

FINAL YEAR PROJECT

**DEVELOPMENT OF FAULT DETECTION MODEL FOR INSTRUMENT IN  
GAS TRANSPORTATION SYSTEM**

By

**WAN AIZAM SYAHIRA BINTI WAN SOHAIMI**

**14295**

Dissertation submitted in partial fulfillment of  
the requirements for the  
Bachelor of Engineering (Hons)  
(Electrical and Electronic Engineering)

SEPTEMBER 2014

Universiti Teknologi PETRONAS  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan

## **CERTIFICATION OF APPROVAL**

### **DEVELOPMENT OF FAULT DETECTION MODEL FOR INSTRUMENT IN GAS TRANSPORTATION SYSTEM**

By

Wan Aizam Syahira binti Wan Sohaimi

A project dissertation submitted to the  
Electrical and Electronic Engineering Programme  
Universiti Teknologi PETRONAS  
in partial fulfillment of  
the requirements for the  
Bachelor of Engineering (Hons)  
(Electrical and Electronic Engineering)

Approved by:

---

(AP Dr Rosdiazli bin Ibrahim)

Project Supervisor

Universiti Teknologi PETRONAS  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan

## **CERTIFICATION OF ORIGINALITY**

This is to certify that I am responsible for the work submitted in this project, that the original work is on my own except as specified in the references and acknowledgements.

---

WAN AIZAM SYAHIRA BINTI WAN SOHAIMI

## **ABSTRACT**

The increasing demands for a better system performance and quality have led to the development of the automation of technical process. However, the system dependability and reliability to overcome the fault in the system is the issues that being addressed regarding the design of control and automation system. There are many research regarding the Fault Detection and Diagnosis (FDD) that has been conducted to enhance the reliability of the system in the plant process. Artificial intelligence such as neural network is one of the powerful tools in detecting and diagnosis the fault occurred. The ability of the neural networks to learn from the experience or past data has shown a great impact in the fault detection efficiency. Apart from that, statistical method which is based on the past data also is being proven to improve the accuracy of the fault detection in any fields. Thus, this preliminary report contains five chapters which are the introduction, literature review, methodology, result and discussion and finally conclusion. The first chapter of the report will give the overview of the project along with the problem statements. Based on the problem statements, the objectives of the project are being derived. The main objective is to develop the best fault detection model to be implemented in the gas transportation system. Next, the second chapter is to review the past research and study in this area in order to synthesize and obtaining information on how the fault detection can be done by using statistical and artificial intelligence methods. On the other hand, the third chapter will outline the project planning and activities throughout the time frame given. The methodology on how the project will be executed also being explained in this chapter. Next, in Chapter 4, the findings and results will be discussed based on the methodology outlined in the previous chapter. Finally, Chapter 5 will summarize and conclude the future planning of the project.

## **ACKNOWLEDGEMENT**

I would like to thank Allah the Almighty for giving me courage and strength to complete my Final Year Project (FYP). I would also like to place on record a deep sense of gratitude to AP Dr Rosdiazli bin Ibrahim who was not just the supervisor but also a mentor who had given me a helpful and useful guidance throughout this two semesters. Not to forget, the author also would like to extend the thanks to Ms. Nurfatihah for her generous guidance as well as for providing information and knowledge for me during completing this project. I am also thankful to Electrical and Electronic Engineering Department of Universiti Teknologi PETRONAS for giving the opportunity for final year student to enhance and develop problem solving skills as well as technical skills. Last but not least, thank you also to all my freinds and family for their contuinuous support for me until I finished this project. In a nutshell, the learning and experience gained throughout the two semesters of final year study is very precious and meaningful to prepare me for the real challenging engineering environment.

## Contents

INTRODUCTION .....	1
1.1 Background .....	1
1.2 Problem Statement .....	2
1.3 Objectives .....	2
1.3 Scope of Study.....	3
1.4 Tools and Software Required .....	3
LITERATURE REVIEW.....	4
2.1 Definition of Fault .....	4
2.2 Types of Fault.....	4
2.3 Fault Detection and Diagnosis .....	4
2.3.1 Principal Component Analysis.....	6
2.3.1.1 T-square and SPE Chart.....	7
2.3.1.2 Example of application on PCA techniques.....	8
2.3.2 Artificial Neural Network .....	10
2.3.2.1 Experiment on ANN technique .....	11
2.4 Summary of literature review .....	13
METHODOLOGY .....	14
3.1 Project Flowchart .....	14
3.2 System Block Diagram.....	16
3.3 Project Activities .....	17
3.4 Gantt Chart (FYP1) .....	18
3.5 Gantt Chart (FYP2) .....	19
RESULT AND DISCUSSION .....	20
4.1 Fault Definition .....	20
4.2 Data Collection and Analysis .....	21
4.3 Principal Component Analysis .....	24
4.4 Artificial Neural Network Algorithm.....	29
4.5 Artificial Neural Network for Fault Classification.....	30
4.5.1 Network training and simulation for faulty data.....	31
4.5.2 Network training and simulation with healthy and faulty data .....	33
4.5.3 Simulation with new data set .....	34
4.6 Fault Detection for Drift Data .....	35

4.6.1	Drift fault detection with normal data.....	35
4.6.2	Drift fault detection with manipulation of drift data.....	37
CONCLUSION.....		39
5.1	Conclusion.....	39
5.2	Recommendation.....	39
REFERENCES.....		40
APPENDICES .....		43
APPENDIX I – Fault Detection Source Code.....		43
APPENDIX II – Fault Classification Source Code .....		50

## List of Figures

Figure 1: Auto-billing correction system .....	3
Figure 2: Fault detection and diagnosis category.....	5
Figure 3: Data-driven approach .....	5
Figure 4: Fault detection and isolation process.....	6
Figure 5: PCA methodology .....	7
Figure 6: SPE chart .....	7
Figure 7: T-square chart .....	7
Figure 8: Residuals plot for drift detection .....	8
Figure 9: Tennessee Eastman process flow sheet.....	9
Figure 10 : Parallel analysis plot for TE process data.....	9
Figure 11: Neural network model .....	11
Figure 12: A backpropagation neural network .....	11
Figure 13: Multilayer ANN for fault classification.....	12
Figure 14: Fault detection and classification block diagram .....	16
Figure 15: Data plot .....	22
Figure 16: Data analysis for each parameter.....	22
Figure 17: Principal Component plot.....	24
Figure 18: Score plot.....	25
Figure 19: New featured data from PCA modelling for healthy data .....	26
Figure 20: New featured data from PCA modelling for faulty data .....	26
Figure 21: Residual generated for each parameter.....	27
Figure 22: SPE Chart .....	28
Figure 23: T-square chart .....	28
Figure 24: Artificial Neural Network Pattern Recognition.....	29
Figure 25: Error between actual and predicted target class .....	31
Figure 26: Confusion matrix for training data .....	32
Figure 27: Confusion matrix for verification .....	34
Figure 28: Residual vs time plot .....	36
Figure 29: Residual vs time plot .....	38



## List of Tables

Table 1: Fault detection delay times during using both methods (time is in hours) ..	10
Table 2: Summary of review .....	13
Table 3: Definition of fault .....	20
Table 4: Operating range of each parameter .....	21
Table 5: Data analysis .....	23
Table 6: Hang and shutdown data isolation in MS Excel workspace .....	23
Table 7: PC number .....	24
Table 8: Neural network parameter.....	29
Table 9: Number of neurons with each resulting percentage error .....	30
Table 10: Fault classes .....	30
Table 11: Fault classification accuracy .....	33
Table 12: Fault classification with healthy data.....	33
Table 13: Fault classification without healthy data.....	35
Table 14: Fault classification with healthy data.....	35
Table 15: Manipulated normal data for first six hour .....	36
Table 16: Manipulated drift data for first six hour.....	37

## **List of Abbreviation**

PCA – Principal Component Analysis

ANN – Artificial Neural Network

SPE – Square Predicted Error

PC – Principal Component

## **Chapter 1**

### **INTRODUCTION**

#### **1.1 Background**

The reliability and availability of instrument data at a metering system plays a crucial part in the gas transportation system as it affects the billing integrity between the gas supplier and their customers. In the industrial process control, a large variation of instruments is available to measure the data and parameter of the control variables. A large amount of data is generated through measurement and these large historical data introduce diverse diagnostic patterns which can be extracted in order to deduce the information about the plant status and thus, causing the problem of important data mining.

In controlling the process, the probability for fault to occur is inevitable. The inaccurate data of the system will contribute to major loss of the company as well as catastrophe effects. Therefore, the challenge lies in building fault detection and prediction system that is able to check the consistency or status of instruments in the metering system as well as to reconstruct unreliable data with an acceptable range of accuracy in the case of faults on the instruments during operation. Thus, in order to realize this, there is a need to design a model that will detect and diagnose the type of fault to ensure the continuity of a reliable process.

## **1.2 Problem Statement**

The accurate measurement of turbine meter, temperature transmitter and pressure transmitter essentially affects the accuracy of the billing integrity process. All instruments are required to provide and transmit reliable data to the flow computer for gas flow monitoring and billing calculation. Typically during operation there are some common unavoidable faults that might happen on instruments such as shutdown fault, hang fault and drift fault. Then main issue is when instruments experience a fault, value readings from instruments become unreliable and this directly affects the calculation of energy consumption. At present during this urgent circumstance, there is limited system that is capable of performing auto-correction data during malfunction as well as the fault detection system to detect and isolate the types of fault. Thus, by having a model to detect the fault for further correction will greatly improve the reliability and availability of the metering system.

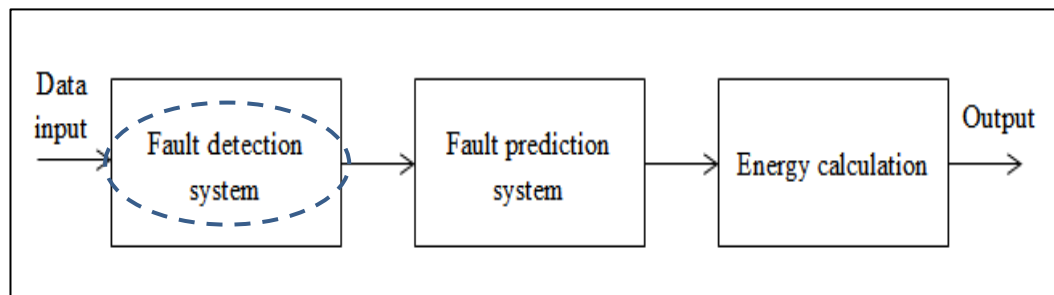
## **1.3 Objectives**

The main aim of the research project is to design and develop a detection model based on advanced approach that has the ability to detect fault of instruments and provide reliable predicted data. The objectives of the research are as follows:-

- a. To investigate and analyze signal from all measurements that is pressure, temperature and volume in order to identify type of instrument faults;
- b. To design and develop detection techniques that is based on empirical approach (i.e. statistical method and ANN);
- c. To evaluate the performance of each proposed detection model based on accuracy, reliability and consistency the model.

### 1.3 Scope of Study

The scope of study for the project is focused on the fault detection method. In current practice in energy calculation and billing, there are only two systems which are to predict the data whenever the fault occurs and also energy and billing calculation. As shown in Figure 1, the overall system for billing correction system shall begin with the fault detection system to detect if there are faults in the instruments used. The inputs of the fault detection system will be the data of control variables from the instruments which include temperature, pressure as well as volume. The system will send the healthy and unhealthy data to the fault prediction system to do correction on the fault data to ensure the accuracy of the energy calculated at the end of the process.



**Figure 1: Auto-billing correction system**

### 1.4 Tools and Software Required

The fault detection techniques will be coded in MATLAB with control toolbox to analyze and obtained the result.

## **Chapter 2**

### **LITERATURE REVIEW**

#### **2.1 Definition of Fault**

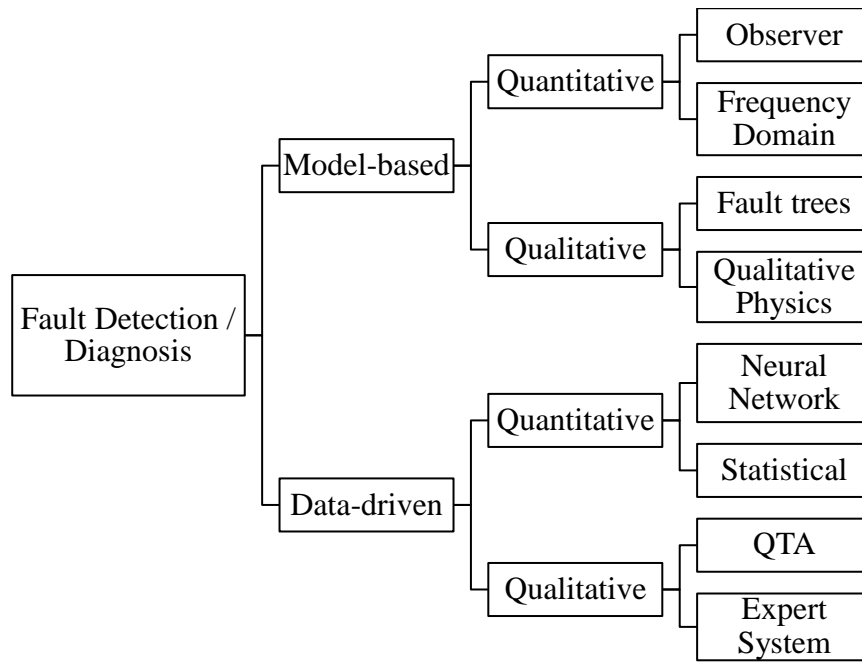
Fault can be defined as an unpermitted deviance of at least one typical property of the system from the acceptable condition [1]. With the vast development of science and technology, automation systems had become much more complex [2,3] and fault is often being associated with the monitoring the automated system [4]. Furthermore, all the process industries nowadays are in the huge competition among each other to become the most reliable and promising in delivering the products as well as maintaining a higher performance and safety against any possible abnormalities [5]. As stated in most researches on the fault detection, it is vital to detect the fault to avoid the deterioration of the system performance and eventually ensure the reliability and safety in most engineering systems.

#### **2.2 Types of Fault**

There are several types of fault that may occur in process plants. According to [1], there are three types of fault that may be occur in the process plants which are sensor faults, actuator faults and component faults. Under the sensor fault there might be hand fault, shutdown fault and drift fault.

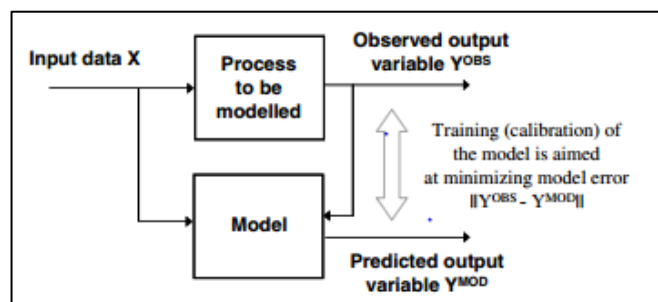
#### **2.3 Fault Detection and Diagnosis**

The detection of fault at an early stage is crucial to maintain the reliability of the system so that the fault diagnosis strategy can take place to do the corrective measure of the data in any systems for future consistency. In the process of monitoring and detecting the fault, efficiency and robustness is the main concern to ensure the system's and operator's health [6]. There are many studies that had been done to determine the best fault detection method. The techniques to perform the fault detection can be broken down into two different categories which are model-based techniques and data-driven techniques as shown in Figure 2.



