1 \ll 1.0$. The number of intervals of the variables should also be appropriately chosen. For a given number of data samples, using a small number of intervals will give a more accurate estimate of the probabilities, but may cancel the details of the distributions and reduce the values of the mutual information. Using a large number of intervals will reveal the changes in the probability distributions over short distances, but the fluctuations caused by a small sample size in each interval may be interpreted as a small scale structure in the distributions and increase the values of the mutual information. A better way would be to partition the range of the variables adaptively (Fraser and Swinney, 1986). The methods described in reference (Silverman, 1986) may also be used for probability estimation. Since the mutual information for different variables is overestimated in a similar way, the estimation errors may have a limited effect on input node selection.

The selection algorithm presented above achieved promising results in pattern recognition applications. However, it was found to be inadequate in system identification type problems. This may be appreciated from the experimental results given later in section 5.2. In system identification the input variables of the network are formed using past input and output data from the system and these are usually strongly dependent on each other. This strong dependence among the variables may lead the algorithm to select an incorrect subset for the network in the sense that the selected variables are different from those of the real system. Assume that the system output $y(t)$ depends on both $y(t-1)$ and $y(t-2)$. The mutual information between $y(t-i)$ and $y(t-i-1)$ are all the same for $i \geq 0$. If variable $y(t-3)$ is selected at an early stage, variable $y(t-2)$ is likely to be excluded. If the system is periodic with a period of T, it is highly possible that the algorithm will select $y(t-T)$ first instead of $y(t-1)$ and $y(t-2)$. Note that the selected variables may be useful for output prediction but may fail to reveal the underlying system dynamics. In the following, we present a selection scheme for system identification applications. This overcame the above problems to a certain extent and achieved improved results for some systems.

Input node selection algorithm B

1. Set S_n as an empty set and S_m to contain all the input variables available. Set the desired number of input nodes K.

2. For each pair of variables $\mathbf{X}_i$ and $\mathbf{X}_j, j \neq i$ in S_m, compute $I(\mathbf{Y}, \mathbf{X}_i \cap \mathbf{X}_j)$.

3. Select the pair $\mathbf{X}_i$ and $\mathbf{X}_j$ such that $I(\mathbf{Y}, \mathbf{X}_i \cap \mathbf{X}_j) \geq I(Y, \mathbf{X}_p \cap \mathbf{X}_q), \{i,j\} \neq \{p,q\}$, move $\mathbf{X}_i$ and $\mathbf{X}_j$ from set S_m into set S_n. Set counter k=2.

4. For $\mathbf{X}_i^{S_m}$ and $\mathbf{X}_j^{S_m}, j \neq i$ in S_m, $\mathbf{X}_k^{S_n} \in S_n$, select $\mathbf{X}_i^{S_m}$ such that $\sum_k I(\mathbf{Y}, \mathbf{X}_k^{S_n} \cap \mathbf{X}_i^{S_m}) \geq \sum_k I(\mathbf{Y}, \mathbf{X}_k^{S_n} \cap \mathbf{X}_j^{S_m})$. Move $\mathbf{X}_i^{S_m}$ from set S_m into set S_n. Set counter k = k + 1.

5. If k = K, output set S_n, otherwise, go to step 4.

4 Orthogonal Least Squares and Hidden Nodes Selection

Assume the output y_i can be represented by

$$y_i = \sum_{j=1}^{n_c} \theta_j \phi(\| \mathbf{x}_i - \mathbf{c}_j \|) + \theta_0 + \xi_i \tag{25}$$

where y_i is the output or class membership, $\mathbf{x}_i$ is the input or the pattern vector, ξ_i represents the moddelling error which is assumed to be uncorrelated with the outputs of the hidden layer nodes $\phi(\| \mathbf{x}_i - \mathbf{c}_j \|)$, $j = 1, ..., n_c$. Rearrange (25) in vector form to yield

$$\mathbf{Y} = \Phi\Theta + \Xi \tag{26}$$

where

$$\begin{aligned}
\mathbf{Y} &= [y_1, \ y_2, \ \ldots, \ y_N]^T \\
\Phi &= [\Phi_0, \ \Phi_1, \ \ldots, \ \Phi_{n_c}] \\
\Theta &= [\theta_0, \ \theta_1, \ \ldots, \ \theta_{n_c}]^T \\
\Xi &= [\xi_1, \ \xi_2, \ \ldots, \ \xi_N]^T \\
\Phi_0 &= [1, \ 1, \ \ldots, \ 1]^T \\
\Phi_i &= [\phi(\| \mathbf{x}_1 - \mathbf{c}_i \|), \phi(\| \mathbf{x}_2 - \mathbf{c}_i \|), \ldots, \phi(\| \mathbf{x}_N - \mathbf{c}_i \|)]^T \qquad i \neq 0
\end{aligned}$$

An orthogonal decomposition of Φ is given as

$$\Phi = \mathbf{PA} \tag{27}$$

where

$$\mathbf{A} = \begin{vmatrix}
1 & \alpha_{1,2} & \alpha_{1,3} & \ldots & \alpha_{1,n_c+1} \\
 & 1 & \alpha_{2,3} & \ldots & \alpha_{2,n_c+1} \\
 & & \ddots & \ddots & \vdots \\
 & & & 1 & \alpha_{n_c,n_c+1} \\
 & & & & 1
\end{vmatrix}$$

is an $(n_c + 1) \times (n_c + 1)$ unit upper triangular matrix and

$$\mathbf{P} = [\mathbf{p}_0, \ \mathbf{p}_1, \ \ldots, \ \mathbf{p}_{n_c}] \tag{28}$$

is an $N \times (n_c + 1)$ matrix with orthogonal columns that satisfy

$$\mathbf{P}^T\mathbf{P} = \mathbf{D} \tag{29}$$

and $\mathbf{D}$ is a positive diagonal matrix

$$\mathbf{D} = diag\{d_0, d_1, \ldots, d_{n_c}\} \tag{30}$$

with

$$d_j = < \mathbf{p}_j, \mathbf{p}_j >, \quad j = 0, 1, \ldots, n_c \tag{31}$$

where $< \bullet, \bullet >$ denotes the inner product. Rearranging equation (26) yields

