Splicing System Based Genetic Algorithms for Developing RBF Networks Models[*]

TAO Jili(陶吉利) and WANG Ning(王宁)[**]
National Laboratory of Industrial Control Technology, Institute of Advanced Process Control, Zhejiang University, Hangzhou 310027, China

Abstract A splicing system based genetic algorithm is proposed to optimize dynamical radial basis function (RBF) neural network, which is used to extract valuable process information from input output data. The novel RBF network training technique includes the network structure into the set of function centers by compromising between the conflicting requirements of reducing prediction error and simultaneously decreasing model complexity. The effectiveness of the proposed method is illustrated through the development of dynamic models as a benchmark discrete example and a continuous stirred tank reactor by comparing with several different RBF network training methods.

Keywords RBF network, structure optimization, genetic algorithm, splicing system

1 INTRODUCTION

Radial basis function (RBF) networks attracted considerable interest in the past because of its several advantages compared with other types of artificial neural networks (ANNs), such as better approximation capabilities, simpler network structures, and faster learning algorithms[1]. However, the selection of appropriate number of basis functions is a critical issue for RBF networks[2]. The number of basis functions controls the complexity of the structure , *i.e.*, the generalization capability of RBF networks. A RBF network, containing very few basis functions, yields poor predictions on new data, *i.e.*, poor generalization, as the model has limited flexibility. The RBF network, containing several basis functions, also yields poor generalization, as it is too flexible and fits the noise in the training data. The best generalization performance is obtained *via* the compromise between the conflicting requirements of simultaneously reducing the prediction error and decreasing the complexity of the model. This trade-off highlights the importance of optimizing the complexity of RBF network to achieve the best generalization.

More specifically, most of the standard RBF training methods require the designer to fix the network structure. These training procedures usually proceed *via* two steps[3]: First, the centers of basis function are determined using clustering method. Second, the calculation of the final-layer weights is reduced to solve a simple linear system using least squares method. Therefore, the first stage is an unsupervised method, and separated from the actual objective to minimize the output prediction error. In this study, the RBF networks are constructed using the input data supervised by the output data.

The inclusion of the structure selection in the formulation of the network optimization problem is desirable, but it results in a rather difficult problem, which cannot be easily solved using the standard optimization methods. An interesting alternative for solving this complicated problem is offered by the use of the recently developed evolutionary computation methods. Perhaps the most popular and successful strategies are the so-called genetic algorithms (GAs), which are stochastic methods based on the principles of natural selection and evolution[4]. GAs have proved to be successful in the structure selection of several types of neural networks, such as BP neural networks[5,6] and recurrent neural networks[7,8]. As to the optimization of RBF networks, Vesin and Gruter used GA to solve the complete optimization problem, but the centers of the potential nodes were restricted among the set of training data[9]. Esposito *et al*. employed a GA based technique to determine the widths of Gaussian functions in RBF networks[10], whereas Sarimveis *et al*. used GA approach to optimize the parameters of RBF networks in terms of the error minimization criterion[11].

In this study, the structure selection is included, and the fitness of each chromosome is calculated on the basis of the prediction error and the structure complexity criterion. To simplify the optimization of RBF network, the radial basis function is chosen as thin-plate-spline function[12], where the determination of widths is not required. Therefore, the GA in this study is used to determine the centers of basis functions and the network structure. The final-layer weights are derived using recursive least squares (RLS) method with the same initial weight vector. The proposed algorithm starts with a random population of RBF networks, which are coded as chromosomes. As all the function centers generated by stochastic chromosomes are not feasible, two novel operators, *i.e.*, elongation and deletion, enlightened by DNA splicing system[13,14], are introduced in the GA approach.

2 SPLICING SYSTEM BASED GA FOR RBF NETWORKS

Generally, the determination of the RBF centers

Received 2006-04-11, accepted 2006-11-01.

[*] Supported by the National Natural Science Foundation of China (No.60421002), and the National High Technology Research and Development Program of China (863 Program, 2006AA040308).

[**] To whom correspondence should be addressed. E-mail: nwang@iipc.zju.edu.cn

is based on a self-organizing clustering approach, such as k-means clustering[15], the nearest neighbor clustering method[16]. The application of the above algorithms requires the transcendental knowledge of an appropriate clustering degree which is difficult to determine, and it considers only the input data. The proposed approach in this study does not require the transcendental knowledge of the plant; moreover, the structure and RBF centers can be synchronously optimized by utilizing the input output data.