**Figure 2: Fault detection and diagnosis category**

Model-based techniques can be defined as the method that requires the system structure and also a well aware operation of law [7,8]. This means that model-based techniques consist of accurate mathematical model or formula which the drawbacks of this techniques are too much time consuming as it characterizing all the potential operational occurrences [5]. In analyzing a large amount of data, data driven method is preferable compared to model-based as it utilizes the historical data to the fault detection and would require less plant information. Data driven method is divided into two types which are qualitative and quantitative analysis. In this project, the method which is under consideration is the quantitative data-driven method as the approach is towards the neural network and statistical techniques which are expected to give the most reliable output.

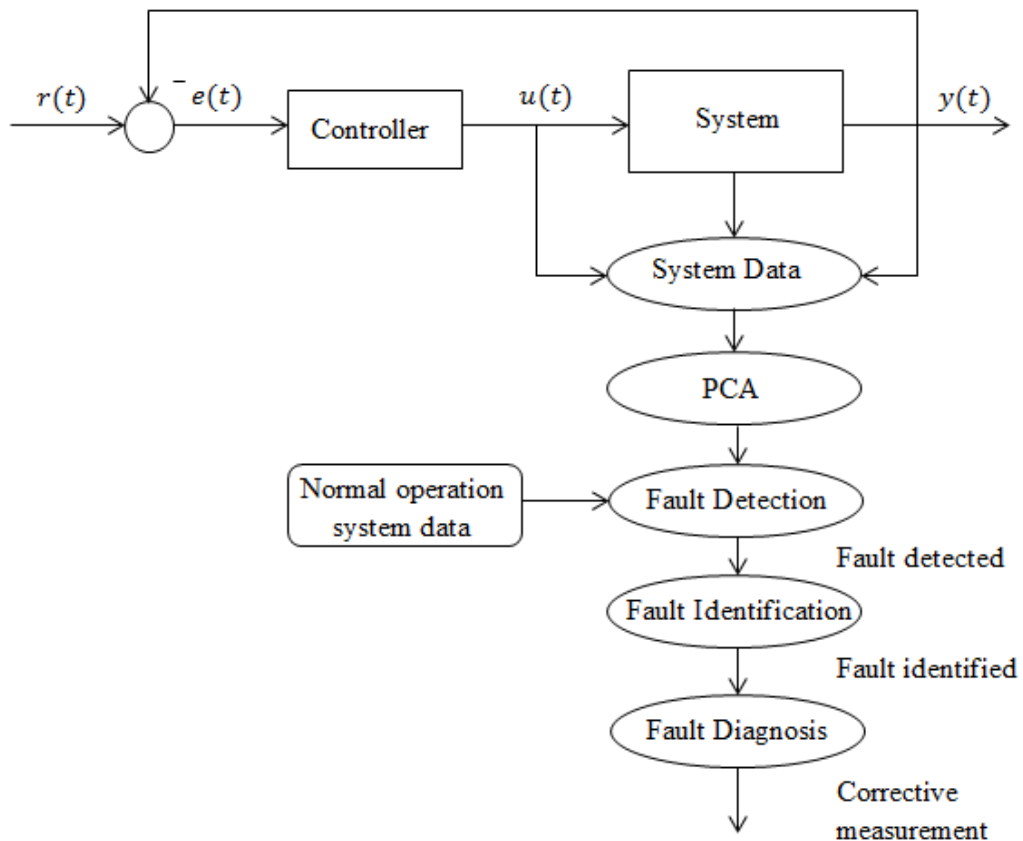


**Figure 3: Data-driven approach**

### 2.3.1 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical approach is detecting fault which fall under the data-driven analysis category [1,5,8]. PCA imply the techniques of reduction of dimensionality of the data which mean it lower the dimensionality of high dimensional process to a much lower dimensional space [1,5].

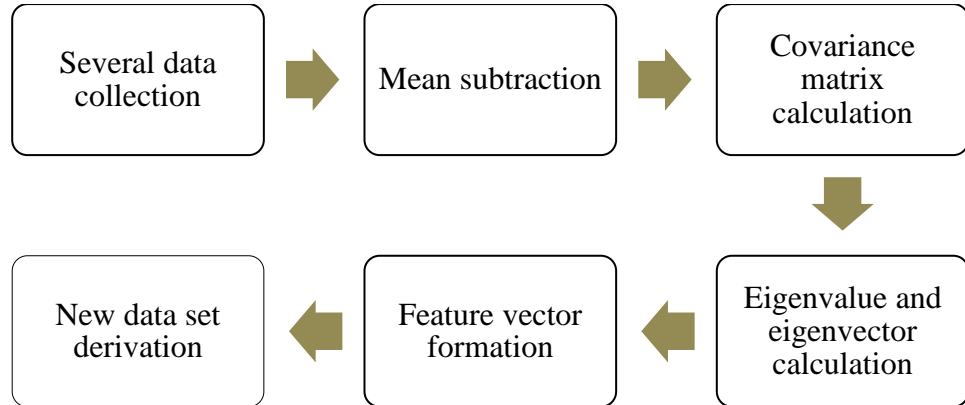
According to [1], the process of fault detection and isolation begin with the extraction of data from the system based on the past data history and the noise effects are being reduced by using PCA techniques. The output of the reduced dimensional data is then being compared with the data under normal operating conditions to determine whether there is fault occurred or not. Consequently, the fault next will be isolated to identify the types and its position in order to further proceed with corrective measurement.



**Figure 4: Fault detection and isolation process**



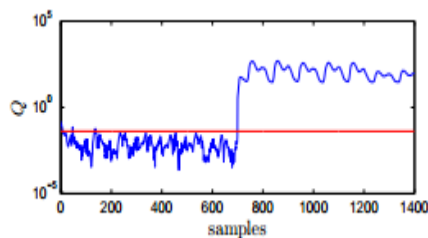
The steps in performing PCA study had been published in many research conducted in the past and the general method of performing PCA algorithm is shown in Figure 5 [8].



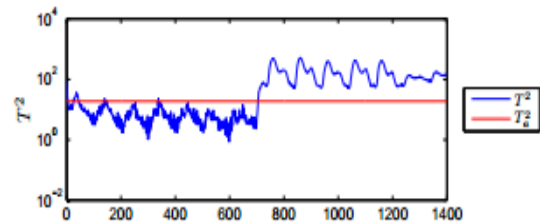
**Figure 5: PCA methodology**

### 2.3.1.1 T-square and SPE Chart

T-square or  $T^2$  Hotelling chart is one of the mostly used techniques to show the fault in the sets of data. It is a chart where the variance in the PCA model is captured. These statistics are generally only available for factor-based models such as PCA. Based on Figure 7, T-square distribution shows the number of data captured by the model while SPE chart in Figure 6 is the data which is not captured by the model and also known as outliers [9].



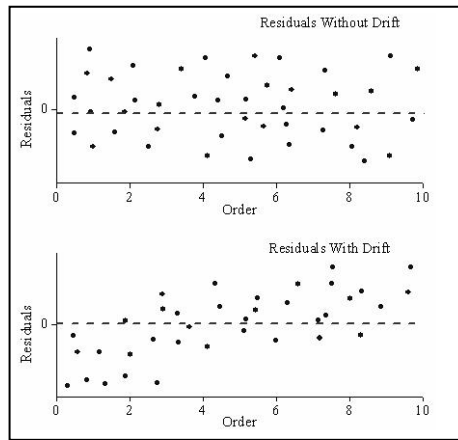
**Figure 6: SPE chart**



**Figure 7: T-square chart**

### 2.3.1.2 Residual for drift detection

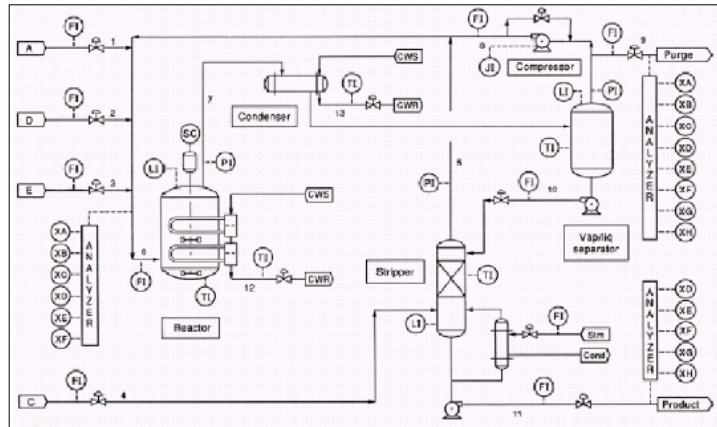
Theoretically, when there is a drift in the data, the residual obtained from statistical calculation will show a significant trend in its plot. The Residual vs. Order of the Data plot can be used to check the drift of the variance during the experimental process, when data are time-ordered. If the residuals are randomly distributed around zero, it means that there is no drift in the process [11]. Figure 8 illustrates the residual plot to indicate if there is a trend of drifting in the data set.



**Figure 8: Residuals plot for drift detection**

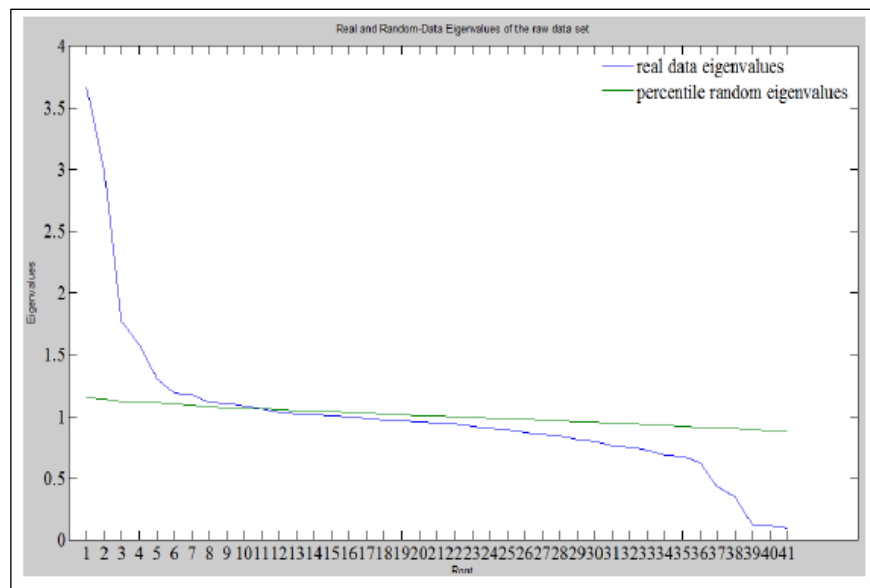
### 2.3.1.3 Example of application on PCA techniques

A study on the fault detection had been done by [1] on Tennessee Eastman (TE) process. The experiment is done to detect the fault during the process by getting sets of data and implementing PCA model to them. The TE process flow sheet is shown in Figure 9 where the number of input signal in the process is found to be 12 while the outputs are 41. However, among the 41 outputs, only 22 are continuously measured and the rest are chromatographs' sampled composition analyses [1,5].



**Figure 9: Tennessee Eastman process flow sheet**

The PCA model is simulated in the MATLAB code and sets of data are generated by applying 21 faults data and one set of healthy data. The PCA simulation is done by training the data under normal operating condition. Then, after the model is trained, it is tested for the 21 faults data and the number of principal component is calculated [1].



**Figure 10 : Parallel analysis plot for TE process data**

The faults are detected by looking directly at the plot of the outputs from simulation or by applying the PCA model built with the output data taken. The results are obtained and then being compared between the observation of the simulation and with the PCA model technique. The results are shown in Table 1 and thus, it can be seen that by implying the PCA

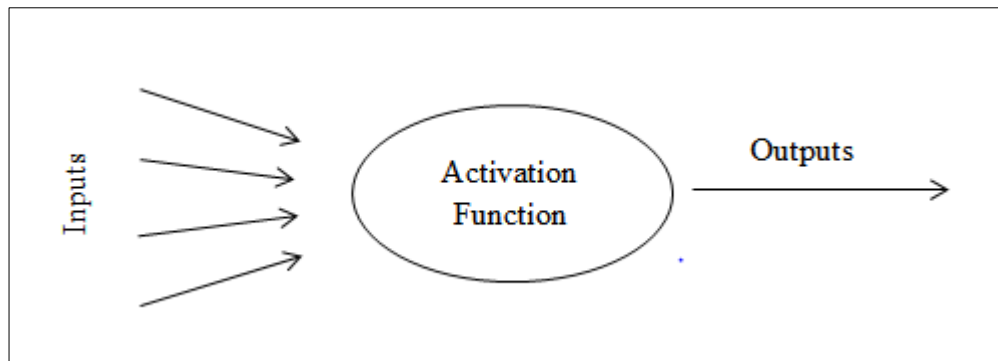
technique in detecting the faults, the delay time is reduced and the fault is significantly detected.

**Table 1: Fault detection delay times during using both methods (time is in hours)**

Outputs	IDV1		IDV2		IDV7		IDV8		IDV11		IDV13	
	1	2	1	2	1	2	1	2	1	2	1	2
D feed(2)	4.4	3.4	9.7	5.2	-	12.1	20.0	12.1	21.3	12.2	12.5	7.2
Reactor level(8)	3.0	1.0	-	3.4	-	10.3	11.2	10.3	-	-	3.2	1.7
Stripper underflow(17)	3.5	2.7	-	4.3	-	-	20.0	19.9	-	-	4.5	2.1
Stripper Steam Flow(19)	-	3.4	-	3.4	-	-	-	12.6	-	10	-	2.1
Component D to reactor(26)	0.91	0.12	-	4.3	-	0.02	6.8	3.9	15.1	5.1	7.3	1.4
Component E to reactor(27)	2.0	1.8	-	3.4	-	-	8.4	7.9	-	-	2.8	1.4

### 2.3.2 Artificial Neural Network

A neural network is a technique which is biologically inspired computational model based on the way in which the human brain functions. The implementation of the human brain to the neural networks is basically natural neurons receive the signals via the synapses located on the dendrites of the neurons. The neuron is activated and emits the signal via the axon when the signal received is beyond the threshold value. Eventually, the signal is passed to another synapse and probably will trigger other neurons [14,16]. The artificial neuron follows the same concept of human brain neuron where it fundamentally consists of inputs, weight, and computation method which responsible in determine the activation of neuron.

