Development of GC Analyzer Model Using Neural Network

By

Areej Babiker Idris Babiker

A project dissertation submitted to the

Electrical & Electronic Engineering Programme

Universiti Teknologi PETRONAS

in partial fulfilment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL & ELECTRONICS ENGINEERING)

JANUARY 2011

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Development of GC Analyzer Model Using Neural Network

by

Areej Babiker Idris Babiker

A project dissertation submitted to the
Electrical and Electronic Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL AND ELECTRONIC ENGINEERING)

Approved by,

____
(Rosdiazli Bin Ibrahim)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
January 2011

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

AREEJ BABIKER IDRIS BABIKER

ACKNOWLEDGMENTS

Praise be to Allah, The Most Gracious and The Most Merciful for His endless blessings throughout my life and the success He granted me during this Final Year Project.

My deepest heart gratitude is to my parents who strived to get me where I'm now and for their endless support and encouragement during this final year.

My utmost appreciation and gratitude towards my supervisor Dr. Rosdiazli Bin Ibrahim, for the dedication of his time and effort, teaching, guiding and helping me in all my work-related tasks despite his many other obligations. My gratitude is extended to Mohamed Adnan, Maryam Jamela and other master students for giving advice whenever it was needed.

Last but not least, I thank my friends and everyone else who contributed in a way or another towards successful completion of this project.

ABSTRACT

Gas chromatography (GC) is the most widely used technique in analytical chemistry. It is an analytical scientific technique to separate a mixture of vaporizable substances and resolve the mixture into single components. Analyzer as hardware has high initial cost, requires frequent maintenance and sometimes fails to provide the accurate outputs. Moreover, the increasing complexity of industrial processes and the struggle for cost reduction, availability, safety and higher profitability requires efficient and reliable instruments. Thus, this final year project is an attempt to develop prototype software that is capable of predicting efficiently plant output and optimize the performance of the model. MATLAB Neural Network and system identification toolboxes were utilized to recommend the best structure to develop this predictive model. The purpose of this report is to show the success and applicability of using neural network in predicting plant output and obtain an alternative measuring system. It presents the followed methodology in achieving project's objectives by giving an overview on neural network and system identification toolboxes and shows a comparison of the performance of Back Propagation Feed Forward Neural Network (BFN) and other System I identification toolbox models. Results demonstrated that neural network model trained using LM provides an adequate result and is suitable for this purposes.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background of Study

Gas chromatography (GC) is the most widely used technique in analytical chemistry which is apposition it has held for over three decades. The popularity and applicability of the technique is principally due to its unchallenged resolving poser for closely related volatile compounds and because of the high sensitivity and selectivity offered by many of the detector systems. The technique is very accurate and precise when used in a routine laboratory as in [1]. However, maintenance costs of such devices are widely recognized as a significant contributory factor in the life cycle cost of a process plant, this proposed predictive tool will help avoiding unnecessary maintenance. Basically, maintenance can be divided into:

- Corrective maintenance which is defined as the maintenance carried out after fault recognition [2].
- Breakdown maintenance which is defined as the maintenance carried out after a failure has occurred [2].
- Preventive maintenance which is the performance of inspection and/or servicing tasks that have been preplanned for accomplishment at specific points in time to retain the functional capabilities of operating equipment and systems [3].

- Predictive maintenance which is a maintenance activity geared to indicating that piece of equipment is on the critical wear curve and predicting its useful life [4].

As known, maintenance after fault occurs has higher cost than maintenance before it occurs. This is due to shutdown of system or facilities on one hand and replacement of damaged equipments on the other hand. The above definitions showed types of maintenance that needed to be done to the system either frequently or when fault occurs. The proposed software predictive tool will predict plant output without the need for GC analyzer besides preventing unneeded maintenance and so would significantly lower maintenance costs and help achieving efficient process control.

A solution was proposed by PETRONAS Penapisan Terengganu (PPTSB) with assistance of Universiti Teknologi PETRONAS (UTP) to overcome failure in GC analyzer using the initiated and developed software called Analyzer². This software is capable of predicting GC analyzer faults before they actually. Though, it's not yet tested besides its requirements of frequent maintenance. Thus, the need for more reliable and efficient tool arises. This was met using the proposed predictive tool software in this project that eliminates the need for both GC analyzer and Analyzer² software and increase the reliability and functionality of the plant. The software was developed using MATLAB Neural Network and system identification toolboxes.

1.2 Problem Statement

As Gas chromatography (GC) became a requirement in chemical plants nowadays, it is necessary to ensure its functionality and accurate results all the time. . On one hand, valuable time is spent in performing some of the early mentioned maintenance types and most of the time no critical need for this maintenance. On the other hand, if maintenance is needed out of the schedule and was not performed properly, equipment failure might occur causing unnecessary shutdown of facilities and so decrease in plant productivity.

Unnecessary frequent maintenance is costly. The facility's shutdown also will decrease the plant reliability and thus will contribute to the downturn of company's profit and margin.

This proposed prototype will not save time and cost only; it will also help improving the plant reliability and productivity by ensuring accurate result and that the instrument is always in good condition. The proposed model is cheaper, user friendly and does not require deep knowledge or operator training. Hence, an attempt to develop simple, easy-to-operate prototype software and ready-to-run model include everything needed to produce high-quality predicted data is incorporated in this project.

1.3 Objective and Scope of Study

This study is carried out to:

- Develop predictive model for GC analyzer.
- Analyze and optimize the performance of the model.
- Recommend the best model structure for the prediction model.

CHAPTER 2

LITERATURE REVIEW

2.1 Gas Chromatography

Gas Chromatography (GC) is a common technique employed to separate and measure components of process streams. Since it has thousands of applications, this technology is widely used for manufacturing operations of almost every industry.

The GC instrument constructed by James and Martin 50 years ago contained most of the features of a modern gas chromatograph a means of controlling the flow of mobile-phase carrier gas, stabilization of the temperature of the column, and a sensitive detector to determine and record the concentrations of separated constituents at the end of the column. These pioneers also introduced the concept of separation efficiency and discussed the influence of parameters such as gas flow rate and diffusion of the sample in the mobile phase. [5]

The main advantage of GC is because of its resolving power, speed and small sample size requirement. The technique used is for separating substances in the vapor state then quantifying the separated substances. GC combines both quantities and semi qualitative methods. The sample must be vaporizable at the temperature of the GC analyzer oven and in either the liquid or gaseous state when injected into the analyzer. Figure 1 below shows the schematic diagram of Gas Chromatography.

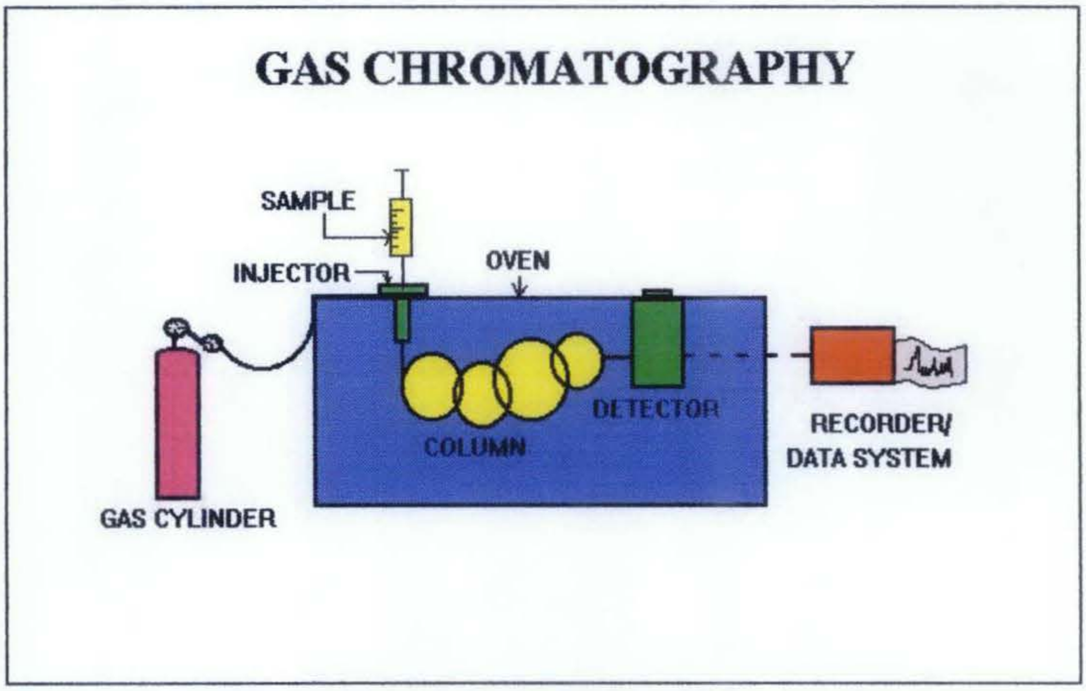

Figure 1 Schematic diagram of Gas Chromatography

The basic output of Gas Chromatography process is concentration. Since GC can measure multiple components, it can represent a concentration of any value from parts per billion to 100%. GC process is suitable for any application where the sample components of interest are vaporizable, sufficiently separable on GC column and measurable on compatible detector. The sample size, the column and the detector can limit the range of output. The detector needs to see sample concentration in the carrier gas that exceeds its level of detection and is less than the level that results in detector saturation. Injecting too large sample into the column can overload it, preventing the normal interaction between the mobile and the stationary phases from taking place. This will result in bad separation. All leading instrument manufacturers produce and market gas chromatographs. In addition, many smaller specialty companies also manufacture and market GC units. [6]

2.2 Analytical Instrumentation

One of the important areas where expert systems have emerged is the field of condition monitoring and fault prediction and analysis. These expert systems are able to combine human experience and physical laws to determine fault patterns. Simultaneously, technical, economical and time constraints on production efficiency, safety, availability, reliability and quality require that this condition monitoring and fault prediction be undertaken on a continuous basis.

The development of instrument techniques for qualitative and quantitative chemical analysis began earlier in the 20th century; this activity was restricted to scientific research laboratories. Early analytical instrument were designed and fabricated by the researches themselves for their own used only. The early, traditional approach to implementing Analyzer was to modify selected laboratory analytical technology to create instrumentation suitable to production plant environment.

Nowadays, analyzer is used almost in every process plant regardless of the way. It plays an important role in ensuring the safety of personnel and equipment.

