# Learning Diagnostic Knowledge Through Neural Networks and Genetic Algorithms

**Jie Zhang**

Department of Chemical & Process Engineering
University of Newcastle upon Tyne
Newcastle upon Tyne NE1 7RU
United Kingdom

**Abstract:** Two approaches for learning diagnostic knowledge from past process data or simulation data using neural networks and genetic algorithms are described in this paper. In the first approach multi-layer feed forward neural networks are used to extract the relations between observed abnormalities and the corresponding faults. Through training, neural networks can acquire and store diagnostic knowledge as network weights. The trained networks can be used to diagnose faults in that they can associate the observed abnormalities with the corresponding faults. In the second approach, genetic algorithms are used to train diagnostic rules from process data or simulation data. Genetic algorithm based learning starts with a group of initial rules and produces new rules through reproduction, crossover and mutation. More fitted rules are preserved and less fitted ones are abandoned. Through this evolution-like procedure, effective and concise diagnostic rules can be discovered. The proposed approaches can ease the knowledge acquisition task in developing knowledge based diagnosis system. The proposed approaches have successfully been applied to a pilot scale mixing process.

**Keywords:** Fault diagnosis, neural networks, genetic algorithms, learning, knowledge based systems.

**Jie Zhang** was born in Hebei, China,on June 5,1965. He obtained his B.Sc. degree in Control Engineering from Hebei Institute of Technology, Tianjin, China, in 1986 and his Ph.D degree in Control Engineering from City University, London, in 1991. His Ph.D research topic is on using Artificial Intelligence in on-line process control and fault diagnosis. He has more than a dozen publications and most of them are about on-line process fault diagnosis. Since May 1991, Dr.Zhang has been a Research Associate in Department of Chemical & Process Engineering, University of Newcastle upon Tyne, United Kingdom. His research interests include knowledge based control systems, on-line process fault detection and diagnosis, intelligent control, qualitative modelling, neural networks, genetic algorithms, and robust process control.

## 1. Introduction

Industrial processes are subject to failures during operations. Failures can damage equipments, reduce production efficiency, lead to plant shut-down, or even result in hazardous conditions. The task of prompt detection and diagnosis of faults becomes more and more important as modern industrial processes get more and more complex and environmental constraints turn to be more and more important. Knowledge based systems have shown their great potential in on-line fault detection and diagnosis [Moor and Kramer 86; Tzafestas 89].

Knowledge based fault diagnosis can be divided into "shallow knowledge" and " deep knowledge" based approaches according to the nature of diagnostic knowledge employed. In the shallow knowledge based approach, the knowledge used is usually the empirical association between symptoms of a fault and the fault itself. The knowledge is often represented by rules and, quite often, uncertain reasoning is used since the knowledge is frequently uncertain. In the deep knowledge based approach, the knowledge used includes models of the process being diagnosed and models of common failures of process units. These models can take various forms, e.g. a set of numerical equations, qualitative models, or rules compiled from a model.

One of the most difficult tasks in developing knowledge based systems is knowledge acquisition. As knowledge engineers may often have little knowledge about the operation of a specific process, all the same, process operators may also know little about knowledge engineering. This issue is referred to as the "knowledge engineering bottleneck" [Moor and Kramer 86; Price and Lee 88]. This is especially the case for most experience (or shallow knowledge) based expert systems.

To ease the task of knowledge acquisition, it would be very desirable that a diagnosis system can learn diagnostic knowledge itself. This paper presents two approaches to learning diagnostic knowledge from past process operation data or simulation data. In the first approach, neural networks are used to extract the relations between abnormal process data and the corresponding faults. After training, the acquired knowledge is stored as the trained network weights and the trained neural network can be used to diagnose faults in that it can associate the observed abnormalities with the corresponding faults. In the second approach, genetic algorithms are used to train diagnostic rules from past process operation data or simulation data. Through genetic algorithm based learning, a set of rules best fitted for the training data is obtained. This approach can be easily integrated with expert systems since the diagnostic knowledge learnt is in the form of rules. As genetic algorithms work on a group of rules, a group of various fitted rules is obtained and it may be used supplementarily in cases where none of them is perfect.

The next section generally introduces neural network techniques and presents a neural network based fault diagnosis approach. A pilot scale mixing process is used to demonstrate this approach. Section 3 presents a method for learning diagnostic rules through genetic algorithms. A brief introduction to genetic algorithm is made and genetic learning method is described. An application to a pilot scale mixing process is presented to illustrate the effectiveness of the proposed approach. The last section contains some discussions and conclusions.

## 2. Learning Diagnostic Knowledge through Neural Networks

### 2.1. Feed Forward Neural Networks

A typical feed forward neural network is shown in Figure 1, where neurons are arranged into layers. The output of each neuron is connected to the inputs of all the neurons in the successive layer through corresponding weights. Input layer neurons in Figure 1 simply fan out their inputs and do not process the inputs. The hidden layer and output layer neurons will process their inputs and are marked as squares in Figure 1. One of these neurons is shown in Figure 2 to illustrate how information is processed. The inputs to the neuron are weighted and then summed to give the signal. NET. The output of the neuron is obtained by applying NET to an activation function F.

$$NET = X_1W_1 + X_2W_2 + ... + X_nW_n \quad (1)$$

$$OUT = F(NET) \quad (2)$$

A commonly used activation function is the Sigmoid function

$$OUT = \frac{1}{1+e^{-NET}} \quad (3)$$

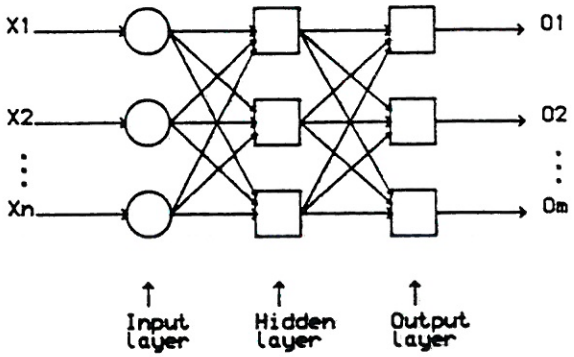


**Figure 1. An artificial neural network**

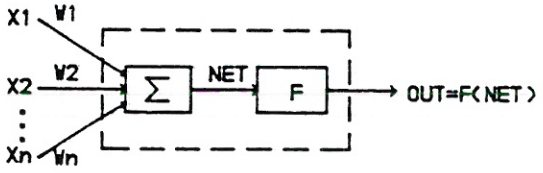


**Figure 2. An artificial neuron**

There are two main operations in the use of feed forward neural networks: training (learning) and generalisation. During training, the network is

provided with input vectors and corresponding target vectors which are called training pairs. An input vector is applied and the output vector of the network is calculated and compared with the corresponding target vector.The difference (error) is fed back through the network and the weights are adjusted according to an algorithm which tends to minimise the error. The input vectors in the training set are applied sequentially and the errors are calculated and weights adjusted for each training pair, until the error for the entire training set is at an acceptable level. On completion of training, the network can operate in a generalisation phase where it produces outputs for similar or novel input patterns.

A widely used training algorithm is the backpropagation training algorithm [Rumelhart et al 86], which belongs to the category of supervised training. The backpropagation training process contains a "forward pass" phase, in which an input vector is applied to the network to produce an output vector, and a "reverse pass" phase, in which the differences between targets and outputs are calculated and the network weights are adjusted to minimise the differences. The weights are adjusted by the following algorithm which minimizes the squared errors.

$$\Delta W_{pq,k}(n+1)=\eta(\delta_{q,k}OUT_{p,j})+\alpha[\Delta W_{pq,k}(n)] \quad (4)$$

$$W_{pq,k}(n+1)=W_{pq,k}(n)+\Delta W_{pq,k}(n+1) \quad (5)$$

where $W_{pq,k}(n)$ is the value of the weight from neuron $p$ in the *jth* layer to neuron $q$ in the next layer (*kth* layer) as step n (before adjustment), $W_{pq,k}(n+1)$ is the weight at step $n+1$ (after adjustment), $\Delta W_{pq,k}(n+1)$ is the adjustment in weight, $OUT_{p,j}$ is the value of OUT for neuron $p$ in the *jth* layer, $\delta_{q,k}$ is a common factor in the gradient of the squared error, $\eta$ is the training rate coefficient, and $\alpha$ is the momentum coefficient. For output-layer neurons (if the *kth* layer is the output layer),

$$\delta_{q,k}=OUT_{q,k}(1-OUT_{q,k})\,(Target_q - OUT_{q,k}) \quad (6)$$

where $Target_q$ is the *qth* element of the target vector corresponding to the *qth* element of the output vector, $OUT_{q,k}$. Finally, for hidden layer neurons (if the *kth* layer is a hidden layer),

$$\delta_{q,k}=OUT_{q,k}(1-OUT_{q,k})\left(\sum_r \delta_{r,l}W_{qr,l}\right) \quad (7)$$

where $W_{qr,l}$ is the weight from neuron $q$ in the *kth* layer to neuron $r$ in the next layer (the *lth* layer). Eq(7) recursively determines the $\delta$ values for each hidden layer.