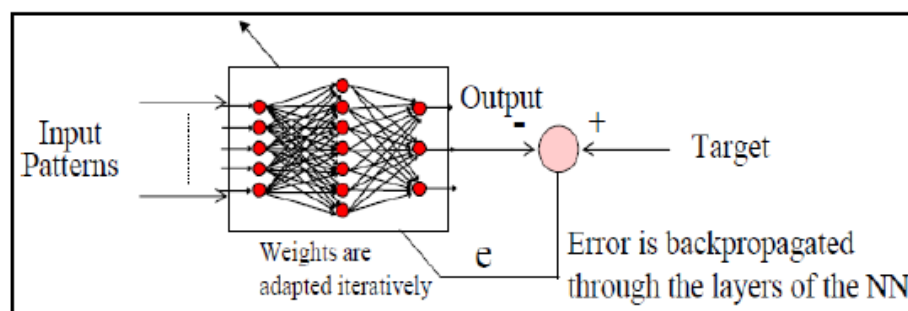


**Figure 11: Neural network model**

ANN is an architecture composed by a large amount of interconnected processing units which is later being combined together to produce an output signal to solve the problems according to the received input signal [17,18,19]. In fault detection system, the ANN is trained based on historical or simulated steady state process data for the purpose of detecting a specified number of assumed faults.

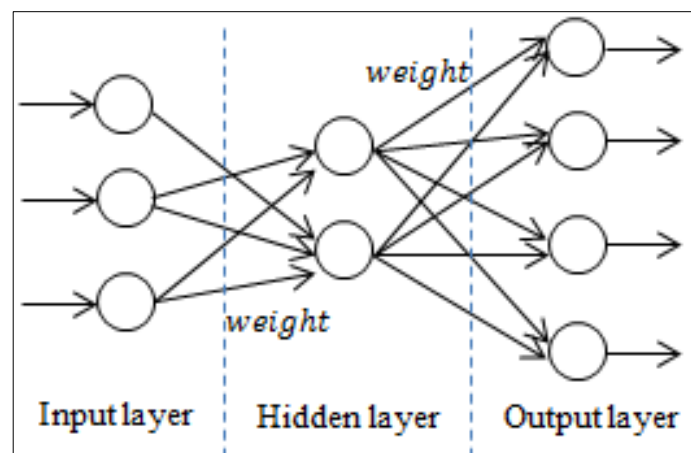
### 2.3.2.1 Experiment on ANN technique

In the research conducted by [19], ANN is used to classify fault in the system of process control rig. In the paper, the back propagation ANN with multilayer perceptron (MLP) is being used as shown in Figure 12.



**Figure 12: A backpropagation neural network**

The structure fault classification system of ANN can be referred to Figure 13 where the information from the fault symptom generation ( $R_1 - R_n$ ) is being fed to a trained neural network. Each output node is corresponding to one of the faults ( $F_1 - F_m$ ). The result of fault diagnosis can be interpreted as the output of a certain node is near to the one indicates the corresponding fault. Meanwhile, the output which is close to zero shows that there is no occurrence of fault. The input of the residual in the project is based from residual model developed [19].



**Figure 13: Multilayer ANN for fault classification**

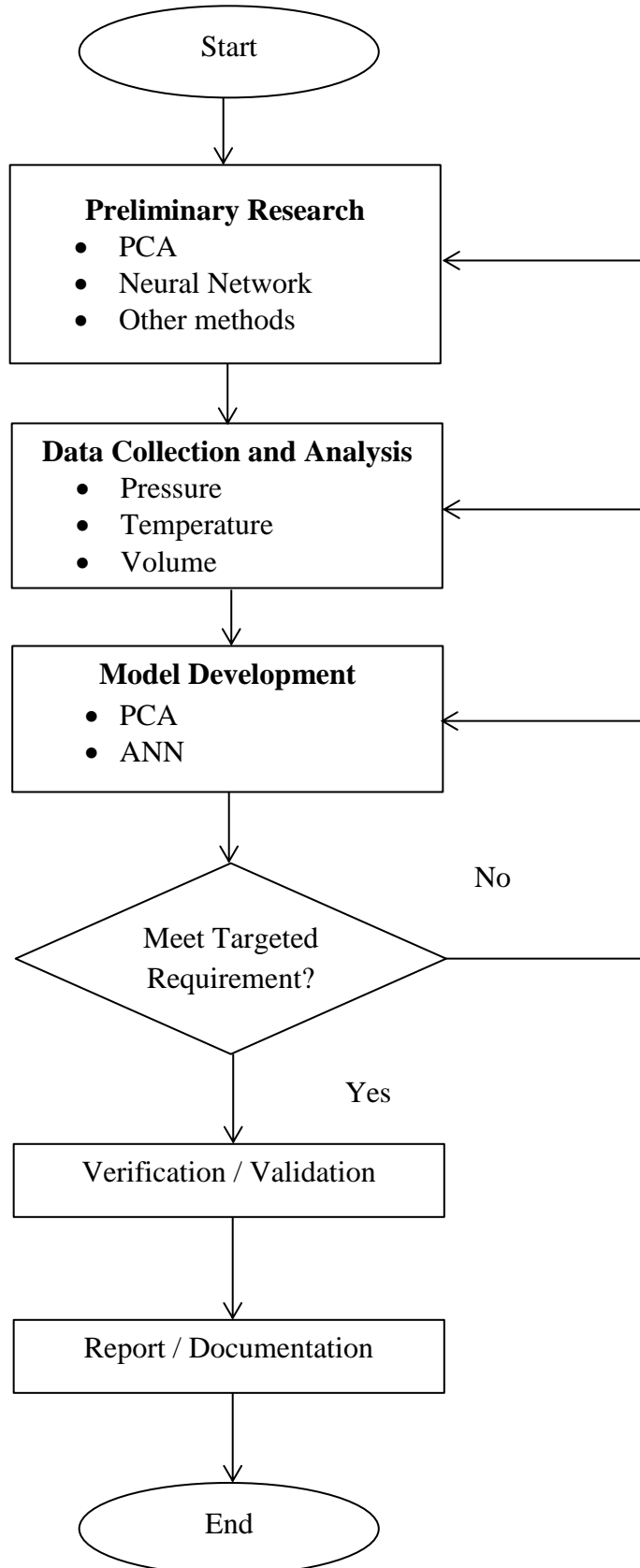
## 2.4 Summary of literature review

**Table 2: Summary of review**

	<b>Fault detection techniques</b>	
	<b>PCA</b>	<b>ANN</b>
Type	<ul style="list-style-type: none"> <li>• Statistical data-driven method</li> </ul>	<ul style="list-style-type: none"> <li>• Non-statistical data-driven method</li> </ul>
Techniques	<ul style="list-style-type: none"> <li>• Normalization</li> <li>• Covariance matrix</li> <li>• Eigenvalue and eigenvector</li> <li>• Feature vector transformation</li> </ul>	<ul style="list-style-type: none"> <li>• Data preparation</li> <li>• Training data</li> <li>• Simulation data</li> </ul>
Advantages	<ul style="list-style-type: none"> <li>• Powerful, well-established technique for data reduction</li> <li>• Can handle multivariable data in the process (i.e. 12 parameters at once)</li> </ul>	<ul style="list-style-type: none"> <li>• Ability to learn and train from the past data</li> <li>• A good model increases the robustness</li> </ul>
Disadvantages	<ul style="list-style-type: none"> <li>• The covariance matrix is difficult to be evaluated in an accurate manner.</li> </ul>	<ul style="list-style-type: none"> <li>• The greater computational burden, the higher the proneness to overfitting,</li> </ul>
Application	<ul style="list-style-type: none"> <li>• Process monitoring</li> <li>• Data mining and analysis</li> </ul>	<ul style="list-style-type: none"> <li>• Pattern recognition</li> <li>• Forecasting data</li> <li>• Fault Classification</li> </ul>
References	[1],[2],[5],[6],[8],[9],[10],[13],[14]	[11],[14],[15],[16],[17],[18],[19],[20]

**Chapter 3**  
**METHODOLOGY**

**3.1 Project Flowchart**





a) Preliminary Research

After the title selection and discussion about the scope of study, a preliminary research had been conducted to ensure that all knowledge and information from every aspects related to the topic are fully covered. As written in Chapter 2, there are three proposed algorithms which are PCA and ANN. It is believed that with an in-depth understanding on each method, the project will be well delivered.

b) Data Collection

The data input for the fault detection system will be taken from the oil and gas metering station. The data parameters include pressure, temperature and volume. These data will then being fed into the system for the fault detection.

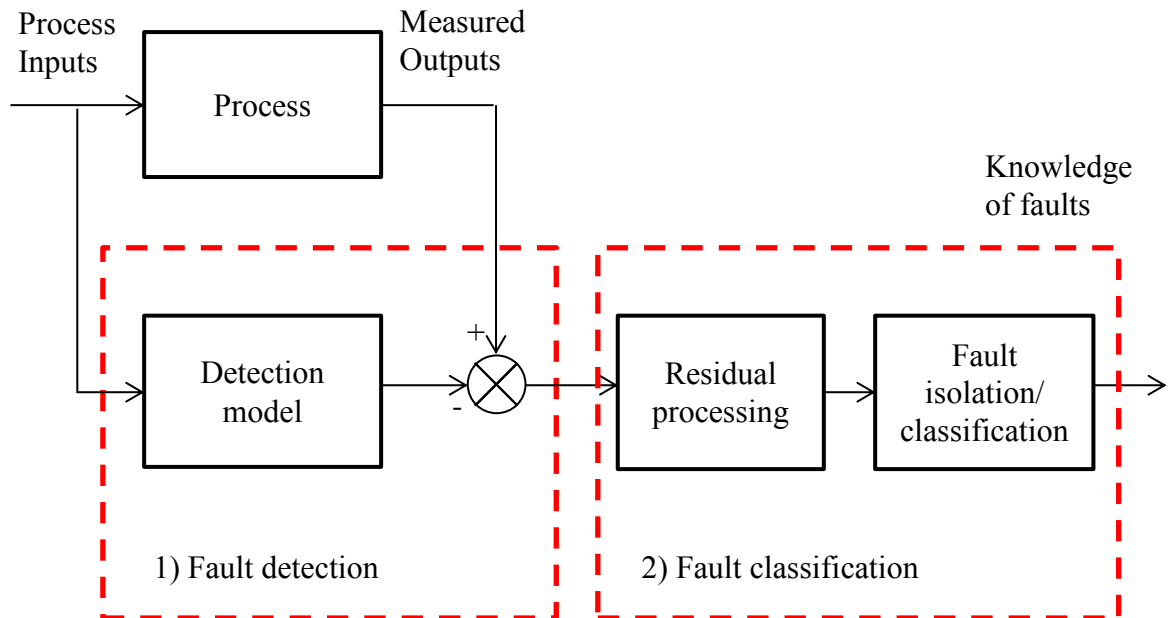
c) Testing and validation

The entire proposed algorithm will be tested with the prior data to verify that the method satisfies the targeted requirement for fault detection. The testing and validation will be repeated for many times with several cases to prove its functionality before it can be implemented to the system.

d) Report and documentation

Finally, after selecting the best method that will provide the accurate detection for the fault, the report and documentation will be done and submitted.

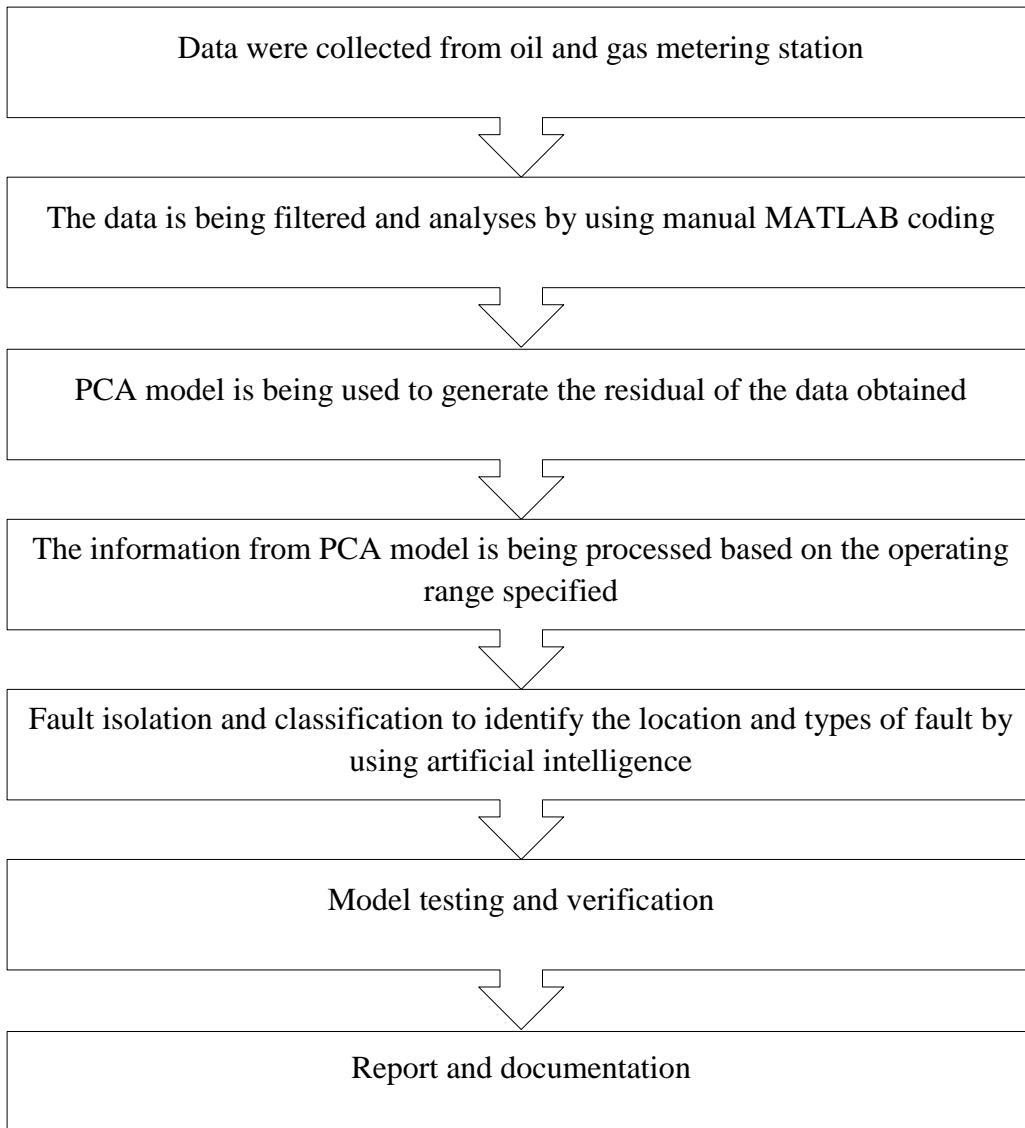
### 3.2 System Block Diagram



**Figure 14: Fault detection and classification block diagram**

Figure 14 shows the system block diagram for fault detection and diagnosis strategy. The system consists of two major parts which are the fault detection and fault classification. The function of fault detection block is to calculate and determine the number of residual of each parameters of the process which include Temperature( $T$ ), Pressure ( $P$ ) and Gross Volume( $V_g$ ). The residual can be generated by implementing the PCA algorithm to the MATLAB software and the result is obtained and analyzed. The second part of the project methodology is the fault classification by utilizing the artificial intelligence where the generated residual from the first block will be processed to isolate the location and time of fault occurred. Then, the fault classification also will give the information on the types of fault.