$$\mathbf{Y} = (\Phi\mathbf{A}^{-1})(\mathbf{A}\Theta) + \Xi = \mathbf{P}\mathbf{g} + \Theta \tag{32}$$

where

$$\mathbf{A}\Theta = \mathbf{g} \tag{33}$$

Because ξ_i is uncorrelated with the output of the hidden layer nodes, it may be shown that

$$\mathbf{g} = \mathbf{D}^{-1}\mathbf{P}^T\mathbf{Y} \tag{34}$$

or

$$g_i = \frac{< \mathbf{p}_j, \mathbf{Y} >}{< \mathbf{p}_j, \mathbf{p}_j >}, \qquad j = 0, 1, \ldots, n_c \tag{35}$$

Several orthogonal least squares methods including the classical Gram-Schmidt, the modified Gram-Schmidt methods and the Householder trasformation method may be used to obtain the trianglar system (33) (Chen, Billings and Luo, 1989). The number of the candidate centres N may be very large, but a small number of centres may be adequate to approximate the underlying functional relationship between $\mathbf{x}_i$ and y_i. These centres can be identified using an efficient forward selection procedure derived in (Chen, Billings, Cowan and Grant, 1990a). The principle of the method is shown below. From equation (30), the sum of the squares of the output is

$$< \mathbf{Y}, \mathbf{Y} > = \sum_{j=0}^{n_c} g_j^2 < \mathbf{p}_j, \mathbf{p}_j > + < \Xi, \Xi > \tag{36}$$

Where the errors are assumed to be uncorrelated with the hidden layer outputs. The error reduction ratio (Billings, Korenberg and Chen, 1988) due to $\mathbf{p}_j$ may be expressed as

$$\mathbf{err}_j = \frac{g_j^2 < \mathbf{p}_j, \mathbf{p}_j >}{< \mathbf{Y}, \mathbf{Y} >} \tag{37}$$

The best candidate centre at each step is the one which achieves the largest error reduction ratio $\mathbf{err}_j$. The selection procedure may be terminated when a desired error tolerance ρ $(0 < \rho < 1)$ is achieved.

$$1 - \sum_{j=0}^{n_c} \mathbf{err}_j < \rho \tag{38}$$

Note that the bias term may or may not be selected for a particular application. The tolerance ρ will affect both the approximation accuracy and the complexity of the network. An appropriate value for ρ, however can be learned during the forward selection procedure and the selection can be automated (Billings and Chen, 1989). The criterion emphasizes the approximation accuracy of the network only. As mention previously, the resulting network may tend to interpolate the particular data set and thus lead to overfitting. A compromise would be to terminate the selection procedure when the objective function given in equation (9) is achieved. In the present work however, we simply terminate the selection procedure when a given number of centres are found for ease of comparison.

5 Experimental Results

5.1 Applications in Pattern Recognition

In this subsection two real data sets will be used to demonstrate the performance of the selection algorithm A described in section three. The first data set is a heart disease dataset and the second is the Australian credit approval dataset. These datasets were obtained from the machine learning databases of the Information and Computer Science Department at the California University. The heart disease dataset contains 270 data samples, each has 13 pattern features which were extracted from 75 pattern features. The Australian credit approval dataset contains 690 data samples, each has 14 pattern features. Both datasets have two classes. In the following, the pattern features are considered as meaningless symbols. The data samples are classified according to a winner-take-all rule.

Experiment 1: The heart disease dataset. The dataset were divided into two parts, the training set contains the first 220 data samples and the remaining 50 data samples were used as a test set. All the data samples were used for input node selection. The mutual information between the classmembership and the pattern features are shown in **Fig 2.** Three subsets each with six pattern features were selected using the input node selection algorithm.

$$\begin{aligned} S_6^1 &= \{\mathbf{X}_{13}, \mathbf{X}_3, \mathbf{X}_{12}, \mathbf{X}_{10}, \mathbf{X}_9, \mathbf{X}_8\} \\ S_6^2 &= \{\mathbf{X}_{13}, \mathbf{X}_{12}, \mathbf{X}_9, \mathbf{X}_{11}, \mathbf{X}_3, \mathbf{X}_8\} \\ S_6^3 &= \{\mathbf{X}_{13}, \mathbf{X}_{12}, \mathbf{X}_9, \mathbf{X}_{11}, \mathbf{X}_3, \mathbf{X}_2\} \end{aligned}$$

where the pattern fearures are presented in the order of selection. Each of the 6 pattern features in the subset S_6^1 has the largest mutual information with the output at each selection step. The three subsets were used as inputs to the **RBF** network respectively and the classification results were compared with the network with all 13 pattern features as inputs. The hidden layer nodes were selected by the **OLS** algorithm. The results are shown in **Fig 3 - 5.** Note that improved performance was obtained when the selected subsets were used as inputs. When compared to each other, the subset S_6^1 and S_6^2 gave similar classification results while S_6^3 performed slightly better when a small number of centres were used and slightly worse when the number of centres was higher than 8. It seems that the input variables can be selected according to the mutual information with the output if they are not highly dependent on other members of the subset.