GA is an optimization algorithm on the basis of Darwinism, which is very flexible. Depending on the features of the problem's solution space, there is a wide range of choices of fitness functions, the coding method, and the genetic operations, and all these factors affect the efficiency of genetic algorithm. This study is focused on the optimization of RBF network using the splicing system based GA.

2.1 Coding method

There are totally $n_r \times n$ real number parameters to be optimized in the RBF network, which means one chromosome should be able to give $n_r \times n$ real number values, where n_r is the number of hidden nodes, n is the number of input nodes. Hence, binary coding chromosome will become very complex, and decimal coding chromosome is used. The structure of the lth chromosome is shown below

$$\boldsymbol{C}_l = \begin{bmatrix} c_{1,1}^l & c_{1,2}^l & \cdots & c_{1,n}^l \\ c_{2,1}^l & c_{2,2}^l & \cdots & c_{2,n}^l \\ \vdots & \vdots & \ddots & \vdots \\ c_{n_r,1}^l & c_{n_r,2}^l & \cdots & c_{n_r,n}^l \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad (1)$$

where $l=1, 2, \cdots, L$, L is the size of the population, n_r is randomly produced between 1 and D, D is the maximum number of hidden nodes, the rows below n_r are set to zeros and do not correspond to the center. The elements of $\mathbf{C}_l$ are computed using the following equation:

$$c_{lj} = x_{j,\min} + r \cdot (x_{j,\max} - x_{j,\min})$$
$$1 \leqslant l \leqslant n_r, \quad 1 \leqslant j \leqslant n \qquad (2)$$

where r is the random number between 0 and 1, $x_{j,\min}$ and $x_{j,\max}$ is the minimum and the maximum values of input variables given in the problem.

2.2 Fitness function

As mentioned in the above sections, the drawbacks of the general training methods of RBF network mainly lie in the absence of global optimization of both the approximation capability and the generalization performance. To overcome these drawbacks, the choice of appropriate fitness function is crucial.

In this study, the training procedures using splicing system based GA are also preceded in two steps: First, the network structure and the basis function parameters are determined using the chromosomes of one population. Second, the final-layer weights are calculated using least squares method. As the direct least squares method cannot obtain the solutions for a bad-conditioned matrix, the output weights of the lth RBF neural network are calculated using the following RLS method[17]:

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \boldsymbol{K}(k)[y(k) - \boldsymbol{X}_r^{\mathrm{T}}(k)\boldsymbol{w}(k-1)]$$
$$\boldsymbol{K}(k) = \boldsymbol{P}(k-1)\boldsymbol{X}_r(k)[\boldsymbol{X}_r^{\mathrm{T}}(k)\boldsymbol{P}(k-1)\boldsymbol{X}_r(k)+1]^{-1}$$
$$\boldsymbol{P}(k) = \boldsymbol{P}(k-1) - \boldsymbol{K}(k)\boldsymbol{K}^{\mathrm{T}}(k)[\boldsymbol{X}_r^{\mathrm{T}}(k)\boldsymbol{P}(k-1)\boldsymbol{X}_r(k)+1] \qquad (3)$$

where $1 \leqslant k \leqslant N$, N is the maximum iterative time, $\boldsymbol{X}_r(k)$ is the n_r^l dimension output vector of the hidden layer, $y(k)$ is the output of the actual system, $\boldsymbol{K}(k)$ is the n_r^l dimension assistant vector, $\boldsymbol{P}(k)$ is the n_r^l-by-n_r^l assistant matrix. From Eq.(3), the computational complexity of RLS solution for one iterative time is obtained as $O\left(\left(n_r^l\right)^2\right)$. Hence, the complexity of RLS solution for the lth network weight vector is $O\left(N\left(n_r^l\right)^2\right)$.

In every generation of GA, the calculation of the output weights completes the formulation of L RBF networks, which can be represented by the pairs ($\boldsymbol{C}_1$, $\boldsymbol{w}_1$), ($\boldsymbol{C}_2$, $\boldsymbol{w}_2$),$\cdots$ and ($\boldsymbol{C}_L$, $\boldsymbol{w}_L$). To obtain good generalization capability of RBF networks, the training data are divided into two groups, one group of data ($\boldsymbol{X}_1$, $\boldsymbol{Y}_1$) are used to calculate the final-layer weights, herein, $N=N_1$ (N_1 is the number of the first group data), and the other group of data ($\boldsymbol{X}_2$, $\boldsymbol{Y}_2$) are utilized to evaluate the produced RBF networks in each generation. This scheme incorporates a testing procedure into the training algorithm, and guarantees good generalization performance of the RBF networks. However, to obtain good approximation capability of RBF networks, the network structure still becomes much complex. The structure complexity of RBF network conflicts with the generalization performance of neural networks. Therefore, the objective function considering both approximation capability and generalization performance is shown as follows.