In general, analyzer can be divided into three categories:

1. Instruments that can measure physical properties of the chain compounds.
2. Instrument that can automate procedures of analytical chemistry.
3. Instrument that can measure the quantitative separation of mixture [7].

Practically, the time used in performing maintenance can be minimized if reliable predictive tool is utilized. Reliability is defined as the probability that an item (component, equipment or system) will operate without failure for a stated period of time under specified conditions [8].

Therefore, instead of maintaining the equipment twice a year, it will be maintained only if fault is predicted. This will lower maintenance cost and increase system/plant productivity.

Some important criteria need to be taken into consideration when designing an analyzer e.g. Packaging, Safety, Utilities, Ambient Temperature, Analyzer Temperature, Materials of Construction, Standardization, Maintenance and Operability, Sensitivity, Repeatability of Reading Accuracy, Reliability and Data Handling System.

However, the proposed predictive tool in this project shall be able to predict the plant output without GC analyzer of analyzer^2 software. This ensures reliable, efficient and healthy analyzer is available when needed.

2.3 Analyzer^2 Prototype Software

As stated earlier, Analyzer^2 prototype software is designed by Engineer Azrin Sani and his team from PETRONAS Penapisan Terengganu Sdn Bhd (PPTSB) [9]. The 2nd version of this analyzer was developed by Universiti Teknologi PETRONAS (UTP) to enhance the system reliability and functionality. The purpose of this design basically is to facilitate predictive maintenance and to act as fault prediction and detection tool. This prototype is capable of detecting six pattern recognition algorithms that will predict the behavior of particular instrument:

1. Moment Correlation Algorithm.
2. High & Low Fluctuation Level Algorithm.
3. High & Low Fluctuation Period Algorithm.
4. Spike Detection Algorithm.
5. Moving Average Algorithm.
6. Average Deviation Algorithm.

The final output of Analyzer² algorithm is confidence level indicator that shows how confidence that the analyzer is healthy. The range is between 0% and 100%. If let say the indicator reads 80% confidence that the instrument is healthy. In term of predictive maintenance, the confidence level will highlight the performance of instrument in a certain period of time and it should prompt maintenance team to check the instrument if the indicator drops below 95% [10].

The project will develop predictive tool that shall replace GC analyzer and analyzer² software. This tool is designed using Neural Network and system identification toolboxes in MATLAB software. The demand for the use of Artificial Neural Networks to solve engineering problems is expected to increase significantly in the next ten years, mainly due to several breakthroughs in this field and also to the limitations of the existing conventional engineering problem solving techniques. Results to date have demonstrated the significant performance advantages of Artificial Neural Networks relative to currently available conventional methods [11].

2.4 Methods

For this project, Neural Network and system identification toolboxes will be used. The designed system should act as predictive tool. More explanation is carried in the next sections.

2.3.1 Neural Network

The first use of artificial neural networks can be dated back to the 1940s. Since then, many different neural network paradigms have been developed during the past few decades. Each paradigm has its own specific internal network, structure, properties and training algorithms that are unique and useful for a particular range of applications [12].

Neural networks have seen an explosion of interest and are being successfully applied across an extraordinary range of problem domains, in areas as diverse as finance, medicine, engineering, geology and physics. Indeed, anywhere that there are problems of prediction, classification or control, neural networks are being introduced. This sweeping success can be attributed to a few key factors:

- *Power*: Neural networks are very sophisticated modeling techniques capable of modeling extremely complex functions. In particular, neural networks are nonlinear.
- *Ease of use:* Neural networks learn by example. The neural network user gathers representative data, and then invokes training algorithms to automatically learn the structure of the data. Although the user does need to have some heuristic knowledge of how to select and prepare data, how to select an appropriate neural network, and how to interpret the results, the level of user knowledge needed to successfully apply neural networks is much lower than would be the case using for example some more traditional nonlinear statistical methods [13]. Refer to Figure 2 below.

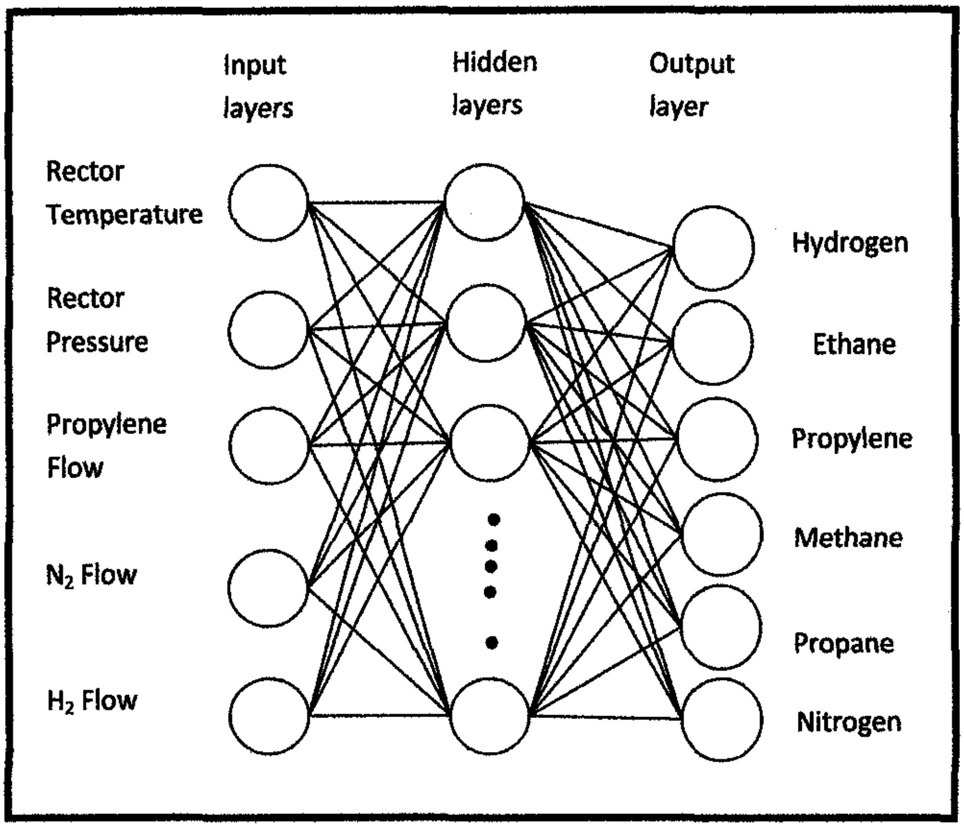

Figure 2: Project Neural Network Model

This figure shows the basic elements of neural network which are input layer, hidden layer and output layer. Inputs and outputs can be of any numbers depending on the complexity of the network. However, the number of hidden layers or processing elements per layer depends on the inputs and outputs number. Those are known as the "art" of the network designer. There is no quantifiable, best answer to the layout of the network for any particular application. On one hand, if the complexity in the relationship between the input data and the desired output increases; the number of the processing elements in the hidden layer should also increase. On the other hand, if the process being modeled is separable into multiple stages, then additional hidden layer(s) may be required but if it is not separable, then additional layers may simply enable memorization of the training set, and not a true general solution effective with other data. For this project 5 inputs and 1 output were used first to develop the model then the model developed to have 5 inputs and 6 outputs as indicted in figure 2 above. The number of the neurons in the hidden layers was determined based on trial and error approach.

In fact, neural network is one of the approaches to do data-driven modeling. Neural network, with its remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. It's consisting of many units i.e. processing unit's analogues to neutrons in the brain. Each node has a node function, associated with it which along with a set of local parameters determines the output of the node, given an input [14]. However, MATLAB Neural Network Toolbox provides comprehensive support for design, implementation, visualization, and simulation.

The type of problem amenable to solution by a neural network is defined by the way they work and the way they are trained. Neural networks work by feeding in some input variables, and producing some output variables. They can therefore be used where you have some known information, and would like to infer some unknown information [15], [16].

After deciding on the problem to be solved using neural networks, data are gathered for analysis then training purposes. The training data set might include numbers of cases, each containing values for a range of input and output variables. The choice of variables is guided by intuition. As a first pass, these data should include variables that have an influence in solving the problem. The beauty about Neural Network is that one doesn't need to know the exact nature of the relationship between inputs and outputs.

The input data are divided into three groups: first group is for training, second group is for prediction, and third is for fraud detection. The network is trained using the training data. Then the second set of data is used for prediction, and later the third ser of data is used for the fraud detection [17].

During training, learning algorithm is accomplished in the following manner. A pattern X is applied to the input of perceptron, and output Y is calculated. If the output is correct (i.e. corresponds to the desired one), the weights are not changed. If not, the weights, corresponding to input connections that cause this incorrect result, are modified to reduce the error [18].

2.3.2 System Identification toolbox

System identification is a methodology for building mathematical models of dynamic systems using measurements of the system's input and output signals.
The process of system identification requires that one:

- Measure the input and output signals from your system in time or frequency domain.
- Select a model structure.
- Apply an estimation method to estimate value for the adjustable parameters in the candidate model structure.
- Evaluate the estimated model to see if the model is adequate for your application needs.

System identification uses the input and output signals you measure from a system to estimate the values of adjustable parameters in a given model structure [19]. Four models under system identification toolbox were used based on the data requirements. The models used were ARX, ARMAX, State-Space and low-order transfer function (Process Model). In those techniques the system is identified by estimating the parameters of the ARX model using input-output data. System identification toolbox was used to get the appropriate models that best fits the data and then M-file was written to save those models and so enable using it in predicting other month's data. Refer to figure 3 below

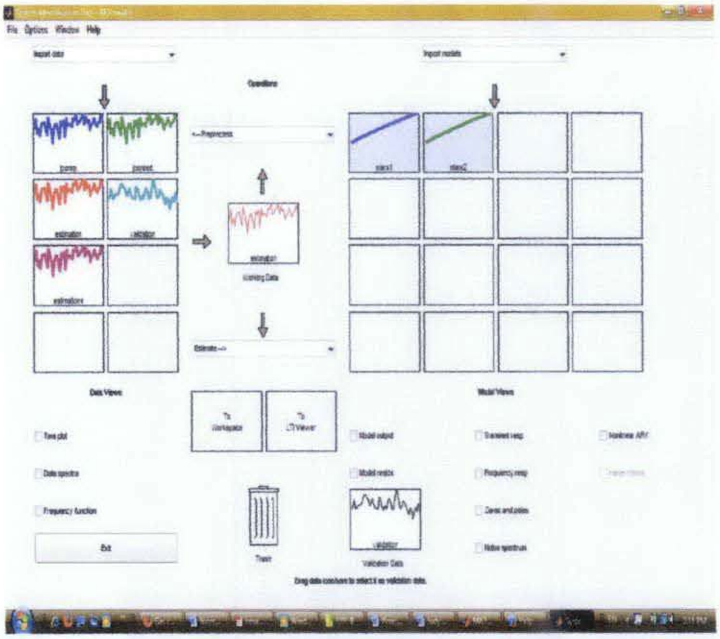

Figure 3 System Identification toolbox

2.3.2.1 ARX model

The term used

```
arx_order=[3 [3 3 3 3 3] [0 0 0 0 0]]
        = [na [nb][nk]]
```

The parameters of the ARX model structure:

The parameters na and nb are the orders of the ARX model, and nk is the delay.

