A Self-Organizing RBF Neural Network Based on Distance Concentration Immune Algorithm

Junfei Qiao, *Member, IEEE*, Fei Li, Cuili Yang, Wenjing Li, and Ke Gu

***Abstract*—Radial basis function neural network (RBFNN) is an effective algorithm in nonlinear system identification. How to properly adjust the structure and parameters of RBFNN is quite challenging. To solve this problem, a distance concentration immune algorithm (DCIA) is proposed to self-organize the structure and parameters of the RBFNN in this paper. First, the distance concentration algorithm, which increases the diversity of antibodies, is used to find the global optimal solution. Secondly, the information processing strength (IPS) algorithm is used to avoid the instability that is caused by the hidden layer with neurons split or deleted randomly. However, to improve the forecasting accuracy and reduce the computation time, a sample with the most frequent occurrence of maximum error is proposed to regulate the parameters of the new neuron. In addition, the convergence proof of a self-organizing RBF neural network based on distance concentration immune algorithm (DCIA-SORBFNN) is applied to guarantee the feasibility of algorithm. Finally, several nonlinear functions are used to validate the effectiveness of the algorithm. Experimental results show that the proposed DCIA-SORBFNN has achieved better nonlinear approximation ability than that of the art relevant competitors.**

***Index Terms*—Distance concentration immune algorithm (DCIA), information processing strength (IPS), radial basis function neural network (RBFNN).**

I. Introduction

THE radial basis function neural network (RBFNN) has been extensively used to model and control nonlinear systems due to its universal approximation ability [1]–[3]. In addition, it is able to approximate any nonlinear function to any desirable accuracy [1] when there are enough neurons. The desired approximation accuracy is primarily achieved by the network size and the parameters of RBFNN.

In order to adjust the network parameters, the gradient-based methods are proposed in [4], [5]. Among them, the error back propagation (BP) algorithm is popular and widely used [6]. However, the BP algorithm has still many shortcomings, such as the time-consuming convergence and poor global-search capability [7]. Compared with the BP algorithm, the recursive least squares (RLS) algorithm has better convergence rate and accuracy [8]. However, RLS involves more complicated mathematical operations and require more computational resources. In addition, the local minimum is not solved [9]. To solve this problem, a variable-length sliding window blockwise least squares (VLSWBLS) algorithm is proposed by Jiang and Zhang [10]. VLSWBLS outperforms the RLS with forgetting factors. Peng *et al*. [11] introduced a continuous forward algorithm (CFA) to optimize the parameters of RBFNNs. As a result, this method realizes major performance in reducing memory usage and computational complexity. Qiao and Han [12] proposed a forward-only computation (FOC) algorithm to adjust the parameters. Unlike the traditional forward and backward computation, the FOC algorithm simplifies the calculation and decreases computational complexity. However, the descriptions about how to automatically adjust network size are seldom seen in these literatures mentioned above.

In fact, a proper structure size can avoid network overfitting and achieve a desired performance. In recent years, many studies have focused on the structure design of the RBFNN. Huang *et al*. [13] proposed a sequential learning method known as the growing and pruning RBF (GAP-RBF) algorithm. In addition, a more advanced model based on the GAP-RBF algorithm (GGAP-RBF) was advocated in [14]. The results show that an RBFNN with a relatively compact structure can achieve a less computational time. However, both algorithms require a complete set of samples for the training process. Generally, it is impossible for designers to obtain a priori knowledge of the training samples before implementation [15]. To solve this problem, an information-oriented algorithm (IOA) [16] was proposed to self-organize the RBFNN structure. The IOA is used to calculate the information processing strength (IPS) of hidden neurons. In addition, it is a computational technique that identifies hidden independent sources from multivariate data. However, most of these self-organizing RBF (SORBF) neural networks adopt the learning algorithms that are based on the gradient decent (GD) algorithm, which may easily trap into a local optimum [17].

To optimize the parameters and network size of an RBFNN simultaneously, the evolutionary algorithms (EAs) are studied to train the RBFNN [18] to achieve a good robustness and a global optimization capability. For example, Feng [19] proposed an SORBF neural network based on the particle swarm optimization (PSO) algorithm. Alexandridis *et al*. [20]

Manuscript received March 30, 2019; revised July 17, 2019; accepted September 26, 2019. This work was supported by the National Natural Science Foundation of China (61890930-5, 61533002, 61603012), the Major Science and Technology Program for Water Pollution Control and Treatment of China (2018ZX07111005), the National Key Research and Development Project (2018YFC1900800-5), and Beijing Municipal Education Commission Foundation (KM201710005025). Recommended by Associate Editor Haibo He. *(Corresponding author: Fei Li.)*

Citation: J. F. Qiao, F. Li, C. L. Yang, W. J. Li, and K. Gu, "A self-organizing RBF neural network based on distance concentration immune algorithm," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 1, pp. 276–291, Jan. 2020.

The authors are with Beijing University of Technology, Beijing 100083, China (e-mail: Junfeig@bjut.edu.cn; lifeiaedu@hotmait.com; clyang5@bjut.edu.cn; wenjing.li@bjut.edu.cn; guke.doctor@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2019.1911852

developed a novel algorithm that used fuzzy means and PSO algorithms to train the RBFNN. The results show that the proposed SORBF neural networks obtain higher prediction accuracy and smaller network structure. Moreover, an adaptive-pso-based self-organizing RBF neural network (APSO-SORBF) was proposed to construct the RBFNN in [21]. This algorithm is adopted to determine the optimal parameters and network size of the RBFNN simultaneously for time series prediction problems. The simulation results illustrate that APSO-SORBF performs better than the other PSO-based RBFNN in terms of the forecast accuracy and the computational efficiency [21]. Compared with other EA algorithms, the PSO algorithm has a faster convergence rate, but it easily traps into a local optimum, which affects the calculation accuracy [22]. To solve this problem, the immune algorithm (IA) was proposed [23]. The IA is a highly parallel, distributed, and adaptive system whose diversity and maintaining mechanism can be used to maintain the diversity of solutions and overcome the "premature" problem of a multi-peak function. Moreover, to break away from the local optimal solution, increasing the diversity of the artificial immune algorithm is very important. In this paper, the distance concentration immune algorithm (DCIA) was proposed to increase the population diversity. In comparison with the artificial immune based on information entropy (IEIA) algorithm, this algorithm does not need requirement for setting any threshold. The results show that the proposed DCIA can significantly increase the global search capability. According to the above analysis, it is necessary and effective for the structure and parameters of the neuron network to be adjusted by DCIA instead of PSO [21]. However, the structure of APSO-SORBF [21] is adjusted by increasing and decreasing the number of hidden layer neurons randomly so that the network is unstable. Thus, how to adjust the structure steadily and present theoretical analysis of algorithm convergence is quite challenging.

To solve the problems as mentioned above, the IPS [16] of hidden neurons was adopted to determine which hidden layer neurons need to be split or pruned when the network of antibodies should be updated. However, the last input sample is used to adjust the parameters of the hidden layer neuron in [16]. Then, the computational accuracy is affected, and the calculation time is lengthened. Based on the analysis above, the information-oriented error compensation algorithm (IOECA) is proposed in this paper. In this algorithm, the input sample with the most frequent occurrence of maximum error is used to set up the parameters of the new hidden neurons. Then the accuracy is increased. In addition, in order to ensure the stability of algorithm, the convergence analysis of the DCIA-SORBF neural network is provided.

The main contributions of this paper are summarized as follows:

1) A DCIA algorithm is adopted to improve the diversity of antibodies. The distance concentration is used to determine the diversity of the artificial immune algorithm. The DCIA algorithm approaches to skip the local minimum and finally find the global minimum. Consequently, it has a higher accuracy than that of many traditional EAs.

2) The immune algorithm is used to adjust the network and parameters according to [16]. However, in [16], the hidden layer neurons are increased or deleted randomly to adjust network structure. Then this caused instability of the system. To solve this, the IPS is used to identify the hidden layer neurons that needed to be deleted and increased according to its ability to identify hidden independent sources from multivariate data.

3) Five samples are used to be calculate simultaneously in [16], and the last one is used to update the parameters of hidden layer neurons, so that the system has large amount of calculation and its computation is not high enough. So that all samples are selected in this paper. Furthermore, the input sample with the most frequent occurrence of maximum error is used to set up the parameters of the new hidden neurons. Therefore, the parameters and structure of the RBFNN can be optimized simultaneously by DCIA-SORBF. Compared with other algorithms, this method has a greater accuracy with a compact structure, and the stability can be ensured.

4) The convergence analysis is provided. The results of multiple experiments testify to the feasibility and efficiency of the DCIA-SORBF algorithm. A convergence analysis of the DCIA-SORBF neural network is provided and the effectiveness is verified via simulations because the convergence of the algorithm is necessary and very important for many actual engineering problems.

The rest of this article is organized as followed: In Section II, brief reviews of the RBF and immune algorithm system model are introduced. In Section III, the details of the DCIA algorithm and DCIA-SORBF are described. In Section IV, the convergence analyses of DCIA-SORBF are provided. In Section V, four experiments are conducted. Finally, the conclusions are presented.

II. Problem Formulation

A. RBF Neural Network

The RBF neural network is a typical feed-forward neural network [24], and it is generally composed of three layers: the input layer, hidden layer and output layer. The structure of the RBF is shown in Fig. 1.

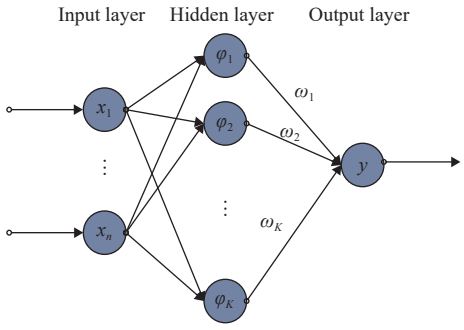


Fig. 1. RBF neural network.

The structure of the RBF neural network is described as follows:

1) The input layer. In this layer, an n-dimensional input vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ is imported to the network, where n

is the number of neurons in the layer.

2) The hidden layer. In this layer, the input variables are converted to the high dimensional space by a nonlinear transformation. There are many activation functions in network, such as Gaussian function, sigmoid function and so on. Here, Gaussian function is used as the activation function. The output is defined as

$$\phi_K(t) = e^{\frac{\|\boldsymbol{x}(t)-\mu_K(t)\|}{\sigma_K^2(t)}}. \tag{1}$$

Among them, $\phi_K(t)$ is the output of t-time hidden layer. $\boldsymbol{x}(t)$ is the input sample matrix at time t, $\mu_K(t)$ is the center of the Kth hidden layer neurons at time t, and $\sigma_K^2(t)$ is the width of the Kth hidden layer neurons at time t. $\|\boldsymbol{x}(t)-\mu_K(t)\|$ is the Euclidean distance between $\boldsymbol{x}(t)$ and $\mu_K(t)$. It is noted that the widths and centers of the activation function is not fixed. They are randomly initiated and then optimized by DCIA.

3) The output layer. In this layer, there is only one node, which is the output of the neural network. The output is given as

$$y = \sum_{j=1}^{K} \phi_j \omega_j. \tag{2}$$

where ω_j is the connection weight between the jth neuron in hidden layer and the network output, and y is the network output. Here, ω_j is randomly initialized.

B. Immune Algorithm System

The artificial immune system is primarily based on the information processing mechanism of the biological immune system [25]. Then, the new algorithm is used to solve complex problems. To describe the algorithm better, several common immunology terms in the artificial immune system are defined as follows:

Definition 1 (Antigen): An antigen refers to the problem of the constraint to be solved. It is defined as the objective function.