$$J(\boldsymbol{C}_l, \boldsymbol{w}_l) = N_2 \ln\left[\sum_{t=1}^{N_2} |Y_2(t) - \hat{Y}_2(t)|\right] + \eta n_r^l \ln(N_2) \qquad (4)$$

Equation (4) expresses a compromise between the cost of modeling errors and the complexity of network structure[13], where $\hat{Y}_2(t)$ is the output of RBF network, N_2 is the number of the second group data, n_r^l is the number of hidden nodes in the lth chromosome, η is the weight coefficient, and $0 \leqslant \eta \leqslant 1$, the greater the η is, the stronger constraint of structure complexity would be considered. In this study, η is set as 1.

2.3 Operators in splicing system based GA

Li *et al*.[13] summarized all possible operations of the DNA splicing systems, such as elongation operation, deletion operation, absent operation, insertion operation, translocation operation, transformation operation permutation operation, *etc*. Above operations actually include three basic operations: selection, crossover, and mutation. Other operations adopted by standard genetic algorithm (SGA) for special problems may improve the performance of SGA.

2.3.1 *Selection operator*

A set of individuals from the previous population must be selected for reproduction. This selection depends on their fitness values. Individuals with good fitness values will most probably survive. There exist different types of selection operators, and in this study roulette wheel method is applied. The probability of the selected individual , $P(\boldsymbol{C}_l)$, is given by:

$$P(\boldsymbol{C}_l)=f(\boldsymbol{C}_l)\Big/\sum_{l=1}^{L}f(\boldsymbol{C}_l) \qquad (5)$$

where $f(\boldsymbol{C}_l)$ is the fitness function of the individual $\boldsymbol{C}_l$, which is obtained by $1/J(\boldsymbol{C}_l, \boldsymbol{w}_l)$. The roulette wheel is placed with L equally spaced pointers. A single spin of the roulette wheel will simultaneously pick all the members of the next population.As the computational complexity in Eq.(4) is $O(N_2)$, the complexity of selection operator in one generation is $O(LN_2)$.

2.3.2 *Crossover operator*

The crossover operator is applied after selection with a probability (p_c), which produces novel individuals, *i.e*., the novel structure and centers of RBF networks. It is executed between the currently selected individual $\boldsymbol{C}_l$ and its subsequent individual $\boldsymbol{C}_{l+1}$, and yields the offspring chromosomes $\boldsymbol{C}_l'$, $\boldsymbol{C}_{l+1}'$. As the number of input nodes (n) is fixed during the whole GA optimization process, the crossover point is chosen between 1 and n. The procedure is demonstrated in Fig.1, which represents the single-point crossover. As the computational complexity of the plus operator for two D-by-n matrices is $O(nD)$, in the worst case (all chromosome pairs execute crossover operator), the crossover operator in one generation requires $O(LnD/2)$ computations.

2.3.3 *Mutation operator*

To effectively explore the search space, mutation is carried out. When the element of an individual is mutated with a probability (p_m), it is replaced by a novel generated element in terms of Eq.(2). Similar complexity can be obtained as in the case of crossover operator, which is $O(LnD)$.

2.3.4 *Splicing operators*

As shown in the description of the crossover operation, different structures of RBF networks can be produced using this genetic operator. However, the operator may yield unreasonable RBF structure as shown in Fig.1, where most of the centers in row 5 and row 6 of $\boldsymbol{C}_l'$ are zeros. Moreover, the crossover operator does not always modify the structure of the parent chromosomes. Therefore, enlightened by the DNA splicing system[13], two more genetic operators, *i.e*., elongation operator and deletion operator are introduced. The elongation operator is used to add a novel node, and a random nonzero vector is created as described in Eq.(2), whereas the deletion operator is utilized to replace the existing unreasonable node $\boldsymbol{c}_i^l\left(\left[c_{i1}^l, c_{i2}^l, \cdots, c_{in}^l\right]\right)$ with a zero vector. Suppose r is a random number between 0 and 1, when $p_\mathrm{e} > r$, the elongation operator is executed. In the worst case (the number of hidden nodes in each chromosome add up to D), the complexity is $O\left(\sum_{l=1}^{L}\left(D-n_\mathrm{r}^l\right)n\right)$. The deletion operator is executed in the place of existing unreasonable node, which is determined by the number of zeros in the node centers. If the number of zeros goes beyond 2, the node is considered as an unreasonable one and will be replaced by deletion operator. In the worst case (all nodes are deleted), the computational complexity is $O\left(\sum_{l=1}^{L}n_r^l n\right)$.

