Proceedings of the 5th International Conference on Advances in Power System Control, Operation and Management, APSCOM 2000, Hong Kong, October 2000.

A NOVEL RADIAL BASIS FUNCTION NEURAL NETWORK FOR FAULT SECTION ESTIMATION IN TRANSMISSION NETWORK

T. S. Bi Y. X. Ni C. M. Shen F. F. Wu *Q. X. Yang

Dept. of Electrical & Electronic Engineering
The University of Hong Kong
Hong Kong, SAR
Email: yxni@eee.hku.hk

*Dept. of Electrical Engineering
North China Electric Power University
Beijing, China

ABSTRACT

In this paper, the application of Radial Basis Function Neural Network (RBF NN) to fault section estimation in power systems is addressed. The orthogonal least square algorithm has been extended to optimize the parameters of RBF NN. In order to assess the effectiveness of RBF NN, a classical Back-Propagation Neural Network (BP NN) has been developed to solve the same problem for comparison. Computer test is conducted on a 4-bus test system and the test results show that the RBF NN is quite effective and superior to BP NN in fault section estimation.

1. INTRODUCTION

To enhance service reliability and to reduce power supply interruption, rapid restoration of power system is essential. As the first step to system restoration, fault section estimation should be implemented quickly and accurately in order to isolate the faulted elements from the rest of the system and to take proper countermeasures to recover normal power supply. However, fault section estimation is difficult, especially for the cases with failure operations of relays and circuit breakers, or multiple faults at the same time. Therefore on-line automatic fault section estimation is significant to the restorative operations.

Several approaches have been investigated such as expert-system-based [1], optimization-based [2] and artificial-neural-network-based [3-6] approaches. Among them, research endeavors have been directed to the Artificial Neural Network (ANN), because it has the capabilities of learning, generalization and fault tolerance. In addition, the neuron computations are parallel which make it suitable for on-line environment. Among all the applications, the most widely used model is Back-Propagation Neural Network (BP NN)[3-5]. As we known, the BP NN structure has to be priori known and the algorithm might converge very slowly and reach a local minimum. These disadvantages limit the applications of BP NN in fault section estimation.

In this paper the Radial Basis Function Neural Network (RBF NN) is proposed for fault section estimation in power systems. The RBF NN [6, 7] can be designed in a fraction of the time as compared with other design approaches in training standard feed-forward networks. Theoretically with enough RBF neurons, the RBF NN can realize zero error to all the training samples. Besides the number of RBF neurons in the hidden layer can be determined during the parameter optimization process. These features make it very attractive in practical use. The orthogonal lest square algorithm [8] has been extended to optimize the parameters of the RBF NN. In order to assess the effectiveness of RBF NN, a classical BP NN has been developed to solve the same problem for comparison. Computer test is conducted on a 4-bus test power system [5] and the results show that the RBF NN is quite effective and superior to the BP NN in fault section estimation.

2. RBF NEURAL NETWORK FOR FAULT SECTION ESTIMATION

2.1 The structure of RBF NN for fault section estimation

The suggested RBF NN has input, hidden and output layers with the hidden layer composed of RBF neurons (See Fig. 1). The input space can be either an actual or a normalized representation. The input signals are forward to the hidden layer, i. e. RBF neuron (RBFN) layer. The i^{th} neuron in the RBFN layer examines the distance between the input vector x and its weight vector u_i and evaluate the radial basis function (RBF) $\varphi_i(x)$ to yield the output. The investigation [8] shows that the choice of the RBF is not crucial to the performance of the RBF network. In our study we take the Gaussian function as the RBF:

$$\varphi_i(x) = exp\left(-[x-u_i]^T[x-u_i]/2\sigma_i^2\right) \quad (i=1,\dots,n_h) \quad (1)$$

where $\varphi_i(x)$ is the output of the i^{th} neuron of the hidden layer; n_h is the number of the hidden neurons; x is the input vector; u_i and σ_i are the center (or weight, a vector with same dimension as x) and the spread (a scalar) of the i^{th} Gaussian function.

