Modelling and Control of Chemical Processes using Local Linear Model Networks

Ahmed Saad A Abdelhadi

A thesis submitted in partial fulfilment of the requirements of the Liverpool John Moores University for the Degree of Doctor of Philosophy

December 2019

ABSTRACT

Recently, technology and research in control systems have made fast progress in numerous fields, such as chemical process engineering. The modelling and control may face some challenges as the procedures applied to chemical reactors and processes are nonlinear. Therefore, the aim of this research is to overcome these challenges by applying a local linear model networks technique to identify and control temperature, *pH*, and dissolved oxygen. The reactor studied exhibits a nonlinear function, which contains heating power, flow rate of base, and the flow rate of air as the input parameters and temperature, *pH*, and dissolved oxygen (pO_2) the output parameters.

The local linear model networks technique is proposed and applied to identify and control the *pH* process. This method was selected following a comparison of radial basis function neural networks (RBFNN) and adaptive neuro-fuzzy inference system (ANFIS). The results revealed that local linear model networks yielded less mean square errors than RBFNN and ANFIS. Then proportional-integral (PI) and local linear model controllers are implemented using the direct design method for the *pH* process. The controllers were designed on the first order *pH* model with 4 local models and the scaling factor is 20.

Moreover, local linear model networks are also used to identify and control the level of dissolved oxygen. To select the best method for system identification, a gradient descent learning algorithm is also used to update the width scaling factor in the network, with findings compared to the manual approach for local linear model networks. However, the results demonstrated that manually updating the scaling factor yielded less mean square error than gradient descent. Consequently, PI and local linear model controllers are designed using the direct design method to control and maintain the dissolved oxygen level. The controllers were designed on first and second order *pO2* model with 3 local models and the scaling factor is 20.

The results for the first order revealed good control performance. However, the results for second order model lead to ringing poles which caused an unstable output with an oscillation in the input. This problem was solved by zero cancellation in the controller design and these results show good control performance.

Finally, the temperature process was identified using local linear model networks and PI and local linear model controllers were designed using the direct design method. From the results, it can be observed that the first order model gives acceptable output responses compared to the higher order model. The control action for the output was behaving much better on the first order model when the number of local models M=4, compared with M=3 and M=5. Furthermore, the results revealed that the mean square error became less when the number of local models M=4 in the controller, compared with having number of local models M=3 and M=5.

ACKNOWLEDGEMENT

First of all, I would to thank my supervisor Dr. Barry Gomm for his support and helping me in my PhD research. He was always available to listen to me and answer my questions and problems which faced me in my research. And I would like to thank him for his advice and time that he gave me to explain and solve problems. And for his patience and his guidance which helped me in all the time of research and writing of this thesis. I would also like to thank my second supervisor Prof. DingLi Yu for his support throughout this research.

Secondly, I would like to thank my lovely wife and lovely son for their love and encouragement during my study. And for supporting me and helping me in all my life and also I would like to thank my parents for their love and support in all my life and also I would like to thank all my brothers and sisters for their love and encouragement.

Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	**Descriptions**
ANFIS	Adaptive neuro-fuzzy inference system
ANN	Artificial neural networks
CSTR	Continuous stirred tank reactor
CH_3COOH	Acetic acid
LLMN	Local linear model networks
MSE	Mean square error
NaOH	Sodium hydroxide
NH_4OH	Ammonium hydroxide
PI	Proportional-integral controller
PID	Proportional-integral-derivative controller
pO_2	Dissolved oxygen
RBFNN	Radial basis function neural network
RLS	Recursive least squares
T	Temperature

LIST OF SYMBOLS

Symbol	Description
$\mathrm{C_i}$	RBF neural network centre
σ_i	RBF neural network width
ϕ_i	Gaussian function
$\mathrm{w_i}$	Network weights
$y_s(k)$	Scaled output
$u_s(k)$	Scaled input
α	Scaling factor
$\mathrm{b_i}$	Bias
η	Learning rate
H^+	Hydrogen ion concentration
F_1	Flow rate of weak acid
C_1	Concentration of acid
F_2	Flow rate of strong base
C_2	Concentration of base
Ka	Acid equilibrium constant
Kw	Water equilibrium constant
e(k)	Prediction error
λ	Forgetting factor
P(k)	Covariance matrix
L(k)	Gain vector

Chapter 1| Introduction

1.1 Background

The technology used in most industrial systems, such as chemical processes, has been subject to rapid change. This is due to the fact that these processes exhibit nonlinearity and face a number of challenges because of their complexities. The motivation for this project arises from the control challenges faced by nonlinear industrial systems: for example, chemical and biochemical industries. This is because the non-linear and uncertain nature of many processes means that improved control cannot be applied simply. Typically, this applies to multivariable industrial systems. Because nonlinearity is a problem of process control, the challenges facing industrial systems, such as chemical reactors, require an enhanced and robust performance controller. Thus, this project aims to investigate the application of local linear model networks (LLMNs) in this area. The ability of local linear model networks to represent nonlinear systems potentially renders them a powerful tool for modelling and controlling the process. This process is a continuous stirred tank reactor, comprising three inputs, heating power (Q), flow rate of base $(f_b),$ and flow rate of air (f_a) and three outputs, temperature (T), dissolved oxygen (pO_2) and $pH.$ The main component of this research will study pH control, which is crucial for industries presently across a range of systems, such as: chemical and biological reactions, boiler water treatment, municipal waste digestion, cooling tower water treatment and acid pickling (Williams *et al.*, 1990). Furthermore, it is a critical element of fermentation etching and coagulation/precipitation. Of many applications, wastewater pH control is typically the most difficult due to the process sensitivity at a $pH = 7$ target and due to the unknown and nonstationary fluid composition. Given the significant nonlinearity behaviour, the control of pH is important in many process industries to maintain the steady-state value. In 1972, McAvoy presented the dynamics and control for a pH in stirred tank reactor by neutralising sodium hydroxide with acetic acid (McAvoy *et al.*, 1972).

The concentration of dissolved oxygen (pO_2) in a continuous tank reactor, such as in an activated sludge system, has emerged as a significant issue for process control parameters (Du *et al.*, 2018). Dissolved oxygen is a vital parameter in chemical reactors and can be easily affected by other variables that yield rapid change in response to the entire system: thereby resulting in a poor controller. Therefore, the second objective of this research is to identify and control the dissolved oxygen process using local linear model networks. The process contains air flow as input and is affected by temperature and dissolved oxygen as output.

Moreover, maintaining the steady-state temperature value is another crucial parameter of the chemical process. For instance, the temperature of water in food products is significant in determining the product quality, economic value, safety and operation of chemical reactors (Luyben., 2007). Due to some challenges controlling the temperature in industrial systems (for example, in heating processes with poor controller performance) materials could be disqualified on the basis of their physical properties when they do not perform successfully (Mugisha *et al.*, 2015). Therefore, the third objective of this study is to identify and control the temperature process using local linear model networks. This will enhance the temperature controller performance in the chemical process, the process comprises heating power as input and temperature as output.

1.2 Problem Statement

Modelling and controlling a chemical process, such as pH, dissolved oxygen and temperature processes, have presented issues over the last decade. Nonetheless, numerous studies have been conducted to improve performance, however, controlling and modelling are still an issue. The significant nonlinearities of chemical processes and complexity of the systems are the major obstacles affecting controller performance. The motivation for this research arises from these challenges to the chemical process such as nonlinearity which lead to poor control performance. Therefore, this research focuses on how to provide a good control for the chemical process.

1.3 Research Novelty

This research is to develop and implement high performance controllers to control and maintain pH, dissolved oxygen and temperature processes in specific ranges. The developed controllers are designed by combining several local models based on a direct design method. The proposed control strategy in this research has the ability to solve instability of the output and the oscillation for nonlinear chemical processes. The main novelties in this research are:

- Most of researchers have studied system identification for chemical processes using simulated data. However, in this research real data have been used for system identification of three different chemical processes and the simulated models have been used in the controller part.
- Modification in the basis function was achieved in the controller design as explained in Section 3.7.4.
- The proposed control method has been proven to be able to deal with instability of the output and the oscillation for nonlinear dissolved oxygen process as explained in Sections 4.4.5 and 4.4.6.

1.4 Aim and Objectives of the Research

The aim of this research is to develop and investigate local linear model network structures for modelling and control strategies for nonlinear chemical processes. The developed models and control will be evaluated using simulation and analysis software called MATLAB/SIMULINK. The objectives of this research are:

1. Develop the methods for designing local linear model networks for a benchmark nonlinear pH process.

2. Investigate the methods using real data and compare them with other approaches, Artificial Neural Network (ANN) and Adaptive Neuro-Fuzzy Inference System (ANFIS) models.

3. Develop and investigate control based on local linear model networks for the pH process.

4. Develop and investigate system identification and control based on local linear model networks for a dissolved oxygen process.

5. Develop and investigate system identification control based on local linear model networks for temperature.

1.5 Scope and Organisation of the Research.

In accordance with the above objectives, the thesis is structured as follows:

Chapter 1 provides a brief introduction to the need for control in chemical processes. This chapter also outlines the research aim and objectives.

Chapter 2 presents the literature review pertaining to the methodology adopted for this study: namely, local linear model networks. The review also includes nonlinear control of chemical processes, *pH* control and review of artificial neural networks for chemical processes.

Chapter 3 presents the implementation of the Simulink model for the *pH* process with PI controller based on internal model control (IMC). It is used to familiarise with the process dynamics and nonlinearity. Moreover, it examines the deficiencies of conventional control applied to this highly nonlinear process. This chapter includes the development of local linear model techniques with application to identification of a *pH* process. Data has been investigated, validated and compared with the radial basis function neural network (RBFNN) and Adaptive Neuro-Fuzzy Inference System (ANFIS) models. Finally, this chapter discusses the design of PI and local linear model controllers that are applied to control *pH* data, and presents the results.

Chapter 4 illustrates system identification of a dissolved oxygen real data using the local linear model networks technique. Another algorithm called gradient descent was used to optimise the network width. The results reveal that local linear model networks continue to elicit enhanced performance by giving less mean square error. Then PI and local linear model controllers are designed on first and second order models and the results are discussed.

Chapter 5 describes system identification of temperature real data using the local linear model networks technique. Then PI and local linear model controllers are designed on first order model and the results are discussed.

In this research MATLAB/SIMULINK software has been used to investigate and achieve the objectives.

Chapter 6 presents the conclusion of this thesis and makes recommendations for further research.

Chapter 2 | Literature Review

This chapter reviews recent approaches to nonlinear control of chemical processes. It concludes by reviewing the main proposed methodology in this research: namely, local linear model networks.

Commonly, the investigation of automatic control system dynamics begins by explaining the calculation procedure for each part of the closed-loop system. The account includes linear or nonlinear differential equations that can be joint with the external quantities acting on the system to design the mathematical model of the system's dynamic behaviour (Vukic *et al.*, 2003).

There are two important problems facing the theory of nonlinear control systems:

1. Analysis problem: comprises theoretical and experimental research to discover the property or appropriate mathematical model of the system.
2. Synthesis problem: determines the construction, parameters and control system elements to achieve the desired performance of a nonlinear control system. Furthermore, a mathematical model should be established alongside the technical realisation of the control model. Since the controlled object typically is known, the synthesis consists of defining a controller in a broader sense.

2.1 Control of Chemical Processes

A control approach has been developed using a minimum variation controller and a linearised neural network model for an operating region of *pH* neutralisation process (Chen and Huang, 2004). The instantaneous linearisation procedure is a possible method to design a gain-scheduling type of the control system at each time interval. It has been demonstrated that it is possible to apply the linear controller scheme to the nonlinear control design with not as much calculation as with the nonlinear control counterpart based on the nonlinear neural network model. However, this is suitable only in a narrow region when the process is operating around a significantly nonlinear point.

Proportional Integral (PI) and Proportional Integral Derivative (PID) controllers have been used in control engineering for decades. However, the PID controllers has not been given much attention until K.S Astrom, T. Hagglund and other researchers have focused on this controller (O'Dwyer, 2006). The most common control algorithm, most feedback loops are controlled by this algorithm. It is implemented in many different forms (Astrom and Hagglund, 1995). Figure 2.1 illustrates PID controller block diagram (Janert, 2013).

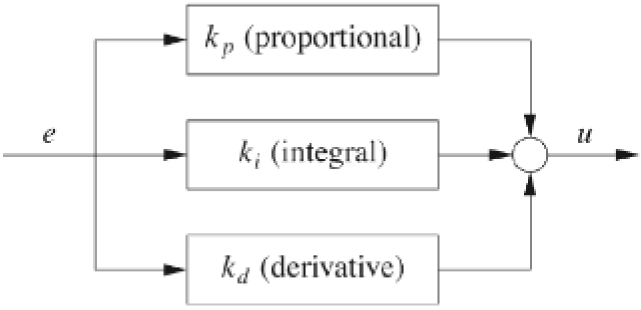


Figure 2-1 PID controller block diagram (Janert, P.K., 2013)

PID controller has been applied to pH neutralisation process. The authors used particle swarm optimization method to tune the PID parameters, the results shown that this method gave improved performance compared with Ziegler Nichols tuning technique. (Bingi *et al.*, 2016).

Investigations have been conducted into numerous non-linear controllers in the *pH* neutralisation process. This process uses a fuzzy non-uniform grid scheduling approach and the method was applied to a weak acid-strong base neutralisation process (Regunath and Kadirkamanathan, 2001). The researchers presented that, although a fuzzy controller is non-linear in general, a PI-type fuzzy controller that uses only error (e) and change in error (Δe) was unable to detect process non-linearities. Consequently, the controller action cannot be made based on knowledge of the process nonlinearity related to different operation regions. By using a fuzzy non-uniform scheduling method, it can be found that integrated absolute error was reduced compared with the standard PI scheduling.

A nonlinear model predictive control (NMPC) has been used widely by many researchers in chemical processes, such as multi-input multi-output reactors. Three variables were investigated: *pH*, temperature and dissolved oxygen with nonlinear dynamics (Yu and Yu, 2005; Yu and Gomm, 2003; Yu *et al*., 1999). By using an adaptive optimisation method (AOM), a nonlinear discrete-time model (NDTM) was derived for each output and the model parameters were estimated from the real data. The developed model was used in a nonlinear MPC scheme and multistep-ahead prediction was achieved for MPC. The authors used the neural network model as a simulation model to select a suitable control parameter alongside the control optimisation for multi-step ahead prediction.

According to Murray-Smith, R. (1994) some applications have been used extensively in nonlinear systems: for example, artificial neural networks and fuzzy systems. However, neural networks cannot benefit straight from a priori knowledge. Furthermore, fuzzy systems sometimes require the capability to improve the structure of membership functions and rules in a data driven manner.

According to Patil and Salunkhe (2008) examined the use of the adaptive neuro fuzzy inference system (ANFIS) to tune the controller for temperature in water and then compared it to a Ziegler-Nichols-tuned PID controller and the results reveal that the performance of the (ANFIS) tuned controller is enhanced compared with the Ziegler-Nichols PID controller. Another study used adaptive neuro-fuzzy inference system (ANFIS) (Gaya *et al*., 2013). The researchers used ANFIS inverse controller to control and maintain dissolved oxygen in an activated sludge process and the results revealed that the ANFIS inverse controller gave a good performance.

Extended Kalman Filter (EKF) has been investigated to estimate the states and feed parameters in the *pH* neutralisation process. The process model has been developed using reaction invariants: the concentrations of reaction invariants of the effluent stream (states) and the feed concentrations (parameters) were estimated online. Experiments and simulations were compared and presented that the states and parameters could be identified well using the extended kalman filter. Moreover, the estimated information could be exploited by the application of state-feedback control methods (Yoo *et al.,* 2004).

Several researchers have focused on temperature control difficulties and attempted to improve the control of temperature by adopting various methods. According to Meng *et al.* (2014), the adaptive PID control based on RBF neural networks to control temperature has been developed. In 2015, Mugisha *et al.* (2015) proposed the temperature controller for industrial heat-treating furnace by using intelligent fuzzy logic and PID controllers. Their results illustrated that the response overshoot for fuzzy logic was smaller than that with the tuned Ziegler-Nichols controller.

Nonlinear model predictive control strategies using neural networks model were designed in the activated-sludge process of wastewater treatment plants and the results compared with a classic PI controllers structure (Goldar *et al.,* 2016).

Liang and Wang (2014) stated that the steam temperature system with large inertia, large time delay and time varying cannot achieve acceptable control performance using a PID controller. They proposed a technique that, uses an internal model cascade control system to control the primary steam temperature system.

Chemical processes such as *pH*, dissolved oxygen and temperature are important factors in fermentation, including food and drink processes. The research area of fermentation processes has been established to investigate techniques for food storage and reduce the risk of pathogenic microorganisms growing in food products. The products produced from the destruction of carbohydrates from bacteria not only contribute to flavour, smell and texture. Moreover, help to identify a suitable product properties. Therefore, the ability to regulator the specific microorganisms that control the microflora of foods is crucial. The quality of food can be increased by fermentation such as in the fermentation of milk to cheese (Steinkraus, 1998).

2.2 Review of pH Control

Some difficulties have emerged in the *pH* process control. For example, the nonlinearity and the process specification make it difficult to develop an accurate mathematical model for *pH* control. Therefore, the *pH* process is one of the most difficult factors to control in chemical process. And good controller should be considered to maintain the *pH* value within a required range. The authors applied a neural network PID control to the pH process and the results shown an acceptable control performance (Yang and Wu, 2016).

System identification and control for *pH* neutralisation process of a weak acid - a strong base has been achieved using neural network model predictive control technique (Tharakan *et al*., 2013). The model predictive control approach has been used with multiple models to investigate the *pH* neutralisation process behaviour in conjunction with an integral action controller. This is a straightforward approach to addressing the changes in non-linearities (Hermansson and Syafiie, 2014). Gomm *et al*. (1996) presented the development of a neural network model of a bench scale in-line *pH* process and the subsequent incorporation of the model in a predictive control strategy. The linear model predictive control algorithm has been applied to control the *pH* neutralisation process. The controller was achieved from one set point to another after the operating region had been split into sub points (Gu and Gupta, 2008).

