[illegible] C-7

FAULT DETECTION AND DIAGNOSIS USING STATISTICAL CONTROL CHARTS AND ARTIFICIAL NEURAL NETWORKS

R.P. Leger[†], Wm.J. Garland[†], W.F.S. Poehlman[‡]

[†] Department of Engineering Physics, McMaster University, Hamilton, Ontario, Canada, L8S 4M1
[‡] Department of Computer Science & Systems, McMaster University, Hamilton, Ontario, Canada, L8S 4K1

ABSTRACT

In order to operate a successful plant or process, continuous improvement must be made in the areas of safety, quality and reliability. Central to this continuous improvement is the early or proactive detection and correct diagnosis of process faults. This research examines the feasibility of using Cumulative Summation (CUSUM) Control Charts and artificial neural networks together for fault detection and diagnosis (FDD). The proposed FDD strategy was tested on a model of the heat transport system of a CANDU nuclear reactor.

The results of the investigation indicate that a FDD system using CUSUM Control Charts and a Radial Basis Function (RBF) neural network is not only feasible but also of promising potential. The control charts and neural network are linked together by using a characteristic fault signature pattern for each fault which is to be detected and diagnosed. When tested, the system was able to eliminate all false alarms at steady state, promptly detect 6 fault conditions and correctly diagnose 5 out of the 6 faults. The diagnosis for the sixth fault was inconclusive.

Key words: fault detection and diagnosis, cumulative summation control chart, artificial neural network, radial basis function, multi-layer perceptron, fault signature pattern

1. INTRODUCTION

In order to operate a successful plant or process, continuous improvement must be made in the areas of safety, quality and reliability. Improvement in these areas will lead to cost reductions which help make the plant a viable operation in a competitive market. Central to the continuous improvement of safety, quality and reliability is the early or proactive detection and correct diagnosis of process faults. A fault can be defined as a non-permitted deviation of a characteristic property of the process which will cause a certain level of deterioration in the performance of the process. The deviation can be caused by temporary or permanent physical changes. This can be compared to a failure, which can be defined as a complete degradation of performance of the process [1]. Obviously, it is desirable to detect and diagnose faults as soon as possible after their occurrence.

The topic of trend analysis or fault detection and diagnosis (FDD) is not new. Several methods of performing this task have been previously investigated and will be briefly discussed in the Background section of this paper. The paper will then discuss the results of a research project intended to determine the feasibility of a specific FDD strategy. The strategy being investigated was to use cumulative summation (CUSUM) statistical control charts and artificial neural networks to accurately and promptly detect and diagnose process faults.

19th CNS [illegible] Symposium, Hamilton [illegible]

2. BACKGROUND

2.1 Strategies for FDD Systems

There are several different strategies for both fault detection and fault diagnosis. For fault detection, important measurable variables, or unmeasurable variables or parameters which are estimated, are tracked and checked that they are within a certain tolerance of their normal values. If the values are not within the specified tolerance, a fault has been detected. Fault detection methods can be divided into two classes, depending on the presence or absence of an appropriate process model:

(1) techniques using only measurable signals (input and output signals from the process)

(2) techniques using the state (dependent) variables and parameters from a known process model.

It was decided to use statistical control charts for fault detection for two reasons. First, it may be difficult to develop a model of the process which will be accurate enough to be used in the FDD system. Secondly, by using control charts for measured signals, the process operators will work with variables with which they are familiar. This familiarity can increase the chance of acceptance of the FDD system at the operator level.

Regardless of which method of detection is used, one of the main objectives of the detection phase is to concurrently minimize the number of false signals that occur when the process is in normal mode and the time between fault occurrence and a true signal. For statistical control charts, the trade-off occurs in the form of how tight or loose the limits for action are set and is quantified by the Average Run Length (ARL). The ARL of any control chart is defined as the average number of samples taken before the control chart gives an "out of control" signal indicating that an intervention must be made in the process. A good control chart scheme should have a long ARL when the process is running on target. This will minimize the number of false signals. Also, the ARL should be small when the process has shifted away from the target by a substantial amount. This will minimize the response time to a change in the process. The ARL is used to determine the desired balance between false signals and response time to a process change.

The main goal of fault diagnosis should be to determine the root cause of the fault. Depending on the nature of the fault, other diagnosis results could include time of occurrence, size and location. There are three main methods of fault diagnosis; human expertise in the form of the plant operator, knowledge-based systems and pattern recognition techniques. Pattern recognition is an attractive choice for fault diagnosis because this is essentially what an operator does when diagnosing faults. Although the operators may not be familiar with the formal techniques for pattern recognition, the concept of associating faults with patterns of instrumentation readings should be familiar. Again, this familiarity should improve the chances of the FDD system being accepted by the process operators. Classical mathematical pattern recognition techniques include forms of the nearest-neighbour rule and linear discriminate functions. Linear discriminate functions are only useful for classifying groups or classes which are linearly separable and the accuracy of the nearest-neighbour rule when used for this type of application is not acceptable [2]. A relatively new method of pattern recognition involves the use of artificial neural networks. There have been several studies reported on where artificial neural networks have been used for fault diagnosis [2, 3, 4, 5]. The ability of artificial neural networks to construct nonlinear decision boundaries or mappings and accurately generalize in light of noisy or incomplete input data are very desirable qualities.

2.2 CUSUM Control Charts

In an FDD application, the goal is to detect when measured data have shifted away from their normal or target value. Hence, charts for controlling process means were required. Also, in order to provide a proactive FDD system, it is imperative that the fault conditions be detected quickly. This can be done by detecting moderate persistent shifts in mean values promptly. A popular type of chart which is sensitive to moderate persistent changes in mean values is the cumulative sum (CUSUM) control chart. This chart was first introduced by E.S. Page in 1954 [6]. As the name implies, this type of chart cumulates deviations of the sample averages from the target or desired value. Once these cumulations reach either a high or low limit, an out-of-control signal is given. CUSUM charts have a useful property for the FDD application. An estimate of the current process mean can be made when a CUSUM signal occurs. This estimate can be used for the fault diagnosis. A typical CUSUM control chart scheme is shown in Figure 1. As observed, it consists of two charts, a run chart plotting the successive differences between the sample average and target, (x-μ), and the control chart. The parameters shown on the control chart, are defined as follows:

k : the threshold for cumulation, which can be defined as the minimum difference between sample average and target that will cause the cumulation to begin. This value is also sometimes referred to as the allowable slack in the process. Typically, k will be set equal to one half of the deviation from target which is to be detected quickly [7].

