7th Seminar on Neural Network Applications in Electrical Engineering, *NEUREL-2004*
Faculty of Electrical Engineering, University of Belgrade, Serbia and Montenegro, September 23-25, 2004

Estimation of Multi-Pattern-to-Single-Pattern Functions by combining FeedForward Neural Networks and Support Vector Machines

Vijaynarasimha H. Pakka, *Student Member, IEEE*, D. Thukaram, *Senior Member, IEEE*, and H. P. Khincha, *Senior Member, IEEE*

***Abstract*—In many fields there are situations encountered, where a function has to be estimated to determine its output under new conditions. Some functions have one output corresponding to differing input patterns. Such types of functions are difficult to map using a function approximation technique such as that employed by the Multilayer Perceptron Networks. Hence to reduce this functional mapping to Single Pattern – to – Single Pattern type of condition, and then effectively estimate the function, we employ classification techniques such as the Support Vector Machines. This paper describes in detail such a combined technique, which shows excellent results for a practical application in the field of Power Distribution Systems.**

***Index Terms*—FeedForward Neural Networks, Support Vector Machines, Function Estimation.**

I. Introduction

FUNCTION estimation commonly known as function approximation (FA) is typically the estimation of the output of an unknown function for a new input pattern, provided the function estimator is given sufficient training sets such that the unknown parameters defining the function are estimated through a learning strategy [1]. This function is usually a model of a practical system. The training sets are obtained usually by simulation of the system in real time. If the training set is given by,

$$\left\{\left(\bar{x}_1,\bar{y}_1\right),\left(\bar{x}_2,\bar{y}_2\right),\left(\bar{x}_3,\bar{y}_3\right),.....\left(\bar{x}_N,\bar{y}_N\right)\right\} \qquad (1)$$

$\bar{x}$: input pattern vector, $\bar{y}$: target vector, N: number of patterns. Then we need to estimate the functional relation between $\bar{x}$ and $\bar{y}$ i.e.,

$$\bar{y} = c_i f\left(\bar{x};\ t_i\right) \qquad (2)$$

c_i = constants in the function, t_i = parameters of the function, $f: S \to \Box$,

where $S = \left\{x \in \Box^n \mid\ a_i \le x_i \le b_i, 1 \le i \le n\right\}$.

Function Approximation by Multilayer Perceptron Networks, like the FeedForward Neural Networks (FFNNs) [2], [3] is proven to be very efficient, considering various learning strategies like the simple Back Propagation or the robust Levenberg Marquardt [10] and Conjugate Gradient approaches. Radial Basis Function Networks (RBFNs) have also been applied to functional approximation [4]. Like networks with nonlinear transfer functions, RBFNs have the ability to represent arbitrary functions.

Assume there are m_0 number of variables in the function $f(\cdot) \Rightarrow\ f\left(\bar{x}_1,\bar{x}_2,.....,\bar{x}_{m_0}\right) \Rightarrow$ approximate function. Now if the true function is $F\left(\bar{x}_1,\bar{x}_2,.....,\bar{x}_{m_0}\right)$, then the FFNN equates this to

$$\sum_{i=1}^{m_1} \alpha_i \phi\left(\sum_{j=1}^{m_0} w_{ij}\bar{x}_{ij} + b_i\right) \qquad (3)$$

Now the objective is to find the parameter values of m_1 and values of all w_{ij}'s, b_i's and α_i's, such that $\left|F(\cdot) - f(\cdot)\right| < \varepsilon$, for all $\bar{x}_1,\bar{x}_2,.....,\bar{x}_{m_0}$. In the next sections, the data sets obtained from a system are considered as perfect measurements, i.e., no presence of noise. This allows us to investigate the entire problem without touching the topic of generalization, which is not of much debate in the idea behind this presentation.

II. Multi-Pattern-to-Single-Pattern Functions

Let us look at the problem of FA as a mapping problem, where, by one-to-one mapping we mean that each input vector has a corresponding and unique target vector. These mappings are simple to model by FFNNs. This relates to a function that has one output for each input. But, this is not the case in many fields. For example, let us study the case of a sine function.