Definition 2 (Antibody): An antibody refers to the candidate solution of the problem. It is described as a candidate solution that corresponds to the objective function.

Definition 3 (Affinity): An affinity refers to the adaptive measure of the candidate solution. It is related to the problem or the candidate solution that corresponds to the target function value.

The antibodies that have a high affinity in the immune system can achieve a high rate of cloning. To maintain the diversity of antibodies, the rate of cloning $P(x_i)$ can be expressed as follows [25]:

$$P(x_i) = \frac{A(x_i)}{C(x_i)} \tag{3}$$

$$\boldsymbol{M}_a(t) = [E_1(t),\ldots,E_i(t),\ldots,E_o(t),R_1(t),\ldots,R_i(t),\ldots,R_s(t)] \tag{4}$$

where $D(x_i)$ is the affinity function of the antibody, $C(x_i)$ is the concentration of the antibody, $\boldsymbol{M}_a(t)$ is the parent generation group, and $a = o + s$, where a is the size of parent generation group, o is the elite number. $E_i(t)$ is the elite solution of the population, and it is selected according to the affinity $A(x_i)$ which is sorted in ascending order. The immune cells $R_i(t)$, which are used to reflect the diversity of individuals, are selected from $o+1$ to a according to $P(x_i)$ that is listed in descending order. And to update the optimal individual, the method is expressed as

$$A(x_i) = \frac{1}{D(x_i)} \tag{5a}$$

$$G(t) = D\left(x_g(t)\right) = \min\left(D\left(\boldsymbol{M}_a(t)\right)\right) \tag{5b}$$

where $D(\cdot)$ is the antibody affinity function. $G(t)$ is the minimum value of population affinity. $x_g(t)$ is the global best solution at time t. Subsequently, the crossover and mutation operations can progress.

In fact, the antibody diversity is very important to immune algorithm. It is closely related to global searching ability of algorithm. For that reason, the R bit comparison method that reflects the similarity degree between antibodies was proposed. However, such an approach is time consuming and has a low calculation accuracy. To avoid this problem, Chun proposed an information entropy-based artificial immune algorithm (IEIA) [26] whose diversity can be satisfied. However, the constant factors of this algorithm that is able to influence the convergence performance, are determined by experience. In addition, the calculation of different antibodies is similar. In view of the above problems, Zheng *et al*. [27] proposed artificial immunity based on the Euclid distance algorithm (EDAI). This algorithm calculates the Euclid distance between two antibodies and if it reaches a certain threshold, the similarity of antibodies can be determined. However, the problem of threshold setting should be solved, which is a tedious process. In light of the above problems, the distance concentration immune algorithm (DCIA) was proposed to increase the diversity. This algorithm can effectively jump from local optimal solution without requirement for setting any threshold. The results show that the calculation accuracy is improved effectively.

III. DCIA-SORBF Neural Network

The performance of the RBFNN primarily relies on its structures and parameters. To optimize them simultaneously, the distance concentration artificial immune algorithm (DCIA) is used. However, if the network structure of the immune cells is randomly increased and decreased [21], system instability will occur. To avoid this situation, the information-oriented error compensation algorithm (IOECA) is proposed, with an aim of obtaining a compact structure of the RBFNN. Subsequently, the accuracy of the algorithm is improved and the stability is guaranteed.

A. DCIA Algorithm

To reflect the diversity of the antibodies, DCIA [25] is used to calculate the concentration. The greater the distance between the antibodies is, the smaller the distance concentration will be. The use of the DCIA algorithm that is conducive to obtaining the optimal solution set rapidly, can ensure the diversity of the antibodies so that a trap into local optimal will not occur. The expressions are

$$C(x_i) = 1 - \frac{d_i}{d} = 1 - \frac{\sum_{j=1, j\neq i}^{m} \|x_i - x_j\|}{\sum_{i=1}^{m}\sum_{j=1, j\neq i}^{m} \|x_i - x_j\|} \tag{6a}$$

$$d = \sum_{i=1}^{m}\sum_{j=1, j\neq i}^{m} \|x_i - x_j\| \tag{6b}$$

$$d_i = d(x_i) = \sum_{j=1, j\neq i}^{m} \|x_i - x_j\| \tag{6c}$$

where x_i is the ith antibody and $C(x_i)$ is the distance concentration of antibody x_i. d is the sum of distances between antibodies in the population. d_i is the sum of the distances between the ith antibody and other antibodies. m is the size of population. In addition, the affinity function $D(x_i)$ is another factor that determines the probability of cloning. The formulas are

$$D(x_i(t)) = \frac{1}{2} f(x_i(t))^2 \tag{7}$$

where $f(x_i(t))$ is the fitness of antibody $x_i(t)$. The detailed process is shown in Algorithm 1.

Algorithm 1 DCIA Algorithm

Input: the size of population, m; iteration times, maxgen; the number of elite archives, o; cossover probability, p_c; mutation probability, p_m, evaluation index of population diversity, p_s, training sample; test samples.

Output: the best global solution $G(t)$

while $i <$ maxgen

for $j = m$ **do**

Calculate the fitness $f(x_i(t))$ of antibody x_i using (9a) and (9b).

Calculate antibody concentration $C(x_i)$ using (6a)–(6c).

Calculate the affinity function of antibodies using (7).

end

Calculate the expectation cloning rate $P(x_i)$ of antibody x_i using (3)–(5b).

Find the best global solution $G(t)$ using (5b).

Generate the elite solution $E_i(t)$ according to the affinity $A(x_i)$.

Select the parent population using (4).

Simulated binary crossover followed by polynomial-based mutation according to [28].

If $f(x_i(t)) < 10^{-3}$ **do**

break;

end

$i = i + 1$

end while

Remark 1: In the immune algorithm, the distance concentration that reflects the diversity of antibodies, is directly calculated in the DCIA algorithm without setting the threshold, and the antibodies with the smaller affinity and lower concentration can be stored in elite archives.

B. DCIA-SORBF Neural Network

To adjust the network size and the parameters during the training process, the DCIA-SORBFNN is proposed in this section. The proposed DCIA-SORBFNN algorithm is summarized in Algorithm 2. From Fig. 2, we can see that an antibody is a complete RBF neuron network as shown in Fig. 2(a) (i.e., the RBF centers, the widths of RBF neuron, and the output weights).

Algorithm 2 DCIA-SORBF Algorithm

Input: the same as Algorithm 1; maximum number of neurons; maxK_i; the input and output dimension of samples, l and v;

Output: the best global solution $G(t)$;

for $j = 1{:}m$ **do**

K_j = round((maxK_j – 1)*rand + 1), A_i = rand((l – 2) × 2 + 2, K_j).

end

while $s <$ maxgen **do**

for $i = 1 : m$

Calculate the fitness $f(x_i(t))$ of antibody x_i using (9a) and (9b).

Find the sample s_k with the most frequent occurrence of maximum error using (14a)–(14f).

Calculate the input and output information processing strengths (IPSs) of the jth hidden neuron in the ith antibody,

$U_{ij}(t)$ and $U_{ij}^{*}(t)$ using (11a) and (11b).

end

Calculate antibody concentration $C(x_i)$ using (6a)–(6c).

Calculate the expectation cloning rate $P(x_i)$ of antibody x_i using (3)–(5b).

Find the best global solution $G(t)$ using (5b).

Generate the elite solution $E_i(t)$ according to the affinity $A(x_i)$.

Self-organize the network structure of antibodies and update parameters of each antibody using (13)–(17).

Select the parent population using (4).Simulated binary crossover followed by polynomial-based mutation according to [28]

If $f(x_i(t)) < 10^{-3}$ **do**

break;

end

$i = i + 1$

end while

Firstly, the population is randomly initialized. Then, different antibodies have different network size and parameters. The initialized variables are given by

$$\boldsymbol{A} = (\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_i, \ldots, \boldsymbol{A}_m) \tag{8a}$$

$$\boldsymbol{A}_i = \left(\mu_{i,1}^T, \sigma_{i,1}, \omega_{i,1}; \mu_{i,2}^T, \sigma_{i,2}, \omega_{i,2}; \ldots; \mu_{i,K}^T, \sigma_{i,K}, \omega_{i.K}\right) \tag{8b}$$

where $\boldsymbol{A}$ is the antibodies population, $\boldsymbol{A}_i$ is the ith antibody which has K hidden layer neurons. $\mu_{i,K}$, $\sigma_{i,K}$, $\omega_{i,K}$ are the center, width and output weight of the Kth hidden neuron in the ith antibody. To self-organize the network, K is a random integer. For the sake of improving the accuracy of algorithm, the error criterion is selected as the fitness value of each antibody. The proposed expression is

$$f(x_i(t)) = e_i(t) \tag{9a}$$

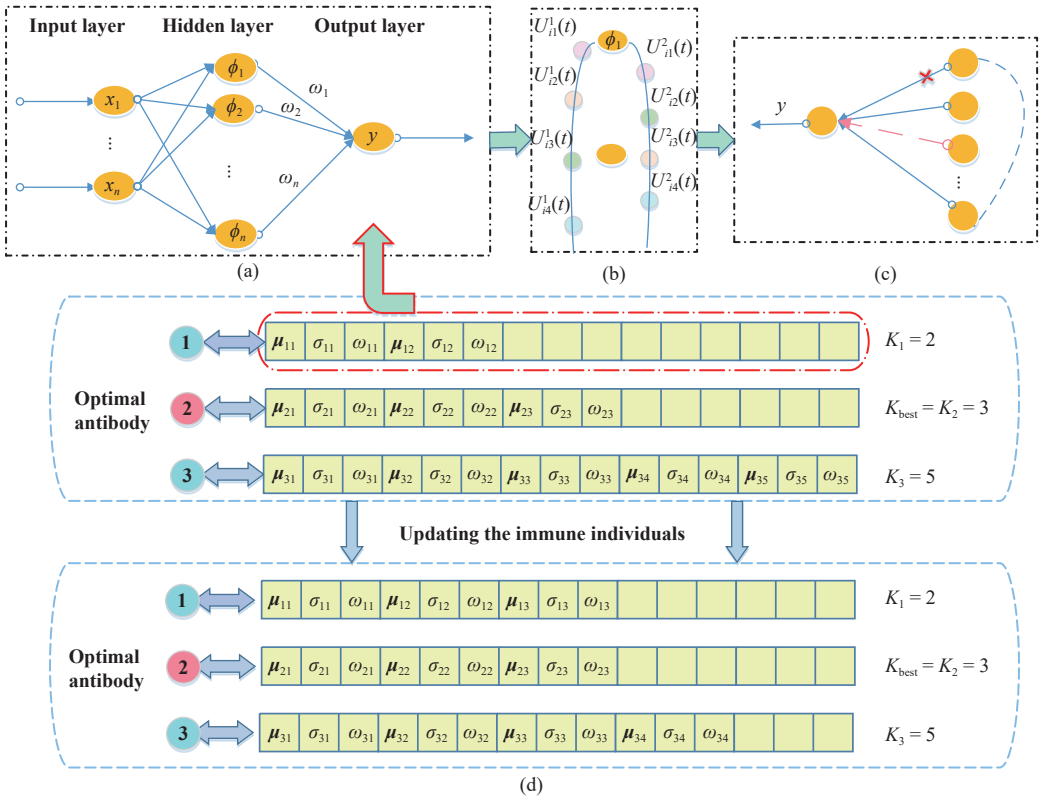


Fig. 2. DCIA-RBFNN neural network training process.

$$e_i(t)=\sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_i(t)-y_{id}(t))^2} \tag{9b}$$

where $e_i(t)$ is the root-mean-square error (RMSE) of the ith antibody. T is the number of training samples, and $y_i(t)$ and $y_{id}(t)$ are the network actual output and predictive output of the ith antibody at time t, respectively. In order to ensure the convergence of the algorithm, it is necessary to have a set range for the value of $D_{\max}(x_i(t))$. $D_{\max}(x_i(t)) < 2|e_i(t)|/((2+n)K_i(t))^{1/2}$. $D_{\max}(x_i(t))$ is the maximum value of $D(x_i(t))$, $K_i(t)$ is the network size of the ith antibody, and n is the number of input variable.