### 3.3 Project Activities



### 3.4 Gantt Chart (FYPI)

Activity	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
Project title selection														
Preliminary research														
Extended proposal submission							▲							
Fault detection/ diagnosis research continuation														
Proposal defense									▲					
Draft report													▲	
Interim report submission														▲

### 3.5 Gantt Chart (FYP2)

Activity	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15
Data filtering and classification	■	■	■												
Development of residual generation model			■	■	■										
Progress report submission								▲							
Fault isolation and classification						■	■	■	■						
Testing and verification									■	■	■	■			
Pre-SEDEX										▲					
Final report													▲		
Technical paper													▲		
VIVA															▲

## Chapter 4

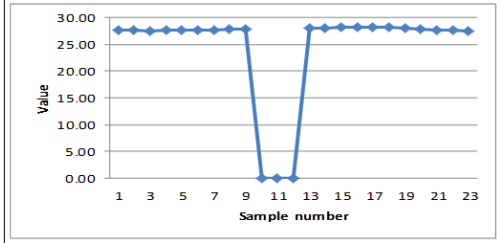
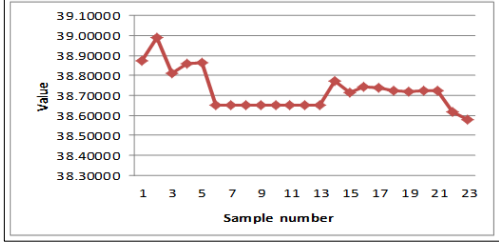
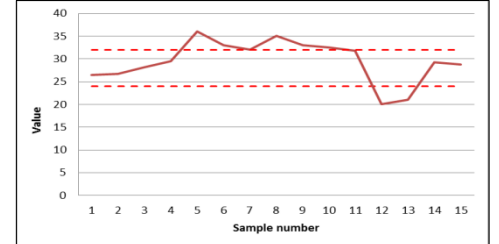
### RESULT AND DISCUSSION

This chapter explained the detailed of project work and the findings obtained. The findings are divided into three parts which are data collection and filtering, fault symptom generation and fault classification. In the data collection and filtering section, the data is filtered based on several types of fault. Next, the residual generation section explained the variation captured in the data and produced the new transformed matrix by utilizing PCA method. Finally, the fault is calssified by artificial neural network model.

#### 4.1 Fault Definition

The types of fault are defined as in Table 3.

**Table 3: Definition of fault**

Types of fault	Definition	Graph
Shutdown	The reading of data shows zero value throughout the process <b>except</b> for Volume.	
Hang	The data reading shows a constant value for at least <b>three</b> consecutive data.	
Out of range	The data falls on the <b>outside of allowed</b> operating range.	

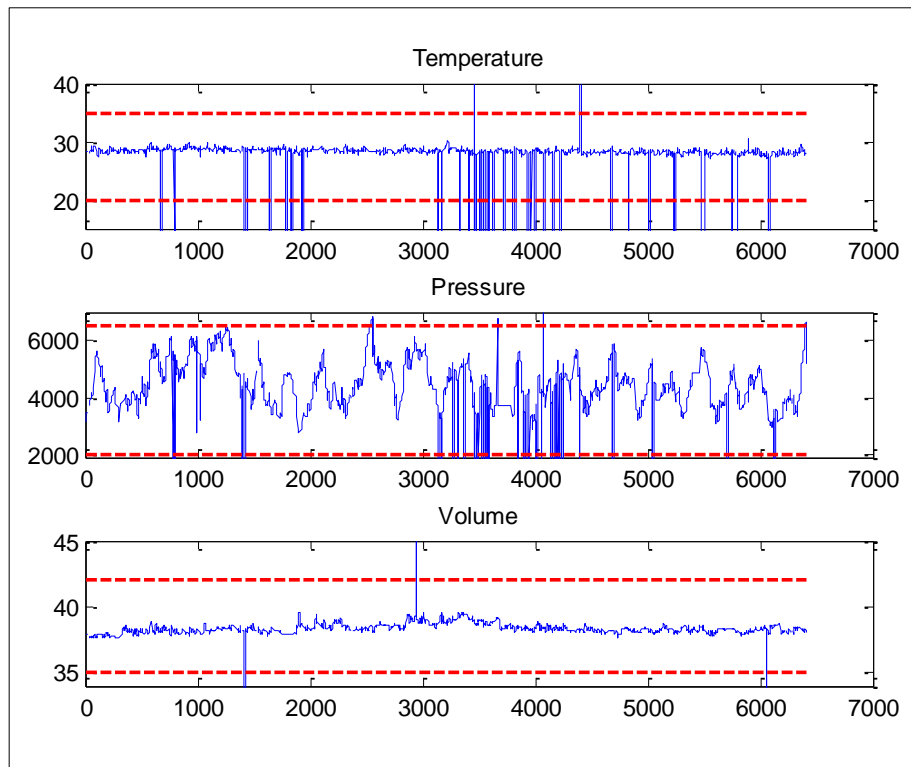
Drift	The data is <b>deviating</b> from the normal operating range at a certain rate.	
-------	---	--

## 4.2 Data Collection and Analysis

The raw data are collected from the metering station which consists of three parameters which are Pressure (P), Temperature (T) and Volume (V). The data is first being analysed manually to investigate the behaviour of each parameter over time. The first data set which is being analysed is composed of 6408 observations which correspond to almost one year of data. Figure 15 shows the data plot for each parameter.

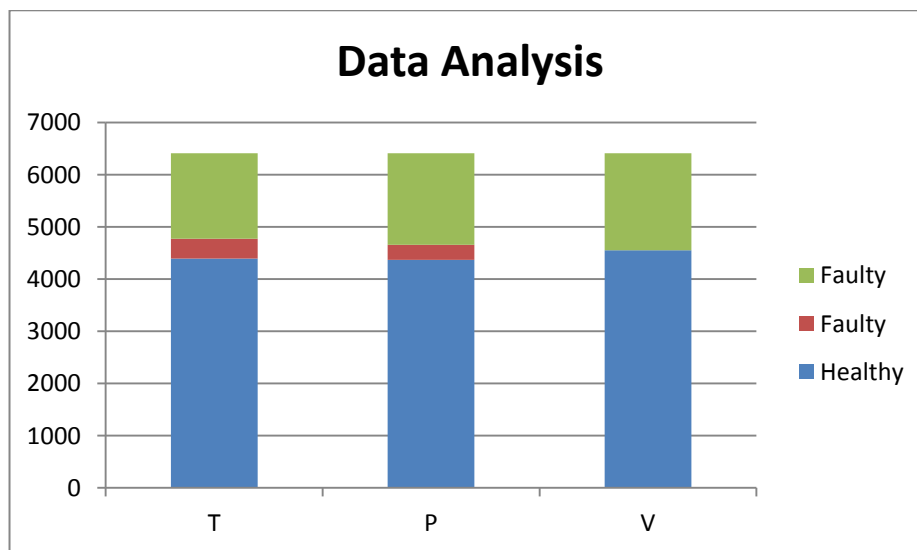
**Table 4: Operating range of each parameter**

Parameter	Lower Operating Limit	Higher Operating Limit
Temperature (°C)	20	35
Pressure (kPag)	2000	6500
Volume (m <sup>3</sup> )	35	42



**Figure 15: Data plot**

The data is then being analysed to extract useful information regarding healthy and faulty data based on the fault definition in Table 3. Figure 16 shows the analysis of the data for each parameter respectively. It can be seen that volume reading has more fault compared to the other two parameters. Table 5 summarize the percentage of healthy and faulty data in the system.



**Figure 16: Data analysis for each parameter**



**Table 5: Data analysis**

Type	No. of data					
	Temperature	%	Pressure	%	Volume	%
Healthy	4390	68.5	4370	68.2	4551	71
Faulty	2018	31.5	2038	31.8	1857	29

Based on the data collected, the data was first being sorted for identifying the first two types of fault which are zero and hang fault.

**Table 6: Hang and shutdown data isolation in MS Excel workspace**

Parameter	Hang fault	Shutdown fault																																																								
Temperature	<table border="1"> <tr><td>4858</td><td></td><td></td><td></td></tr> <tr><td>4859</td><td></td><td></td><td></td></tr> <tr><td>4860</td><td>27.62343</td><td></td><td></td></tr> <tr><td>4861</td><td>27.62343</td><td></td><td></td></tr> <tr><td>4862</td><td></td><td></td><td></td></tr> <tr><td>4863</td><td></td><td></td><td></td></tr> <tr><td>4864</td><td></td><td></td><td></td></tr> </table>	4858				4859				4860	27.62343			4861	27.62343			4862				4863				4864				<table border="1"> <tr><td>674</td><td>0</td><td></td><td></td></tr> <tr><td>675</td><td>0</td><td></td><td></td></tr> <tr><td>676</td><td>0</td><td></td><td></td></tr> <tr><td>677</td><td>0</td><td></td><td></td></tr> <tr><td>678</td><td>0</td><td></td><td></td></tr> <tr><td>679</td><td>0</td><td></td><td></td></tr> <tr><td>680</td><td>0</td><td></td><td></td></tr> </table>	674	0			675	0			676	0			677	0			678	0			679	0			680	0		
4858																																																										
4859																																																										
4860	27.62343																																																									
4861	27.62343																																																									
4862																																																										
4863																																																										
4864																																																										
674	0																																																									
675	0																																																									
676	0																																																									
677	0																																																									
678	0																																																									
679	0																																																									
680	0																																																									
Pressure	<table border="1"> <tr><td>268</td><td>3781.187</td><td></td><td></td></tr> <tr><td>269</td><td>3781.187</td><td></td><td></td></tr> <tr><td>270</td><td></td><td></td><td></td></tr> <tr><td>271</td><td></td><td></td><td></td></tr> <tr><td>272</td><td></td><td></td><td></td></tr> <tr><td>273</td><td></td><td></td><td></td></tr> <tr><td>274</td><td></td><td></td><td></td></tr> </table>	268	3781.187			269	3781.187			270				271				272				273				274				<table border="1"> <tr><td>781</td><td>0</td><td></td><td></td></tr> <tr><td>782</td><td>0</td><td></td><td></td></tr> <tr><td>783</td><td>0</td><td></td><td></td></tr> <tr><td>784</td><td>0</td><td></td><td></td></tr> <tr><td>785</td><td>0</td><td></td><td></td></tr> <tr><td>786</td><td>0</td><td></td><td></td></tr> <tr><td>787</td><td></td><td></td><td></td></tr> </table>	781	0			782	0			783	0			784	0			785	0			786	0			787			
268	3781.187																																																									
269	3781.187																																																									
270																																																										
271																																																										
272																																																										
273																																																										
274																																																										
781	0																																																									
782	0																																																									
783	0																																																									
784	0																																																									
785	0																																																									
786	0																																																									
787																																																										
Volume	<table border="1"> <tr><td>275</td><td></td><td>38.23999</td><td></td></tr> <tr><td>276</td><td></td><td>38.23999</td><td></td></tr> <tr><td>277</td><td></td><td>38.23999</td><td></td></tr> <tr><td>278</td><td></td><td></td><td></td></tr> <tr><td>279</td><td></td><td>38.22162</td><td></td></tr> <tr><td>280</td><td></td><td>38.22162</td><td></td></tr> <tr><td>281</td><td></td><td></td><td></td></tr> </table>	275		38.23999		276		38.23999		277		38.23999		278				279		38.22162		280		38.22162		281				Not available since the zero fault is not considered for volume reading.																												
275		38.23999																																																								
276		38.23999																																																								
277		38.23999																																																								
278																																																										
279		38.22162																																																								
280		38.22162																																																								
281																																																										

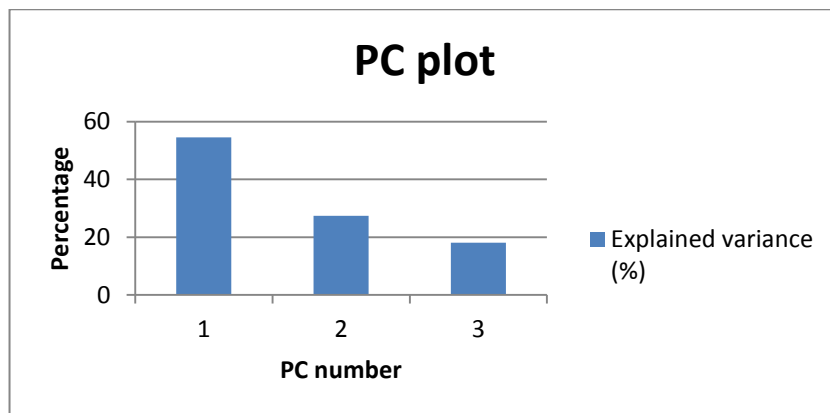
The information of hang data is being exported to Microsoft Excel in order to isolate the location and value of fault reading as shown in Table 6. Then, all the data are being fed to PCA model.

### 4.3 Principal Component Analysis

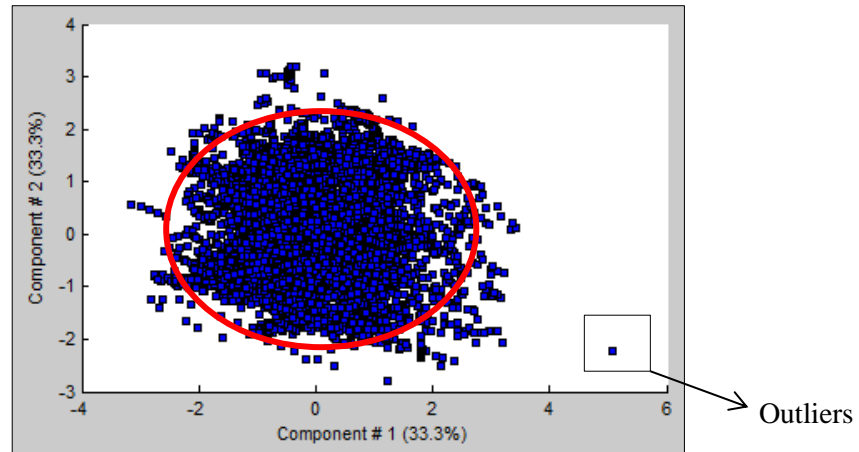
After the entire hang and zero fault data had been analysed, the data is fed to the PCA model for residual generation. These data are tested for any outliers that will lead to drift data. The motivation to use PCA technique is that it reduces the dimensionality of the data and when the data is sufficiently large, the dimensionality reduction helps in capturing any variations in the data. The data is first being pre-processed by normalizing the mean and unit variance. This is also called scaling the data in order to treat all three parameters equally. In this project, the raw data is  $X \in R^{m \times n}$  where  $m$  is number of samples or observations and  $n$  is the number of parameters.