H.P.Vijaynarasimha is with the Department of Electrical Engineering, Indian Institute of Science, Bangalore 560 012, INDIA (phone: +91-80-23495190; e-mail: vijay@ee.iisc.ernet.in).

D.Thukaram is with the Department of Electrical Engineering, Indian Institute of Science, Bangalore 560 012, INDIA (phone: +91-80-22932362; fax: +91-80-23600444; e-mail: dtram@ee.iisc.ernet.in).

H.P.Khincha is with the Department of Electrical Engineering, Indian Institute of Science, Bangalore 560 012, INDIA (phone: +91-80-22932509; e-mail: hpk@ee.iisc.ernet.in).

Assume that a system has the characteristic shown in Fig. 1a, which has to be estimated. Let the estimation be done by a FFNN with nonlinear transfer functions. The data sets that are available for training of the FFNN are the data corresponding to the two cycles a and b (extreme end cycles in Fig. 1a).

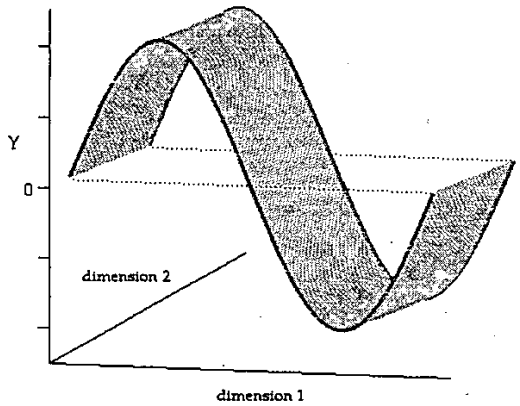

Fig. 1a. Sine wave characteristics of a sample system. Front sine represents cycle 'a' and back sine represents cycle 'b'.

For convenience Fig. 1a is redrawn as Fig. 1b in 2D. Inputs X_{ai} and X_{bi} have same output Y_i. This value is stored by the FFNN in the form of a straight line, if the FFNN is of lower order. For both the inputs running through one cycle, we have a set of such straight lines with varying amplitudes (Fig. 2). However, if the FFNN used were of higher order, then instead of straight lines we would have arbitrary curves in Fig. 2.

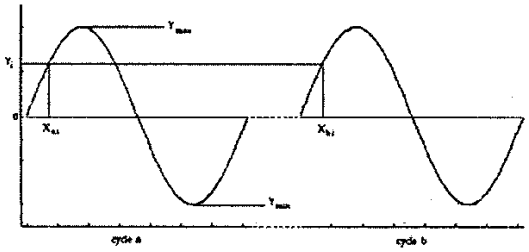

Fig. 1b. Simpler representation of Fig. 1a.

Now suppose there exists an intermediate sine cycle between cycles a and b. if p has similar shape and size as of a and b, then, we can estimate its Y throughout the cycle just by noting the Y values at corresponding intermediate point X_p in Fig. 2. This estimation of Y turns out to be equal to that at X_a or X_b.

Now instead of p being similar to a or b, suppose it to be of different size as shown in Fig. 3. Now, as in Fig. 2, if we estimate Y's at $X=X_p$, the results would not match with that of the true function represented by p. this is due to the fact that the curves joining the two sets of vertical points in Fig. 2 are still straight lines, though in reality they are of the shape of curves with amplitudes (Y's) at X_p different from that at X_a or X_b.

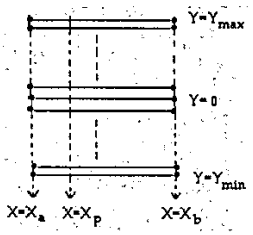

Fig. 2. Graphical depiction of "how FFNN stores input-to-output functional relationship".

Let us name this type of mapping as Two-way mapping or Multi-Pattern-to-Single-Pattern mapping in general, because, estimation of the actual function is the first FA problem, and estimation of the shapes of the curves (lines in Fig. 2) is the second FA problem, i.e., two different input patterns X_{ai} and X_{bi} correspond to a single output pattern Y_i. This is illustrated graphically in Fig. 2 by straight lines.