### 2.2 On-line Process Fault Diagnosis Using Neural Networks

The proposed fault diagnosis system is based on the fact that a neural network can learn. The training data used are a set of symptoms and the corresponding faults. After training, the neural network can find the relationship between a specific symptom and the corresponding fault, and store this information as the trained weights. Since the information about the monitored process is obtained through on-line measurements, the symptoms are represented by on-line measurements. The proposed neural network based diagnosis system is shown in Figure 3.

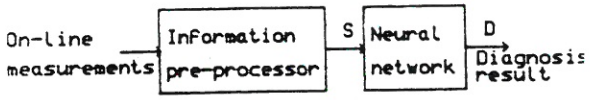


**Figure 3. Neural network based diagnosis**

The on-line measurements are processed, for example scaled, by an information pre-processor into a suitable form which can be applied directly to the network. This processed information is known as the "symptom vector", S, and the outputs of the network indicate the diagnosis result and is termed the "diagnosis vector", D. The training data can be obtained from past experience or from simulation analysis.

The neural network is trained off-line and the trained network is used on-line. During the

training phase, a set of symptom-fault pairs is applied to the network, and the network weights are adjusted by the backpropagation training algorithm. The training time is affected by the network structure, training parameters, and the number of training pairs, and may be a long time and, therefore, it is performed off-line. Once a network is trained it is ready to be used for diagnosis. When abnormalities occur in the on-line measurements, the information pre-processor will process the measurements and produce a symptom vector which is then applied to the trained network, and the diagnosis result is presented by the diagnosis vector. The generalisation phase can then be performed in a sufficiently short time for on-line implementation.

## 2.3 Neural Network Based On-line Diagnosis of a Mixing Process

### 2.3.1 The Mixing Process

The above described neural network based on-line diagnosis technique has been applied to a pilot scale mixing process which is shown in Figure 4, where two tanks in cascade receive hot water and cold water supplies. Both streams enter tank 1 where mixing takes place. The contents of tank 1 passes to tank 2 and subsequently out to a pool tank. Measurements of levels and temperatures in both tanks are available and the level and temperature in tank 2 are controlled by a rule-based controller [Zhang et al 88] resident on a BBC-B microcomputer, which communicates with and is supervised by a Micro-VAX computer. As a main function of supervision, the on-line fault diagnosis system resides on the Micro-VAX computer.

### 2.3.2 Neural Network Based Fault Diagnosis

The information pre-processor shown in Figure 3 for this application is a quantitative to qualitative value converter, which converts the quantitative increments into measurements and controller outputs into their qualitative forms: increase, steady, and decrease. By using such an information pre-processor, training data can be condensed and training effort could be eased. The elements of the symptom vector are qualitative deviations in

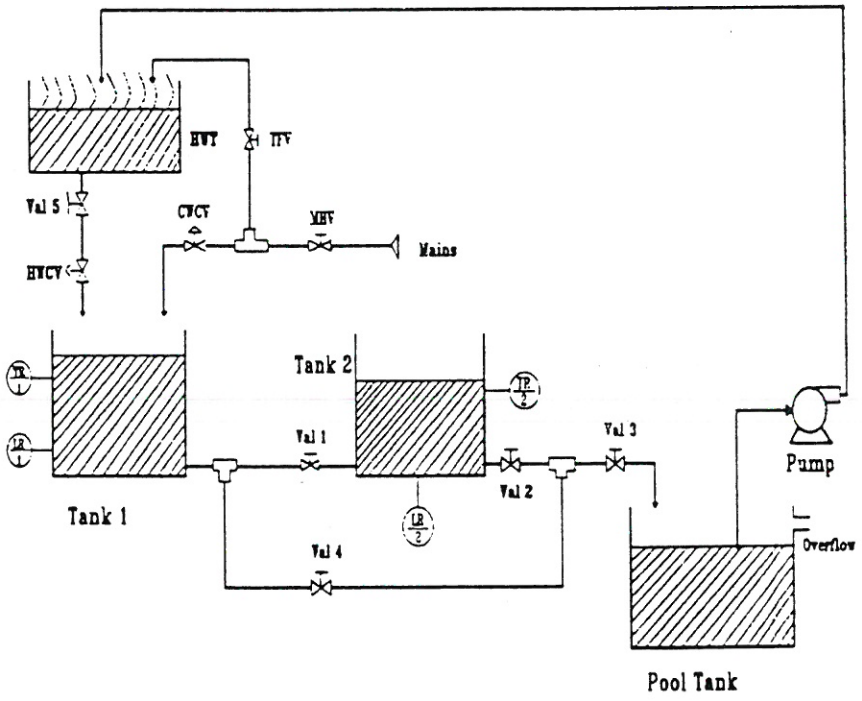


**Figure 4. The mixing process**

measurements and manipulated variables. Here we use 0.1.2. and 3 to represent information unavailable, decrease, steady, and increase, respectively. The assumption that some information may be unavailable has a practical meaning. For example, during operation, some sensors may be out of service and, therefore, the information from these sensors is unavailable.

In the mixing process, the available on-line information includes four measurements and two controller outputs, which determine that there should be six neurons in the input layer, each corresponding to a particular information source. The possible faults that may occur are considered to be: sensor failures, hand valve 1 is blocked, hand valve 2 is blocked, cold water control valve fails and gives a high output, cold water control valve fails and gives a low output, hot water control valve fails and gives a high output, and hot water control valve fails and gives a low output. Since sensor failures may be present in several forms and do not result in relatively fixed symptoms, at this stage, we only consider the other six failures, which determine that there should be six output-layer neurons. Each output-layer neuron corresponding to a particular fault and its output lies in the range (0.1). When its output is close to 1, it indicates that the corresponding fault has occurred. This output can be taken as an approximate measure of the possibility that a fault has occurred, and only those faults with more than 60% possibility are accepted.

The number of hidden layers and the number of hidden layer neurons are determined by the complexity of the classification task. A practically effective approach in selecting the number of hidden layers and hidden layer neurons is to start from a small structure (with one hidden layer and a small number of hidden neurons) and gradually increase network complexity until performance gets satisfactory. Here, we find that a single hidden layer with five neurons gives quite good results and this structure is selected.

### 2.3.3 Network Training

The training data have been obtained from simulation studies of a previously developed diagnosis system [Zhang et al 90], by inserting a fault into the process model and recording the resulted deviations in simulated measurements and controller outputs. The complete training data are listed in Table 1, where S and T are the symptom vector and the target vector, respectively. The elements of S: s1, s2,..., s6, are qualitative deviations of temperatures in tank 1 and tank 2, levels in tank 1 and tank 2, and cold and hot water control valve openings, respectively. Each element of a target vector corresponds to a specific fault and can take the values of either 1 or 0, with 1 representing the occurrence of the corresponding fault and 0 standing for no occurrence.

**Table 1**: Training data for the neural network based diagnosis system for the mixing process

| Training pairs | | | | | | | Faults |
|---|---|---|---|---|---|---|---|
| S: | 1 | 1 | 1 | 1 | 3 | 3 | Hot water control |
| T: | 0 | 0 | 0 | 0 | 0 | 1 | valve fails low |
| S: | 2 | 2 | 1 | 1 | 3 | 1 | Cold water control |
| T: | 0 | 0 | 0 | 0 | 1 | 0 | valve fails low |
| S: | 2 | 2 | 3 | 1 | 3 | 2 | Hand valve 1 |
| T: | 0 | 0 | 0 | 1 | 0 | 0 | is blocked |
| S: | 3 | 2 | 2 | 3 | 1 | 2 | Hand valve 2 |
| T: | 0 | 0 | 1 | 0 | 0 | 0 | is blocked |
| S: | 3 | 3 | 3 | 2 | 1 | 1 | Hot water control |
| T: | 0 | 1 | 0 | 0 | 0 | 0 | valve fails high |
| S: | 2 | 2 | 3 | 3 | 1 | 3 | Cold water control |
| T: | 1 | 0 | 0 | 0 | 0 | 0 | valve fails high |
| S: | 2 | 2 | 2 | 2 | 2 | 2 | |
| T: | 0 | 0 | 0 | 0 | 0 | 0 | No fault |

The learning rate parameter. $\eta$, was set at 0.8, the momentum coefficient, $\alpha$, was set at 0.5, and the initial weights were assigned to small uniformly-distributed random values between -0.1 and 0.1. The stopping criterion used for the training process is that the largest error in the error space is less than 5%.

### 2.3.4 Performance of the Neural Network Based Diagnosis System

The trained network has been tested in both simulations and real plant applications. In simulation studies, the network is tested on a set of incomplete and partially incorrect symptoms, in which some elements in the symptom vector were different from their corresponding items in the training data. These partially incorrect symptoms may be due to measurement noise or some inappropriate parameters in the information pre-processor [Zhang et al 91]. If the training data are obtained from simulation analysis, then any model-plant mismatch may also result in these incorrect symptoms.

