A COMBINED NEURAL AND GENETIC APPROACH TO SENSOR PLACEMENT

K. Worden, A.P.Burrows and G.R. Tomlinson
Dynamics and Control Research Group
School of Engineering
The University
Oxford Road
Manchester M13 9PL
United Kingdom

Abstract

This paper describes an approach lo *fault* detection and *classification* using *neural networks and genetic algo*rithms. Given *the existence of an effective fault detection procedure, the problem arises as to how the sensors should be placed for optimal efficiency* of *the detector. In* this paper, a *neural network* is *used to locate and clas*sify *faults and a* Genetic *Algorithm (GA)* determines *an optimal (or near optimal) sensor distribution.*

1 INTRODUCTION

The basic problem of fault detection is to deduce the existence of a defect in a structure from measurements taken at sensors distributed on the structure. The problem is essentially *one* of *pattern recognition*. A number of recent papers - [1] and [2] are representative examples have proposed the use of artificial neural networks for fault diagnostics, and it has been shown that they show promise. In the most basic, supervised learning, approach to deriving a neural network, the network is presented with pairs of data vectors, the input being the vector of measurements from the system and the output being the desired fault classification. At each presentation of the data, the internal structure of the network is modified in order to bring the actual network outputs into correspondence with the desired outputs. This iterative procedure is terminated when the network outputs have the required properties over the whole training set. In [1] and [2] the training data was provided by Finite Element (FE) analysis. This has the advantage of allowing a large range of boundary conditions and static/dynamic load cases to be analysed. FE analysis is a little unrealistic as there is no limit on the spatial resolution of the data which is obtained e.g. strains or modeshapes. In reality, the number of sensors available will be limited and this will of course place restrictions on the resolution of data. As a result, it would be necessary in practice to optimise the number and location of sensors for a given problem.

This paper describes three approachs to the problem of sensor placement. The first is based on sequentially deleting sensors which contribute least to the probability of correctly classifying a fault. The second procedure iteratively introduces sensors in order to maximise that probability. The third approach uses a Genetic Algorithm (GA) to determine the optimum distribution of sensors.

2 FAULT DETECTION IN A CANTILEVER PLATE

A cantilever plate of dimensions 300 × 200 × 2.5mm was chosen as the basic structural model. A discretisation into 20 × 20 finite elements was used to generate the training sets for a neural network. Faults of different severity were simulated by "removing" small groups of elements i.e. setting their value of Young's modulus to zero. The 20 × 20 mesh was sub-divided into a 4 × 4 grid, with each grid bounding a group of 5 × 5 elements into which one of three fault severities could be introduced by deleting one, five or nine elements as shown in Fig. 1. These are referred to as fault levels 1, 2 and 3 respectively.

Measurements were assumed to be available at only the nodes of the 4 × 4 grid as shown in Fig. 2, thus only a sub-set of the total information from the FE model was being used.

The basic input/output topology of the fault diagnostic neural network was fixed by the geometry of the model. Measurements were available at the nodes of the 4 × 4 grid (apart from the nodes at the fixed edge), therefore the input layer of the network required 20 nodes. The outputs of the network were required to indicate the existence of faults at the centres of the grid squares and therefore totalled 16. The geometry of the problem did not fix the number and structure of the network hidden layers, this is a matter for trial and error despite the current amount of attention focused on this problem.

The network was trained to respond roughly in proportion to the severity of a fault (which was related to the area of the damage, Fig. I), the desired outputs were set at 0.1, 0.5 and 1.0 for the three fault levels. A single hidden layer was used giving a final network structure with 20:18:16 nodes.

The measurements used to train the networks were modeshapes and curvatures. The curvatures were obtained by twice differentiating the modeshape data on the original 20 × 20 mesh. In practice, the curvatures could be sampled directly using strain gauges.