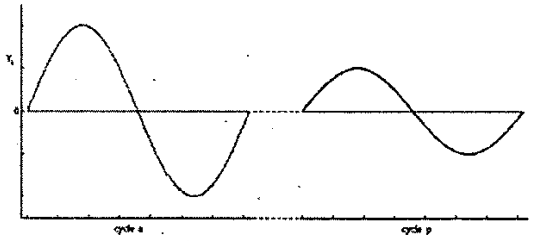

Fig. 3. Different characteristics of the same system.

The misestimation of p is mainly due to insufficient data (cycles) between a and b (if the FFNN used is of lower order). Even if there were data between a and b, this would have called for a strain on the FFNN to learn the entire input space, i.e., higher order FFNNs have to be used in such cases. This strain is because it has to learn in both directions, one in the direction of the sine propagation and the other in the direction of the vectors joining a and b. Instead of working to find the order of the FFNN that has to be employed to approximate such complex functions, we can use simpler FFNNs and still efficiently approximate the complex function. This is done by labeling the datasets and correspondingly classifying them using Support Vector Classifiers (SVCs), which are then combined suitably with the FFNNs so as to give an effective approximation to the overall true 'complex' function of the system under study. The FFNNs are trained using the Levenberg Marquardt algorithm [10] with regularization term, and the SVCs are trained by the simple but yet very fast algorithm of Sequential Minimal Optimization (SMO) [12], [14].

III. Function Estimation by Combined FFNNs and SVCs

We have described in detail, what is meant by the term Multi-Pattern-to-Single-Pattern Functional Mappings. These types of characteristics are often encountered in the modeling of practical systems. To describe and apply the proposed

approach to a practical system, a live topic in the field of power engineering is considered. Fault Location in Transmission and Distribution Industry has received quite interest in the last two decades. Ever since Neural Nets have risen as powerful Function Approximators, the area of fault location has received much more attention [7]. The problem of Fault Location in Distribution Systems, is described in brief.

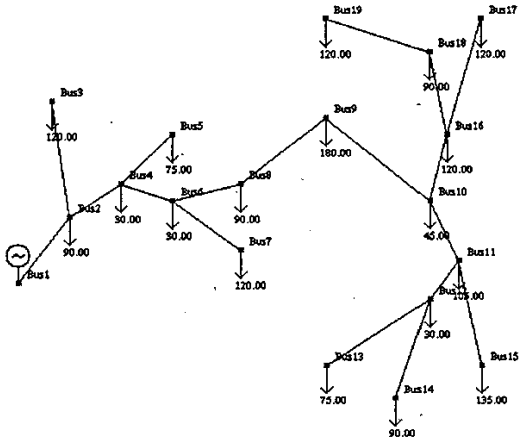

Fig. 4. A practical 19-node (Bus) 11KV distribution system feeder.

Consider a practical 11 KV, 19 node Distribution Feeder shown in Fig. 4. Each node is a distribution transformer with a specified load. The feeder line has a resistance (R) and reactance (X) of 0.0086 *and* 0.0037 *p.u./km* respectively. As R/X ratio is fixed, let us consider X as the only variable.

For the detection fault location, we need to consider various practical aspects involved in the day-to-day operation of a distribution system. In a single day we have various loading patterns, which have to be simulated, and also we need to consider various types of faults that occur in a realistic scenario. During fault conditions, if we consider the three-phase voltage and current measurements at the substation (node 1) as our input elements, we can predict the location of fault by the output of the function estimator. This output is the reactance of the line, which in turn is the length of the faulty part of the line measured from node 1. This is a *single-pattern-to-single-pattern* type of functional mapping, as each measurement vector produces a corresponding and unique output pattern. The other practical factors mentioned above, lead to the *Multi-Pattern-to-Single-Pattern Functional Mapping*, which has to be mapped exactly for the estimation of fault location in real time. For generating the data sets for training, the following procedure is adopted:

- A Short Circuit is simulated with a particular type of fault (Line-Ground (LG), Line-Line (LL), Line-Line-Ground (LLG), Symmetrical 3phase) at a particular node, and at a particular Source Short Circuit (SSC) level (this is to simulate the loading patterns of the system).
- Measurements are noted at the substation.
- Now the SSC level, fault type, and the fault nodes are varied throughout their range, individually, and the data set is built up. The SSC range is from 20MVA to 50 MVA in steps of 5MVA.

