Multiobjective criteria for neural network structure selection and identification of nonlinear systems using genetic algorithms

G. P.Liu and V.Kadirkamanathan

Abstract: An approach to model selection and identification of nonlinear systems via neural networks and genetic algorithms is presented based on multiobjective performance criteria. It considers three performance indices or cost functions as the objectives, which are the euclidean distance (L_2-norm) and maximum difference (L_∞-norm) measurements between the real nonlinear system and the nonlinear model, and the complexity measurement of the nonlinear model, instead of a single performance index. An algorithm based on the method of inequalities, least squares and genetic algorithms is developed for optimising over the multiobjective criteria. Genetic algorithms are also used for model selection in which the structure of the neural networks is determined. The Volterra polynomial basis function network and the gaussian radial basis function network are applied to the identification of a liquid-level nonlinear system.

1 Introduction

Nonlinear system identification can be posed as a nonlinear functional approximation problem. From the Weierstrass theorem [1] and the Kolmogorov theorem [2] in approximation theory, it is known that the polynomial and many other approximation schemes can approximate a continuous function arbitrarily well. In recent years, many nonlinear system identification approaches, particularly identification using neural networks [3–13] based on the universal approximation theorem [14], are applications of a similar mathematical approach.

For nonlinear system identification using the approximation approach, two key questions are important: how to judge the accuracy for the nonlinear function being approximated and how to choose nonlinear function units to guarantee the accuracy. Many of nonlinear system identification approaches fix the number of nonlinear function units and use only a single performance function, *e.g.* L_2-norm of the difference between the real nonlinear system and the nonlinear model which results in the well-known least-squares algorithm, to measure and judge the accuracy of the identification model and to optimize the approximation. The assumption behind choosing the L_2-norm is that the noise in the process and measurements has gaussian (normal) distributions.

However, in nonlinear system identification there are often a number of objectives to be considered. The objectives are often conflicting and no identification which can be considered best with respect to all objectives exists. Hence, there is an inevitable trade-off between objectives, for example, the distance measurement and maximum difference measurement between the real nonlinear system and the nonlinear model. Model comparison methods, such as information criterion [15], bayesian model selection [16] and minimum description length (MDL) [17], consider two such objectives, namely, Euclidean distance (L_2-norm) and model complexity. These procedures allow the selection of the best amongst a small number of candidate models [16]. We consider in addition, the L_∞-norm of the difference between the real nonlinear system and the nonlinear model because it represents the accuracy bound of the approximation achieved by the estimated model.

These considerations lead to the study of multiobjective nonlinear system identification. This paper presents three multiobjective performance functions to measure the approximation accuracy and the complexity of the nonlinear model for noise with mixed distribution. Those functions are the L_2-and L_∞-norms of the difference measurements between the real nonlinear system and the nonlinear model, and the number of nonlinear units in the nonlinear model. Genetic algorithms are used to search for a suboptimal set of nonlinear basis functions of the model to simplify model estimation. Two neural networks are applied for the model representation of the nonlinear systems. One is the Volterra polynomial basis function (VPBF) network and the other is the gaussian radial basis function (GRBF) network. We also develop a numerical algorithm for multiobjective nonlinear model selection and identification using neural networks and genetic algorithms. Two examples in identification of a nonlinear system and approximation of a nonlinear function with mixed noise demonstrate the operation of the algorithm.

2 Nonlinear modelling with NNs

The modelling of nonlinear systems has been posed as the problem of selecting an approximate non-linear function

IEE Proceedings online no. 19990501

DOI: 10.1049/ip-cta:19990501

Paper first received 22nd July 1998 and in revised form 20th April 1999

G.P. Liu is with the Energy Technology Centre, ALSTOM, Leicester LE8 6LH, UK
E-mail: guoping.liu@energy.alstom.com

V. Kadirkamanathan is with the Department of Automatic Control & Systems Engineering, University of Sheffield, Sheffield S1 3JD, UK

between the inputs and the outputs of the systems. For a single-input single-output system, it can be expressed by the NARMAX model [18] that is,

$$y(t) = f(y(t-1), y(t-2), \ldots, y(t-n_y), u(t-1), u(t-2), \ldots, u(t-n_u)) + e(t) \quad (1)$$

where $f(\cdot)$ is an unknown nonlinear function, y is the output, u is the control input and e is the noise, respectively, n_y, n_u, are the corresponding maximum delays. It is assumed that the noise $e(t)$ is a white noise. For the colour noise case, the modelling of the system using neural networks below needs some slight modifications, as suggested in [19]. The nonlinear function $f(\cdot)$ in the above NARMAX model can be approximated by a single-layer neural network, i.e. a linear combination of a set of basis functions [20].

$$f^*(\mathbf{x}, \mathbf{p}) = \sum_{k=1}^{N} w_k f_k(\mathbf{x}, \mathbf{d}_k) \quad (2)$$

where

$$\mathbf{x} = [y(t-1), y(t-2), \ldots, y(t-n_y), u(t-1), u(t-2), \ldots, u(t-n_u)] \quad (3)$$

$f_k(\mathbf{x}, \mathbf{d}_k)$ ($k = 1, 2, \ldots, N$) is the basis function and $\mathbf{p}$ is the parameter vector containing the weights w_k and the basis function parameter vectors $\mathbf{d}_k$. If the basis functions $f_k(\mathbf{x}, \mathbf{d}_k)$ do not have the parameters $\mathbf{d}_k$, then it is denoted by $f_k(\mathbf{x})$. Two sets of basis functions are used in this paper: a set of the Volterra polynomial basis functions (VPBF) and a set of the Gaussian radial basis functions (GRBF).

Multivariate polynomial expansions have been suggested as a candidate for nonlinear system identification using the NARMAX model [3]. The Volterra polynomial expansion [21] has been cast into the framework of nonlinear system approximations and neural networks [5, 22]. A network whose basis functions consist of the Volterra polynomials is named as the Volterra polynomial basis function network. Its functional representation is given by

$$f(\mathbf{x}) = f^*(\mathbf{x}, \mathbf{p}) + O(\mathbf{x}^3) \quad (4)$$

$$\begin{aligned} f^*(\mathbf{x}, \mathbf{p}) &= a + \mathbf{x}^T\mathbf{b} + \mathbf{x}^T\mathbf{C}\mathbf{x} \\ &= a + b_1x_1 + b_2x_2 + \ldots + c_{11}x_1^2 + c_{12}x_1x_2 \\ &\quad + c_{22}x_2^2 + \ldots \\ &= [a, b_1, b_2, \ldots, c_{11}, c_{12}, c_{22}, \ldots] \\ &\quad \times [1, x_1, x_2, \ldots, x_1^2, x_1x_2, x_2^2, \ldots]^T \\ &= \sum_{k=1}^{N} w_k f_k(\mathbf{x}) \end{aligned} \quad (5)$$

where

$$[w_1, w_2, w_3, \ldots, w_{n+2}, w_{n+3}, w_{n+4}, \ldots, w_N] = [a, b_1, b_2, \ldots, c_{11}, c_{12}, c_{22} \ldots, c_{nn}], \quad (6)$$

$$[f_1, f_2, f_3, \ldots, f_{n+2}, f_{n+3}, f_{n+4}, \ldots, f_N] = [1, x_1, x_2, \ldots, x_1^2, x_1x_2, x_2^2, \ldots, x_n^2] \quad (7)$$

are the set of linear weights and the set of basis functions being linearly combined, respectively, and $\mathbf{x} \in \Re^n$.