The following equation shows the form of the ARX model.

$A(z)y(k) = B(z)u(k - n) + e(k)$

where $u(k)$ is the system inputs

$y(k)$ is the system outputs

n is the system delay

$e(k)$ is the system disturbance

$A(z)$ and $B(z)$ are polynomial with respect to the backward shift operator z^{-1} and defined by the following equations.

2.3.2.2 ARMAX model

The term used

```
armax_order= [3 [3 3 3 3 3] 3 [0 0 0 0 0]];
```

= [na nb nc nk]

This is a linear input-output polynomial model. The model structure is as following

$y(t) + a1y(t-1) + \ldots\ldots\ldots + a_{nz}y(t-n_a)=$

$b_{1u}(t-n_k)+\ldots\ldots\ldots+ b_{nu}(t-n_k-n_b+1) + e(t) +c_{1e}(t-1)+\ldots+c_{nc}e(t-n_c)$

y(t) represents the output at time t

u(t) represents the input at time t

na isthe number of poles for the dynamic model

nb is the number of zeros plus 1

nc is the number of poles for the disturbance model

nk is the number of samples before the input affects output of the system (called the delay or dead time ofthe model)

e(t) is the white-noise disturbance [19].

2.3.2.3 State Space

The term used

```
statespace_order=2
```

The State-Space block implements a system whose behavior is defined by:

X= Ax + Bu

Y= Cx + Du

Where x is the state vector, u is the input vector and y is the output vector. The model used has order of 2. It has unforced response starting from given initial states.

2.3.2.3 low-order transfer function (Process Model)

The term used

```
process_model='P2'
```

Continuous-time process models are low-order transfer functions that describe the system dynamics using static gain, a time delay before the system output responds to the input, and characteristic time constants associated with poles and zeros. Such models are popular in the industry and are often used for tuning PID controllers, for example. Process model parameters have physical significance.

One can specify different process model structures by varying the number of poles, adding an integrator, or including a time delay or a zero. The highest process model order you can specify in this toolbox is three, and the poles can be real or complex (underdamped modes) [19].

CHAPTER 3

METHODOLOGY

3.1 Procedure Identification

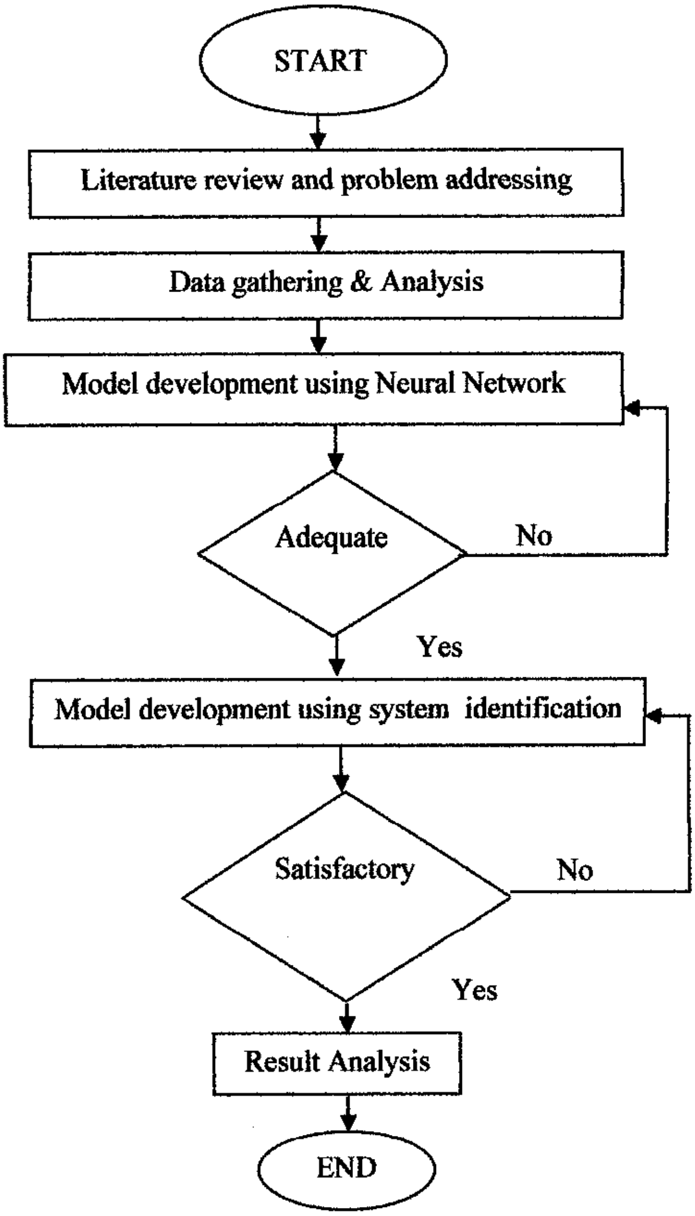

Figure 4: Project Flow Chart.

The final year project methodology is to be conducted in two semesters. In the first semester, literature review and problem addressing, Data gathering & Analysis and development of prediction system will be performed using neural network method. However model development using system identification toolbox will be conducted in the 2nd semester.

As its evident from the flow chart, first stage was to search about prediction system, its past, current and future situation and how to utilize neural network in modeling this system besides addressing the actual problem that produced this project. Next was data gathering and Analysis. Data was gathered from PETRONAS plant and then was analyzed to enable choosing the most suitable samples to model the system using neural network toolbox in MATLAB software. Those steps were completed satisfactorily within this 1st semester.

Starting from this 2nd semester more training was performed using neural network model to get accurate result and adequate model. After completing this, the module was developed using MATLAB system identification toolbox. Final step was testing the prototype software and observation was recorded and discussed.

3.2 Tools and Equipment

MATLAB software toolboxes:

3.2.1 Neural Network

3.2.2 System Identification toolbox

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Data Analysis

A new set of data of about nine months was analyzed. Starting with January 2010 and ending with September 2010. The data was collected from chemical plant that uses GC analyzer. Number of processes such as normalization, correlation and means removal where performed in order to get the data ready for use. Total number of samples was 144,000 samples divided on 9 months with average of 16000 samples per month. The data is for two analyzers each of which has five inputs and six outputs. Since the project is about predicting the output using the inputs, the main focus will be on the output.

Outputs of each month were plotted using MATLAB and comparison between data was carried out to choose good training data. Besides this, data was correlated to check the relationship between data inputs and outputs using Microsoft Excel.

Based on this analysis, data of March and July months was chosen to represent the training and validation data respectively. Appendix D shows the correlation coefficients of 6 outputs and 5 inputs of March and July months' data.

Table 1: Description of Correlation Coefficient

Correlation Coefficients	Description
$0.65 \leq x < 1.00$	Strong relation
$0.65 \leq x < 1.00$	Moderate relation
$0.00 \leq x < 0.25$	Weak relation

With reference to Table 1 that has the description of correlation coefficients stated earlier, output 1, 2, 4, 5, 6 and input1 are excluded because of their low and weak relationship which will not contribute much to Neural Network training. The chosen data with new correlation is listed in Table 2 below with one output – output3- and 4 inputs –input 2 through 5-. Inputs and Outputs nature is represented in Table.4 below.

Table 2: March month data correlation

	input1	input2	input3	input4	Output3
input1	1				
input2	0.375997	1			
input3	0.403857	0.729158	1		
input4	0.315128	0.679963	0.569141	1	
Output3	-0.41978	-0.70669	-0.89919	-0.70669	1

4353 samples are chosen to represent the training data and the rest (4000) samples are chosen as validation one.

July month with its 6 outputs and 5 inputs correlated samples presented in Appendix C shows the weak relationship between some inputs and outputs. So, outputs 1, 2, 4, 5, 6 are excluded. Table 3 below shows the chosen inputs and output.

Table 3: July month data correlation

	input	Input2	Input3	Input4	Input5	Output3
Input1	1					
Input2	0.19577	1				
Input3	0.31947	0.409395	1			
Input4	0.26294	0.366869	0.546027	1		
Input5	0.26796	0.456805	0.778992	0.717636	1	
Output3	-0.33	0.45172	0.720691	0.715657	0.93105	1

These data of 4640 samples is chosen to be training data while the rest are validation one.

Basically, plant outputs are gases as shown in Table 4.

Table 4: Plant outputs and inputs respectively

AT-4002A -output1	CH_6 (Ethane)	Input1	H2 Flow
AT-4002B-output2	H_2 (Hydrogen)	Input2	N2 Flow
AT-4002D-output3	C_3H_6 (Propylene)	Input3	Reactor Temperature
AT--4002E-output4	C_2H_4 (Ethylene)	Input4	Reactor Pressure
AT-4002H-output5	N_2 (Nitrogen)	Input5	Propylene Flow (Raw)
AT-4002G-output6	C_3H_8 (Propane)		

To check the output performance, the two months data was plotted as shown in Figure 5 and 6 below. The chosen output is AT-4002B which is Hydrogen gas (H_2) and presented in red color.