2.4 The procedure of splicing system based GA

The whole processes of the optimization of RBF networks are described in the following steps.

Step 1: Generate the code for L chromosomes randomly in the search space.

Step 2: Calculate the corresponding L weight vectors using RLS method and compute the performance index f for each individual.

Step 3: Select the chromosomes for the generation of new chromosomes of the next generation according to the selection operator.

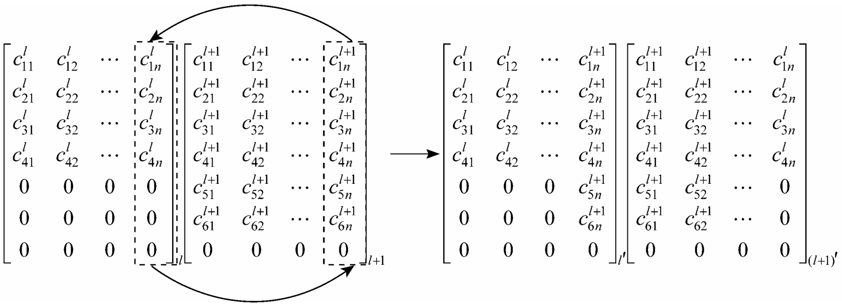

Figure 1 Schematic diagram of the crossover operation

Step 4: Choose a point randomly in the range $[1, n]$, and exchange the codes of the pairs of chromosomes reproduced in Step 3. Repeat this for all the $p_c L/2$ pairs of parents.

Step 5: Implement mutation, replace the element of the current chromosome with the novel element generated by Eq.(2).

Step 6: Execute the splicing operators, when the conditions are met.

Step 7: Repeat Steps 2 to 6 until a termination criterion is met. This can be the set of maximum number of evolutions, or the set of minimum improvement of the best performance in successive generations. Moreover, Elitism, the inclusion of the best current set in the next population, is used throughout.

Considering the complexity of one generation in the whole procedure, the basic operations of one generation being performed and the worst case complexities associated with it, are as follows:

(1) L RLS solutions of weight vector is $O\left(\sum_{l=1}^{L} N_1 \left(n_r^l\right)^2\right)$;

(2) Selection operator is $O(LN_2)$;

(3) Crossover operator is $O(LnD/2)$,

(4) Mutation operator is $O(LnD)$;

(5) Elongation operator is $O\left(\sum_{l=1}^{L} (D - n_r^l) n\right)$, and

(6) Deletion operator is $O\left(\sum_{l=1}^{L} n_r^l n\right)$.

As can be seen, the overall complexity of the above algorithm is $O\left(\sum_{l=1}^{L} N_1 \left(n_r^l\right)^2\right)$. Once the number of nodes $\left(n_r^l\right)$ in the hidden layer increases, the computation significantly increases. Moreover, as the elitism strategy is adopted throughout the whole evolutionary procedure, the proposed GA can be completely converged[18].

3 SIMULATION RESULTS

The performance of the proposed methodology is evaluated by applying it on two different systems: a nonlinear benchmark problem described by a discrete input output model and a nonlinear continuous stirred tank reactor.

3.1 Simulation tests on a discrete input output model

The discrete input output model is described as follows[19].

$$y(k) = \frac{y(k-1)y(k-2)y(k-3)u(k-2)\left[y(k-3)-1\right]+u(k-1)}{1+y(k-2)^2+y(k-3)^2} \tag{6}$$

where

$$u(k) = \begin{cases} \sin(\dfrac{2\pi k}{250}), & k \leqslant 500 \\ 0.8\sin(\dfrac{2\pi k}{250}) + 0.2\sin(\dfrac{2\pi k}{25}), & k > 500 \end{cases}$$

The objective of this application is to utilize the proposed methodology to obtain suitable RBF configuration for modeling the aforementioned system. The input of RBF model consists of two previous values of u and three previous values of y.

$$\boldsymbol{x}(k) = [u(k-1)\ u(k-2)\ y(k-1)\ y(k-2)\ y(k-3)] \tag{7}$$

The proposed methodology is compared with nearest neighbor clustering method and k-means clustering method, which are used to train the centers of RBF network. 1000 data points are produced according to Eq.(6), where the first 500 data points are used to optimize the RBF networks, and the other 500 data points are used to test the performance of the given RBF networks. The operational parameters used by the proposed algorithm can be seen in Table 1.