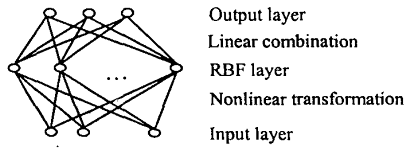

Fig.1 A schematic diagram of a RBF neural network

The spread σ_i (>0) represents the range of $\|x-u_i\|$ in the input space to which the RBF neuron should respond. Usually the spread should be no more than the possible maximum distance between the input vector and the center of the RBF and is determined based on experiments.

The output layer generates the desired output through linear mapping of the output of the RBF hidden layer. The output of the neuron j in the output layer will be:

$$o_j = \sum_{i=1}^{n_h} v_{ij} \cdot \varphi_i(x) \quad (j = 1,\ldots,n_o) \qquad (2)$$

where n_o is the total number of the output neurons. v_{ij} is the weight or the contribution factor of the i th hidden neuron to the j th output neuron.

It is clear that to fast and effectively determine the centers $\{u_i\}$ of the RBF and the weights $\{v_{ij}\}$ of the output layer based on the given training samples are the key tasks in RBF NN design. Indeed once the RBF centers $\{u_i\}$ have been fixed, the output weights $\{v_{ij}\}$ can be solved for based on (2) without difficulty since $\{\varphi_i(x)\}_{i=1}^{n_h}$ and the corresponding desired output $\{o_j\}_{j=1}^{n_o}$ are known for all the training samples. Therefore the key question in building up the RBF NN is how to work out RBF centers appropriately based on the training sets, which is addressed in the following section.

2.2 The training algorithms of RBF NN

Assume we have n_i and n_o neurons in the input and output layer respectively. We also assume there are N sets of training samples, i.e. $x(t) \in R^{n_i}, t = 1,\ldots,N$. To determine the number of neurons in the hidden layer n_h and the corresponding centers for hidden neurons, the simplest method is the Exact Design method [7]. The method can generate a RBF NN with its outputs exactly equal to the desired outputs and free of errors when it is fed with an input vector x existing in the training samples. In this method we create as many hidden neurons as the number of training samples N, and set the hidden neuron weights to $u_i = x(i), i = 1,\ldots,N$. It is clear that according to (1) and (2) for the input $x(i) = u_i$, the corresponding output equal to the desired output. However if the number of training samples N is large, the number of hidden neurons will be too large to be acceptable. In order to solve the problem, the Orthogonal Least Squares (OLS) algorithm suggested in [8] will be extended in this paper to optimize the number of hidden neurons, the centers of the RBFs and the weights of the output layer.

In the OLS approach, the two mappings in the RBF NN can be denoted in a matrix form:

$$D = \Phi \cdot V + E \qquad (3)$$

where matrix Φ corresponds to the first mapping and is called as regression matrix. It takes the form: $\Phi(N \times n_h) = [\varphi_1 \ldots \varphi_l \ldots \varphi_{nh}] = [\varphi(1) \ldots \varphi(2) \ldots \varphi(N)]^T$.

Its l th column and t th row element $\varphi_l(x(t))$ is the output of the l th hidden neuron corresponding to the input vector of $x(t)$. Matrix $V_{(n_h \times n_o)}$ is the weight matrix defined in (2) corresponding to the second mapping (see Fig. 1). Matrix $D(N \times no) = [d_1 \ldots d_m \ldots d_{no}] = [d(1) \ldots d(t) \ldots d(N)]^T$, with similar structure as matrix Φ, is the desired output for all the training samples. The error matrix $E_{(N \times n_o)} = [\varepsilon_1 \ldots \varepsilon_m \ldots \varepsilon_{n_o}]$ represents the deviation of RBF NN output from the desired output D. E is assumed to be uncorrelated with the regressors Φ and should be as small as possible after training of the RBF NN.