**Table 2**: Performance of the diagnosis system under partial and partially incorrect information

| Test No. | Symptom vectors, Diagnosis vectors, Faults | | | | | | |
|---|---|---|---|---|---|---|---|
| | S: | $0^?$ | 1 | 1 | 1 | 3 | $2^*$ |
| 1 | D: | 0.0108 | 0.0002 | 0.0013 | 0.1356 | 0.0181 | 0.7257 |
| | Fault: Hot water control valve fails low | | | | | | |
| | S: | 2 | 2 | $0^?$ | $2^*$ | 3 | 1 |
| 2 | D: | 0.0001 | 0.0773 | 0.0863 | 0.0043 | 0.8554 | 0.0075 |
| | Fault: Cold water control valve fails low | | | | | | |
| | S: | 2 | $1^*$ | 3 | 1 | 3 | $1^*$ |
| 3 | D: | 0.0177 | 0.0315 | 0.0000 | 0.9412 | 0.0252 | 0.0109 |
| | Fault: Hand valve 1 is blocked | | | | | | |
| | S: | $2^*$ | 2 | 2 | 3 | 1 | $1^*$ |
| 4 | D: | 0.0179 | 0.2686 | 0.6050 | 0.0005 | 0.0287 | 0.0013 |
| | Fault: Hand valve 2 is blocked | | | | | | |
| | S: | 3 | 3 | $2^*$ | 2 | $0^?$ | 1 |
| 5 | D: | 0.0051 | 0.8508 | 0.1028 | 0.0041 | 0.0791 | 0.0001 |
| | Fault: Hot water control valve fails high | | | | | | |
| | S: | $0^?$ | 2 | 3 | $2^*$ | 1 | 3 |
| 6 | D: | 0.8003 | 0.0012 | 0.0007 | 0.5242 | 0.0001 | 0.2958 |
| | Fault: Cold water control valve fails high | | | | | | |

The symptoms and the diagnosis results are shown in Table 2, where the incorrect elements in the symptom vector are marked with "*", and the unavailable elements are marked with "?". It can be seen that the neural network based diagnosis system under partially incorrect and partially unavailable information performs well. An explanation for the good performance could be that that the neural network has the ability of abstraction in that it can extract the essential features in the training data. Therefore, when some new data, resembling the training data to some extent, are applied to the neural network, the network can classify the data into appropriate categories.

The network trained on the simulation data has been applied to the real process and the results are also very satisfactory. In one instance, a hot water control valve failure giving a low output was initiated by turning off the hand valve in series with the valve (see Figure 4). The measurements covering this event are shown in Figure 5. The diagnosis system observed that the temperature measurements were abnormal at 110 seconds, when it swiftly collected another four sets of measurements to eliminate the effects of measurement noise. The abnormality was presented in all the five sets of measurements, and then the information pre-processor calculated the symptom vector as

$$S^T = (1\,1\,1\,2\,3\,3).$$

Comparing this with the first training pair in Table 2, it is noticed that the 4th element is different from its counterpart in the training data. The diagnosis result for this symptom is

$$D^T = (0.0343\ 0.0000\ 0.0127\ 0.0264\ 0.0186\ 0.9641).$$

which indicates that the failure, hot water control valve fails giving a low output, has occurred with a high possibility. It can be seen that the diagnosis result is very accurate.

Several faults have been tested in a similar way as in the above example. The symptom vectors, obtained from on-line measurements, and the diagnosis results are shown in Table 3. The elements in the symptom vectors, which are different from their counterparts in the training pairs, are marked with "*". Table 4 shows that the network trained by simulation data performs extremely well on the real process. In Tests 1 and 2, the same failure has been initiated, and the resulting symptom vectors are different, which may be due to measurement noise or to the

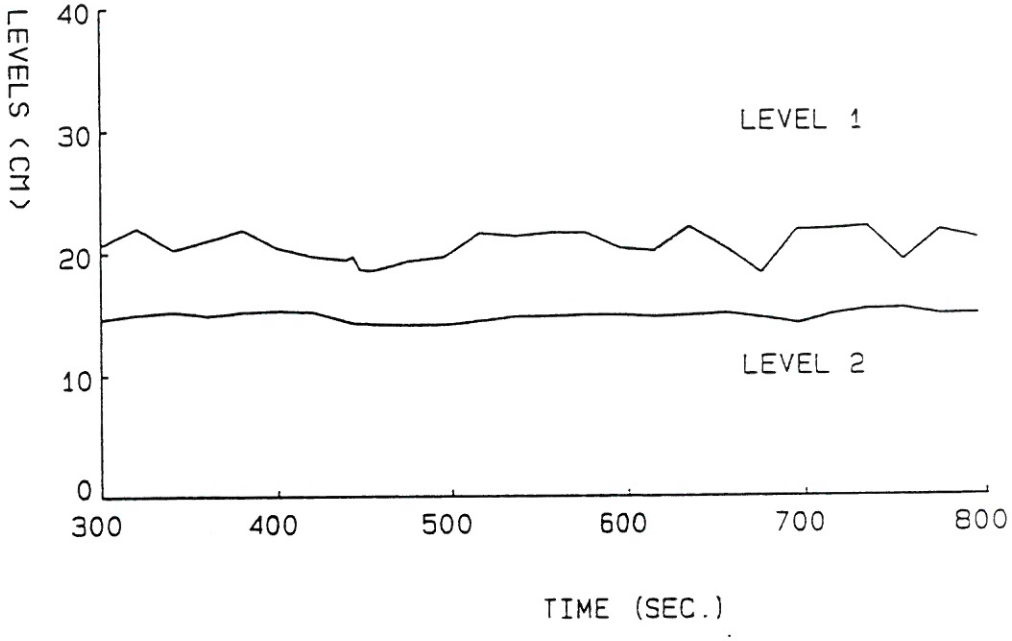


**Figure 5. (a) On-line level measurements**

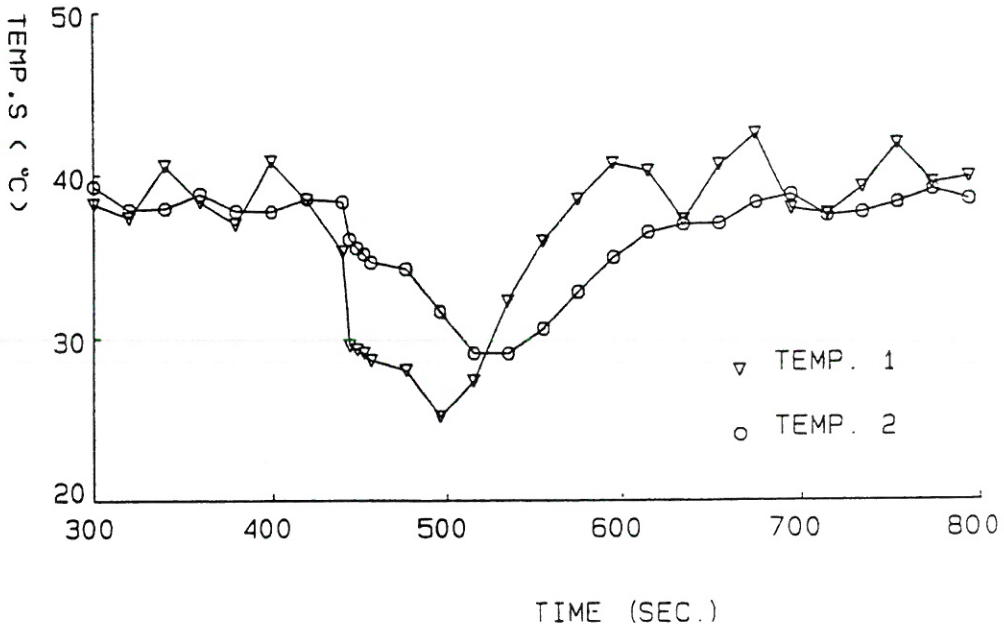


**Figure 5. (b) On-line temperature measurements**

operating conditions being different when the failure was initiated. However, the correct diagnosis result has been obtained for both tests, thus demonstrating the robust nature of the neural network based diagnosis system, in that it can tolerate measurement noise and model-plant mismatch to some extent. In Test 1, the 4th element of the symptom vector is different from its counterpart in the training data, and the diagnosis result shows high accuracy (0.9641). In Test 2, in addition to the 4th element, the 3rd element is also different from its counterpart in the training data, and in this case, the diagnosis accuracy drops down a little bit (0.8769). This demonstrates the graceful degradation in the performance of neural network based diagnosis systems.

The proposed approach has also been applied to the fault diagnosis of a CSTR (continuous stirred tank reactor) system and details can be found in [Zhang and Roberts 92a].