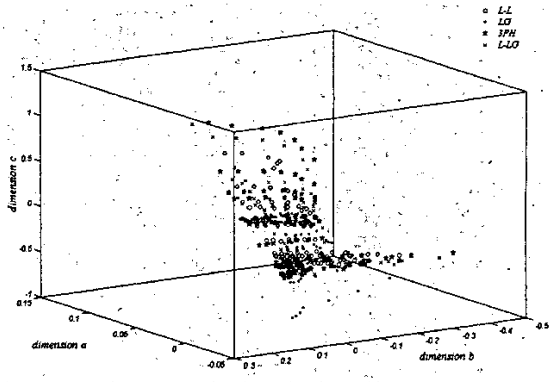

Fig. 5. Dataset of the complete function estimation problem.

We see from Fig. 5 that estimating this complex function is quite difficult for an individual FFNN with any architecture. Hence, the first Function Breakup is by labeling the data according to their fault types and then classifying them by a SVC. In real time, this SVC block classifies the type of fault of an input pattern and the function estimator corresponding to this fault type does the remaining job. This is seen from Fig. 6, where the function looks less complex and can be modeled with less difficulty.

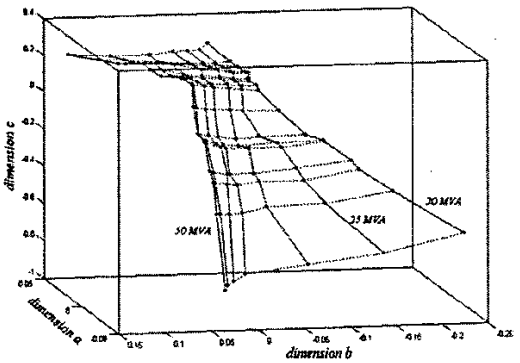

Fig. 6. Dataset corresponding to LG fault. Each dot represents fault on a node, the solid curves represent variation of fault position (nodes), and the dotted curves represent variation of the SSC level.

A SVC is trained, with data of each solid curve being labeled according to their SSC levels. As there are 7 SSC levels that are simulated, we have 6 binary classifiers, which classify patterns to a particular SSC level for further use by the function approximators. Now the work of the function approximators (in this case the FFNN) is cut down to estimation of the solid curves, i.e., data relating to one fault type and one SSC level. If n is the number of SSC levels that are simulated, then the number of classifiers chosen is (n – 1). Thus, there are 6 binary classification problems for each of the SSC level classifiers

'SVM LG' to 'SVM 3ph' to solve during training process, e.g., Classifier 1 classifies faults of 20 MVA and 25 MVA. 'SVM LG' in Fig. 7 refers to SSC level classifier that is trained with Line to Ground faults. The function value $f(x)$ of (A6) points to the class the pattern belongs to i.e., each SVC outputs the pattern as a +ve or -ve function value, indicative of it belonging to either class.

TABLE I
CLASSIFYING LG FAULTS OF TWO LEVELS

Classifier No	Classes (MVA)	32 MVA	33 MVA
1	20 – 25	-3.3638	-3.8435
2	25 – 30	**-2.4922**	-3.3665
3	30 – 35	**1.0156**	**-0.5925**
4	35 – 40	3.8867	**2.4713**
5	40 – 45	7.1996	5.8708
6	45 – 50	8.6052	6.9059

Table I describes the classification of 32 MVA and 33 MVA SSC level faults as that of 30 MVA and 35 MVA SSC levels respectively. The results correspond to LG faults of levels 32 MVA and 33 MVA, simulated on a node. The $f(x)$ value of the 32 MVA fault changes sign at classifier nos. 2, 3 (in third column of Table I - the value of $f(x)$ changes from –2.4922 to +1.0156) and the common class between these two classifiers being 30 MVA, we classify this fault as one that occurred in the group of 30 MVA. Similarly for the 33 MVA fault, which is categorized as belonging to 35 MVA class. The proposed combined approach to the function estimation problem relevant to this application is depicted in the form of a block diagram below. Also, the results in Table II show that the estimation of the unknown function (B6) by the proposed approach has errors in the negligible range of 0.5 – 1.0 %.