In the OLS method the regression matrix Φ is decomposed into:

$$\Phi = W \cdot A \qquad (4)$$

where A is an $n_h \times n_h$ upper triangular matrix:

$$A = \begin{bmatrix} 1 & \alpha_{12} & \alpha_{13} & \cdots & \alpha_{1n_h} \\ 0 & 1 & \alpha_{23} & \cdots & \alpha_{2n_h} \\ 0 & 0 & \cdots & & \cdot \\ \cdot & \cdot & & & \cdot \\ & & & 1 & \alpha_{n_h-1,n_h} \\ 0 & 0 & & 0 & 1 \end{bmatrix} \qquad (5)$$

while $W_{(N \times n_h)} = [w_1 \ldots w_l \ldots w_{n_h}]$ is an orthogonal matrix, i. e.

$$W^T \cdot W = H \qquad (6)$$

or

$$\begin{cases} w_l^T \cdot w_l = \sum_{t=1}^{N} w_l(t) \cdot w_l(t) = h_l & 1 \le l \le n_h \\ w_l^T \cdot w_j = 0, \quad (l \ne j) \end{cases} \qquad (7)$$

where $H_{(n_h \times n_h)}$ is a diagonal matrix and h_l is its l^{th} diagonal element. Substitute (4) into (3) and define:

$$A \cdot V = G_{(n_h \times n_o)} \qquad (8)$$

According to the property of the orthogonal matrix W, the ideal orthogonal least square solution for G is given as:

$$\hat{G} = H^{-1} \cdot W^T \cdot D \qquad (9)$$

with its element calculated by

$$\hat{g}_{lm} = w_l^T \cdot d_m / \left(w_l^T \cdot w_l\right), \quad 1 \le l \le n_h, 1 \le m \le n_o \qquad (10)$$

Then (3) can be rewritten as (with '^' on G omitted):

$$D = W \cdot G + E \qquad (11)$$

or in the vector form:

$$d_m = W \cdot g_m + \varepsilon_m, \quad 1 \le m \le n_o \qquad (12)$$

Since w_l and w_p are orthogonal to each other for $l \neq p$ and ε_m is uncorrelated to w_l, we can define an energy function ($d_m{}^T \cdot d_m$) for the m th output neuron and have:

$$d_m{}^T \cdot d_m = \sum_{l=1}^{n_h} g_{lm}^2 w_l^T w_l + \varepsilon_m^T \cdot \varepsilon_m \qquad (13)$$

Then the mean total output energy for all training samples will be:

$$N^{-1} \cdot \sum_{m=1}^{n_o} \left(d_m^T d_m\right) = N^{-1} \sum_{m=1}^{n_o} \left(\sum_{l=1}^{n_h} g_{lm}^2 w_l^T w_l + \varepsilon_m^T \cdot \varepsilon_m \right) \quad (14)$$

It is clear that $N^{-1} \sum_{m=1}^{n_o} \sum_{l=1}^{n_h} g_{lm}^2 w_l^T w_l$ dominants the right hand side of (14). A relative *error reduction ratio* $[err\_red]_l$ related to the l^{th} regressor w_l can be defined as:

$$[err\_red]_l = \sum_{m=1}^{n_o} g_{lm}^2 w_l^T w_l \Big/ \left(\sum_{m=1}^{n_o} \left(d_m^T d_m\right) \right) \quad 1 \le l \le n_h \ (15)$$

This ratio offers an effective index for seeking a subset of significant regressors $\{w_l\}_{l=1}^{n_h}$ for the given training sample set $\{x(t)\}_{t=1}^{N}$.

The training is implemented through iterations. In each iteration a new RBF neuron is added to the hidden layer. The input vector, which results in the largest error reduction ratio, will determine the center of the new added neuron. Then the error of the new neural network is checked. The iteration will be terminated when the error is small enough.

Through the suggested training process the RBF NN has learned the knowledge implied in the training samples and is ready for fault section estimation. It can be seen that the RBF NN designed by the OLS algorithm will be the same as that designed by the so-called exact design method when the number of hidden neurons is up to the number of the training samples. Therefore the exact design method can be considered as a special case of the OLS algorithm. This means that the maximum iteration number of the OLS algorithm will be no more than the number of the training samples and the output error reaches zero for the training samples at this situation. Therefore the RBF NN is very attractive for fault section estimation.