**Table 3**: Performance of the diagnosis system on the real process

| Test No. | Symptom vectors. Diagnosis vectors. Faults | | | | | |
|---|---|---|---|---|---|---|
| | S: 1 | 1 | 1 | 2* | 3 | 3 |
| 1 | D: 0.0343 | 0.0000 | 0.0127 | 0.0264 | 0.0186 | 0.9641 |
| | Fault: Hot water control valve fails low | | | | | |
| | S: 1 | 1 | 2* | 2* | 3 | 3 |
| 2 | D: 0.1577 | 0.0001 | 0.0027 | 0.1549 | 0.0036 | 0.8769 |
| | Fault: Hot water control valve fails low | | | | | |
| | S: 2 | 2 | 1 | 1 | 3 | 1 |
| 3 | D: 0.0000 | 0.0303 | 0.0111 | 0.0333 | 0.9557 | 0.0235 |
| | Fault: Cold water control valve fails low | | | | | |
| | S: 2 | 3* | 1 | 1 | 3 | 1 |
| 4 | D: 0.0000 | 0.0433 | 0.0163 | 0.0234 | 0.9535 | 0.0159 |
| | Fault: Cold water control valve fails low | | | | | |
| | S: 2 | 2 | 3 | 1 | 2* | 3* |
| 5 | D: 0.1251 | 0.0066 | 0.0000 | 0.8939 | 0.0032 | 0.0542 |
| | Fault: Hand valve 1 is blocked | | | | | |
| | S: 2* | 2 | 2 | 3 | 1 | 2 |
| 6 | D: 0.2032 | 0.0156 | 0.6780 | 0.0003 | 0.0018 | 0.0280 |
| | Fault: Hand valve 2 is blocked | | | | | |

# 3 . Learning Diagnostic Rules through Genetic Algorithms

## 3.1 A Brief Introduction of Genetic Algorithms

Genetic algorithms are search algorithms based on the mechanics of natural selection and natural genetics [Holland 57; Goldberg 89]. Genetic algorithms start with a group of knowledge structures which is usually coded into binary strings. These structures are evaluated within some environment and a measure of the fitness (the strength) of a structure should be defined. The fitnesses of all individual structures are calculated and a new set of structures are then formulated by randomly selecting structures from the current group. This process is usually called reproduction or selection. Each structure is selected with a probability determined by its fitness and, through such a means, the structures fitted for the environment will survive and those not fitted will be extinct. The selected group of structures are then undergoing certain genetic operations, such as crossover and mutation, which provide a structured yet randomised information exchange among the structures. For example, through the crossover operation, the structures will be paired and each pair of structures will randomly exchange a certain amount of information. This will be demonstrated later. The resulting structures are then evaluated and the process is repeated until satisfactory structures have been produced.

Genetic algorithms differ from conventional optimisation and search procedures in the following ways: 1) genetic algorithms work with a coding of the parameter set, not the parameters themselves; 2) genetic algorithms search from a population of points, not a single point; 3) genetic algorithms use pay-off (objective function) information, not derivatives or other auxiliary knowledge; 4) genetic algorithms use probabilistic transition rules, not deterministic rules [Goldberg 89]. All these enable genetic algorithms to perform robustly and be applied to many problems. Genetic algorithms are mainly applied in optimisation and machine learning and a survey of them is provided in [Goldberg 89].

### 3.2 Learning Diagnostic Rules with Genetic Algorithms

#### 3.2.1 Rule Coding

Since genetic algorithms work with a group of structures which is usually coded into binary strings, the diagnostic rules which are to be evaluated through genetic algorithms should then be coded. Diagnostic rules used here are in the following form:

**IF** $S_1$ & $S_2$ & ... & $S_n$

**THEN** *Fault*

which states that if symptoms $S_1$ to $S_n$ are present then the ith fault occurs.The symptoms $S_1$ to $S_n$ used here correspond to n different on-line information sources, which could be on-line measurements and controller outputs,and each symptom is considered to take one of the following values: increase,steady,decrease, and "*" which is a wild card and means that the corresponding symptom is not important. When applying a rule,"*" can match with any values. By introducing this wild card,the condition parts of all the diagnostic rules will be of the same length and the corresponding parts of the rules will represent the same observations.

Each symptom in the condition part of a rule is coded by a two- bit binary with "00" standing for "decrease", "01" standing for "steady", "10" standing for "increase" and "11" standing for "*",the wild card. By such a means, a rule whose condition part includes n symptoms will be coded into a 2n-bit binary string consisting of "0"s and "1"s.

#### 3.2.2 Training Data

Training data are divided into different groups corresponding to the different possible faults. Each group of training data represents the qualitative states of the process under the corresponding fault with different severities. A group of training data representing normal operating conditions is also required. Training data could be obtained from simulation studies or from previous operating experience of the process. If a model of the diagnosed process is available, then the training data can be obtained by inserting a fault to the model at various severities and then recording the qualitative increments of measured variables and controller outputs.

#### 3.2.3 Fitness Functions

To use genetic algorithms, the fitness of each individual structure,which is calculated from a pre-defined fitness function,is required. The fitness of an individual structure is a measure indicating how fitted the structure is. In the self-learning of diagnostic rules, a better rule should have more applications when tested by the training data corresponding to this rule and less incorrect applications when tested by the other training data. A better rule should also be simpler in that its condition part contains more "*"s. Let NSA be the number of successful applications of a rule when tested by the training data corresponding to this rule,NIA be the number of incorrect applications of the rule when tested by the rest of the training data,and NOS be the number of "*"s in the condition part of the rule,then the fitness of the rule ,F, can be calculated as follows:

$$F=\alpha NSA - \beta NIA + \gamma NOS + M \qquad (8)$$

where $\alpha$, $\beta$ and $\gamma$ are positive weighting coefficients. M is a positive offset to ensure that F will not be negative. The requirement that F should not be negative is determined by the genetic algorithm used here. In the reproduction phase,an individual rule is selected with a probability which is equal to the ratio of the fitness of the rule and the sum of the fitnesses of all the rules in the generation. Therefore,the fitness of a rule should not be less than zero.

The fitness function can also be chosen as follows:

$$F=\frac{\alpha NSA}{\beta NIA+1} + \gamma NOS \qquad (9)$$

where $\alpha$, $\beta$ and $\gamma$ are positive weighting coefficients.

A problem associated with the first fitness function is that sometimes M could be quite large and this makes the fitness function insensitive to changes in NSA,NIA, and NOS. By inspecting the first order partial derivatives (suppose that NSA,NIA,and NOS could change continuously), it is found that the second fitness function is more preferable than

the first one. Differentiating F in Eq(9), with respect to NSA,NIA, and NOS respectively,gives

$$\frac{\delta F}{\delta NSA}=\frac{\alpha}{\beta NIA+1} \tag{10}$$

$$\frac{\delta F}{\delta NIA}=\frac{-\beta NSA}{(\beta NIA+1)^2}=\frac{-\beta NSA}{\beta NIA+1}\frac{\delta F}{\delta NSA} \tag{11}$$

$$\frac{\delta F}{\delta NOS}=\gamma \tag{12}$$

From Eq(10) and Eq(11), it can be seen that the second fitness function is more sensitive to changes in NSA and NIA when a rule is close to the desired rule (with large NSA and small NIA).

The values of the parameters $\alpha, \beta$ and $\gamma$ in Eq(9) can affect the result of learning. The choices of $\alpha, \beta$ and $\gamma$ are determined by the quantity of training data and the user's objectives. Eq(11) shows that large $\beta$ will impose more penalty on incorrect applications. Suppose that one rule has NSA = 15, NIA = 2,and NOS = 0, and a second rule has NSA = 12,NIA = 1, and NOS = 0. If it is chosen that $\alpha = 5$, $\beta = 0.5$, and $\gamma = 2$, then the fitnesses for the two rules are 37.5 and 40,respectively and the second rule is more preferable.If it is chosen that $\alpha = 5$, $\beta = 0.1$, and $\gamma = 2$, then the fitnesses for the two rules are 62.5 and 55.6, respectively and the first rule is better. The experiments follows show that large $\beta$ will produce rules which are more " cautious " in that they rarely have any incorrect applications but they could miss some faults,while small $\beta$ will produce rules which generally will not miss the corresponding faults but could produce incorrect diagnoses.

### 3.2.4 The Genetic Algorithm

The genetic algorithm used here is the three operator algorithm [Goldberg 89] with slight modifications. Modifications are required in cases where the best structure in the new generation (after reproduction, crossover,and mutation) is not as good as the best one of the previous generation to preserve the best structure. In such situations, the worst rule in the current generation is replaced by the best rule in the previous generation. By such means, there will be no such dangers that the best rule ever discovered will be lost. This is similar to De Jong's elitist model [Goldberg 89]. The genetic algorithm therefore contains the following phases: reproduction, crossover, mutation, and preservation of the best rule.

#### (a) Initial population

There are several reported approaches to initialising the knowledge structure of a genetic algorithm. The most commonly used is random initialisation.This approach requires the least knowledge acquisition effort and provides a lot of diversity for the genetic algorithm to work with. Another approach is to seed the initial population with existing knowledge [Grefenstette 87]. By such means,there will be more fitted structures in the initial population. In this paper,we use the second approach. To build initial rules for a specified fault,the training data corresponding to this fault are inspected and the different patterns in the training data are found.The initial rules are then developed by copying these different patterns and then deviating the symptoms in possible directions (i.e. not deviating from increase to decrease or vice versa). From the knowledge about a particular failure, it may be found that under this fault certain measurements will definitely deviate in certain directions. In creating the initial population,these symptoms are mainly kept unaltered and the only acceptable alteration is "*" whereas other symptoms are deviated in various possible directions to give a large diversity.