SH_i and SL_i : the cumulation terms and are calculated as follows:

$$SH_i = \max\left[0, SH_i^{\#} + SH_{i-1}\right]$$

$$SL_i = \max\left[0, SL_i^{\#} + SL_{i-1}\right]$$

where :

$$SH_i^{\#} = \text{high side cumulation term}$$

$$= \left(x_i - \text{Target}\right) - k$$

$$SL_i^{\#} = \text{low side cumulation term}$$

$$= -\left(x_i - \text{Target}\right) - k = \left(\text{Target} - x_i\right) - k$$

$SH_{(i-1)}$, $SL_{(i-1)}$ are the previous values of SH and SL,
x_i is the average sample reading,
max[a,b] indicates the maximum of a and b.

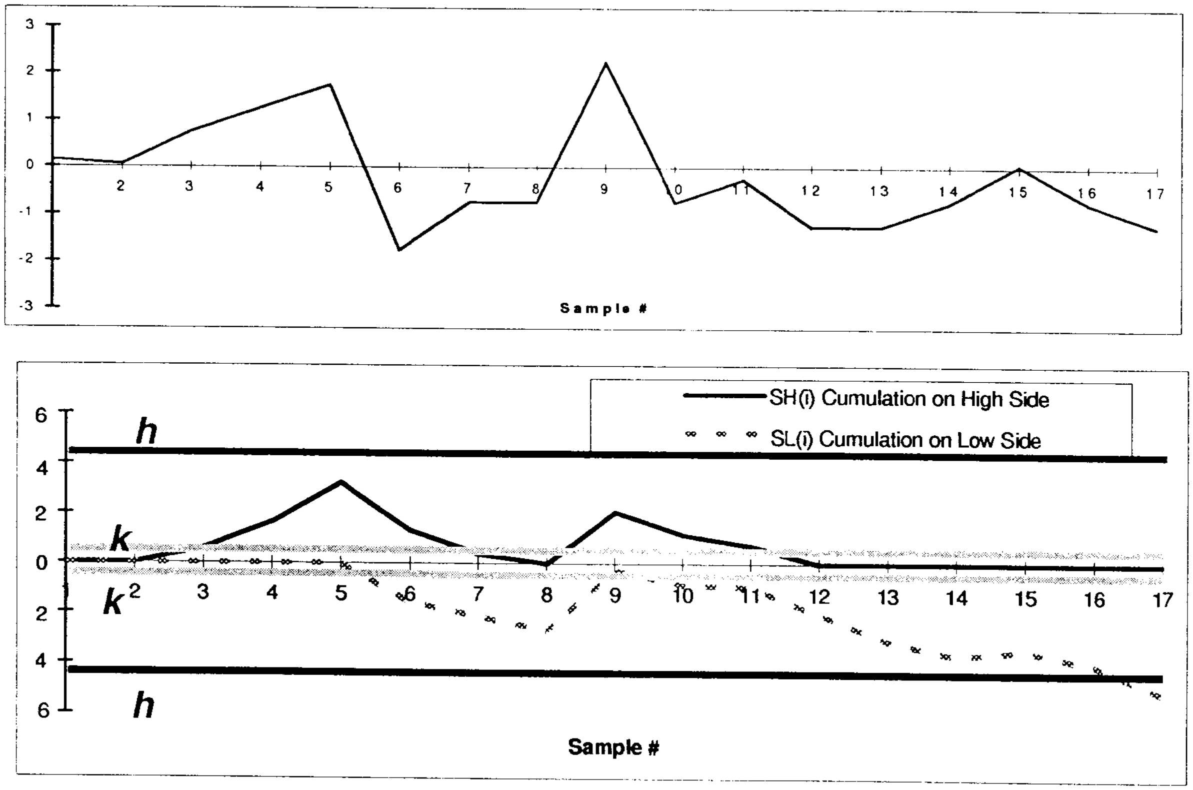


Figure 1 : Typical CUSUM Scheme

h : the control limit. If either SH or SL cumulate above h, intervention in the process is required.
Using the above terms, the estimate of the current process mean can be calculated from the following equations:

<u>**Mean Estimate for Cumulation on High Side** (SH(i) > 0.0)</u>

Current Mean Estimate = Target + k + $SH_{(i)}/N_H$

<u>**Mean Estimate for Cumulation on Low Side** (SL(i) > 0.0)</u>

Current Mean Estimate = Target - k - $SL_{(i)}/N_L$

where:

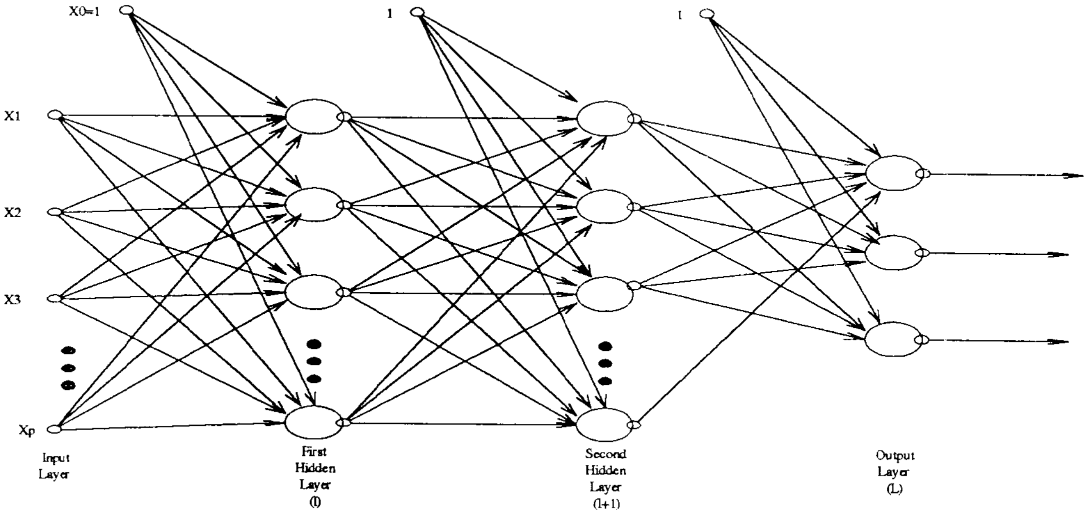
SH_i, SL_i are the respective high and low cumulations
N_H, N_L are the number of samples taken between the start of the cumulation and the alarm

2.3 Artificial Neural Networks

An artificial neural network simulation or neural network can be defined as a massively parallel distributed processor that can store experimental knowledge and make it available for future use [8]. The knowledge is acquired via a learning process and stored in inter-neuron connection strengths known as synaptic weights. The learning process can be completed by one of two methods; supervised or unsupervised. With respect to pattern classification, supervised learning is used when the classification of all training input patterns is known. For example, each training pattern would be classified as a normal mode pattern or a pattern representing one of the fault modes. This is the context of the fault diagnosis problem being investigated and hence this method of learning will be used. There are two basic network architectures used with supervised learning; the multi-layer perceptron (MLP) network and the radial-basis function (RBF) network. Each architecture will be described below.