Table 1 GA parameters used in the discrete model and CSTR examples

Algorithm parameters	Discrete example	CSTR example
number of chromosomes L	30	30
maximum of hidden nodes D	40	150
number of generations G	150	150
probability of crossover p_c	0.8	0.8
probability of mutation p_m	0.1	0.1
probability of elongation p_e	0.01	0.01

The simulation results obtained using the above two methods and the proposed GA are shown in Figs.2—5. Fig.2 shows the fitting curve of prediction values and real values using nearest neighbor clustering method, and the plots of the estimation error are depicted in Figs.3—5. As the nearest neighbor clustering method was used for calculating the number of Gaussian functions and their corresponding function centers, σ should be set in advance. In Fig.3, σ is selected as 0.2 using trial and error method. In Fig.4, the structure of RBF network is obtained using k-means clustering method with 20 clusters in terms of nearest neighbor clustering method, and σ is optimized using SGA. The number of hidden nodes in RBF network and the sum of absolute values of modeling error (S) using the above 3 methods are listed in Table 2. By comparing the results in Table 2 and the plots in Figs.3—5, it can be observed that the proposed approach minimizes the error to a small extent using an almost equivalent network structure. Moreover, it is needless to set the network parameter, such as σ to 0.2 and clusters to 20.

Table 2 Simulation results using the 3 different methods

Methods	Number of hidden nodes	S
nearest neighbor clustering	20	5.2876
k-means clustering & SGA	20	1.9928
proposed GA	22	0.2765

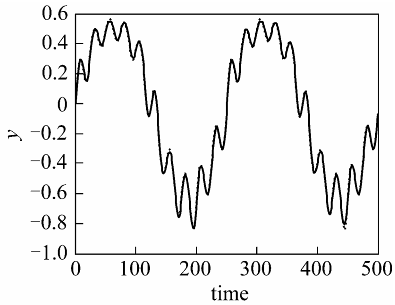

Figure 2 RBF network predictions using nearest neighbor clustering method

- - - - neural predictions; —— real values

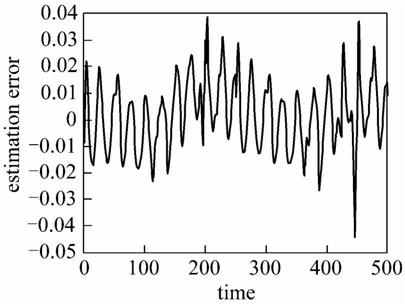

Figure 3 Estimation of error using nearest neighbor clustering method

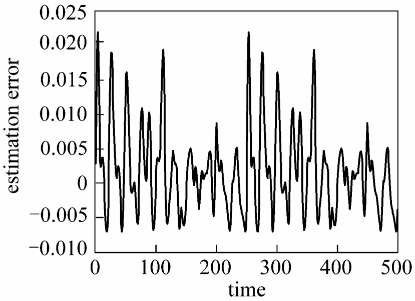

Figure 4 Estimation of error using *k*-means clustering method and SGA

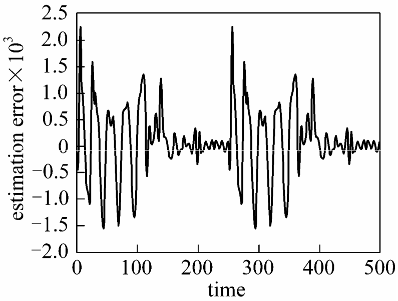

Figure 5 Estimation error using proposed method

3.2 Simulation tests on a continuous stirred tank reactor

In this study, a nonisothermal CSTR process is considered, which is characterized using the following dynamic equations[20]:

$$\dot{x}_1 = -x_1 + Da(1-x_1)\exp\left(\frac{x_2}{1+x_2/\varphi}\right)$$

$$\dot{x}_2 = -(1+\delta)x_2 + BDa(1-x_1)\exp\left(\frac{x_2}{1+x_2/\varphi}\right) + \delta u \quad (8)$$

where x_1 and x_2 represent the dimensionless reactant concentration and reactant temperature, respectively, the physical parameters in the CSTR model equations are: Da, φ, B and δ, which correspond to the Damökhler number, the activated energy, heat of reaction, and heat transfer coefficient, respectively. The values of the system parameters can be obtained as: $Da=0.072$, $\varphi=20$, $B=8$, $\delta=0.3$. The control input, u, is the dimensionless temperature of the coolant. Because of the hard input constraints, u lies between −5 and 5.