As seen in Fig. 2, the 2nd immune antibody is the optimal antibody which is obtained by (9a) and (9b) at time t. Then the optimal size of RBF neural network is obtained and the dimensions of other antibodies need to be updated (Fig. 2(d)). The network size which satisfies the following condition, is updated

$$K_i=\begin{cases}\text{Case1 or Case3} & K_{\text{best}}<K_i\\ \text{Case2 or Case3} & K_{\text{best}}\geq K_i\end{cases} \tag{10}$$

where K_{best} is the network size of the optimal antibody, and K_i is the network size of the ith antibody (i = 1, 2, …, s). Each adjustment process divides or deletes a hidden layer neuron. However, in [21] a neuron is deleted or added randomly and the network sometimes is unsteady.

To update the dimensions stably, the information-oriented error compensation algorithm (IOECA) is proposed in this paper. In IOECA, the information processing strengths (IPSs) [16] according to the independent information between the neurons in the hidden layer and their independent contribution to the output neurons, the information processing intensity (IPSs) of the neurons in the hidden layer in the learning process is calculated to identify the neurons in the hidden layer that need to be updated. The methods are given as

$$\begin{cases}U_{ij}(t)=\dfrac{1}{q}\displaystyle\sum_{q=1}^{S}e^{-\|x(t-q+1)-c_{ij}(t-q+1)\|}\\ U_{ij}^{*}(t)=H_{ij}(t)\end{cases} \tag{11a}$$

$$H_{ij}(t)=\frac{\displaystyle\sum_{q=1}^{S}d_{ij}(t-q+1)}{\displaystyle\sum_{q=1}^{S}\sum_{j=1}^{K}d_{ij}(t-q+1)}$$

$$q=1,\ldots,S;\ j=1,\ldots,K \tag{11b}$$

where $U_{ij}(t)$ and $U_{ij}^{*}(t)$ are the input and output information processing strengths (IPSs) of the jth hidden neuron in the ith antibody. S is the number of the samples. $H_{ij}(t)$ is the independent component contribution. In IPSs, by calculating the independent information of the neurons in the hidden layer, the contribution of the neurons in the independent hidden layer to the output neurons is obtained. Here, the information-oriented algorithm (IOA) which is an independent component analysis method is used. The expressions [16] are shown in (12a)–(12f)

$$\boldsymbol{C}_i(t)=\boldsymbol{\psi}_i(t)\gamma_i(t) \tag{12a}$$

$$\boldsymbol{C}_i(t)=\begin{pmatrix} d_{i1}(t-S+1) & d_{i1}(t-S+2) & \cdots & d_{i1}(t) \\ \vdots & \vdots & \ddots & \vdots \\ d_{ij}(t-S+1) & d_{ij}(t-S+2) & \cdots & d_{ij}(t) \\ \vdots & \vdots & \ddots & \vdots \\ d_{iK}(t-S+1) & d_{iK}(t-S+2) & \cdots & d_{iK}(t) \end{pmatrix} \tag{12b}$$

where $\boldsymbol{C}_i(t)$ is the independent contribution matrix of the ith antibody, $d_{ij}(t)$ is the independent contribution of the jth hidden neuron. Among $\boldsymbol{\Psi}_i(t)=[\phi_i(t-S+1),\ \ldots,\ \phi_i(t-1),\ \phi_i(t)]^T$ is the output matrix of hidden layer in the ith antibody, $j=1,\ldots,K$. $\boldsymbol{\gamma}_i(t)$ is the coefficients matrix which is given as

$$\boldsymbol{\gamma}_i(t)=\boldsymbol{\sigma}^{-1}(t)\boldsymbol{\psi}_i(t)\boldsymbol{\eta}_i(t)\boldsymbol{\varepsilon}_i(t) \tag{12c}$$

where $\boldsymbol{\sigma}_i(t)$, $\boldsymbol{\eta}_i(t)$, $\boldsymbol{\varepsilon}_i(t)$ are the covariance matrix of $\boldsymbol{\Psi}_i(t)$, the whitening matrix of $y_i(t)$, $y_i(t)=[y_i(t-S+1),\ldots,y_i(t-1),y_i(t)]$ and the whitening transformation matrix of $y_i(t)$, respectively. $\boldsymbol{\sigma}_i^{-1}(t)\ \boldsymbol{\Psi}_i(t)$ is a decorrelation process for $\boldsymbol{\Psi}_i(t)$ which can represent the independence between the hidden neurons. $\boldsymbol{\sigma}_i(t)$, $\boldsymbol{\Psi}_i(t)$ and $\boldsymbol{\varepsilon}_i(t)$ are given as

$$\boldsymbol{\sigma}_i(t)=E\left\{\boldsymbol{\psi}_i(t)\boldsymbol{\psi}_i^T(t)\right\} \tag{12d}$$

$$\boldsymbol{\eta}_i(t)=\boldsymbol{\Lambda}_i^{-\frac{1}{2}}(t)\boldsymbol{U}_i^T(t) \tag{12e}$$

$$\boldsymbol{\varepsilon}_i(t)=\boldsymbol{\Lambda}_i^{-\frac{1}{2}}(t)\boldsymbol{U}_i^T(t)y_i(t). \tag{12f}$$

$\boldsymbol{U}_i(t)$ and $\boldsymbol{\Lambda}_i(t)$ are the eigenvector and eigenvalue matrices of $y_i(t)$, respectively. Moreover, $\boldsymbol{\eta}_i(t)\boldsymbol{\varepsilon}_i(t)$ is used to reduce the correlation between the output layer and the hidden neurons. According to the competitive capacities of hidden neurons which are obtained by IPSs. The adjustment rules are given as shown below:

Case 1 (Neuron Splitting Rule): In fact, if $U_{ij}(t)$ is larger, the input samples are closer to the center of the jth hidden neuron. In addition, the jth hidden neuron is more active than these input samples. Meanwhile, based on (11a) and (11b), if $U_{ij}^*(t)$ is larger, the hidden neurons are more sensitive to the output neurons. Therefore, $U_{ij}(t)$ and $U_{ij}^*(t)$ can be used to describe the information processing ability of the hidden neurons. Then, the condition is described as

$$\begin{cases} U_{i\_\max}(t)=\max\ \boldsymbol{U}_i(t) \\ U_{i\_\max}^*(t)=\max\ \boldsymbol{U}_i^*(t) \end{cases} \tag{13}$$

where $\boldsymbol{U}_i(t)=[U_{i1}(t),\ldots,U_{i(K-1)}(t),U_{iK}(t)]$ and $\boldsymbol{U}_i^*(t)=\left[U_{i1}^*,\ldots,U_{i(K-1)}^*(t),U_{iK}^*(t)\right]$ are the input and output IPS vectors of hidden neurons, respectively. Only when the maximum input and output IPSs of the jth hidden neuron are satisfied can a new hidden neuron be added to the network. As for $U_{i1}(t)$ and $U_{i1}^*(t)$ in Fig. 2(b), the 1st hidden neuron should be split for the new one. In addition, it is important to determine the parameters of the new neuron. Then $x_i(t)$ and $y_i(t)$ of the samples should be determined firstly according to (11a) and (12c)–(12f). Where, IPSs [16] is used to identify the hidden layer neurons that need to be split or pruned during each iteration. However, in document [16], when the parameters need to be updated, the last of every five input samples is selected to adjust the parameters of the new hidden layer neurons. In this way, the parameters will be updated many times during each iteration. This not only affects the accuracy of calculation, but also consumes a lot of time. At the same time, in the calculation process, some samples will have poor learning effect, which will affect the overall calculation accuracy. To solve the above problem, this paper proposes an information-oriented error compensation algorithm. In this algorithm, the input samples with the highest frequency of maximum error in all samples are used to set new hidden layer neuron parameters. Thus, the calculation time is reduced while the accuracy is improved. The method of error compensation is

$$\boldsymbol{P}=(p_1,\ldots,p_k,\ldots,p_S) \tag{14a}$$

$$\boldsymbol{E}(t)=(e_1(t),\ldots,e_k(t),\ldots,e_S(t)) \tag{14b}$$

where $\boldsymbol{P}$ is the sample matrix, P_k is the kth sample, S is the size of $\boldsymbol{P}$. $\boldsymbol{E}(t)$ is the error matrix of the sample matrix $\boldsymbol{P}$ at time t.

$$\boldsymbol{n}_E=\left(n_{e_1},\ldots,n_{e_k},\ldots,n_{e_S}\right) \tag{14c}$$

$$n_{e_k}=\max\boldsymbol{n}_E \tag{14d}$$

$\boldsymbol{n}_E$ is the matrix of the sample maximum error calculation occurrences. n_{e_k} is the maximum of n_E. The kth sample P_k has the largest number and times of errors in the whole iteration process, $k\in 1, 2, \ldots, S$. X_k and Y_k are the input and output matrices for all samples, respectively. They can be described as

$$\boldsymbol{X}_k=(x_{k1},\ldots x_{ko},\ldots,x_{kl}) \tag{14e}$$

$$\boldsymbol{Y}_k=(y_{k1},\ldots,y_{ko},\ldots,y_{kv}) \tag{14f}$$

where x_{ko} and y_{ko} are the oth input and output values of sample P_k respectively. l and v are the input and output dimension of samples, separately. Subsequently, the parameters of the new neuron are given as

$$\begin{cases} \boldsymbol{c}_{i\text{new}}=\alpha\boldsymbol{c}_{ij}(t)+\beta\boldsymbol{X}_k(t) \\ \boldsymbol{\sigma}_{i\text{new}}=\alpha\sigma_{ij}(t) \\ \boldsymbol{\omega}_{i\text{new}}=\boldsymbol{Y}_k\omega_{ij}(t) \end{cases} \tag{15}$$

where $\boldsymbol{c}_{ij}(t)$, $\boldsymbol{\sigma}_{ij}(t)$ and $\boldsymbol{\omega}_{ij}(t)$ represent respectively the center, radius and weight of the jth hidden pre-split neuron in the ith immune cell at time t, and $\boldsymbol{c}_{i\text{new}}(t)$, $\boldsymbol{\sigma}_{i\text{new}}(t)$ and $\boldsymbol{\omega}_{i\text{new}}(t)$ are respectively the center, radius and connection weights of the new added hidden neuron, among which $\alpha\in[0.95, 1.05]$ and $\beta\in[0, 0.1]$. The sample that has the most error calculation occurrences is trained with compensation by (15), so that the accuracy of algorithm can be improved.

Case 2 (Neuron Deleting Rule): Based on the information processing ability of hidden neurons, the jth hidden neuron is deleted if the IPSs of the hidden neurons satisfy the following conditions:

$$\begin{cases} U_{i\_\min}(t)=\min U_i(t) \\ U_{i\_\min}^*=\min U_i^*(t). \end{cases} \tag{16}$$

The connection weights of the j'th hidden neuron will be updated according to

$$\omega_{ij'}'(t)=\omega_{ij'}(t)+\boldsymbol{Y}_k\omega_{ij'}(t). \tag{17}$$

Before the jth hidden neuron is cut off, the j'th hidden

neuron is the one that is nearest the jth hidden neuron. Before and after the jth hidden neuron is cut off. $\omega_{ij'}'(t)$ and $\omega_{ij'}(t)$ are the connection weights between the j'th hidden neuron and the output layer, respectively. The center and radius of the j'th hidden neuron remain unchanged after the jth hidden neuron is deleted.