2.3 RBF NN versus BP NN

In fault section estimation, the RBF NN is superior to the BP NN although the latter has been successfully used in various aspects. The two neural networks are both multi-layer feed-forward neural networks. It is known that their performances are strongly dependent to the number of hidden neurons. In RBF NN case the optimal number of hidden neurons can be obtained during the training process, however it is difficult for BP NN to determine the optimal number of hidden neurons. In RBF NN case, the maximum number of iterations will be no more than the number of the training samples and the output error reaches zero for the training samples at this situation. However the BP NN uses gradient decent method to minimize the error and the error might converge very slowly and the residual error might be unacceptable. Therefore the RBF NN can be built up in a fraction of time as compared with the BP NN with convergence and accuracy guaranteed. Generally speaking, the BP NN stores the knowledge globally in its neurons while the RBF NN locally. For the fault section estimation in power systems we think the RBF NN is superior as compared with the BP NN.

3. COMPUTER TEST RESULTS

3.1 Training and performance of the RBF NN

A simple 4-bus power system [14] is used as the test system (Fig. 2), in which there are 9 protected components: 4 buses, 1 transformer and 4 transmission lines. The protection relay system considered in the computer test is a simplified system, which includes transmission lines main protection (*MLP*) and backup protection (*BLP*), main protection for buses (*MBP*) and the transformer (*MTP*).

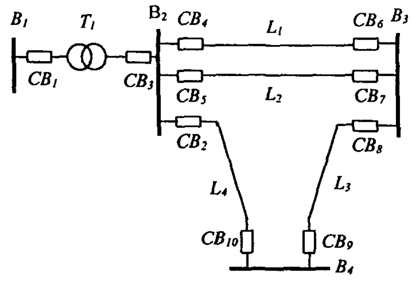

Fig. 2 Test system diagram

In computer tests, forty typical fault scenarios (N=40) are worked out to constitute the training sample set. For each fault scenario, the states of all relays and circuit breakers (0 or 1) are taken as the NN inputs (n_i =33). The states of the 9 system components (4 buses, 1 transformer and 4 lines) are the outputs (n_o=9). If a certain output approaches to 1, then the corresponding component is considered in fault.

Based on the given training samples, RBF NN designed by both the exact design method and the OLS algorithm have been implemented. For the Exact Design method, the number of the hidden neurons equals to the number of the training samples n_h=N=40. The RBF NN has zero error for the training samples via setting its hidden neuron centers equal to the input samples. For a large system the number of hidden neurons will be too large to be acceptable in practical use.

Table 1 shows the number of hidden neurons, number of iterations when the OLS method is used to build up the RBF NN with the spread $\sigma = 2$ and the tolerance $\rho = 10^{-2}$ and 10^{-3} respectively. It can be seen that when ρ increases, the number of hidden neurons is reduced. It is indeed a compromise between estimation accuracy and training/working cpu time. We shall use $\rho = 10^{-2}$ in the rest of the paper.

Table 1 RBF NN with different error tolerance

No. of hidden neurons	37	39
No. of iterations	37	39
Tolerance	$\rho = 10^{-2}$	$\rho = 10^{-3}$

Table 2 presents one training result as an example. Suppose a fault occurs on bus 4. If the main protection *MBP4* of bus 4 refused to operate and the back-up protection BLP_2 and BLP_8 operate correctly to trip the CB_2 and CB_8 to disconnect bus 4 from the rest of the system. The diagnosis outputs and the desired outputs of the RBF NN based on the OLS method are listed in Table 2. We can see the two outputs are very close to each other. Similar results are obtained for all the other training samples, which are not listed here.

Table 2 Diagnosis results of one training sample

	B_1	B_2	B_3	B_4	T_1	L_1	L_2	L_3	L_4
d_j	0	0	0	1	0	0	0	0	0
o_j	-0.000	0.00	-0.000	1.000	0	0.000	0.000	0	-0.00

In order to examine the generalization capability of the designed NN we select other fault scenarios not existing in the training set for testing. Only 12 cases are listed in Table 3. All of them are severe cases with up to 2 mal-operations of relays and circuit breakers or up to 2 simultaneous faults. The corresponding diagnosis results are listed in Table 4. From the output vectors, we can conclude that for all the test cases the faulted elements are recognized correctly.

The test results show that the RBF NN based on the OLS algorithm has very good performance in fault section estimation.