2.3.1 MLP Architecture

Figure 2 depicts a multi-layer perceptron network.



<u>Figure 2 :</u> **Multi-Layer Perceptron Network (adapted from [8])**

In Figure 2, the large circles in each layer represent neurons and the lines connecting the neurons represent the synaptic weights of the network. The neuron may be represented mathematically by the following equations:

$$v_k = \sum_{j=0}^{p} w_{kj} x_j$$

$$y_k = \varphi(v_k)$$

where:

w_{kj} = input weights to neuron k
x_j = output values from the previous layer
v_k = input to the transfer function of neuron k
φ = transfer function of neuron k
y_k = output from neuron k

Typical transfer functions include linear functions, hard limit functions and log-sigmoid and tan-sigmoid functions. The values of the synaptic weights are determined by training the network using the back-propagation algorithm. The back-propagation algorithm consists of two passes through the network layers. In the forward pass, an input pattern or signal is propagated through the layers of the network while the synaptic weights are held constant. This results in the network output or response to the input pattern. The network response is then subtracted from the desired response and the error is propagated backwards through the network. During this pass the synaptic weights are updated. This process is continued until a pre-determined error goal is reached. The error goal is expressed as the sum of the squared error (SSE), calculated as follows:

$$\text{SSE}= \sum_{i=1}^{\#\text{patterns}} \sum_{j=1}^{\#\text{outputs}} (\text{desired} - \text{actual})_{ij}^{2}$$

A detailed description of the back-propagation algorithm can be found in Haykin [8].

2.3.2 RBF Architecture

Radial basis function networks are another type of network architecture. They use one hidden layer to perform a nonlinear mapping from the input space to a new space. This mapping is constructed using a number of nodes in the hidden layer. The total network consists of three layers, as shown in Figure 3. The first layer is composed of p input nodes, where p is the number of dimensions in the input vector. The hidden layer is composed of the nonlinear processing units or nodes which are connected directly to all input nodes. The output layer is used to linearly combine the output of the hidden nodes, according to the following formula:

$$y_l = \sum_{i=0}^{M} w_{il} G_i + b_l \qquad m=0,1, \ldots M$$

Typically, unnormalized Gaussian functions centered at t_i are used. They are defined as:

$$G\left(\left\|x - t_i\right\|^2\right) = \exp\left(\frac{-\left\|x - t_i\right\|^2}{\sigma_i^2}\right)$$

where:

$\|x - t_i\|$ = the Euclidean distance between a p dimensional input vector, x and the i^{th} center, t_i.
σ_i = width or standard deviation of the Gaussian function centered at t_i.

There are different learning strategies which can be used to determine the synaptic weights of the radial basis function network. These strategies depend on how the free parameters of the network are to be adjusted. The free parameters of the network are the number of centers, locations of the centers, t_i, and width of the centers, σ_I. The simplest learning strategy is to randomly choose the centers of the radial basis functions from the training data and assume they are fixed. The functions can then be normalized by setting the widths of the functions equal to:

$$\sigma = \frac{d}{\sqrt{2M}}$$

where:

d=maximum distance between the chosen centers
M = number of centers

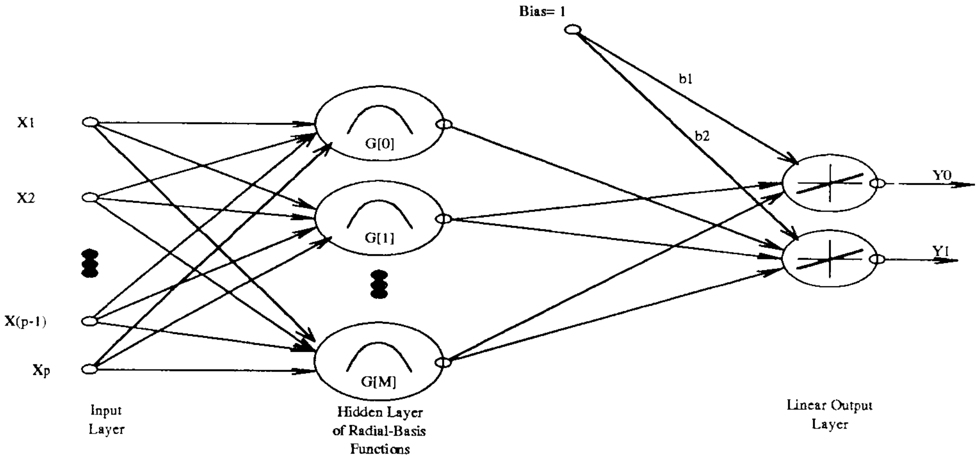


Figure 3 : Radial Basis Function (RBF) Network (adopted from [8])

Finally, the weights of the output layer, w_{ij}, can be calculated directly as follows:

$$\mathbf{w} = \mathbf{G}^{+}\mathbf{d}$$

where:

$\mathbf{d}$ = matrix of the desired output vectors

$\mathbf{G}^{+}$ is the pseudoinverse of the matrix $\mathbf{G}$

A detailed description of the different learning strategies can again be found in Haykin [8].

2.3.3 Choice of Neural Network Architecture

The two architectures described above have their own advantages and limitations. The multi-layer perceptron networks trained using the backpropagation algorithm are characterized by long training times but can generalize well in regions of the input space were there is little training data. The radial basis function networks can be trained quickly but may require a large number of hidden nodes to produce a smooth mapping due to the exponential nature of the Gaussian functions. For the FDD application, the optimum choice for neural network architecture is not immediately clear. Therefore, it was decided to test each architecture for the task of fault diagnosis and compare their performance.