Experiment 2: The Australian credit approval dataset. The data set were divided into a training set and a test set. The training set contains the first 540 samples

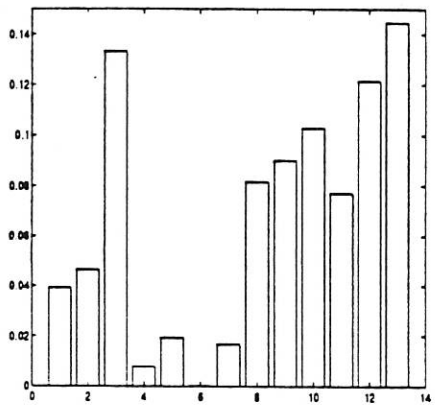


Figure 2: Mutual information (vertical axis) for the heart disease dataset between the output and the pattern features $\mathbf{X}_1, \ldots, \mathbf{X}_{13}$ (horizontal axis)

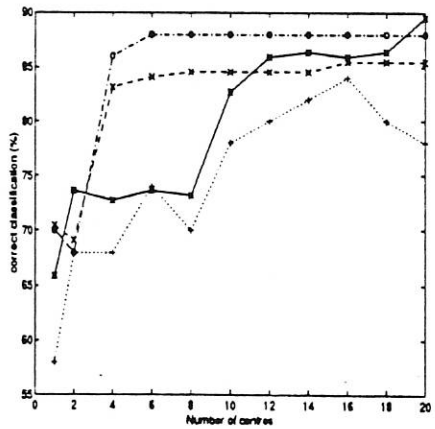


Figure 3: Classification results for the heart disease dataset. '*': training set, '+': test set (all the 13 pattern features were used as network inputs), 'x': training set, 'o': test set (only 6 pattern features $\{\mathbf{X}_{13}, \mathbf{X}_3, \mathbf{X}_{12}, \mathbf{X}_{10}, \mathbf{X}_9, \mathbf{X}_8\}$ were used as network inputs)

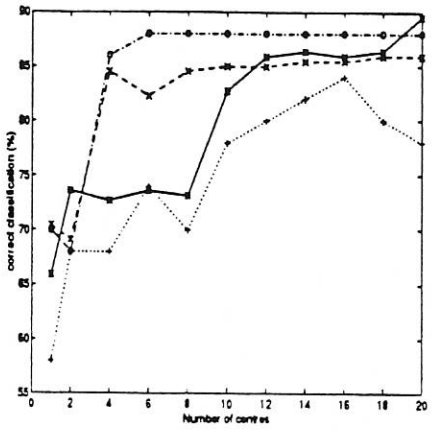


Figure 4: Classification results for the heart disease dataset. '*': training set, '+': test set (all the 13 pattern features were used as network inputs), 'x': training set, 'o': test set (only 6 pattern features $\{X_{13}, X_{12}, X_9, X_{11}, X_3, X_8\}$ were used as network inputs)

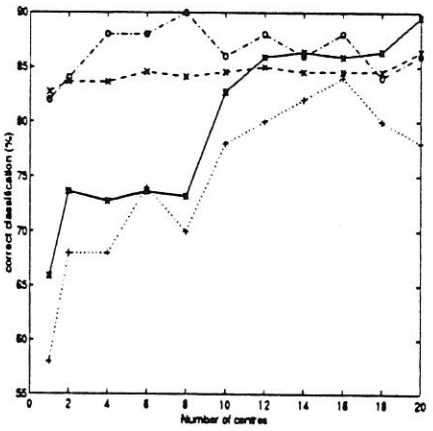


Figure 5: Classification results for the heart disease dataset. '*': training set, '+': test set (all the 13 pattern features were used as network inputs), 'x': training set, 'o': test set (only 6 pattern features $\{X_{13}, X_{12}, X_9, X_{11}, X_3, X_2\}$ were used as network inputs)

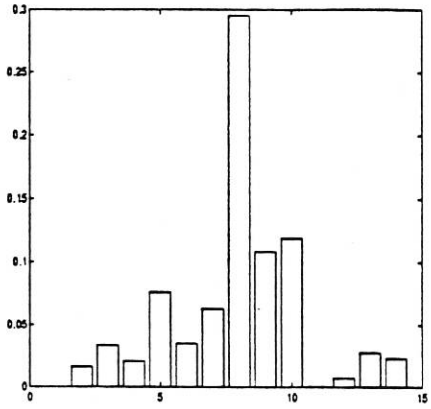


Figure 6: Mutual information (vertical axis) for the Australian credit approval dataset between the output and the pattern features $\mathbf{X}_1, \ldots, \mathbf{X}_{14}$ (horizontal axis)

and the test set contains the remaining 150 samples. The mutual information between the output and pattern features are plotted in **Fig 6**. Three subsets with 5 pattern features were selected.

$$\begin{aligned} S_5^1 &= \{\mathbf{X}_8, \mathbf{X}_{10}, \mathbf{X}_9, \mathbf{X}_5, \mathbf{X}_7\} \\ S_5^2 &= \{\mathbf{X}_8, \mathbf{X}_5, \mathbf{X}_9, \mathbf{X}_7, \mathbf{X}_{13}\} \\ S_5^3 &= \{\mathbf{X}_8, \mathbf{X}_5, \mathbf{X}_9, \mathbf{X}_7, \mathbf{X}_3\} \end{aligned}$$

Again the pattern features are presented in the order of selection. The subset S_5^1 contains the five pattern features which are most informative about the output. The three subsets were then used as network inputs respectively and the network performance was compared with the case when all the 14 pattern features were used as inputs. The classification results are presented in **Fig 7 - 9**. It may be seen that a higher classification rate was achieved when the selected subsets were used as network inputs. The effectiveness of the selection algorithm may be further appreciated from **Fig 10**, in which the classification performance of the network was compared when the subset S_5^2 ($\{\mathbf{X}_8, \mathbf{X}_5, \mathbf{X}_9, \mathbf{X}_7, \mathbf{X}_{13}\}$) and the first five pattern features ($\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4, \mathbf{X}_5\}$) were used as network inputs respectively. The former achieved a much higher classification rate than the later.

5.2 Applications in System Identification

In this subsection, four simulated examples will be used to investigate the performance of the selection algorithms presented in section three.