The objective is to build discrete dynamic models for predicting the state variable x_1 and x_2 using the input output data. The input vector of RBF network model is selected as follows.

$$\boldsymbol{x}(k) = [u(k-1)\ u(k-2)\ u(k-3)\ y(k)\ y(k-1)\ ] \quad (9)$$

The proposed method is applied to optimize both the structure and the centers of RBF networks. As the problem is relatively complex, the number of hidden nodes obtained using nearest neighbor clustering is 169 corresponding to $\sigma=0.3$, which is too complicated to be adopted. Hence, the results of the proposed algorithm are compared with another RBF identification scheme, *i.e.*, the improved S&A GA in Ref.[11]. The same radial basis function is adopted and the maximum number of hidden nodes is chosen as 150.

For all simulations that follow, a set of 700 input output data points are created by randomly selecting the values of the input variable within the space [−5, 5]. The first 300 data are used in the training procedure to calculate the connection weights; the second 200 data are also used during the training process to evaluate the RBF network in each generation, whereas the remaining 200 data are used to test the efficiency of the ultimate RBF network. To test the generalization performance of RBF networks, the values of the output variables are modified by adding random noise chosen from a uniform distribution at the interval [−3%, +3%] of the maximum values of the given variable.

To evaluate the approximation capability and generalization performance, both the training error and testing error are compared as shown in Figs.6—9. Once the best structure and the centers of RBF network are optimized using GA according to the first and second data sets, the weights between the hidden layer and the output layer are updated using RLS method according to the second 200 data, the training errors are thus obtained, and then, the weights are fixed and the testing errors are calculated in terms of the third 200 data. Figs.6 and 8 show the responses of RBF network predictor optimized using S&A GA with RLS method because the LS method in Ref.[11] could not always obtain the inverse of the matrix. By comparing with Figs.7 and 9 through RBF networks using S&A GA, training errors are obtained, which are smaller than that by the use of proposed GA, *i.e.*, the sum of absolute values of the training error (S_1) using S&A GA is smaller than that by the use of proposed GA, and they demand much more hidden nodes regardless of the complexity of the