In 1982, the internal model control (IMC) method was introduced by Garcia and Morari, The researchers found that the IMC structure provides a suitable structure of the closed-loop stability issue altogether and provides the opportunity to address the central issues of control system performance (Garcia and Morari, 1982). This has been extended to include multi input-multi output (MIMO) discrete time systems (Garcia and Morari, 1985). In the study conducted by Darab *et al*. (2012), an internal model control application was implemented for the *pH* neutralisation process. The research results reveal the ability of this method to test for reference tracking and disturbances rejection.

In the process industries, the application of local linear model networks was used for non-linear system identification of the *pH* process and compared with the radial basis function neural network (Abdelhadi *et al.*, 2014). The results demonstrate that local linear model networks technique is preferable as it yields less mean square error.

The gradient descent algorithm for updating the weights of the neural networks was compared with the genetic algorithm (GA). This was achieved for cancer and diabetes dataset and their results revealed that the error of GA for cancer is better compared to gradient descent. However, for diabetes the gradient descent performed better (Ahmad *et al.*, 2010). Based on internal model control for nonlinear *pH* control, GA was proposed by Mwembeshi *et al.*, (2004). First and second order internal model control transfer function models were implemented and the results showed better performance compared with a conventional Ziegler Nichols-tuned PI controller.

2.3 Review of Artificial Neural Networks and Model Identification for the Chemical Processes

The earliest work of artificial neural networks can be traced back to the 1940's when McCulloch and Pitts introduced a computer model called neural networks (McCulloch and Pitts, 1943). Since then the neural networks have been used extensively as powerful computational tools in the modelling and control of nonlinear dynamic systems (Narendra and Parthasarathy,1990; Bhat and McAvoy,1990), and in the chemical process area (Himmelblau, 2008; Himmelblau, 2000; Gomm *et al.*, 2000).

An artificial neural network is a computational model, it is used to solve nonlinear functions. The artificial neural networks has class so called radial basis function networks, it have drawn significant attention from researchers. Figure 2.2 illustrates the structure of radial basis function, which comprises three layers: input layer, hidden layer and output layer. (Sundararajan *et al.*, 1999).

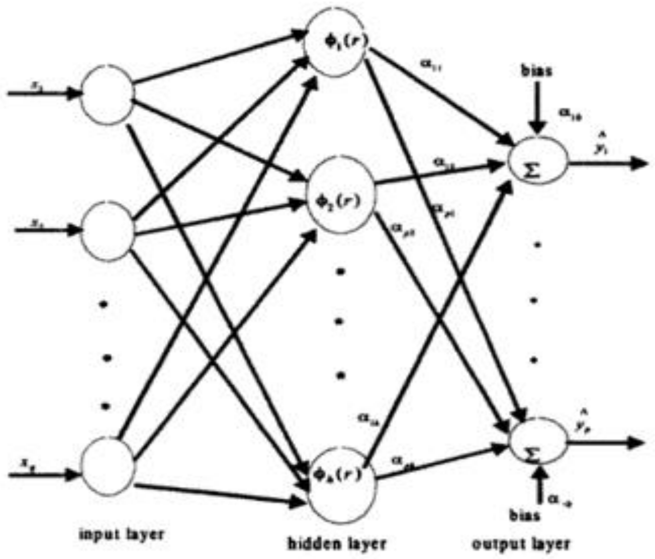


Figure 2-2 The basic structure of RBF neural networks

Adaptive control based on direct exploitation of the non-linear multivariable dynamic model of a wastewater treatment system has been presented. The model was designed from mass balance equations and, combined with a joint observer estimator for on-line tracking of the unavailable states and parameters. It was estimated that the dissolved oxygen concentration was the only measurable state variable of the system. An adaptive linearising regulator has been applied to control and maintain the pollutant substrate and the dissolved oxygen concentrations at required levels by acting respectively on the dilution rate and the air flow rate (Nejjari *et al.*, 1999; Nejjari *et al.*, 1997).

Multivariable data-based model has been developed for dissolved oxygen and nitrate in wastewater treatment (Ghavipanjeh, 2006). Then the Proportional-Integral-Plus (PIP) control method has been developed using the identified multivariable model to control and maintain dissolved oxygen and nitrate simultaneously. The robustness of the controller has been investigated and the response of the controller was better compared to the multi-loop controller response.

Artificial neural networks have been applied to a nonlinear biochemical process (alcohol fermentation). Simulation results revealed the ability of artificial neural networks to model nonlinear systems when measurements were affected by noise. The benefit of this control structure is that it requires very simple mathematical devices for the control movement calculation. Linear model predictive control (LMPC) and proportional-integral-derivative (PID) controllers have been achieved and the results compared with the neural network model based predictive control (NNMPC) strategy (Nagy *et al.*, 2007).

The population balance equations (PBE) for the cell mass distribution to the substrate mass balance has been used to formulate the dynamic model of the continuous yeast bioreactor. Moreover, the model was solved mathematically by spatially discretising the PBE using orthogonal collocation on finite elements (Zhu *et al.*, 2000). The results of the nonlinear ordinary differential equation model was linearised to yield a linear state space model that accurate for MPC synthesis. The implementable control strategy has been developed for oscillating yeast cultures.

The feedback linearisation approach, which transforms a nonlinear process into a linear process, has been presented for the SISO and the (non-square) MIMO case (te Braake *et al.*, 1998). This approach was established on both an affine and non-affine neural network model of the nonlinear process. This model is nearly feedback-linearised by means of a state feedback. The resulting nearly feedback-linearised system contains of an affine neural model and a nonlinear state feedback, whereby the combination of the neural model and nonlinear state feedback performs as same as a specified linear input-output model.

A nonlinear model predictive control (NMPC) approach using recurrent neural networks (RNN) has been developed for a single-input (dilution rate), single-output (cell con- centration) process to control the uncertain nonlinear process. The modelling of the networks has been investigated using a recurrent Elman network using back propagation through time (BPTT). Then comparison between the NMPC design and the tuned IMC-PI controller was investigated and the results demonstrated that NMPC performed better and is good for controlling a bioreactor. (Sivakumaran *et al.*, 2006).

Khare and Singh (2010) proposed a conventional PID controller and internal model control and model based control technique to control the temperature of outlet of the heat exchange system. The results revealed that internal model control gives better response compared to the conventional PID.

The water quality found in rivers, ponds and lakes is very important in human life. A single feedforward artificial neural network model and multiple neural networks were investigated to improve water quality index prediction in a river basin located in Perak, Malaysia. The investigation's findings revealed that multiple neural networks improves the water quality by reducing the mean square error, compared with a single feedforward artificial neural network (Ahmad *et al*., 2017).

Neural networks has another type of network called multilayer perceptron, it has widely used in chemical process. It consists of an input layer, one or more hidden layer to calculate the nodes and output layer. Figure 2.3 illustrates the general structure of multilayer perceptron (Sapuan and Mujtaba, 2009). Ay and Kisi, (2011) presented a comparison between multilayer perceptron, radial basis neural network and multilinear regression for estimating the daily dissolved oxygen of Fountain Creek have been investigated. The results shown that the MLP and RBNN improves the model estimation by analysing the temperature and dissolved oxygen of water and mixtures at downstream and upstream stations than the MLR models.

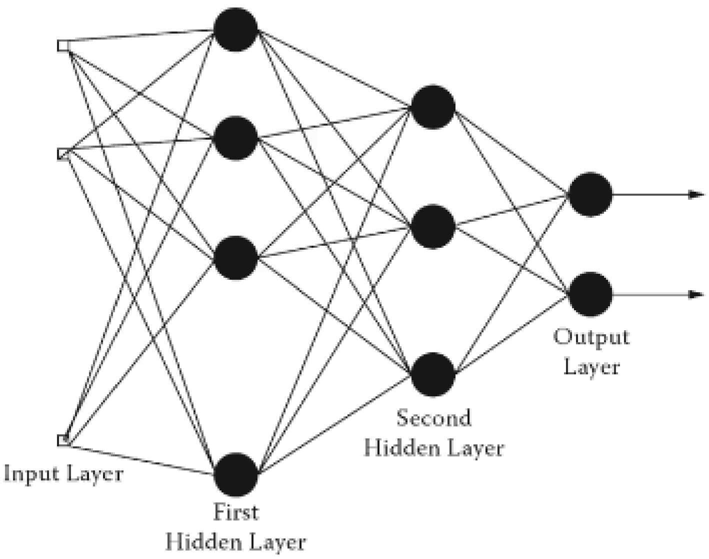


Figure 2-3 Multilayer perceptron structure (Sapuan and Mujtaba , 2009)

Multilayer perceptron (MLP) neural networks have been used to model multivariable chemical process. This method was chosen after comparison with radial basis function (RBF) networks. Then model predictive control were applied to control the chemical process and the results shown the ability of this strategy to control chemical process (Yu *et al.*, 1999).

System identification and control of pH process have been investigated using real-coded genetic algorithm, the process was identified by genetic algorithm then the PID controller parameters were also tuned using this method. The results shown that the output response tracking the set point change with small overshoot and fast rise time (Valarmathi *et al.*, 2009).

System identification of dissolved oxygen (DO) and pH processes have been investigated using recursive least squares method. The results shown that the ability of this technique to estimate the model parameters. Then self-tuning generalized minimum variance (ST-GMV) control were applied to the fermentation process to control the pH and DO levels and this controller provided improved results for the pH and dissolved oxygen at the required level (Hitit *et al.*, 2017).

System identification of pH process have been investigated using nonlinear Hammerstein-wiener model. Then the results have been compared with the ARX and nonlinear auto regressive with exogenous input (NARX) models and shown that the performance of nonlinear Hammerstein-wiener model is better than the other two models by giving less mean square error and mean absolute error (Rattanawaorahirunkul *et al*., 2016).

System identification for pH plant has been done using self-organizing map neural network technique. Then multiple-model adaptive controller was implemented to control pH level using the self-organizing map and its results shown results shown improved behaviour with faster and steady transient response compared to the self-tuning regulator controller (Bashivan *et al*., 2008).

The prediction of effluent concentrations of biochemical oxygen demand and suspended solid for wastewater treatment plant has been investigated using artificial neural networks. The results shown the capability of using artificial neural networks to predict wastewater treatment plant (Hamed *et al*., 2004).

Comparison between response surface methodology (RSM) and artificial neural network-genetic algorithm (ANN-GA) have been investigated in fermentation media optimisation. The results shown that the (ANN-GA) is performing 3 times better and improves modelling accuracy compared with the (RSM) (Desai *et al*., 2008).

Neural network back propagation (BP-PID) controller and Fuzzy-PID controller were individually combined with PID to control the dissolved oxygen concentration in the SBR process of paper mill. The authors compared the results with and without noise and found that the fuzzy-PID controller is performing better with noise compared with the BP-PID controller by decreased 10% of aeration energy. However, the fuzzy-PID controller without noise is not doing as good as the BP-PID controller (Shen *et al*., 2016).

2.4 Review of Model Predictive Control

A nonlinear model predictive control (NMPC) has been used widely by many researchers in different fields. For example, neural network model based predictive controller has been applied to a nonlinear multivariable chemical process (Kittisupakorn *et al.*, 2009). It has been used to control the concentrations of pickling in a steel pickling process. The results shown that this method gave better control action against oscillation due to disturbances and robust compared with conventional PI controller. Figure 2.4 illustrates the basic diagram of model predictive control (Findeisen and Allgöwer, 2002).

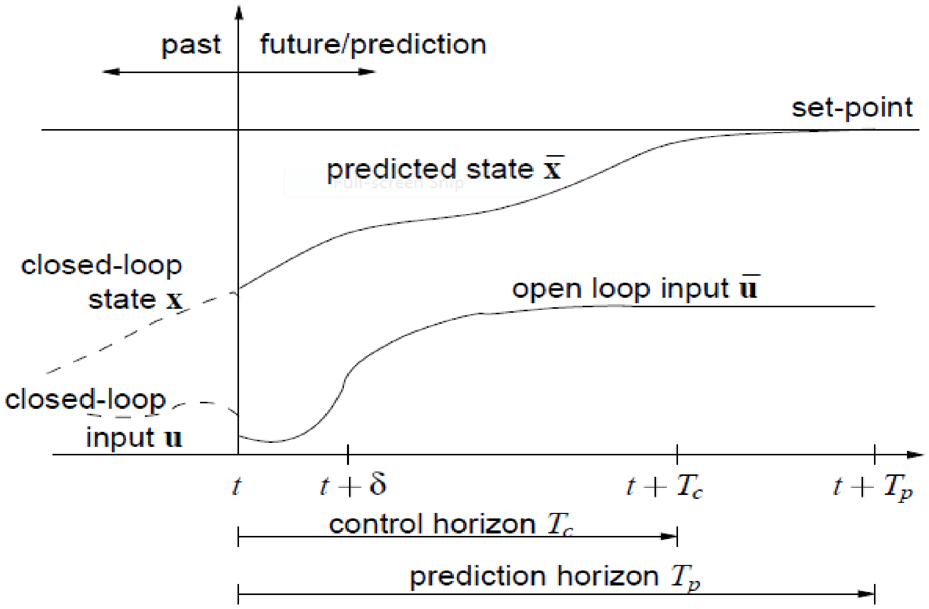


Figure 2-4 Basic diagram of model predictive control (Findeisen and Allgöwer, 2002)

The wiener model predictive control (WMPC) has been developed to control pH process. The results and responses were compared with linear model predictive control and PID controller (Norquay *et al.*, 1999). The performance of MPC exhibits better response than the linear MPC and PID control by reducing the amount of buffering and modelling error in the system.

Model identification of a pH neutralisation process has been investigated using Wiener–Laguerre model. The model is used in a nonlinear model predictive control framework based on the sequential quadratic programming (SQP) algorithm. Second order nonlinear polynomial along with two Laguerre filters has been selected after several experiments. The authors compared the NMPC controller based on Wiener–Laguerre model with the MPC controller based on a wiener model (Mahmoodi *et al.*, 2009). The results showed that with the NMPC the quality of modelling with the rate of convergence of SQP improved in a reasonable time. In addition, Allgöwer and Zheng, (2012) presented that feedback model predictive control needs to cope with uncertainty and is suitable for linear system. However, the ability of feedback model predictive control for nonlinear systems is still an issue.

Nonlinear model predictive controller based on fuzzy kalman filter and augmented state fuzzy kalman filter has been applied to continuous stirred tank reactor to control temperature. The results shown acceptable servo and regulatory behaviour (Prakash and Senthil, 2008).

Nonlinear model predictive control based on self-organizing migrating algorithm have been developed in pH process. The authors used this technique to control and maintain the pH value and the results were compared with the adaptive PID controller, and it was observed that the (NMPC) performed better than adaptive PID controller by minimising response overshoot and computational time (Degachi *et al.*, 2018).

Model predictive control based PID controller has been used to control temperature process. Then the results were compared to conventional PID and fuzzy PID controller, the authors presented that the performance of model predictive control based PID controller exhibits better transient characteristics than the other two methods with no overshoot, delay time, less rise time and fast settling time (Poongodi and Sudhanan, 2015).

Multi-rate nonlinear model predictive control based on neural networks have been applied to multivariable chemical process. The simulation results shown that this method was performing better than the PID controller with fast response, small overshoot and less mean square error (Yu and Yu, 2007).

Nonlinear Hammerstein model algorithmic control algorithm has been applied to control pH process. Then the results shown that the (NHMAC) provides good stability and robustness in modelling despite with large error modelling compared with the linear model algorithmic control (LMAC) and the nonlinear PID controller (Zhiyun *et al.*, 2013).

Neural network model predictive control has been applied to control multivariable process. The control results shown that the set-point tracking and disturbance rejection were acceptable and better than PID controllers (Yu and Yu, 2003). In Holenda *et al* (2008) model predictive control has been applied to control dissolved oxygen in activated sludge wastewater process.

Nonlinear model predictive control has been used to control pH neutralisation process. The authors developed neural network wiener identification method to the process. The results demonstrated that the response of nonlinear model predictive controller was improved without any overshoot and with less error compared with linear model predictive control (Arefi *et al.*, 2006). However, model predictive control is facing a significant issue due to nonlinearity behaviour. Moreover, nonlinearity does affect implementation, which is required generally at each time solution of an optimal control problem. The significant difference in mathematical programming is not between linear and nonlinear, but between convex and nonconvex (Allgower and Zheng, 2000).

2.5 Adaptive Fuzzy Logic Control Review

Self-tuned fuzzy logic control has been developed and applied to a laboratory scale pH neutralisation systems. The adaptive fuzzy controller was evaluated to control different pH operating regime and the results support of using this method by giving faster response time, better settling time and less set-point deviation (Singh *et al.*, 2015).

According to Bhandare and Kulkarni (2015) presented that the fuzzy logic controller gave improved settling time and low overshoot compared to the conventional PID controller in controlling the liquid level.

Boobalan *et al.* (2013) applied fuzzy based PID controller to continuous stirred tank reactor to control temperature. Then the results have been compared to conventional PID controller and Ziegler Nichols method. The response of fuzzy based PID controller was performing well and gave better set change tracking than the ziegler-nichols conventional PID controllers.

In order to control reactor concentration and temperature in continuously stirred tank reactor, multi model adaptive fuzzy controller has been investigated. This was achieved by linearising the system around various operating regimes. Their results shown a good performance of the controller response (Gogoria *et al*., 2015).

Wang and Yuan (2012) used self-tuning fuzzy PID controller to control grate cooler pressure based on Kalman filter. The results shown that the heat energy recovery efficiency has been improved.

2.6 Local Linear Networks Modelling and Control of Nonlinear Systems

Local linear model networks have been used by many researchers to study the behaviour of nonlinear systems. Johansen and Foss (1992a, 1992b, 1993), presented that the concept and the idea of local model networks is to split the operating regime into sum of local models which are suitable for the valuation of a non-linear system.

