Adaptive Spread Coefficient-based RBF-NN for Complex Signals Modeling

Yibin Song¹, Zhenbin Du²
School of Computer Science, Yantai University, Shandong, China
¹syb2874@163.com; ²zhenbindu@163.com

Abstract
As an efficient method on the fitting or approximating for complex signals, the Radial Base Function Neural Network (RBF-NN) is widely used in signal modeling. During the training process, the spread coefficient (Sc) is one of important parameters in the RBF-NN learning algorithm. A suitable Sc can speed up the signal fitting process. This paper presents an improved RBF-NN learning method based on the adaptive spread coefficient for the signal approximation of complex systems. The improved algorithm is applied to the learning and approximating process of the nonlinear signal. The simulations showed that the presented RBF-NN has good effects on speeding up the training and approaching process. Meanwhile, the learning convergence of the improved algorithm is more excellent than that of normal algorithm.

Keywords
RBF Neural Networks, Adaptive Spread Coefficient, Complex Signals Modelling, Learning Convergence.

Introduction
RBF Neural Network (RBF-NN) is widely used on the fitting or approximating for the complex signals or the model recognition of the nonlinear systems. In RBF - NN, the Radial Base Function is adopted as mapping function. The mapping relation between different layers will be confirmed as soon as the centre point being confirmed. But the linear mapping is applied between hidden layer and output layer, and the weighted linearly sum of hidden layers is used for the output of Neural Network (NN). The linking weights are adjustable parameters of Network. So, the linking weights of network can be evaluated by linear equation sets or by RLS method directly. The learning speed of NN will be more quickly. Meanwhile, the fussy calculation and the local minimizing can be avoided.

The RBF-NN is a typical local approaching network. Only a few weights need to be adjusted for the output of network. Therefore, the RBF-NN could complete the learning process more quickly than that of BP-NN.

Based on the principle of RBF-NN, this paper presents an improved RBF-NN for the signal approximating, which is based on the adaptive spread coefficient. It is also suitable for the identification of the structure-unknown complex signals. An actual signal approximating of a nonlinear model is applied to validate the effects of the improved algorithm. The simulations showed that the presented method has good effects on speeding up the learning process and approximating performance, especially suitable for the real-time request of complex signals modeling.

Improved Algorithm for RBF-NN

RBF-NN Structure Analysis
For the RBF-based Neural Network architecture, direct mapping was adopted between input and hidden layer. But the mapping between hidden layer and output layer adopts the weighted linearly sum of hidden layers as the mapping mode (shown as Fig. 1).

![FIG. 1: NORMAL STRUCTURE OF RBF-NN](image)

This structure of NN can reduce the complexity of computational problems so as to speed up the learning process. It is suitable especially for accomplishing the function approaching, model identification and sorting process quickly.

Adaptive-Parameter-Based algorithm for RBF-NN
In RBF-NN, the input vectors of lower dimensions are mapped to the hidden space of higher dimensions at first. The hidden cells selected the base function to realize the vector conversion, and then, sorted or identified by output layer. The form of base function F
can be described as:

\[
f(X) = \Lambda_0 + \sum_{j=1}^{M} \Lambda_j \Phi \left\| X - C_j \right\|
\]

(1)

Where: \( \Phi \left\| X - X_i \right\| \) is the Radial Base Function (RBF).
\( \left\| \cdot \right\| \) denotes Euclid Norm.

\( X \in \mathbb{R}^n, \ \Lambda_j \in \mathbb{R}^m \) (j=1,2,…… M)

\( C_j \in \mathbb{R}^n \) is the center of RBF.

\( \Lambda_0 \in \mathbb{R}^m \) is the constant vector.

Suppose \( \Lambda = [\lambda_{mj}]^T \) (j=1,2,…… M), the equation (1) can be re-described as:

\[
f_j(X) = \lambda_{0j} + \sum_{j=1}^{M} \lambda_{mj} \Phi \left\| X - C_j \right\|
\]

(2)

If Gauss function is adopted as the Radial Base Function (RBF), it can be presented as:

\[
\Phi = G \left\| X - C_j \right\|^2 = \exp \left( -\frac{1}{d_m} \left\| X - C_j \right\|^2 \right)
\]

(3)

Where, \( d_m \) is the longest distance between the selected centres.

According to the different ways for selection of RBF center, RBF-NN is usually applied the methods as random selection, Self-Organization Learning (SOL) or supervisory learning, etc. This paper applied self-organization learning algorithm to choose the center of RBF so as to complete the network learning. In SOL algorithm, the central position of RBF is set automatically. The weights of output layer can be calculated by error-correction-learning algorithm. So, it is known that the SOL algorithm is a mixed learning algorithm essentially. The function of SOL is to adjust the center of RBF to the key area of input space.