#### (b) Reproduction

Fitnesses of the rules are calculated in the reproduction or selection phase.A new generation of rules, with the same size as the current generation, is produced by randomly selecting rules from the current generation.The probability that a rule being selected is defined as follows:

$$P(i)=\frac{F(i)}{\sum_{j=1}^{N} F(j)} \tag{13}$$

where P (i) is the probability that the ith rule will be selected, F (i) is the fitness of the ith rule, N is the total number of rules in the current generation,

and j is a running index taking the values from 1 to N. By such means, more fitted rules will be selected with more chances and produce more copies in the new generation.

**(c) Crossover**

After the reproduction phase,the rules are randomly mated to undergo the crossover operation. Each pair of rules will undergo crossover operation with the probability PCROSS, which is generally quite large (e.g. 0.95). Through the crossover operation,the two rules in a pair will exchange a certain amount of inform- ation. The amount of information exchanged is determined by the crossover site which is randomly selected among all possible sites with a uniformly distributed probability. For example, given the following two rules:

**IF** $S_{11}$ & $S_{12}$ & $S_{13}$ & $S_{14}$ **THEN** F

**IF** $S_{21}$ & $S_{22}$ & $S_{23}$ & $S_{24}$ **THEN** F

there are three possible crossover sites in the condition parts of the rules, and if the crossover site is chosen at the first possible site, i.e. between the first and the second symptoms,then the crossover operation will produce the following two new rules:

**IF** $S_{11}$ & $S_{22}$ & $S_{23}$ & $S_{24}$ **THEN** F

**IF** $S_{21}$ & $S_{12}$ & $S_{13}$ & $S_{14}$ **THEN** F

The crossover operation is an important means for combining different knowledge structures in order to produce new knowledge structures.

**(d) Mutation**

Following the crossover operation is the mutation operation which performs bit by bit changes in the coded rules (i.e. from 0 to 1 or from 1 to 0) with a probability, PMUT, which is typically very small (e.g. 0.001). The mutation operator can occasionally produce new knowledge structures.

**(e) Preserving the best discovered rule**

There may exist such situations that the best rule in the current generation is not as fitted as the best one in the previous generation. This could be the result of crossover disruptions [Liepins and Vose 90]. If such situations are encountered,the worst rule in the current generation is replaced by the best rule of the previous generation, which is the best rule ever discovered. Through such a means, the best ever discovered rule will never be lost.

### 3.3 Learning Diagnostic Rules for the Mixing Process

#### 3.3.1 Learning Diagnostic Rules

The available on-line information on the mixing process consists of four measurements and two controller outputs,which determine that the condition part of a diagnostic rule should contain six symptoms each one of them corresponding to an on-line information source.The rules are coded into several 12-bit binary strings. The training data are generated through simulations of the following six faults: cold water control valve fails and gives a high output,cold water control valve fails and gives a low output,hot water control valve fails and gives a high output,hot water control valve fails and gives a low output,hand valve 1 is blocked,and hand valve 2 is blocked. During simulation,these faults were initiated at a variety of severities and the corresponding responses of the process are recorded and transmitted into a symbolic form.Corresponding to the six faults,there are six groups each containing 16 sets of training data. Apart from these, there is another group of training data representing the normal operating conditions. Corresponding to the six faults,there are six groups of rules to be evaluated. Each group contains 30 rules and through genetic algorithm based learning,it is required to discover the best rule for each fault.

The crossover probability, PCROSS, is set to 0.95, the mutation probability, PMUT, is set to 0.001, the parameters used in the fitness function $\alpha$, $\beta$ and $\gamma$ are set as 6, 0.15, and 2.2,respectively. The values of $\alpha$,$\beta$ and $\gamma$ are chosen through trial and error but are roughly guided by the quantity of training data. Equations (10) to (12), and the discussions following these equations. For example, Eq (11) shows that at a given stage (with given NSA ana NIA) $\beta$ will determine the relation between the effect of changing NSA on F and that

of changing NIA on F and at a certain stage the two effects can be balanced by choosing certain $\beta$ .

The performance of the learning system is presented in Figures 6 to 11, which illustrate the fitness of the best rule and the average fitness of each group during learning. Clearly, these Figures demonstrate that better rules are discovered during learning. The best rules learnt for each group are listed below:

Rule No. l:

**IF**

Cold water flow Increase

Hot water flow Decrease

**THEN**

Cold water control valve fails low

Rule No.2:

**IF**

Level in tank 1 Increase

Level in tank 2 Increase

Cold water flow Decrease

**THEN**

Cold water control valve fails high

Rule No.3:

**IF**

Temp. in tank 1 Decrease

Temp. in tank 2 Decrease

**THEN**

Hot water control valve fails low

Rule No.4:

**IF**

Temp. in tank 1 Increase

Temp. in tank 2 Increase

**THEN**

Hot water control valve fails high

Rule No.5:

**IF**

Level in tank 1 Increase

Cold water flow Increase

**THEN**

Hand valve 1 is blocked

Rule No.6:

**IF**

Level in tank 1 Steady

Level in tank 2 Increase

Cold water flow Decrease

**THEN**

Hand valve 2 is blocked

The fitnesses and the values of NSA,NIA ,and NOS of the best rule and the second best rule learnt for each fault are provided in Table 4.It can be seen from Table 4 that the rules learnt for fault 1 and faults 3 to 5 are perfect in that there is no missed fault when tested by their corresponding training data (NSA = 16) and no wrong diagnosis when tested by the other training data (NIA = 0).It should be realized that due to the effects of measurement noise, the dynamics of the process,various severities of faults represented in the training data,and the representation of training data which is determined by the form of the rules to be learnt, there may exist overlappings between different groups of training data and this determines that perfect rules may not exist for some faults. The rules learnt for fault 2 and fault 6 are not as satisfactory as other rules. By inspecting the two groups of training data corresponding to the two faults it is found that there exist three sets of overlapping data between the two groups and this determines that the learnt rules for the two faults could not be as satisfactory as other rules. The best rule learnt for fault 2 will not miss any faults when tested by its corresponding training data (NSA = 16) but can result in five incorrect diagnoses when tested by the other training data (NIA = 5). The second best rule learnt for this fault will miss three faults when tested by its corresponding training data (NSA = 13) and can result in three incorrect diagnoses when tested by the other training data (NIA = 3). Since the learning system produces a group of fitted rules after learning, these rules may be used supplementarily in cases where none of these rules is perfect.