Radial basis functions were introduced as a technique for multivariable interpolations [23], which can be cast into an architecture similar to that of the multilayer perceptron [24]. Radial basis function networks provide an alternative to the traditional neural network architectures and have good approximation properties. One of the commonly used radial basis function networks is the Gaussian radial basis function (GRBF) neural network, also called the localised receptive field network [25]. The nonlinear function approximated by the GRBF network is expressed by

$$f^*(\mathbf{x}, \mathbf{p}) = \sum_{k=1}^{N} w_k \exp(-(\mathbf{x} - \mathbf{d}_k)^T\mathbf{C}_k(\mathbf{x} - \mathbf{d}_k)) \quad (8)$$

where $\mathbf{C}_k$ is the weighting matrix of the kth basis function, and $\mathbf{p}$ is the parameter vector containing the weights w_k and the centres $\mathbf{d}_k (k = 1, 2, \ldots, N)$. For the sake of simplicity, let $\mathbf{C}_k = \mathbf{I}$ in this paper.

3 Model selection by GAs

Many different techniques are available for optimising the design space associated with various systems. In recent years, the direct-search techniques, which are problem-independent, have been proposed as a possible solution for the difficulties associated with the traditional techniques. One direct-search method is the genetic algorithm (GA) [26]. (In [26], it is stated that the GA searches from a population of points, not a single point and uses probabilistic and not deterministic transition rules.) Genetic algorithms are search procedures which emulate the natural genetics. They are different from traditional search methods encountered in engineering optimisation [27]. Recently, genetic algorithms have been applied to control system design (*e.g.* [27, 28]). GAs have also been successfully used with neural networks to determine the network parameters [29, 30], with NARMAX models [31] and for nonlinear basis function selection for identification using Bayesian criteria [7]. This paper applies the GA approach to the model selection and identification of nonlinear systems using multiobjective criteria as the basis for selection. The idea used here for model selection is similar to that proposed in [30] but is independently developed in a different way [32].

The model selection can be seen as a subset selection problem. For the model represented by the VPBF network, the principle of model selection using the genetic algorithms can be briefly explained as follows: For the vector $\mathbf{x} \in \Re^n$, the maximum number of the model terms is given by $N = (n+1)(n+2)/2$. Thus, there are N basis functions which are the combination of 1 and the elements of the vector $\mathbf{x}$. Then there are 2^N possible models for selection. Each model is expressed by an N-bit binary model code $\mathbf{c}$, i.e. a chromosome representation in genetic algorithms. If some bits of the binary model code $\mathbf{c}$ are zeros, it means that the basis functions corresponding to these zero bits are not included in the model. For example, if the vector $\mathbf{x} \in \Re^3$, the maximum number of the model terms is 10. Then there are 1024 possible models. Each model can be expressed by a 10-bit binary model code. Thus the Volterra polynomial basis functions are

$$\begin{aligned} \mathbf{f}^T &= [f_1, f_2, \ldots, f_{10}] \\ &= [1, x_1, x_2, x_3, x_1x_2, x_2x_3, x_1x_3, x_1^2, x_2^2, x_3^2] \end{aligned} \quad (9)$$

If the 10-bit binary model code is $\mathbf{c} = [1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0]$, the model can be written as

$$\begin{aligned} f^*(\mathbf{x}, \mathbf{p}) &= \mathbf{p}^T \text{ diag } (\mathbf{c})\mathbf{f} \\ &= [p_1, p_4, p_7, p_9][f_1, f_4, f_7, f_9]^T \\ &= p_1 + p_4x_3 + p_7x_1x_3 + p_9x_2^2 \end{aligned} \quad (10)$$

For the model represented by GRBF network, the maximum number of the model terms is given by N, the number of the Gaussian functions, and there are 2^N possible models for selection and also N possible radial basis functions with their centres $\mathbf{d}_k$. Thus a chromosome representation in genetic algorithms consists of an N-bit binary model code $\mathbf{c}$ and N real number basis function centres $\mathbf{d}_k$ ($k=1, 2, \ldots, N$), i.e.,

$$[\mathbf{c}, \mathbf{d}_1^T, \mathbf{d}_2^T, \ldots, \mathbf{d}_N^T] \tag{11}$$

For example, if $N=5$, $\mathbf{x} \in \Re_2$ and the chromosome

$$[[0\,1\,0\,0\,1], [d_{11}, d_{12}], \ldots, [d_{51}, d_{52}]] \tag{12}$$

then the model is given by

$$f^*(\mathbf{x}, \mathbf{p}) = w_2 \exp\left(-\sum_{j=1}^{2}(x_j - d_{2j})^2\right) + w_5 \exp\left(-\sum_{j=1}^{2}(x_j - d_{5j})^2\right) \tag{13}$$

It is evident that only the basis functions corresponding to the nonzero bits of the binary model code $\mathbf{c}$ are included in the selected model. Given a parent set of binary model codes and basis function parameter vectors, a model satisfying a set of performance criteria is sought by the numerical algorithm in Section 5.

4 Multiobjective performance criteria

This section presents multiobjective performance criteria for nonlinear model selection and identification. Let us define the following performance functions:

$$\phi_1(\mathbf{p}) = \|f(\mathbf{x}) - f^*(\mathbf{x}, \mathbf{p}\|_2 \tag{14}$$

$$\phi_2(\mathbf{p}) = \|f(\mathbf{x}) - f^*(\mathbf{x}, \mathbf{p})\|_\infty \tag{15}$$

$$\phi_3(\mathbf{p}) = \sigma(\mathbf{c}) \tag{16}$$

where $\|\cdot\|_2$ and $\|\cdot\|_\infty$ are the L_2- and L_∞-norms of the function $(\cdot)$, $\sigma(\mathbf{c})$ is the number of the non-zero elements in the binary model code $\mathbf{c}$. For model selection and identification of nonlinear systems, there are good reasons for giving attention to the performance functions $\phi_i(\mathbf{p})(i=1, 2, 3)$. The practical reasons for considering the performance function $\phi_1(\mathbf{p})$ is even stonger than the other performance functions $\phi_2(\mathbf{p})$ and $\phi_3(\mathbf{p})$. Statistical considerations show that it is the most appropriate choice for data fitting when errors in the data have a normal distribution. Often the performance function $\phi_1(\mathbf{p})$ is preferred because it is known that the best approximation calculation is straightforward to solve. The performance function $\phi_2(\mathbf{p})$ provides the foundation of much of approximation theory. It shows that when this is small, the performance function $\phi_1(\mathbf{p})$ is small also. But the converse statement may not be true. A practical reason for using the performance function $\phi_2(\mathbf{p})$ is based on the following. In practice, an unknown complicated nonlinear function is often estimated by one that is easy to calculate. Then it is usually necessary to ensure that the greatest value of the error function is less than a fixed amount, which is just the required accuracy of the approximation. The performance function $\phi_3(\mathbf{p})$ is used as a measure of the model complexity. A smaller performance function $\phi_3(\mathbf{p})$ indicates a simpler model in terms of the number of unknown parameters used. Under similar performances in $\phi_1(\mathbf{p})$ and $\phi_2(\mathbf{p})$ by two models, the simpler model is statistically likely to be a better model [33].