System identification for nonlinear systems has been proposed by Johansen and Foss (1995). The researchers divided an operating regime into several local models and each local model contains a validity function. LLMNs is recognised as a generalisation of a radial basis function network (RBFN), in which individual neurons are replaced by local sub-models with basic functions. These functions define the regions of validity of individual sub-models, according to the expected operating regions of a plant (Murray-Smith and Johansen, 1997). The local linear model networks (LLMNs) is a powerful technique dealing with nonlinear systems, such as identifying a heat recovery steam generator system that has been achieved with least error (Jamali and Jazayeri-Rad, 2010).

Local linear model tree (LOLIMOT) is known as a class of Takagi-Sugeno-Kang neuro fuzzy technique, it has demonstrated its capability compared with other neuro-fuzzy networks in learning nonlinear systems and pattern recognition (Pedram *et al*., 2006). A local linear model tree (LOLIMOT) method which does not require the parameters of the fuel cells has been applied to examine dynamic modelling of a solid oxide fuel cell (SOFC) (Marzooghi *et al*., 2012). Local linear model networks (LLMN) can be a very effective tool used for modelling. The inversion of LLMN yields real possibilities to involve the networks in the control systems problem (Nentwing *et al*., 2010).

The local linear model has a simple structure, can be implemented easily and is suitable for non-linear system identification since the output error can be used as a criterion for structure selection; meanwhile, the prediction error can be applied to parameter estimation (Nelles and Isermann, 1996). Furthermore, it is fast in nonlinear system identification for the turbocharger with measured signals during road driving (Nelles *et al*., 1996).

A fuzzy online system identification for a single input single output (SISO) nonlinear system has been presented by (Xie and Rad 1999). It has been formed using the local linear dynamics. The authors used the measurements of input and output were used to identify the continuous-time fuzzy input output model. Simulation results have been established that the fuzzy on-line identifier has the ability to match the time-varying nonlinear system within ±5% accuracy.

A locally linear RBF network-based state-dependent autoregressive (LLRBF-AR) has been modelled by employing local linear RBF networks. This method incorporates advantages obtained from a state dependent autoregressive model in nonlinear dynamics explanation and approximation of new nodes. LLRBF-AR model is far preferable to existing models. The other distinct advantage of LLRBF-AR over the RBF type-based models is the smaller number of centres required for prediction accuracy (Gan *et al*., 2010).

Weighted outputs for local models have been used as a form of output of local linear model networks (LLMNs). Identification of the construction features of LLMNs could also be described using local models. Interpretability is one of the benefits of LLMNs, which allows the total model to be divided into smaller operating regimes (Hametner and Jakubek 2013).

Gao *et al*. (2002) presented a non-linear model predictive controller using local model networks and local control networks for continuous stirred tank reactor CSTR. The researchers applied this method to system identification and control of a non-linear system. The simulation results revealed the advantage of using the combination of local controller network and model predictive controller for nonlinear systems, as this method allows the controller to perform well for all operating regions.

The local linear model networks technique was used to identify the non-linear *pH* process and the identified LLMN model of the *pH* process was tested as an independent model, PI and local linear model controllers for different operation points were achieved (Abdelhadi *et al.*, 2014).

Attempts have been made to reduce the complexity of control problems in the chemical process by using a local linear model tree structure. According to Petchinathan *et al*. (2014), the ability of using local linear model tree structure was used for system identification of the *pH* neutralisation process. They compared this method with an adaptive neuro fuzzy system and the results demonstrated that local linear model networks performed better and achieved less mean square error. Moreover, settling time in control response was less than that in adaptive neuro fuzzy system. Zhang and Morris (2001) proposed nonlinear model predictive control strategy using a recurrent neuro-fuzzy network has been applied to a pH neutralisation process. This was done by dividing the operating region to several local linear models. The results revealed that this method improve the control performance and gave satisfactory response.

Nelles and Tomizuka (2000) investigated the local linear model networks for nonlinear system identification. A comparison was also made of the nonlinear autoregressive with exogenous input (NARX) model structure and new nonlinear orthonormal basis functions (NOBF) model structure. The researchers presented that the NOBF method is an encouraging alternative for identification of nonlinear system.

Internal model control strategy based on local linear models has been applied to the control of the *pH* neutralisation process. The results were compared with internal model control based on multi-layer perceptron neural networks and demonstrated that a local linear model controller performs better due to the oscillatory behaviour of the controllers based on multi-layer perceptron neural networks (Kharaajoo *et al.*, 2003).

The internal model control (IMC) scheme has been used extensively to the control chemical process. Fink and Nelles (2001) presented the nonlinear IMC based on local linear models for heat exchanger. The results were compared with conventional PID controller and revealed that the IMC achieves a good control performance compared with the conventional PID controller.

The comparison between locally linear model tree LLMT algorithm, radial basis function (RBF) neural network and multi-layer perceptron (MLP) neural network have been done in a cosmetics industry application. The results revealed that the (LLMT) performed better and yield lower estimation error (Vahdani *et al.*, 2012)

Local Linear model networks can be applied to the fermentation process. As described by Foss *et al.* (1995), the simulation results of nonlinear model predictive control using local linear model networks in the fermentation process demonstrate the ability of using this technique in such a complex system.

2.7 Motivation for an LLMN Approach to Chemical Process Control

Recently, the use of local linear model networks for nonlinear dynamic systems has become well known. According to Nelles *et al.* (2000), there are some benefits to using local linear model networks. Firstly, local linear model networks can be fast in a training network, as it does not need a long time. Secondly, their structure is simple and the model can be formulated in a set of Takagi Sugeno fuzzy rules. Thirdly, they are reliable and capable of estimating the accuracy of the model's prediction for the data and the same data can produce the same model. Local linear model trees technique has been used for system identification of nonlinear dynamic system by a combination of generalized orthonormal basis functions and local linear model trees (LOLIMOT). The main idea is to approximate an unknown function from data by the interpolation of numerous local linear models (Nelles, 1997).

2.8 Summary

This chapter has considered and discussed a review of researchers and publications pertaining to chemical processes. From the literature review, it was observed that nonlinearity is the most significant challenge that are facing control of chemical processes. It can be demonstrated that there remains a need for a good controller performance to solve these issues that are facing nonlinear processes. There are many applications have been used in the control area, such as radial basis function neural networks, multi-layer perceptron neural networks, and local linear model networks. This thesis introduces the investigation of local linear model networks strategy to identify and control *pH*, dissolved oxygen and temperature processes. The aim is to design a local linear model controller around different operating points and then combine them together to achieve a good control performance.

Chapter 3 | Development of the Methods for Designing Local Linear Models for System Identification and Control of Real *pH* Data

3.1 Introduction

Overall, the difficulties of sustaining the *pH* value at an optimum value is a common issue in the chemical industry, such as waste water treatment or fermentation processes. This is due to the nonlinearity and dynamic behaviours of the *pH* process. Therefore, a good controller should be considered to maintain the *pH* value at the optimum point in order to obtain of product of good quality (Sung *et al.*, 1995).

This chapter describes the benefit of using local linear model networks to identify and control a nonlinear *pH* process. The process contains input, which is flow rate of base (f_b), and the *pH* as an output. The local linear model networks (LLMNs) technique was selected for its ability to reduce the mean square error by comparing the results with radial basis function neural networks and adaptive neuro-fuzzy inference system (ANFIS) models. The procedures and results of these comparisons are discussed in this chapter. Then, PI and local linear model controllers were designed and applied to the process of controlling pH.

3.2 *pH* Process Characteristics and Simulink Model Implementation

The purpose of this section is to investigate and develop the best control method to maintain *pH* at a desired value in a chemical processes. PI controller was applied to the *pH* Simulink model and the results reveal the ability to use a PI controller to control and maintain the value of *pH*. This was done to understand the dynamics and nonlinearity behaviours of the *pH* process and investigate the good controller for the *pH* process.

3.2.1 The Dynamic Model for CSTR

The *pH* neutralisation process investigated in this research is derived from first principles and verified by experimental results by (McAvoy *et al.*, 1972). The model contains weak acid, which is neutralised by a strong base and the description for the flow dynamics of concentrations of the influent compositions into the continuous stirred-tank reactor (CSTR) followed by a static non-linearity characterising the physio-chemical equilibrium conditions between these concentrations. Figure 3.1 illustrates the *pH* neutralisation process which will be developed in this section using the dynamic model presented by McAvoy and this process has flow rate of base and acid as inputs and *pH* as output.

The equation of *pH* is defined mathematically by the negative decimal logarithm of the hydrogen ion concentration $[H^+]$ as the following:

$$pH = -log_{10}[H^+] \tag{3.1}$$

Furthermore, to evaluate the unknown value of the H^+ the flow rate of acid and base of *pH* neutralisation process model are applied. The dynamic model for the CSTR is as follows:

$$\mathrm{F_1C_1 - (F_1 + F_2)\xi = \frac{Vd\xi}{dt}} V \tag{3.2}$$

$$\mathrm{F_2C_2 - (F_1 + F_2)\varepsilon = \frac{Vd\varepsilon}{dt}} V \tag{3.3}$$

Where ξ and ε are concentrations of acid and base, F_1 is the flow rate of weak acid, C_1 is concentration of acid and flow rate of strong base F_2 with concentration of base C_2

This chapter considers the case of acetic acid $\mathrm{CH_3COOH}$ (weak acid) as constant (0.081 l/min) neutralised by sodium hydroxide NaOH (strong base) as varying (0.1 to 0.7 l/min). Operating conditions of the process are presented in Table 3.1. Figure.3.2 illustrates the Simulink model, which is constructed using this derivation of the dynamic model to characterise the *pH* neutralisation process between acetic acid $\mathrm{CH_3COOH}$ (weak acid) and sodium hydroxide NaOH (strong base) (Regunath and Kadirkamanathan 2001).

Table 3.1 pH process parameters

Variable	Definition	Nominal setting
V	Volume of tank	1L
F1	Acid flow rate	0.081 l/min
F2	Base flow rate	0.512 l/min
C1	Acid concentration	0.32 moles/l
C2	Base concentration	0.05005 moles/l
Ka	Acid equilibrium constant	1.8×10^{-5}
Kw	Water equilibrium constant	1.0×10^{-14}

It is noticeable from equation 3.1, that the *pH* value can be determined by identifying the H^+. Depending on the chemical species used, the *pH* titration curve (Figure 3.3) varies. Further, the electro-neutrality condition states that the sum of the charges of all ions in the solution must be zero. Based on the titration curve and electro-neutrality condition, a non-linear algebraic equation can be described as stated in equation 3.4.

$$[\mathrm{H}^+]^3 + [\mathrm{H}^+]^2\{\mathrm{K_a} + \varepsilon\} + [\mathrm{H}^+]\{\mathrm{K_a}(\varepsilon - \xi) - \mathrm{K_w}\} - \mathrm{K_a}\mathrm{K_w} = 0 \tag{3.4}$$

By substituting the total concentrations of acid, ξ and base, Ɛ from equation 3.2 and 3.3 into equation 3.4, the concentration of hydrogen ions, H^+ can be obtained to solve equation 3.1.

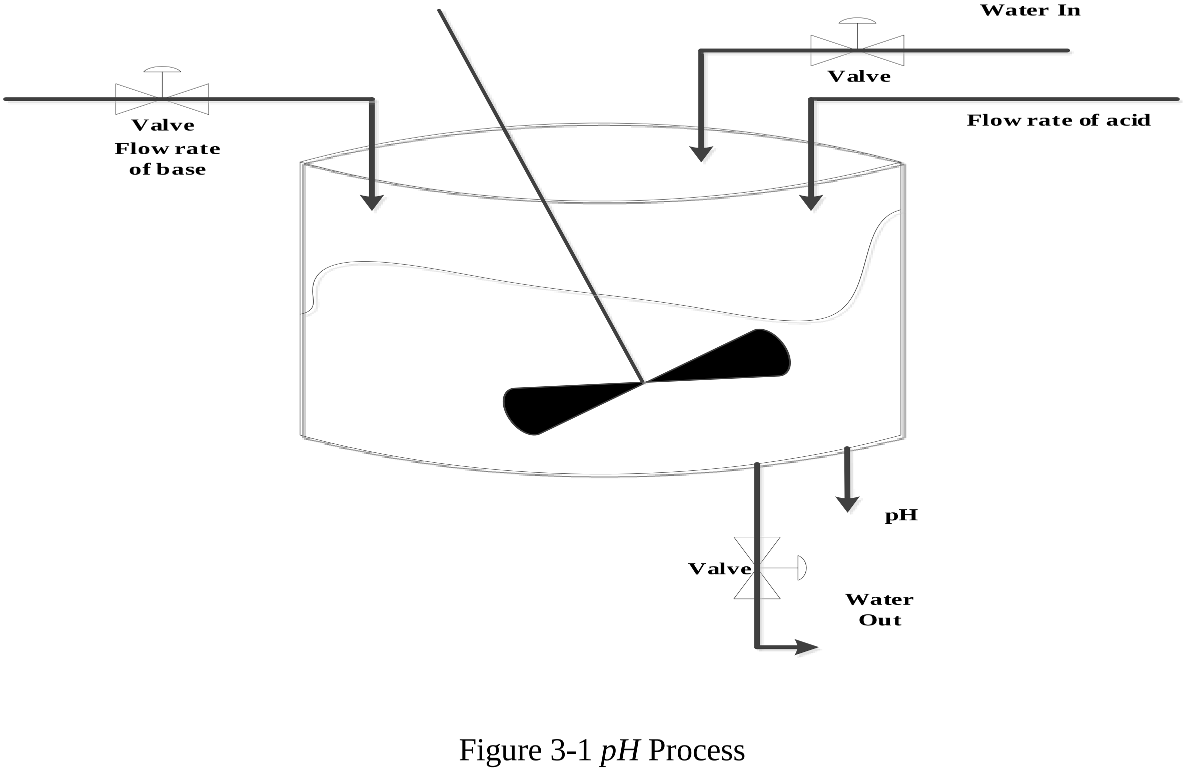


Figure 3-1 *pH* Process

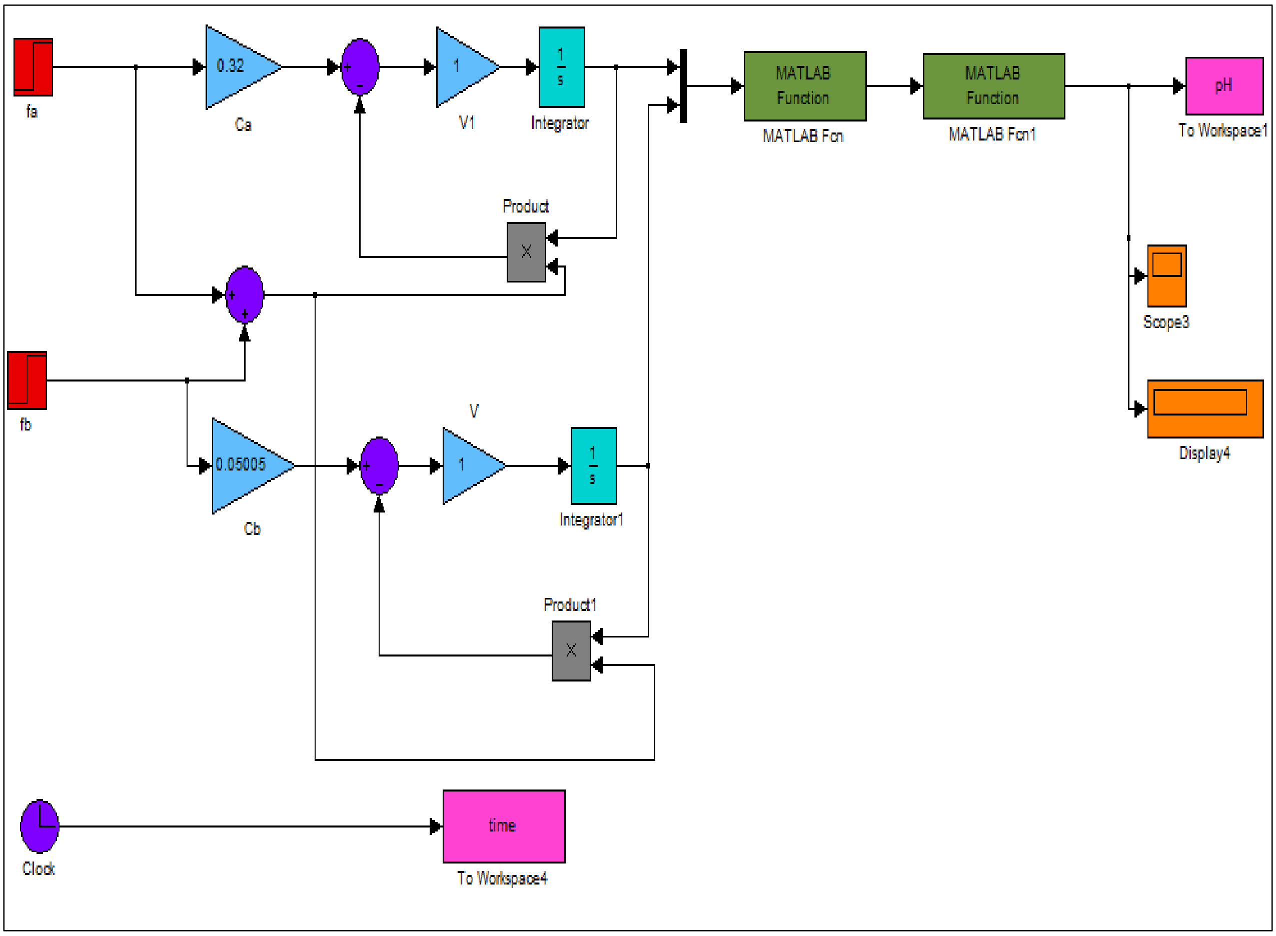


Figure 3-2 Simulink model of pH neutralisation process

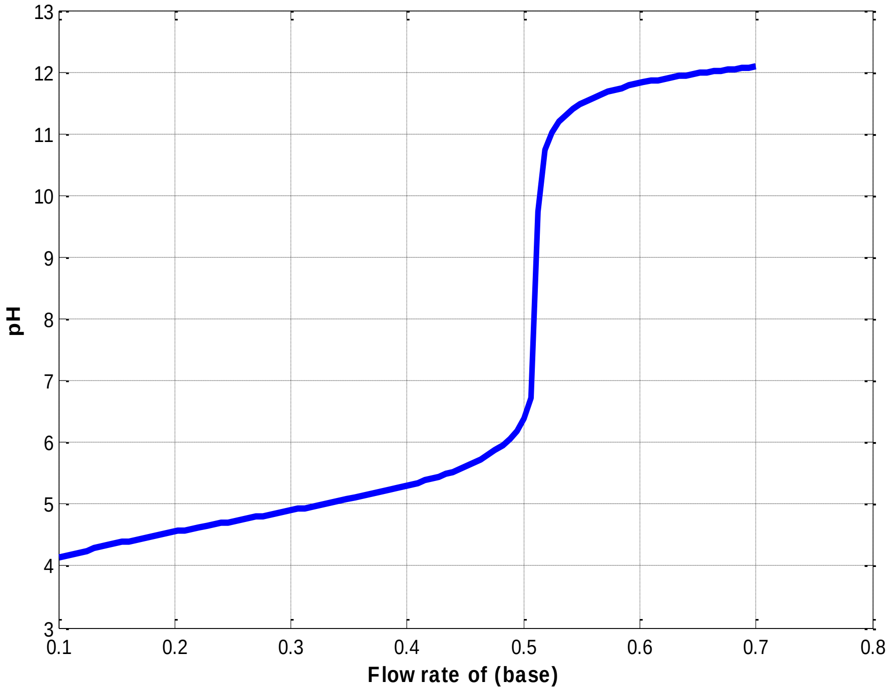


Figure 3-3 pH titration response

3.2.2 PI Controller Design Based on Internal Model Control (IMC) for the System

The internal model control (IMC) technique was applied to identify a suitable PI control parameter. This is performed to investigate and get familiar with the dynamic behaviour of *pH* process under the influence of PI control. The mathematical computation and simulations results are discussed in more detail later in this chapter.