**Table 7: PC number**

PC no.	Explained variance (%)
1	54.52
2	27.42
3	18.05



**Figure 17: Principal Component plot**



**Figure 18: Score plot**

Figure 17 shows the eigenvalue of each principal component and Figure 18 shows the score plot in two dimensions after dimension reduction. Based on the analysis done, the significant number of principal component is found to be 2 because the third component is small and assumed to be negligible. From the Component #1 vs. Component #2 plot, the outliers are clearly seen because the correlation between the first two components is stronger than the third component. The red circle in Figure 18 demonstrates the data which is considered as normal or healthy. As the selected number of principal component is 2, the equations for transforming the data into new feature matrix are given by:

$$T = XP \quad \text{for} \quad P \in R^{n \times a}$$

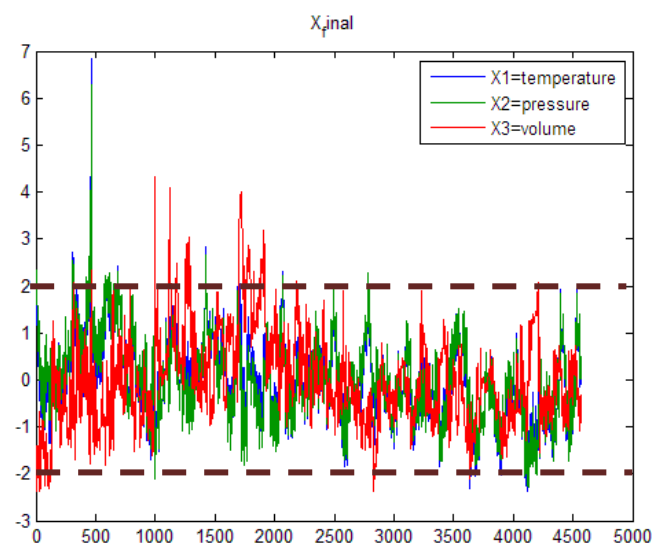
$$\hat{X}_{final} = TP^T$$

*Where*

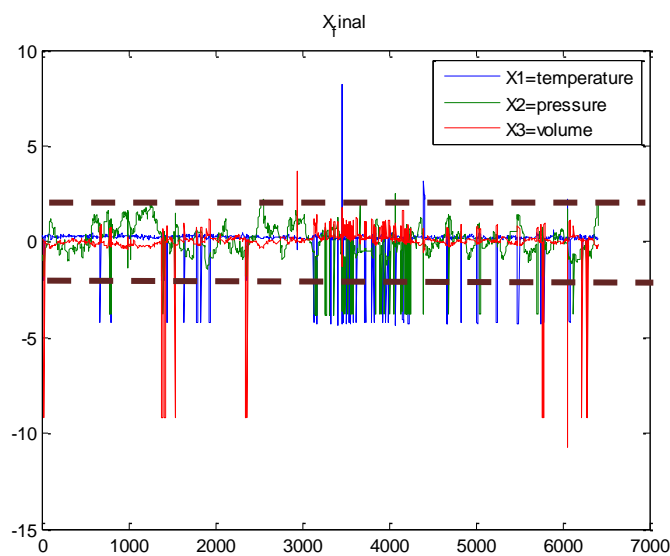
T= the score vector which stores information about relationship between the samples.

P= the loading vector which stores information about relationships between the parameters.

Figure 19 shows the new featured data from back transforming the score and loading vector with original data for healthy data. It can be seen that the variation of the data is concentrated along the x-axis between the ranges of -2 and 2. Slight variation might happen due to some data marginally out of range. Meanwhile, Figure 20 shows the new featured data for faulty data inside the data set. The variation of data is huge where the range extended from -10 to 10. This indicates the fault in the measurement and a significant pattern of data is obtained in order to do detection of fault. Later on the fault classification section, the specific location and the number of faults will be further explained and discussed.



**Figure 19: New featured data from PCA modelling for healthy data**

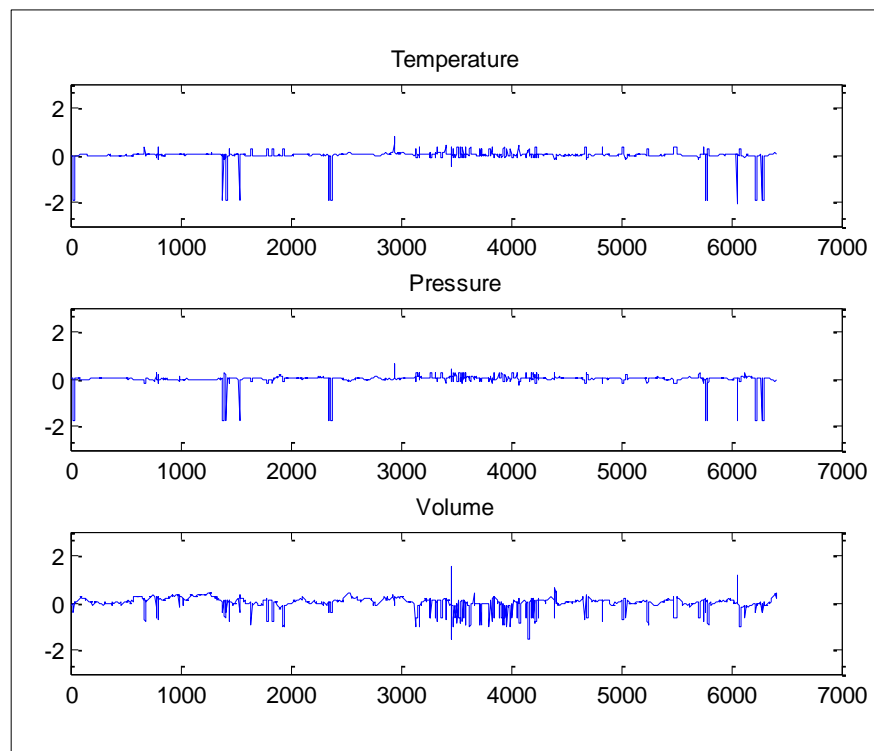


**Figure 20: New featured data from PCA modelling for faulty data**

Finally, the residual matrix is formed by calculating the difference in the original data and the transformed data by PCA. The residual is treated independently for each parameter and the residuals for each parameter can be observed in Figure 21. Eventually, the matrix of residual has an important role in detecting outliers that might be present in the original data set  $\mathbf{X}$  which is a feature that can be used effectively for data inspection and diagnostics.

The residual matrix is calculated by the following equation:

$$E = X - \hat{X}$$



**Figure 21: Residual generated for each parameter**

The information of the residual can be used to construct the SPE chart in Figure 22 which displays the unexplained variance in the data or the outliers of the model. It can be seen that there are some data that exceed the confidence limit of 99% at certain time. It shows that there are some abnormalities in the data which is not captured by the model. Meanwhile, Figure 23 which is the T-squared chart representation shows the explained variance which is captured by the PCA model.

Similarly, there are obvious plots of abnormal data which exceed the confidence limit of the model. Thus, the fault is detected and observed in this part.

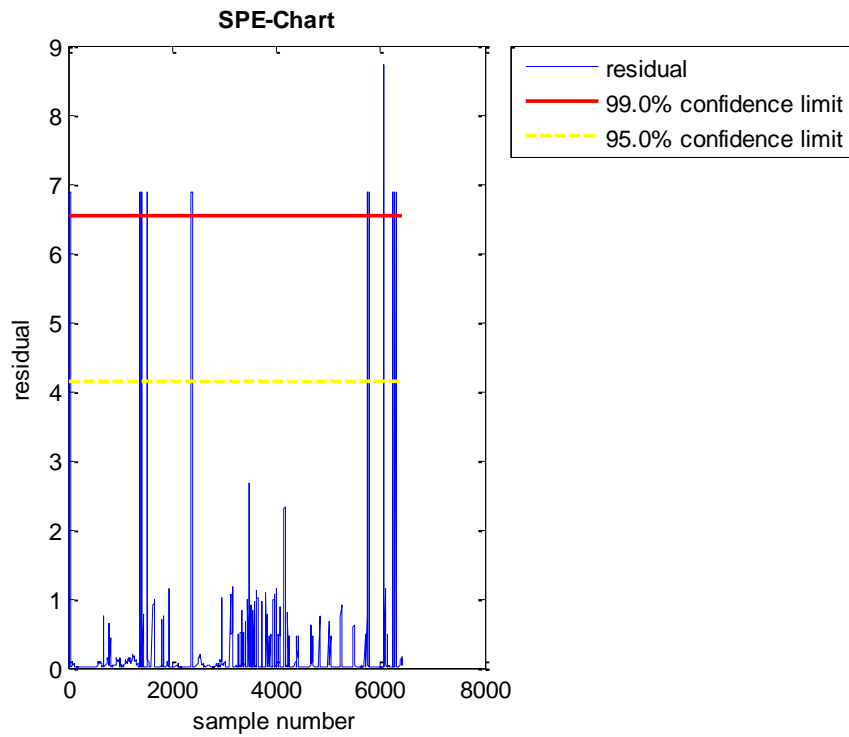


Figure 22: SPE Chart

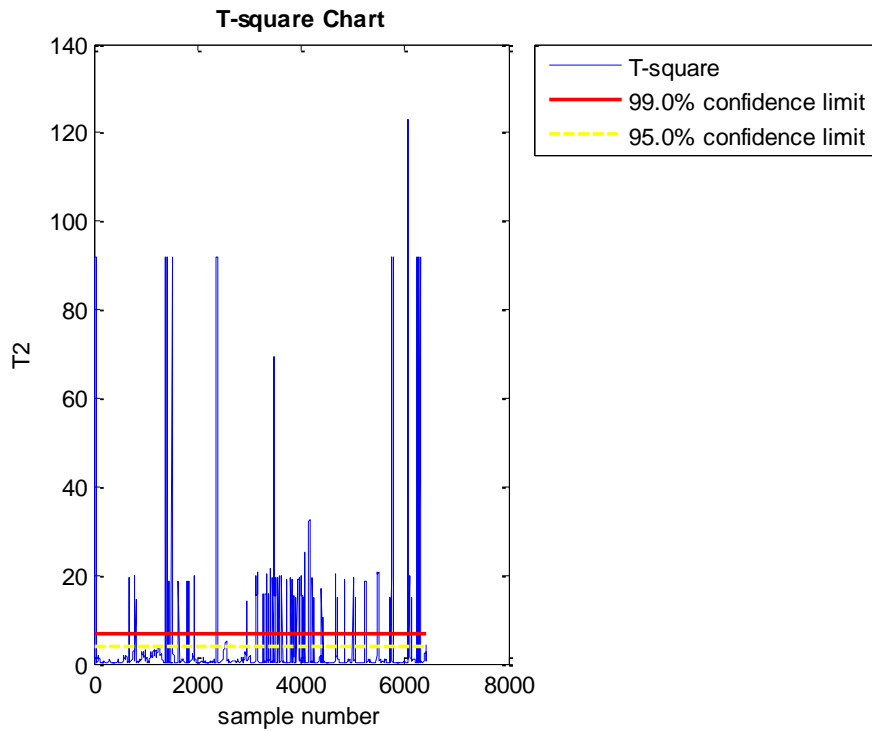
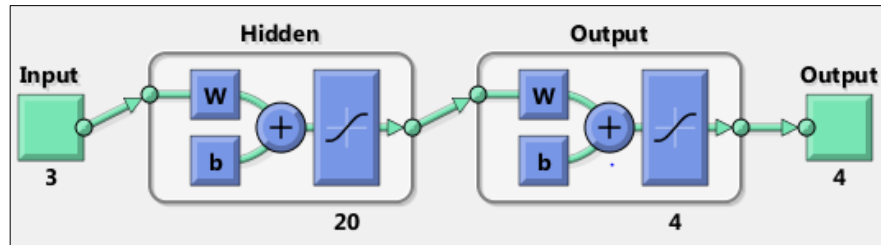


Figure 23: T-square chart

The next step of this project is to evaluate the data in order to classify the location of faults.

#### 4.4 Artificial Neural Network Algorithm

In this project, the neural network has been developed by using MATLAB m-file coding. The selected type of network is pattern recognition neural network. It is a two layer feed-forward neural network model with one hidden layer and one output layer as show in Figure 24. The network model is trained with Bayesian Regulation (BR).



**Figure 24: Artificial Neural Network Pattern Recognition**

The parameters for the network are as below:

**Table 8: Neural network parameter**

Parameter	Algorithm
Divide function	dividerand
Transfer function	Tansig, purelin
Training algorithm	Bayesian Regulation
Performance	Mean squared error
Plot	Confusion plot

The algorithm chosen is based on the best performance of the network.

## 4.5 Artificial Neural Network for Fault Classification

Table 9 shows the number of neuron with their respective validation performance in mean-squared error (mse). The best performance is at neuron 2 with 119 numbers of epochs.

**Table 9: Number of neurons with each resulting percentage error**

Hidden Neuron number	No. of epoch	Validation data (error)	Training data (error)
1	47	7.912 %	16.16 %
2	119	8 %	9.18 %
3	87	8.12 %	11.27 %
4	19	8.04 %	10.67 %
5	243	8.11 %	10.19 %
6	175	8.2 %	10.09 %
7	132	8.15 %	10.2 %
8	161	8.09 %	9.9 %

For the purpose of fault classification, the artificial neural network for pattern recognition is used. This type of neural network is able to classify the data according to their attributes which would belong to certain classes. A network with three inputs, one hidden layer and four outputs are being created as shown in Figure 24. The classes of faults are categorized as in Table 10.