To give a feel for the usefulness of the multiobjective approach as opposed to single-objective design techniques, let us consider the minimisation of the cost functions $\phi_i(\mathbf{p})(i=1, 2, 3)$. Let the minimum value of ϕ_i be given by ϕ_i^*, for $i=1, 2, 3$, respectively. For these optimal values ϕ_i^* there exist corresponding values given by $\phi_j[\phi_i^*](j \neq i, j=1, 2, 3)$, for $i=1, 2, 3$, respectively, and the following relations hold:

$$\min\{\phi_1[\phi_2^*], \phi_1[\phi_3^*]\} \geq \phi_1^* \tag{17}$$

$$\min\{\phi_2[\phi_1^*], \phi_2[\phi_3^*]\} \geq \phi_2^* \tag{18}$$

$$\min\{\phi_3[\phi_1^*], \phi_3[\phi_2^*]\} \geq \phi_3^* \tag{19}$$

If one of the performance functions ϕ_i $(i=1, 2, 3)$ is minimised individually (single-objective approach), unacceptably large values may result for other performance functions $\phi_j(j \neq i, j=1, 2, 3)$. Generally, there does not exist a solution for all performance functions $\phi_i(\mathbf{p})$ for $i=1, 2, 3$ to be minimised by the same parameter vector $\mathbf{p}$. Following the method of inequalities [34, 35], we reformulate the optimisation into a multiobjective problem as

$$\phi_i(\mathbf{p}) \leq \varepsilon_i, \quad \text{for } i = 1, 2, 3 \tag{20}$$

where the positive real number ε_i represents the numerical bound on the performance function $\phi_i(\mathbf{p})$ and is determined by the designer. Generally speaking, the number ε_i is chosen to be a reasonable value corresponding to the performance function ϕ_i according to the requirements of the practical system. For example, ε_1 should be chosen between the minimum of ϕ_1 and the practical tolerable value on ϕ_1. The minimum of ϕ_1 can be known by the least-squares algorithm. The practical tolerable value means if ϕ_1 is greater than it, the modelling result can not be accepted. In addition, if ε_i is chosen to be an unreachable value, Section 5 shows how to deal with this problem.

5 Numerical algorithm

With three objectives (or cost functions) for model selection and identification, the numerical algorithm is not a straightforward optimisation algorithm, such as for the least-squares algorithm. This section develops the numerical algorithm which uses genetic algorithm approaches and the method of inequalities to get a numerical solution satisfying the performance criteria.

Now, we normalise the multiobjective performance functions as the following:

$$\psi_i(\mathbf{p}) = \frac{\phi_i(\mathbf{p})}{\varepsilon_i} \tag{21}$$

Let Γ_i be the set of parameter vectors $\mathbf{p}$ for which the ith performance criterion is satisfied

$$\Gamma_i = \{\mathbf{p} : \psi_i(\mathbf{p}) \leq 1\} \tag{22}$$

Then the admissible or feasible set of parameter vectors for which all the performance criteria hold is the intersection

$$\Gamma = \Gamma_1 \cap \Gamma_2 \cap \Gamma_3 \tag{23}$$

Clearly, $\mathbf{p}$ is an admissible parameter vector if and only if

$$\max\{\psi_1(\mathbf{p}), \psi_2(\mathbf{p}), \psi_3(\mathbf{p})\} \leq 1 \tag{24}$$

which shows that the search for an admissible $\mathbf{p}$ can be pursued by optimization, in particular by solving

$$\min_{\mathbf{p}}\{\max\{\psi_1(\mathbf{p}), \psi_2(\mathbf{p}), \psi_3(\mathbf{p})\}\} \tag{25}$$

subject to Eqn. 24. The optimisation needs to be carried out using iterative schemes. let $\mathbf{p}^q$ be the value of the parameter vector at the qth iteration step in optimisation, and define

$$\Gamma_i^q = \{\mathbf{p} : \psi_i(\mathbf{p}) \leq \Delta^q\}, \quad \text{for} \quad i = 1, 2, 3 \tag{26}$$

where

$$\Delta^q = \max\{\psi_i(\mathbf{p}^q)\} \tag{27}$$

and also define

$$\Gamma^q = \Gamma_1^q \cap \Gamma_2^q \cap \Gamma_3^q \tag{28}$$

$$E^q = \psi_1(\mathbf{p}^q) + \psi_2(\mathbf{p}^q) + \psi_3(\mathbf{p}^q) \tag{29}$$

Γ^q is the qth set of parameter vectors for which all performance functions satisfy

$$\psi_i(\mathbf{p}) \leq \Delta^q, \quad \text{for} \quad i = 1, 2, 3 \tag{30}$$

It is clear that Γ^q contains both $\mathbf{p}^q$ and the admissible set Γ. E^q is a combined measurement of all performance functions. If we find a new parameter vector $\bar{\mathbf{p}}^q$, such that

$$\bar{\Delta}^q < \Delta^q \tag{31}$$

or

$$\bar{\Delta}^q = \Delta^q \quad \text{and} \quad \bar{E}^q < E^q \tag{32}$$

where $\bar{\Delta}^q$ and $\bar{E}^q$ are defined similarly to Δ^q and E^q, then we accept $\bar{\mathbf{p}}^q$ as the next value of the parameter vector. Then we set $\mathbf{p}^{q+1} = \bar{\mathbf{p}}^q$. We then have

$$\psi_i(\mathbf{p}^{q+1}) \leq \psi_i(\mathbf{p}^q), \quad \text{for} \quad i = 1, 2, 3 \tag{33}$$

and

$$\Gamma \subset \Gamma^{q+1} \subset \Gamma^q \tag{34}$$

so that the boundary of the set in which the parameters are located has been moved towards the admissible set, as shown in Fig. 1. The process of finding the optimisation solution is terminated when both Δ^q and E^q cannot be reduced any further. But the process of finding an admissible parameter vector $\mathbf{p}$ stops when

$$\Delta^q \leq 1 \tag{35}$$

i.e., when the boundaries of Γ^q have converged to the boundaries of Γ. Eqn. 35 is always achievable if ε_i properly set, for $i = 1, 2, 3$. On the other hand, if the Δ^q persists in being larger than 1, this may be taken as an indication that the performance criteria may be inconsistent, while their magnitude gives some measure of how closely it is possible to approach the objectives. In this case, some of the parameters ε_i need to be increased. Generally speaking, the parameter ε_i corresponding to the largest normalized

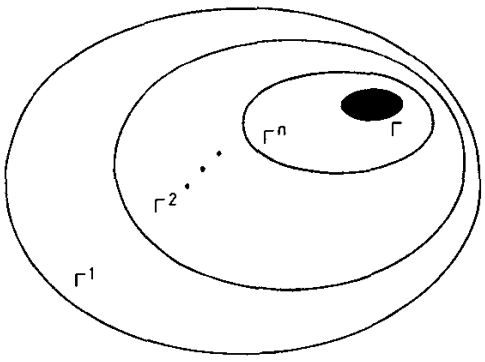

Fig. 1 *Movement of boundary of set Γ^q*

performance function $\psi_i(\mathbf{p}^q)$ should be considered to increase first and then that corresponding to the second largest one and so on. This means that some of the performance criteria should be relaxed until they are satisfied. From a practical viewpoint, the approximate optimal solution is also useful if the optimal solution is not achievable. Genetic algorithms have been used in multiobjective optimisation and have provided better results over conventional search methods [27, 36, 37] Here, we combine genetic algorithms with that of least squares in deriving the estimation algorithm.