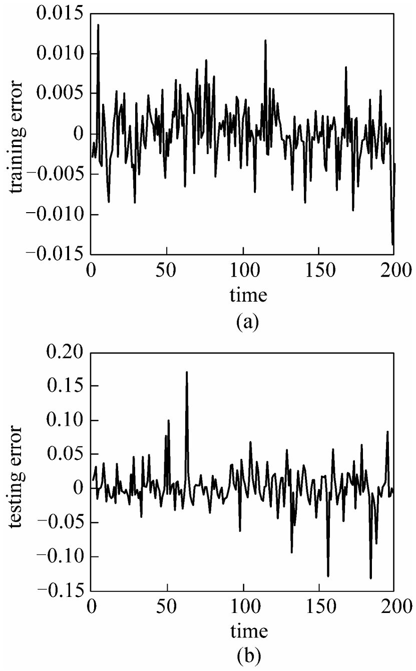

Figure 6 Training and testing error for x_1 using S&A GA in RBF network

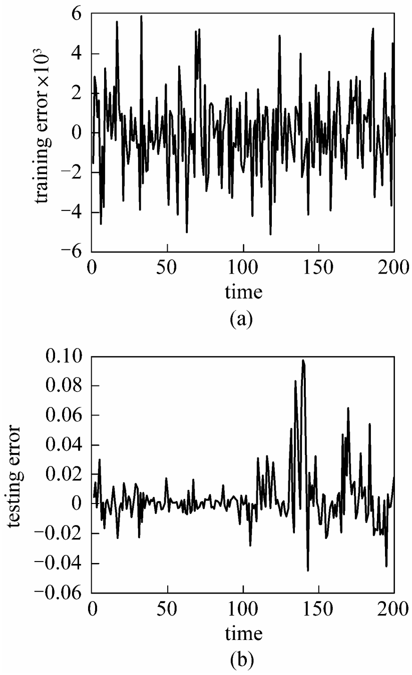

Figure 8 Training and testing error for x_2 using S&A GA in RBF network

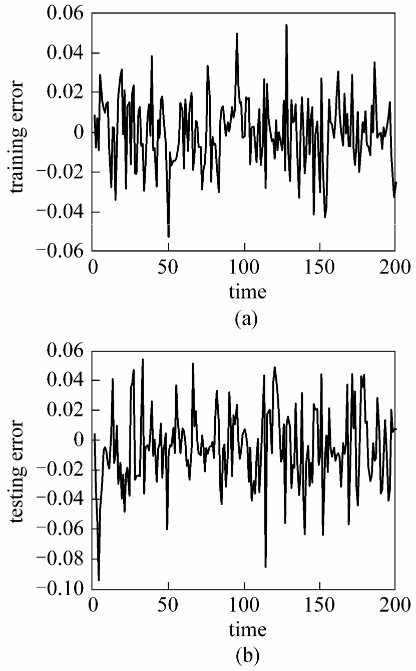

Figure 7 Training and testing error for x_1 using proposed GA in RBF network

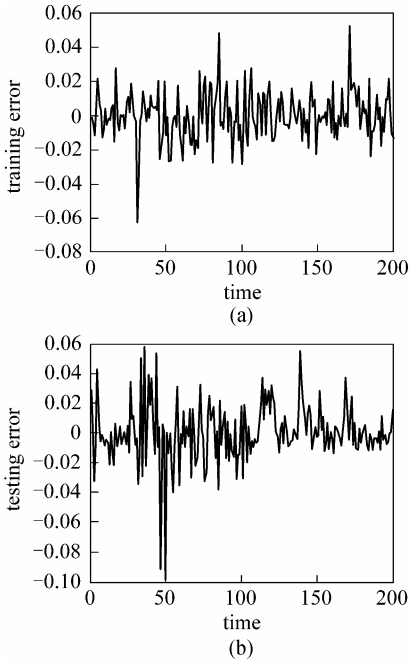

Figure 9 Training and testing error for x_2 using proposed GA in RBF network

network structure. Moreover, the sum of the absolute value of the testing error (S_2) are quite similar, and some points of testing error in Figs.6 and 8 are smaller than that in Figs.7 and 9, which can also be testified using the maximum testing error ($E_{2\max}$). This may be caused by the over-fitting of RBF network with several hidden nodes. Table 3 shows that, as the number of hidden nodes using S&A GA is much more than that by the use of proposed GA, the runtime has markedly increased, which is consistent with the analysis of computational complexity. All algorithms are programmed by MATLAB7.01 using the computer with Celeron(R) CPU 2.4GHz and RAM 256MB.

4 CONCLUSIONS

This article presents a splicing system based GA for the optimization of both structure and centers of the RBF network model. This algorithm is based on

Table 3 The simulation results using two improved GA

Methods	x_1					x_2				
	Number of nodes	S_1	S_2	$E_{2\max}$	Runtime, s	Number of nodes	S_1	S_2	$E_{2\max}$	Runtime, s
S&A GA	146	0.5654	3.8694	0.1752	4.5582×10^3	136	0.3206	2.2326	0.0973	4.5076×10^3
proposed GA	47	2.7059	3.9223	0.0945	872.7500	45	2.1181	2.7235	0.0983	699.4690

input-output data, and its objective function considers both approximation capability and generalization performance of the RBF network. In this manner, the network simultaneously retains a reasonable size and effectively describes the whole system. Different simulations of benchmark problem and typical nonlinear dynamic CSTR system are performed to illustrate the effectiveness of the proposed method. The results show that this method can produce highly accurate prediction and keep a relatively simple network structure.

NOMENCLATURE

B	heat of the reaction
$\boldsymbol{C}_l$	the lth n-by-D chromosome matrix
$\boldsymbol{c}_i$	the ith node center vector
D	maximum number of the hidden nodes
Da	Damökhler number
$E_{2\max}$	maximum of the testing error
f	the fitness function
G	maximum number of generations
J	the objective function
$\boldsymbol{K}$	n_{r}^{l} dimension assistant vector in the lth network in RLS algorithm
L	size of the population
N	maximum iterative time
n	number of the input nodes
n_{r}^{l}	number of hidden nodes of the lth network
$\boldsymbol{P}$	n_{r}^{l} -by- n_{r}^{l} assistant matrix in the lth network in RLS algorithm
p_{c}	probability of crossover
p_{e}	probability of elongation
p_{m}	probability of mutation
r	random number between 0 and 1
S	sum of the absolute value of the modeling error
S_1	sum of the absolute value of the training error
S_2	sum of the absolute value of the testing error
u	control input of the actual system
$\boldsymbol{w}_l$	weight vector of the lth RBF network
$\boldsymbol{X}_{\mathrm{r}}$	n_{r}^{l} dimension output vector of the hidden layer in RLS algorithm
$\boldsymbol{X}_1$	N_1 n-dimension input vectors
$\boldsymbol{X}_2$	N_2 n-dimension input vectors
$\boldsymbol{x}$	input vector of the RBF network
x_1	reactant concentration
x_2	reactor temperature
$\boldsymbol{Y}_1$	N_1 output values of the system
$\boldsymbol{Y}_2$	N_2 output values of the system
y	output value of the actual system
δ	heat transfer coefficient
σ	width of the Gaussian function
φ	activated energy