Training the network on the first mode shape data gave very good results, the difficulty for the network was in reporting accurately the lowest level of fault; the desired output of 0.1 being commensurate with the output variances of the network. This was partly due to the problem of small changes occurring in the mode shapes when the damage was located in the elements at the free end of the cantilever plate. Fig. 3 shows these effects in a comparison between the desired and actual network outputs over the whole training set. Fig 3a shows the comparison for network output 1 which diagnoses problems in an element near the fixed end of the cantilever. The data corresponding to a fault in element 1 is at the beginning of the training set so this is where output 1 is required to be high. The remainder of the data corresponds to faults elsewhere, so output 1 is required to be zero. The noise in the output is low and a correct quantification of the damage is obtained. In contrast Fig. 3b shows the output corresponding to element 4 near the free end, the level of variation is above the minimum value of 0.1 required to detect fault level 1. It may be possible to minimise this insensitivity by weighting the modal responses for certain elements to give more significance in the training data set; this was not investigated in this work.

Had the network been trained using the 20 × 20 mesh instead of the 4 × 4 mesh, improvements in the diagnosis would probably have occurred. However, this highlights the problem of what is possible in practise in terms of the number of measurement points available compared with the data set available from an FE model; the latter can be orders of magnitude higher.

3 SENSOR PLACEMENT

As already stated, using an FE model to generate a training data set for a neural network has few restrictions in terms of the number of data points (corresponding to computed displacements, strains etc). For the plate used in this work, the modeshapes were defined at twenty points (the actual number available was 400), which may be impractical from a modal test point of view. Thus the question arises as to how many sensors are necessary to produce a diagnostic network with a given probability of error and where the optimum positions of the sensors would be. For this study it is assumed that the sensors can only occur be placed at the twenty locations shown in Fig. 2.

The first approach to this problem was based on a defined *measure* of fitness. An initial state is assumed in which all the sensor locations are occupied. If a single sensor is removed, there are clearly twenty possibilities for nineteen-sensor patterns. Data from each of these sensor distributions are used to train diagnostic networks and the the ability of the trained networks to locate/classify the faults are ranked according to the measure of fitness. The *fittest* is considered to be the best nineteen-sensor distribution and the corresponding missing sensor is permanently deleted. This leaves a distribution of nineteen sensors and the procedure is repeated to find the best eighteen-sensor distribution etc.

This is clearly a time consuming process and an procedure was used which allowed automatic fitting of many networks. The networks were trained on the full data sets but used an input mask option which instructed them to disregard data from missing sensors. For example, if measurements from twenty sensors are available, and data from only three sensor locations, say 3, 10 and 17, is used the network program is passed the mask 00100000010000001000. This representation is used throughout this work.

In order to establish a measure of fitness, a normalised mean square error *(MSE) was* defined between the desired network responses y_i, and those estimated by the

network after training $\hat{y}_i$,

$$MSE(\hat{y}) = \frac{100}{N_T \sigma_i^2} \sum_{j=1}^{N_T} (y_i(j) - \hat{y}_i(j))^2$$

where i represents the i^{th} output neuron and N_T is the number of training sets indexed by j. σ_i^2 is the variance of the output y_i. The network performance can be judged on the average value of the *MSE over* the whole set of outputs or the maximum of the set of output *MSEs can* be used.

The first modeshape for the level 3 fault cases were used to train the networks. In this study, noise was added to the computed (FE) modeshapes during the training phase to simulate measurement error and allow the network to generalise appropriately. In the first study, the fitness was established by ranking the networks according to the average and maximum error and number of misclassifications and awarding points accordingly.

In order to test the answers, the best 10-sensor distribution was chosen and compared to ten randomly generated sensor distributions and two distributions suggested by symmetry (deletion of alternate sensors). **Table 3** shows the test results and it can be seen that the algorithmic approach produces the best results in terms of the lowest misclassifications and prediction errors. Fig. 4 shows the best 10-sensor distribution from the algorithm.

Previous work has shown that the use of curvatures as training features for locating faults seems to be more reliable than direct modeshapes [3]. The procedure was therefore repeated using curvatures calculated from the modeshapes for the level 3 faults. As before, the fitness was judged on the average error, maximum error and probability of misclassification. Also a different strategy was assessed. This was an 'insert' strategy where the procedure starts with diagnostics generated for all one-sensor patterns. In this simulation the probability of error alone was used to determine fitness. The distribution with the lowest probability of error is carried forward to the next stage when another sensor is added. The process continues until all twenty sensors have been added. The results for the curvature diagnostics are summarised in Fig. 5 which shows the probability of error as a function of sensor number. The 'insert' strategy using only probability of error is shown to be consistently superior to the 'delete' strategy using the mixed fitness measure. Further, to be sure of a greater than 99% chance of successfully locating the faults, it appears that a 5-sensor distribution is sufficient.