The steps of the identification algorithm to be executed for the GA implementation are as follows:

Step 1: Chromosomal representation
Each chromosome in the population consists of an N-bit binary model code c and a real number basis function parameter vector $\mathbf{D}$, where N is the number of the basis functions for the nonlinear model selection. For example, for the VPBF network there is not the vector $\mathbf{D}$ and for the GRBF network the vector $\mathbf{D}$ contains all basis function centres $\mathbf{d}_k (k = 1, 2, \ldots, N)$, i.e., $\mathbf{D} = [\mathbf{d}_1^T, \mathbf{d}_2^T, \ldots, \mathbf{d}_N^T]$

Step 2: Generation of initial population
The M chromosomes $[\mathbf{c}, \mathbf{D}]$ for the initial population are randomly generated, where M is the population size and is often chosen to be an odd number.

Step 3: Evaluation of performance functions
Given the j-th binary model code $\mathbf{c}_j$ and basis function parameter vector $\mathbf{D}_j$, then the jth nonlinear model is known. Using the least squares algorithm, the jth weight vector $\mathbf{w}_j$ can be computed easily, based on the data of the vector $\mathbf{x}$, the binary model code $\mathbf{c}_j$ and the basis function parameter vector $\mathbf{D}_j$. Then evaluate the normalised performance functions $\psi_i(\mathbf{s}_j)(i = 1, 2, 3)$, where $\mathbf{s}_j = [\mathbf{w}_j, \mathbf{c}_j, \mathbf{D}_j]$, and

$$\Delta_j = \max_{i=1,2,3} \psi_i(\mathbf{s}_j) \tag{36}$$

$$E_j = \sum_{i=1}^{3} \psi_i(\mathbf{s}_j) \tag{37}$$

These computations are completed for all M sets of chromosomes, i.e. $j = 1, 2, \ldots, M$.

Step 4: Selection
According to the fitness of the performance functions for each chromosome, delete the $(M - 1)/2$ weaker members of the population and reorder the chromosomes. The fitness of the performance functions is measured by

$$F_j = \frac{1}{\Delta_j}, \quad \mathbf{forj} = \mathbf{1}, \mathbf{2}, \ldots, \mathbf{M} \tag{38}$$

Step 5: Crossover
Offspring binary model codes are produced from two-parent binary model codes so that their first half elements are preserved. The second half elements in each parent are exchanged. The average crossover operator is used to produce offspring basis function parameter vectors. The average crossover function is defined as

$$\frac{\mathbf{d}_{j+1} + \mathbf{d}_j}{2}, \quad \text{for } j = 1, 2, \ldots, \frac{M-1}{2} \tag{39}$$

Then the $(M - 1)/2$ offsprings are produced.

Step 6: Mutation
A mutation operator, called creep [27], is used. For the binary model codes, it randomly replaces one bit in each offspring binary model code with a random number 1 or 0.

For the offspring basis function parameter vectors, the mutation operation is defined as

$$\mathbf{D}_j + \beta\xi_j, \quad \text{for } j = 1, 2, \ldots, \frac{M-1}{2} \tag{40}$$

where β is the maximum to be altered and $\xi_j \in [-1, 1]$ is a random variable with zero mean.

Step 7: Elitism
The elitist strategy copies the best chromosome into the succeeding generation. It prevents the best chromosome being lost in the next generation. It may increase the speed of domination of a population by a super individual, but on balance it appears to improve genetic algorithm performance. The best chromosome is defined as one satisfying

$$E_b = \min_{l \in \{1,2,\ldots,M\}} \{E_l : E_l \leq E_m - \alpha(\Delta_l - \Delta_m) \quad \text{and} \quad \Delta_l \leq \Delta_m + \delta\} \tag{41}$$

where

$$\Delta_m = \min_{j=1,2,\ldots,M} \{\Delta_j\} \tag{42}$$

E_m and E_l are corresponding to Δ_m and Δ_l, which are defined in Eqn. 37. $\alpha > 1$ and $\delta \ll \alpha$ is a small positive number, which are given by the designer. α and δ are chosen such that $\alpha\delta > \delta$, e.g. $\alpha = 1.1$ and $\delta = 0.05$. This means that sacrificing Δ_m a bit makes significant improvement on E_b. Thus, the best chromosome is one that has the smallest E_b at the neighbourhood of the E_m.

Step 8: New offsprings
Add the $(M-1)/2$ new offsprings to the population which are generated in a random fashion. Actually, the new offsprings are formed by replacing randomly some elements of the best binary model code and mutating the best basis function parameter vector with a probability

Step 9: Stop check
Continue the cycle initiated in Step 3 until local convergence of the algorithm is achieved. This local convergence is defined as the population satisfying

$$\Delta_j - \Delta_b \leq \varepsilon \quad \text{for} \quad j = 1, 2, \ldots, (M-1)/2 \tag{43}$$

where Δ_b is corresponding to E_b, and ε is a positive number. This implies that the difference between the chromosomes in the first half population and the best chromosome is small in the sense of their performance measurement Δ_j.

Take the best solution in the converged generation and place it in a second 'initial generation'. Generate the other $M-1$ chromosomes in this second initial generation at random and begin the cycle again until a satisfactory solution is obtained or Δ_b and E_b cannot be reduced any further. In addition, for mixed noise distribution, the least squares algorithm in step 3 should be replaced by a more robust modified least squares algorithm as suggested in [38].

6 Examples

The first example one considers identification of a real system. The second demonstrates approximation of a nonlinear function with a mixed noise with different variance.

Table 1: Algorithm parameters

Parameter	VPBF Network	GRBF Network
Model term number N	45	10
Chromosome length	45	50
Variable vector x	$\begin{bmatrix} y(t-1) \\ y(t-2) \\ y(t-3) \\ y(t-4) \\ u(t-1) \\ u(t-2) \\ u(t-3) \\ u(t-4) \end{bmatrix}$	$\begin{bmatrix} y(t-1) \\ y(t-2) \\ u(t-1) \\ u(t-2) \end{bmatrix}$
ε_1	1.5	1.5
ε_2	0.3	0.3
ε_3	7	7

6.1 Example 1

We use the data generated by a large-scale pilot liquid-level nonlinear system with zero mean gaussian input signal [31]. 1000 pairs of input-output data were collected. The first 500 pairs were used in the model selection and identification of the system, while the remaining 500 pairs for validation test. The Volterra polynomial basis function network and the gaussian radial basis function network were applied to select and identify the model of the system by the numerical algorithm developed in Section 5.

The time lags n_y and n_u were obtained by a trial and error process based on estimation of several models. During the simulation, it was found that for the VPBF network, if n_y and n_u were greater than 4, the performance functions improved very little. Similarly, for the GRBF network, if n_y and n_u were greater than two the performance functions did not reduce significantly. It is clear that the time lags n_y and n_u for the VPBF network are different from those for the GRBF network. The main reason is those two networks use different kinds of basis functions which have different properties. The parameters for the algorithm are given in Table 1.