Case 3 (Neuron Retaining Rule): If the input and output IPSs of the hidden neurons are neither the maximum nor minimum information strength (such as (13) or (16)), the structure of the ith immune cell does not change. With the self-organizing mechanism, the structure of the immune cell can be automatically organized to improve the performance. In addition, the prediction accuracy of the system will be improved based on the error compensation method.

Remark 2: In [21], the neuron that is split or deleted is random. Then, the network is unstable. To solve this problem, the IOA is used to provide the split or deleted rules in the artificial immune algorithm.

Remark 3: In IOECA-SORBF, the error compensation is used. When the structure of the RBF network needs to be split or deleted, the parameters of the new neuron need to be updated. However, in [16], the parameters of the last sample in every five samples are used to update the newly added hidden layer neurons, so that the accuracy cannot be ensured. To increase the network prediction accuracy, the neural network sample with the most frequent occurrence of error is used to regulate the parameters of the new neuron. Therefore, the output error is compensated appropriately, and the prediction accuracy of the RBF neural network is increased.

IV. Convergence Analysis

For the proposed DCIA-SORBFNN, the convergence of the algorithm is an important issue and needs to be carefully investigated. In this section, the analysis of convergence is provided in detail to guarantee the successful application of the proposed DCIA-SORBFNN. Furthermore, one can obtain a better understanding of the DCIA-SORBFNN through this analysis.

A. Convergence Analysis of DCIA Algorithm

The operations in each generation of crossover, mutation, and antibody concentration regulation actually correspond to the state transition process, which come from one antibody population $p_i^{(t)} = \{x(t) = i\}$ to another antibody population $p_j^{(t+1)} = \{x(t+1) = j\}$. P_{ij} is the transfer process from the population $p_i^{(t)}$ to the population $p_j^{(t+1)}$. P_{ij} is only related to the previous population state and is independent of evolutionary algebra. Therefore, the antibody state transfer process can be regarded as a finite homogeneous Markov chain. The stability of crossover, mutation and selection operation is to prove respectively below.

Definition 4: Set up F_k as the best antibody in the kth moment, F^* is the antigen of the problem to be solved, if and only if $\lim_{k\to\infty} \boldsymbol{P}(F_k = F^*) = 1$ is established, the DCIA algorithm is convergent [29].

According to the definition in [29], the crossover operator may be regarded as a random total function whose domain and range are R. The state space $R = IB^N = IB^{l.n} = \{0,1\}^{l.n}$, where n is the population size, and l is the number of genes. i.e., each state of R is mapped probabilistically to another state. Therefore, the state transition matrix C is stochastic.

The probability that state i becomes state j after mutation can be aggregated as [29] because the mutation operator is applied independently to each gene/bit in the population

$$v_{ij} = p_m^{H_{ij}}(1-p_m)^{N-H_{ij}} > 0 \tag{18}$$

where $p_m \in (0,1)$, all $i, j \in S$, H_{ij} denotes the Hamming distance between the binary representations of state i and state j. The length of each antibody is set as l, and the hamming distance between state i and j is defined as

$$H_{ij} = \sum_{k=1}^{l} \left| g_{ik} - g_{jk} \right| \tag{19}$$

where g_{ik} and g_{jk} are the kth gene of the x_i and x_j immune cells, respectively. The same holds for other operators and their transition matrices. Thus, $\boldsymbol{M}$ is positive. The probability $\boldsymbol{S}$ whose selection does not alter the state generated by mutation is column-allowable. Subsequently, the state transition of one generation antibody is completed, where $\boldsymbol{P} = \boldsymbol{SCM}$ is positive [29].

Definition 5: A square matrix $\boldsymbol{A}$: $n \times n$ is said to be reducible, if A can be brought into the form (with square matrices $\boldsymbol{C}$ and $\boldsymbol{T}$)

$$\begin{pmatrix} \boldsymbol{C} & \boldsymbol{0} \\ \boldsymbol{R} & \boldsymbol{T} \end{pmatrix} \tag{20}$$

by applying the same permutations to rows and columns.

Theorem 1: Let $\boldsymbol{P}$ a reducible stochastic matrix, where C: $m \times m$ is a primitive stochastic matrix and R, $T \neq 0$. Then

$$\boldsymbol{P}^{\infty} = \lim_{k\to\infty} \boldsymbol{P}^k = \lim_{k\to\infty} \begin{pmatrix} \boldsymbol{C}^k & \boldsymbol{0} \\ \sum_{i=1}^{k-1} \boldsymbol{T}^i \boldsymbol{R} \boldsymbol{C}^{k-i} & \boldsymbol{T}^k \end{pmatrix} = \begin{pmatrix} \boldsymbol{C}^{\infty} & \boldsymbol{0} \\ \boldsymbol{R}^{\infty} & \boldsymbol{0} \end{pmatrix}. \tag{21}$$

It is a stable stochastic matrix with $\boldsymbol{P}^{\infty} = 1'\boldsymbol{P}^{\infty}$, where $\boldsymbol{P}^{\infty} = P^0 P^{\infty}$ is unique regardless of the initial distribution, and $\boldsymbol{P}^{\infty}$ satisfies: $p^{\infty}{}_i > 0$ for $1 \le i \le m$; $p^{\infty}{}_i = 0$ for $m < i \le n$.

Theorem 2: The immune algorithm based on distance concentration is globally convergent to probability 1.

Proof: The state transition matrix of immune algorithm is denoted by $\boldsymbol{U} = (u_{ij})$, $u_{ij} \in [0, 1]$, $\sum_{j=1}^{n} u_{ij} = 1$ [29]. And the state transition matrix $\boldsymbol{U}$ is stochastic matrices.The states of transition matrix $\boldsymbol{U}$ are as follows: the first state is global optimal solution. The second state is global suboptimal solution, …, the nth state is the solution of the worst. Then, for any state, $u_{ii} + \sum_{j=i+1}^{n} u_{ij} \to u_{ii}$, and $u_{ii} = 0, \forall\ j > i$, the upgraded matrix can be written as

$$\boldsymbol{U} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ u_{21} & u_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ u_{n1} & u_{n2} & \cdots & u_{nn} \end{pmatrix}. \tag{22}$$

With $\boldsymbol{P} = \boldsymbol{SCM}$ the transition matrix for DCIA becomes

$$\boldsymbol{P}^{*}=\begin{pmatrix}\boldsymbol{P} & & & \\ & \boldsymbol{P} & & \\ & & \ddots & \\ & & & \boldsymbol{P}\end{pmatrix}\begin{pmatrix}1 & 0 & \cdots & 0\\ u_{21} & u_{22} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ u_{n1} & u_{n2} & \cdots & u_{nn}\end{pmatrix}$$

$$=\begin{pmatrix}\boldsymbol{P} & & & \\ \boldsymbol{P}u_{21} & \boldsymbol{P}u_{22} & & \\ \vdots & \vdots & \ddots & \\ \boldsymbol{P}u_{n1} & \boldsymbol{P}u_{n2} & \cdots & \boldsymbol{P}u_{nn}\end{pmatrix}=\begin{pmatrix}\boldsymbol{C} & \boldsymbol{0}\\ \boldsymbol{R} & \boldsymbol{T}\end{pmatrix} \tag{23a}$$

$$\boldsymbol{R}=\begin{pmatrix}\boldsymbol{P}u_{21}\\ \vdots\\ \boldsymbol{P}u_{n1}\end{pmatrix},\ \boldsymbol{T}=\begin{pmatrix}\boldsymbol{P}u_{22} & \cdots & 0\\ \vdots & \ddots & \vdots\\ \boldsymbol{P}u_{n2} & \cdots & \boldsymbol{P}u_{nn}\end{pmatrix} \tag{23b}$$

$\boldsymbol{C}=[1]$ is a first order stochastic matrix. The submatrices $\boldsymbol{P}u_{a1}$ with $a \geq 2$ can be gathered in a rectangular matrix $R \neq 0$. According to Definition 5, the state transition matrix is reducible [29]. From Theorem 1

$$\boldsymbol{P}^{*\infty}=\lim_{k\to\infty}\boldsymbol{P}^{*k}=\begin{pmatrix}1 & 0\\ \boldsymbol{R}^{\infty} & 0\end{pmatrix} \tag{24a}$$

$$\lim_{k\to\infty}\boldsymbol{P}(F_k=F^{*})=1. \tag{24b}$$

So that Theorem1 and Definition 4 may be used to prove that the DCIA converges to the global optimum [29]. ■

B. Convergence Analysis of DCIA-SORBF Algorithm

According to the above description, a simple and efficient DCIA algorithm with a special fitness function is used to automatically construct the RBFNN. The goal of the DCIA algorithm is to construct an appropriate RBFNN. After the DCIA algorithm returns a set of immune cells, the parameters and size of the DCIA-SORBFNN corresponding to each immune cell can be obtained by using (8b).

To provide the theoretical basis for the applications, this section presents the convergence analysis of the DCIA-SORBF neural network based on the convergence analysis of the DCIA algorithm. The convergence analysis of the DCIA-SORBFNN is summarized in Theorem 3.

Theorem 3: If the bounds of the predefined maximum $D_{\max}(x_i(t)) < 2|E_i(t)|/((2+n)K_i(t))^{1/2}$, then the DCIA-SORBFNN is convergent, and $\boldsymbol{E}_i(t) \to 0$ as $t \to 0$, $i = 1, 2, \ldots, s$.

Proof: Consider the following Lyapunov function:

$$D_i(t)=\frac{1}{2}e_i^2(t). \tag{25a}$$

According to (9b), the system error is [30]

$$e_i(t)=\sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_i(t)-y_{id}(t))^2}=\sqrt{J}. \tag{25b}$$

Then the change in the Lyapunov function between two steps is

$$\Delta D_i(t)=D_i(t+1)-D_i(t)=\frac{1}{2}\left[e_i^2(t+1)-e_i^2(t)\right]. \tag{26}$$

In addition, the error change is denoted:

$$e_i(t+1)=e_i(t)+\Delta e_i(t). \tag{27}$$

Then, the strictly differential formula of the RMSE is

$$\Delta e_i(t)=\frac{\partial e_i(t)}{\partial \boldsymbol{\mu}_i(t)}[\Delta\boldsymbol{\mu}_i(t)]^T+\frac{\partial e_i(t)}{\partial \boldsymbol{\sigma}_i(t)}[\Delta\boldsymbol{\sigma}_i(t)]^T+\frac{\partial e_i(t)}{\partial \boldsymbol{\omega}_i(t)}[\Delta\boldsymbol{\omega}_i(t)]^T \tag{28}$$

where $\boldsymbol{\omega}_i(t)=[\omega_{i,1}(t),\omega_{i,2}(t),\ldots,\omega_{i,Ki}(t)]$, $\boldsymbol{\mu}_i(t)=[\mu_{i,1}(t),\mu_{i,2}(t),\ldots,\mu_{i,Ki}(t)]$, and $\boldsymbol{\sigma}_i(t)=[\sigma_{i,1}(t),\sigma_{i,2}(t),\ldots,\sigma_{i,Ki}(t)]$, are the three adjusted parameters in the RBFNN.