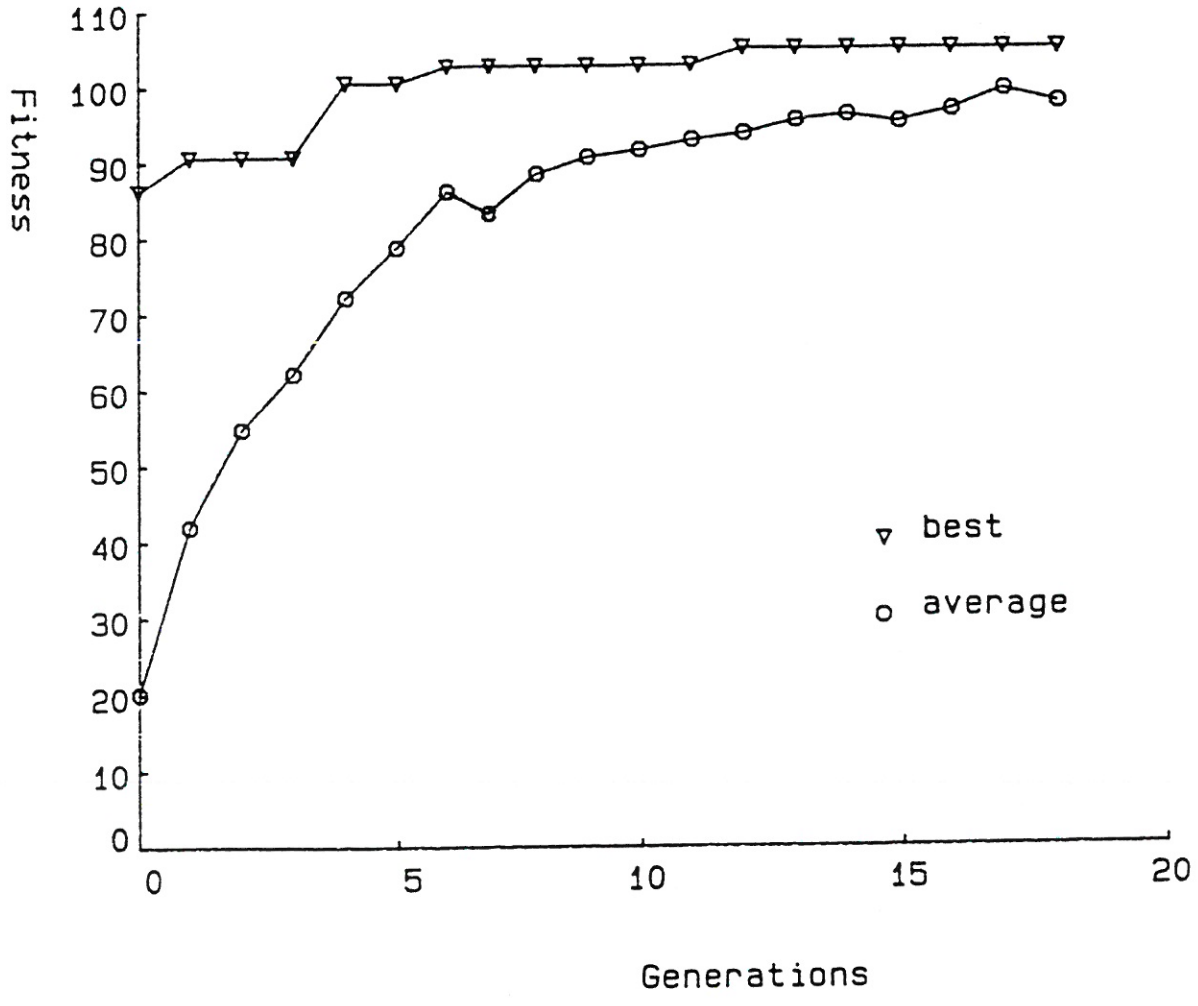


**Figure 6. Largest and average fitness in learning rule No. 1 for the mixing process**

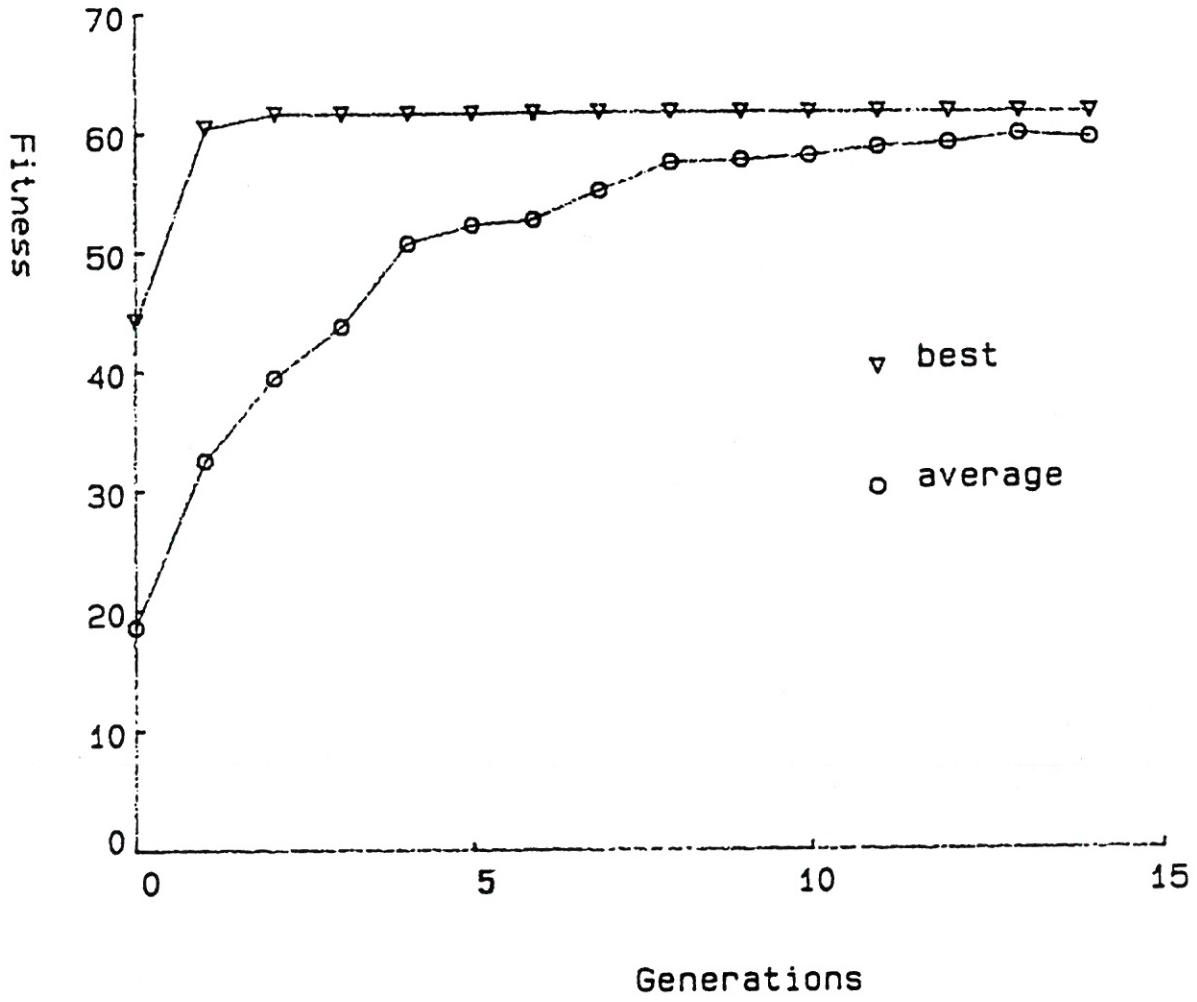


**Figure 7. Largest and average fitness in learning rule No. 2 for the mixing process**

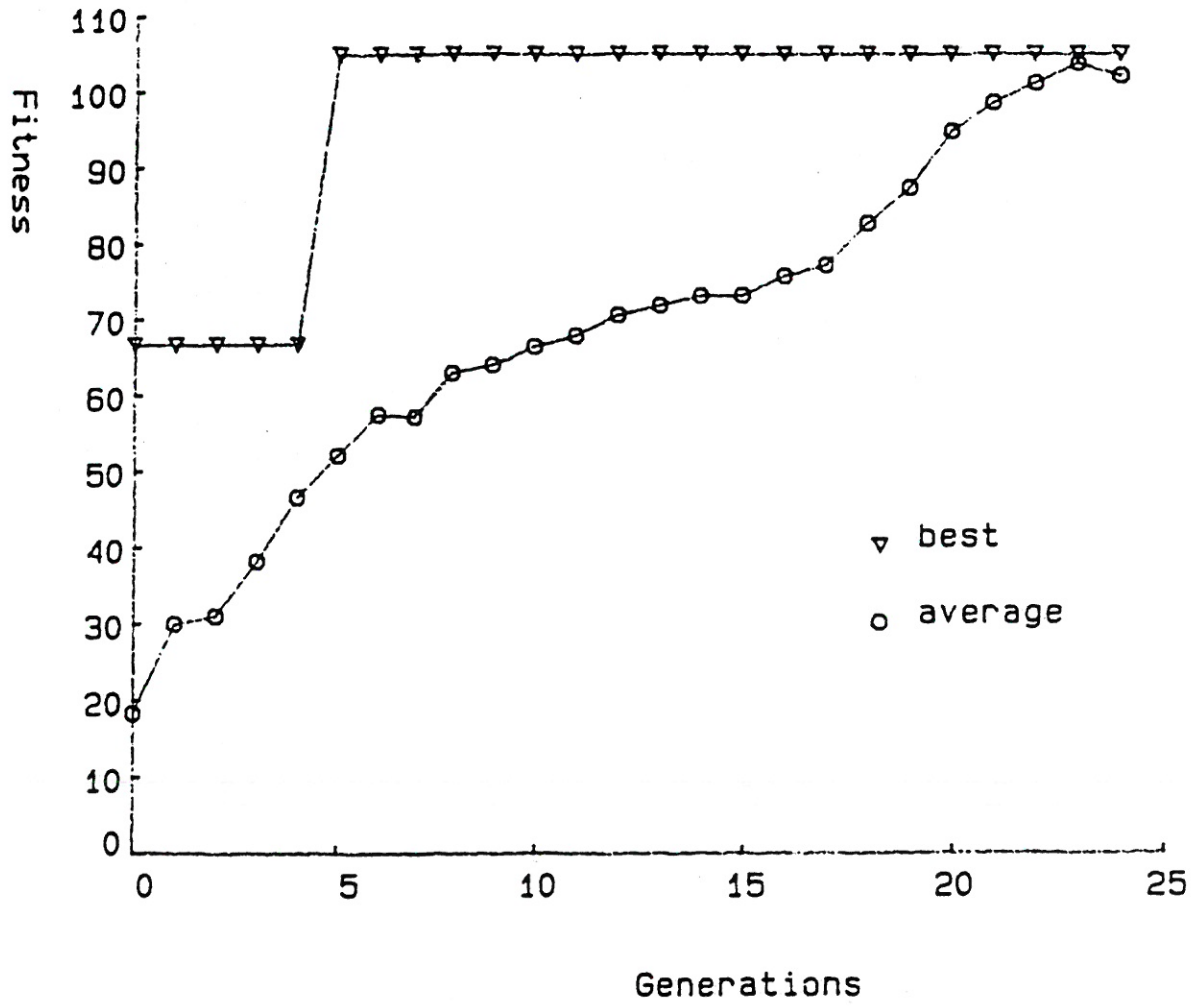


**Figure 8. Largest and average fitness in learning rule No. 3 for the mixing process**

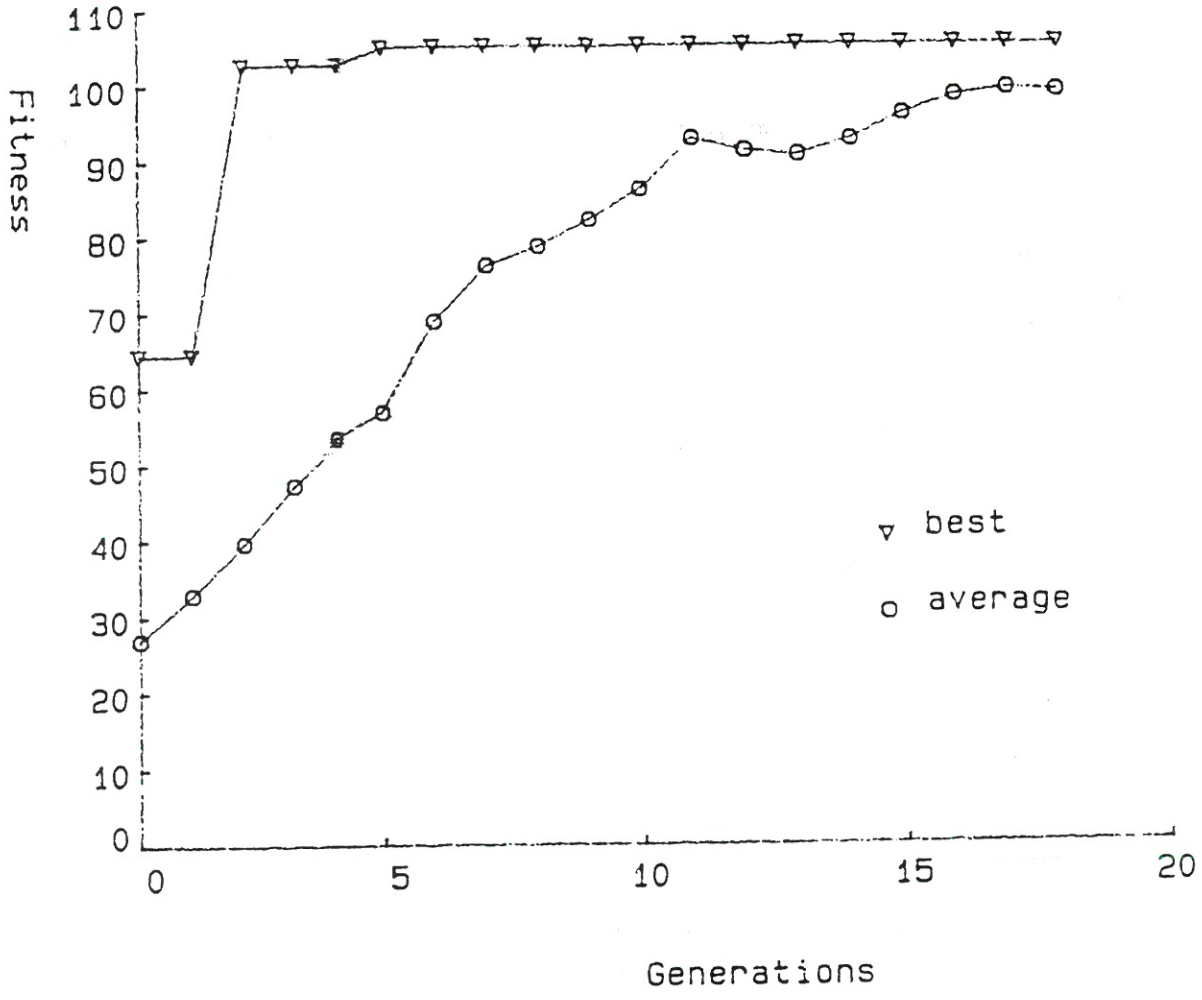


**Figure 9. Largest and average fitness in learning rule No. 4 for the mixing process**

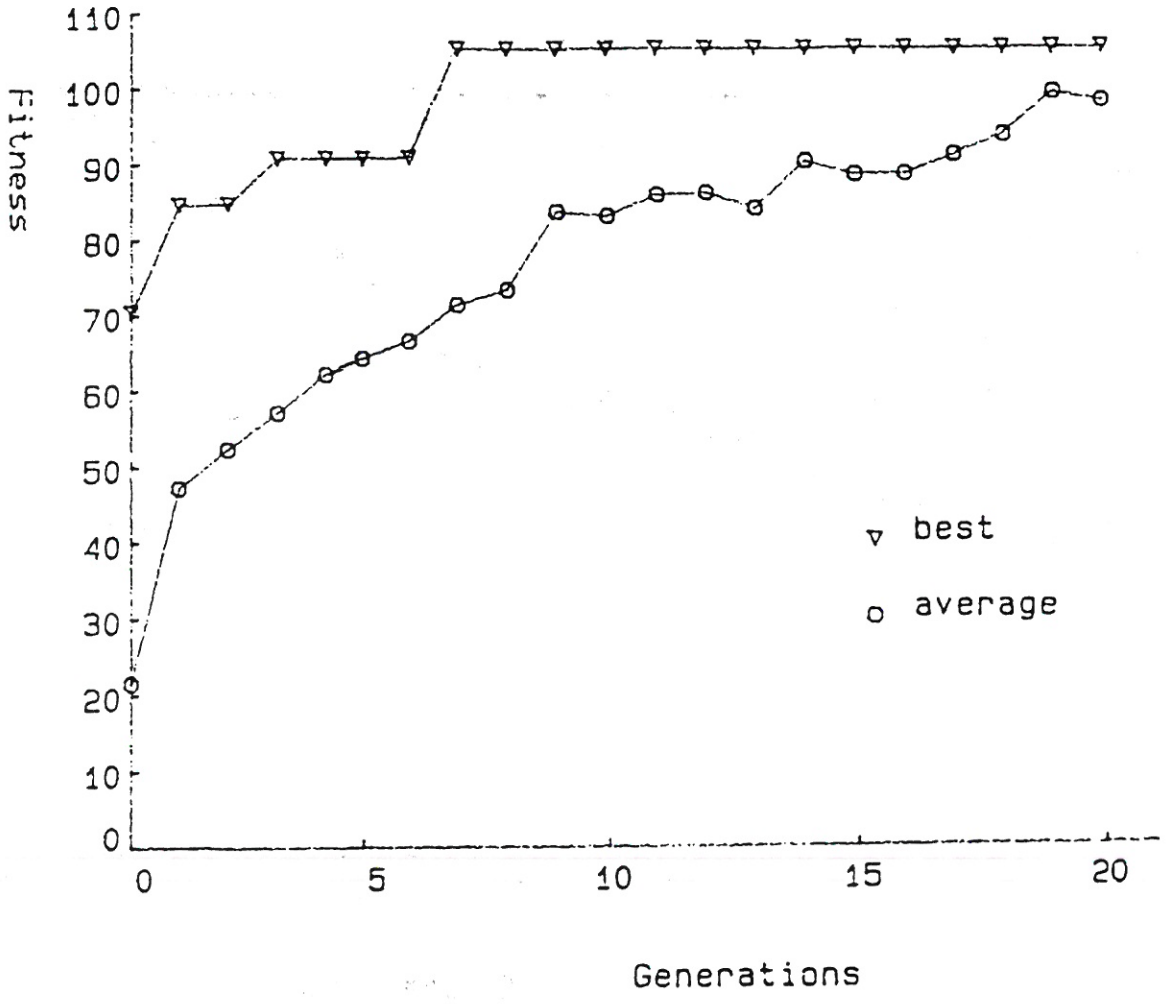


**Figure 10. Largest and average fitness in learning rule No. 5 for the mixing process**

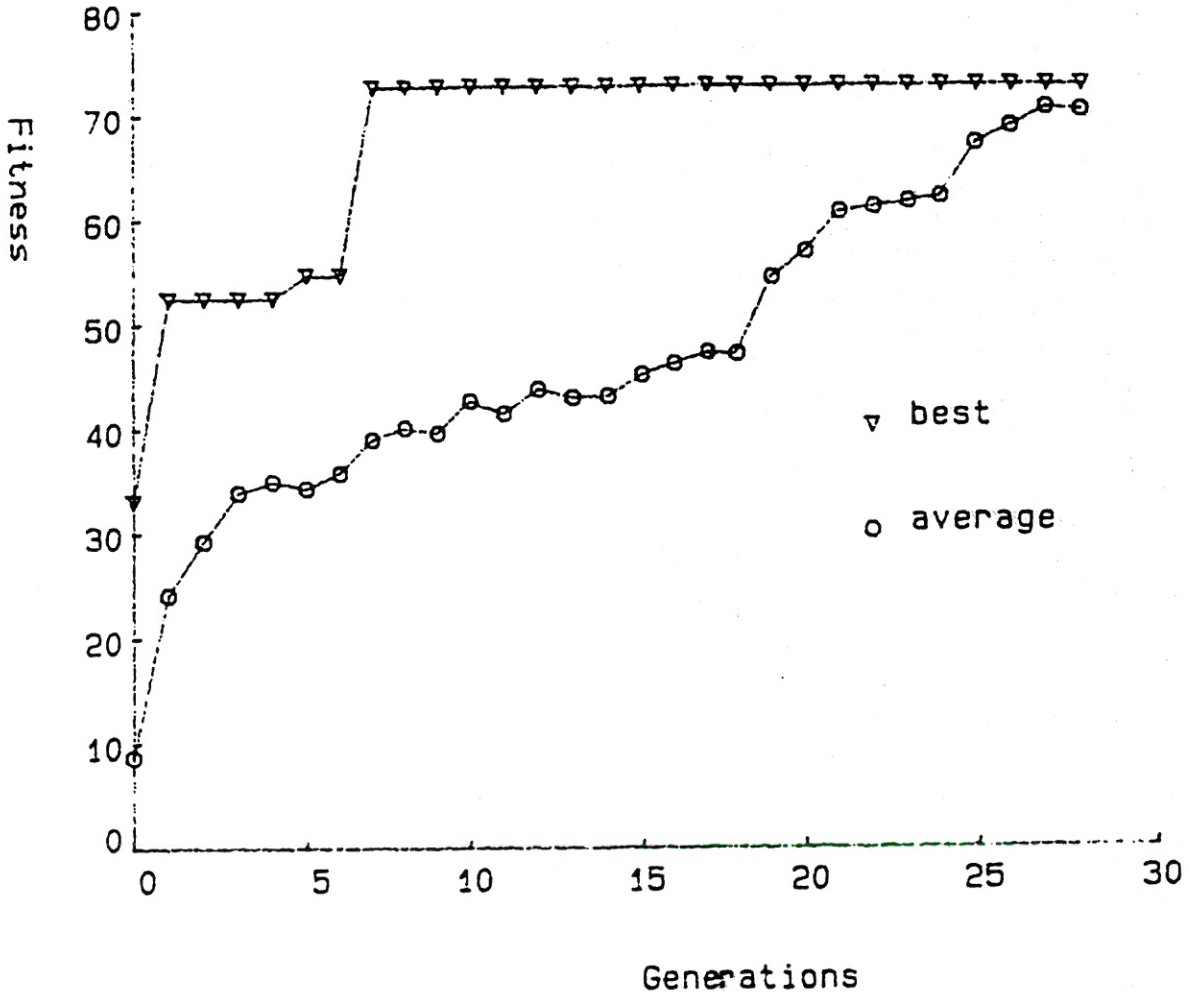


**Figure 11. Largest and average fitness in learning rule No. 6 for the mixing process**

**Table 4**: Fitnesses of the best rule and the second best rule for each fault

| Fault No. | Fitness | NSA | NIA | NOS |
|---|---|---|---|---|
| 1 | 104.8 | 16 | 0 | 1 |
| | 102.6 | 16 | 0 | 3 |
| 2 | 61.5 | 16 | 5 | 3 |
| | 58.2 | 13 | 3 | 2 |
| 3 | 104.8 | 16 | 0 | 4 |
| | 66.6 | 10 | 0 | 3 |
| 4 | 104.8 | 16 | 0 | 1 |
| | 102.6 | 16 | 0 | 3 |
| 5 | 104.8 | 16 | 0 | 4 |
| | 102.6 | 16 | 0 | 3 |
| 6 | 72.6 | 11 | 0 | 3 |
| | 64.4 | 10 | 0 | 2 |

In another experiment, the weighting coefficient $\beta$ was increased to 0.5 while the other coefficients were unchanged.This imposes heavy penalty on incorrect diagnoses. In this case,the best rule learnt for fault 2 is:

**IF**

Level in tank 1 Increase

Cold water flow Decrease

Hot water flow Increase

**THEN**

Cold water control valve fails high

which will miss 7 faults when tested by its corresponding training data (NSA = 9) but it will not produce any incorrect diagnoses when tested by the other training data (NIA = 0).

The result of learning is also affected by the training data.To produce better rules,the training data should cover a large variety of potential faults.This requirement also occurs in neural network based fault diagnosis.