4 SENSOR PLACEMENT USING A GENETIC ALGORITHM

For the sake of completeness, a brief discussion of genetic algorithms (GAs) will be given here, for more detail the reader is referred to the standard introduction to the subject [4]. The training data used for this section were the level 3 curvature data.

Genetic algorithms are optimisation algorithms which evolve solutions in a manner analogous to the Darwinian principle of natural selection. They differ from more conventional optimisation techniques in that they work on encoded forms of the possible solutions. Each possible solution i.e. each set of possible parameters in solution space is encoded as a gene. The most usual form for this gene is a binary string e.g. 00011010110. The first hurdle in setting up a problem for solution by genetic algorithm methods is working out how best to encode the possible solutions as genes. In the case of the sensor location problem, a natural coding is provided by the input masks described earlier e.g. the gene 00100010000000001000 represents a solution in which transducers are placed at positions 3, 7 and 17.

Having decided on a representation, the next step is to generate, at random, an initial population of possible solutions. The number of genes in a population depends on several factors, including the size of each individual gene, which itself depends on the size of the solution space.

Having generated a population of random genes, it is necessary to decide which of them are fittest in the sense of producing the best solutions to the problem. To do this, a fitness function is required which operates on the encoded genes and returns a single number which provides a measure of the suitability of the solution. These fitter genes will be used for mating to create the next generation of genes which will hopefully provide better solutions to the problem. Genes are picked for mating based on their fitnesses. The probability of a particular gene being chosen is equal to its fitness divided by the sum of the fitnesses of all the genes in the population. Once sufficient genes have been selected for mating, they are paired up at random and their genes combined to produce two new genes. The most common method of combination used is called *crossover*. Here, a position along the genes is chosen at random and the substrings from each gene after the chosen point are switched. This is 1 point crossover. In 2 point crossover a second position is chosen and the gene

substrings switched again. There is a natural fitness measure for the sensor location problem, namely the inverse of the probability of misclassification. This is modified here by the addition of a penalty function which suppresses solutions which do not have the desired number of sensors. A simple quadratic penalty function was used.

If a gene in a particular generation is extremely fit, i.e. is very close to the required solution, it is almost certain to be selected several times for mating. Each of these matings however involves combining the gene with a less fit gene so the maximum fitness of the population may be lower in the next generation. To avoid this, a number of the most fit genes can be carried through unchanged to the next generation. These very fit genes are called the elite.

To prevent a population from stagnating, it can be useful to introduce perturbations into the population. New entirely random genes may be added at each generation. Such genes are referred to as new ***blood. Also,*** by analogy with the biological process of the same name, genes may be mutated by randomly switching one of their binary digits with a small probability.

With genetic methods it is not always possible to say what the fitness of a perfect gene will he. Thus the iterative process is usually continued until the population is dominated by a few relatively fit genes. One or more of these genes will generally be acceptable as solutions.

Note that the gene encoding used here is only suitable for numbers of sensors close to ten. This is because the number of distinct n-sensor genes is $\binom{20}{n}$ and this is very strongly peaked at ten. This means that genes introduced into the population by the various operations will have highest probability of having close to ten sensors. For this reason the GA was only used to generate distributions with 9 to 12 sensors. Fifty generations were used in each case, the GA used a single member elite and introduced five new blood at each step. The resulting probabilities of misclassification for the fittest gene in each case are given below.

Number of sensors	Probability of error
9	0.0017
10	0.0017
11	0.0011
12	0.0016

Comparison with Fig. 5 shows that the GA sensor placements outperform those from the heuristic method of the previous section. In fact, the results were quite close, the IO-sensor distribution from the insert strategy had eight sensors in common with that from the GA.

This is very much a preliminary study. Work continues using more complex gene encodings which do not bias the search towards the 10-sensor distributions.

ACKNOWLEDGEMENTS

This work has been supported by Aircraft Systems, The Defence Research Agency.

REFERENCES

[1] **Kudva, J., Munir,** N., and Tan P.W.
Damage detection in ***Smart Structures using Neural Networks.***
Smart Materials and Structures, Vol.1, p100-112, 1992.