To cluster, least distance is the learning goal of SOL algorithm. The k-mean clustering algorithm was adopted. The input samples were decomposed to M classes and M clustering centres were obtained. The steps of clustering algorithm are:

Step 1: To choose randomly M samples, from the input samples \( X_j (j=1,2,\ldots,N) \), as the initial clustering centres \( C_i (i=1,2,\ldots,M) \).

Step 2: To distribute input samples \( X_j (j=1,2,\ldots,N) \) to every \( C_i \) and to form clustering sets \( \theta_i (i=1,2,\ldots,M) \).

The following condition should be met:

\[
d_i = \min_{j} \left\| X_j - C_i \right\|
\]

(4)

\( (j=1,2,\ldots,N \ i=1,2,\ldots,M) \)

Where \( d_i \) is the minimal Euclidean distance.

Step 3: To calculate the sample's mean (i.e. clustering center \( c_i \)) in \( \theta_i \):

\[
C_i = \sum_{j=1}^{M} \frac{X_j}{M}
\]

(5)

Where \( M \) is the number of input sample in \( \theta_i \)

Step 4: To repeat above calculation till the change of distribution of clustering centre less than designed value \( \varepsilon \) is obtained.

Step 5: After the decision of RBF centre, the mean square error \( \sigma = \frac{d_m}{\sqrt{2M}} \) can be calculated and then the output of hidden layer is acquired by equation (2).

The Error-Correction-Learning algorithm can calculate the linear weights between hidden and output layer. Suppose that the output of the kth neuron of output layer is \( \hat{y}_k \) and that of the jth neuron of hidden layer is \( g_j \). The relation is:

\[
\hat{y}_k = \sum_j w_{kj} \cdot g_j
\]

(6)

If the actual output is \( y_k \), the error is:

\[
e_k = y_k - \hat{y}_k
\]

The purpose of BP learning is to amend linking weights so as to minimize the error index \( E = f(\sum e_k^2) \) and to meet the desired performance \( J \).

When input mode is \( X \), therefore, the correcting value of \( w_{kj} \) should be:

\[
\Delta w_{kj} = -\alpha \frac{\partial E}{\partial w_{kj}}
\]

(7)

Where, \( \alpha \) is adjusting factor for learning rate.

The correcting equation is:

\[
w_{kj}(k+1) = w_{kj}(k) + \Delta w_{kj}
\]

(8)

In order to improve the convergence of algorithm and its learning effects, a dynamic correcting factor can be set in the correcting equation, that is:

\[
w(k+1) = w(k) + \alpha(1-\eta)T(k) + \eta T(k-1)
\]

(9)

where, \( 0<\alpha<1, \ 0 \leq \eta < 1 \) is a dynamic factor and \( T(k) = -\frac{\partial E}{\partial w(k)} \) is the direction of negative grads in the kth time learning.

However, simulations showed that this method is not perfect to improve the learning process. The key question is how to select the learning rate. Sometimes, it is a very difficult decision for the algorithm. The difficulty is how to give attention to both the learning speed and the convergence of algorithm. If a rate-adaptive factor \( \delta(k) \) is used in the learning algorithm,
the learning performance can be improved significantly. The improved equation is described as equation (10). Where, \( \lambda = \text{Sign} [T(k)T(k-1)] \) is the direction of grads, \( \delta(k) \) is the adaptive factor associated with learning error \( \varepsilon \).

\[
\begin{align*}
    w(k+1) &= w(k) + \alpha(k)T(k) \\
    \alpha(k) &= \delta(k)\alpha(k-1) \\
    \delta(k) &= \varepsilon \cdot 2^k
\end{align*}
\] (10)

The spread coefficient (Sc) is one of important parameters in the RBF-NN learning algorithm. The different decision of Sc will affect the learning performance directly. In fact, a suitable Sc can speed up the signal fitting process obviously. This paper presented an improved RBF learning method based on the adaptive spread coefficient for the signal approximation of complex models. The adaptive spread coefficient is described as:

\[
Sc = Sc + \eta \cdot (e^\zeta - 1)
\] (11)

Where, \( \zeta = \max[err] - err^* \),
\( \eta \) is fitting factor,
\( err^* \leq 0.0001 \) is the desired learning error.