### 3.3.2 Testing the Rules on the Real Process

The learnt rules have been tested on the real process. In an experiment, the following three faults: hot water control valve fails and gives low output, cold water control valve fails and gives low output, and partial blockage of hand valve 1, were separately initiated twice. They were all successfully diagnosed by the corresponding rules and no incorrect diagnosis occurred. Another fault, partial blockage of hand valve 2, was initiated four times and on two occasions the correct diagnosis was obtained whereas on the other two occasions a wrong diagnosis, cold water control valve fails high, was produced. This is due to the fact that the corresponding rule is not perfect and, as described previously, it was caused by the overlappings of training data for fault 2 and fault 6.The rules could be improved by filtering out the bad training data prior to learning.The rules could also be improved by including additional features, such as the magnitudes of deviations in measurements, in their condition parts to increase their resolution.

The proposed approach has also been successfully applied in the learning of diagnostic rules for a CSTR system and details about this can be found in [Zhang and Roberts 92b].

## 4. Discussions and Conclusions

Two approaches for learning diagnostic knowledge are proposed in this paper. Each approach has its advantages and disadvantages.It is shown that the neural network based approach has better generalization properties in that it can tolerate model-plant mismatch to some extent and works well under partially incorrect and partially unavailable information. The knowledge acquired in this approach is stored as the network weights and it is very difficult to comprehend such knowledge. On the other hand, the knowledge acquired in the genetic algorithm based approach is in the form of diagnostic rules and is easy to understand. The genetic algorithm based approach is less robust than the neural network based approach in that the learnt rules are rather crisp and have difficulties in dealing with uncertain information.

The proposed genetic algorithm based learning approach can be extended in future studies to include more complicated form of rules in that the condition parts of the rules shall not only include

the qualitative deviations in on-line measurements but also other features,such as magnitude of the deviation and the shapes in measurement curvatures. A further improvement would be to learn fuzzy diagnostic rules so that uncertain information can be properly handled.