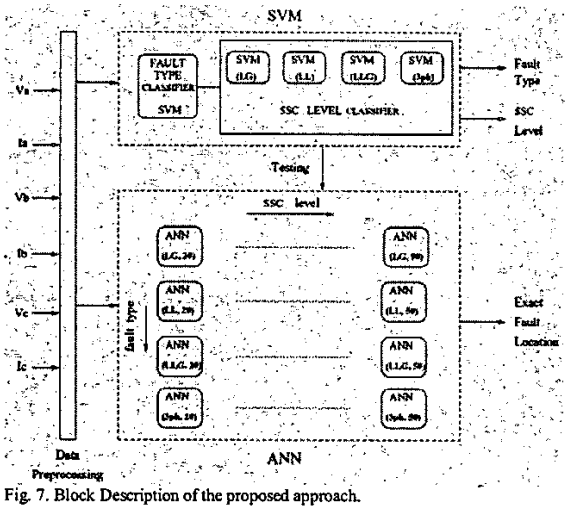

Fig. 7. Block Description of the proposed approach.

TABLE II
TRAINING PATTERNS AND TARGETS FOR LG FAULTS AT A SOURCE SHORT CIRCUIT LEVEL OF 20 MVA AND THE CORRESPONDING OUTPUTS.

Bus No	V_a	V_b	V_c	I_a	I_b	I_c	Target	Output
Source Short Circuit Level: 20 MVA								
2	0.51	0.92	0.98	17.99	1.91	2.02	0.1110	0.1115
3	0.79	0.95	1.00	10.24	1.97	2.08	0.2960	0.2925
4	0.64	0.93	0.99	14.52	1.93	2.09	0.1665	0.1670
5	0.72	0.94	1.00	12.15	1.95	2.10	0.2220	0.2227
6	0.70	0.94	1.00	12.80	1.95	2.11	0.2035	0.2057
7	0.78	0.95	1.00	10.40	1.97	2.11	0.2775	0.2782
8	0.8	0.95	1.00	9.87	1.99	2.14	0.2960	0.2939
9	0.86	0.96	1.01	7.77	2.02	2.14	0.4070	0.4088
10	0.91	0.97	1.01	5.86	2.06	2.14	0.5920	0.5909
11	0.92	0.97	1.01	5.48	2.06	2.14	0.6475	0.6478
12	0.92	0.98	1.01	5.27	2.06	2.14	0.6845	0.6819
13	0.94	0.98	1.00	4.69	2.07	2.13	0.8140	0.8177
14	0.94	0.98	1.00	4.63	2.07	2.13	0.8325	0.8328
15	0.94	0.98	1.00	4.62	2.07	2.13	0.8325	0.8354
16	0.92	0.97	1.01	5.48	2.06	2.14	0.6475	0.6478
17	0.94	0.98	1.00	4.49	2.07	2.12	0.8695	0.8733
18	0.94	0.98	1.00	4.62	2.07	2.13	0.8325	0.8354
19	0.95	0.98	1.00	4.18	2.07	2.12	0.9805	0.9790

IV. COMPARISON WITH SINGLE FFNN

It is proven that 3-layered FFNNs can represent arbitrary mappings [5]. For comparing the proposed approach with a single 3-layered FFNN, the mapping in Fig. 5 is considered. This function being complex, the number of hidden units required by the single FFNN has to be optimally selected. In the present case, the number of hidden neurons chosen is 10.

The training patterns are chosen as in Fig. 5, by simulating 4 types of faults at 7 SSC levels at 18 buses of the feeder. The test patterns were generated similarly, but by simulating faults at the midpoints of two buses in Fig. 4. Hence the number of training /test patterns = 4x7x18 = 504.