3. GENERIC FDD SYSTEM METHODOLOGY

The methodology for the purposed FDD strategy is shown in Figure 4. It is essentially a three step process; detection of the fault by a CUSUM control chart signal, generation of a pattern to describe the process at the time of fault detection (fault signature pattern), and fault diagnosis via pattern recognition with neural networks. The link between the detection and diagnosis for a fault is the fault signature pattern (FSP). The fault signature pattern should contain as much information as possible about the process at the time of fault occurrence. In order to do this, the estimates of the current process means calculated from the CUSUM schemes were used. As observed from the formulas given in Section 2.2, the current mean can be estimated at any time there is a cumulation, that is, when SH(i) or SL(i) > 0. Therefore, the fault description pattern can be generated using the current estimates of the process means for all analyzers when a signal occurs in one analyser. This method will help capture the dynamics of the process. For example, a certain fault may cause the readings of all analyzers to increase, but some quicker than others. Typically, the first CUSUM signal will be generated by the analyzer which is moving away from its target the quickest. However, when this signal is generated, other readings may have also increased but have not generated signals. Using their CUSUM schemes, their current means can also be estimated using the current cumulations. Hence, the fault description pattern will include information about the dynamics of the entire process.

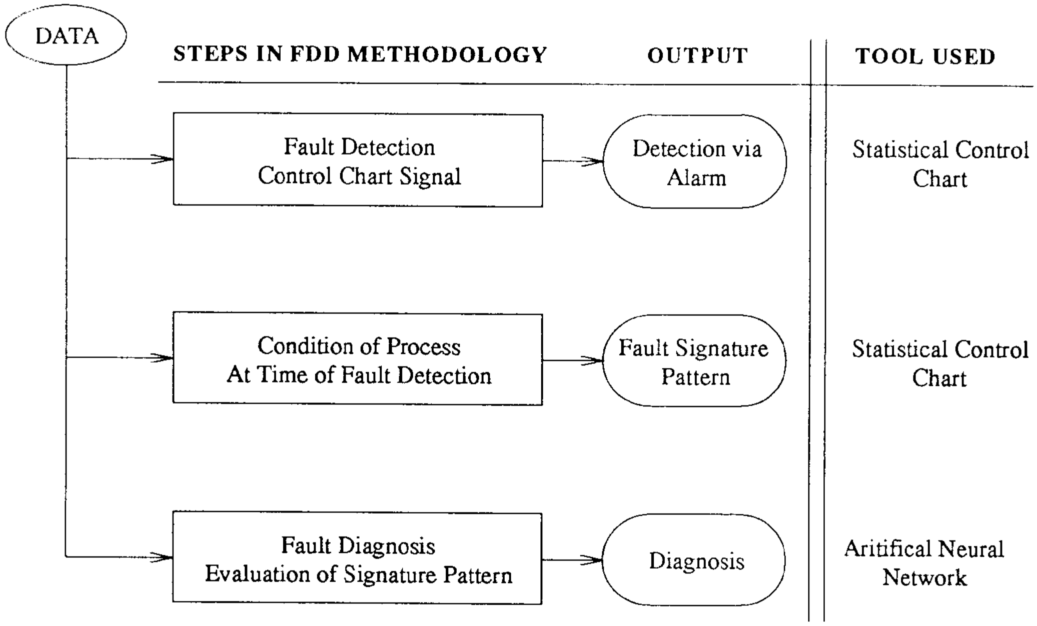


Figure 4 : Proposed FDD Methodology

4. TESTING APPARATUS

In order to determine the feasibility of the purposed FDD methodology, a test bed was required. The test bed used was a model of the heat transport system of a CANDU nuclear reactor. A diagram of the model is shown in Figure 5.

The loop can be divided into two sections, primary side and secondary side, similar to a nuclear reactor. The primary side consists of water being pumped in a figure of eight loop, as shown in Figure 5. As the water flows through the core it is heated by the pipes electrically. The water then flows up through one cooling tower around the U-tube and down through the second cooling tower. It then flows through a pump and into a second core section. As observed from Figure 5, the flow through each of the two core sections is in opposite directions and the loop is symmetric. The primary side flow is identified as the dark solid lines. The instrumentation on the primary side consists of four temperature sensors and two flow sensors. T2 and T4 measure the temperatures at the core inlets and T3 and T5 measure the temperatures at the core outlets. There are two flow orifices and pressure transducers, F1 and F2, located immediately after the two pumps.

The secondary side is defined as the cooling water side. The cooling water enters the bottom of four cooling towers, flows upwards removing heat from the primary side and exits at the top of each tower. The secondary side flow is identified as the light solid lines in Figure 5. The instrumentation on the secondary side consists of five temperature sensors. T1 measures the inlet cooling water temperature. T6 and T7 measure the outlet temperatures from towers 1 and 2 respectively, while T8 and T9 measure the outlet temperatures from towers 4 and 3 respectively. The exact locations of all 11 sensors are shown in Figure 5.

The data acquisition system for the model was set up to collect measurements from the 11 sensors every 0.25 seconds. The measurement data was written to a file on a PC hard disk in binary format. The program created a new data file every five minutes.