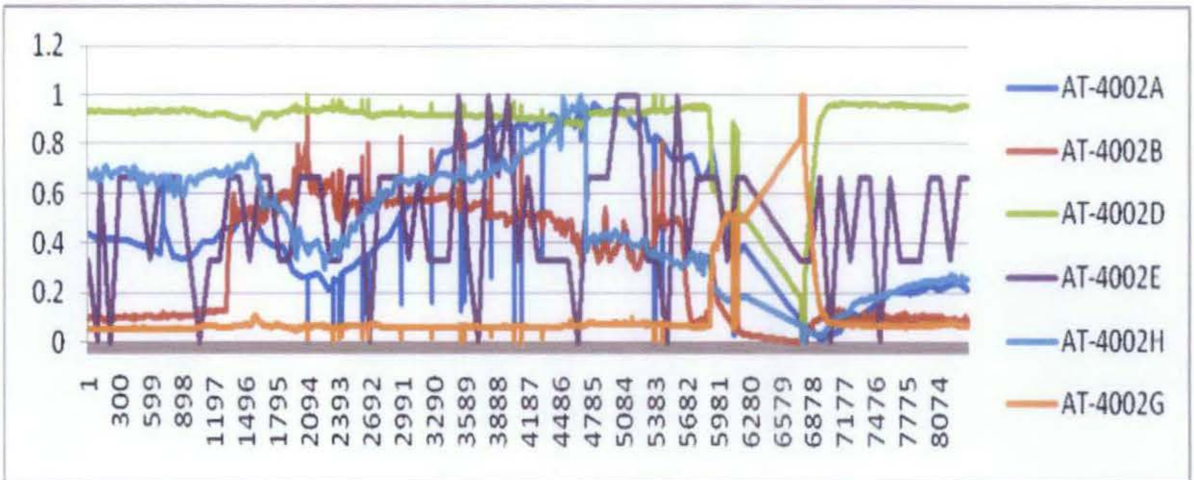

Figure 5: The 6 Outputs of March Month with the Redline Representing the Chosen Output

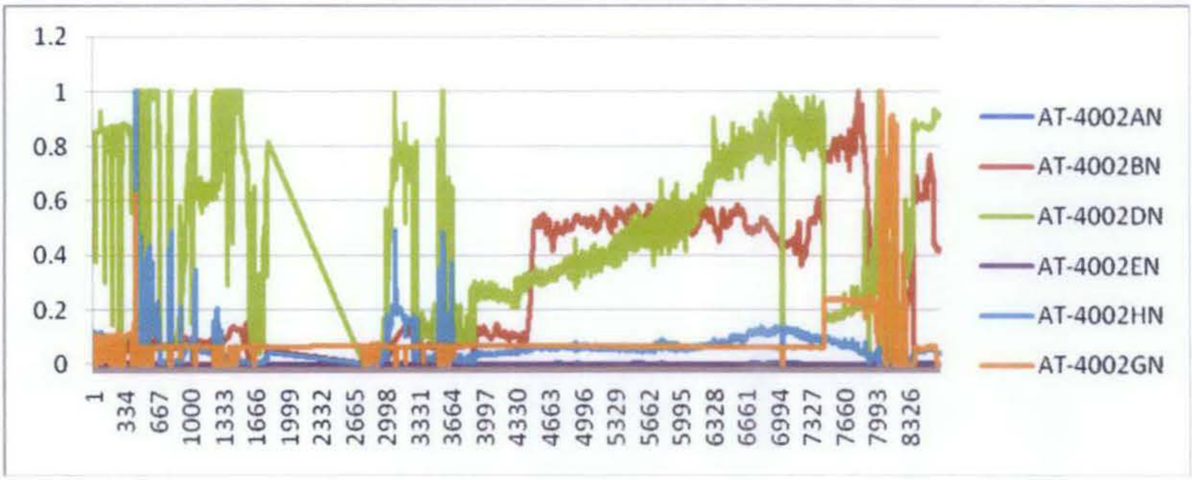

Figure 6: The 6 outputs of July Month With The Redline Representing The Chosen Output

4.2 Applying Neural Network:

After choosing the data, MATLAB Neural Network toolbox was used to train the network. The general Structure of Neural Network model was shown earlier in Figure 2. Two layers network with feedforward network were chosen for the overall network architecture. Based on the data, Multi Layer Architecture is used for there are 4 inputs and 1 output. Data structure is sequential for the samples were taken every 5 minutes. Thus, training is in incremental mode. The network was trained and adjusted sot that particular input leads to a specific target output. Comparison between output and target was carried out till good result was achieved.

After long journey in developing and correcting the program in MATLAB M-file as attached in Appendix B, the transfer function, number of neuron for each layer, number of e the epochs –iterations- and the desired performance goal was the most important parameters that was chosen based on trial and error procedure.

During the Neural Network training, performance graph was plotted as shown in Figure 7 below. This graph shows the error goal and the improvement in the error after every epoch. Sometimes the network may not reach the goal if the number of epochs or neurons is not enough. This can overcome by increasing either numbers or any of them.

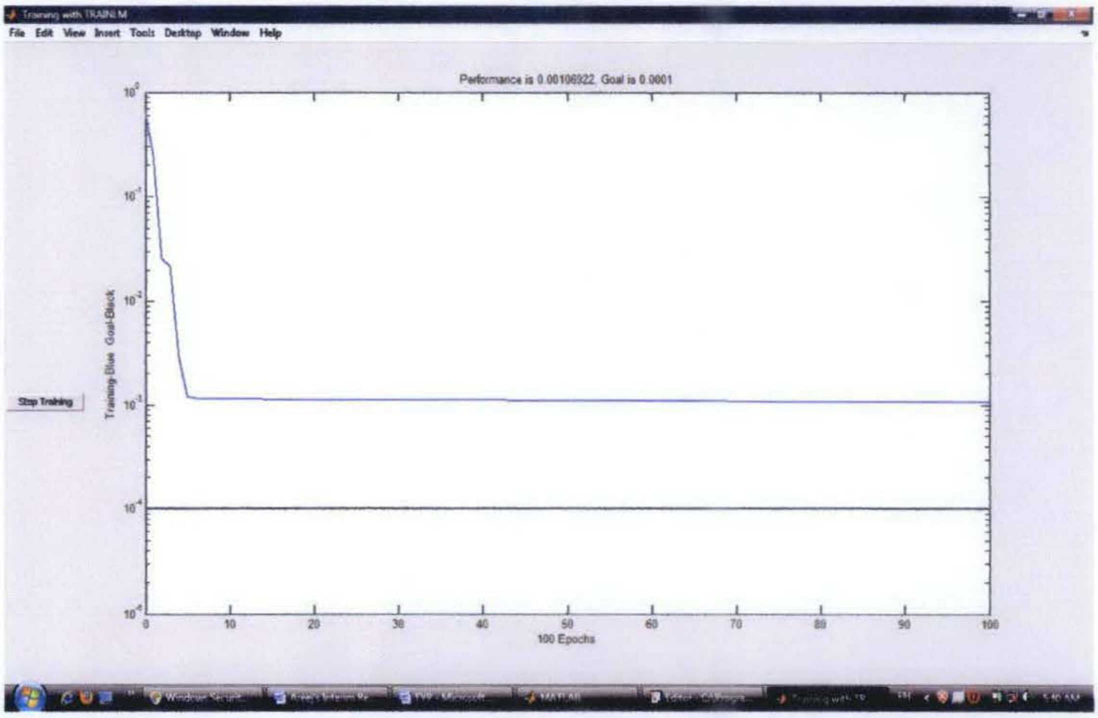

Figure 7: Graph of Training Performance

Once the network with the desired goal is achieved, the output performance is evaluated based on the calculated error. If the error value is acceptable then the model is acceptable as well. Figure 8 and 9 shows the error calculated for the training and validation set of data and the improvement occurred after modifying some parameters such as number of neurons, epochs, set of data and the desired goal.