TABLE III
RESULTS OF COMPARISON OF SINGLE 3-LAYERED FFNN WITH THE COMBINED APPROACH

	Single FFNN	FFNN – SVM Combined Approach
Training Data	113 / 504	29 / 504
Test Data	227 / 504	51 / 504

'113 / 504' means 113 training patterns out of a total of 504 training patterns produced an error (deviation from true value) of more than 1%..

V. CONCLUSION

We see from section IV that the errors in the case of the proposed 'function reducing' technique are less than that of a single FFNN with larger number of hidden neurons. There are two major advantages of this technique:

- Firstly, a simpler FFNN is enough for Function Approximation, as the complexity of the overall Function

is reduced. Hence, there is no need to give more significance to the number of hidden neurons to be selected.

- Secondly, the SVCs extract more information from the training/input space (e.g., fault type, SSC level), which are always useful apart from just Function Approximation.

Also, the proposed approach shows more usefulness when the dimension of the function approximation problem increases enormously. The application of the proposed technique under such circumstances has to be probed in more detail and also its comparison with other approximation methods has to be made to obtain a clearer picture about the abilities of this method.

APPENDIX

A. Support Vector Classifiers

For the training of the SVM the Sequential Minimal Optimization (SMO) algorithm is used, which is very fast and robust for the present problem.

For training data from the i^{th} and j^{th} classes, we solve the following binary classification problem:

$$\begin{aligned} \min_{w^{ij},b^{ij},\xi^{ij}} \quad & \frac{1}{2}(w^{ij})^T w^{ij} + C\sum_t \xi_t^{ij} \\ & (w^{ij})^T \phi(x_t) + b^{ij} \geq 1-\xi_t^{ij}, \quad if\ y_t = i, \\ & (w^{ij})^T \phi(x_t) + b^{ij} \leq -1+\xi_t^{ij}, \quad if\ y_t = j, \\ i.e., \quad & y_t[(w^{ij})^T \phi(x_t) + b^{ij}] \geq 1-\xi_t^{ij}, \quad t=1,\ldots,k, \\ & \xi_t^{ij} \geq 0. \end{aligned} \tag{A1}$$

where $\phi : \mathbf{R}^n \rightarrow H$ is some nonlinear function.

The training data x_i are mapped to a higher dimensional space by the function ϕ and C is the penalty parameter. The training data x_i are mapped to a higher dimensional space by the function ϕ and C is the penalty parameter. The term $\frac{1}{2}(w^i)^T w^i$ is minimized because this would mean maximizing the margin $2/\|w_i\|$ between the two classes of training data. The penalty term $C\sum_t \xi_t^{ij}$ is required when the classes are not linearly and perfectly separable. The Lagrangian for the above problem is:

$$\begin{aligned} L(w,b,\xi_t;\alpha,\beta) &= \frac{1}{2}(w)^T w + C\sum_{t=1}^{N} \xi_t \\ &+ \sum_{t=1}^{N} \alpha_t[1-\xi_t - y_t(w^T\phi(x_t)+b)] - \sum_{t=1}^{N} \beta_t \xi_t \end{aligned} \tag{A2}$$

By the Wolfe dual problem [20] of maximizing this lagrangian, we get the optimum weights and bias terms as:

$$w = \sum_{t=1}^{N} \alpha_t y_t \phi(x_t) \quad \sum_{t=1}^{N} \alpha_t y_t = 0 \quad C = \alpha_t + \beta_t \tag{A3}$$
$$1 \leq t \leq N, \ \alpha \geq 0, \beta \geq 0, \ 0 \leq \alpha_t \leq C$$

Substituting these in the Lagrangian, we get the dual problem to be solved to get the optimum Lagrangian variables:

$$\begin{aligned} &\max L = q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i=1}^{N} y_i y_j \phi^T(x_i)\phi(x_j)\alpha_i\alpha_j \\ &subject\ to \ \sum_{t=1}^{N} y_i\alpha_i = 0, \quad 0 \leq \alpha_i \leq C \quad 0 \leq i \leq N \end{aligned} \tag{A4}$$

Instead of the dot product $\phi^T(x_i).\phi(x_j)$ we use the dot product Kernel $K(x_i,x_j) = e^{\frac{-[(x_i-x_j)^T(x_i-x_j)]}{2\sigma^2}}$ to avoid the explicit calculation of the function ϕ.