Experiment 3: A Linear Stochastic System. 1000 data samples were generated from the following stochastic system

$$y(t) = 1.5y(t-1) - 0.7y(t-2) + u(t-3) + 0.5u(t-4) + e(t) - e(t-1) + 0.2e(t-2) \quad (39)$$

where the system input $u(t)$ was a sequence of uniform distribution with zero mean and a variance of 1.38, the system noise was a Gaussian sequence with zero mean and a standard

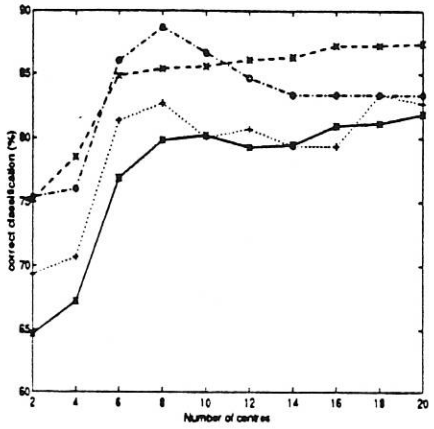


Figure 7: Classification results for the Australian credit approval dataset. '*': training set, '+': test set (all the 14 pattern features were used as network inputs), 'x': training set, 'o': test set (only 5 pattern features $\{X_8, X_{10}, X_9, X_5, X_7\}$ were used as network inputs)

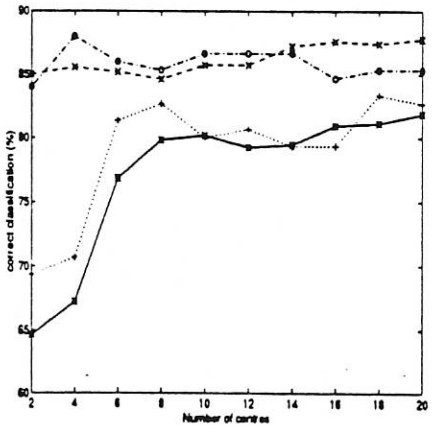


Figure 8: Classification results for the Australian credit approval dataset. '*': training set, '+': test set (all the 14 pattern features were used as network inputs), 'x': training set, 'o': test set (only 5 pattern features $\{X_8, X_5, X_9, X_7, X_{13}\}$ were used as network inputs)

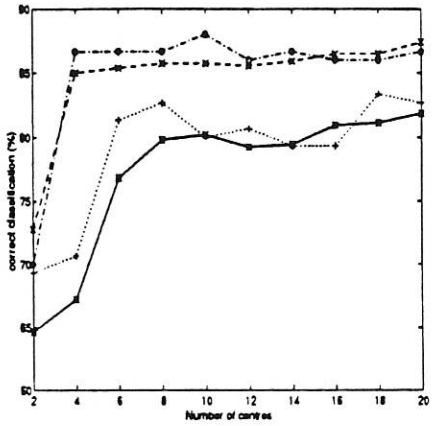


Figure 9: Classification results for the Australian credit approval dataset. '*': training set, '+': test set (all the 14 pattern features were used as network inputs), 'x': training set, 'o': test set (only 5 pattern features $\{\mathbf{X}_8, \mathbf{X}_5, \mathbf{X}_9, \mathbf{X}_7, \mathbf{X}_3\}$ were used as network inputs)

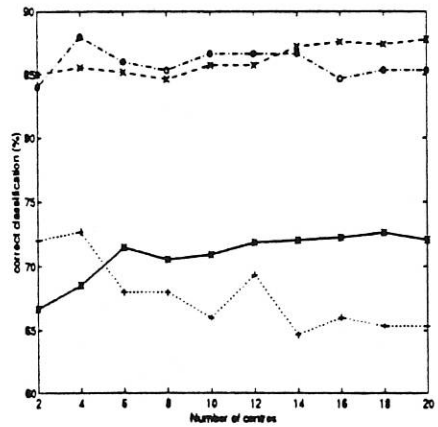


Figure 10: Classification results for the Australian credit approval dataset. '*': training set, '+': test set (the first 5 pattern features $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4, \mathbf{X}_5\}$ were assigned as network inputs), 'x': training set, 'o': test set (5 pattern features $\{\mathbf{X}_8, \mathbf{X}_5, \mathbf{X}_9, \mathbf{X}_7, \mathbf{X}_{13}\}$ were selected as network inputs)

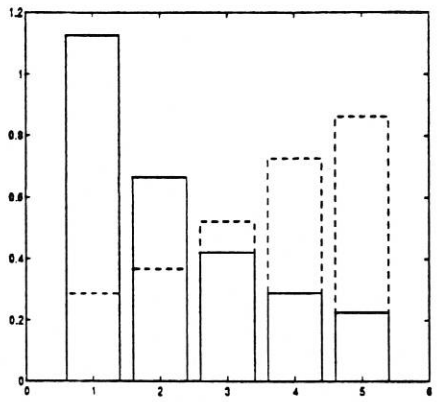
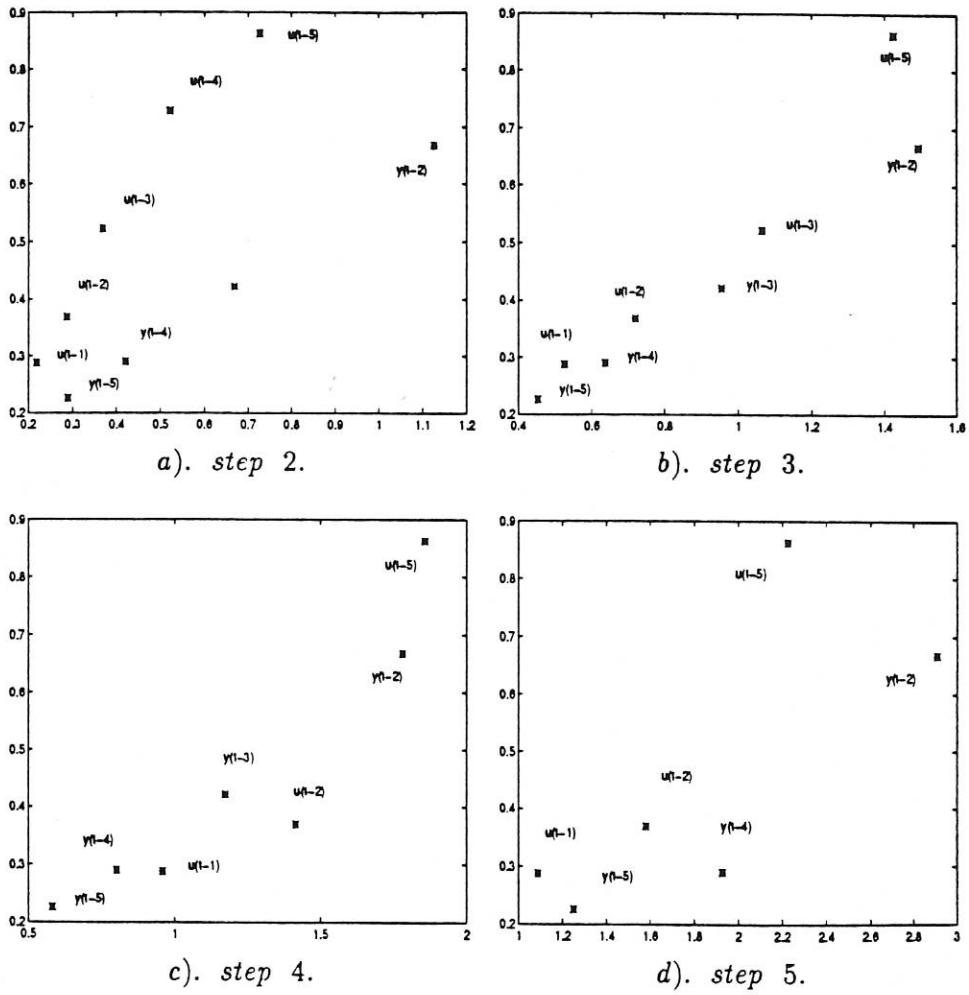