**Table 10: Fault classes**

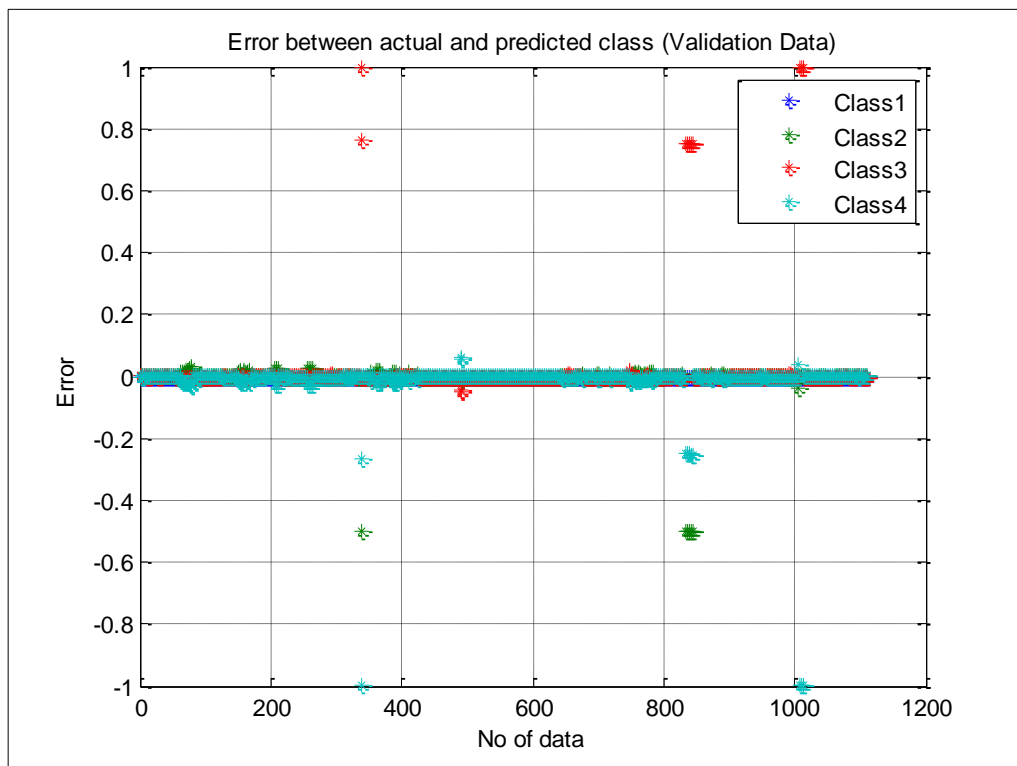
Class	Types of faults
1	Normal
2	Zero
3	Out of range
4	Hang



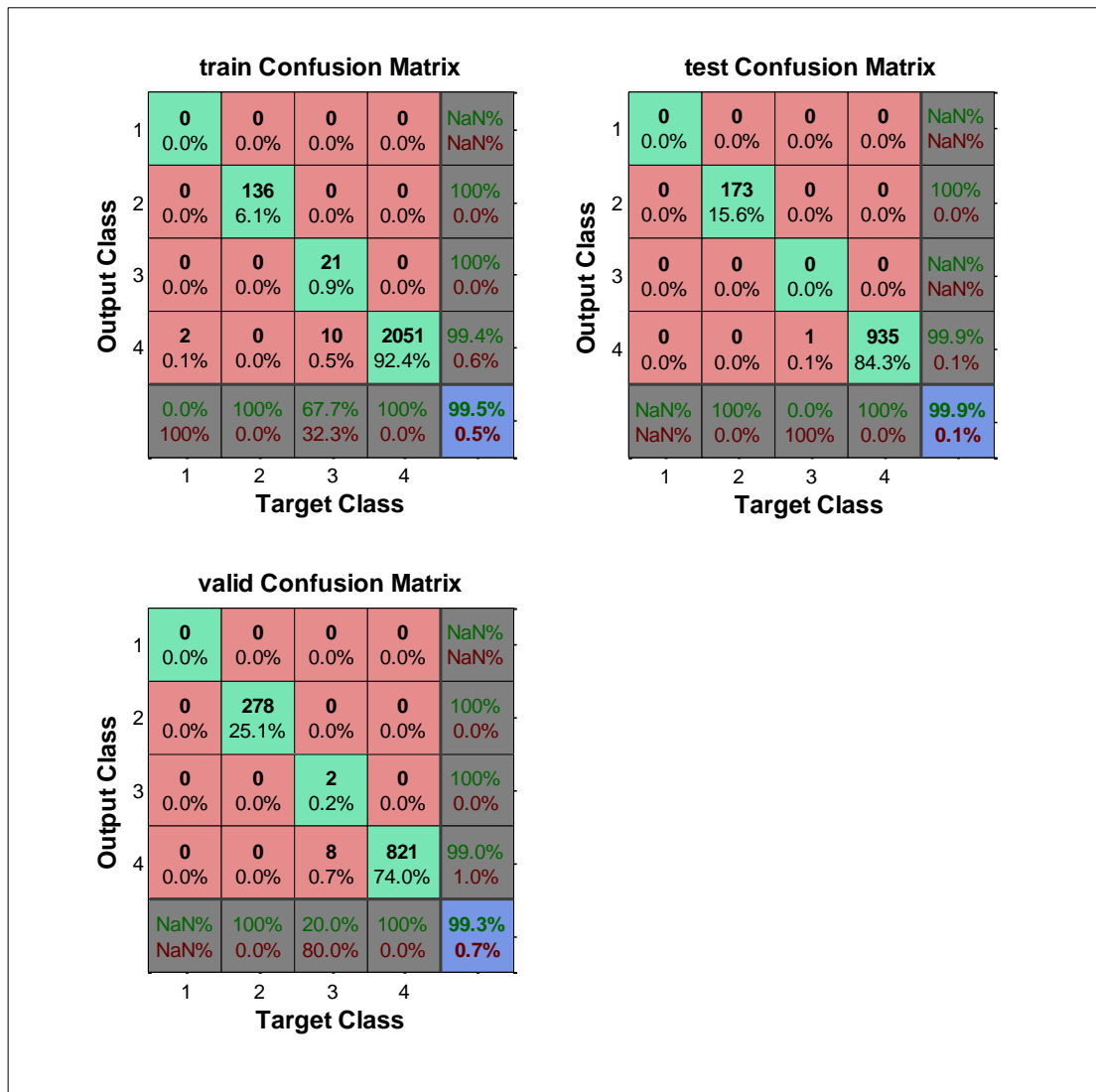
The data from the PCA model is the input to the neural network. The target class is defined based on Table 10 and the assignment of the target is done in a separate m.file. The network trained the faulty data first in order for it to recognize the pattern of faulty data before integrating them with the healthy data for overall performance. The outputs of the training data are being compared with the target data in the confusion matrix plot.

#### 4.5.1 Network training and simulation for faulty data

Firstly, the network is train by isolating the healthy and faulty data. The error for each fault classification is shown in Figure 25. The errors between the target class and actual class are small.



**Figure 25: Error between actual and predicted target class**



**Figure 26: Confusion matrix for training data**

Figure 26 shows the confusion matrix from the neural network. In this matrix, the output of the training data is being compared with the actual target class data. The green diagonal cell show the data which are correctly match to the corresponding target class. The overall percentage of classification is 99.7%. The next step is to simulate the network with the presence of healthy data to test whether the network can correctly recognize the fault.

**Table 11: Fault classification accuracy**

<b>Fault Type</b>	<b>Shutdown</b>	<b>Out of range</b>	<b>Hang</b>
<b>Match</b>	587	33	3807
<b>Mismatch</b>	2	10	0
<b>% match</b>	99.6%	76.7%	100%

From Table 11, when the data is trained and simulated by only faulty data, the fault are classified in a good manner as most of the fault symptom is classified based on the respective types. The accuracy for shutdown and hang are high while for out of range is 76.7% and can be improved by adding sufficient out of range data for training.

#### 4.5.2 *Network training and simulation with healthy and faulty data*

The combination of healthy and faulty data is then being trained with the same network from the faulty data training. Its purpose is to determine whether the fault can be classified with the presence of healthy data in the training network.

**Table 12: Fault classification with healthy data**

<b>Fault Type</b>	<b>Shutdown</b>	<b>Out of range</b>	<b>Hang</b>
<b>Match</b>	583	32	3807
<b>Mismatch</b>	6	11	0
<b>% match</b>	98.9%	74.4%	100%

Here, shutdown fault and hang fault are correctly classified where they have percentage match of above 90% while the out of range only classify 74.4% of the fault. Thus, it can be said that by integrating healthy and unhealthy data, the model is able to classify the types of fault in an almost accurate manner.

### 4.5.3 Simulation with new data set

The neural network model is then being simulated with other set of data to verify its functionality and tests for robustness. The results are shown below.

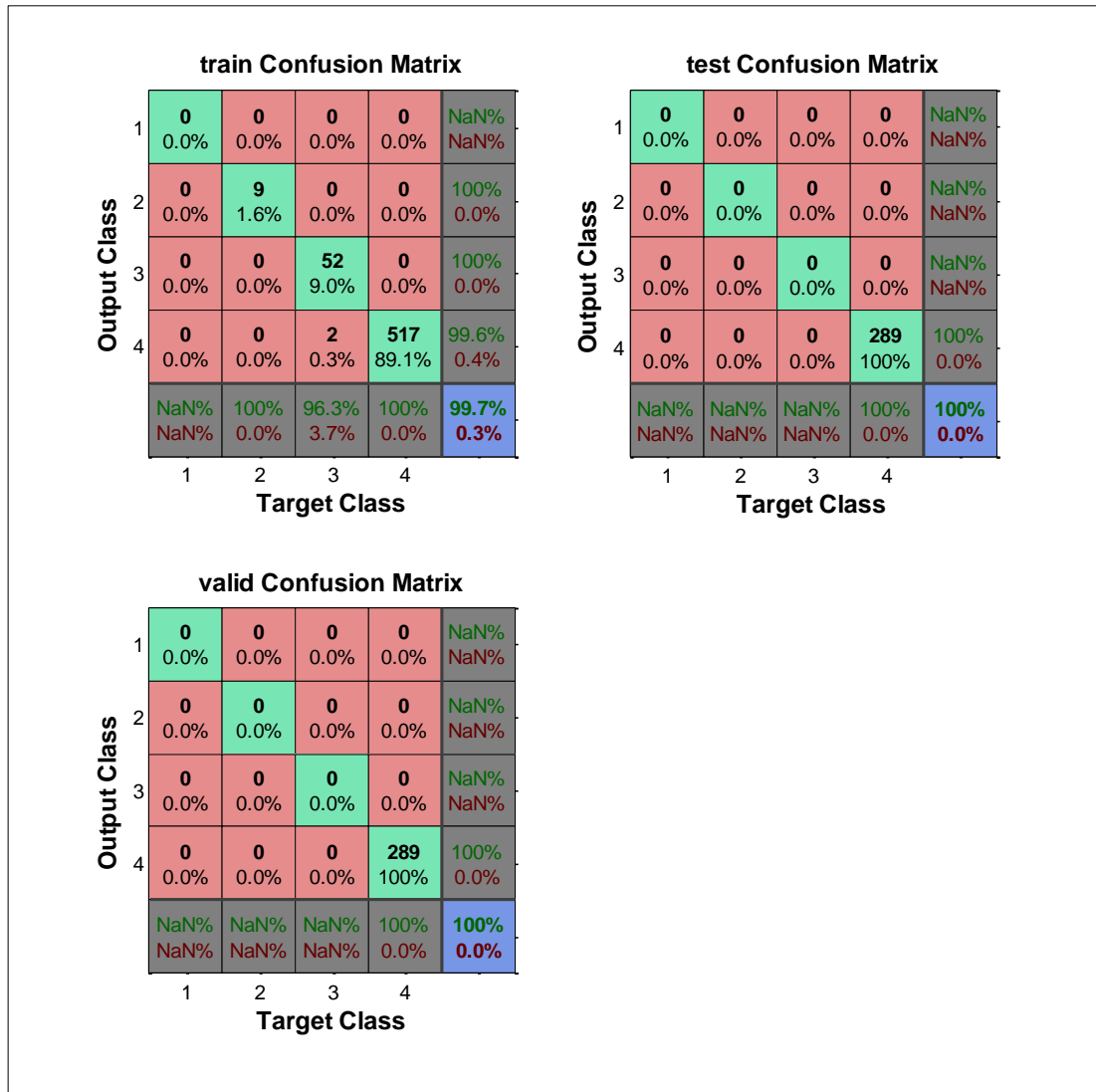


Figure 27: Confusion matrix for verification

Figure 27 shows the confusion matrix for training data after being simulated by using different data set. It also shows a high accuracy of classification where the percentage is 99.7 %

**Table 13: Fault classification without healthy data**

<b>Fault Type</b>	<b>Shutdown</b>	<b>Out of range</b>	<b>Hang</b>
<b>Match</b>	9	52	1095
<b>Mismatch</b>	0	2	0
<b>% match</b>	100%	96.2%	100%

**Table 14: Fault classification with healthy data**

<b>Fault Type</b>	<b>Shutdown</b>	<b>Out of range</b>	<b>Hang</b>
<b>Match</b>	9	52	1095
<b>Mismatch</b>	0	2	0
<b>% match</b>	100%	96.2%	100%

Table 13 and 14 summarized the percentage of classification with respect to each classes for overall data set. Thus, artificial neural network with pattern recognition is able to classify the types of fault even though it is being trained and simulated with healthy data.

#### **4.6 Fault Detection for Drift Data**

In order to detect the drift fault, all the fault data is being filtered out because the presence of faulty data in the data set will disturb the data trend. Thus, when all the faulty data are being removed, the remaining data which is healthy is being fed to PCA model to find its residual. Then, the residual plot is obtained and the trend of the residual over time is observed.

##### *4.6.1 Drift fault detection with normal data*

The data collected is manipulated in order to make the data is concentrated around the mean over the time. This is an indication that there is no drift in the particular set of data as shown in Table 15 . In Figure 28, the data trend shows almost a straight slope where the residuals are evenly distributed across the zero value. Thus, it can be said that there is no drift in the data set.