VPBF Network

Since the maximum number of the model terms is 45, there are 2^{45} possible models for selections. But, after 210 generations optimal model has been found by the algorithm. The performance functions are

$$\phi_1(\mathbf{p}) = 1.8000, \quad \phi_2(\mathbf{p}) = 0.3965, \quad \phi_3(\mathbf{p}) = 3 \tag{44}$$

The model represented by the VPBF network is

$$y(t) = 1.3234y(t-1) - 0.3427y(t-2) + 0.075y(t-4)u(t-2) \tag{45}$$

The convergence of the performance functions with respect to generations are given in Figs. 2 and 3. It shows that the performance functions converge in about 100 generations. In fact, in generation 94, the performance functions are $\phi_1 = 1.8119$, $\phi_2 = 0.4071$, and $\phi_3 = 3$. After that, no improvement is made until in generation 208 $\phi_1 = 1.8$, $\phi_2 = 0.3965$ and $\phi_3 = 3$. The measured and estimated outputs, and the residual error of the system for the training data are shown in Fig. 4. The measured and estimated outputs, and estimation error of the system for the validation test of the model identified via the VPBF network is illustrated in Fig. 5. Clearly, the performance functions ϕ_1 and ϕ_2 are very close to the desired requirements. But they do not satisfy them. This may be result from the general

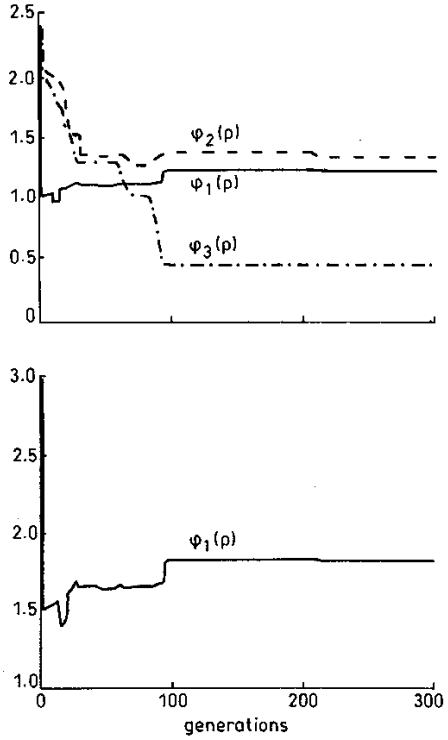

Fig. 2 *Convergence of performance functions using VPBF network*

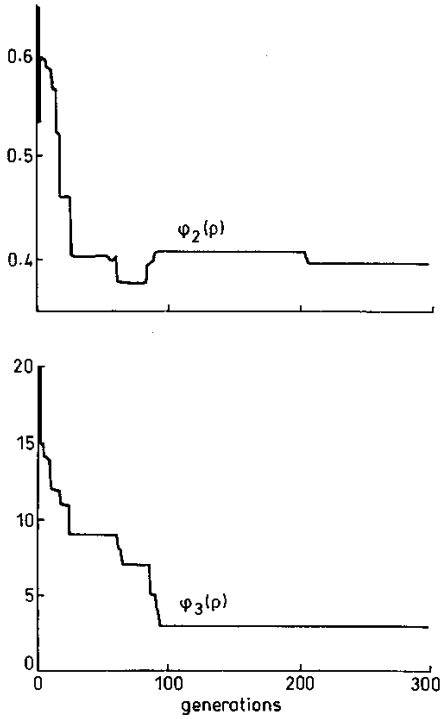

Fig. 3 *Convergence of performance functions using VPBF network*

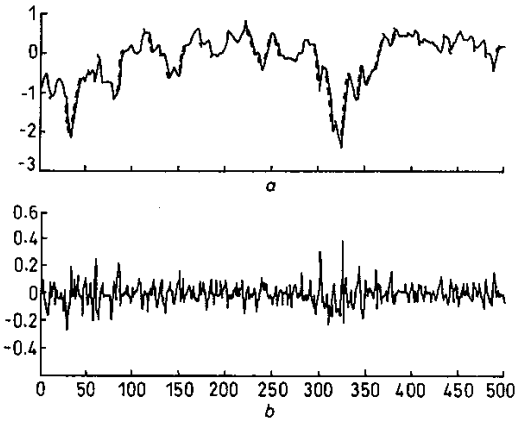

Fig. 4 *Training data results for system using VPBF network*
a Measured and estimated outputs
b Estimation error

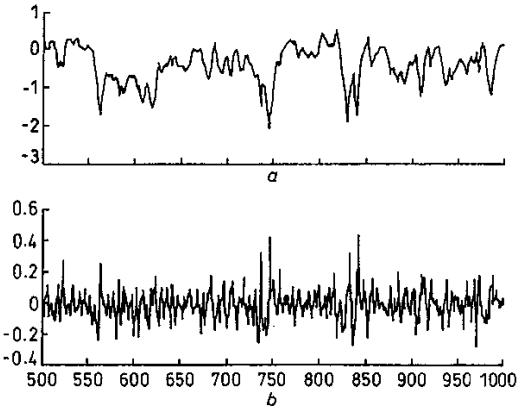

Fig. 5 *Validation results for the system using VPBF network*
a Measured and estimated outputs
b Estimation error

drawback (premature convergence) of the genetic algorithms.

The GRBF Network

Although the maximum number of the model terms is only 10 (i.e. 1024 possible models for selection), the search dimension of the basis function centre parameters is 40 in real number space (i.e. infinite possibilities for selection). After 700 generations the performance criteria are almost satisfied. At this stage, $\phi_1(\mathbf{p}) = 1.5643$, $\phi_2(\mathbf{p}) = 0.2511$, $\phi_3(\mathbf{p}) = 5$. To obtain the better performance, the basis function parameter vector was searched for another 100 generations using the algorithm with the fixed number of the model terms, i.e. let $\phi_3(\mathbf{p}) = 5$ for this case. Finally, the performance functions are

$$\phi_1(\mathbf{p}) = 1.2957, \quad \phi_2(\mathbf{p}) = 0.1724, \quad \phi_3(\mathbf{p}) = 5 \qquad (46)$$

The model represented by the GRBF network is

$$y(t) = \sum_{i=1}^{5} w_i \exp\left(-\sum_{j=1}^{2}(y(t-j) - d_{ij})^2 - \sum_{j=1}^{2}(u(t-j) - d_{ij})^2\right) \qquad (47)$$