Figure 11: '-' mutual information between the output $y(t)$ and $y(t-i)$ for the stochastic system, vertical axis: mutual information $I(y(t), y(t-i))$, horizontal axis: $y(t-i), i = 1, \ldots, 5$, '- -' mutual information between the output $y(t)$ and $u(t-i)$ for the stochastic system, vertical axis: mutual information $I(y(t), u(t-i))$, horizontal axis: $u(t-i), i = 1, \ldots, 5$.

derivation of 0.2. Assume that 6 input variables are to be selected among 10 candidate variables $\{y(t-1), \ldots, y(t-5), u(t-1), \ldots, u(t-5)\}$. The mutual information between the output $y(t)$ and the 10 candidate variables are shown in **Fig 11**. The five most informative variables about the output are $y(t-1), u(t-5), u(t-4), y(t-2)$ and $u(t-3)$. Note that the four correct regressors for this system are included in this set. Consider the selection of five input variables using the selection algorithm A. The first variable to be selected is $y(t-1)$. A plot of $I(y(t), \mathbf{X}_i)$ against $I(y(t-1), \mathbf{X}_i)$ is shown in **Fig 12a**. The variables $u(t-3)$, $u(t-4)$ and $u(t-5)$ seem to be the suitable candidates and $u(t-4)$ was selected at this step because it gave a good compromise between $I(y(t), \mathbf{X}_i)$ and $I(y(t-1), \mathbf{X}_i)$. A plot $I(y(t), \mathbf{X}_i)$ against $I(y(t-1), \mathbf{X}_i) + I(u(t-4), \mathbf{X}_i)$ is shown in **Fig 12b**. The variable $u(t-3)$ may be selected at this step again because it provided a good compromise between $I(y(t), \mathbf{X}_i)$ and $I(y(t-1), \mathbf{X}_i) + I(u(t-4), \mathbf{X}_i)$. Continuing this procedure, **Fig 12c** was obtained. If $y(t-2)$ was selected, a perfect variable set would be achieved. However, if $y(t-3)$ is selected as was done here, the next plot will be as shown in **Fig 12d** and the variable $y(t-2)$ will certainly be excluded. This is because $y(t-2)$ has a high dependence on both $y(t-1)$ and $y(t-3)$, so that after $y(t-3)$ was selected $y(t-2)$ became redundant. It must be emphasized that this kind of situation may not cause any problem for pattern recognition applications because there is usually a lower dependence among the pattern features. When the selection algorithm B was applied to this system, the following variable set $\{y(t-1), u(t-4), u(t-5), y(t-2), u(t-3)\}$ was obtained for several different partitions of the variables $y(t)$ and $u(t)$.



a). *step* 2. *b*). *step* 3.

c). *step* 4. *d*). *step* 5.

Figure 12: Plots showing the selection procedure of algorithm A for example 3. vertical axes: $I(\mathbf{Y}, \mathbf{X}_i^{S_m})$, horizontal axes: $\sum_k I(\mathbf{X}_k^{S_n}, \mathbf{X}_i^{S_m})$. $\mathbf{X}_i^{S_m} \in S_m$ (set of avaiable variables), $\mathbf{X}_k^{S_n} \in S_n$ (set of selected variables)

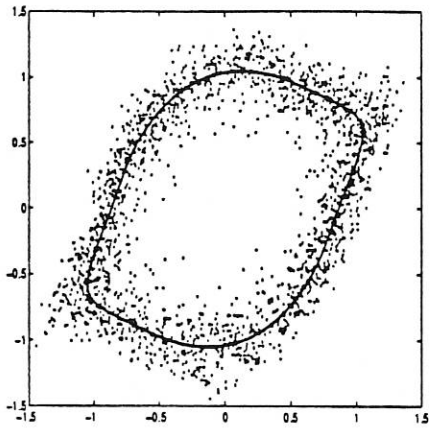


Figure 13: Pseudo-phase portrait of the autonomous system in example 4, $y(t)$ (vertical) vs $y(t-1)$ (horizontal), '-' data without noise, '..' data with noise.