$$\begin{aligned}\Delta\boldsymbol{\omega}_i(t)&=-\frac{\partial J}{\partial\omega}=-2e_i(t)\frac{\partial e_i(t)}{\partial\boldsymbol{\omega}_i(t)}=D_{i,\omega}(t)\\ \Delta\boldsymbol{\mu}_i(t)&=-\frac{\partial J}{\partial\mu}=-2e_i(t)\frac{\partial e_i(t)}{\partial\boldsymbol{\mu}_i(t)}=D_{i,\mu}(t)\\ \Delta\boldsymbol{\sigma}_i(t)&=-\frac{\partial J}{\partial\sigma}=-2e_i(t)\frac{\partial e_i(t)}{\partial\boldsymbol{\sigma}_i(t)}=D_{i,\sigma}(t)\end{aligned} \tag{29}$$

where $\Delta\boldsymbol{\omega}_i(t)$, $\Delta\boldsymbol{\mu}_i(t)$ and $\Delta\boldsymbol{\sigma}_i(t)$ are the parameter updating rules. $\boldsymbol{D}_i(t)=[D_{i,\mu}(t), D_{i,\sigma}(t), D_{i,\omega}(t)]$ is the affinity function of ith immune cell. According to the above analysis, the factorization of (30) is described as

$$\Delta D_i(t)=2e_i(t)\Delta e_i(t)+\Delta e_i^2(t)=2e_i^2(t)\boldsymbol{\Omega}(t)(\boldsymbol{\Omega}(t)-1) \tag{30}$$

where

$$\boldsymbol{\Omega}(t)=\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\omega}_i(t)}\right|^2+\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\mu}_i(t)}\right|^2+\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\sigma}_i(t)}\right|^2. \tag{31}$$

The following conditions can be obtained [21] because the bounds of the predefined maximum distance concentration is adjusted dynamically, and $D_{\max}(x_i(t)) < 2|E_i(t)|/((2+n)K_i(t))^{1/2}$

$$\begin{aligned}\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\omega}_i(t)}\right|^2&=\left|\frac{D_{i,\omega}(t)}{2e_i(t)}\right|^2<\frac{K_i(t)}{(2+n)K_i(t)}\\ \left|\frac{\partial e_i(t)}{\partial\boldsymbol{\mu}_i(t)}\right|^2&=\left|\frac{D_{i,\mu}(t)}{2e_i(t)}\right|^2<\frac{K_i(t)}{(2+n)K_i(t)}\\ \left|\frac{\partial e_i(t)}{\partial\boldsymbol{\sigma}_i(t)}\right|^2&=\left|\frac{D_{i,\sigma}(t)}{2e_i(t)}\right|^2<\frac{K_i(t)}{(2+n)K_i(t)}\end{aligned} \tag{32}$$

where $K_i(t)$ is the number of hidden neurons at time t for the ith immune cell.

$$\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\omega}_i(t)}\right|^2+\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\mu}_i(t)}\right|^2+\left|\frac{\partial e_i(t)}{\partial\boldsymbol{\sigma}_i(t)}\right|^2<1 \tag{33}$$

which leads to

$$\Delta D_i(t)<0. \tag{34}$$

Thus, $e_i(t)$ is bounded for $t \geq t_0$. Moreover, through the Lyapunov-like lemma, it is implied that

$$\lim_{t\to\infty}e_i(t)=0. \tag{35}$$

Therefore, $e_i(t)\to 0$ as $t \to \infty$, $i = 1, 2, \ldots, s$.

In addition, the structure of self-organizing phase stage converges according to [31], thus, the convergence of the proposed DCIA-SORBF neural network is proved. ■

Remark 4: Based on the above discussion, in the adjustment phase of the parameter and network size, the convergence of DCIA-SORBF neural network can be maintained according to formulas from (25a) to (35). Then, the convergence of DCIA-SORBF neural network that is necessary for successful applications, can be guaranteed by using the DCIA algorithm.

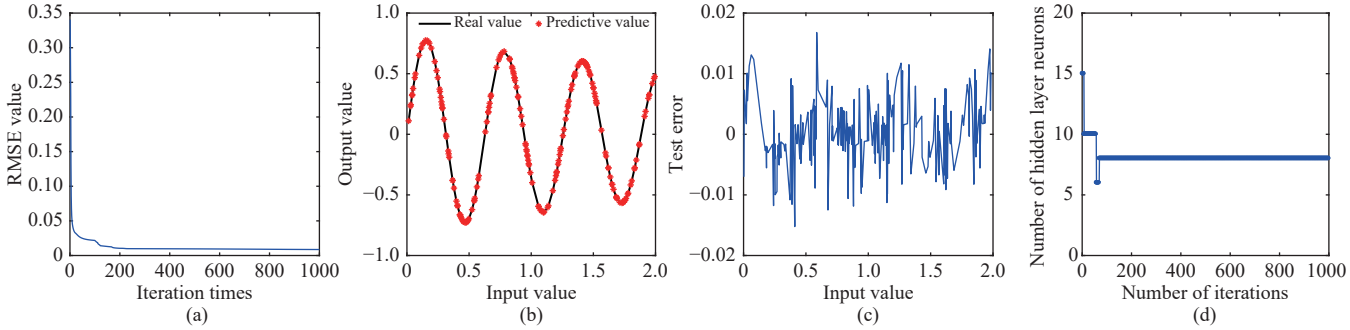


Fig. 3. Experimental results of the proposed DCIA-SORBF neural network for function approximation. (a) RMSE values against iterations. (b) Approximation error versus input values. (c) Approximation results from the APSO-SORBF neural network. (d) Number of hidden neurons against iterations.

TABLE I
COMPARISON OF DIFFERENT ALGORITHMS FOR THE FUNCTION APPROXIMATION

Algorithm	Testing RMSE		No. of hidden neurons	Testing time (s)	Mean/Dev. rank
	Mean	Dev.			
DCIA-SORBF	0.0122	0.0035	8	0.0032	1/2
APSO-SORBF [21]	0.0133*	0.0056*	9*	0.0039*	2/3
AI-RBF [37]	0.0235	0.0092	10	0.0062	4/5
GAP-RBF [13]	0.0415*	0.0087*	19*	0.0087*	8/4
PSO-RBF [36]	0.0368*	0.0164*	13*	0.0054*	7/6
AI-PSO-RBF [34]	0.0295*	0.073*	11*	0.0042*	6/7
SAIW-PSO-RBF [35]	0.0197*	0.0026*	11*	0.0046*	3/1

* The results are also listed in the original papers.

V. DCIA-SORBF SIMULATION AND APPLICATION

In this section, five systems are used to demonstrate the effectiveness of DCIA-SORBF. They are the Mackey-Glass time series prediction, nonlinear system identification, the Lorenz time series prediction focusing on nonlinear system modeling or prediction problems, and the effluent total phosphorus (TP) prediction. Whereas TP is an actual industrial problem in a wastewater treatment process (WWTP). In addition, the fitness is used to reflect the diversity of DCIA-SORBF. And six algorithms are used to compare the performance with it. They are APSO-SORBF [21], GAP-RBF [13], (AI-PSO-RBF) [34], and stability adaptive inertia weight PSO-based RBF (SAIW-PSO-RBF) [35], separately. All the examples were programmed in MATLAB R2014a and ran on a PC with a clock speed of 2.60 GHz and 4 GB RAM under a Microsoft Windows 8.0 environment.

A. Function Approximation

In this example, the DCIA-SORBF neural network is used to approximate the following benchmark problem:

$$y = 0.8e^{-0.2x}\sin(10x). \tag{36}$$

This function is used to examine many popular algorithms in [13], [32] and [33]. There are 300 training patterns that are generated randomly on the domain $[0, 2]$, along the X-direction. Similarly, the testing samples are also randomly produced in the range $[0, 2]$. In addition, the testing set contains 200 samples. In Fig. 3, four indices are used to reflect the DCIA-SORBF neural network performance: the number of hidden neurons, the training RMSE, the approximation error, and the testing output. In addition, the proposed DCIA-SORBF algorithm is compared with six other algorithms. All algorithms use the same training data sets and test partitions. And the initial parameters of DCIA-SORBF neural network are set as: the cross probability $p_c = 0.4$, the mutation probability $p_m = 0.35$, the parameter of diversity evaluation $p_s = 0.85$, the elite file size $E_a = 60$, and the maximum number of neurons in an antibody $max\_num = 60$.

To compare these algorithms, four performance parameters are shown in Table I. This table shows the results of the mean value and standard deviation (Dev.) of testing RMSE, number of hidden neurons, and testing time. The results show that the structure of DCIA-SORBF neural network is the most compact one for its self-organizing capability. Moreover, the proposed DCIA-SORBF neural network requires the least testing time of all algorithms. And Wilcoxon' rank is added to verify the effectiveness of the proposed DCIA-SORBF algorithm. Mean/Dev. rank is used to rank the results of mean and Dev. Among them, the left side is the ranking result of mean, the right side is the ranking result of Dev. We can see that DCIA-SORBF gets the first mean and the second Dev. value.

B. Mackey-Glass Time Series Prediction

The Mackey-Glass time series prediction problem which is one of the benchmark problems. It is used to assess the performance of learning algorithms [21]. The time series prediction is generated by the following equation:

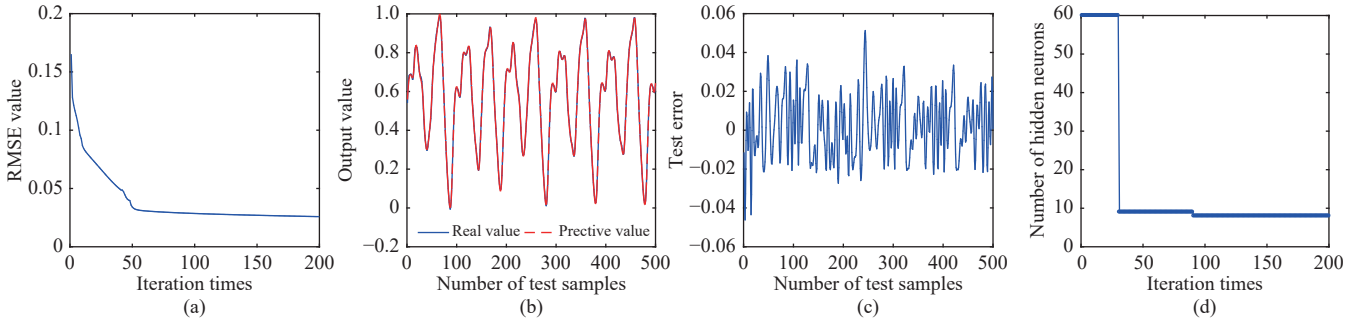


Fig. 4. Experimental results of the proposed DCIA-SORBF neural network for Mackey-Glass time series. (a) RMSE values against iterations. (b) Approximation error versus input values. (c) Approximation results from the APSO-SORBF neural network. (d) Number ofhidden neurons against iterations.