3.2.2.1 PID control

PID controller has been used widely since the 1950's and remains an effective method today. PID controllers are used in industrial systems such as to control and regulate variables of chemical processes for example, *pH*, temperature and dissolved oxygen. PID control contains the following three properties:

1. Proportional control

$u(t) = ke(t)$ $\qquad\qquad$ $D(s) = K$

2. Integral control

$u(t) = \frac{K}{T_I}\int_0^I e(\mu)d(\mu)$ $\qquad\qquad$ $D(s) = \frac{K}{T_I s}$

3. Derivative control

$u(t) = KT_D\dot{e}(t)$ $\qquad\qquad$ $D(s) = KT_D s$

Where T_I is the integral time or rest time, T_D is the derivative time, and K is the position feedback gain. Therefore, the transfer function can be combined as in equation (3.5).

$$D(s) = \frac{u(s)}{e(s)} = K(1 + \frac{1}{T_I s} + T_D s) \tag{3.5}$$

Proportional feedback control can reduce errors to disturbances. The Proportional, Integral and Derivative controllers are combined to form the classical PID controller (Franklin *et al.*, 1998).

As the process does not have time delays, consequently, a first-order model without delay is a more reasonable method for designing the PI controller (see equation 3.7). The first-order model has a very important property which is a step response for process steady-state gain (K) obtained from the calculation of ratios of steady-state change in the value of output (*pH*) to step change in input size f_b (equation 3.6). The plotting for this reaction is named a reaction curve. Clearly the input is fixed to a new value and held there, thereby allowing the process to reach a steady-state (Seborg *et al.*, 2004).

$$\mathrm{K} = \frac{\Delta pH}{\Delta f_b} = \frac{\mathrm{y_2 - y_1}}{\mathrm{u_2 - u_1}} \tag{3.6}$$

Here, ΔpH is the difference between set point change for the *pH* output ($\mathrm{y_1}$and $\mathrm{y_2}$) and calculated by:

$$\Delta\mathrm{pH} = \mathrm{y_2 - y_1} = 5 - 4.5 = 0.5$$

Where Δf_b is the difference between set point change for the input ($\mathrm{u_1}$and $\mathrm{u_2}$) and calculated using the following:

$$\Delta f_b = \mathrm{u_2 - u_1} = 0.336 - 0.19 = 0.146$$

Hence the gain,

$$\mathrm{K} = \frac{0.5}{0.146} = 3.42$$

Here, sampling time is chosen to be 0.1 sec and the time constant $T = 2.9sec$ was obtained from the open loop response which is presented in Figure 3.4. Note that this result is consistent with the 63.2% method.

$$pH = \Delta pH \times 0.632 + \text{initial value for } pH = 0.5 \times 0.632 + 4.5 = 4.81$$

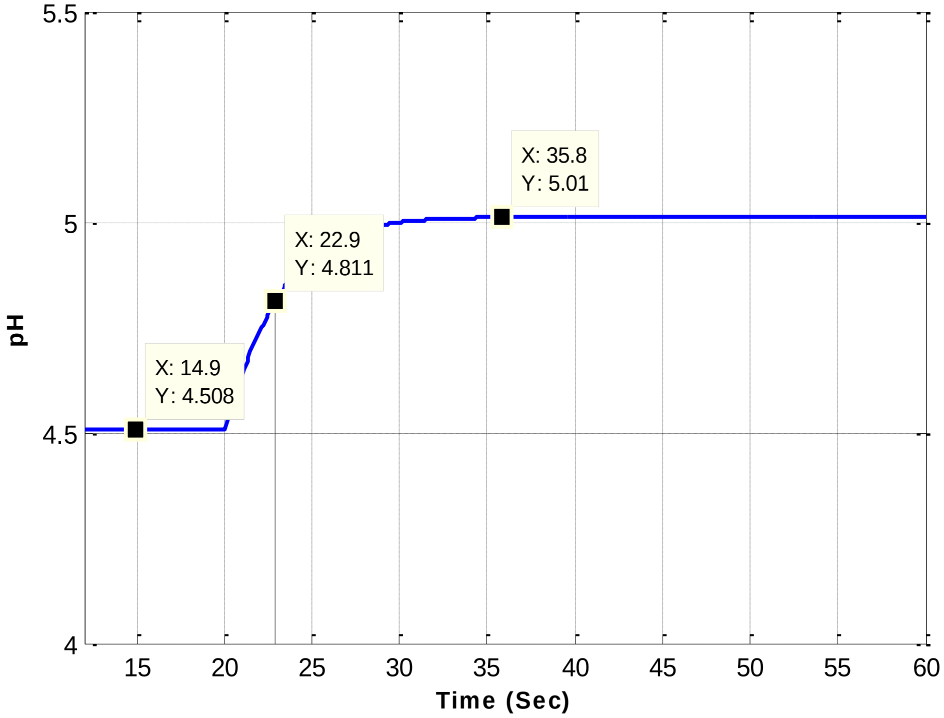


Figure 3-4 Open loop step response of *pH* Simulink model

Therefore, the resulting process model is

$$\text{Transfer function} \qquad \mathrm{G} = \frac{\mathrm{K}}{\mathrm{Ts+1}} = \frac{3.42}{2.9\mathrm{s}+1} \tag{3.7}$$

This transfer function is applied to the *pH* process as can be seen in Figure 3.5. Following the simulation, the results revealed an acceptable response for the model (see Figure 3.6).

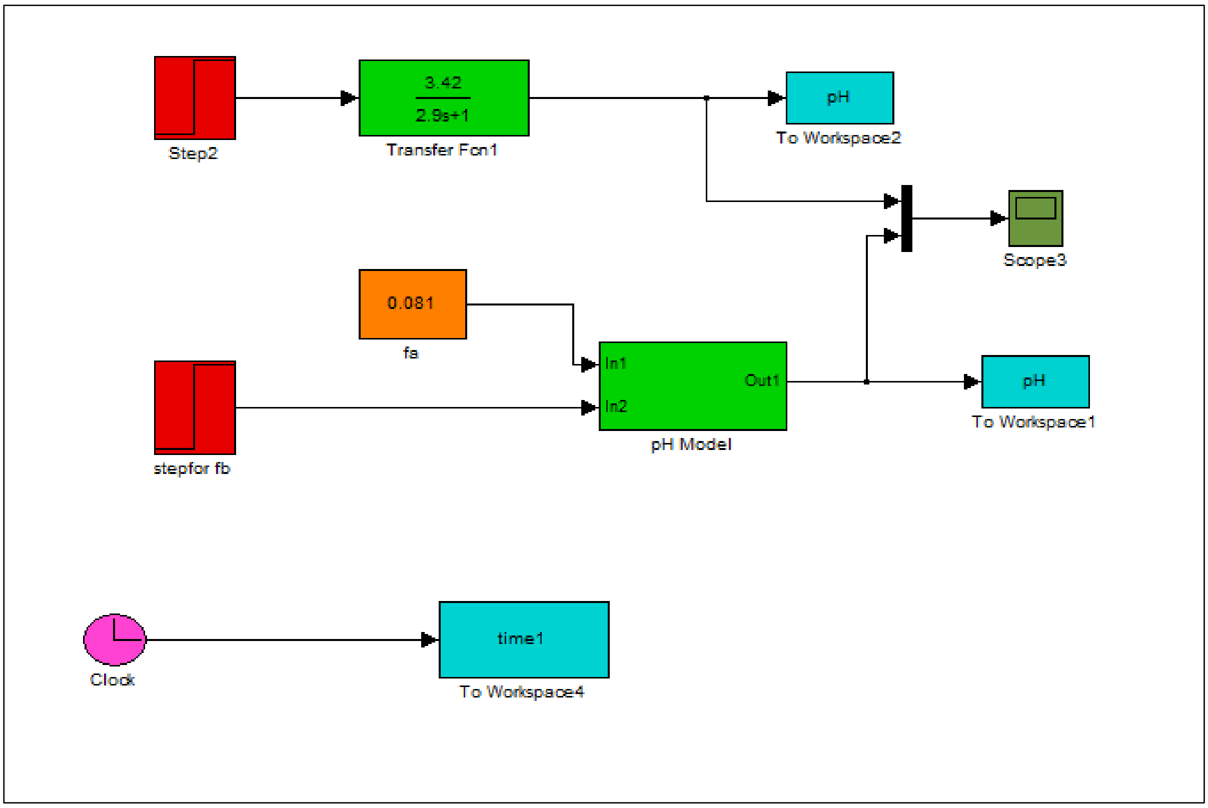


Figure 3-5 Transfer function and *pH* Simulink model

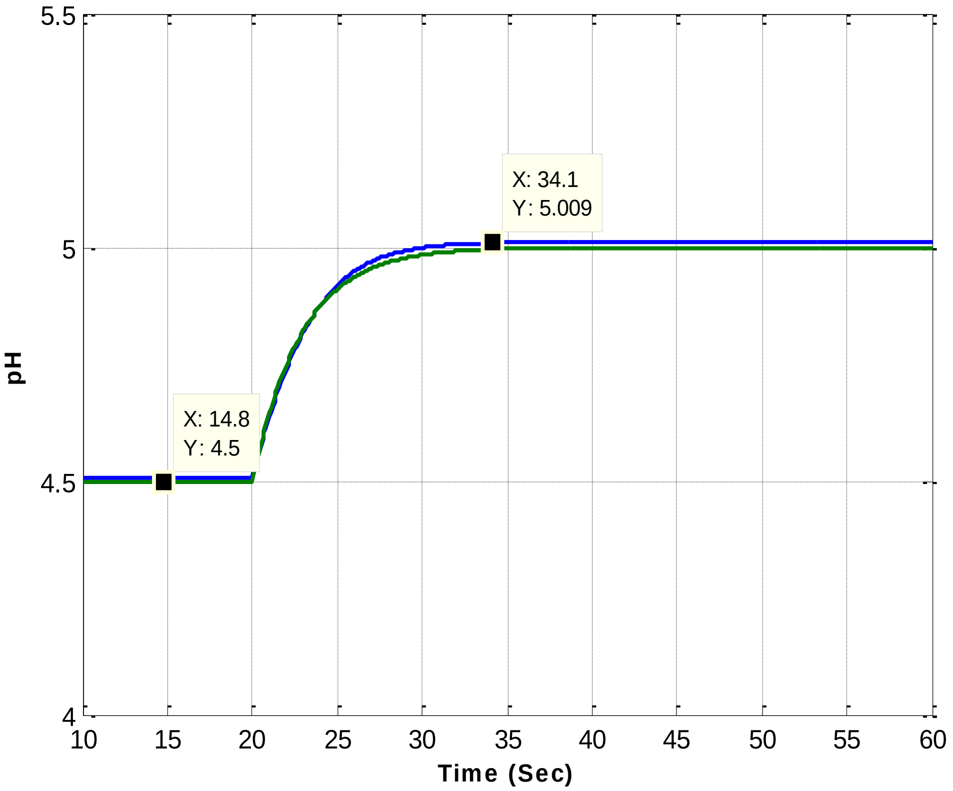


Figure 3-6 Open loop step response of *pH* with transfer function response

A table for PID controller tuning relations for the parallel form was derived by Chien and Fruehauf, (1990) for common types of process models and is presented in Appendix B. Figure 3.7 illustrates how the PI controller based on internal model control is applied to the *pH* model and if we analyse Figures 3.8 and 3.9 response for PI with and without noise for operation point from 4.5 to 5 was achieved. This method demonstrates an acceptable control performance for chemical processes.

Table 3.2 presents the PI gains for the four regions. After obtaining the steady state gain and time constant in each region then Proportional Integral (PI) controller parameters were calculated by the internal model control method as follows:

$$\tau_c = \frac{T}{2} = \frac{2.9}{2} = 1.45 \text{ sec}$$

$$K_C = \frac{T}{\tau_c K} = \frac{2.9}{1.45 * 3.42} = 0.584$$

$$K_i = \frac{K_C}{\tau_i} = \frac{0.584}{2.9} = 0.2013$$

Where K_C is proportional gain and it depends on gain, K, K_i is integral gain, and τ_i is integral time which is equivalent to time constant. τ_c is a key decision in IMC and can be generalised to the time constant as $T > \tau_c$ then τ_c can be selected to be $\tau_c = \frac{T}{2}$. It can be seen from Figures 3.8 and 3.9 that internal model control is a suitable technique for designing PI parameters to control the nonlinear model for small change around an operating point.

Table 3.2 PI gains for five pH regions

pH region	K_C	K_I
4.5-5	0.584	0.2013
5-5.5	0.4	0.17
5.5-6	0.16	0.08
6-6.5	0.064	0.035
6.5-7	0.024	0.0101
7-7.5	0.0075	0.0033