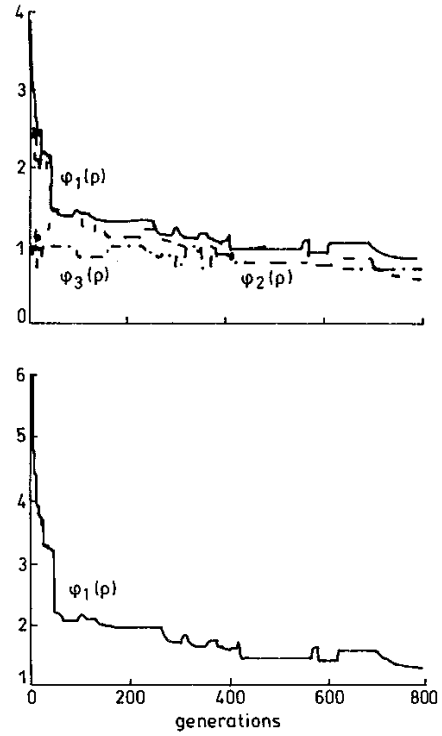

Fig. 6 *Convergence of the performance functions using GRBF network*

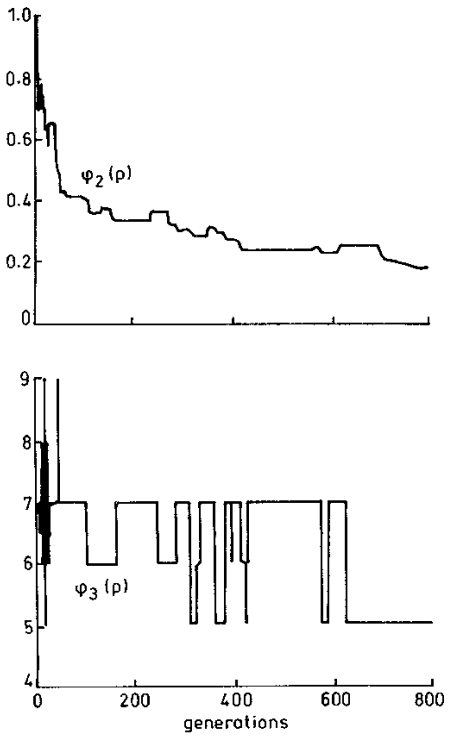

Fig. 7 *Convergence of the performance functions using GRBF network*

where

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \end{bmatrix} = \begin{bmatrix} -2.6363 \\ -1.2470 \\ -1.7695 \\ 0.9437 \\ -0.5341 \end{bmatrix},$$

$$\{d_{ij}\} = \begin{bmatrix} -2.1577 & -1.8855 & -0.8975 & -0.2841 \\ -1.2717 & -2.2730 & 0.3445 & 0.3315 \\ -0.6345 & -1.1223 & -1.1615 & -0.3666 \\ 0.7344 & 1.0223 & 0.5469 & 0.1989 \\ -1.2336 & -0.5928 & 0.3212 & 0.5754 \end{bmatrix} \quad (48)$$

The performance of the GRBF network is shown in Figs. 6–9. Figs. 6 and 7 shows the convergence of the performance functions with respect to generations. The measured and estimated outputs, and residual error of the system for the training data for the model identified via the GRBF network is shown in Fig. 8. The measured and estimated outputs, and estimation error of the system for the validation test data for the model identified via the GRBF network are shown in Fig. 9.

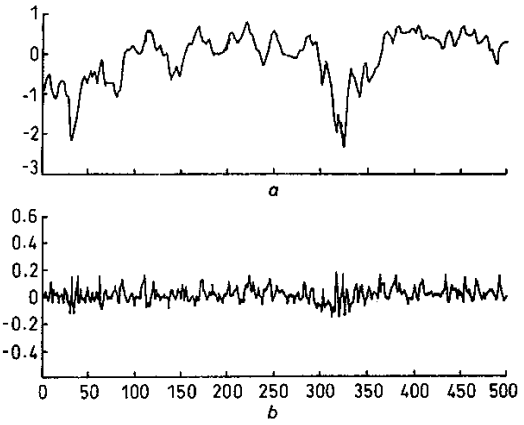

Fig. 8 *Training data results for system using GRBF network*
a Measured and estimated outputs
b Estimation error

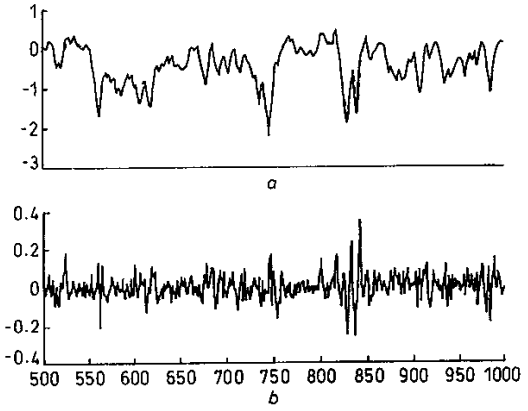

Fig. 9 *Validation results for system using GRBF network*
a Measured and estimated outputs
b Estimation error

To see the importance of the L_∞ norm measure of the accuracy bound of the approximation, the performance function $\phi_2(p)$ is not used in the next simulation. So, only two performance functions ϕ_1 and ϕ_3 are considered. Their required upper bounds ε_1 and ε_3 are still set to be 1.5 and 7. The simulation procedure is exactly the same as the above. The following performance is obtained:

$$\phi_1(\mathbf{p}) = 1.2900, \quad \phi_3(\mathbf{p}) = 4 \tag{49}$$

The weight vector and centres of the network are

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \begin{bmatrix} 1.2394 \\ -2.4092 \\ -2.8293 \\ -2.5141 \end{bmatrix},$$

$$\{d_{ij}\} = \begin{bmatrix} 1.3219 & 0.4971 & 0.4451 & 0.0935 \\ -0.5826 & -2.1796 & 0.2636 & 0.5724 \\ -1.6041 & -0.0912 & -0.7477 & -0.0275 \\ -2.1362 & -1.9554 & -0.7189 & -0.2974 \end{bmatrix} \tag{50}$$

The simulation results are shown in Figs. 10–12. It is clear from the results that although the performance functions ϕ_1 and ϕ_3 are reduced, the maximum difference ϕ_2 of the approximation for identification and validity test is much greater than the previous case. So it shows that if the performance functions ϕ_1 and ϕ_3 are sacrificed somewhat, the performance function ϕ_2 is improved significantly.

The selection, identification and validation results for the large pilot scale liquid-level nonlinear system show that the VPBF network is simpler than the GRBF network, but the performance of the latter is better than that of the former. However, it is difficult to conclude that the GRBF

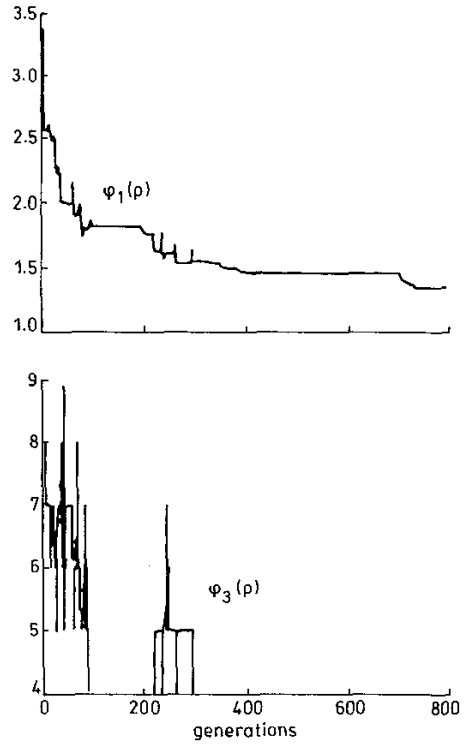

Fig. 10 *Convergence of performance functions using GRBF network without* ϕ_2

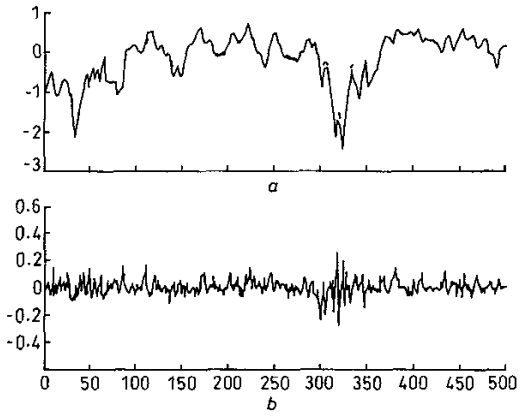

Fig. 11 *Training data results for system using GRBF network without* ϕ_2
a Measured and estimated outputs
b Estimation error