TABLE II
COMPARISON OF DIFFERENT ALGORITHMS FOR THE MACKEY-GLASS TIME SERIES PREDICTION

Algorithm	Testing RMSE		No.of hidden neurons	Testing time (s)	Mean/Dev. rank
	Mean	Dev.			
DCIA-SORBF	0.0116	0.0075	9	0.0035	1/1
APSO-SORBF [21]	0.0135*	0.0095*	11*	0.0039*	2/2
AI-RBF [37]	0.0151	0.0128	11	0.0042	3/3
GAP-RBF [13]	0.0321*	—	19*	—	7/—
PSO-RBF [36]	0.0208*	0.0249*	12*	0.0047*	6/6
AI-PSO-RBF [34]	0.0189*	0.0132*	11*	0.0043*	5/4
SAIW-PSO-RBF [35]	0.0166*	0.0145*	11*	0.0053*	4/5

* The results are also listed in the original papers.

$$x(t+1) = (1-\alpha)x(t) + \frac{bx(t-\tau)}{1+x^{10}(t-\tau)} \tag{37a}$$

where $\alpha = 0.1$, $b = 0.2$, and $\tau = 17$, and the initial condition $x(0) = 1.2$. The value $x(t + \Delta t)$ is predicted from the previous values $\{x(t), x(t - \Delta t), \ldots, x(t - (l - 1)\Delta t)\}$. In this paper, the prediction model is given by

$$x(t+\Delta t) = f(x(t), x(t-6), x(t-12), x(t-18)). \tag{37b}$$

In the simulation experiment, 1700 data points were selected from $t = 1$ to $t = 1700$. Among them, the first 1200 data points are used for training, and the last 500 data points are used as test data. The initial network size is set to 60. And $pc = 0.45$, $pm = 0.55$, $ps = 0.9$, $Ea = 60$ are selected as the best parameters. The experimental results are shown in Fig. 4. The proposed algorithm can track the Mackey-Glass time series problem well, and the test error is within the range of [–0.05, 0.05], and the test error is small. The network structure is constantly adjusted during the process of iteration. Finally, the performance is the best when the network structure is 8. As can be seen from Table II, DCIA-SORBF has the smallest Mean and Dev values compared with other algorithms. At the same time, it has the smallest test time because the algorithm has the smallest network structure. In addition, APSO-SORBF [21] outperforms other algorithms except DCIA-SORBF and ranks second. And AI-RBF [37], SAIW-PSO-RBF [35], AI-PSO-RBF [34] and PSO-RBF [36] ranked third to sixth, respectively. GAP-RBF [13] has the worst test error. Therefore, DCIA-SORBF has the smallest test error and the best system stability compared with other algorithms.

C. Nonlinear System Identification

The nonlinear system is given by

$$y(t+1) = 0.72y(t) + 0.025y(t-1)u(t-1) + 0.01u^2(t-2) + 0.2u(t-3). \tag{38}$$

There are two input values, $y(t)$ and $u(t)$, and the output $y(t + 1)$. The nonlinear system is used in [17], [13], and [36] to demonstrate the performance of a neural network. The training inputs were obtained from two parts. Half of them were sequenced uniformly over the interval [–2, 2], and the others were generated by $1.05 \times \sin(t/45)$. Besides, 2400 and 1000 samples were selected for training and testing. The testing samples of input were set as

$$u(t) = \begin{cases} \sin\left(\dfrac{\pi t}{25}\right), & 0 < t < 250 \\ 1.0, & 250 < t < 500 \\ -1.0, & 500 < t < 750 \\ 0.3\sin\left(\dfrac{\pi t}{25}\right) + 0.1\left(\dfrac{\pi t}{32}\right) + 0.6\left(\dfrac{\pi t}{10}\right), & 750 < t < 1000 \end{cases} \tag{39}$$

where $u(t)$ is the input signal which used to determine the identification results for the testing signal. To evaluate the performance of DCIA-SORBF neural network, its results are compared with those of six other neural networks. Fig. 5 records the RMSE values, prediction results, prediction errors and the number of hidden neurons for DCIA-SORBF. The number of hidden neurons self-organization adjustments is shown in Fig. 5 (d). And we can see that DCIA-SORBF neur-

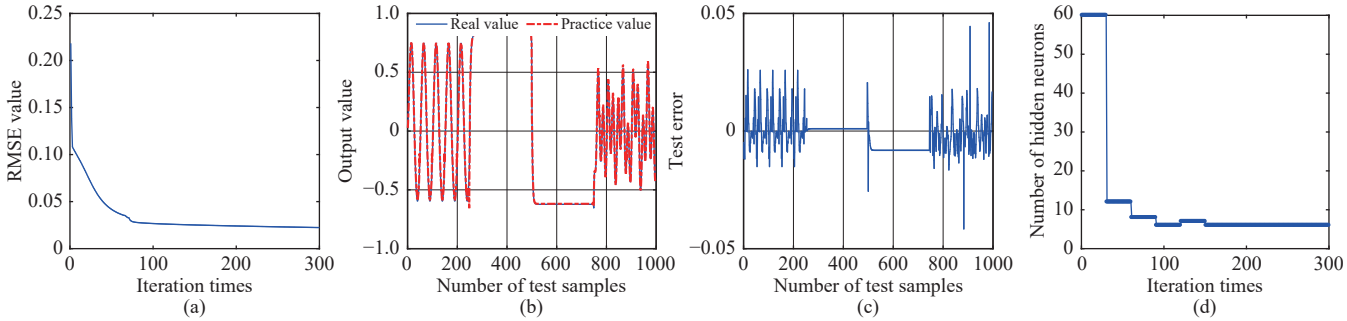


Fig. 5. Experimental results of the proposed DCIA-SORBF neural network for nonlinear system. (a) RMSE values against iterations. (b) Approximation error versus input values. (c) Approximation results from the APSO-SORBF neural network. (d) Number of hidden neurons against iterations.

TABLE III
COMPARISON OF DIFFERENT ALGORITHMS FOR THE NONLINEAR SYSTEM IDENTIFICATION

Algorithm	Testing RMSE		No.of Hidden neurons	Testing time (s)	Mean/Dev. rank
	Mean	Dev.			
DCIA-SORBF	0.0724	0.0035	7	0.0039	1/1
APSO-SORBF [21]	0.0916	0.0116	11*	0.0047*	2/4
AI-RBF [37]	0.1049	0.052	11	0.0062	4/7
GAP-RBF [13]	0.2229*	0.0165*	15*	0.0068*	6/6
PSO-RBF [36]	0.2564*	0.0126*	12*	0.0049*	7/5
AI-PSO-RBF [34]	01536*	0.0109*	14*	0.0051*	5/2
SAIW-PSO-RBF [35]	0.0934*	0.0113*	13*	0.0056*	3/3

* The results are also listed in the original papers.

al network performs well and the test error remains within the range [–0.05, 0.05]. Therefore, it can predict the nonlinear system function well.

Table III exhibits the detailed results of the different algorithms. Four indices are selected to reflect the performances. They are the number of hidden neurons, the mean value and standard Dev. of the testing RMSE, and the testing time. In Table III, the mean value and standard dev. of the testing RMSE are the smallest for DCIA-SORBF. The number of the hidden neurons is the smallest and the testing time is the least for DCIA-SORBF. And APSO-SORBF is the second only to DCIA-SORBF, ranking second. PSO-RBF performed poorest for nonlinear system identification. This example shows that the DCIA-SORBF neural network has better identification ability and its structure is more compact.

D. Lorenz Time Series Prediction

The Lorenz time series system is a mathematical model for atmospheric convection that is also widely used as a benchmark in many applications [33]. As a 3-D and highly nonlinear system, the Lorenz system is governed by

$$\begin{cases} \dfrac{dx(t)}{dt} = a_1 y(t) - a_1 x(t) \\ \dfrac{dy(t)}{dt} = a_2 x(t) - x(t)z(t) - y(t) \\ \dfrac{dz(t)}{dt} = x(t)y(t) - a_3 z(t) \end{cases} \tag{40}$$

where a_1, a_2, and a_3 are the system parameters $a_1 = 10$, $a_2 = 28$, and $a_3 = 8/3$; $x(t)$, $y(t)$, and $z(t)$ are the 3-D space vectors of the Lorenz system. In this example, the fourth-order Runge-Kutta approach with a step size 0.01 is adopted to generate the Lorenz samples, and only the Y-dimension samples $y(t)$ are used for the time series prediction. For 3400 data samples generated from $y(t)$, the first 2400 samples were taken as training data, and the last 1000 samples were used to check the proposed model. The ratio is close to 7: 3. The test results in Fig. 6 show that DCIA-SORBF neural network performs well and the test error remains within the range [–0.05, 0.05]. And the network structure is constantly adjusted during the process of iteration. Finally, the performance is the best when the network structure is 9. Moreover, six algorithm: APSO-SORBF[21], GAP-RBF [13], MRL-QPSO-RBF [33], PSO-RBF [36], AI-PSO-RBF [34], and SAIW-PSO-RBF [35] are compared with DCIA-SORBF in Table IV. This comparison show that the DCIA-SORBF neural network has the smallest mean value error and standard Dev. And the testing RMSE is far better than other algorithms except AI-RBF. In addition, AI-RBF which is worse than DCIA-SORBF is better than the other five algorithms. However, GAP-RBF shows the worst performance. Then the results indicate that DCIA-SORBF has best identification ability for Lorenz time series prediction than the other proposed algorithms.

E. Effluent TP Prediction in WWTP

The effluent TP is an important parameter for evaluating the performance of a WWTP [30]. However, the values of the effluent TP are difficult to measure due to the biological

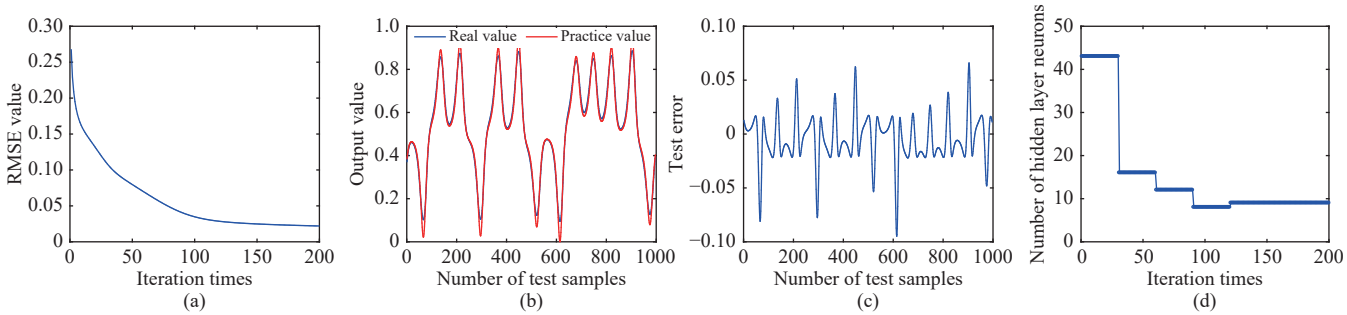


Fig. 6. Experimental results of the proposed DCIA-SORBF neural network for lorenz time series prediction. (a) RMSE values against iterations. (b) Approximation error versus input values. (c) Approximation results from the APSO-SORBF neural network. (d) Number of hidden neurons against iterations.

TABLE IV
COMPARISON OF DIFFERENT ALGORITHMS FORTHE LORENZ TIME SERIES SYSTEM

Algorithm	Testing RMSE		No.of hidden neurons	Testing time (s)	Mean/Dev. rank
	Mean	Dev.			
DCIA-SORBF	0.0958	0.026	9	0.0075	1/1
APSO-SORBF [21]	0.1726*	0.054*	5*	0.0069*	3/3
AI-RBF	0.1049	0.052	11	0.0062	2/2
GAP-RBF [13]	2.3294*	—	70*	—	7/—
PSO-RBF [36]	0.2673*	0.095*	6*	0.0076*	6/6
AI-PSO-RBF [34]	0.2017*	0.058*	6*	0.0076*	5/4
SAIW-PSO-RBF [35]	0.1981*	0.073*	5*	0.0072*	4/5

* The results are also listed in the original papers.

characteristics of the activated sludge process. The availability of the effluent TP is often associated with expensive capital and maintenance costs [37]. Therefore, the proposed DCIA-SORBF neural network is used to predict the values of the effluent TP in this experiment.