[2] **Worden,** K., **Ball,** A.D. and Tomlinson, G.R.
Fault Location in a Framework Structure using Neural Networks.
Smart Materials and Structures, Vol.2, p189-200, 1993.

[3] **Worden,** K. and Tomlinson, G.R.
Damage Location and Quantification Using Neural Networks.
Engineering Integrity Assessment ed. J.H.Edwards, J.Kerr & P.Stanley pp.11.31, 1994.

[4] Goldberg, D.E.
Genetic Algorithms ***in Search,*** Machine ***Learning and Optimisation.***
Addison Wesley, 1989

Source of Pattern	Sensor Pattern	Average Error	Maximum Error	Failures
Algorithm	01010101011000011011	20.03	53.11	0
Symmetry	01010101010101010101	26.74	62.86	1
	10101010101010101010	31.66	72.26	1
Random	01101101000011001110	32.55	77.39	0
	11000100100111001011	29.80	73.85	0
	00010110001111100101	29.32	75.56	1
	01010101001000111101	25.45	67.05	1
	01110111000001101010	32.12	73.20	1
	00010111000110111100	30.12	78.86	1
	11101101110000101000	27.10	79.58	1
	11011101100100000101	29.06	78.79	2
	11011101110000100100	28.84	80.19	3
	11011101110010000001	28.07	78.25	2

Table 1. Test of 'optimum' 10 sensor pattern against random patterns; feature is first modeshape