The SMO algorithm searches through the feasible region of the dual problem and maximizes the above objective function, find finding the optimal values of the Lagrangian multipliers. The conditions for optimality can be stated as

$$\begin{aligned} \alpha_i = 0 \quad &\Rightarrow \ y_i f(x_i) \geq 1 \\ 0 < \alpha_i < C \ &\Rightarrow \ y_i f(x_i) = 1 \\ \alpha_i = C \quad &\Rightarrow \ y_i f(x_i) \leq 1 \end{aligned} \tag{A5}$$

Where $f(x)$ is given by,

$$f(x) = \sum_{i \in S} \alpha_i y_i K(x_i, x) - b \tag{A6}$$

Here $S = \{i : \alpha_i > 0\}$ & $b = y_j - \sum_{i \in S} \alpha_i y_i K(x_i, x_j)$ for some j such that $0 < \alpha_j < C$.

B. Functions in the Fault Location Problem

There is a definite relation between the measurements at the sending end, and the distance of fault from this end, for the faulted phase. Consider a fault of a general type occurring on a bus (or on a line section) at a distance of p from the Source side.

$V_{i(0)}^{abc}$, $V_{i(f)}^{abc}$: Voltages at bus *i*, initially and during fault conditions.

Z_{ii}^{abc} : Driving Point Impedance of bus *i*.

I_f^{abc}, Z_f^{abc} : Fault current, fault impedance

$V_{p(f)}^{abc}$: During fault voltage at fault point *p*.

$$V_{p(f)}^{abc} = Z_f^{abc} I_f^{abc} \tag{B1}$$

Voltage of bus *i* during fault is

$$V_{i(f)}^{abc} = V_{i(0)}^{abc} - Z_{ii}^{abc} I_f^{abc} \tag{B2}$$

Similarly voltage at fault point *p* during fault is

$$V_{p(f)}^{abc} = V_{p(0)}^{abc} - Z_{pp}^{abc} I_f^{abc} \tag{B3}$$

Substituting (B1) in (B3), the fault current is obtained as

$$I_f^{abc} = \left(Z_f^{abc} + Z_{pp}^{abc}\right)^{-1} . \; V_{p(0)}^{abc} \qquad \text{(B4)}$$

Substituting the value of I_f^{abc} from (B4) in (B2),

$$V_{i(f)}^{abc} = V_{i(0)}^{abc} - Z_{ip}^{abc} . \left(Z_f^{abc} + Z_{pp}^{abc}\right)^{-1} . V_{p(0)}^{abc} \qquad \text{(B5)}$$

This is the value of voltage at bus i, during steady state fault conditions. From (B5), it is seen that, the voltage at bus i during fault is a function of initial voltage at buses i and p, driving point impedance of bus p, transfer impedance between buses i and p, and the fault impedance. By considering $V_{i(f)}^{abc}$ as a measurement at the sending end, we obtain the distance of the faulty bus p from the sending end. (The distance is implicit of the terms Z_{ip}^{abc} and Z_{pp}^{abc}).

In simple analogy, (B5) is $\bar{y} = f\left(\bar{x}\right)$ (B6)

where,

$\bar{x} = \left(V_s^{abc}, I_s^{abc}, R_f^{abc}\right)$ = Measurements at sending end,

$\bar{y} = g(Z_{pp}^{abc}, Z_{ip}^{abc})$ = Distance of fault from sending end.

g is a function relating the distance of fault node from the source, and the corresponding terms of the Z-bus matrix.

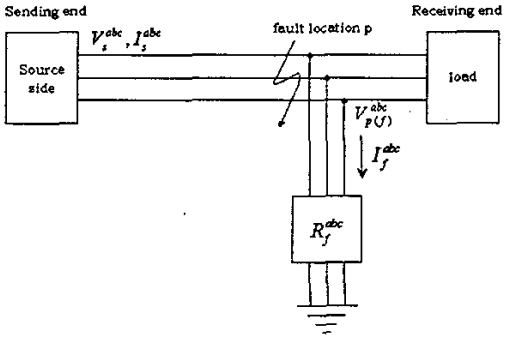

Fig. 8. 3-phase Radial network with general fault at distance p from the source side.