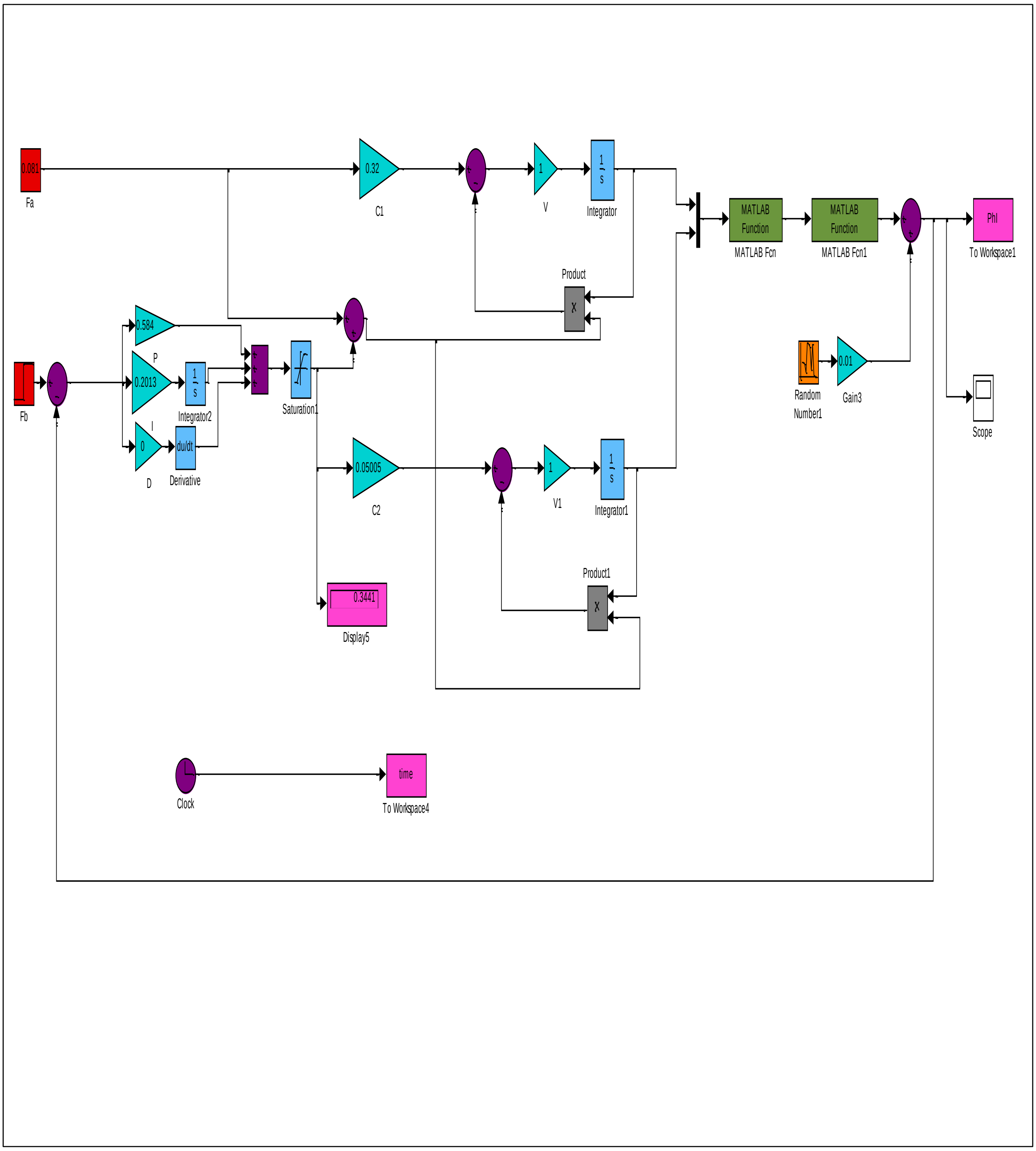


Figure 3-7 Simulink model of *pH* with PI controller

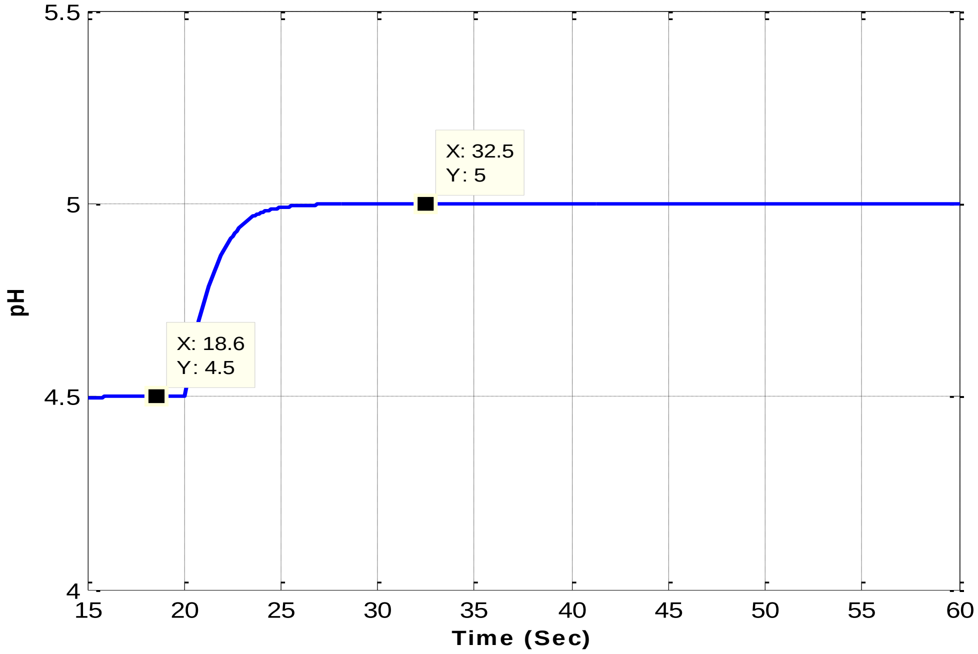


Figure 3-8 Response of PI controller without noise for *pH* process

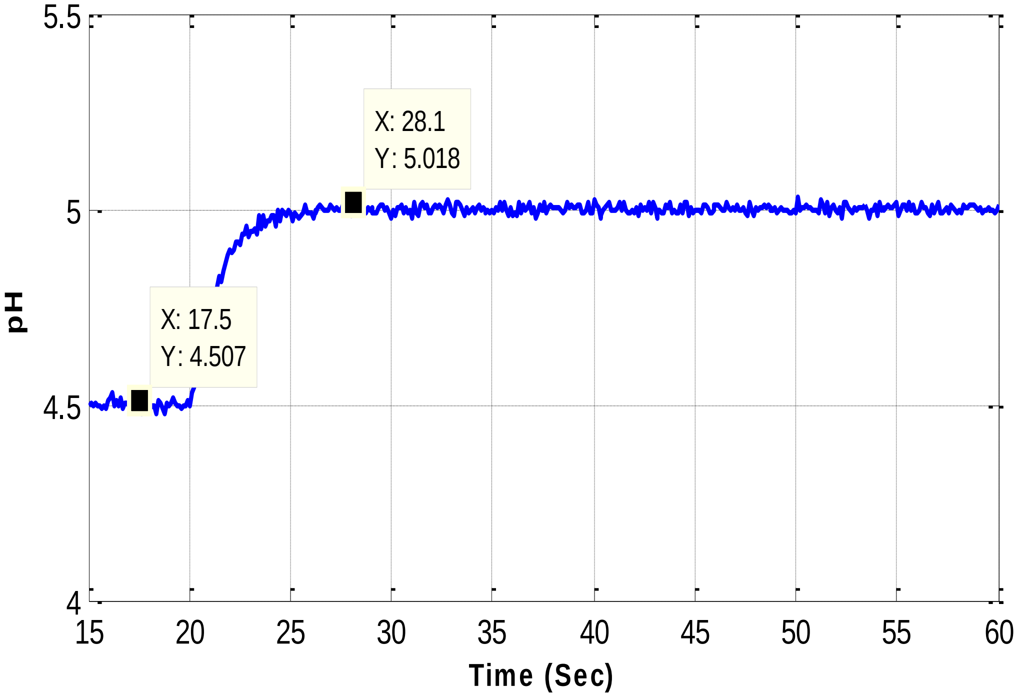


Figure 3-9 Response of PI controller with noise for *pH* process

3.3 Investigation of Artificial Neural Networks for Identification of Real *pH* Data

Artificial neural networks have been used extensively in numerous fields, such as in industrial systems. This section describes and explains the system identification of real *pH* data using radial basis function neural networks. For network training and validation and to yield acceptable results, different model order, different width scaling and different number of hidden nodes are investigated in this section.

3.3.1 Process Description

The experimental *pH* neutralisation process investigated in this chapter is illustrated in Figure 3.1.This process was established in the laboratory to collect the real pH data. It comprises of a continuously stirred tank reactor (CSTR) to which the chemical solutions ammonium hydroxide NH_4OH (base) and acetic acid CH3COOH (acid) are added. The acid flow rate is kept constant and the base flow rate (0 to100 ml/min) is adjustable by a servo pump to regulate the *pH* in the tank. The data was collected with the process under closed loop PID control with the pH set point varied to drive the pH over the operating range between pH 6-8. It was found difficult to tune the PID controller to cover the nonlinear operating range but the data collected was considered suitable for investigating nonlinear identification. The liquid level in the tank (and hence the liquid volume) is maintained at a constant value and the concentrations of the acid and base are both 0.1mol/l. The liquid in the tank is stirred continuously to make sure the pH is consistent throughout the tank. It has been revealed in the experiments that the coupling between variables is very significant. A personal computer with analogue I/O is connected to the process to sample the measurements and issue the control output (Yu and Gomm 2003). The 2000 real data samples are scaled and divided into two parts, 1200 data samples for the network training and the remaining 800 data samples for test. Suitable sample time for all variables was selected to be 10 seconds.

3.3.2 Radial Basis Function Neural Network

There is a three-layered feedforward network, which is an implementation of the radial basis function neural network (RBFNN); the layers are input layer, hidden layer and output layer (Shen *et al.*, 2011). The input layer has signal source units and the hidden layer has a number of nonlinear units which are achieved by the requirements. The relationship from input layer to hidden layer is nonlinear, while from hidden layer to output layer is linear. RBFNN is an activation function of the units in the hidden layer, as illustrated in Figure 3.10. Here $\mathrm{x} = [\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{q}]^T$ is an input vector and $\mathrm{w} = (\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{n})$ are the weights in the output layer. The activation function is Gaussian and represented as $\mathrm{f}_\mathrm{i}(\mathrm{x}) = [\mathrm{f}_\mathrm{i}(\|\mathrm{x} - \mathrm{C}_\mathrm{i}\|)],\ \mathrm{i} = 1, 2, \ldots, \mathrm{n}$, where n represents the number of neurons in hidden layer. The activation function of RBFNN is un-normalised and can be obtained using equation 3.8:

$$\phi_i(k) = f_i\left(\left\|x - c_i\right\|\right) = \exp\left(-\frac{\left\|x - c_i\right\|^2}{(\alpha\sigma_i)^2}\right) \tag{3.8}$$

Where σ_i is the width and C_i is the centre of the activation function, the RBFNN output can then be calculated by equation 3.9:

$$\mathrm{y(k)} = \sum_{\mathrm{i=1}}^{\mathrm{nh}} \phi_\mathrm{i}(\mathrm{k})\mathrm{w}_\mathrm{i} \tag{3.9}$$

Where w_i are the connecting weights from hidden layer to output layer, the networks are compared in terms of the mean square error:

$$\mathrm{MSE} = \frac{1}{\mathrm{M}} \sum_{\mathrm{k=1}}^{\mathrm{M}} [\mathrm{y}_\mathrm{s}(\mathrm{k}) - \mathrm{y(k)}]^2 \tag{3.10}$$

Where M is length of data, $\mathrm{y}_\mathrm{s}(\mathrm{k})$ is the scaled output of the process and y(k) is the prediction output (RBFNN output), the centres c were calculated using K-means algorithm and widths σ in hidden layer nodes of the RBFNN were calculated by P-nearest neighbours method respectively. The recursive least squares (RLS) algorithm was used for RBFNN training to update weights to calculate RBFNN output (Nelles, 2002):

$$\varepsilon(k) = y(k) - \phi_i^{T}(k)\hat{w}(k-1) \qquad (3.11)$$

$$L(k) = \frac{P(k-1)x(k)}{\lambda + \phi_i^{T}(k)P(k-1)\phi_i(k)} \qquad (3.12)$$

$$\hat{w}(k) = \hat{w}(k-1) + L(k)\varepsilon(k) \qquad (3.13)$$

$$P(k) = \frac{1}{\lambda}\{P(k-1) - L(k)[P(k-1)\phi_i(k)]^{T}\} \qquad (3.14)$$

Where

$\varepsilon(k)$ is the prediction error

$L(k)$ is the gain vector

$P(k)$ is the covariance matrix

$\phi_i(k)$ is the activation function outputs

λ is the forgetting factor

$y(k)$ is the target output

3.3.2.1 Recursive Least Squares (RLS)

The idea behind using recursive least squares (RLS) is to calculate a new update for the network weights ($\hat{w}$) each time new data comes in. for each weight update a constant calculation time is needed for the recursive least squares (Nelles, 2002).

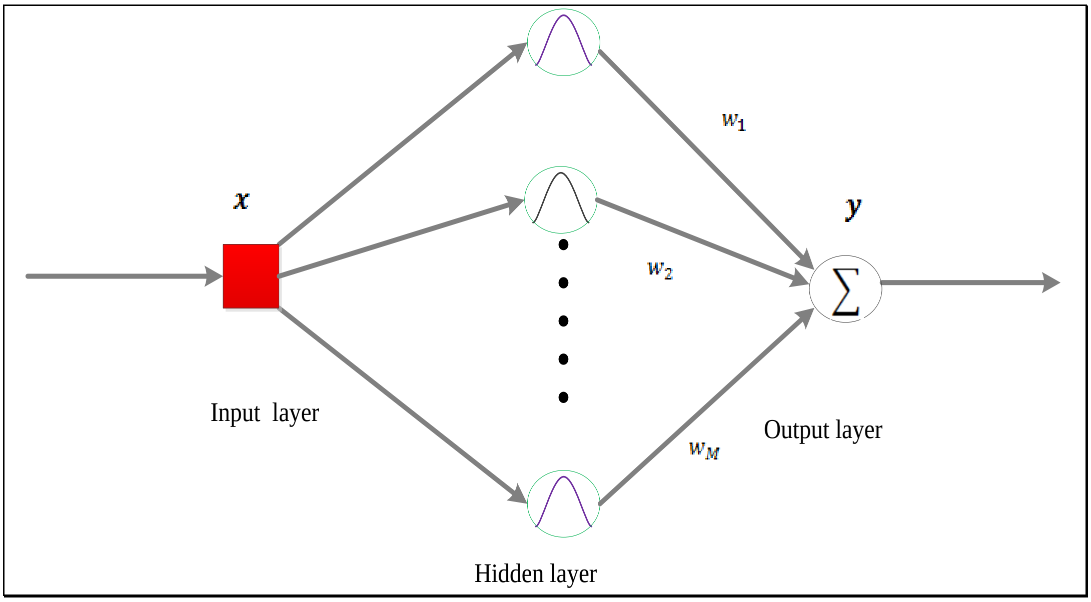


Figure 3-10 Structure of an RBFN network

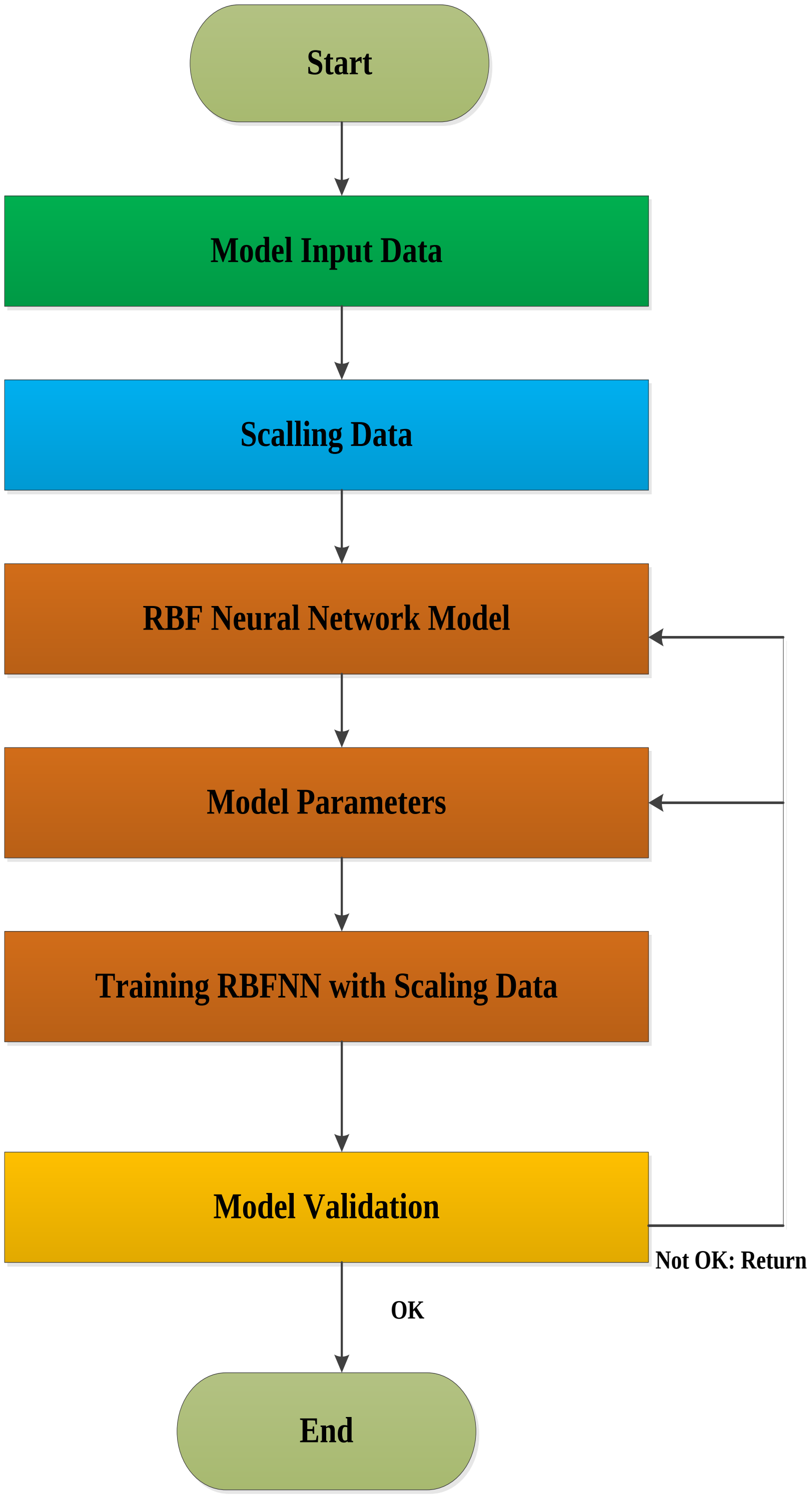


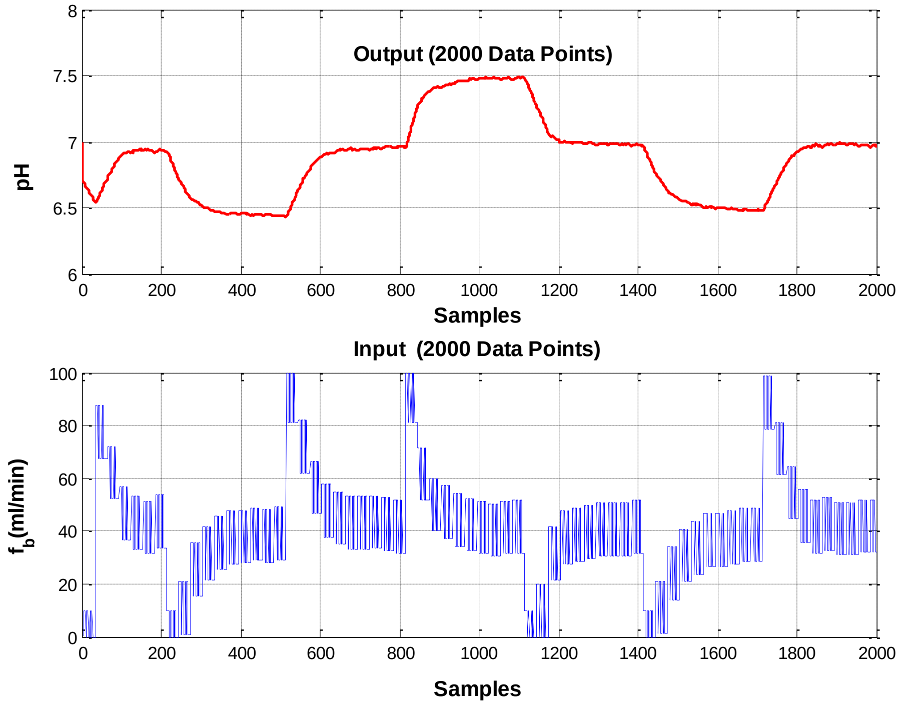
Figure 3-11 RBFNN development steps

3.3.2.2 Real Data Identification Results and Discussion

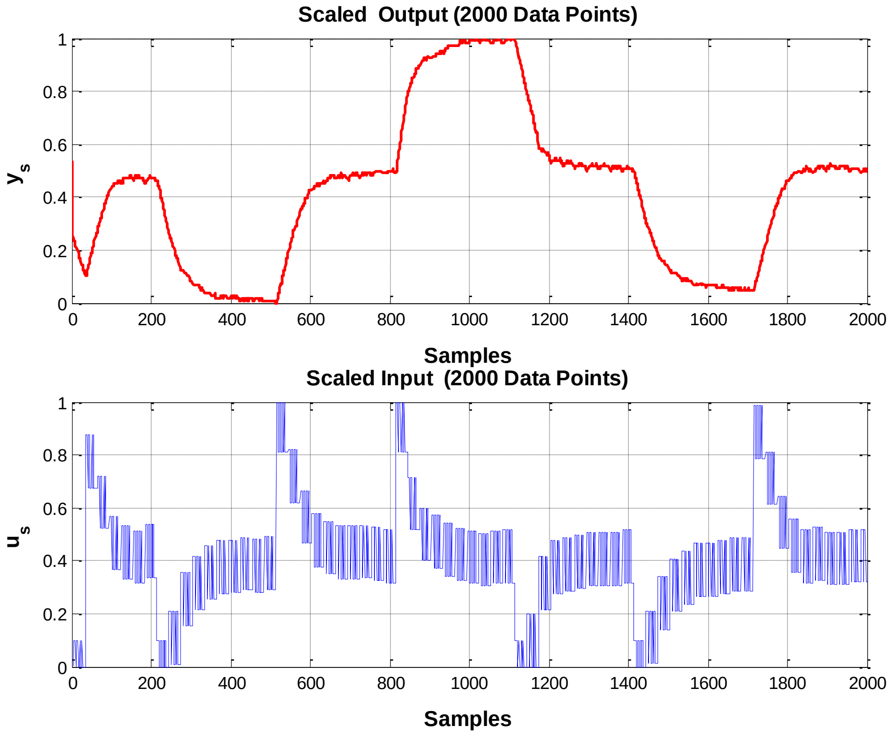
In this section RBFNN is used to predict *pH* output y_s (k) which is scaled for system identification and u_s(k) is the scaled flow rate of base as input. Figure 3.11 presents the RBFNN development steps. The 2000 real data samples need to be scaled before training. The data is split into two part: the first 1200 data samples were used for training and the other 800 data samples for testing and validation. Figure 3.12 a illustrates the 2000 real *pH* data and Figure 3.12 b presents the 2000 scaling data of *pH*, where all the data are scaled using equations 3.15 and 3.16:

$$u_s(\mathrm{k}) = \frac{\mathrm{u(k)} - \mathrm{u_{min}}}{\mathrm{u_{max}} - \mathrm{u_{min}}} \qquad (3.15)$$

$$\mathrm{y}_s(\mathrm{k}) = \frac{\mathrm{y(k)} - \mathrm{y_{min}}}{\mathrm{y_{max}} - \mathrm{y_{min}}} \qquad (3.16)$$



a) Measured real *pH* data for network training and validation (Sample time=10 sec)



b) Scaled real *pH* data for network training and validation (Sample time=10 sec)

Figure 3-12 Measured and scaled real *pH* data for network training and validation. (Sample time=10 sec)

Where u_{min} and y_{min} are lower limits, u_{max} and y_{max} are upper limits of the input and output data respectively, the RBFNN input is set as x vector (see equation 3.17).

$$x(k) = [y(k-1) \ldots y(k-n) \ldots\ldots u(k-1) \ldots u(k-n)]^T \qquad (3.17)$$

The RBFNN has two inputs when order $\mathrm{n} = 1$ as given in equation 3.17, where y(k) in this equation illustrates the scaled *pH* output and u(k) is the scaled flow rate of base as input. In this work, different numbers of hidden layer nodes as well as different order models of network inputs have been used in training experiments. The recursive least squares algorithm is used for training the networks to update weights with initial values set as, $\mathrm{w} = 1.0 \times 10^{-6} \times \mathrm{U}_{\mathrm{nh}\times 1}$,$\mathrm{P}(0) = 1.0 \times 10^{8} \times \mathrm{I}_{\mathrm{nh}}$, where λ here is chosen as 0.999, nh is the number of hidden layers, U is the matrix whose components are ones and I stands for identity matrix. From RBFNN experiments, it was found that if λ is greater than 0.98 and less than 1 it gives less error between measured output and prediction output. Hence a value of λ=0.999 was chosen for further experiments. The RBFNN output is calculated using equation 3.9 after the activation functions have been calculated using equation 3.8 and the weights are then updated using recursive least squares calculations.

The first order model structure n=1 in equation 3.17 with 6 hidden nodes is selected after various experiments that achieve minimum prediction error and best choice for the widths in the hidden nodes by multiplying by a scaling factor ($\sigma \times 20$). The results indicate a first order model, n=1 as shown in Figure 3.13, achieves the smallest train and test errors and this was the order chosen. The effect of varying the widths in the hidden nodes by multiplying by a scaling factor ($\sigma \times \alpha$) is shown in Figure 3.14 for networks with n=1 and nh= 6. It can be seen that the best choice giving minimum prediction error is α=20 ($\sigma x20$ in Figure. 3.14). While Figure.3.15 illustrates comparison between mean square error for RBFNN training and test with different hidden nodes and same model order (n=1) and width scaling (α=20).

It is observed that the best number of hidden nodes is 6 and gives acceptable results and good match between model output and RBFNN output for model training and testing as it is shown on simulation results (see Figures.3.16 and 3.17). Mean square error MSE for training data is 0.000012341 and mean square error MSE for test data is 0.0000077384 . These are low values of MSE indicating good model accuracy. The error for train and test data is presented in Figures 3.18 and 3.19.

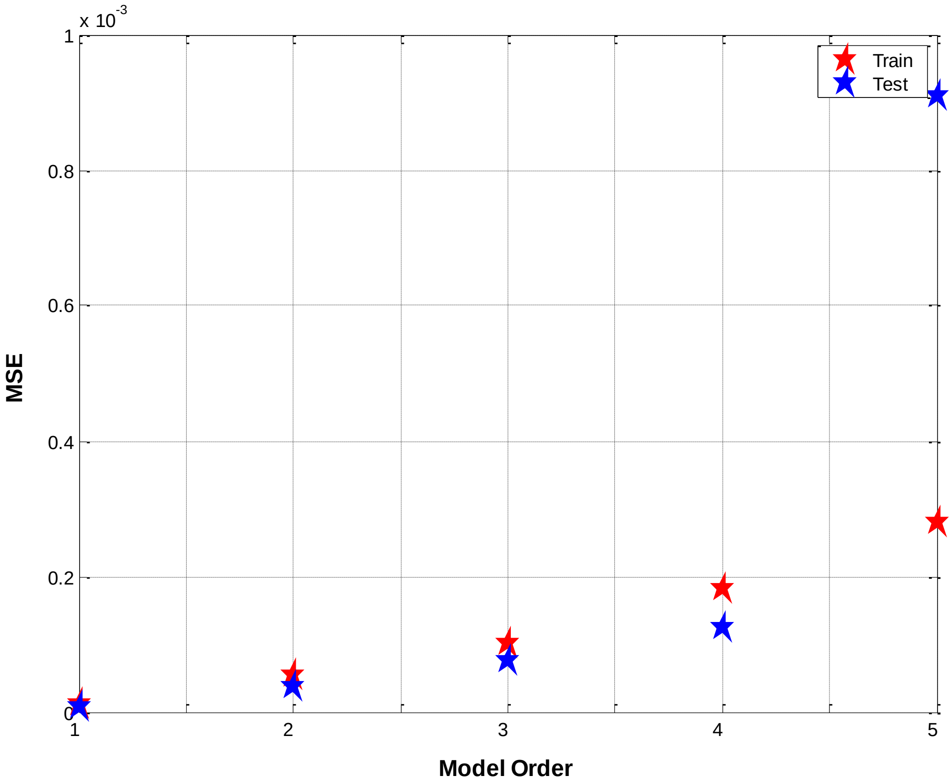


Figure 3-13 Comparison between MSE for RBFNN training and test with different model order

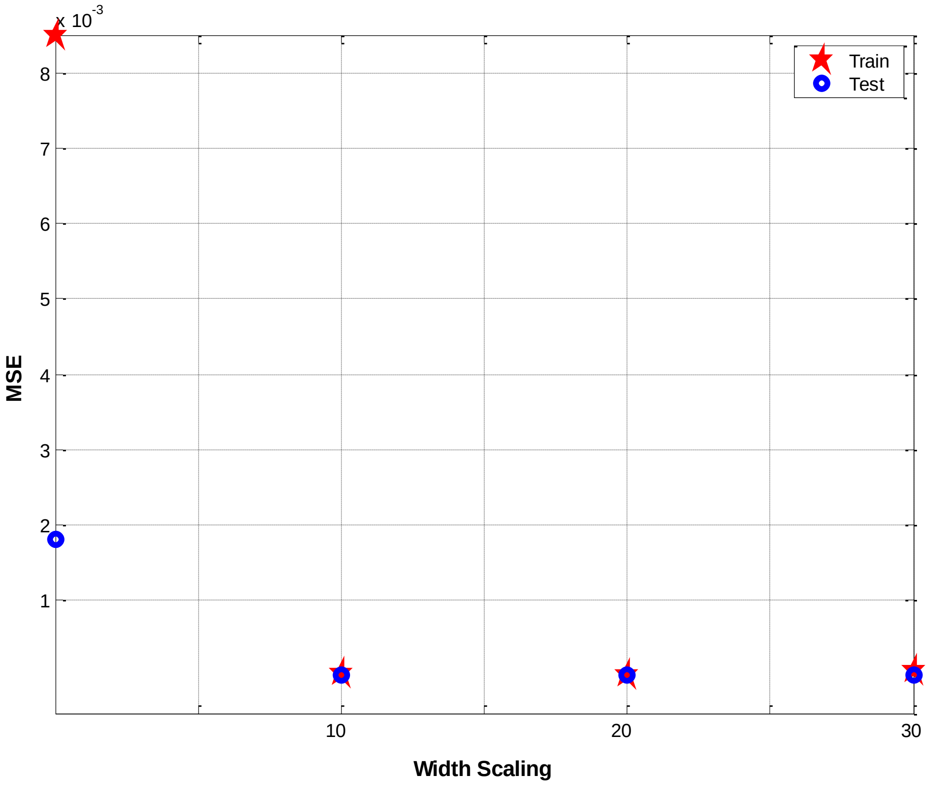


Figure 3-14 Comparison between MSE for RBFNN training and test with different width scaling

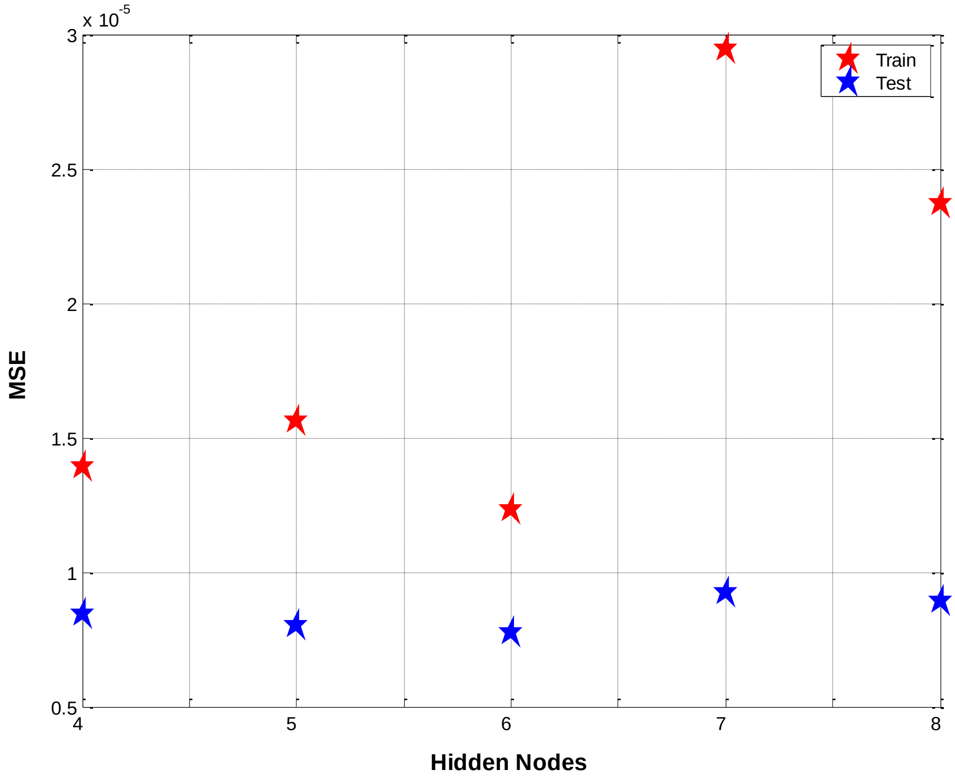


Figure 3-15 Comparison between MSE for RBFNN training and test with different Hidden Nodes

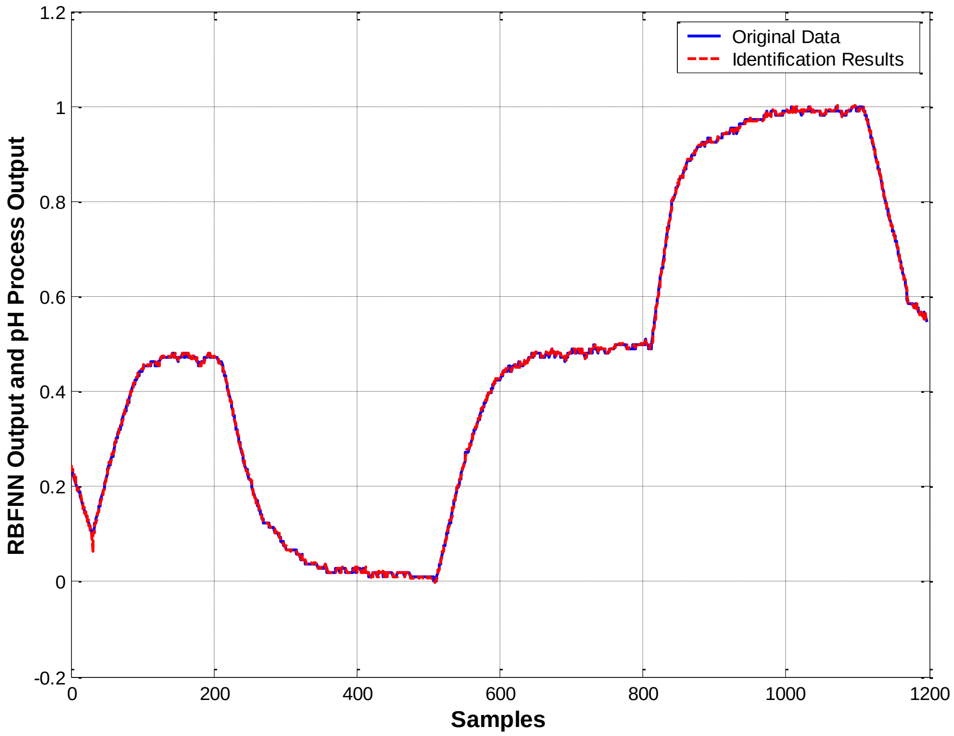


Figure 3-16 RBFNN Identification results on training data for first order *pH* model. (Scaled data, sample time=10 sec)

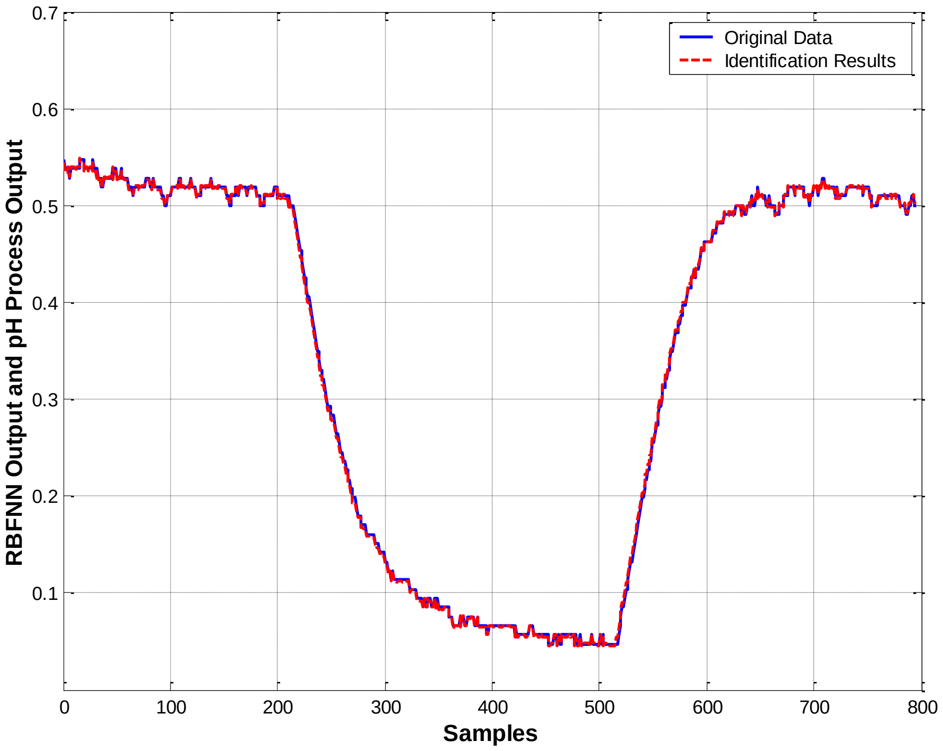


Figure 3-17 RBFNN Identification results on test data for first order *pH* model. (Scaled data, sample time=10 sec)

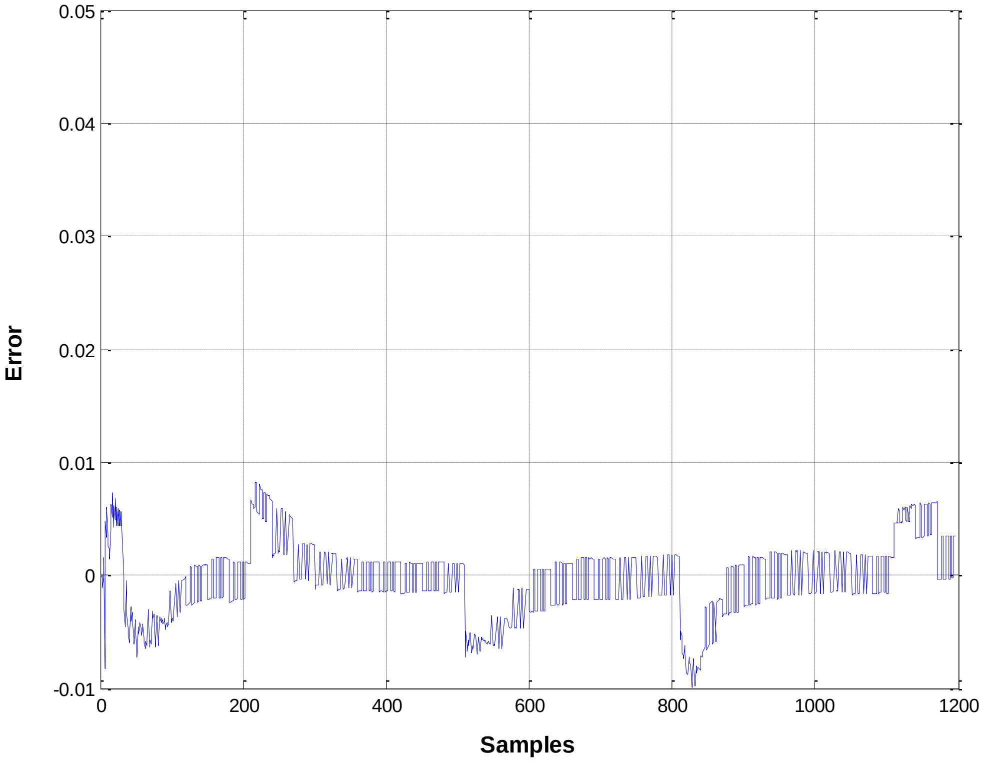


Figure 3-18 Error between RBFNN output and model output on training data.