## REFERENCES


GOLDBERG,D.E., **Genetic Algorithms in Search,** OPTIMISATION & MACHINE LEARNING, Addison-Wesley Publishing Company,1989.

GREFENSTETTE,J.J., **Incorporating Problem Specific Knowledge into Genetic Algorithms,** in L.Davis (Ed.) Genetic Algorithms and Simulated Annealing, PITMAN PRESS,London,1987.

HOLLAND,J.H., **Adaptation in Natural and Artificial Systems,**The University of Michigan Press,1975.

LIEPINS, G.E. and VOSE,M.D., **Representational Issues in Genetic Optimisation,** JOURNAL OF EXPERIMENTAL & THEORETICAL ARTIFICIAL INTELLIGENCE, Vol.2, 1990, pp.101-115.

MOOR,R.L. and KRAMER,M.A., **Expert Systems in On-line Process Control,** CHEMICAL CONTROL,III,Asilomer,CA.,1986, pp.839-867.

PRICE,C. and LEE,M., **Application of Deep Knowledge**, ARTIFICIAL INTELLIGENCE IN ENGINEERING,Vol.3,No.1,1988,pp.12-17.

RUMELHART,D.E., HINTON,G.E. and WILLIAMS,R.J., **Learning Internal Representations by Error Propagations,** in D.E.Rumelhart and J.L.McClelland (Eds.) Parallel Distributed Processing: Explor- ations in the Micro-structure of Cognition, Vol.1, Foundations, MIT PRESS, MA.,1986.

S. G. Tzafestas (Ed.) **Knowledge-Based System Diagnosis,Supervision and Control,** PLENUM PRESS,1989.

ZHANG,J.,ROBERTS,P.D. and ELLIS,J.E., **An Application of Expert Systems Techniques to the On-line Control and Fault Diagnosis of a Mixing Process,** JOURNAL OF INTELLIGENT AND ROBOTIC SYSTEMS,1988,pp.209-223.

ZHANG,J., ROBERTS,P.D. and ELLIS,J.E., **Fault Diagnosis of a Mixing Process Using Deep Qualitative Representation of Physical Behaviour,**JOURNAL OF INTELLIGENT AND ROBOTIC SYSTEMS, Vol.3,No.2, 1990,pp.103-115.

ZHANG, J.,ROBERTS,P.D. and ELLIS,J.E., A **Self-learning Fault Diagnosis System,** TRANSACTIONS OF THE INSTITUTE OF MEASUREMENT AND CONTROL,Vol.13, No.1, 1991,pp.29-35.

ZHANG,J. and ROBERTS,P.D., **On-line Process Fault Diagnosis Using Neural Network Techniques,** TRANSACTIONS OF THE INSTITUTE OF MEASUREMENT AND CONTROL,Vol.14,No.4,1992a,pp.179-188.

ZHANG,J. and ROBERTS,P.D., **Use of Genetic Algorithms in the Training of Diagnostic Rules for Process Fault Diagnosis,** KNOWLEDGE-BASED SYSTEMS, Vol.5, No.4,1992b, pp.277-288.