The relation in (B6) will become complex, once varying load conditions, fault resistance, and different types of faults on a feeder, are incorporated. This function would then correspond to numerous inputs mapping onto a single target, e.g., a LG fault occurring at the same location on a feeder, but at different SSC levels. Such type of mapping is difficult to model by a single FFNN. Hence, this paper proposed a new, combined approach, wherein the SVM breaks the complexity of (B6), and the FFNN estimates the unknown parameters in (B6) based on supervised learning.

REFERENCES

[1] D. A. Sprecher, "On the structure of continuous functions of several variables," *Transactions of mathematical Society*, 115, 1964, pp. 340–355.

[2] K. Hornik, "Multilayer feedforward networks are universal approximators," *Neural Networks*, 2, 1986, pp. 359–366.

[3] E. K. Blum, and L. K. Li, "Approximation theory and feedforward networks," *Neural Networks*, Vol. 4, 1991, pp. 511–515.

[4] N. Dyn, "Interpolation and Approximation by radial and related functions," *C.K. Chui, L.L. Schumaker and J.D. Ward, Eds., Approximation Theory* VI: Vol. 1 (Academic Press, 1989) pp. 639–659.

[5] B. Irie, and S. Miyake, "Capacity of Three Layered Perceptrons," *Proc. IEEE ICNN I*, 1988, pp 641–648.

[6] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, Vol. 4, 1991, pp. 251–257.

[7] T. S. Bi, Y. X. Ni, C.M. Shen, F. F. Wu, and Q. X. Yang, "A novel radial basis function neural network for fault section estimation in transmission network," *Proc 5th Int Conf. Adv in Power System Ctrl, Oprn and Mgmt*, Hong Kong, 2000, Volume: 1, pp. 259 - 263.

[8] T. Masters, *Advanced algorithms for neural networks: a C++ sourcebook*, 1995, New York: Wiley.

[9] Chih-Wei Hsu, and Chih-Jen Lin, "A comparison of Methods for Multi-Class Support Vector Machines," Available at: www.csie.ntu.edu.tw/~cjlin/papers

[10] M. T. Hagan, and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Trans. Neural Networks*, vol. 5, pp. 989-993, Nov. 1994.

[11] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, 1998.

[12] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," *Adv in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, Dec. 1998.

[13] R.Fletcher, *Practical Methods of Optimization*, Wiley, New York, 2000.

[14] J. C. Mishra (Ed), *Computing and Information Sciences: Recent Trends*. Narosa Publishing House, New Delhi, 2003.

H.P.Vijaynarasimha (S'04) received the B.E. degree in Electrical & Electronics Engineering from Bangalore University in 2001. Currently he is pursuing his M.Sc (Engg) degree in Electrical Engineering at the Indian Institute of Science, Bangalore. His research interests include computer aided power system analysis, distribution automation, AI techniques and applications, Kernel Based methods, Statistical Theory.

D.Thukaram (SM'90) received the B.E. degree in Electrical Engineering from Osmania University, Hyderabad in 1974, M.Tech degree in Integrated Power Systems from Nagpur University in 1976 and Ph.D. degree from Indian Institute of Science, Bangalore in 1986. Since 1976 he has been with Indian Institute of Science as a research fellow and faculty in various positions and currently he is Professor. His research interests include computer aided power system Analysis, reactive power optimization, voltage stability, distribution automation and AI applications in power systems.

H.P.Khincha (SM'82) received the B.E. degree in Electrical Engineering from Bangalore University in 1966. He received M.E. degree in 1968 and Ph.D. degree in 1973 both in Electrical Engineering from the Indian Institute of Science, Bangalore. Since 1973 he has been with Indian Institute of Science, Bangalore as faculty where currently he is Professor. His research interests include computer aided power system analysis, power system protection, distribution automation and AI applications in power systems.