Table 3 Severe test cases

Operated relays and tripped CBs	Fault section
1. MBP_1 BLP_2 CB_1	B_1
2. MBP_2 MLP_7 CB_2 CB_3 CB_4 CB_5	B_2
3. MBP_3 BLP_5 BLP_9 CB_5 CB_6 CB_9	B_3
4. MBP_4 BLP_2 CB_9 CB_{10}	B_4
5. MTR_1 BLP_{10} CB_1 CB_3 CB_{10}	T_1
6. MLP_4 MLP_6 BLP_5 BLP_9 CB_4 CB_5 CB_9	L_1
7. MLP_7 BLP_1 BLP_6 BLP_{10} CB_1 CB_6 CB_7 CB_{10}	L_2
8. MLP_8 MLP_9 BLP_2 CB_8 CB_9	L_3
9. MLP_2 BLP_8 CB_2 CB_8	L_4
10. MTP_1 MLP_8 MLP_9 CB_1 CB_3 CB_8 CB_9	T_1 L_3
11. MBP_4 MLP_2 MLP_{10} CB_2 CB_9 CB_{10}	B_4 L_4
12. MLP_4 MLP_5 MLP_6 MLP_7 CB_4 CB_5 CB_6 CB_7	L_1 L_2

3.2 Comparison with the BP NN

A BP NN is developed for comparison with the RBF NN. It has same maximum tolerance $\rho = 10^{-2}$ with learning rate $\eta = 0.09$ and momentum $\alpha = 0.8$. When the number of the hidden neurons of the BP NN is 37, i. e. equal to that of the RBF NN, the error curve of its training process is given in Fig.3. It can be seen that 5326 iterations should been conducted in order to make $\rho = 10^{-2}$, which is much slower than that of the RBF NN. The cpu time for training is listed in Table 5 for comparison.

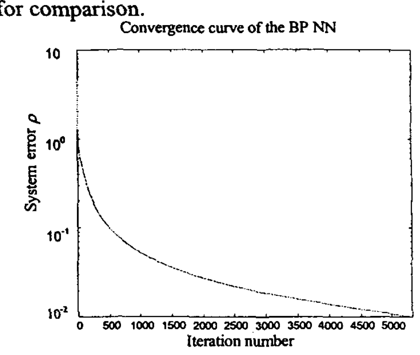

Fig. 3 Convergence of the BP NN

Table 5 Training time for neural networks

	RBF NN (OLS)	BP NN
No. of hidden neurons	37	37
No. of iterations	37	5326
CPU time (sec.)	1.54	133.14

The same 12 severe cases are used to test the performance of the BP NN. The results are listed in Table 6. We can see the output from the BP NN is quite poor as compared with the RBF NN. Besides it yields mal-estimation in case 1 and unclear outputs for severe cases 7, 9, 11 and 12 in real applications.

From this test, we can see that the RBF NN can work much better than the BP NN in fault section estimation under the same training error.

4. CONCLUSION

The RBF NN for fault section estimation in power systems is investigated in this paper. The OLS algorithm has been extended to optimize the parameters of the RBF NN and is proved to be very efficient. It can be seen that the RBF NN has very attractive features in fault section detection. Computer test results show that the RBF NN can be built up in a fraction of time as compared with the widely used BP NN. Computer tests also show that the RBF NN performs very well in fault section estimation especially in the severe cases when there are mal-operations of relays and circuit breakers, which is superior to the BP NN.