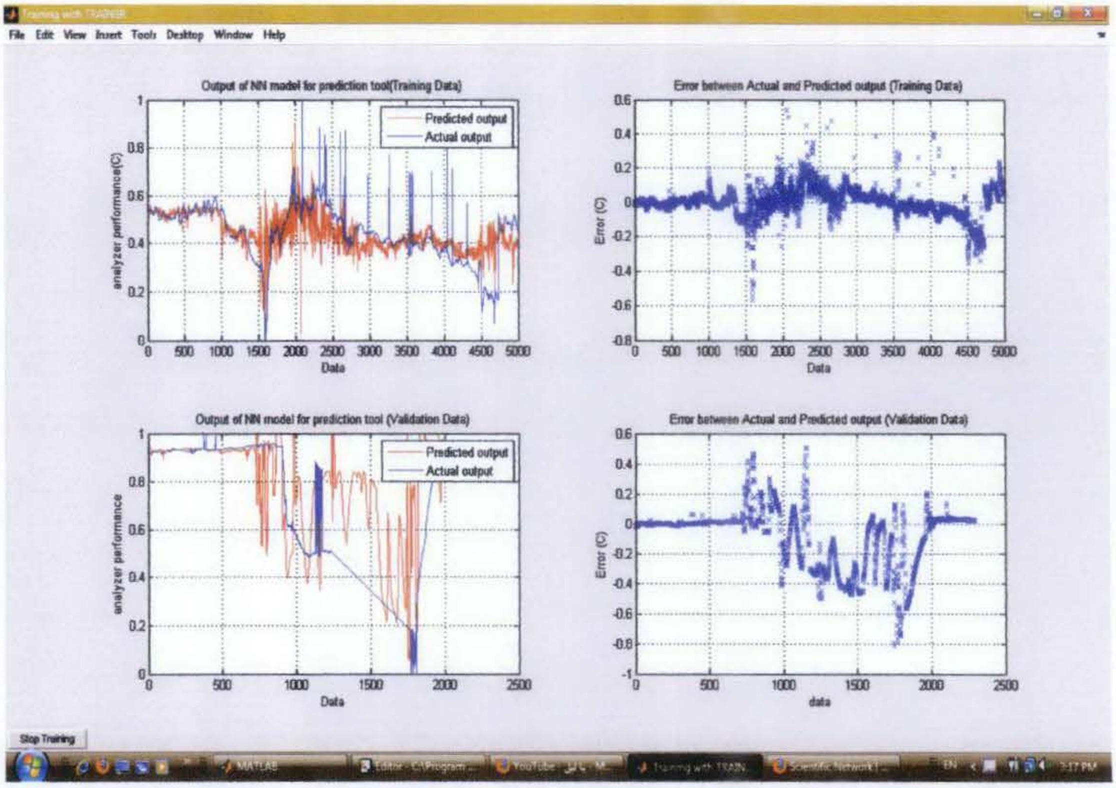

Figure 8: Neural Network Model

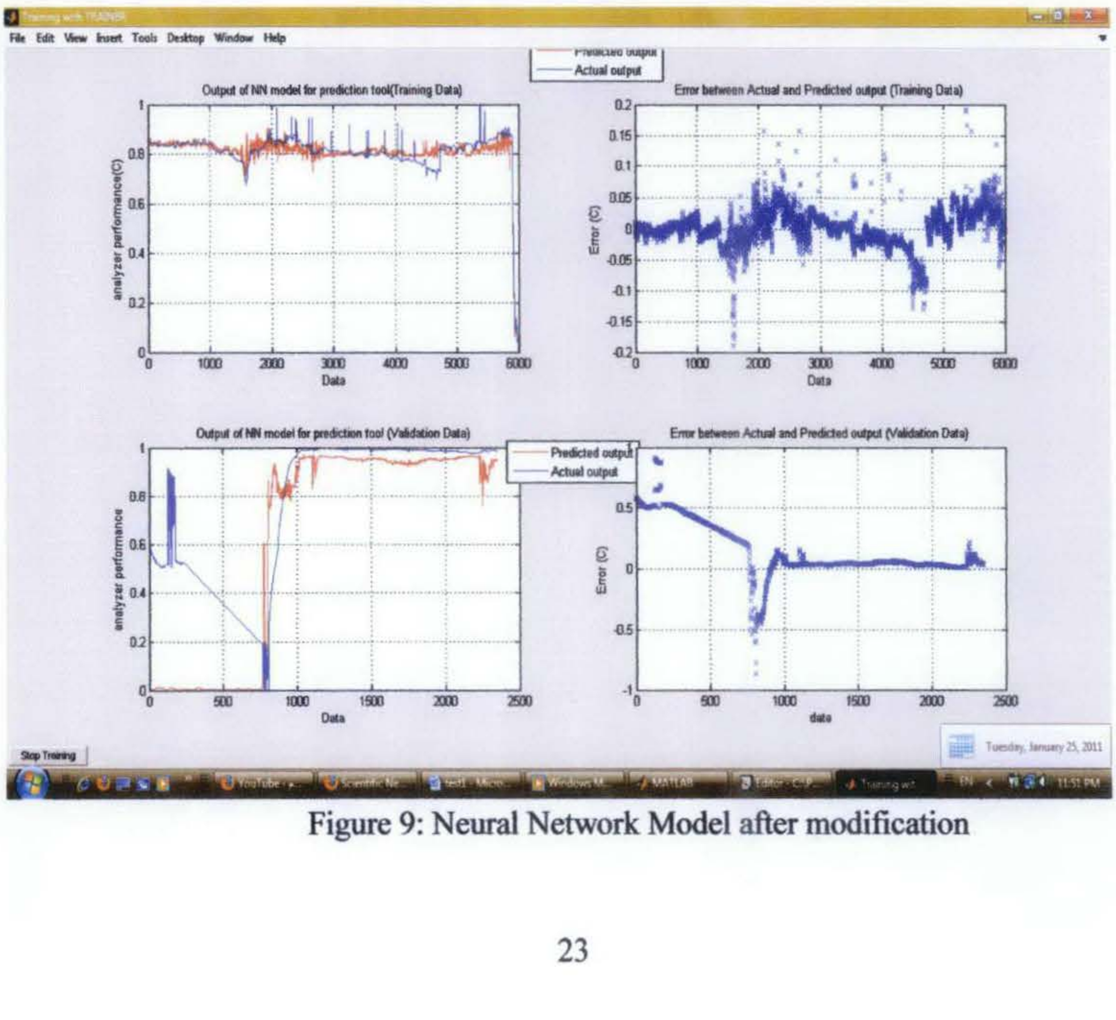

Figure 9: Neural Network Model after modification

The following table summarizes the architecture and performance value for the Neural Network model

Table 5: Neural Network Model Architecture

Parameters	Prediction tool
Types of Network	Newff
Network Layers	2
Number of Neurons	91
Learning rate	0.1
Training function	Trainbr
Numbers of epoch	1000
Transfer function	logsig

The below table shows the error analysis for Neural Network model

Table 6: Error Analysis for Neural Network Model

Data Set	Number of data	RMSE	Index
Training	8000	03.00	0.67
Validation	8000	26.66	0.98

From the plotted graphs of Neural Network performance, the training result is very good while the validation one is not acceptable. However, there are few outliers in the graphs. This could be due to the set of data since the used process variables as input has large deviation that can be related to the plant processes. Also as seen from the output graphs, those chosen set of data has large variation and high error. This would contribute to the result on a way or another. Utilizing July month data for its output and inputs have stronger relationship and using all the 5 inputs rather than four we could get better result as in the below figures. Those data were then divided into training and validation sets. 6000 samples were chosen as training set and the remaining data (2639) were used for validation process. Many trails using ten different types of training functions were performed. Based on the results as shown in Table 7, the most accurate outcome was obtained using Trainlm function.

Table 7 :Training functions RMSE

Training Function	**RMSE**	
	Training	***Validation***
Trainlm	1.46	0.768
Trainb	6.14	13.65
Trainbfg	28.2	32.58
Trainbr	2.31	22.50
Traincgb	3.08	28.12
Traincgp	8.15	27.21
Traingd	18.20	16.07
Traingdx	29.78	25.67
Trainscg	32.68	59.32

Based on this, Neural Network with Lavenberg Marquardt Algortithm was used and after many trails results were demonstrated as following:

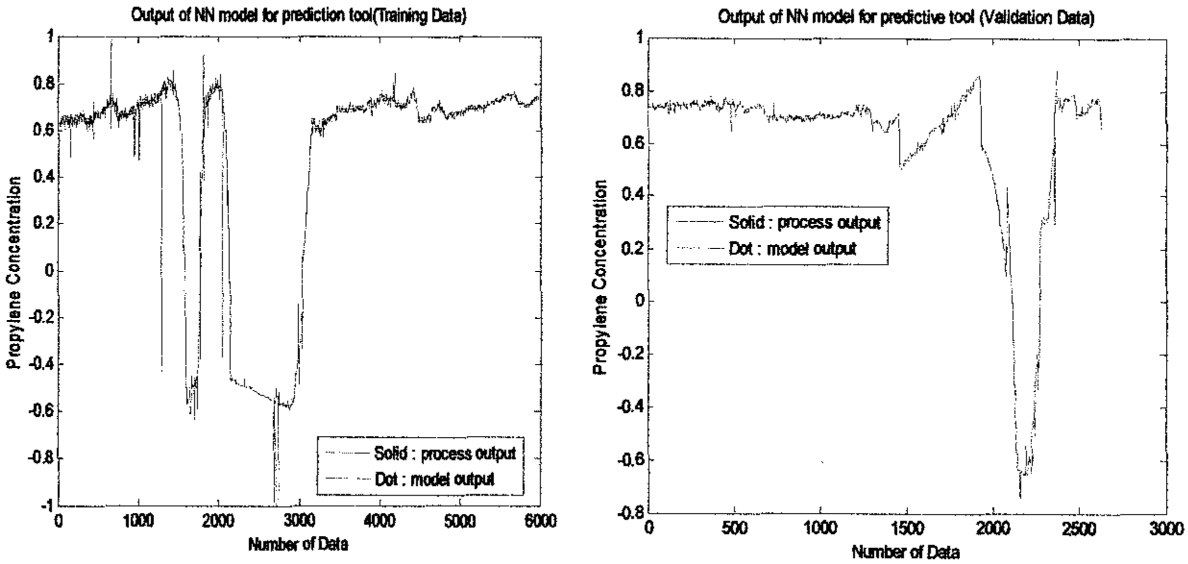

Figure 10: Training and Validation set of July month

Error analysis are shown in table 8 below

Table 8: Error Analysis for Neural Network Model

Data Set	Number of data	RMSE
Training	6000	1.46
Validation	2639	0.768

Since the above model is considered adequate with acceptable range of RMSE error and to ensure good performance of the model, the whole month 8639 samples were chosen as training data while other month of 8700 samples were chosen as validation set. Figure.11 shows the performance of the network

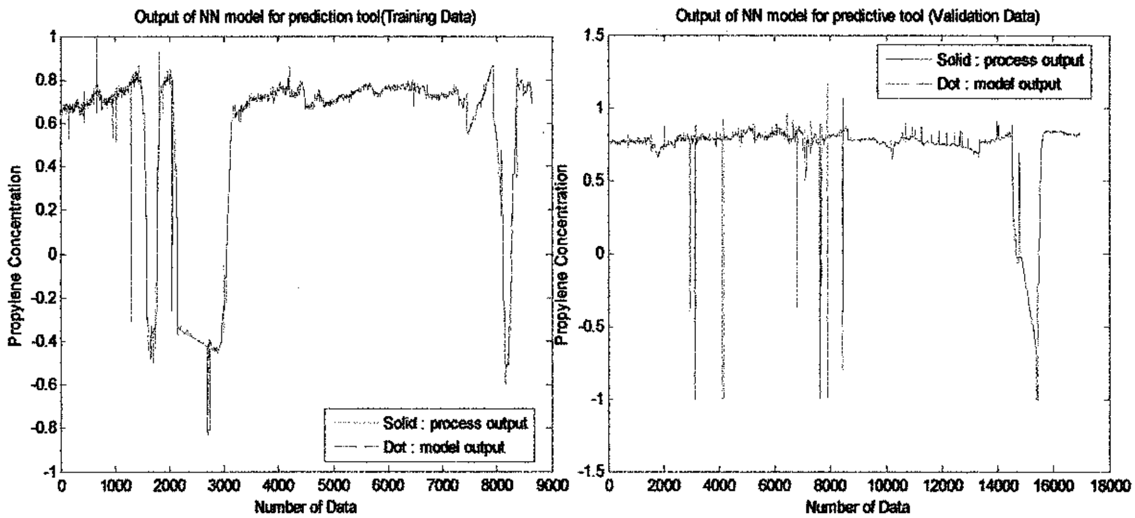

Figure 11: Training and Validation set of July and March months respectively

Error analysis are shown in table9 below

Table 9: Error Analysis for Neural Network Model

Data Set	Number of data	RMSE
Training	8639	1.37
Validation	8700	3.02

Generally, the overall performance of Neural Network is encouraging since the main goal is to achieve prediction tool that is capable of predicting GC analyzer outputs.

4.3 Applying System Identification toolbox:

The best result of neural network model was obtained using July month data, this data were used in system identification toolbox to get the adequate model that can be later applied to get GC analyzer output of different inputs. The data was divided to 6000 samples as estimation and 2639 as validation one. Figure 12 below shows the models performance.