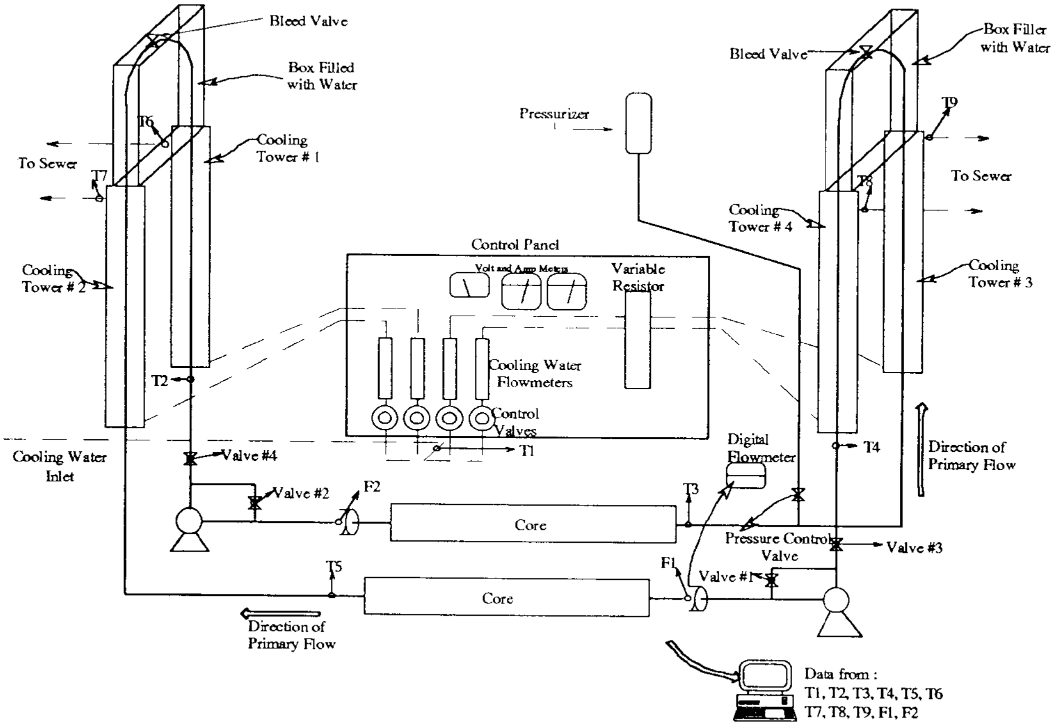


Figure 5 : Diagram of Model Heat Transport System

In order to run the loop at a steady state condition, four operational parameters must be set and monitored. These parameters are the power supplied to the core, the flowrate of the primary side water, the flowrates of the secondary side water through each of the cooling towers and the pressure on the primary side. The power level is controlled by the variable resister and it is monitored by volt and amp meters. The resistor and meters are located on the control panel. The primary side flowrate is controlled by the two flow control valves, labeled valves #3 and #4 in Figure 5. Typically, they are throttled back from their fully open position. In order to try and maintain the symmetry of the loop, both valves are throttled back by the same amount. The primary side flowrate is monitored by the digital flowmeter, connected to flow orifice #1, as shown in Figure 5. The secondary side flowrates are controlled individually for each tower. The control valves are located on the control panel. The individual flowrates are monitored by the four tube flowmeters, also located on the control panel. Finally, the pressure of the loop is set by bleeding any air out of the system and pressurizing the model to 4.0 psig.

5. FDD SYSTEM IMPLEMENTATION METHODOLOGY

The methodology for developing the FDD system for the model was a seven step process. This process is outlined in Figure 6. Each step will be detailed below.

Step 1: In order to develop the FDD system, some fault scenarios which were to be detected and diagnosed had to be chosen. It was decided to attempt to detect and diagnose six different faults. The faults used were:

1) 10% power increase, F1
2) 10% power decrease, F2
3) cooling water shut off to all four cooling towers, F3
4) cooling water shut off to the right hand side cooling towers (towers 1 and 2), F4
5) right hand side re-circulating pump valve open (valve 1), F5
6) left hand side re-circulating pump valve open (valve 2), F6

The locations of the cooling water towers and re-circulation valves are shown in Figure 5.

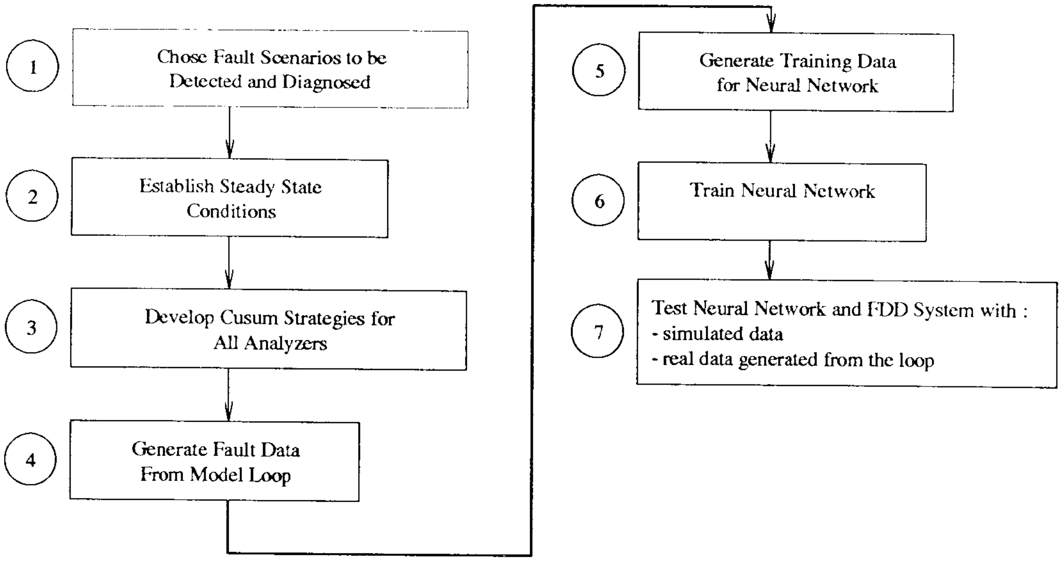

1
Chose Fault Scenarios to be Detected and Diagnosed
2
Establish Steady State Conditions
3
Develop Cusum Strategies for All Analyzers
4
Generate Fault Data From Model Loop
5
Generate Training Data for Neural Network
6
Train Neural Network
7
Test Neural Network and FDD System with :
- simulated data
- real data generated from the loop

Figure 6 : FDD System Development Flowchart

Step 2: The next step in the FDD development was to establish the steady state conditions of the loop. Steady state was required in order to define the aim values to be used in the CUSUM calculations. As previously mentioned, for the model, four operational parameters had to be set to establish steady state conditions.

Step 3: With the steady state conditions established, the CUSUM schemes for each analyzer were determined. In order to calculate the CUSUM scheme for each analyzer, three parameters were required: the aim, the standard deviation, and the magnitude of the change from the aim which was desired to be detected quickly. The aim and standard deviation could be calculated from the data collected in Step 2. It was decided to try and detect shifts of 2 standard deviations from the aims quickly. This size of deviation was chosen so that the signature pattern for each fault would be distinct when a CUSUM alarm occurred. This point will be expanded on in Step 5.

Step 4: The next step was to generate signature patterns of current mean estimates for each of the test fault scenarios described in Step 1. Generally, this data could be acquired by implementing the faults in the actual process, if possible, from historical data, if available, or by simulation. In this case, the model was run at steady state conditions and then a fault was implemented. The signature pattern was defined as the values of the 11 current mean estimates when the first CUSUM signal occurred after the fault was started. It should also be noted that a signature pattern of the steady state condition was defined as the aim values of each analyzer.

Step 5: One approach to fault diagnosis could be to look for the exact signature patterns determined in Step 4 when CUSUM signals occurred. However, it would be highly unlikely that the exact signature pattern would occur again, given the natural variation in the analyzer measurements. This is one driving force for the use of neural networks for fault diagnosis: their ability to generalize with noisy data. Therefore, Step 5 was to create training data for the neural network. This was done by adding normally distributed or Gaussian noise to each analyzer reading for each signature pattern. The noise had a mean of 0 and standard deviation equal the standard deviation of each analyzer at steady state, as determined in Step 3. The procedure of adding noise with a variation of one standard deviation and attempting to detect shifts of two standard deviations was designed to generate training data distinct enough to allow the neural network to learn the conditions for each fault. This method allows training data to be generated from one occurrence of the fault in the actual system. This addresses the problem of having a vast amount of data for steady state conditions and very little data for fault conditions.