Experiment 4: An autonomous System. 2000 data samples were generated from the following autonomous system

$$\begin{aligned} y(t) &= \left(0.8 - 0.5e^{-y^2(t-1)}\right) y(t-1) - \left(0.3 + 0.9e^{-y^2(t-1)}\right) y(t-2) \\ &\quad +0.1\sin\left(y(t-1)\pi\right) + e(t) \end{aligned} \tag{40}$$

where $e(t)$ was a Gaussian sequence with zero mean and a standard derivation of 0.1. A pseudo phase portrait of the system is plotted in **Fig 13**. The clean data forms a limit cycle as shown by the solid line in the plot. Assume that four variables are to be selected from the following candidate variables $\{y(t-1), \ldots, y(t-10)\}$. The mutual information between the output $y(t)$ and the 10 variables are shown in **Fig 14**. It may be seen that the four most informative variables about the system output are $\{y(t-5), y(t-8), y(t-3), y(t-2)\}$. Two pseudo-phase portraits of the system were plotted in **Fig 15** with $y(t)$ versus $y(t-5)$ and $y(t)$ versus $y(t-8)$ respectively. Compare them with the pseudo-phase portrait in **Fig 13**, it may be seen that the variable y(t) is closely related to $y(t-5)$ and $y(t-8)$. This may be the reason why the algorithm failed to select $y(t-1)$. The selection algorithm B was then used to select the input variable. For several partitions of the variable $y(t)$, algorithm B selected the following four variables in the order of $\{y(t-1), y(t-2), y(t-3), y(t-5)\}$.

Experiment 5: A nonlinear system. This system is taken from reference (Haber and Unbehauen, 1990) (Haber and Unbehauen, 1990). The system is given as

$$\begin{aligned} y(t) &= -0.6377y(t-1) + 0.07298y(t-2) + 0.03597u(t-1) + 0.06622u(t-2) \\ &\quad +0.06568u(t-1)y(t-1) + 0.02375u^2(t-1) + 0.05939 \end{aligned} \tag{41}$$

A data sequence of length 1000 was generated from this nonlinear system. The system input u(t) was a uniformly distributed sequence. To make the data more realistic, a Gaussian sequence was added to the output $y(t)$ as noise. The resulting signal to noise ratio was 25 db. The input $u(t)$ and noisy output $y(t)$ are shown in **Fig 16**. Assume that 5 variables are to be

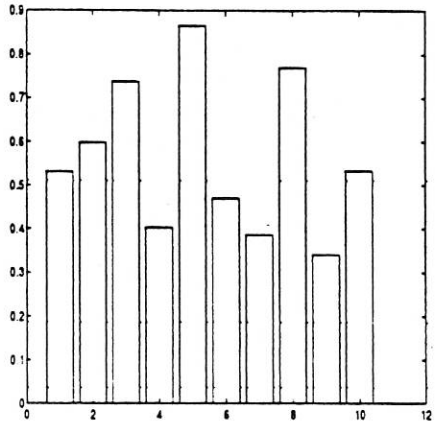


Figure 14: Mutual information (vertical axis) between the output $y(t)$ and the candidate variables $\{y(t-1), \ldots, y(t-10)\}$ (horizontal axis)

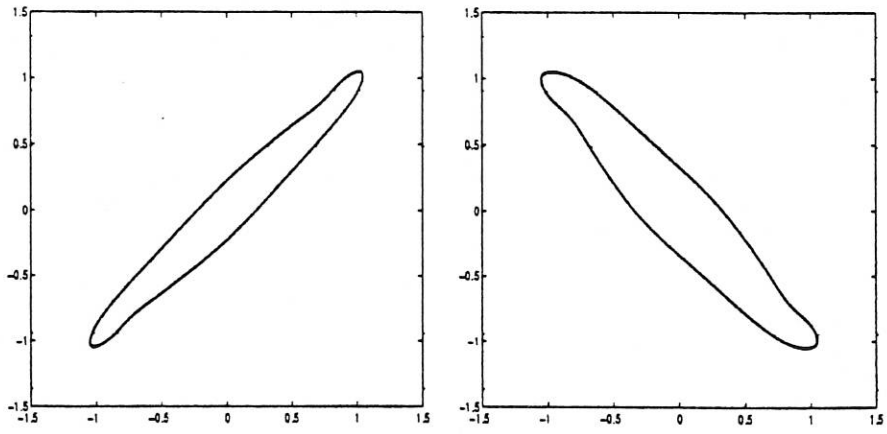
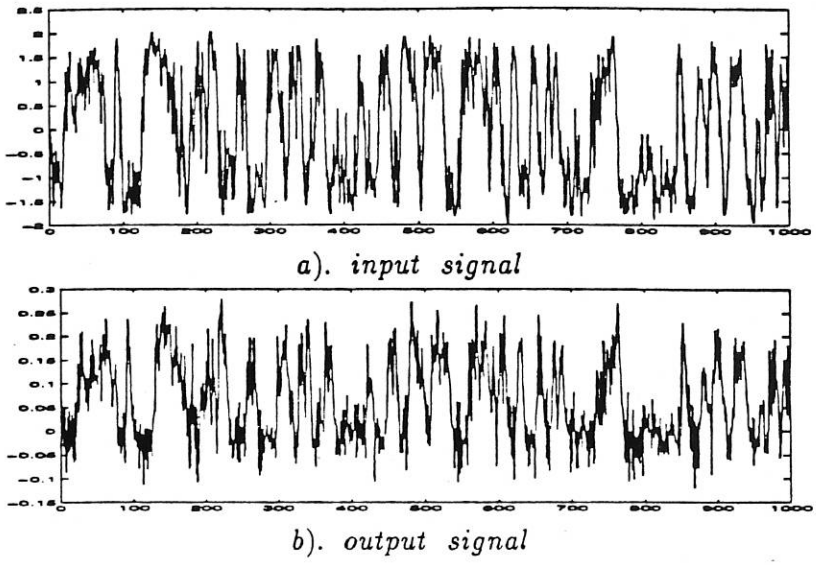


Figure 15: Pseudo-phase portrait of the autonomous system in example 4, left: $y(t)$ (vertical) vs $y(t-5)$ (horizontal), right: $y(t)$ (vertical) vs $y(t-8)$ (horizontal)



a). input signal

b). output signal

Figure 16: The input and noisy output signals for experimental 5

selected from the following 10 candidate variables $\{y(t-1),\ldots,y(t-5),u(t-1),\ldots,u(t-5)\}$. For several different partitions of the input and output variables, the following variable sets were selected by using the selection algorithm B respectively.