In order to get a suitable spread factor, the relative error of learning \( \zeta \) has been introduced into the training algorithm. The value of Sc can be modified adaptively with the training errors. Simulations showed that the amended algorithm can improve the training performance and simplify the structure of neural network satisfactorily.

**Simulations**

According to the characteristic of RBF-NN, RBF-NN was employed to identify the complex model via learning about the sampled I/O data. For the structure-unknown system, RBF-NN needs to model the unknown object and identify the behaviours of the system by learning the sampling data.

Suppose that the complex model is described as the following equations:

Input:

\[
u(k) = 0.25 \cdot [u^2(k - 1) - \sin^3(\frac{3k\pi}{97})]
\] (12)

Output of the model:

\[
y(k) = \frac{[y^2(k - 1) + 1] \cdot y(k - 2) \cdot u^3(k - 1) + \exp[u^2(k - 2)]}{1 + y^2(k - 1)}
\] (13)

As an example, 200 pairs and 400 pairs of input data \( u(k) \) and output data \( y(k) \) were sampled respectively as the learning samples. The training results and the learning errors are shown in Fig. 2 – Fig. 5.

The simulations showed that the RBF-NN could accomplish the desired signal fitting and model identifying satisfactorily. The desired learning aim had been achieved by the improved RBF-NN with the adaptive Spread coefficient (Sc) only after 155-epoch trainings. It is more speedy and accurate than normal RBF-NN, which needs the trainings for 197 epochs. On the other hand, it means that there are only 155 hidden neurons needed to construct the improved RBF-NN. But the normal RBF-NN needs 197 hidden neurons to get the same training aim. Detail comparing results are shown in table 1.

Simulations showed clearly that the learning effect of improved RBF-NN is much significantly than that of normal RBF-NN, on not only the training speed but also the network structure.

It can be found that the improved RBF-NN is much suitable for the modelling and signal fitting of complex plants or systems. It is also an effective method for the prediction and control of complex systems.

**Conclusions**

The outstanding feature of RBF-based Neural
Network are its accurate learning results and quick convergence. This paper has presented an improved algorithm for the RBF-NN. After applying to the identification of a typical nonlinear system and simulations, some conclusions can be made.

Although the RBF-NN can take effect on speeding up the learning process, the improved RBF-NN with adaptive $\varepsilon$ can has quicker training speed, more simplified network structure and more excellent learning performance. It can improve the modelling and identifying process of complex plants obviously. Therefore, the presented RBF-NN is an effective way for the realization of the real-time control of complex systems.

![Graph showing learning error during the trained process of 215 epochs](image1)

**FIG. 4: LEARNING ERRORS OF OF FIXED $\varepsilon$-BASED RBF-NN**

![Graph showing learning error during the trained process of 148 epochs](image2)

**FIG. 5: LEARNING ERRORS OF ADAPTIVE $\varepsilon$-BASED RBF-NN**

<table>
<thead>
<tr>
<th>Training algorithm</th>
<th>Pair of samples</th>
<th>Training epochs</th>
<th>Desired variable of learning error</th>
<th>Mean-Squared Error of NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBF-NN</td>
<td>200</td>
<td>197</td>
<td>0.0001</td>
<td>4.32e-3</td>
</tr>
<tr>
<td>Improved RBF-NN</td>
<td>200</td>
<td>155</td>
<td>0.0001</td>
<td>4.76e-3</td>
</tr>
<tr>
<td>RBF-NN</td>
<td>400</td>
<td>215</td>
<td>0.0001</td>
<td>2.04e-3</td>
</tr>
<tr>
<td>Improved RBE-NN</td>
<td>400</td>
<td>148</td>
<td>0.0001</td>
<td>2.85e-3</td>
</tr>
</tbody>
</table>

**ACKNOWLEDGMENT**

This work was sponsored by National Natural Science Foundation of China (61175053). The first author is the primary contact.

**REFERENCES**


**Yibin Song.** was born in Anhui, China. Oct. 01, 1957. Graduated in Hefei University of Technology, China, and earned Bachelor’s degree in Hefei University of Technology (1982), majoring in industrial automation; Master’s Degree in industrial automation in Hefei University of Technology (1988), China.

Up to now, he has worked in the university for 30 years. His research interests include artificial intelligence, machine learning, complex system modeling, etc and has published about 40 academic papers.

At present, Prof. Song is the member of IEEE SMC Society, the Chinese automation society, the Chinese computer society.

**Zhenbin Du.** PHD, associate professor. Majoring in automation. The member of Chinese automation society. His research interests include intelligent control methods, nonlinear system modeling, etc.