Step 6: Different neural network architectures were trained using the training data generated in Step 5. Both MLP and RBF networks were trained and tested.

Step 7: The entire FDD system, consisting of the CUSUM schemes and neural networks developed in the previous steps, was tested using data from the model.

6. RESULTS

6.1 Experimental Design

The data were collected from the model in a series of two experiments. One series was designed to collect development data and the second series was collected to test the final system. Each experiment consisted of collecting 0.5 hours of steady state data and up to 0.5 hours of fault data. Each series of experiments collected data for all fault conditions. The data acquisition rate of 0.25 samples per second was considered excessive for this application when the nature of the faults to be detected was taken into account. Therefore, it was decided to develop the FDD system based on a new data point from the model every 10 seconds. This data point was calculated from the average of the 40 samples collected by the data acquisition system during the 10 second period.

6.2 CUSUM Schemes

Three quantities are required to design a CUSUM scheme; the target values for each analyser, μ , the standard deviation for each analyser, σ , and the shift in mean which is desired to be detected quickly, Δ . The development of CUSUM charts involves two basic assumptions; that the individual observations are independent (no autocorrelation) and approximately normally distributed. Nonnormality is a more serious problem when individual observations are used by virtue of the Central Limit Theorem. This theorem states that as the size of a subgroup of samples increases, the distribution of the subgroup averages will approach a normal distribution. This is regardless of the distribution of the individual measurements. However, when the individual measurements are used for the control chart, their distribution must be checked for normality. One method is to plot and inspect the histograms of the data.

The data used to develop the CUSUM schemes was taken from 2 hours of the steady state data collected during the first set of experiments. This resulted in 720 data points. When histograms of this data for each analyser were plotted, the distributions appeared to be approximately normal, although some are better than others. This result would be influence by the fact that each "individual" observation is actually the average of 40 measurements. Based on the histograms, it was decided that this data was acceptable for calculating the control chart schemes. The means of the steady state data were calculated for each analyser. These values are shown in Table 1.

Analyser	Steady State Operating Point
T1	10.24°C
T2	42.75°C
T3	50.17°C
T4	41.75°C
T5	48.84°C
T6	26.20°C
T7	28.78°C
T8	26.23°C
T9	26.02°C
F1	7.46 V
F2	7.15 V

Table 1 : Steady State Signature Pattern

The values in the table above were used as the steady state signature pattern.

The standard deviation for each analyser, σ , was estimated from the steady state data points used to calculate the target values above. The standard deviations were estimated using the standard formula :

$$\sigma = \frac{s}{c_4}$$

where: $s = \sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}{n-1}}$

For samples sizes greater than 50, (n>50), c_4 is approximately 1.0. The values of the estimates of σ for each analyser are shown in Table 2.

Analyser	Estimate of σ
T1	0.07°C
T2	0.35°C
T3	0.47°C
T4	0.33°C
T5	0.42°C
T6	0.27°C
T7	0.29°C
T8	0.24°C
T9	0.26°C
F1	0.03 V
F2	0.05 V

Table 2 : Standard Deviation Estimates

Finally, it was decided to set Δ , the shift in mean which is desired to be detected quickly, equal to $2 * \sigma$. This was done to allow the neural network training data to be distinct. Given the above data, the CUSUM schemes were set up according the four steps below:

1. threshold = $k = \Delta / 2 = \sigma$
2. $k^* = k / \sigma = 1.0$
3. chose $h^* = 3.5$ to give ARL(0)=2670, ARL(Δ)=4.25
4. control limit = $h = h^* * \sigma$

The average run lengths given above were obtained from tables found in Marquardt [9]. Given that data points were available every 10 seconds, one could expect a false alarm when the process is exactly on target every 7.4 hours and a shift of Δ should be detected in 42.5 seconds.

The above schemes were tested on the four days of steady state data to determine how many false alarms would occur. Considering there was 2 hours of steady state data, at most, one false alarm for each analyser would be expected. When the schemes were applied to the data, 300 false alarms occurred. The source of the false alarms was determined to be the day to day range of the mean values for each analyser. This in effect caused a strong auto correlation in the data which violated one of the basic assumptions of the data to be tracked with CUSUM charts. To compensate for this auto correlation, the threshold limits were calculated by adding k to the maximum daily mean and subtracting k from the minimum daily mean. This in effect widened the threshold for cumulation and therefore increased the value of Δ for each analyser. The actual control limits were also doubled to $7.0 * \sigma$. This reduced the number of false alarms to 4 which was an acceptable values. The final CUSUM schemes are summarized in Table 3.

Analyser	Aim	Δ	Upper Threshold Limit	Lower Threshold Limit	Control Limit (=7.0*σ)
T1 (°C)	10.24	2.89 σ	10.33	10.13	0.48
T2 (°C)	42.75	3.47 σ	43.41	42.40	2.43
T3 (°C)	50.17	3.46 σ	51.03	49.40	3.31
T4 (°C)	41.75	3.42 σ	42.38	41.25	2.30
T5 (°C)	48.84	3.39 σ	49.63	48.19	2.97
T6 (°C)	26.20	3.99 σ	26.82	25.76	1.87
T7 (°C)	28.78	4.06 σ	29.37	28.17	2.07
T8 (°C)	26.23	2.85 σ	26.56	25.88	1.69
T9 (°C)	26.02	4.10 σ	26.60	25.54	1.81
F1 (V)	7.46	2.31 σ	7.50	7.42	0.23
F2 (V)	7.15	3.46 σ	7.22	7.03	0.39

Table 3 : **Final CUSUM Strategies**

6.3 Fault Signature Patterns

The fault signature patterns were defined as the first pattern associated with the first realistic CUSUM alarm after each fault was initiated in the model. The patterns were taken from the data collected in the first set of experiments. A summary of the fault signature patterns and the times required for detection is given in Table 4.