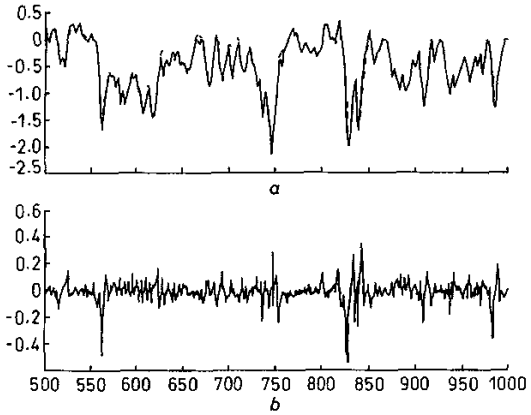

Fig. 12 *Validation results for system using GRBF network without* ϕ_2
a Measured and estimated outputs
b Estimation error

model is better than the VPBF model or *vice versa*. On the same set of experiments, the Bayesian method selection and identification with Gaussian noise assumptions leads to very similar performance as previously but needed 11 and 16 basis functions (hidden units) for the VPBF and GRBF networks [7]. The identified model here is much simpler.

6.2 Example 2

Consider the following underlying nonlinear function to be approximated.

$$f^*(x) = 1.1(1 - x + x^2)\exp(-0.5x^2) \tag{51}$$

where x is a variable. A random sampling of the interval $[-4, 4]$ is used in obtaining the 40 input–output data for approximation. To see the effect of noise, the output of the function f to a given input x is given by

$$f(x) = f^*(x) + e \tag{52}$$

where e is a mixed noise. The noise consists of uniformly and normally distributed noises, i.e.

$$e = \frac{1}{\sqrt{2}}(e_{U[0,\sigma]} + e_{D[0,\sigma]}) \tag{53}$$

where $e_{U[0,\sigma]}$ is a zero mean uniform noise with finite variance σ and $e_{D[0,\sigma]}$ is a zero mean normal noise with finite variance σ. It is assumed that the uniform noise $e_{U[0,\sigma]}$ and the normal noise $e_{D[0,\sigma]}$ are uncorrelated. Thus, the mean and variance of the mixed noise e are zero and σ, respectively. Here, the Gaussian radial basis function network was used to approximate the nonlinear function by the numerical algorithm developed in Section 5. Three cases were considered in this simulation. The first used three performance functions $[\phi_1(p), \phi_2(p), \phi_3(p)]$ during approximation. The second considered two performance functions $[\phi_1(p), \phi_3(p)]$. The third used only one performance function $[\phi_1(p)]$. The effects of the mixed noise with different variance on the performance functions $\phi_1(p)$, $\phi_2(p)$ and $\phi_3(p)$ for the three cases are illustrated in Figs. 13–15, respectively. The performance of the approximation of the nonlinear function changes little at low-level variance of noise and the multiobjective case using three performance criteria gives a good approximation even though three performance functions conflict each other.

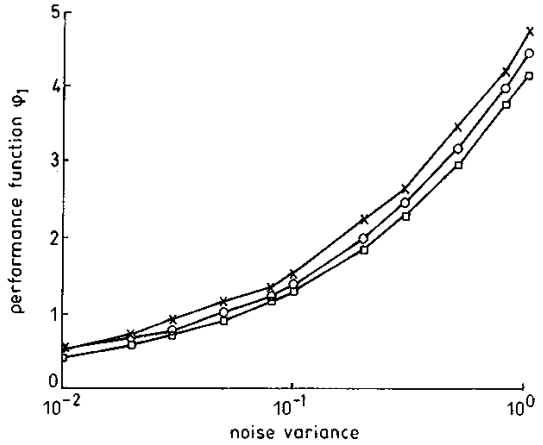

Fig. 13 *Performance function $\phi_1(p)$ against noise variance σ*
—×— φ_1, φ_2, φ_3
—○— φ_1, φ_3
—□— φ_1

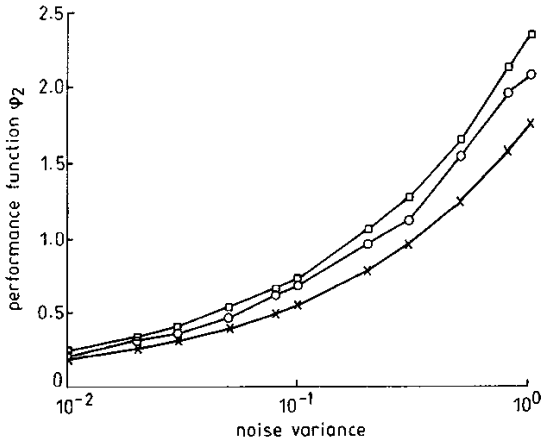

Fig. 14 *Performance function $\phi_2(p)$ against noise variance σ.*
—×— φ_1, φ_2, φ_3
—○— φ_1, φ_3
—□— φ_1

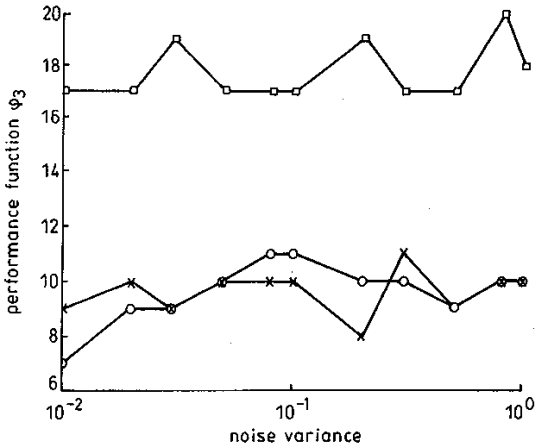

Fig. 15 *Performance function ϕ_3 (p) against the noise variance σ*
—×— φ_1, φ_2, φ_3
—○— φ_1, φ_3
—□— φ_1

7 Conclusions

This paper has addressed the problems of model selection and identification of nonlinear systems using neural networks, genetic algorithms and multiobjective optimisation techniques. A set of performance functions that measure approximation accuracy and a model complexity measure are used as the multiobjective criteria in the identification task. The optimisation is carried out using genetic algorithms which selects the nonlinear function units to arrive at the simplest model necessary for approximation, along with optimising the multiobjective criteria. The Volterra polynomial basis function network and the gaussian radial basis function are subjected to the algorithm in the task of a liquid-level nonlinear system identification. The model selection procedure results in determining the relevant linear and second-order nonlinear terms for the VPBF model and in selection of the basis function centres for the GRBF model. The experimental results demonstrate the convergence of the developed algorithm and its ability to arrive at a simple model which approximates the nonlinear system well. The approach developed can also be extended in many ways, for example, adaptively modify the numerical bounds on the performance functions. Furthermore, cross-validation techniques can be used to guide the optimisation and also in the adaptation of the bounds on the performance functions.

8 Acknowledgments

The authors thank Professor S.A. Billings for his assistance and gratefully acknowledge the support of Engineering and Physical Science Research Council (EPSRC) of UK under the contract GR/J46661. The authors wish to thank the reviewers and editors for their valuable suggestions.