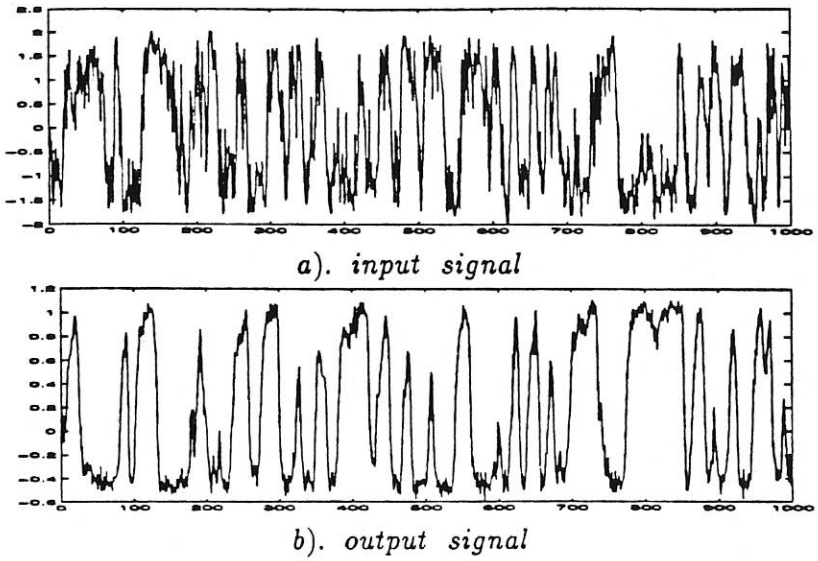
$$\begin{aligned}
S_5^1 &= \{u(t-1),u(t-2),y(t-1),y(t-2),u(t-3)\}\\
S_5^2 &= \{u(t-1),u(t-2),y(t-1),y(t-2),u(t-3)\}\\
S_5^3 &= \{u(t-1),u(t-2),y(t-1),y(t-2),u(t-5)\}\\
S_5^4 &= \{u(t-1),u(t-2),y(t-2),y(t-1),y(t-5)\}\\
S_5^5 &= \{u(t-1),u(t-2),y(t-2),y(t-1),y(t-5)\}
\end{aligned}$$

The variables in the subsets were presented in the order of selection. Note that all the four system variables were selected.

Experiment 6: A simulated turbogenerator. This example was also taken from reference (Haber and Unbehauen, 1990). This is a single input signal output model between the field current and the frequency of the generated voltage in a turbogenerator. The system equation was given as

$$\begin{aligned}
y(t) &= 0.84y(t-1)-0.0628u(t-2)-0.0675u(t-3)-0.0215u(t-4)-0.0613y^2(t-1)\\
&\quad -0.053y(t-1)u(t-2)-0.0526y(t-2)u(t-3)-0.0071u(t-2)u(t-3)\\
&\quad -0.0234y(t-1)u^2(t-2)-0.044y(t-1)u^2(t-3)+0.0573y^2(t-1)u(t-2)\\
&\quad -0.02y^2(t-3)-0.00113 \qquad (42)
\end{aligned}$$

A data sequence of 1000 was generated from this nonlinear system. The system input u(t) was a sequence of uniform distribution. A Gaussian sequence with zero mean and a standard derivation of 0.05 was added to the output $y(t)$ to simulate the effects of noise. This gave a signal noise ratio of 42 db. The input $u(t)$ and noisy output $y(t)$ are shown



a). input signal

b). output signal

Figure 17: The input and noisy output signals for experimental 6

in **Fig 17**. Assume that 8 variables are to be selected from the following 12 candidate variables $\{y(t-1),\ldots,y(t-6),u(t-1),\ldots,u(t-6)\}$. For several different partitions of the input and output variables, the following variable sets were selected by using the selection algorithm B respectively.

$$
\begin{aligned}
S_8^1 &= \{y(t-1),u(t-3),y(t-2),y(t-3),u(t-4),y(t-4),u(t-5),u(t-2)\}\\
S_8^2 &= \{y(t-1),u(t-3),y(t-2),u(t-4),y(t-3),y(t-4),u(t-2),u(t-5)\}\\
S_8^3 &= \{y(t-1),u(t-2),y(t-2),u(t-3),y(t-3),u(t-4),y(t-4),y(t-5)\}\\
S_8^4 &= \{y(t-1),u(t-3),y(t-2),y(t-3),u(t-4),y(t-4),u(t-2),y(t-5)\}\\
S_8^5 &= \{y(t-1),u(t-3),y(t-2),y(t-3),u(t-4),y(t-4),u(t-2),u(t-5)\}\\
S_8^6 &= \{y(t-1),u(t-3),y(t-2),y(t-4),u(t-4),y(t-3),u(t-2),u(t-5)\}
\end{aligned}
$$

The variables in the subsets were again presented in the order of selection. Again all the 6 system variables were picked up by the algorithm.

6 Conclusions

In this paper, mutual information and orthogonal least squares algorithms were used to select the input layer nodes and the hidden layer nodes for **RBF** networks. The input variables to the network were selected according to the information content with respect to the output. This not only reduced the complexity of the network but also improved the network performance as demonstrated by using two real pattern recognition data sets. Since the number of input nodes were reduced before network training, the method also considerably reduced the computational power required. When applied to system identification problems, the selection algorithm picked out all the true variables of the system

although a slightly larger variable set was obtained. The selection algorithms will certainly be useful in pattern recognition and system identification applications. In pattern recognition applications, the input variables are usually the available pattern features. Some of the pattern features may be redundant and some may be irrelevant to the problem. These features are certainly unnecessary and should be eliminated before network training. In system identification applications, the input variables are usually formed from past input output data. Lower and upper limits of the time delay are usually determined by using *a priori* knowledge or by trial and error. These limits are often overestimated. For example, the lower limit is usually assumed to be unity and the upper limit to be larger than some time constant. Too small a time delay may result in a deterioration in network performance, while too large a time limit may increase the computational time.