Due to the influence of accuracy measurement, operation and measurement method, water quality abrupt change and so on, the collected data have a certain degree of error. Moreover, direct soft sensor modeling of unprocessed data will inevitably lead to poor performance system and unreliable prediction results. Therefore, in order to ensure the reliability and accuracy of soft sensing, it is necessary to eliminate abnormal data. The existing TP prediction of wastewater treatment is mostly processed by noise reduction. However, all the collected data are real data, and some noise data are hard to avoid, so we do the total phosphorus experiment with real data. We obtain 367 sets of data from a small sewage treatment plant in Beijing from June to August 2015. 267 sets of data are used as training samples and 110 sets of data are used as test samples. The ratio of training samples to test samples is 7:3. In this experiment, the proposed DCIA-SORBF neural network is used to predict the values of the effluent TP. And the easy-to-measure process variables include: the temperature, oxidation reduction potential, influent TP, dissolved oxygen, pH and total soluble solid, which are selected as the input variables of the DCIA-SORBF neural network.

The experimental results are shown in the Fig. 7. As can be seen from the graph, DCIA-SORBF can better predict TP value with a small prediction error and the error is between –0.015 and 0.015. However, when noise data appear in the 70th to 90th samples, the algorithm still has a certain degree of prediction distortion. The algorithm has better robustness and the prediction error is within acceptable range when noise occurs. Therefore, the algorithm can still track and predict the results of total phosphorus. In addition, the comparison results are recorded in Table V. We can see that DCIA-SORBF is compared with other six algorithms. From the results, we can see that DCIA-SORBF has a smallest mean testing RMSE and a more compact structure for TP prediction. Then, DCIA-SORBF has the shortest prediction time. The above results show that the DCIA-SORBF is more suitable and effective than the other SORBF neural networks on predicting the effluent TP values.

In order to avoid the randomness caused by the experimental results, In Table VI, at the same time, in order to judge the overall performance of the algorithm, the experimental results of five test functions are counted and sorted. The specific results such as Table VI can be seen. rank sum on mean and rank sum on Dev. are the sum of mean and Dev. ranking of all test functions for each algorithm. As you can see, for all test functions, the sum of mean and Dev. rankings of DCIA-SORBF is the smallest. In addition, sum rank on all the problems is the sum of the sorting of all test functions mean and Dev. for each algorithm. Final rank on all the problems is the final ranking result of each algorithm. The experimental results show that DCIA-SORBF has the smallest test error and the best stability, so the algorithm has the best prediction performance.

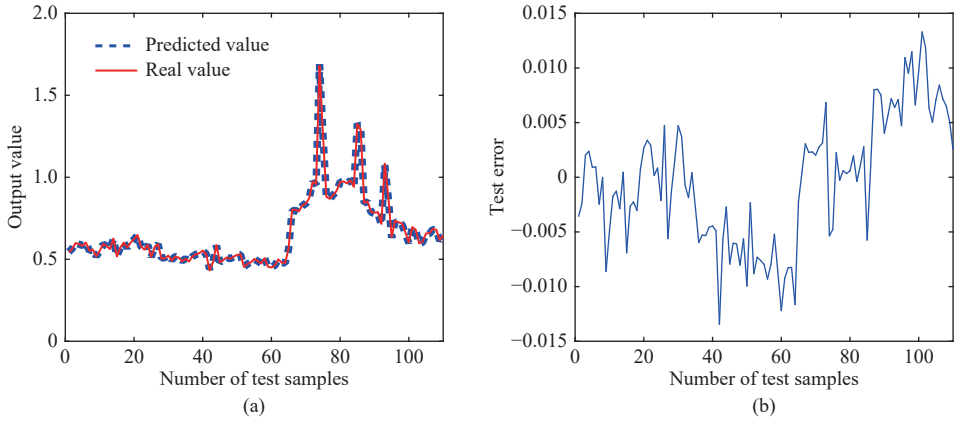


Fig. 7. Experimental results of the proposed DCIA-SORBF neural network for effluent TP prediction. (a) Prediction results for the different algorithms versus sample number. (b) Prediction error versus sample number.

TABLE V
COMPARISON OF DIFFERENT ALGORITHMS FOR EFFLUENT TP PREDICTION IN WWTP

Algorithm	Testing RMSE		No. of hidden neurons	Testing time (s)	Mean/Dev. rank
	Mean	Dev.			
DCIA-SORBF	0.0102	0.0027	10	0.0097	1/1
APSO-SORBF [21]	0.0127*	0.0025*	12*	0.0120*	2/2
AI-RBF [37]	0.0201	0.0076	12	0.0580	5/5
GAP-RBF [13]	0.0356*	0.0085*	18*	0.0630*	6/6
PSO-RBF [36]	0.1602*	0.0915*	14*	0.0055*	7/7
AI-PSO-RBF [34]	0.0191*	0.0052*	12*	0.0290*	4/4
SAIW-PSO-RBF [35]	0.0159*	0.0049*	12*	0.0410*	3/3

* The results are also listed in the original papers.

F. The Fitness

In order to verify the diversity of DCIA algorithm, Mackey-Glass time series prediction problem is used as the standard test function in this experiment. Meanwhile, the average value of fitness (test error here) and the best fitness value of all antibodies were taken as test indicators. In order to verify the effectiveness of distance concentration method with immune algorithm, this paper compares it with the self-organizing RBF neural network (IA-SORBF) based on artificial immunity. Among them, the number of iterations is 100. The experimental results are shown in Fig. 8. We can see that IA-SORBF has gradually converged in less than 10 generations, while DCIA-SORBF needs at least 20 generations. Therefore, IA-SORBF is more likely to fall into local optimum. In addition, the fitness average of all antibodies given in Fig. 8 (a) shows that the fluctuation range of DCIA-SORBF is larger than that of IA-SORBF, which also shows that the difference of antibodies in DCIA-SORBF algorithm is greater and the diversity of antibodies is better. At the same time, as shown in Fig. 8 (b), the optimal fitness value of DCIA-SORBF algorithm is smaller. Therefore, the algorithm has smaller test error and better diversity, so it can better approximate the global optimal solution.

VI. DISCUSSION

In order to verify the effectiveness of the proposed algorithm, The experimental results show that the diversity of the proposed DCIA algorithm is significantly increased compared with that of IA without distance concentration algorithm, so that the proposed DCIA algorithm can better jump out of the local optimum. Secondly, five test functions are used in this paper. From Figs. 3–7, it can be seen that DCIA-SORBF can predict the value of the objective function better, and the error is small. At the same time, the algorithm can adjust the network structure adaptively with the change of sample data, and finally make the RBF network structure the most compact. In addition, the proposed DCIA-SORBF algorithm is compared with six algorithms: APSO-SORBF [21], AI-RBF [13], GAP-RBF [33], PSO-RBF [35], AI-PSO-RBF [25], and SAIW-PSO-RBF [48]. As can be seen from Tables I–V, except Lorenz time series system, DCIA-SORBF algorithm has the smallest test error in other test functions. Therefore, it has the best prediction accuracy. Meanwhile, for the RMS error value, the RMS error of the algorithm is the smallest except for the function approximation of the test function so that the system has high stability. However, since RBF network structure can be self-organized, we can see from the table that DCIA-SORBF has the smallest network structure for function approximation, nonlinear system identification and effluent TP prediction in WWTP, which effectively avoids the redundancy of network structure and thus has the shortest computing time. However, for the more

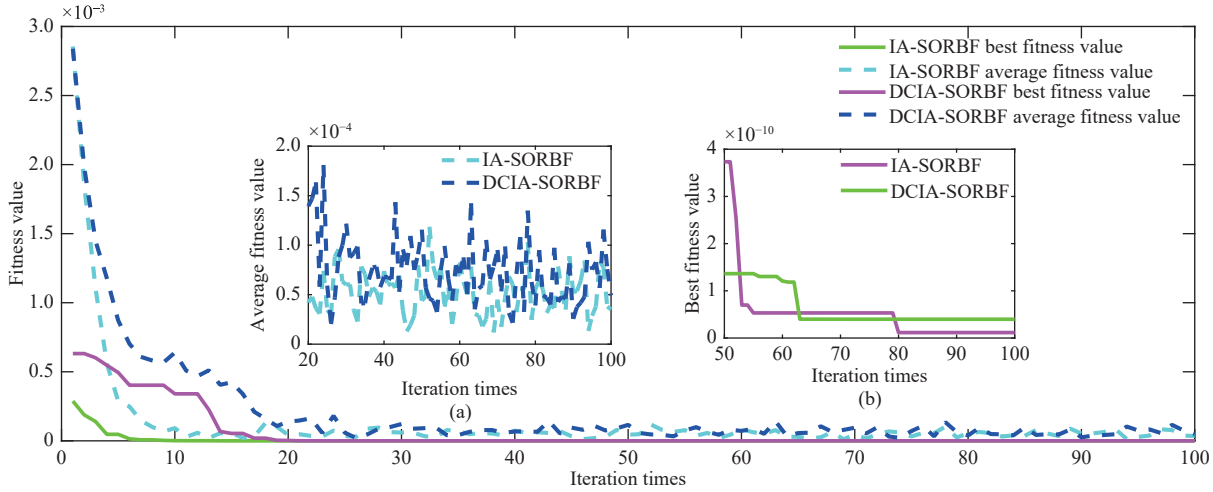


Fig. 8. The diversity comparison of DCIA-SORBF.

TABLE VI
FINAL RANK OF ALL THE ALGORITHMS ON THE ZDT, WFG, AND DTLZ PROBLEMS

Problems	Algorithms						
	DCIA-SORBF	APSO-SORBF	AI-RBF	GAP-RBF	PSO-RBF	AI-PSO-RBF	SAIW-PSO-RBF
Function approximation	1/2	2/3	4/5	8/4	7/6	6/7	3/1
Mackey-Glass timeseries prediction	1/1	2/2	3/3	7/—	6/6	5/4	4/5
Nonlinear system identification	1/1	2/4	4/7	6/6	7/5	5/2	3/3
Lorenz time series system	1/1	3/3	2/2	7/—	6/6	5/4	4/5
Effluent TP prediction in WWTP	1/2	2/2	5/5	6/6	7/7	4/4	3/3
Rank sum on mean	5	11	17	34	33	25	17
Rank sum on Dev.	7	14	22	—	30	21	17
Sum rank on all the problems	11	25	40	—	63	46	34
Final rank on all the problems	1	2	4	—	6	5	3

complex Lorenz time series system, a larger network structure is needed to improve the prediction accuracy, and the experimental results show that DCIA-SORBF error is the smallest except AI-RBF. Compared with other algorithms, it has the smallest root mean square error. For Mackey-Glass time series prediction function, the network structure of the algorithm is slightly larger than others, but the error and root mean square error are the smallest. Finally, in order to calculate the performance of the proposed DCIA-SORBF algorithm, Wilcoxon' rank is used to analyze the experimental results. From Table VI, through five test functions, the algorithm has the smallest error and root mean square error compared with other algorithms, which proves that DCIA-SORBF algorithm has the highest prediction accuracy and better stability. At the same time, the statistical results of final rank on all the problems show that the algorithm has the best performance.

CONCLUSION

In this paper, a SORBF neural network is presented to model uncertain nonlinear systems, and the network size and parameters are simultaneously optimized by the proposed DCIA algorithm. In addition, to overcome the shortcoming of easily falling into a local optimum of other algorithms, the distance concentration algorithm that can increase the diversity of immune cells, is adopted. However, while adjusting the structure of the RBFNN, ensuring the stability of the network is quite challenging. For this purpose, the information-oriented algorithm (IOA) is applied to identify which antibodies need to be updated. To increase the network prediction accuracy and reduce the calculation tasks, the sample with the most frequent occurrence of error is used to regulate the parameters of the new neuron. Additionally, the convergence of DCIA-SORBF is demonstrated theoretically for a practical application. Finally, the experimental results demonstrate that the proposed DCIA-SORBF algorithm is more effective in solving nonlinear learning problems. Moreover, the good potential of the proposed techniques in real-world applications is demonstrated from our simulation results over several benchmark problems and an engineering modeling task.