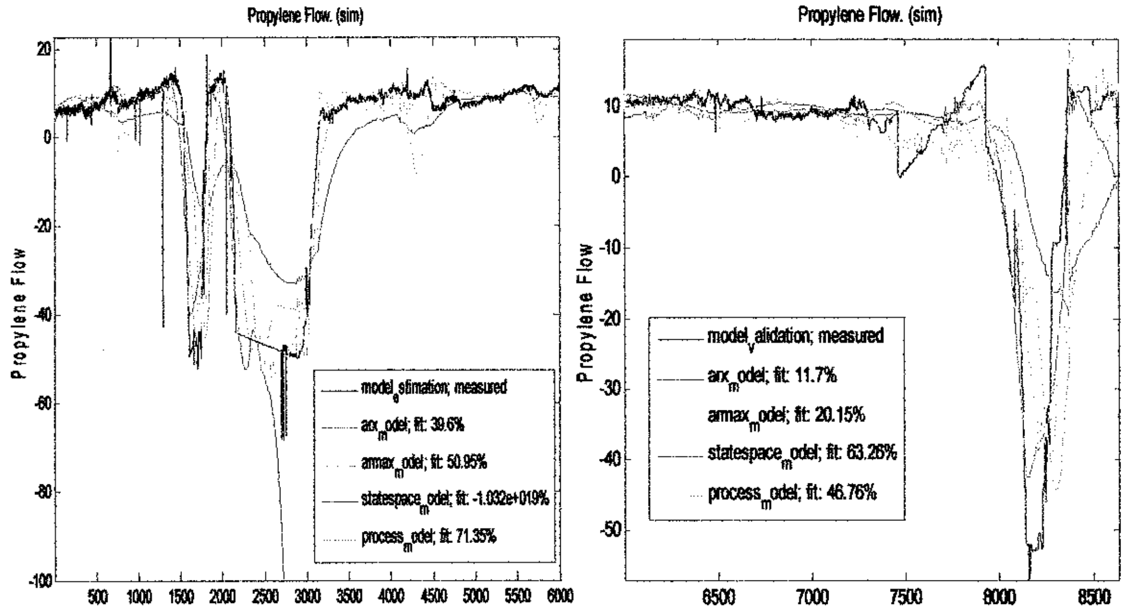

Figure 12: Training and validation performance

From the above results, the best fitting model was State-Space model 2nd order transfer function with 63.26% fitness followed by Process model with 46.76% fitness. The RMSE of the four above models were as following (Table 10):

Table 10 RMSE values for system identification models

Model	RMSE Value
ARX model	19.5404
ARMAX model	19.518
State-Space model	10.46
Process Model	17.525

To check the reliability of the model more data were examined using other months. July month data with 8639 samples were used as estimation set whereas August month data with 8700 samples were used as validation one. Figure 13 below represent those models.

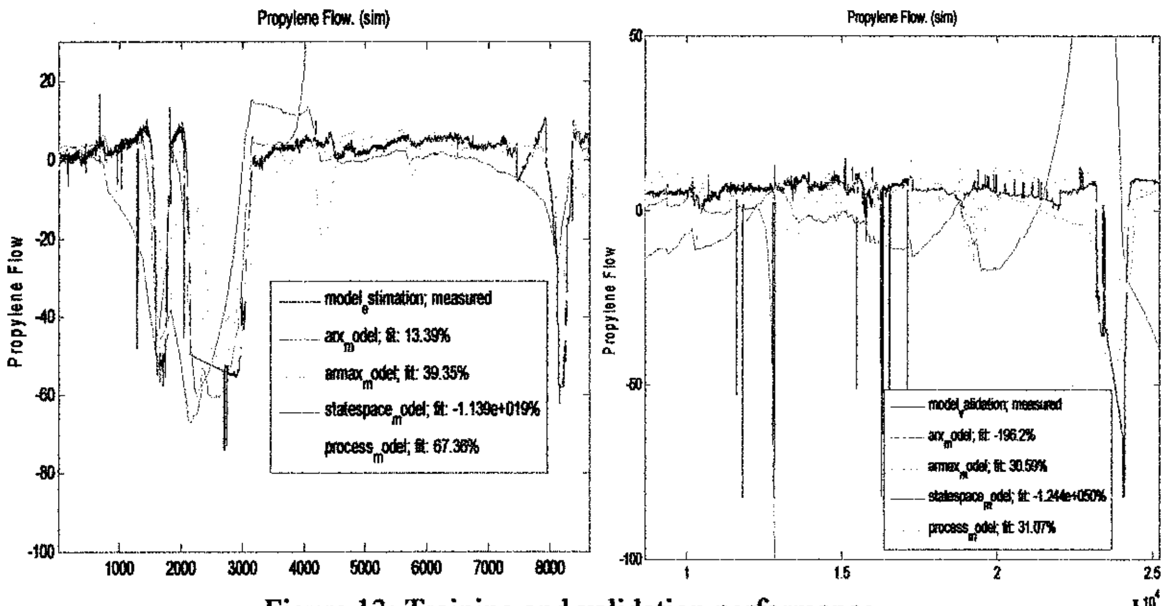

Figure 13: Training and validation performance

Obviously, process model with 3rd order transfer function has the best performance with 31.07% fitness which is higher by 0.08% than the closer model to it –ARMAX model-. The RMSE of the four above models were as following (Table 11):

Table 11 RMSE values for system identification models

Model	RMSE Value
ARX model	84.733
ARMAX model	20.23
State-Space model	Large value
Process Model	19.96

Overall, results demonstrate that neural network model using Levenberg-Marquardt can be rated as the best among other models.

4.4 Developing neural network code:

After obtaining adequate result using neural network, the model was developed to handle six outputs instead of one. In order to develop its performance more iteration were obtained as shown in Table 12. The more the number of iterations the best the model is. data used to generate these outputs were March month and results were as following:

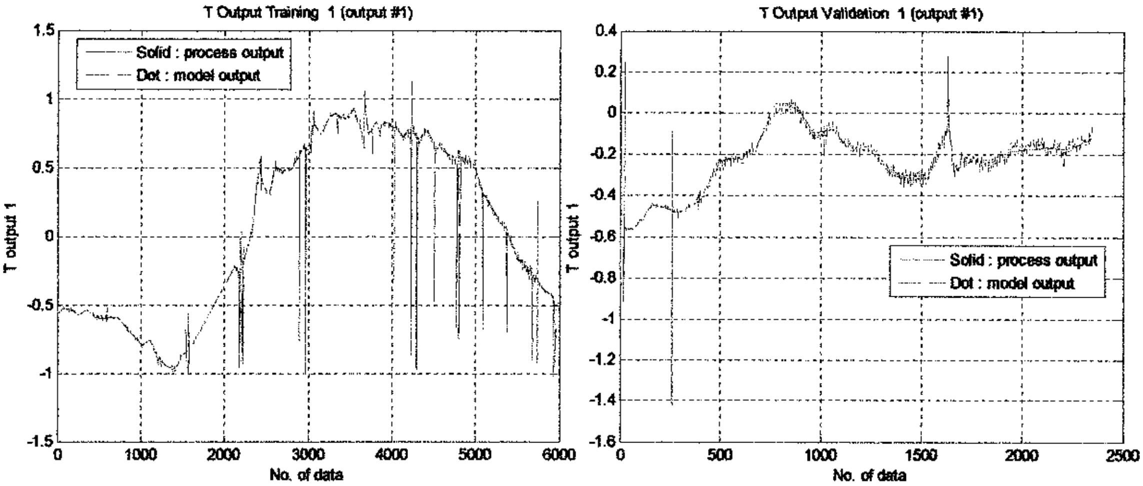

Figure 14: Training and validation performance of output 1

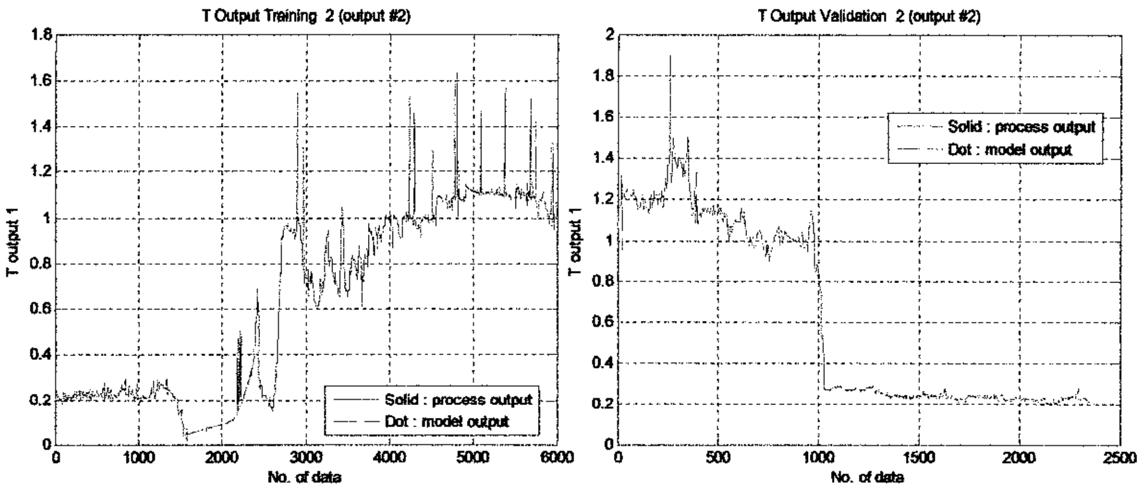

Figure 15: Training and validation performance of output 2

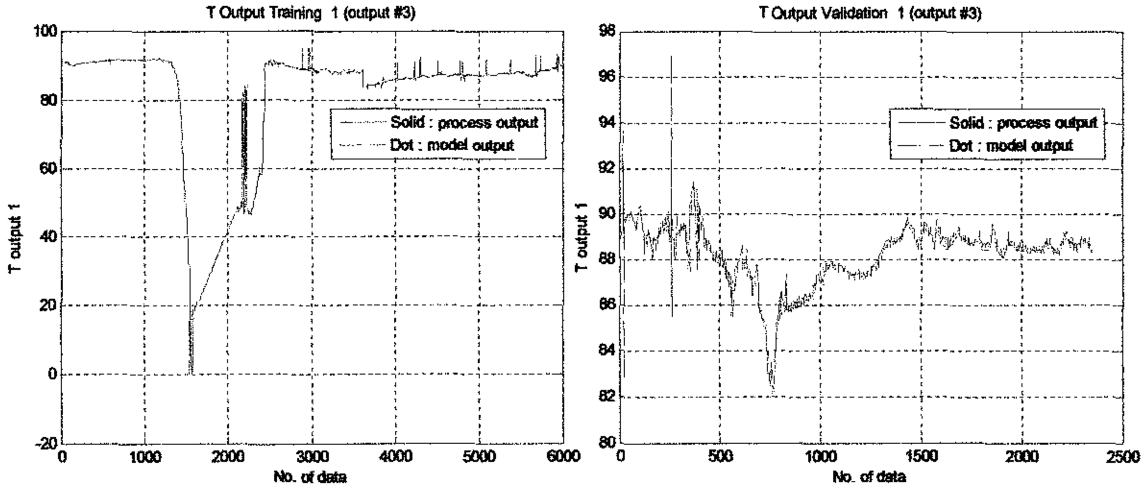

Figure 16: Training and validation performance of output 3

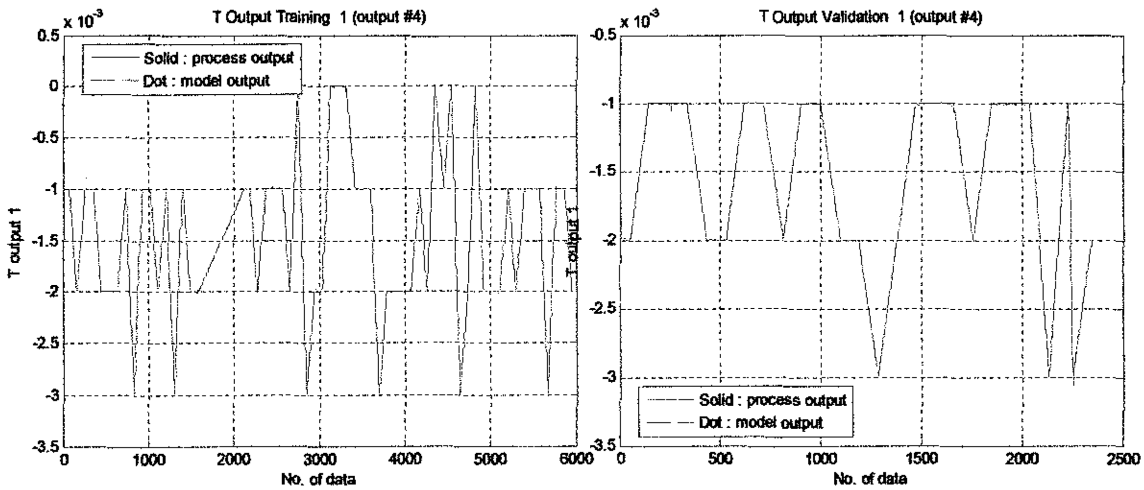

Figure 17: Training and validation performance of output 4

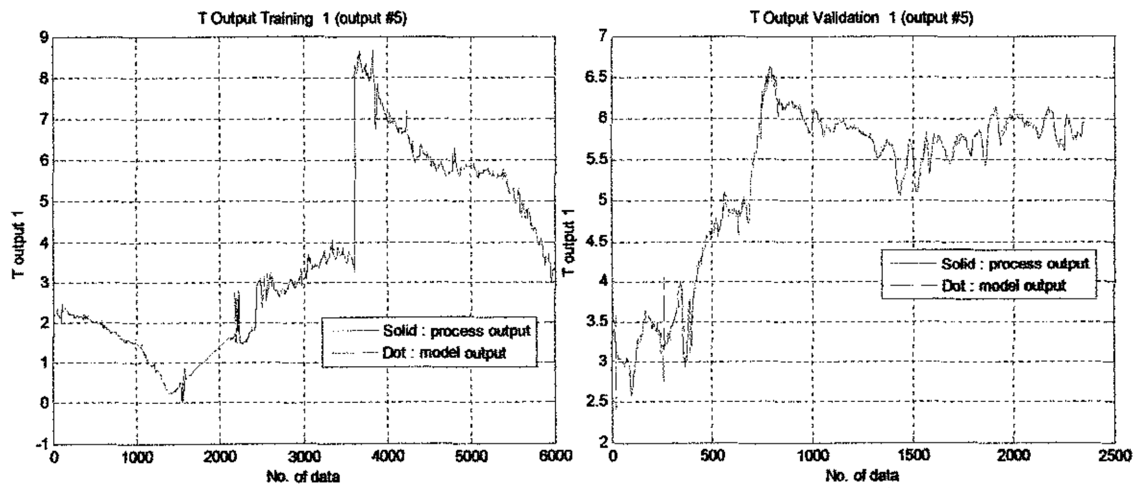

Figure 18: Training and validation performance of output 5

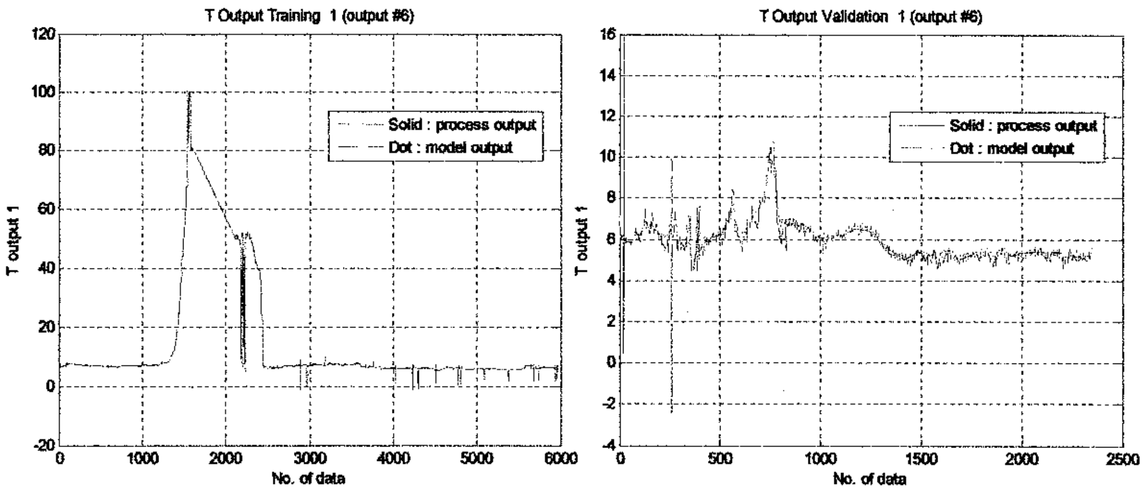

Figure 19: Training and validation performance of output 6

Table 12 Effect of number of iterations on training and validation performance

Number of Iterations		Output1	Output2	Output3	Output4	Output5	Output6
1st	**Training**	6.8607	2.7583	2.2159	0.26567	1.6445	2.1506
	Validation	9.799	8.7082	1.7818	0.98022	1.6776	1.6654
2nd	**Training**	7.3498	2.8515	1.9901	0.27409	1.6335	01.9318
	Validation	8.7618	7.7082	1.3074	0.96991	0.71715	01.3889
3rd	**Training**	6.0112	2.7129	1.964	1.6232	2.0537	2.0005
	Validation	6.5925	3.1246	1.1348	1.96	1.2136	1.1578
4th	**Training**	6.8288	2.7512	2.0022	0.59879	2.0387	1.9992
5th	**Training**	3.1908	1.6854	1.0155	0.16348	1.3482	0.77544
	Validation	2.453	0.13467	1.8297	00.93629	4.7739	2.6985
6th	**Training**	2.9796	01.5545	0.98354	00.1771	01.2221	00.71999
	Validation	2.7934	8.4461	4.4014	1.0988	4.0461	04.9805
7th	**Training**	2.1938	1.5206	1.0639	0.20776	1.2536	0.79791
	Validation	2.0889	2.4971	1.4314	0.64242	1.2209	2.5961

4.5 The Need to Know the Plant Structure:

Basically, there is a high need to know the plant structure to interpret the values of the inputs versus the outputs. By knowing the structure and the process inside the plant one can determine the inputs with the strongest relation to the output which will help reducing the number of inputs. Also this will help knowing best outputs that can be used to detect the six algorithms mentioned earlier and so reducing the outputs variables for better training result. Knowing this structure will also help explaining the behavior of the system and predicting properly and accurately the future values. Moreover, it will enable choosing the appropriate and most suitable functions and representation of the data when using MATLAB Neural Network. Refer to Figure 20.

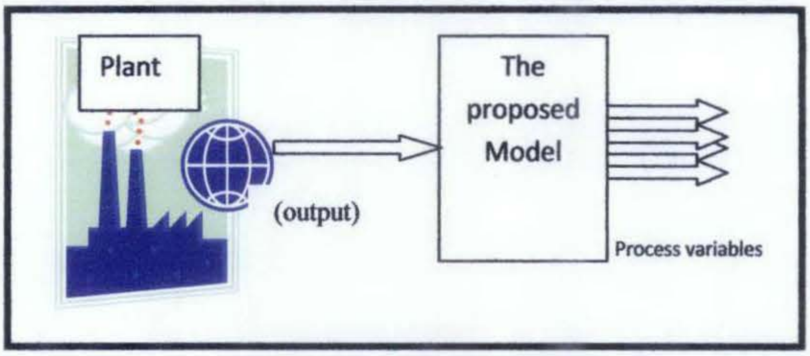

Figure 20: The Planned Project

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

Detailed and practical prototype software that acts as GC measuring system was developed in this final year project. The software designed using neural network and system identification tools and implemented via MATLAB Toolboxes.

Set of data of thermal friction process of nine months duration was analyzed. Based on the analysis, July samples were chosen to train neural network, ARX, ARMAX, State-Space and low-order transfer function (Process Model) models. The possibility of using neural network for predicting the output of Gas Chromatograph analyzer was explored. GC analyzer samples were used to train and validate neural network model. Other models were used to support this claim.

It is concluded from the experimental work that neural network can be used for predicting the output of GC and so can be an alternative measuring system of GC analyzer.

5.1 Recommendations

Since the application of Neural Network in such project is new approach, good improvement and development can be done to add more features to the software. More tests and implementation will be done to this model in order to get more accurate one and so effective predictor.

It is recommended to add detection feature to the prototype software. The impact of developing an accurate detection and prediction tool will certainly save large amount of cost and ensure safety of the personnel and equipment. Thus modeling will be performed again with new set of data if obtained to get rid of the current data weaknesses such as large variation between the process variables. Also, a visit for the plant should be conducted to get closer to the plant process and so interpret more accurately the chosen data and exclude the one with high variations and the inputs with the least relationship to the outputs

Future work will be basically improvement in Neural Network detective model.