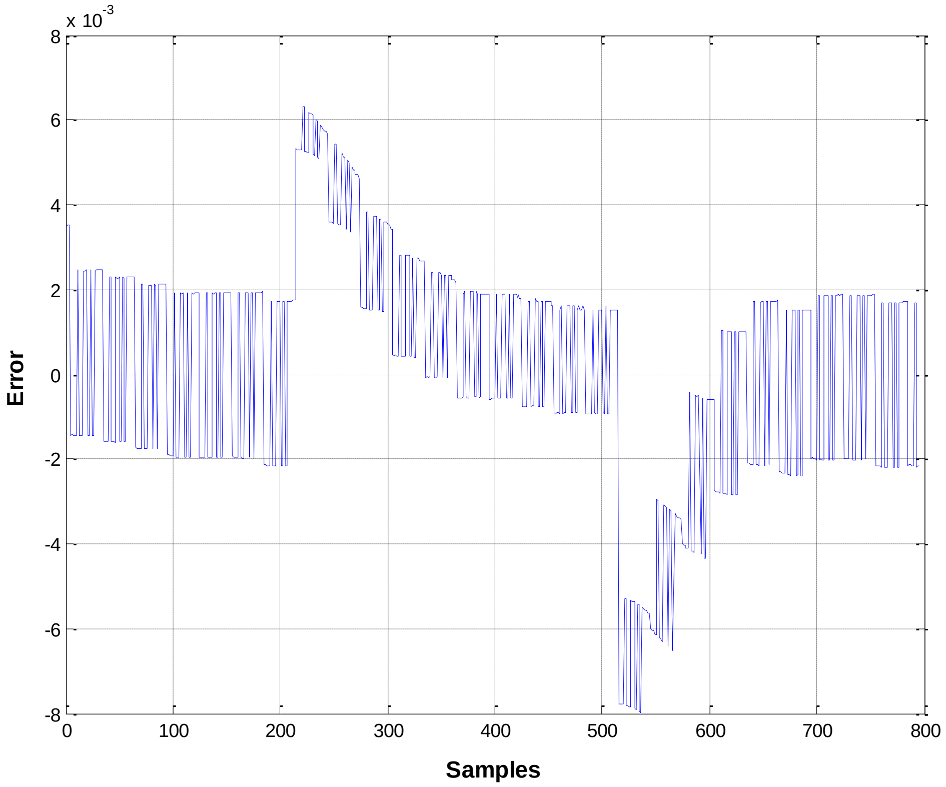


Figure 3-19 Error between RBFNN output and model output on test data.

3.4 Investigation of Local Linear Model Networks for Identification of Real *pH* Data

In the past decade, significant attention has been paid to the application of another type of network, known as local linear model networks (LLMN), for system identification and control. This section explains the development of LLMN for *pH* system identification to predict *pH* output, and the results are also discussed in this chapter.

3.4.1 Modelling of Local Linear Model Networks

System identification of complex nonlinear systems has been achieved by local linear model networks. Figure 3.20 illustrates the structure of dynamic local model networks where the inputs are represented by $\mathrm{x} = [\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{q}]^{\mathrm{T}}$ and the output $\hat{\mathrm{y}}(\mathrm{k})$, respectively. Here, validity function and local linear models are depending on input x. Each model has the validity function ($\emptyset_\mathrm{i}$) and its model parameters (w_i) the local model for the output $\hat{\mathrm{y}}_\mathrm{i}(\mathrm{k})$ is gained by equation 3.18 (Nelles, 2002; Hametner and Jakubek ,2010).

$$\widehat{y_i}(k) = x^T(k)w_i \tag{3.18}$$

$$w_i = [w_{1i} \dots w_{2i} \dots w_{3i} \dots w_{qi}]^T$$

The x(k) contains past inputs and outputs and local affine model structure is implemented for dynamic models:

$$x(k) = [y(k-1) \dots y(k-n) \dots\dots u(k-1) \dots u(k-n)]^T \tag{3.19}$$

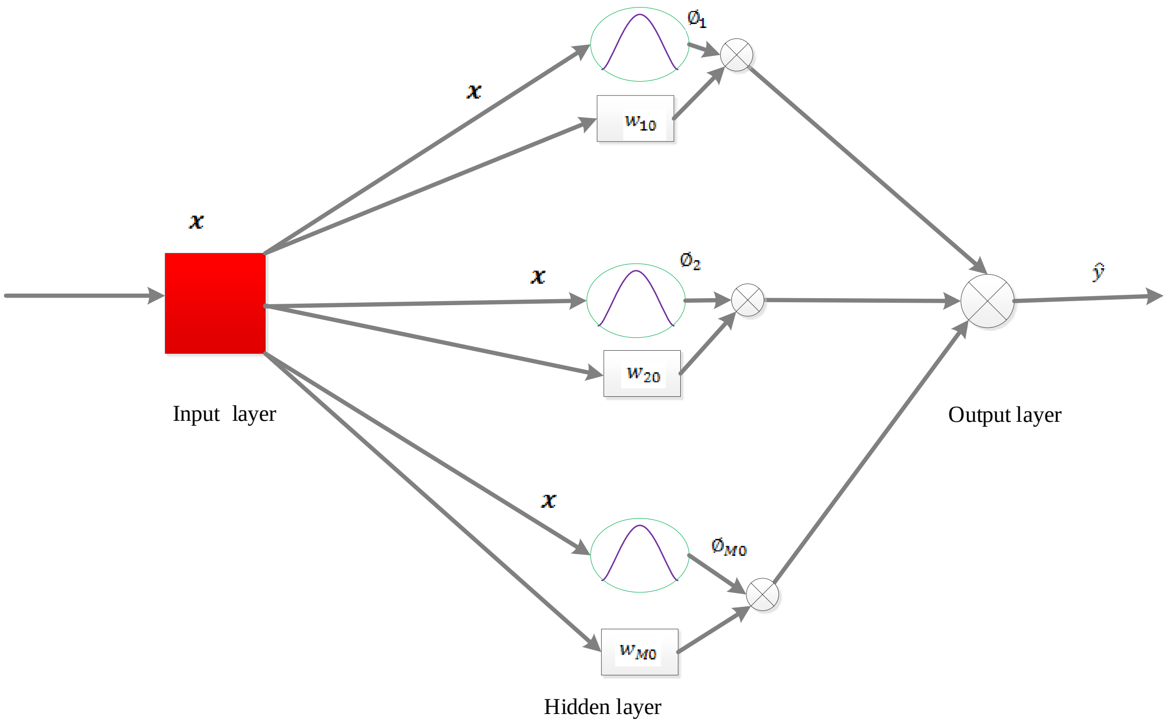


Figure 3-20 Local Linear Model Network Structure

All local estimations $\hat{y}_i(k)$ are used to form the global model output $\hat{y}(k)$ by weighted aggregation.

$$\hat{y}(k) = \sum_{i=1}^{M} \emptyset_i(k)\, \widehat{y_i}(k) \tag{3.20}$$

Here

$$\phi_i(k) = \phi_i(\tilde{x}(k)) = \exp\left(-\frac{\left\|\tilde{x}(k) - c_i\right\|^2}{(\alpha\sigma_i)^2} \right) \tag{3.21}$$

Equation (3.21) describes the calculation of the validity function $\emptyset_{\mathrm{i}}(\mathrm{k})$, which is un-normalised. M stands for the number of local models, σ_i is the width and C_i is the centre of ith activation function. Clearly, the input vector for the membership functions $\tilde{\mathrm{x}}(\mathrm{k})$ could be selected differently from the input vector for the local model (Nelles, 2002). In this chapter, $\tilde{\mathrm{x}}(\mathrm{k}) = \mathrm{y}(\mathrm{k}-1)$ was selected for the validity function calculation. Shorten and Murray-Smith (1994) presented that normalisation of the validity function could have some effect on their shape, such as change in shape potential loss of smoothness of the representation. In addition, these effects may cause the loss of the individuality of local models and cause changeable behaviour in dynamic models. Hence, un-normalised validity functions equation 3.21 are used in this study.

3.4.1.1 Real Data Identification Results and Discussion

In this section, the LLMN structure as presented in Figure.3.20 is used to predict *pH* output y_s (k) which is scaled. The 2000 real data samples were scaled using equations 3.15 and 3.16 in section 3.3 before network training and being divided into two groups. The first 1200 data samples were used for training and the other 800 data samples for testing and validation.

In this section different number of local models and different orders of network model inputs have been used in training experiments and equation 3.19 is selected when model order $\mathrm{n} = 1$. The centres and widths in the validity functions of the LLMN were calculated using K-means algorithm and P-nearest neighbours method respectively. Initial model parameters are set as,$\mathrm{w} = 1.0 \times 10^{-6} \times \mathrm{U}_{\mathrm{M}\times 2\mathrm{n}}$, $\mathrm{P}(0) = 1.0 \times 10^{8} \times \mathrm{I}_{\mathrm{M}\times 2\mathrm{n}}$ and λ is selected as 0.999. Then weights are updated using recursive least square algorithm for model output validation. Figure 3.21. Illustrates the comparison between mean square error for LLMN training and test with different model order and same number of local models which is 4. The results indicate a first order model, n=1, achieves the smallest train and test errors and for this reason the order was chosen. The effect of varying the widths in the local models by multiplying by a scaling factor $(\sigma \times \alpha)$ is presented in Figure 3.22 for networks with n=1 and 4 local models. Figure 3.23 shows comparison between MSE for LLMN training and test with different number of local models with n=1. Figures 3.24 and 3.25 show the training and test results for n=1 and a structure with 4 local models and scaling factor $\alpha = 20$, that was selected after the experiments, and gives minimum prediction errors. From the results of LLMN without bias it can be clearly noticed that its capability to reduce the mean square errors with 4 local models is 0.000010414 for training and for test is 0.0000075046. The errors for train and test data are shown in Figures 3.26 and 3.27. Furthermore, other experiments were conducted, including a bias parameter $\mathrm{b_i}$ which is added to LLMN output as, $\hat{\mathrm{y}}(\mathrm{k}) = \sum_{\mathrm{i}=1}^{\mathrm{M}} \emptyset_{\mathrm{i}}(\mathrm{k})\, \hat{\mathrm{y}}_{\mathrm{i}}(\mathrm{k}) + \mathrm{b_i}$ .The comparison results for the error for train and test data are presented in Figures 3.28 and 3.29. In this case, the mean square error for training is 0.000011049 and mean square error for test is 0.0000081488.

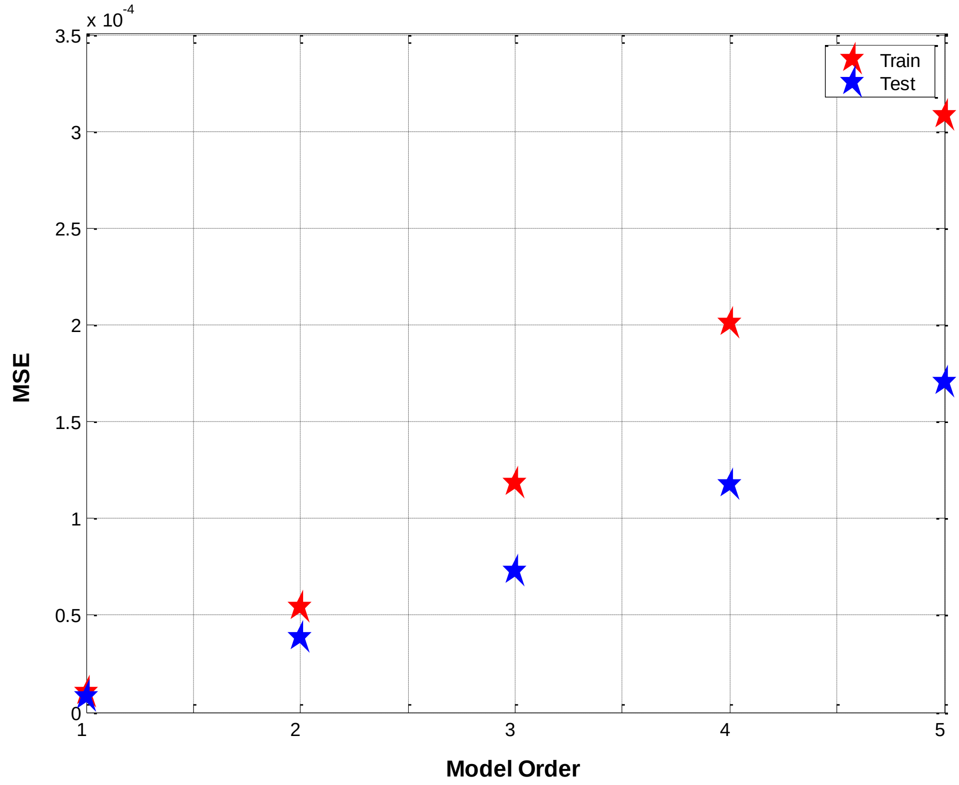


Figure 3-21Comparison between MSE for LLMN training and test with different model

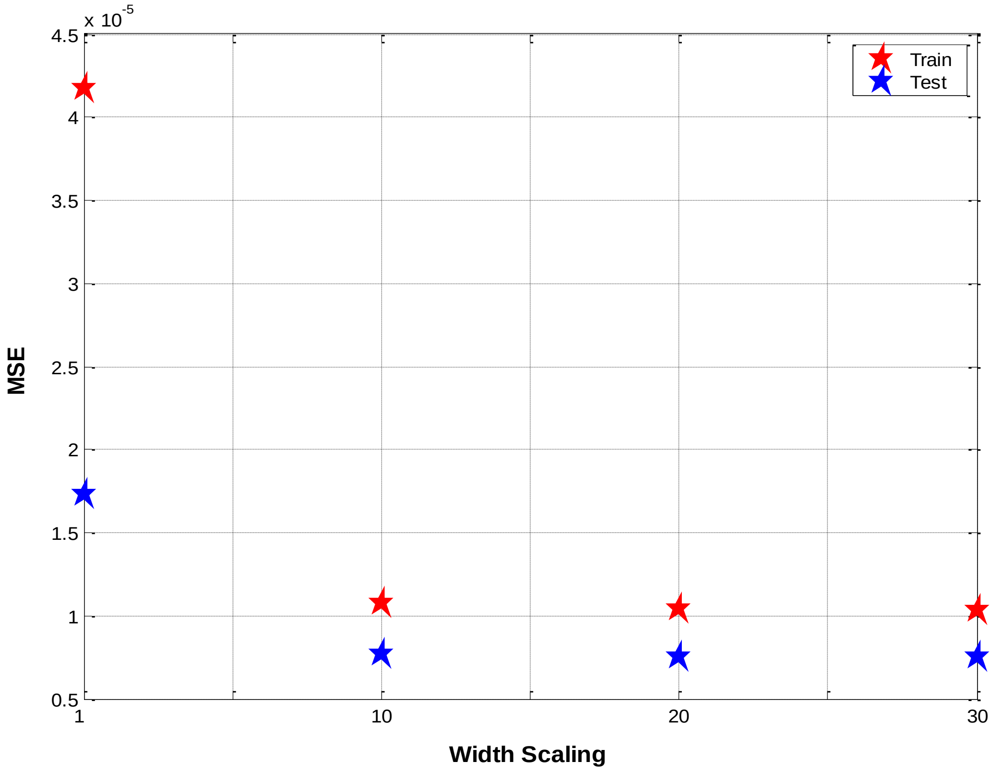


Figure 3-22 Comparison between MSE for LLMN training and test with different width scaling