Analyser	F1	F2	F3	F4	F5	F6
T1(°C)	10.24	10.94	10.90	10.54	10.24	10.15
T2(°C)	43.24	42.75	43.53	43.50	43.16	42.75
T3(°C)	51.27	49.61	50.64	50.77	50.78	50.65
T4(°C)	42.19	41.74	42.49	41.75	42.14	42.15
T5(°C)	49.53	48.28	49.33	48.84	49.40	49.30
T6(°C)	26.20	26.20	26.60	26.77	25.89	26.20
T7(°C)	28.46	28.78	28.78	29.14	29.11	29.21
T8(°C)	26.44	26.23	26.84	26.23	26.23	26.23
T9(°C)	26.02	26.02	26.51	26.02	26.02	26.02
F1(V)	7.46	7.46	7.50	7.42	5.78	6.24
F2(V)	7.15	7.14	7.16	7.15	5.66	5.95
Detection Time	3.7 min.	7.2 min.	1.8 min.	2.0 min.	0.5 min.	0.5 min.

Table 4 : **Fault Signature Patterns**

6.4 Neural Network Design

Both the RBF and MLP networks were designed using MATLAB Neural Network Toolbox, Version 4.2 by MathWorks Inc. Training data for the neural networks was generated using the procedure described in Step 5 of Section 5 and the steady state and fault signature patterns described above. The noise used had a mean of 0 and standard deviation equal to those shown in Table 2, for each analyser respectively. It was normally distributed because that distribution was assumed for each analyser. For this application, the training set consisted of 350 patterns, fifty training patterns for each of the seven states were generated. A testing set was also created, consisting of 5 patterns from each state for a total of 35 testing patterns. The output layer for each network consisted of 7 nodes, one node representing each of the seven states. The output for each pattern contained a 1 in the proper location and a 0 elsewhere. For example, the output for SS would be $[1,0,0,0,0,0,0]^T$ while the output for F3 would be $[0,0,0,1,0,0,0]^T$. For this application, the classification of a pattern was determined by the neuron with the largest output in this output layer.

6.4.1 MLP Network Design

The initial architecture of the MLP network was chosen to be one hidden layer consisting of 10 neurons and an output layer consisting of 7 neurons. The log-sigmoid transfer function was used for all neurons because of it's 0 to 1 output range. When this network was trained, the SSE stopped decreasing after 2000 epochs and was not at an acceptably low level. Several changes were made to the training procedure, including changing the momentum constant, changing the initial weights and increasing the number of neurons in the hidden layer to 15. However, the problem was only overcome when the input patterns were normalized to the SS signature pattern. This provided input patterns containing values all close to 1. After this change, the network was trained for 100000 epochs and the SSE was reduced to 18.4. The classification accuracy for the training data was 96.2% while the classification accuracy for the 35 testing patterns was 91.4%.

6.4.2 RBF Network Design

The free parameters of the radial basis function network are the location, number and width of the centers. In the MATLAB training function for RBF networks, the maximum number of centers or neurons, width of the centers and the error goal is specified by the user. The function then starts with one center and calculates the sum of the squared error. The locations of the subsequent centers are chosen from the training data. Centers are added until the error goal is reached or the maximum number is used. The location of each new center is based on which training data point will reduce the squared error the most.

The RBF network was designed in two stages. First, the classification accuracy of the network was calculated for different numbers of centers for both the training and testing data. The spread of the centers was set to the default value of one. The optimum number of centers was found to be 105. Then the number of centers was held at 105 and the width of the centers was varied. The width was varied from 0.1 to 5.0. The optimum width was found to be 1.14. Using this architecture of 105 neurons with spreads of 1.14, the classification accuracy of the training and testing data was 98.3% and 94.3% respectively.

6.5 FDD System Performance

The performance of the FDD system was analyzed from two different perspectives. First, the performance was analysed with respect to the number of false alarms generated during normal operation mode when the two tools, CUSUM charts and neural networks, were used in combination and individually. Secondly, the performance of the system with respect to fault detection and diagnosis during faulty operating modes was analysed. Both performance measures will be discussed below. The performance of the FDD system was evaluated using the data from the second set of experiments.

To evaluate the performance with respect to false alarms, the number of alarms which occurred during steady state for each experiment was determined. The number of alarms was calculated using the CUSUM charts and neural networks individually and then together. Before this evaluation could be completed, guidelines for determining when an alarm was present needed to be set for all three cases. For the CUSUM chart alone, this was straight forward. A false alarm would be considered present if there was a CUSUM alarm. For the neural networks alone, some guidelines needed to be set. During the training of the networks, the classification results were determined from the largest output from the neurons in the output layer. For this performance evaluation, it was decided to add the additional guideline that the value of the winning output had to be greater than 0.5. If all outputs were less than 0.5 the output of the neural network would be considered inconclusive. That is, the system was not considered at steady state but no conclusive fault diagnosis could be made. This was done to prevent a conclusion from being made when all the outputs from the network were small. Finally, when both the CUSUM charts and neural networks were used together, an false alarm was considered present only if both the CUSUM chart "and" neural network indicated so.

Table 5 shows the number of false alarms which occurred during the steady state operation. It is important to note that when the CUSUM charts and RBF network were used in combination, no false

	Individual Applications			Combination Applications	
	CUSUM	RBF	MLP	CUSUM & RBF	CUSUM & MLP
Total	44	17	68	**0**	11

Table 5 : False Alarms During Steady State Data

alarms were generated by the FDD system at steady state. This result indicates that using the two tools in combination to confirm each others results is more effective than using the tools separately.

The performance of the FDD system in terms of actual faulty process conditions was also evaluated using the tools individually and in combination. For the diagnosis phase, it was decided to use the first fault indication, other than an inclusive output, as the fault diagnosis. If the output of the network was inconclusive and a CUSUM alarm was present, the process was considered not at steady state but no conclusive diagnosis could be made. The results of the fault detection and diagnosis are shown in Table 6.

The following general observations can be made from Table 6. First, as indicated in the table, there is obviously no diagnosis when CUSUM charts are used alone. The next point to note is associated with the diagnosis of F5 with both the RBF and MLP networks. These are highlighted by F5* in Table 6. In these cases, F6 was actually diagnosed 20 seconds after the fault was initiated and F5 was correctly diagnosed at the 30 second mark. This was expected because the flowrates for the F6 FSP were larger than F5, while all other measurements were approximately the same. Hence, as the flowrates dropped, F6 would first be identified, followed by F5 as the flowrates reached the steady state values for F5. This was considered a minor fact considering that F5 was correctly diagnosed 30 seconds into the fault and that the diagnosis then remained correct for the duration of the fault.