REFERENCES

[1] Elizabeth Prichard, co-ordinating author, p2003 *Practical Laboratory Skills Training Guides, Gas Chromatography.*

[2] A. D. Telang, Amit Telang, *COMPREHENSIVE MAINTENANCE MANAGEMENT Polices, strategies and Options,* Eastern Economy Edition, page 106.

[3] Anthony M. smith, Glenn R. Hinchcliffe P.2004, *RCM GATEWAY TO WORLD CLASS MAINTENANCE,* , page 19

[4] Joel Levitt P.2003, *Complete Guide to Preventive and Predictive Maintenance,* First Edition, page 92.

[5] B.J Baugh. p 1993, *Gas Chromatography,* New York, Oxford University Press Inc. page 1

[6] Edited by Robert L.Grob, Ph.D. and Eugene F.Barry, Ph.D. p(2004). *Modern Practice of Gas Chromatography,* Fourth Edition, New Jersey, John Wiley & Sons, Inc. Page 20.

[7] Helmut J.Maier, 1962, *Analytical Instrumentation for Process Monitoring and Control,* IRE Transactions of Industrial Electronics

[8] J.D Andrews and T.R.Moss, *Reliability and Risk Assessment, Longman Sientific & Technical,* page19.

[9] Khairul Azri Bin Jamaluddin, *Design, develope and perform robustness test of analyzer² VER2,* Universiti Teknologi PETRONAS, June 2010, page13.

[10] Khairul Azri Bin Jamaluddin, *Design, develope and perform robustness test of analyzer² VER2,* Universiti Teknologi PETRONAS, June 2010

[11] Mo-Yuen Chow p.1997, *Methodologies of using neural network and fuzzy logic Technologies for Motor incipient fault detection*, page 8.

[12] Mo-Yuen Chow p.1997, *Methodologies of using neural network and fuzzy logic Technologies for Motor incipient fault detection,* page 97.

[13] StatSoft Electronic Statistics Textbook, creators of STATISTICA Data Analysis Software and Services. 15 October 2010 < http://www.statsoft.com/textbook/neural-networks >

[14] S. N. Sivanandam, Sumathi & Deepa, *Introduction to neural network using MATLAB 6.0,* The Mc-Graw Hill companies, page 3.

[15] Fausett, L. P.(1994). *Fundamentals of Neural Networks.* New York: Prentice Hall.

[16] Patterson, D. P.(1996). *Artificial Neural Networks.* Singapore: Prentice Hall.

[17] new Rafik Aziz Aliev and Bijan Fazlollahi p.2004, *Soft Computing and its Applications in Business and Economics,* Springer-Verlag Berlin Heidelberg, page 341.

[18] new Rafik Aziz Aliev and Bijan Fazlollahi p.2004, *Soft Computing and its Applications in Business and Economics,* Springer-Verlag Berlin Heidelberg, page 90.

[19] Lennart Ljung, *System Identification Toolbox™ Getting Started Guide,* 1988–2010, The MathWorks, Inc

APPENDICES

APPENDIX B

Neural network code

```
clear all;

close all;
clc;

load FYPj;  %load matlab file with data

%load data from workspace
x = data(:,1:4)';  %separate input and output, x=input
y = data(:,5)';    %separate input and output  y=output

%divide data into TRAINING and VALIDATION
%get the number of input and number of data
train_data = 8000; %number of TRAINING data
validation_data =8000; %number of VALIDATION data
numofvar = size(x,1); %number of input
numofout = size(y,1); %number of input

for m=1:numofvar
    for n=1:train_data
    x_t(m,n)=x(m,n);
    end
end

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

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

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

%dATA nORMALIZATION
for row=1:numofvar
    x_t1(row,:)=((1/max(x_t(row,:)-min(x_t(row,:)))).*(x_t(row,:)-
(min(x_t(row,:)))));
end
```

```
%normailze the validation data

for row=1:numofvar
    x_v1(row,:)=((1/max(x_v(row,:)-min(x_v(row,:)))).*(x_v(row,:)-
(min(x_v(row,:)))));
 end

%normalize the Training data
y_t1=((1/max(y_t)-min(y_t))*(y_t-min(y_t)));

%normailze the validiation data
y_v1=((1/max(y_v)-min(y_v))*(y_v-min(y_v)));

t=minmax(x);
%set network properties
%number of neurons for layer 1 and layer 2
neuron_1 = 51; %number of neurons for layer 1
neuron_2 = 1; %number of neurons for layer 2

%network and parameters
net=newff(t,[51 1],{'logsig','purelin'},'trainlm');
net.trainParam.show = 50;
net.trainParam.lr = 0.01;
net.trainParam.epochs =3000;
net.trainParam.goal = 0.0001;

%set the weights and biases (make sure all are 0)
net.IW{1,1}; %weights of 1st layer
net.LW{2,1}; %weights of 2nd layer
net.b{1}; %bias of 1st layer
net.b{2}; %bias of 2nd layer
%train the network
 [net,tr]=train(net,x_t1,y_t1);
%simulate the network
ytrain = sim(net,x_t1);  %simulate the network
test1=mse(ytrain-y_t1)
ytrain1= (((max(y_t1)-min(y_t1))*(ytrain))+min(y_t1));
etrain=y_t-ytrain1;
yvalid=sim(net,x_v1);
yvalid1=(((max(y_v1)-min(y_v1))*(yvalid))+min(y_v1));
evalid=y_v-yvalid1;
yvalid1=yvalid';
ytrain1=ytrain';

%plot graph
%plot the actual and predicted H2 (output) training and validation
data
subplot(2,2,1);
plot (yvalid1,'r');
hold on;
plot (y_v,'b');
xlabel('Data');
ylabel('analyzer performance');
```

```
title('Output of NN model for prediction tool (Validation Data)');
legend('Predicted output','Actual output');
grid on;

%plot the different between the actual and predicted H2 (output)
training
%and validation data
plot(evalid,'x');
xlabel('data');
ylabel('Error (C)');
title('Error between Actual and Predicted output (Validation
Data)');
grid on;

%plot the actual and predicted H2 (output) from TRAINING data
subplot(2,2,3);
plot (ytrain1,'r');
hold on;
plot (y_t,'b');
xlabel(' Data');
ylabel('analyzer performance(C)');
title('Output of NN model for prediction tool(Training Data)');
legend('Predicted output','Actual output');
grid on;

%plot the different between the actual and predicted H2 (output)
from TRAINING data
subplot(2,2,4);
plot(etrain,'x');
xlabel('Data');
ylabel('Error (C)');
title('Error between Actual and Predicted output (Training Data)');
grid on;
%error analysis

rmse_valid = sqrt(mse(evalid)) %mean square error
index_valid = (sum((evalid).^2)/sum((y_t1-mean(y_t1)).^2))*100
rmse_train = sqrt(mse(etrain)) %mean square error
```

APPENDIX C

System Identification code

```
clc
clear

inputs = xlsread('JOOREE','Sheet2');
output = xlsread('JOOREE','Sheet2');
U1=inputs(:,1);
U2=inputs(:,2);
U3=inputs(:,3);
U4=inputs(:,4);
U5=inputs(:,5);
Y=output(:,6);

U1=U1-mean(U1);
U2=U2-mean(U2);
U3=U3-mean(U3);
U4=U4-mean(U4);
U5=U5-mean(U5);

Y=Y-mean(Y);

U=[U1 U2 U3 U4 U5];

data_model=iddata(Y,U,1);

set(data_model,'InputName',{'H2flow';'N2Flow';'flow';'Temp';'Pressur
e'},'OutputName',{'Propylene Flow'});

% ---------------------------
%       Estimation Data
% ---------------------------

model_estimation =data_model(1:5000);

% ---------------------------
%       Validation Data
% ---------------------------

model_validation =data_model(5001:7777);

% -----------------------------------
%          Model Orders
% -----------------------------------

arx_order=[3 [3 3 3 3 3] [0 0 0 0 0]];
armax_order= [3 [3 3 3 3 3] 3 [0 0 0 0 0]];
statespace_order=2;
process_model='P2';
```

```
% ------------------------------
%        System Identification
% ------------------------------

arx_model=arx(model_estimation,arx_order,'InitialState','estimate');
armax_model=armax(model_estimation,armax_order,'InitialState','estim
ate');
statespace_model=n4sid(model_estimation,statespace_order,'InitialSta
te','estimate');
process_model=pem(model_estimation,process_model);

 figure(1)

compare(model_estimation,arx_model,armax_model,statespace_model,proc
ess_model)
 figure(2)

compare(model_validation,arx_model,armax_model,statespace_model,proc
ess_model)

arx_model_top.EstimationInfo
armax_model_top.EstimationInfo
statespace_model_top.EstimationInfo
process_model_top.EstimationInfo
```

APPENDIX D

Correlation Coefficients Values for March and July Months

Table 1: March month data correlation

	output1	output2	output3	output5	output6	input2	input3	input4	input5
output1	1								
output2	0.488288	1							
output3	0.219536	0.372954	1						
output5	0.59096	0.512242	0.422504	1					
output6	-0.28743	-0.43362	-0.99233	-0.52552	1				
input2	0.367461	0.663621	0.380033	0.415621	-0.41978	1			
input3	0.279813	0.202941	0.704174	0.42977	-0.71275	0.375997	1		
input4	0.208539	0.370135	0.895244	0.47059	-0.89919	0.403857	0.729158	1	
input5	0.241948	0.235157	0.697601	0.427578	-0.70669	0.315128	0.679963	0.569141	1

Table 2: July month data correlation

	output1	output2	output3	output4	output5	output6	input1	input2	input3	input4	inp
out1	1										
out2	0.076971	1									
out3	0.054966	0.153579	1								
out4	0.99548	0.075977	0.045806	1							
out5	0.414913	0.231144	0.562017	0.416623	1						
out6	-0.00676	0.149495	-0.24845	-0.00806	-0.13256	1					
ut1	-0.0089	-0.15912	-0.05236	-0.00693	-0.05241	0.063184	1				
ut2	-0.01395	0.684656	0.192485	-0.01632	0.172751	0.070225	- 0.19577	1			
ut3	0.034664	0.354239	0.249761	0.033947	0.330857	-0.10001	- 0.31947	0.409395	1		
ut4	0.018688	0.391015	0.315843	0.014953	0.311961	0.017554	- 0.26294	0.366869	0.546027	1	
ut5	0.023706	0.438626	0.389603	0.019088	0.322789	0.00647	- 0.26796	0.456805	0.778992	0.717636	