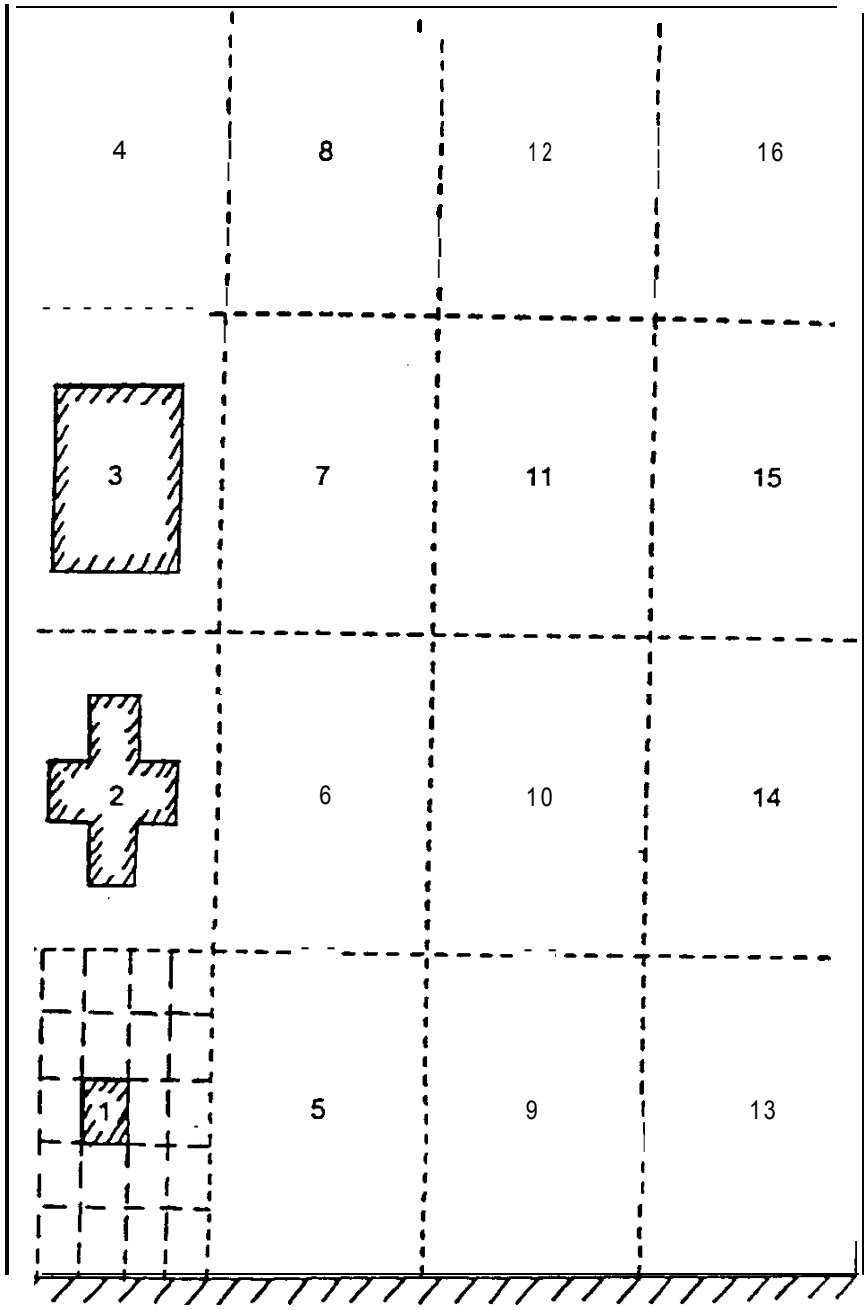


Figure 1 : Numbering system for fault locations and the three fault severity levels indicated by hatched areas.