Table 15: Manipulated normal data for first six hour

Hour	Temperature (°C)	Pressure (kPag)	Volume (m <sup>3</sup> )
1	26	3455.62	39.21484
2	25.91	3502.26	39.07755
3	26.34	3559.12	38.84657
4	28	3588.32	38.64243
5	30.56	3600.58	38.598
6	29.02	3598.28	38.64771

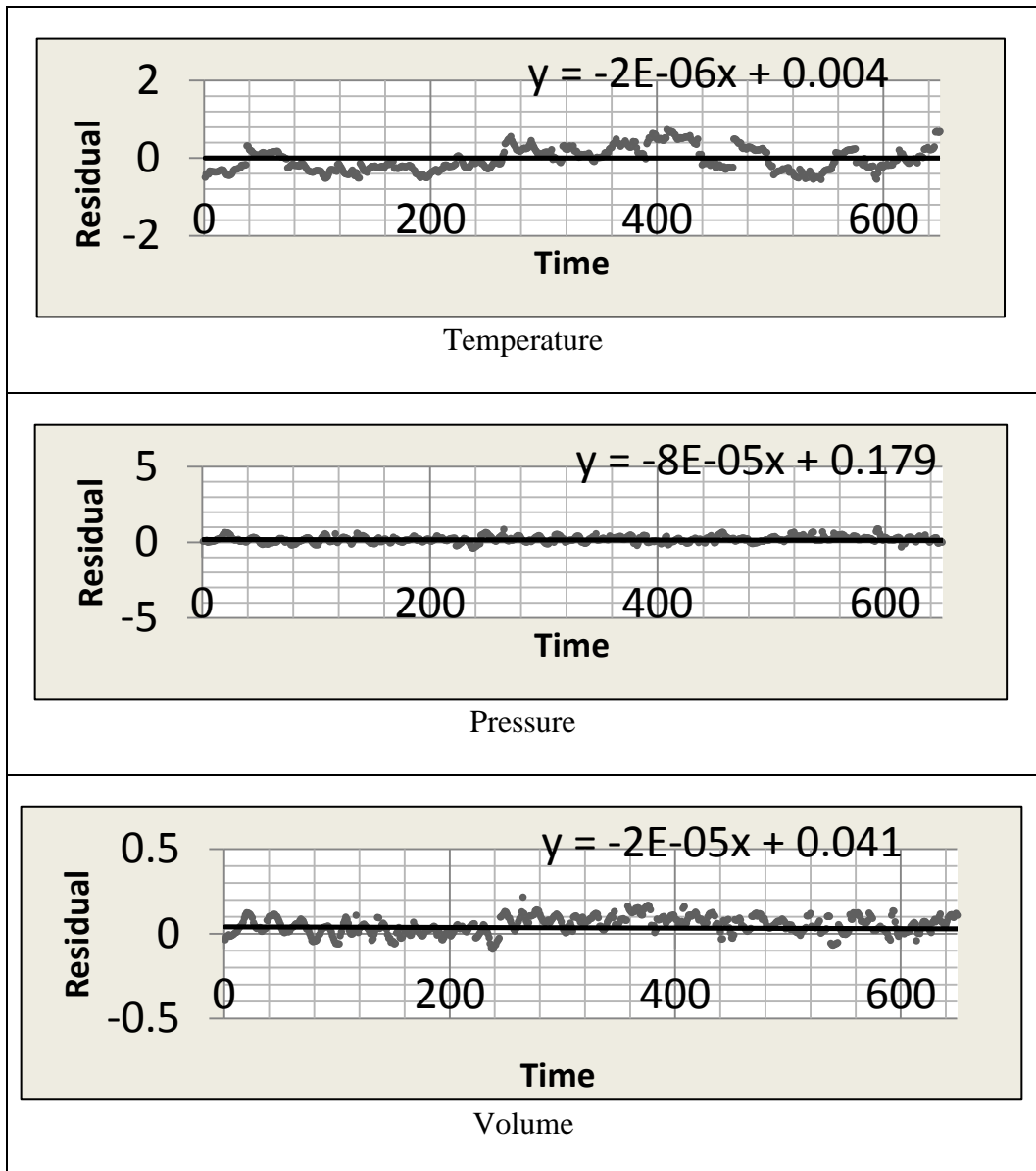


Figure 28: Residual vs time plot

#### 4.6.2 Drift fault detection with manipulation of drift data

Then, the data is then being manipulated in order to observe the residual vs. Time plot trend. The data is manipulated so that it will have an exponential increment over time. A slight increment in each hour of data will eventually contribute to the drifting in the data set. Table 16 displays the manipulated data in hourly sequence. The increment can be expressed as :

$$a_{i+1} = a_i - 0.003$$

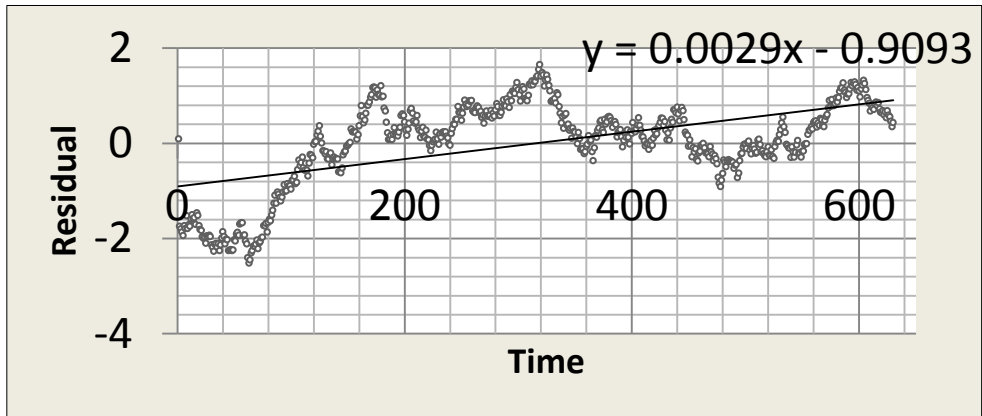
$$b_{i+1} = b_i + e^{a_i}$$

where  $a_i = 5.5$  and  $b_i =$  parameter value at hour 1

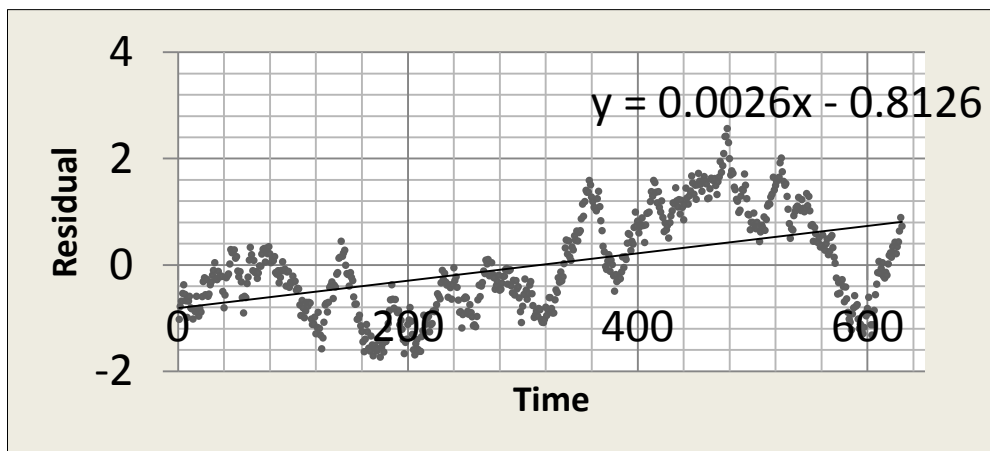
**Table 16: Manipulated drift data for first six hour**

Hour	Temperature (°C)	Pressure (kPag)	Volume (m <sup>3</sup> )
1	26	3455.62	39.21484
2	26.00409	3455.624	39.21893
3	26.00819	3455.628	39.22303
4	26.0123	3455.632	39.22714
5	26.01642	3455.636	39.23126
6	26.02056	3455.641	39.2354

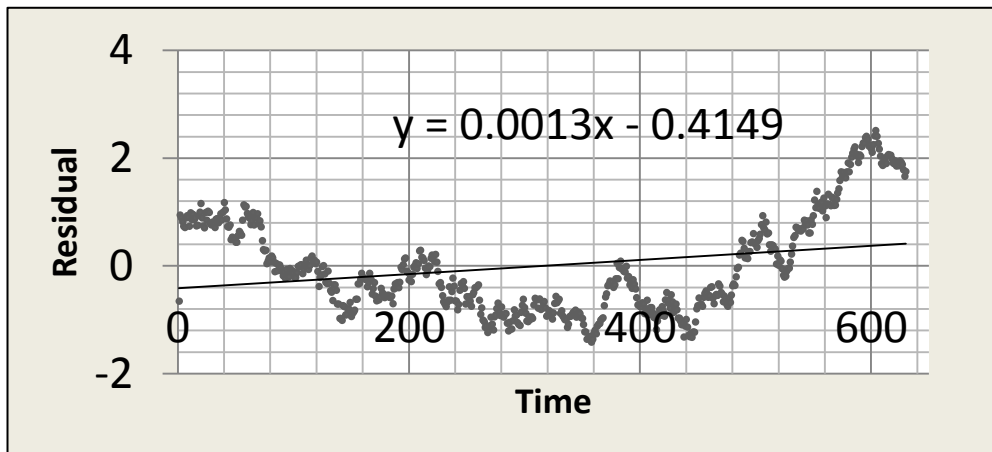
The data is then being fed to the PCA model to obtain the residual and to identify the drift in the data set. It can be seen in Figure 29 that the residuals tend to increase over time which shows that there is a drift in the data set.



Temperature



Pressure



Volume

Figure 29: Residual vs time plot



## **Chapter 5**

### **CONCLUSION**

#### **5.1 Conclusion**

The vast and variation of techniques to detect the fault over the years has increased the reliability of the systems and eventually reduce the major loss of equipment and profit. A lot of past researches had proved that there are several methods which made fault detection become a lot easier. Thus, the best fault detection method should was considered in this project in order to implement in the gas transportation system. Last but not least, good detection techniques will improve the time as well as the accuracy of the detection. In this project, it has been proved that PCA model is able to detect any outlier in the data and hence, providing the information regarding the abnormalities in the data. Meanwhile, artificial neural network also proved that it can classify the faults according to their respective class.

#### **5.2 Recommendation**

The future continuation of this project is to implement the model developed to the fault diagnosis strategy for further improvement. The fault diagnosis can be modelled by using intelligent approach which is neural network or another statistical method such as Fisher Discriminant Analysis which capable in handling non-linear data.

## REFERENCES

- [1] Kandula, V. K. (2011). "FAULT DETECTION IN PROCESS CONTROL PLANTS USING PRINCIPAL COMPONENT ANALYSIS." The Department of Electrical Engineering, VIT University.
- [2] D. Solomatine, L. M. S. a. R. J. A. (2008). Chapter 2, Data-Driven Modelling: Concepts, Approaches and Experiences. *Practical Hydroinformatics. Water Science 17 and Technology Library 68*,.
- [3] Chiang, L. H., Kotanchek, M. E., & Kordon, A. K. (2004). Fault diagnosis based on Fisher discriminant analysis and support vector machines. *Computers & chemical engineering*, 28(8), 1389-1401.
- [4] Yunhong, G., & Yibo, L. (2009, 12-14 Aug. 2009). *Fault Prediction Model Based on Phase Space Reconstruction and Least Squares Support Vector Machines*. Paper presented at the Hybrid Intelligent Systems, 2009. HIS '09. Ninth International Conference on.
- [5] Alaei, H., Salahshoor, K., & Alaei, H. (2013). A new integrated on-line fuzzy clustering and segmentation methodology with adaptive PCA approach for process monitoring and fault detection and diagnosis. *Soft Computing*, 17(3), 345-362. doi: 10.1007/s00500-012-0910-9
- [6] Garcia-Alvarez, D., Fuente, M. J., & Sainz, G. (2011, 5-9 Sept. 2011). *Design of residuals in a model-based Fault Detection and Isolation system using Statistical Process Control techniques*. Paper presented at the Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on.

- [7] Yang, Q. (2004). MODEL-BASED AND DATA DRIVEN FAULT DIAGNOSIS METHODS WITH APPLICATIONS TO PROCESS MONITORING. *Electrical Engineering and Computer Sciences CASE WESTERN RESERVE UNIVERSITY*.
- [8] Smith, L. I. (2002). "A tutorial on Principal Components Analysis." from [https://www.ce.yildiz.edu.tr/personal/songul/file/1097/principal\\_components.pdf](https://www.ce.yildiz.edu.tr/personal/songul/file/1097/principal_components.pdf)
- [9] Nomikos, P., & MacGregor, J. F. (1994). Monitoring batch processes using multiway principal component analysis. *AIChE Journal*, 40(8), 1361-1375.
- [10] Origin Lab (n.d) .16.8.1 Graphic Residual Analysis . Retrieved on June 26, 2014 from <http://www.originlab.com/doc/Origin-Help/Residual-Plot-Analysis>
- [11] Ping, L. H. (2010). "Development of a Fault Prediction and Detection System for Analyzer2 Ver 2." Electrical & Electronics Engineering, Universiti Teknologi PETRONAS.
- [12] Pearson's Correlation Coefficient, r, In University of the West England. Retrieved June 21, 2014 from <http://hsc.uwe.ac.uk/dataanalysis/quantInfAssPear.asp>
- [13] Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., & Yin, K. (2003). A review of process fault detection and diagnosis: Part III: Process history based methods. *Computers & chemical engineering*, 27(3), 327-346.
- [14] Dunia, R., & Joe Qin, S. (1998). Joint diagnosis of process and sensor faults using principal component analysis. *Control Engineering Practice*, 6(4), 457-469.

- [15] *A Closer Look At The Advanced CODAS Moving Average Algorithm.*  
DATAQ INSTRUMENTS from  
[www.dataq.com/support/documentation/pdf/article\\_pdfs/an8.pdf](http://www.dataq.com/support/documentation/pdf/article_pdfs/an8.pdf)
- [16] Jack, L. B., & Nandi, A. K. (2002). FAULT DETECTION USING SUPPORT VECTOR MACHINES AND ARTIFICIAL NEURAL NETWORKS, AUGMENTED BY GENETIC ALGORITHMS. *Mechanical Systems and Signal Processing*, 16(2–3), 373-390. doi:  
<http://dx.doi.org/10.1006/mssp.2001.1454>
- [17] G. Carlos. Artificial Neural Networks for Beginners. from  
<http://arxiv.org/ftp/cs/papers/0308/0308031.pdf>
- [18] Li, B., Zhang, W.-g., Ning, D.-f., & Yin, W. (2007, 5-7 Sept. 2007). *Fault Prediction System Based on Neural Network Model*. Paper presented at the Innovative Computing, Information and Control, 2007. ICICIC '07. Second International Conference on.
- [19] Rahman, R. Z. A., et al. (2013). Model-based fault detection and diagnosis optimization for process control rig. Control Conference (ASCC), 2013 9th Asian.
- [20] Samanta, B., & Al-Balushi, K. R. (2003). Artificial neural network based fault diagnostics of rolling element bearings using time-domain features. *Mechanical systems and signal processing*, 17(2), 317-328.