Table 4 Diagnosis results for the severe test cases

	B_1	B_2	B_3	B_4	T_1	L_1	L_2	L_3	L_4
1	0.6973	0.0433	-0.040	0.1224	0.1258	0.0764	0.0764	0.0588	-0.060
2	-0.260	1.0613	-0.064	0.1100	0.0448	0.0683	0.0062	0.0605	-0.027
3	-0.187	0.0291	0.9340	0.0973	0.0178	0.2287	-0.047	-0.049	-0.033
4	-0.163	0.0377	-0.027	1.1200	0.0532	0.0468	0.0468	0.0686	-0.063
5	-0.182	-0.001	-0.037	0.1082	1.0194	0.0559	0.0458	0.0546	-0.073
6	-0.083	-0.113	0.0247	0.0167	0.0036	0.9486	-0.028	-0.006	0.0589
7	0.1206	0.1114	0.0481	-0.045	-0.031	-0.056	0.6607	-0.015	0.0072
8	-0.219	0.0286	-0.014	0.0223	0.0368	0.0477	0.0477	1.0225	-0.073
9	0.0392	0.0121	0.0007	0.0018	0.0266	0.0242	0.0242	0.0195	0.8724
10	0.2042	0.0824	-0.0805	0.4012	0.6533	0.2166	0.2166	0.7877	-0.0732
11	-0.0211	0.0854	-0.0263	0.8228	0.0787	0.1299	0.2178	0.0959	0.6047
12	-0.0230	0.0419	-0.0789	0.3106	0.1299	0.7204	0.7204	0.1599	-0.1019

Table 6 Diagnosis results of the BP NN

	B_1	B_2	B_3	B_4	T_1	L_1	L_2	L_3	L_4
1	1.2520	-0.123	-0.042	-0.189	1.0264	0.0111	-0.089	-0.103	0.3211
2	-0.122	1.2367	0.2223	-0.067	-0.088	-0.017	0.4762	-0.164	0.2652
3	-0.488	0.4371	0.8769	-0.183	-0.308	-0.034	0.2926	-0.197	-0.132
4	0.3696	-0.147	01330	0.5690	0.0632	0.0459	0.0972	-0.232	0.2714
5	-0.038	0.0599	-0.036	0.0558	0.8181	0.0764	0.1014	-0.194	0.2502
6	-0.913	0.4115	0.1598	0.3779	-0.393	0.9267	0.1216	0.4251	0.1755
7	0.3257	0.1357	0.0082	-0.139	0.1839	-0.003	0.4774	-0.023	-0.181
8	-0.078	-0.364	-0.090	-0.043	0.2948	0.1027	0.1955	0.7175	0.4074
9	-0.039	0.1225	-0.138	-0.199	0.1367	0.0271	-0.116	-0.198	0.4382
10	0.1213	-0.2257	-0.3323	-0.3273	0.9946	0.1217	0.3311	0.7746	-0.6055
11	-0.0053	-0.2192	-0.4905	0.4408	0.1367	0.2847	0.1222	0.2208	0.7455
12	0.4011	0.4818	-0.1127	-0.0620	-0.3644	0.6372	0.4867	0.1615	-0.0095

REFERENCES

[1] Fukui C. and Kawakami J., "An expert system for fault section estimation using information from protective relays and circuit breakers", IEEE Transaction on Power Delivery, 1986, (4), pp. 83-90

[2] Wen F.S. and Chang C.S., " Possibilistic-diagnosis theory for fault-section estimation and state identification of unobserved protective relays using Tabu-search method", IEE Proceedings on Generation, Transmissions and Distribution, 1998, 145(6), pp. 722-730

[3] Navarro V.; da Silva A.L.; de Carvalho L.A.V. and Zaverucha G., "Artificial neural networks for power systems diagnosis", 1994 IEEE International Conference on Neural Networks, 1994, 6, pp. 3738-3743

[4] Sun Y., Jiang H. and Wang D., "Fault synthetic recognition for an EHV transmission line using a group of neural networks with a time-space property", IEE Proceedings: Generation, Transmission and Distribution, 1998, 145(3), pp. 265-270

[5] Aygen Z. E., Seker S., Bagriyanik M., Bagriyanik F. G. and Ayaz E, "Fault section estimation in electrical power systems using artificial neural network approach", 1999 IEEE Transmission and Distribution Conference, 1999, pp. 466-469

[6] Narendra K. G., Sood V. K., Khorasani K. and Patel R., "Application of a Radial Basis Function (RBF) neural network for fault diagnosis in a HVDC System", IEEE Transactions on Power Systems, 1998, 13(1), pp. 177-183

[7] Schalkoff, Robert J., " Artificial neural networks", New York : McGraw-Hill, c1997.

[8] Chen S., Cowan C.F.N. and Grant P.M., "Orthogonal least squares learning algorithm for radial basis function networks", IEEE Transactions on Neural Networks, 1991, 2(2), pp.302-309