The performance of the input node selection algorithms were demonstrated based on a **RBF** architecture. But it is obvious that the algorithms can also be applied to other network architectures and system identification algorithms in general. Since the orthogonal least squares algorithm selects a group of hidden layer nodes in a suboptimal way, it therefore provides a reliable platform for comparing the network performance when different input variables are used as input nodes to the network.

However, due to the strong dependence between the variables in system identification applications care must be taken in applying the selection algorithms. Our experience indicates that the input signal may have a strong influence on the final results and the task is not always straight forward. It is important to ensure that the partition of the variables reflects the underlying distribution of the variables as well as possible and not to introduce too larger an error in the estimation of the probabilities.

Acknowledgements

The authors gratefully acknowledge that this work was supported by the EPSRC under the contract GR/H35286. GLZ would also like to thank Mr. T. S. Sze for his help on the orthogonal least squares algorithm.

References

Ashrafi, S., Conway, D., Rokni, M., Sperling, R., Roszman, L., and Cooley, J. (1993). Solar flux forecasting using mutual information with an optimal delay. *Advances in the Astronautical Sciences*, 84(1-2):901 - 913.

Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. on Neural Networks*, 5(4):537–550.

Billings, S. A. and Chen, S. (1989). Extended model set, global data and threshold model identification of severely non-linear systems. *Int. J. Control*, 50:1897.

Billings, S. A., Korenberg, M. J., and Chen, S. (1988). Identification of nonlinear output-affine systems using an orthogonal least squares algorithm. *Int. J. Control*, 19:1559–1568.

Broomhead, D. S. and Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321 - 355.

Chen, S., Billings, S. A., Cowan, C. F. N., and Grant, P. M. (1990a). Non-linear systems identification using radial basis functions. *Int. J. Systems Sci.*, 21(12):2513–2539.

Chen, S., Billings, S. A., Cowan, C. F. N., and Grant, P. W. (1990b). Practical identification of narmax models using radial basis functions. *Int. J. Control*, 52(6):1327–1350.

Chen, S., Billings, S. A., and Grant, P. W. (1992). Recursive hybrid algorithm for non-linear system identification using radial basis function network. *Int. J. Control*, 55(5):1051–1070.

Chen, S., Billings, S. A., and Luo, W. (1989). Orthogonal least squares methods and their application to nonlinear system identification. *Int. J. Control*, 50:1873–1896.

Conese, C. and Maselli, F. (1993). Selection of optimal bands from tm scenes through mutual information analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 48(3):2–11.

Deco, G., Finnoff, W., and Zimmermann, H. G. (1995). Unsupervised mutual information criterion for elimination of overtraining in supervised multilayer networks. *Neural Computation*, 7:86–107.

Fraser, A. M. and Swinney, H. L. (1986). Independent coordinates for strange attractors from mutual information. *Physical Review A*, 33(2):1134 - 1140.

Haber, R. and Unbehauen, H. (1990). Structure identification of nonlinear dynamic systems - a survey on input/output approaches. *Automatica*, 26(4):651–677.

Holcomb, T. and Morari, M. (1991). Local training for radial basis function networks: Towards solving the hidden unit problem. *Proc. American Control Conference*, pages 2331 - 2336.

Jones, D. S. (1979). *Elementary Information Theory.* Clarendon Press, Oxford.

Kaviori, S. N. and Venkata Subramanian, V. (1993). Using fuzzy clustering with ellipsoidal units in neural networks for robust fault classification. *Computers Chem. Eng.*, 17(8).

Lee, S. and Rhee, M. K. (1991). A gaussian potential function network with hierarchically self-organizing learning. *Neural Networks*, 4:207 – 224.

Li, W. (1990). Mutual information functions versus correlation functions. *Journal of Statistical Physics*, 60(5-6):823–837.

Light, W. A. (1992). Some aspects of radial basis function approximation. *Approximation Theory, Spline Functions and Applications*, 356:163 – 190.

Linsker, R. (1989). How to generate ordered maps by maximizing the mutual information between input and output signals. *Neural Computation*, 1:402–411.

Martinrie, J. M., Albano, A. M., Mees, A. I., and Rapp, P. E. (1992). Mutual information, strang attractors, and the optimal estimation of dimension. *Physical Review A*, 45(10):7058–7064.

Moody, J. and Darken, C. (1989). Fast learning in networks of locally-tunned processing units. *Neural Computation*, 1:281 – 294.

Musavi, M. T., Ahmed, W., Chan, K. H., Faris, K. B., and Hummels, D. M. (1992). On the training of radial basis function classifiers. *Neural Networks*, 5:595 – 603.

Noonan, J. P. and Marcus, J. R. (1990). Minimum mutual information in image restoration. *Kybernetes*, 19(6):34–41.

Poggio, T. and Girosi, F. (1990). Network for approximation and learning. *Proceedings of IEEE*, 78(9):1481 – 1497.

Powell, M. J. D. (1992). Radial basis functions in 1990. In *Advances in Numerical Analysis*, pages 105 – 210.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis.* Chapman and Hall.

Treves, A. and Panzeri, S. (1995). The upward bias in measures of information derived from limited data samples. *Neural Computation*, 7:399–407.

Vogt, M. (1993). Combination of radial basis function neural networks with optimized vector quantization. *Proceedings of the IEEE International conference on Neural Networks*, 3:1841–1846.

Xu, L., Krzyzak, A., and Oja, E. (1993). Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. *IEEE Trans. on Neural Networks*, 4(4):636–649.

Zheng, G. L. and Billings, S. A. (1994). Radial basis function network training using a fuzzy clustering scheme. (submitted for publication).

List of Figures