## APPENDICES

### APPENDIX I – Fault Detection Source Code

```
tic
clc
clear
%% -----raw data load-----%

X = xlsread('C:\Users\User\Documents\Final
Final\FYP2\data6410driftmod','A1:C6408');

%load('PCAFAULTDRIFT.mat');
trainData1=X(:,:);
totale1=numel(trainData1);
%checkData1=X(2870:5738, : );
[k,1]=size(trainData1);
t=1:k;

figure(1)
subplot (3,1,1);
plot(trainData1(:,1))
title('Temperature')
line('xData', [0 k], 'yData', [20 20], 'LineStyle', '--',
'LineWidth', 2, 'Color', 'r');
line('xData', [0 k], 'yData', [35 35], 'LineStyle', '--',
'LineWidth', 2, 'Color', 'r');
ylim([15 40])

subplot (3,1,2);
plot(trainData1(:,2))
title('Pressure')
line('xData', [0 k], 'yData', [2000 2000], 'LineStyle', '--',
'LineWidth', 2, 'Color', 'r');
line('xData', [0 k], 'yData', [6500 6500], 'LineStyle', '--',
'LineWidth', 2, 'Color', 'r');
ylim([1900 7000])

subplot (3,1,3);
plot(trainData1(:,3))
title('Volume')
line('xData', [0 k], 'yData', [35 35], 'LineStyle', '--',
'LineWidth', 2, 'Color', 'r');
line('xData', [0 k], 'yData', [42 42], 'LineStyle', '--',
'LineWidth', 2, 'Color', 'r');
ylim([34 45])

%% -----fault,zero and missing data filter-----%
zero=0;
missing=0;
%missing data
missingT=isnan(trainData1(:,1));
missT=0;
for i=1:k
```

```

        if missingT(i)==1
            missT=missT+1;
        end

    end

    end
    strtrim(sprintf('total temperature missing are %d', missT))

    missingP=isnan(trainData1(:,2));

    missP=0;
    for i=1:k

        if missingP(i)==1
            missP=missP+1;
        end

    end

    end
    strtrim(sprintf('total pressure missing are %d', missP))
    missingV=isnan(trainData1(:,3));
    missV=0;
    for i=1:k

        if missingV(i)==1
            missV=missV+1;
        end

    end

    end
    strtrim(sprintf('total volume missing are %d', missV))

    for i=k:-1:1
        for j=1:l
            if isnan(trainData1(i,j))

                trainData1(i,:) =0;

            end

        end
    end
    %zero fault
    for i=k:-1:1
        for j=1:l-1
            if trainData1(i,j)==0

                strtrim(sprintf('data zero at hour %d for parameter %d',
i,j))
                zero(i,j)=trainData1(i,j);
            else
                zero(i,j)=NaN;
                % trainData1(i,:) =[];
                % k=k-1;
            end
        end
    end
end

```

```

end
strtrim(sprintf('total zero faults are %d', zero))
percentzero=(zero/totalel)*100;
filename='C:\Users\User\Documents\Final Final\FYP2\zeros.xlsx';
xlswrite(filename,zero)

%hang fault

for i=1:1:k-2
    for j=1:1:l
        if trainData1(i,j)== 0 && trainData1(i+1,j) ==0
&&trainData1(i+2,j) ==0
            hang(i:i+2,j) =NaN;

            else if
trainData1(i,j)==trainData1(i+1,j)&&trainData1(i+1,j)==trainData1(i+
2,j)

                strtrim(sprintf('data hang at %d hour %d and %d and %d',
trainData1(i,j),i,i+1,i+2))

                hang(i:i+2,j)=trainData1(i:i+2,j);

            else

                hang(1,j)=NaN;
                hang(2,j)=NaN;
                hang(i+2,j)=NaN;

            end
            if trainData1(1,j)==hang(2,j)==hang(3,j)
                hang(1,j)=hang(2,j);

                if trainData1(1,j)==0 && hang(2,j)== 0 && hang(3,j)==0
                    hang(1,j)=[];
                end
            end
        end
    end
end
end

H1=hang(:,1);
one=H1(find(H1>0));
totalT=numel(one);
H2=hang(:,2);
two=H2(find(H2>0));
totalP=numel(two);
H3=hang(:,3);
three=H3(find(H3>0));
totalV=numel(three);

strtrim(sprintf('total hang faults are %d', totalT+totalP+totalV))
percenthang=((totalT+totalP+totalV)/totalel)*100

filename='C:\Users\User\Documents\Final Final\FYP2\hangSs.xlsx';
xlswrite(filename,hang)

```

```

[m,n]=size(trainData1);
%% PCA Algorithm
Xmean=mean(trainData1);
sd=std(trainData1);
for i=1:m
    for j=1:n
        zeromean(i,j)= trainData1(i,j)-Xmean(1,j);
    end
end
for i=1:n, %normalization loop

norm_column=(trainData1(:,i)-Xmean(i))./sd(i);
norm_X(:,i)=norm_column;
end

figure(2)
for i=1:3,
subplot (3,1,i);
plot(norm_X(:,i))
title('Normalize Data')
line('xData', [0 k], 'yData', [3 3], 'LineStyle', '--', ...
'LineWidth', 2, 'Color','r');
line('xData', [0 k], 'yData', [-3 -3], 'LineStyle', '--',
'LineWidth', 2, 'Color','r');
    if i==1
        title('Temperature')

    else if i==2
        title('Pressure')
    else
        title('Volume')
    end

end

end

%% Score,loading vector and residual generation
i=0;
[coeff,score,latent,tsquare,explained,mu] = princomp(norm_X);

for i=1:3, %convert latent to square matrix
latent_mat(i,i)=latent(i,1);
end
i=0;
mn_score=mean(score); %mean of score matrix
sd_score=std(score); %standard deviation of score matrix

[score_row, score_column]=size(score);
for i=1:score_column, %normalization loop
nSCORE_column=(score(:,i)-mn_score(i))./sd_score(i);

```



```

score_norm(:,i)=nSCORE_column;
end

i=0;
no_princomp=1;
score=score(:,1:no_princomp);
score_square=(score_norm.^2);

for i=1:n
column_t(i,:)=score_square(i,1:no_princomp)./(latent(1:no_princomp,1)
)');
tsquare(i,1)=sum(column_t(i,:));
end

trainData1_final=score_norm*(coeff)';

r=norm_X-trainData1_final;
i=0;

inputf1=trainData1_final(:,1);
inputf2=trainData1_final(:,2);
inputf3=trainData1_final(:,3);

figure(3)
plot(trainData1_final)
legend('X1=temperature','X2=pressure','X3=volume')
title('X_final')

figure(4)
plot(r)
legend('X1=temperature','X2=pressure','X3=volume')
title('Residual Generated')

figure(5)
subplot(3,1,1);
title('Residual Generated for each parameter')
plot(r(:,1))
ylim([-3 3])
title('Temperature')
subplot(3,1,2);
plot(r(:,2))
ylim([-3 3])
title('Pressure')
subplot(3,1,3);
plot(r(:,3))
ylim([-3 3])
title('Volume')

filename='C:\Users\User\Documents\Final Final\FYP2\Res.xlsx';
xlswrite(filename,r)

save('norm_X.mat','Xmean','sd','norm_X','-append');
save('residual.mat','Xmean','sd','-append');
save('trainData1.mat','trainData1','-append');

q=r.*r;

```

```

[r_row, r_column]=size(r);
SPE=sum(q');

figure(6)
plot (SPE) %SPE Chart
chisquare_99=chi2inv(0.99,1-1);
chisquare_95=chi2inv(0.95,1-1);
theta1=sum(latent((no_princomp+1):1,1));
theta2=sum(latent((no_princomp+1):1,1).^2);
theta3=sum(latent((no_princomp+1):1,1).^3);
g=theta2/theta1;
h=(theta1^2)/theta2;
h0=1-((2*theta1*theta3)/(3*(theta2^2)));
z_99=norminv(1-0.01);
z_95=norminv(1-0.05);
SPE_threshold99=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_99*((2*theta2*(h0^2))^0.5))/(theta1)))^(1/h0)); %SPE limit
alternative 1
SPE_threshold95=theta1*((1-(theta2*h0*(1-h0)/(theta1^2)) +
((z_95*((2*theta2*(h0^2))^0.5))/(theta1)))^(1/h0));

line('XData', [0 k], 'YData', [SPE_threshold99 SPE_threshold99],
'LineStyle', '-', ...
'LineWidth', 2, 'Color', 'r');
line('xData', [0 k], 'yData', [SPE_threshold95 SPE_threshold95],
'LineStyle', '--', ...
'LineWidth', 2, 'Color', 'y');
legend('residual', '99.0% confidence limit', '95.0% confidence
limit', ...
'Location', 'NorthEastOutside')
title('SPE-Chart', 'FontWeight', 'bold')
xlabel('sample number')
ylabel('residual')
finv_99=finv(0.99,no_princomp,(score_row-no_princomp)); %F alpha
(no_princomp, (no of sample - no_princomp))
finv_95=finv(0.95,no_princomp,(score_row-no_princomp)); %F alpha
(no_princomp, (no of sample - no_princomp))
thold_99=((no_princomp)*(score_row-
1)*(score_row+1))/(score_row*(score_row-no_princomp))*finv_99;
thold_95=(no_princomp)*(score_row-
1)*(score_row+1)/(score_row*(score_row-no_princomp))*finv_95;
%subplot (2,1,2);

figure(7)
plot(tsquare) %T2 chart
line('XData', [0 k], 'YData', [thold_99 thold_99], 'LineStyle', '-',
...
'LineWidth', 2, 'Color', 'r');
line('xData', [0 k], 'yData', [thold_95 thold_95], 'LineStyle', '--
', ...
'LineWidth', 2, 'Color', 'y');
legend('T-square', '99.0% confidence limit', '95.0% confidence
limit', ...
'Location', 'NorthEastOutside')
title('T-square Chart', 'FontWeight', 'bold')
xlabel('sample number')
ylabel('T2')

filename='C:\Users\User\Documents\Final Final\FYP2\finaldatas.xlsx';
xlswrite(filename,trainData1_final)

```

```
mapcaplot(score_norm)
cumsumX=cumsum(latent)./sum(latent);
figure(8)
biplot(coeff(:,1:2), 'Scores', coeff(:,1:2), 'VarLabels', ...
        {'X1' 'X2'}
```

## APPENDIX II – Fault Classification Source Code

```
%% ----- %  
%                               data load  
% ----- %  
clear  
clc  
load('inputnn.mat');  
load('targetnn.mat');  
load('healthy.mat');  
load('testtarget.mat');  
load('net.mat');  
load('net.mat');  
  
inputs = inputnn';  
targets = targetnn';  
tests = healthy';  
targettest=testtarget';  
neuron_1=4;  
neuron_2=10;  
  
%% ----- %  
%                               data division  
% ----- %  
  
numofdata=size(inputs,2);  
  
%check even or odd  
if rem(numofdata,2)==0;  
    numdata=numofdata-2;  
else  
    numdata=numofdata-3;  
end  
  
train_data=0.5*numdata;  
if rem(train_data,2)==0;  
    trainB=train_data-2;  
else  
    trainB=train_data-3;  
end  
  
validation_data=0.5*trainB;  
%test_data=numofdata-train_data-validation_data;  
  
test_data=0.5*trainB;  
numofvar=size(inputs,1);  
numofout=size(targets,1);  
  
for m=1:numofvar  
    for n=1:train_data  
        x_t(m,n)=inputs(m,n);  
    end  
end
```

```

end

for m=1:numofvar
    for n=1:validation_data
        x_v(m,n)=inputs(m,n+train_data);
    end
end

for m=1:numofvar
    for n=1:test_data
        x_te(m,n)=inputs(m,n+train_data+validation_data);
    end
end

for m=1:numofout
    for n=1:train_data
        y_t(m,n)=targets(m,n);
    end
end

for m=1:numofout
    for n=1:validation_data
        y_v(m,n)=targets(m,n+train_data);
    end
end

for m=1:numofout
    for n=1:test_data
        y_te(m,n)=targets(m,n+train_data+validation_data);
    end
end

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivide
%net.divideFcn = 'dividerand'; % Divide data randomly
%net.divideMode = 'sample'; % Divide up every sample
%net.divideParam.trainRatio = 50/100;
%net.divideParam.valRatio = 25/100;
%net.divideParam.testRatio = 25/100;

%% -----%
%                               network properties
% -----%

net=patternnet(10,'trainbr');
net=newff(x_t,y_t,neuron_1,{'tansig','purelin'},'trainbr');

net.trainParam.show=50;
net.trainParam.lr=0.001;
net.trainParam.epochs=1000;
net.trainParam.goal=0.001;

net1.layers{1}.initFcn='initnw';
net.inputs{1}.processFcns = {'mapminmax'};
net.outputs{1}.processFcns = {'mapminmax'};
net.performFcn = 'mse'; % Mean squared error

```

```

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
net.plotFcns = {'plotperform', 'plottrainstate', 'ploterrhist', ...
    'plotregression', 'plotfit', 'plotconfusion'};

%checking the weights and biases (make sure all are 0)
net.IW{1,1}; %weight of first layer
%net.LW{2,1}; %weight of second layer
net.b{1}; %bias of first layer
%net.b{2}; %bias of second layer

%% -----%
%           network training and simulation with faulty data
% -----%

net.init(net);
[net,tr] = train(net,x_t,y_t); % train the network

ytrain=sim(net,x_t); %simulate the network
yvalid=sim(net,x_v); %simulate the network
ytest=sim(net,x_te);

etrain=y_t-ytrain;
evalid=y_v-yvalid;

%yttotal=ytrain+yvalid+ytest;

figure,
plotconfusion(y_t,ytrain,'train',y_te,ytest,'test',y_v,yvalid,'valid
')

%% -----%
%           data division for healthy
% -----%

%check even or odd
numofdata1=size(tests,2);
if rem(numofdata1,2)==0;
    numdata1=numofdata1-2;
else
    numdata1=numofdata1-3;
end

train_data1=0.5*numdata1;

if rem(train_data1,2)==0;
    train1=train_data1-2;
else
    train1=train_data1-3;
end

validation_data1=(0.5*train1);
test_data1=0.5*train1;

```

```

numofvar1=size(tests,1);
numofout1=size(targettest,1);
for m=1:numofvar1
    for n=1:train_data1
        x_t1(m,n)=tests(m,n);
    end
end

for m=1:numofvar1
    for n=1:validation_data1
        x_v1(m,n)=tests(m,n+train_data1);
    end
end

for m=1:numofvar1
    for n=1:test_data1
        x_tel(m,n)=tests(m,n+train_data1+validation_data1);
    end
end

for m=1:numofout1
    for n=1:train_data1
        y_t1(m,n)=targettest(m,n);
    end
end

for m=1:numofout1
    for n=1:validation_data1
        y_v1(m,n)=targettest(m,n+train_data1);
    end
end

for m=1:numofout1
    for n=1:test_data1
        y_tel(m,n)=targettest(m,n+train_data1+validation_data1);
    end
end

%% -----%
%     network training and simulation with healthy data
%     -----%

net.init(net);
[nets,trs] = train(net,x_t1,y_t1); % train the network

ytrain1=sim(net,x_t1); %simulate the network
yvalid1=sim(net,x_v1); %simulate the network
ytest1=sim(net,x_tel);
%ytotall=ytrain1+yvalid1;
figure,
plotconfusion(y_t1,ytrain1,'train',y_v1,yvalid1,'valid',y_tel,ytest1
,'test')
title('healthy data')

etrain1=y_t1-ytrain1;
evalid1=y_v1-yvalid1;

```

```

%% -----%
%                               error calculation
% -----%

%for validation data
rmse_valid=sqrt(mse(evalid))
percenterror_valid=mean((abs(y_v-yvalid)/y_v)*100)

%for training data
rmse_train=sqrt(mse(etrain))
percenterror_train=mean((abs(y_t-ytrain)/y_t)*100)

%for validation data
rmse_valid1=sqrt(mse(evalid1))
percenterror_valid1=mean((abs(y_v1-yvalid1)/y_v1)*100)

%for training data
rmse_train1=sqrt(mse(etrain1))
percenterror_train1=mean((abs(y_t1-ytrain1)/y_t1)*100)

%% -----plot graph-----

% plot the difference between the actual and predicted from training
data
figure
%subplot(2,1,2);
plot(evalid,'*');
xlabel('No of data');
ylabel('Error ');
title('Error between actual and predicted class (Validation Data)');
legend('Class1','Class2','Class3','Class4','Class5');
grid on;

% plot the difference between the actual and predicted from training
data
figure
%subplot(2,1,2);
plot(etrain,'*');
xlabel('No of data');
ylabel('Error ');
title('Error between actual and predicted class (Training Data)');
legend('Class1','Class2','Class3','Class4','Class5');
grid on;

end

```