In addition, the parameters have a great correlation with the predictive results. In the future research work, we will choose the best parameters adaptively according to different situations. At the same time, when there is some noise or interference data, how to get accurate prediction results is another direction and focus of our next research.

REFERENCES

[1] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function network," *Neural Computation*, vol. 3, no. 2, pp. 246–257, Jul. 1991.

[2] Y. W. Wong, K. P. Seng, and L. M. Ang, "Radial basis function neural network with incremental learning for face recognition," *IEEE Trans. Systems*, *Man*, *and Cybernetics*, *Part B* (*Cybernetics*), vol. 41, no. 4, pp. 940–949, Aug. 2011.

[3] H. G. Han, X. L. Wu, and J. F. Qiao, "Real-time model predictive control using a self-organizing neural network," *IEEE Trans. Neural Network and Learning Systems*, vol. 24, no. 9, pp. 1425–1436, Sep. 2013.

[4] V. Tomenko, "Online dimensionality reduction using competitive learning and radial basis function network," *Neural Networks*, vol. 24, no. 5, pp. 501–511, Jun. 2011.

[5] A. Alexandridis, E. Chondrodima, and H. Sarimveis, "Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization," *IEEE Trans. Neural Networks and Learning Systems*, vol. 24, no. 2, pp. 219–230, Feb. 2013.

[6] R. Zhang, Z. B. Xu, G. B. Huang, and D. Wang, "Global convergence of online BP training with dynamic learning rate," *IEEE Trans. Neural Networks and Learning Systems*, vol. 23, no. 2, pp. 330–341, Jan. 2012.

[7] Z. B. Xu, R. Zhang, and W. F. Jing, "When does online BP training converge?" *IEEE Trans. Neural Networks*, vol. 20, no. 10, pp. 1529–1539, Aug. 2009.

[8] B. D. Chen, S. L. Zhao, P. P. Zhu, and J. C. Principe, "Quantized kernel recursive least squares algorithm," *IEEE Trans. Neural Networks and Learning Systems*, vol. 24, no. 9, pp. 1484–1491, May 2013.

[9] M. S. Al-Batah, N. A. M. Isa, K. Z. Zamli, and K. A. Azizli, "Modified recursive least squares algorithm to train the hybrid multilayered perceptron (HMLP) network," *Applied Soft Computing*, vol. 10, no. 1, pp. 236–244, Jan. 2010.

[10] J. Jiang and Y. M. Zhang, "A novel variable-length sliding window blockwise least-squares algorithm for on-line estimation of time-varying parameters," *Int. J. Adaptive Control and Signal Processing*, vol. 18, no. 6, pp. 505–521, Aug. 2004.

[11] J. X. Peng, K. Li, and G. W. Irwin, "A novel continuous forward algorithm for rbf neural modelling," *IEEE Trans. Automatic Control*, vol. 52, no. 1, pp. 117–122, Jun. 2007.

[12] J. F. Qiao and H. G. Han, "Identification and modeling of nonlinear dynamical systems using a novel self-organizing RBF-based approach," *Automatica*, vol. 48, no. 8, pp. 1729–1734, Aug. 2012.

[13] G. B. Huang, P. Saratchandran, and N. Sundararajan, "An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks," *IEEE Trans. Systems*, *Man*, *and Cybernetics*, *Part B* (*Cybernetics*), vol. 34, no. 6, pp. 2284–2292, Dec. 2004.

[14] G. B. Huang, P. Saratchandran, and N. Sundararajan, "A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation," *IEEE Trans. Neural Network*, vol. 16, no. 1, pp. 57–67, Jan. 2005.

[15] N. Vuković and Z. Miljković, "A growing and pruning sequential learning algorithm of hyper basis function neural network for function approximation," *Neural Network*, vol. 46, no. 1, pp. 210–226, Jan. 2013.

[16] H. G. Han, Y. N. Guo, and J. F. Qiao, "Self-organization of a recurrent RBF neural network using an information-oriented algorithm," *Neurocomputing*, vol. 225, pp. 80–91, Nov. 2016.

[17] J. Lian, Y. Lee, S. D. Sudhoff, and S. H. Zak, "Self-organizing radial basis function network for real-time approximation of continuous-time dynamical systems," *IEEE Trans. Neural Networks*, vol. 19, no. 3, pp. 460–474, Mar. 2008.

[18] R. V. Babu, S. Suresh, and A. Perkis, "No-reference JPEG-image quality assessment using GAP-RBF," *Signal Processing*, vol. 87, no. 6, pp. 1493–1503, Jun. 2007.

[19] H. M. Feng, "Self-generation RBFNs using evolutional PSO learning," *Neurocomputing*, vol. 70, no. 1, pp. 241–251, Dec. 2006.

[20] A. Alexandridis, E. Chondrodima, and H. Sarimveis, "Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization," *IEEE Trans. Neural Network and Learning Systems*, vol. 24, no. 2, pp. 219–230, Dec. 2012.

[21] H. G. Han, W. Lu, Y. Hou, and J. F. Qiao, "An adaptive-PSO-based self-organizing RBF neural network," *IEEE Trans. Neural Network and Learning Systems*, vol. 29, no. 1, pp. 104–117, Jan. 2018.

[22] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2–3, pp. 235–306, Jun. 2002.

[23] Y. Zhong, L. Zhang, B. Huang, and P. X. Li, "An unsupervised artificial immune classifier for multi/hyperspectral remote sensing imagery," *IEEE Trans. Geoscience and Remote Sensing*, vol. 44, no. 2, pp. 420–431, Jan. 2006.

[24] M. J. Er, S. Wu, J. Lu, and H. L. Toh, "Face recognition with radial basis function (RBF) neural networks," *IEEE Trans. Neural Networks*, vol. 13, no. 3, pp. 697–710, Feb. 2002.

[25] T. Liu, Y. C. Wang, Z. J. Wang, and J. Meng, "Distance concentration-based artificial immune algorithm," *J. China University of Mining and Technology*, vol. 15, no. 2, pp. 81–85, Jun. 2005.

[26] J. S. Chun, M. K. Kim, H. K. Jung, and S. K. Hong, "Shape optimization of electromagnetic devices using immune algorithm," *IEEE Trans. Magnetics*, vol. 33, no. 2, pp. 1876–1879, Mar. 1876.

[27] R. R. Zheng, Z. Y. Mao, and X. X. Luo, "Analysis and research of improved artificial immune algorithm," *Computer Engineering and Applications*, vol. 39, no. 34, pp. 35–37, Apr. 2003.

[28] Q. Z. Lin, J. Y. Chen, Z. H. Zhan, W. N. Chen, C. A. C. Coello, Y. L. Yin, C. M. Lin, and J. Zhang, "A hybrid evolutionary immune algorithm for multiobjective optimization problems," *IEEE Trans. Evolutionary Computation*, vol. 20, no. 5, pp. 711–729, Oct. 2016.

[29] G. Rudolph, "Convergence analysis of canonical genetic algorithms," *IEEE Trans. Neural Network*, vol. 5, no. 1, pp. 96–101, 1994.

[30] M. Han, J. Fan, and J. Wang, "A dynamic feedforward neural network based on Gaussian particle swarm optimization and its application for predictive control," *IEEE Trans. Neural Networks*, vol. 22, no. 9, pp. 1457–1468, Sep. 2011.

[31] H. G. Han, S. Zhang, and J. F. Qiao, "An adaptive growing and pruning algorithm for designing recurrent neural network," *Neurocomputing*, vol. 242, pp. 51–62, Jun. 2017.

[32] T. T. Xie, H. Yu, J. Hewlett, P. Rozycki, and B. Wilamowshi, "Fast and efficient second-order method for training radial basis function network," *IEEE Trans. Neural Network and Learning Systems*, vol. 23, no. 4, pp. 609–619, Apr. 2012.

[33] H. G. Han, W. D. Zhou, J. F. Qiao, and G. Feng, "A direct self-constructing neural controller design for a class of nonlinear systems," *IEEE Trans. Neural Network and Learning Systems*, vol. 26, no. 6, pp. 1312–1322, Jun. 2015.

[34] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Applied Soft Computing*, vol. 11, no. 4, pp. 3658–3670, Jun. 2011.

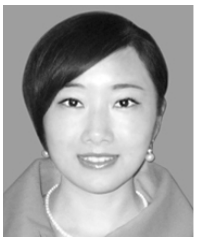
[35] M. Taherkhani and R. Safabakhsh, "A novel stability-based adaptive inertia weight for particle swarm optimization," *Applied Soft Computing*, vol. 38, pp. 281–295, Jan. 2016.

[36] H. M. Feng, "Self-generation RBFNs using evolutional PSO learning," *Neurocomputing*, vol. 70, no. 1–3, pp. 241–251, Dec. 2006.

[37] F. Li, C. L. Yang, and J. F. Qiao, "A novel RBF neural network design based on immune algorithm system", in *Proc. 36th Chinese Control Conf.* (*CCC*), pp. 4598-4603, Jul. 2017.

Junfei Qiao (M'11) received the B.E. and M.E. degrees in control engineering from Liaoning Technical University, Fuxin, China, in 1992 and 1995, respectively, and the Ph.D. degree from Northeast University, Shenyang, China, in 1998. From 1998 to 2000, he was a Post-Doctoral Fellow with the School of Automatics, Tianjin University, Tianjin, China. He is currently a Vice-President and Professor with the Beijing University of Technology, Beijing, China. His current research interests include neural networks, intelligent systems, self-adaptive/learning systems, and process control systems.

Dr. Qiao is a member of the IEEE Computational Intelligence Society. He is currently a reviewer for over 20 international journals, such as the *IEEE Transactions on Fuzzy Systems*, *IEEE Transactions on Neural Networks and Learning Systems*.



Fei Li is currently working toward the Ph.D. degree at Beijing University of Technology, Beijing, China. Her research interests include in intelligent control, multi-objective optimization algorithm and analysis and design of neural networks.

Cuili Yang received the B.E. degree from China University of Petroleum, Dongying, China, in 2008, the M.S. degree from Tianjin University, Tianjin, China in 2010, and the Ph.D. degree from City University of Hong Kong, Hong Kong, China, in 2014. She is currently a Lecturer at Beijing University of Technology. Her current research interests include computational intelligence, modeling and control for wastewater treatment process.

Wenjing Li received her bachelor degree at Xi'an Jiaotong University in 2007, and Ph.D. degree at the Institute of Automation, Chinese Academy of Sciences in 2013. She is now an Associate Professor in Beijing University of Technology. Her research interests include cognitive neuroscience, pattern recognition, and intelligent system.

Ke Gu received the B.S. and Ph.D. degrees in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2009 and 2015, respectively. He is currently a Professor with Beijing University of Technology, Beijing, China. His research interests include environmental perception, image processing, quality assessment, and machine learning. He received the Best Paper Award from the IEEE Transactions on Multimedia, the Best Student Paper Award at the IEEE International Conference on Multimedia and Expo in 2016, and the Excellent Ph.D. Thesis Award from the Chinese Institute of Electronics in 2016. He was the Leading Special Session Organizer in the VCIP 2016 and the ICIP 2017, and serves as a Guest Editor for the *Digital Signal Processing Journal*. He is currently an Associate Editor of the *IEEE Access and the IET Image Processing*. He is a Reviewer for 20 top SCI journals.