9 References

1 POWELL, M.J.D: 'Approximation Theory and Methods' (Cambridge University Press, 1981)
2 SPRECHER, D.A.: 'On the structure of continuous functions of several variables', *Trans. Am. Math. Soc.*, 1965, **115**, pp. 340–355
3 BILLINGS, S.A., and CHEN, S.: 'Neural Networks and System Identification', *in* WARWICK, K. et al. (Eds.): 'Neural networks for systems and control', 1992, pp. 181–205
4 CHEN, S., BILLINGS, S.A., and GRANT, P.M.: 'Nonlinear system identification using neural networks', *Int. J. Control*, 1990, **51**, (6), pp. 1191–1214

5 KADIRKAMANATHAN, V.: 'Sequential learning in artificial neural networks', 1991. Ph. D Thesis, University of Cambridge, U.K.
6 KADIRKAMANATHAN, V., and NIRANJAN, M.: 'A function estimation approach to sequential learning with neural networks', *Neural Comp.*, 1993, **5**, pp. 954–957
7 KADIRKAMANATHAN, V.: 'Bayesian inference for basis function selection in nonlinear system identification using genetic algorithms', *in* SKILLING, J. and SIBISI, S. (Eds.) 'Maximum entropy and Bayesian methods' (Kluwer, 1995)
8 KUSCHEWSKI, J.G., HUI, S., and ZAK, H.S.: 'Application of feedforward neural networks to dynamical system identification and control', *IEEE Trans. Control Syst. Technol.*, 1993, **1**, (1), pp. 37–49
9 LIU, G.P., KADIRKAMANATHAN, V., and BILLINGS, S.A.: 'Sequential identification of nonlinear systems by neural networks', Proceedings of the 3rd European *Control* Conference, Rome, 1995, pp. 2408–2413
10 LIU, G.P., KADIRKAMANATHAN, V., and BILLINGS, S.A.: 'Stable sequential identification of continuous nonlinear dynamical systems by growing RBF networks', *Int. J. Control*, **65**, (1), pp. 53–69
11 NARENDRA, K.S., and PARTHASARATHY, K.: 'Identification and control of dynamical systems using neural networks', *IEEE Trans. Neural Netw.*, 1990, **1**, (1), pp. 4–27
12 QIN, S.Z., SU, H.T., and MCAVOY, T.J.: 'Comparison of four net learning methods for dynamic system identification', *IEEE Trans. Neural Net.*, 1992, **3**, (1), pp. 122–130
13 WILLIS, M.J., MONTAGUE, G.A., MASSIMO, C. Di., THAM, M.T., and MORRIS, A.J.: 'Artificial neural networks in process estimation and control, *Automatica*, 1992, **28**, (6), pp. 1181–1187
14 CYBENKO, G.: 'Approximation by superpositions of a sigmoidal function', *Math. Control Signals Syst.*, 1989, **2**, pp. 303–314
15 AKAIKE, H.: 'A new look at the statistical model identification', *IEEE Trans.*, 1974, **AC-19**, pp. 716–723
16 MACKAY, D.J.C.: 'Bayesian interpolation', *Neural Comput.*, 1992, **4**, pp. 415–447
17 RISSANEN, J.: 'Stochastic Complexity in Statistical Inquiry' (World Scientific, Singapore, 1989)
18 CHEN, S., and BILLINGS, S.A.: 'Representations of non-linear systems: the NARMAX model', *Int. J. Control.*, 1989, **49**, (3), pp. 1013–1032
19 NERRAND, O., ROUSSELRAGOT, P., PERSONNAZ, L., and DREYFUS, G.: 'Training recurrent neural networks: why and how? An illustration in dynamical process modeling', *IEEE Trans. Neural Netw.*, 1994, **5**, (2), pp. 178–184
20 LIU, G.P., KADIRKAMANATHAN, V., and BILLINGS, S.A.: 'On-line identification of nonlinear systems using Volterra polynomial basis function neural networks', *Neural Netw.*, 1998, **11**, pp. 1645–1657
21 SCHETZEN, M.: 'The Volterra and Wiener theories of nonlinear systems' (Wiley, New York, 1980)
22 RAYNER, P.J., and LYNCH, M.: 'A new connectionist model based on a nonlinear adaptive filter'. Proceedings of the International Conference on *Acoustics, speech and signal processing*, Glasgow, UK, 1989, pp. 1191–1194
23 POWELL, M.J.D.: 'Radial basis functions for multivariable interpolation: A review', *in* MASON, J.C. and COX, M.G. (Eds.): 'Algorithms for Approximation' (Oxford University Press, 1987) pp. 143–167
24 BROOMHEAD, D.S., LOWE, D.: 'Multivariable functional interpolation and adaptive networks', *Complex Systems*, 1988, **2**, pp. 321–355
25 MOODY, J.E., and DARKEN, C.J.: 'Fast learning in networks of locally-tuned process units', *Neural Comput.*, **1**, 1989, pp. 281–294
26 GOLDBERG, D.E.: 'Genetic Algorithm in Search, Optimization, and Machine Learning' (Addison Wesley, 1989)
27 DAVIS, L., (Ed.): 'Handbook of Genetic Algorithms' (Van Nostrand Reinhold, New York, 1991)
28 PATTON, R.J., and LIU, G.P., 'Robust control design via eigenstructure assignment, genetic algorithms and gradient-based optimization', *IEE Proc. Control Theory Appl.*, 1994, **141**, (3), pp. 202–208
29 SCHAFFER, J.D., CARUANA, R.A., and ESHELMAN, L.J.: 'Using genetic search to exploit the emergent behavior of Neural Networks', *Physica D*, 1990, **42**, (1–3), pp. 244–248
30 WHITEHEAD, B.A., and CHOATE, T.D.: 'Evolving space-filling curves to distribute radial basis functions over an input space', *IEEE Trans. Neural Net.*, 1994, **5**, (1), pp. 15–23
31 FONSECA, C.M., MENDES, E.M., FLEMING, P.J., and BILLINGS, S.A.: 'Nonlinear model term selection with genetic algorithms', *in* Proceedings of IEE/IEEE workshop on *Natural algorithms for signal processing*, 1993, 27/1–27/8,
32 LIU, G.P., and KADIRKAMANATHAN, V.: 'Multiobjective criteria for nonlinear model selection and identification with neural networks', Report No. 508, Automatic Control and Systems Engineering Department, 1994, University of Sheffield
33 GEMAN, S., BIENESTOCK, E., and BOURSAT, R.: 'Neural networks and the bias/variance dilemma', *Neural Comput.*, 1992, **4**, pp. 1–58
34 WHIDBORNE, J.F., and LIU, G.P.: 'Critical Control Systems: Theory, Design and Applications' (Research Studies Press Limited, U.K., 1993)
35 ZAKIAN, V., and AL-NAIB, U.: 'Design of dynamical and control systems by the method of inequalities', *Proc. IEE*, 1973, **120**, pp. 1421–1427
36 HAJELA, P., and LIN, C.Y.: in 'Genetic search strategies in multicriterion optimal design', *Struct. Optim.*, 1992, **4**, pp. 99–107
37 SCHAFFER, J.D., ELBAUM, L.: 'Multiple objective optimization with vector valued genetic algorithms', *in* GREFENSTETTE, J.J (Ed.): Proceedings of the First International Conference on *Genetic Algorithms*, 1985, pp. 93–100
38 CHEN, D.S., JAIN, R.C.: 'A robust back propagation learning algorithm for function approximation', *IEEE Trans. Neural Netw.*, 1994, **5**, (3), pp. 467–479