There was one incorrect fault diagnosis in the testing data. This occurred in the diagnosis of F5 with the MLP network. As observed in Table 6, the MLP network diagnosed F5 as F6. This may have resulted from the similarity in the two FSP's and may be corrected with more network training. It is interesting to note that, except for the first 20 seconds, the RBF network is able to correctly diagnosis both F5 and F6. There was one inclusive fault diagnosis in the testing data, which occurred for F3 with the RBF network. After F3 was implemented, the results of the network was inclusive, that is, no outputs were greater that 0.5. However, the RBF network, along with the CUSUM charts, did indicate that the process is no longer at steady state. As observed in Table 6, the MLP network did correctly diagnosis F3 for the testing data. The inconclusive diagnosis by the RBF network appears to be caused by a low drift in T8 prior to the fault initiation. This drift caused some CUSUM alarms during the steady state operation but did not effect to output of the RBF network. When the steady state value for T8 was substituted in the raw test data, F3 was correctly diagnosed by the RBF network. Finally, it should be noted that the detection times for all faults ranged from 20 seconds for F6 to 5.0 minutes for F2.

	Individual Applications			Combination Applications	
Actual Fault	**CUSUM**	**RBF**	**MLP**	**CUSUM & RBF**	**CUSUM & MLP**
	Diagnosis	Diagnosis	Diagnosis	Diagnosis	Diagnosis
F1	(N/A)	F1	F1	F1	F1
F2	(N/A)	F2	F2	F2	F2
F3	(N/A)	INCLU	F3	INCLU	F3
F4	(N/A)	F4	F4	F4	F4
F5	(N/A)	F5*	F6	F5*	F6
F6	(N/A)	F6	F6	F6	F6

* see text above

Table 6 : Detection and Diagnosis Performance Results

The two networks tested in this study had distinct and different responses as the faults progress. This is shown in Figure 7, which shows the networks' response to F1 from the testing data. The signal from the RBF network dies away as the fault progresses while the signal from the MLP network remains prominent. This trend was observed in other faults as well. The RBF network appears closer to reality as it shows the fault as a transient whereas the MLP network seems to show the fault as a new steady state.

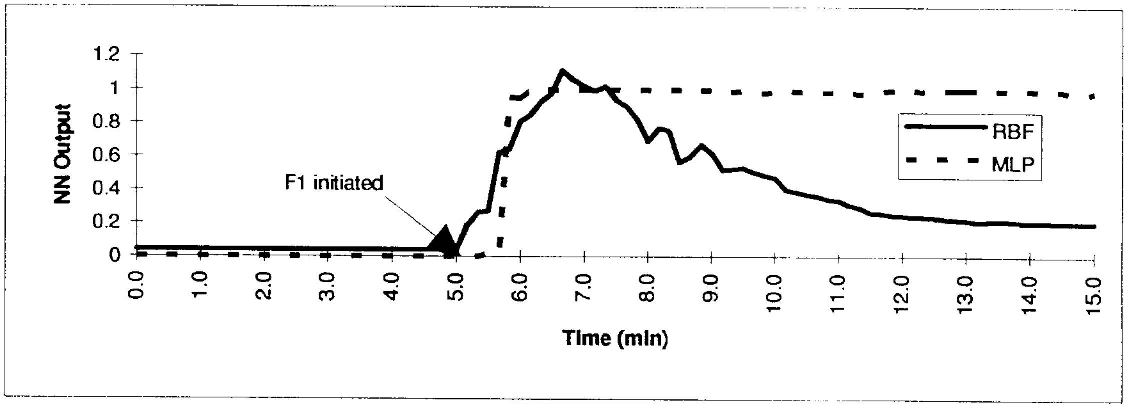


Figure 7 : Neural Networks Response to F1

7. CONCLUSIONS

Based on the previous discussion, the following conclusions can be made regarding the development of the FDD system:

(1) Using CUSUM charts and a RBF neural network in combination and linked using fault signature patterns for specific faults is a feasible FDD system. When the two tools are used together for fault detection, false alarms while the process is at steady state can be eliminated. Also, using the tools in combination will not adversely effect the systems' performance when actual faults are present.

(2) The RBF network appears to be the best choice of neural network for this application, considering its' performance for both steady state and fault conditions. In terms of steady state conditions, the RBF network, in combination with the CUSUM charts, was able to eliminate all false alarms. The combination of the MLP network and CUSUM charts could not eliminate all false alarms at steady state. Both the RBF network and MLP network, along with the CUSUM charts, could promptly detected all fault conditions. In terms of diagnosis, the RBF network made one inconclusive diagnosis during testing while the MLP network made one erroneous diagnosis. Although both results are not desirable, an inconclusive diagnosis would be preferred to an erroneous one. Finally, as the faults progress, the RBF network gives a more realistic picture of the situation, showing the fault as a transient as opposed to a new steady state, which the MLP network predicts. Once the RBF no longer recognizes the fault, the output is inconclusive.

(3) In practical applications, widening the thresholds for CUSUM schemes is a feasible technique for compensating for auto-correlation in the data. In this application, this technique did not have an adverse effect on the results.

8.0 REFERENCES

1. Himmelbleu, *"Fault Detection and Diagnosis in Chemical and Petrochemical Processes"*, University of Texas, Austin, Texas, U.S.A., Elsevier Scientific Publishing Company, 1978.

2. Sorsa and H.N. Koivo, *"Application of Aritificial Neural Networks in Process Fault Diagnosis"*, Automatica, Vol. 29, No. 4, 1993, pp. 843-849.

3. J.C. Hoskins and D.M. Himmelblau, *"Artificial Neural Network Models of Knowledge Representation in Chemical Engineering"*, Comput. Chem. Eng., Vol. 12, No. 9/10, 1988, pp. 881-890.

4. K. Watanabe, I. Matsuura, M. Abe, M. Kubota, and D.M. Himmelblau, *"Incipient Fault Diagnosis of Chemical Processes via Aritificial Neural Networks"*, AIChE Journal, Vol. 35, No. 11, November 1989, pp. 18031812.

5. V. Venkatasubramanian and K. Chan, *"A Neural Network Methodology for Process Fault Diagnosis"*, AIChE Journal, Vol. 35, No. 12, December 1989, pp. 1993-2002.

6. E.S. Page, *"Continuous Inspection Schemes"*, Biometrika, Vol. 41, 1954, pp. 100-115.

7. J.M. Lucus, *"The Design and Use of V-Mask Control Schemes"*, Journal of Quality Technology, Vol. 8, No. 1, January 1976, pp. 1-12.

8. S. Haykin, *"Neural Networks, A Comprehensive Foundation"*, Macmillan College Publishing Company Inc., 1994.

21. Marquardt, *"PQM - Product Quality Management"*, E.I. du Pont de Nemours & Co. Engineering Department, Applied Statistics Group, Wilmington, Delaware, 19898. 1988 Edition.