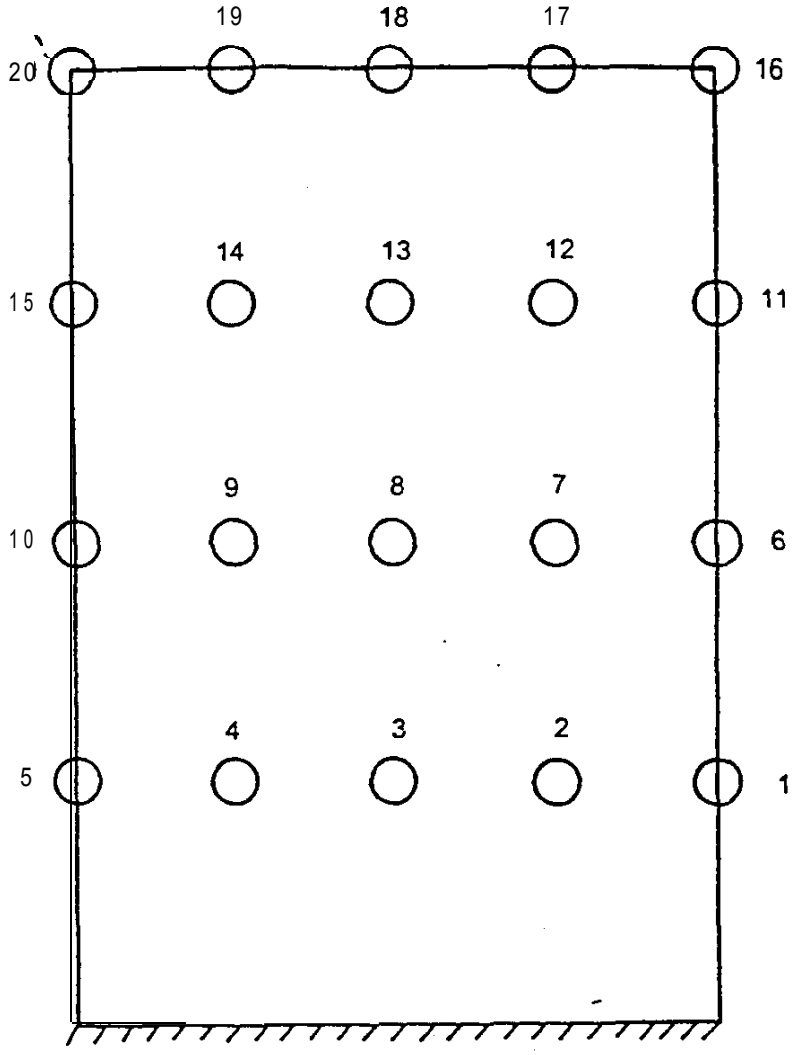


Figure 2 : Numbering system for sensor locations

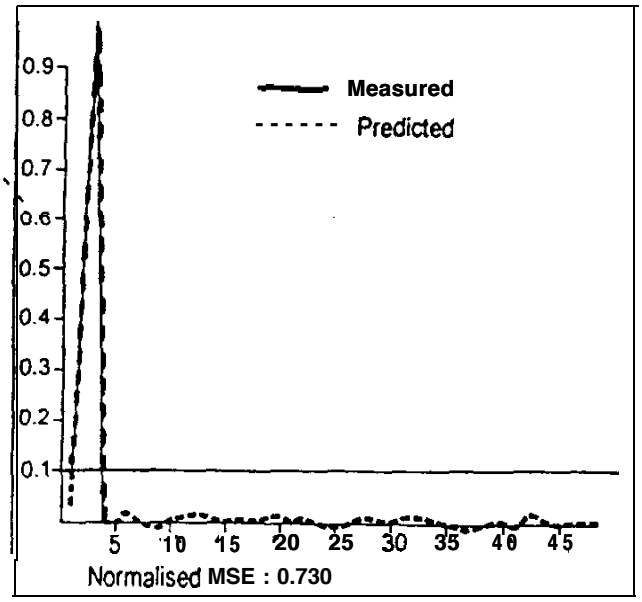


Figure3(a). Comparison between desired output 1 and network prediction for location/severity network trained on the first modeshape.

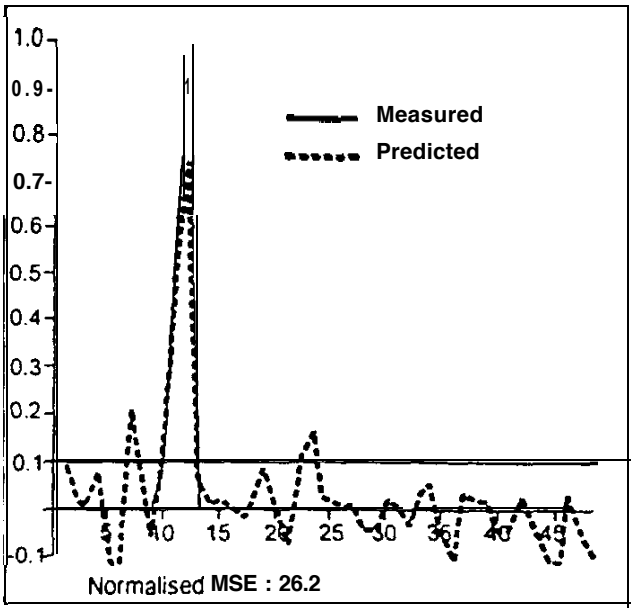
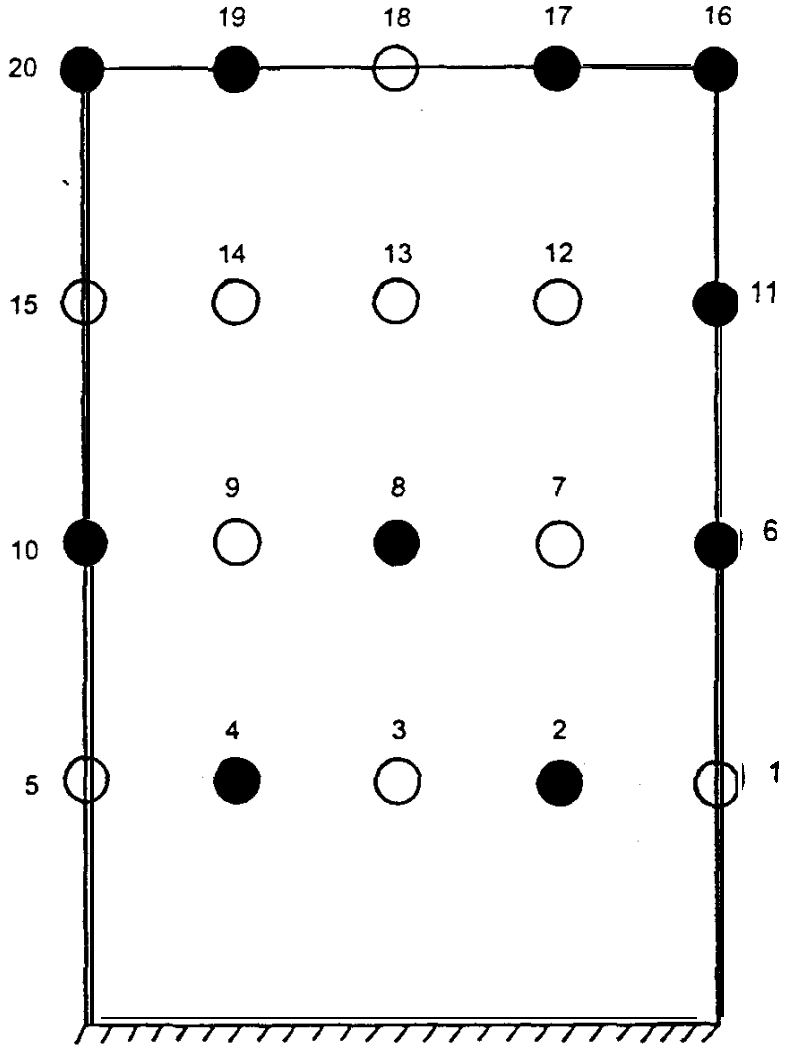