REFERENCES

1 Park, J., Sandberg, I., "Universal approximation using radial basis function networks", *Neural Comput.*, **3**(2), 246—257(1991).

2 Prechelt, L., "Automatic early stopping using cross validation: quantifying the criteria", *Neural Network.*, **11**(4), 761—767(1998).

3 Alexandridis, A., Sarimveis, H., Bafas, G., "A new algorithm for online structure and parameter adaptation of RBF networks", *Neural Network.*, **16**(7), 1003—1017(2003).

4 Holland, J.H., Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, USA (1992).

5 Wang, Y., Yao, P., "Simulation and optimization for thermally coupled distillation using artificial neural network and genetic algorithm", *Chin. J. Chem. Eng.*, **11**(3), 307—311(2003).

6 Yang, T., Lin, H.C., Chen, M.L., "Metamodeling approach in solving the machine parameters optimization problem using neural network and genetic algorithms: A case study", *Robot Comput. Integrated Manuf.*, **22**(4), 322—331(2006).

7 Blanco, A., Delgado, M., Pegalajar, M.C., "Real-coded genetic algorithm for training recurrent neural networks", *Neural Network.*, **14**(1), 93—105(2001).

8 Delgado, M., Pegalajar, M.C., "A multiobjective genetic algorithm for obtaining the optimal size of a recurrent neural network for grammatical inference", *Pattern Recognit.*, **38**(9), 1444—1456(2005).

9 Vesin, J.M., Gruter, R., "Model selection using a simplex reproduction genetic algorithm", *Signal Process.*, **78**(3), 321—327(1999).

10 Esposito, A., Marinaro, M., Oricchio, D., Scarpetta, S., "Approximation of continuous and discontinuous mappings by a growing neural RBF-based algorithm", *Neural Network.*, **13**(6), 651—665(2000).

11 Sarimveis, H., Alexandridis, A., Mazarakis, S., Bafas, G., "A new algorithm for developing dynamic radial basis function neural network models based on genetic algorithms", *Comput. Chem. Eng.*, **28**(1), 209—217(2004).

12 Chen, S., Cowan, C.F.N., Grant, P.M., "Orthogonal least squares learning algorithm for radial basis function networks", *IEEE Trans. Neural Netw.*, **2**(2), 302—309(1991).

13 Li, S.C., Xu, J., Pan, L.Q., "Operational rules for digital coding of RNA sequences based on DNA computing in high dimensional space", *Bull. Sci. Tech. Soc.*, **18**(6), 388—398(2003).

14 Jonoska, N., Seeman, N.C., DNA Computing, Springer-Verlag, New York (2002).

15 Pham, D.T., Dimov, S.S., Nguyen, C.D., "Selection of *K* in *k*-means clustering", *Proc. Inst. Mech. Eng. C. Mech. Eng. Sci.*, **219**(1), 103—119(2005).

16 Zhang, X., Song, J., "RBF neural networks based on dynamic nearest neighbor-clustering algorithm and its application in prediction of MH-Ni battery capacity", *Transactions of China Electrotechnical Society*, **20**(11), 84—87(2005).

17 Hou, Y.B., Wang M., Wang, L.Q., System Identification and Its MATLAB Simulation, Science Press, Beijing (2004).

18 Li, M.Q., Kou, J.S., Basic Theories and Applications of Genetic Algorithm, Science Press, Beijing (2002).

19 Liu, G.P., Kadirkamanathan, V., Billings, S.A., "On-line identification of nonlinear systems using Volterra polynomial basis function neural networks", *Neural Netw.*, **11**(9), 1645—1657(1998).

20 Chen, C.T., Peng, S.T., "Learning control of process systems with hard input constraints", *J. Process Control*, **9**(2), 151—160(1999).