Figure3(b). Comparison between desired output 4 and network prediction for location/severity network trained on the first modeshape.



Number of sensors	10
Code	01010101011000011011
Network	10:12:16
Average error	20:03
Maximum error	53.11
Misclassification	0

Figure 4: Best lo-sensor pattern for modeshape diagnostic. Algorithm based on three separate fitness measures

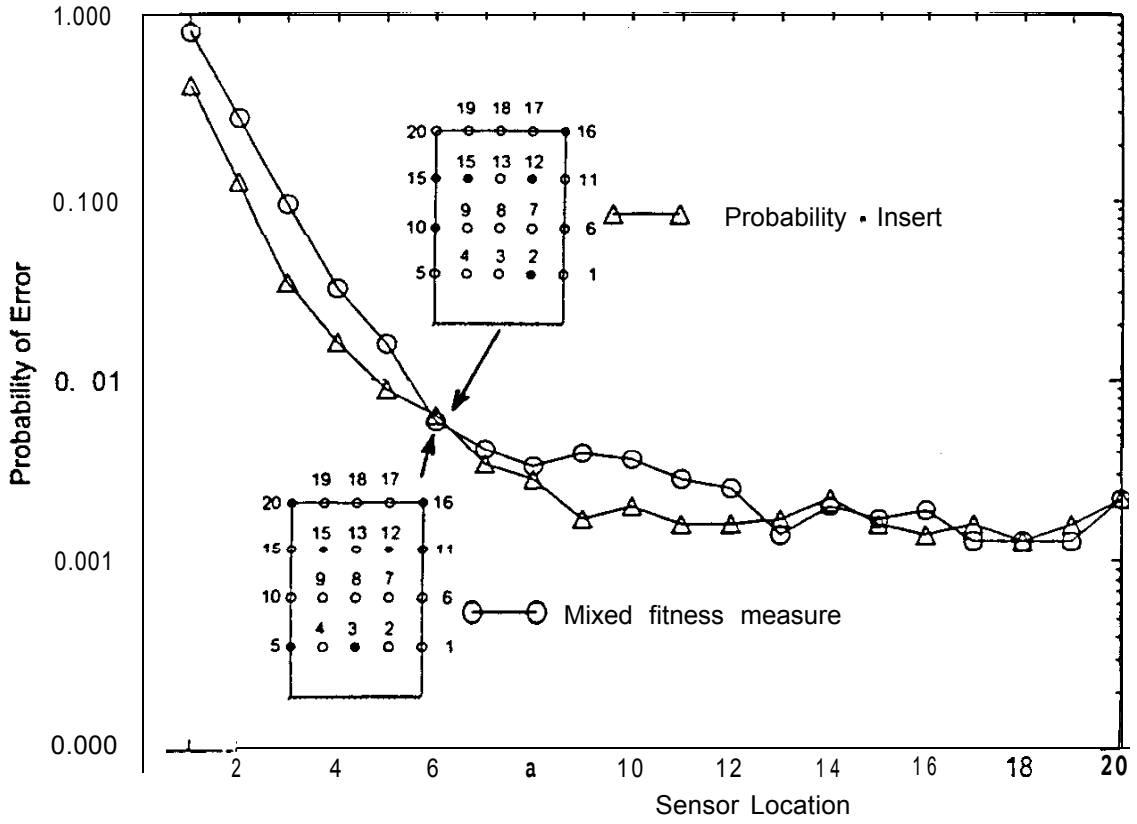


Figure 5 : Probability of error as a function of sensor number.