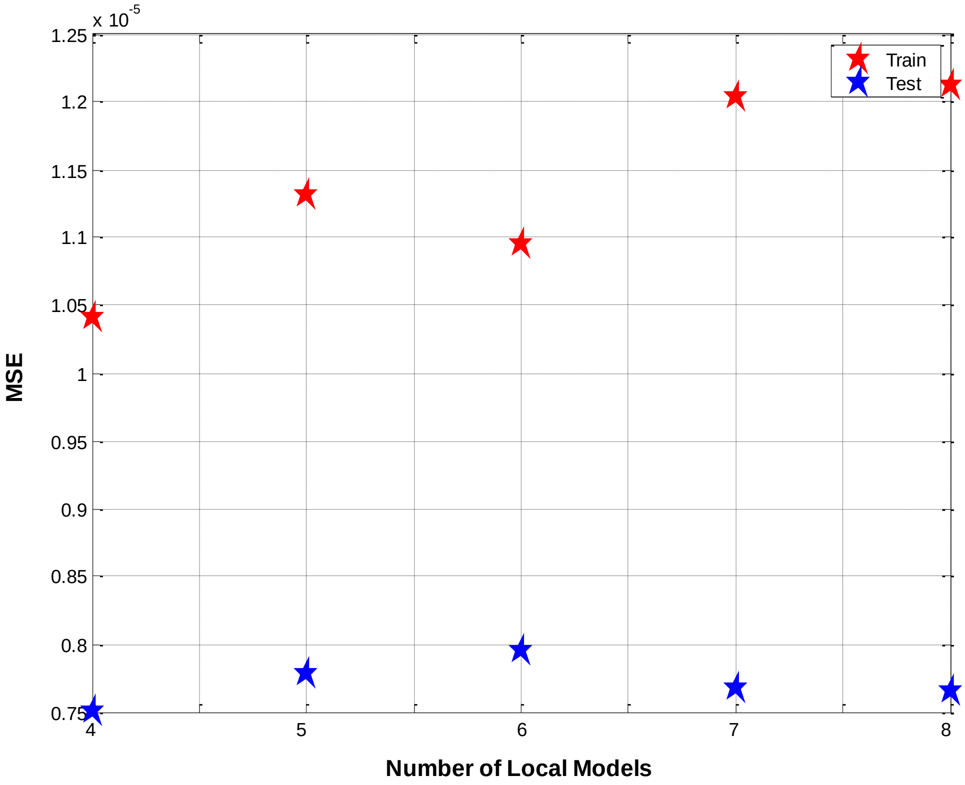


Figure 3-23 Comparison between MSE for LLMN training and test with different Number of local models

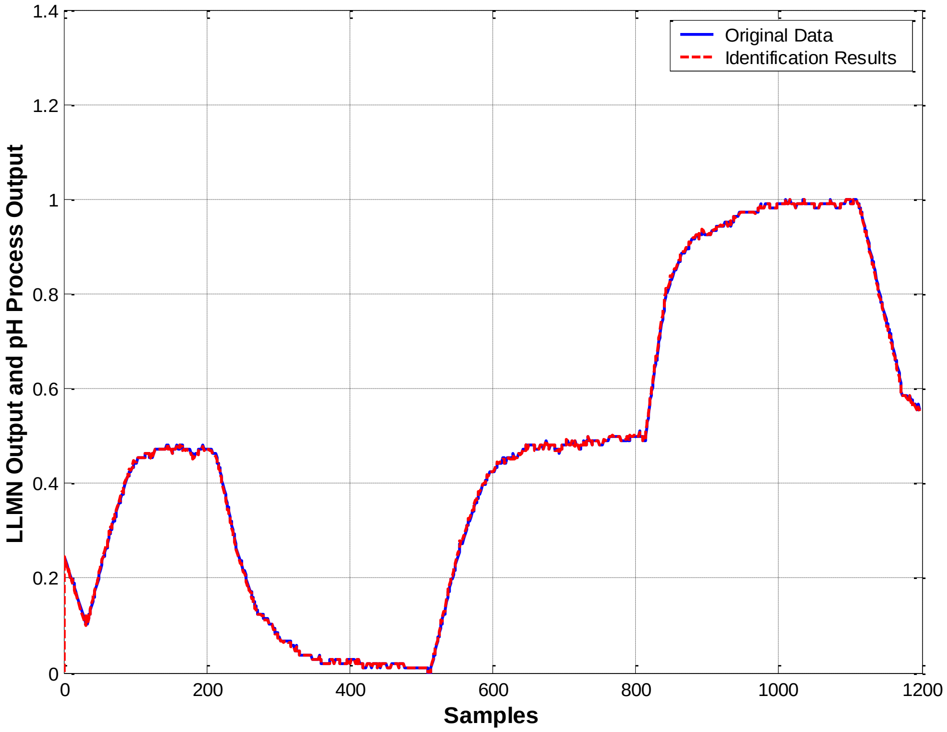


Figure 3-24 LLMN Identification results on training data for first order *pH* model. (Scaled data, sample time=10 sec)

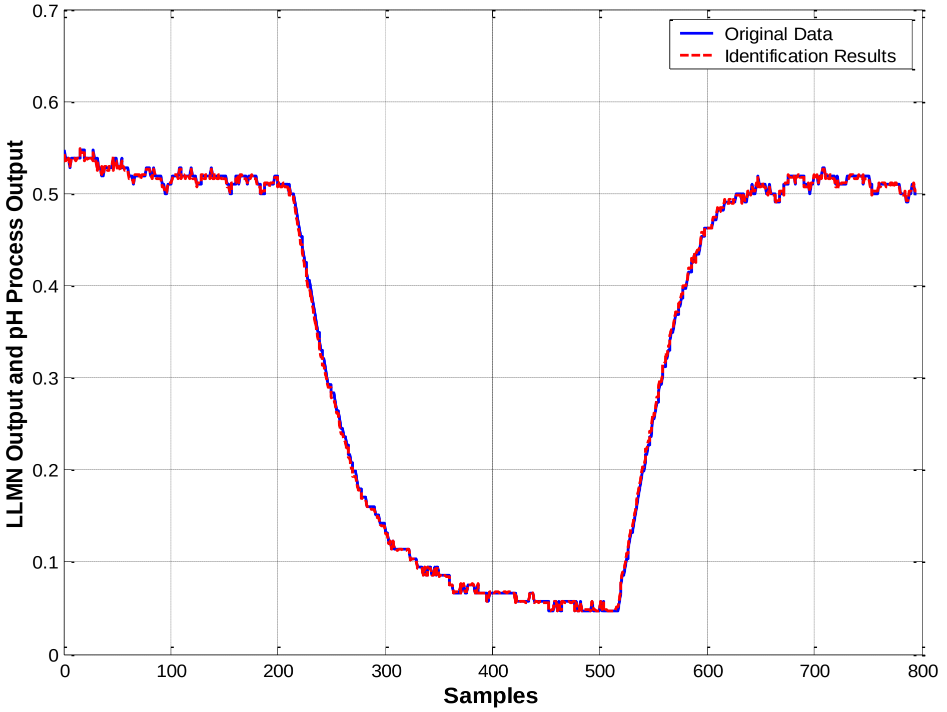


Figure 3-25 LLMN Identification results for test data for first order *pH* model. (Scaled data, sample time=10 sec)

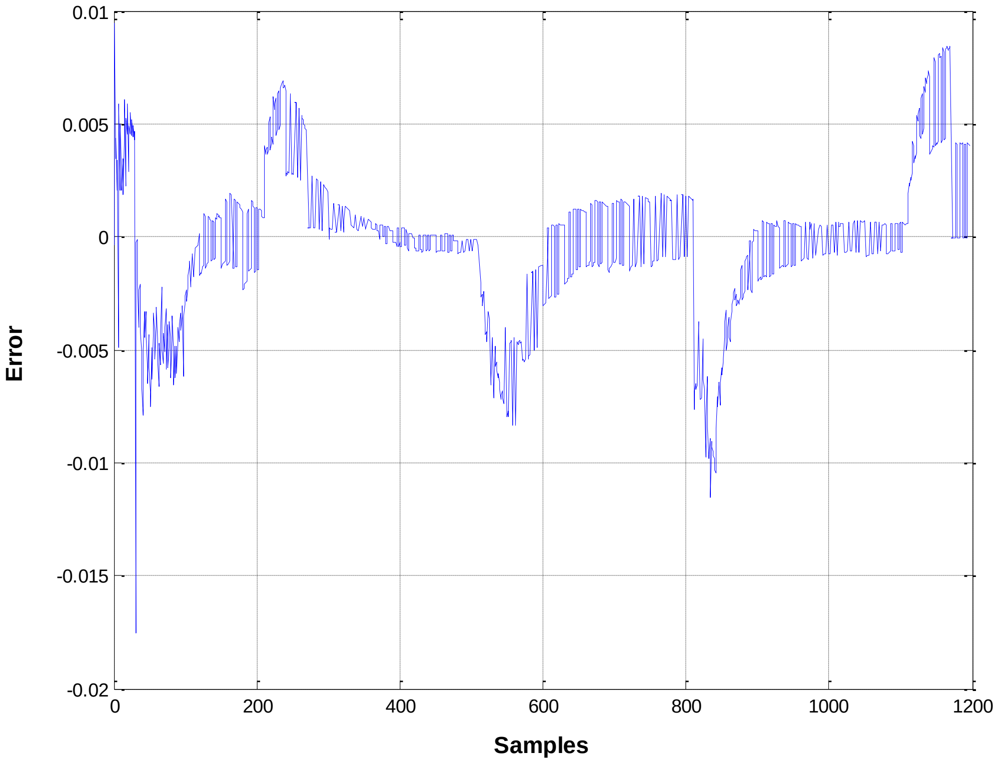


Figure 3-26 Error between LLMN output and model output on training data.

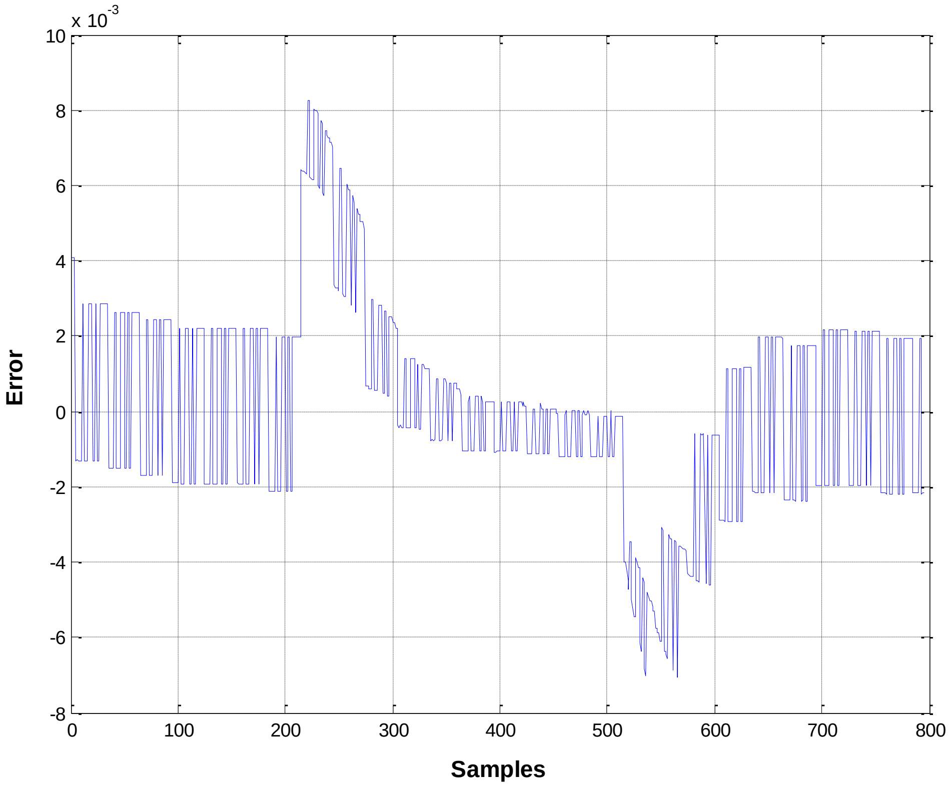


Figure 3-27 Error between LLMN output and model output on test data.

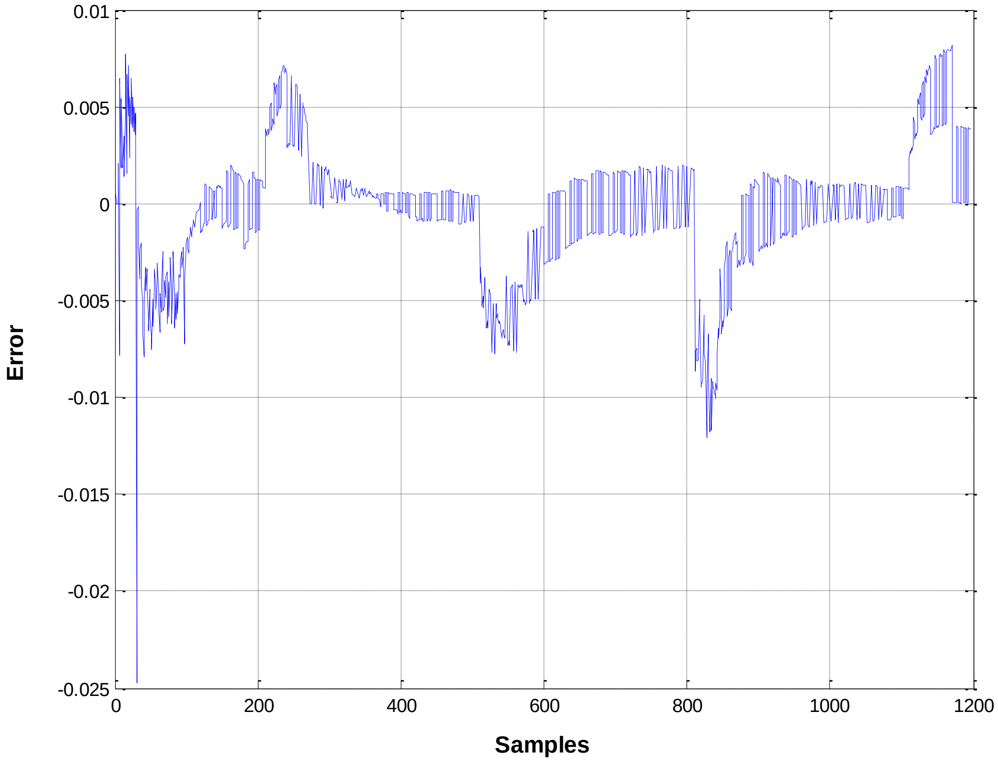


Figure 3-28 Error between LLMN with bias output and model output on training data.

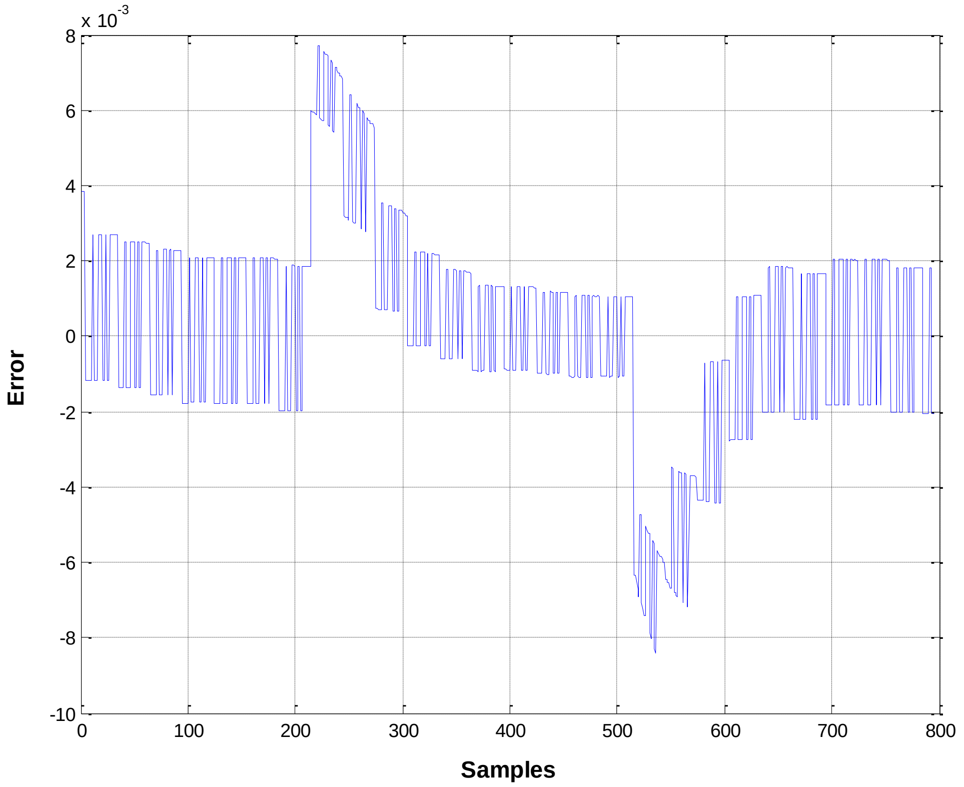


Figure 3-29 Error between LLMN with bias output and model output on test data.

3.4.2 Simulation of Local Linear Model Networks Independent Model

The idea behind the independent model is to test further the accuracy and validation of the identified local linear model network in the previous section. As an independent model, once the local linear model network is trained to update to weight and provided with input u (k), the network outputs $\hat{y}(k)$ are delayed and fed back to the network input. The network here can be used independently (Hunt *et al.*, 1992; Gomm *et al.*, 1997; Doherty *et al.*, 1997). The network input vector became as

$$x(k) = [\hat{y}(k-1) \dots \hat{y}(k-n)\ \ u(k-1) \dots u(k-n)]^T \tag{3.22}$$

The simulation of the LLMN independent model is used as a feedback network to test the system identification network, as detailed in section 3.4.1 and will be used for control simulation. The differences between independent model and one step ahead prediction is that greater errors are found between the outputs for independent response due to the process behaviour. The predicted output of the LLMN independent model for training and test data for network structure when n=1 with 4 local models and scaling factor $\alpha = 20$ are illustrated in Figures.3.30 and 3.31.

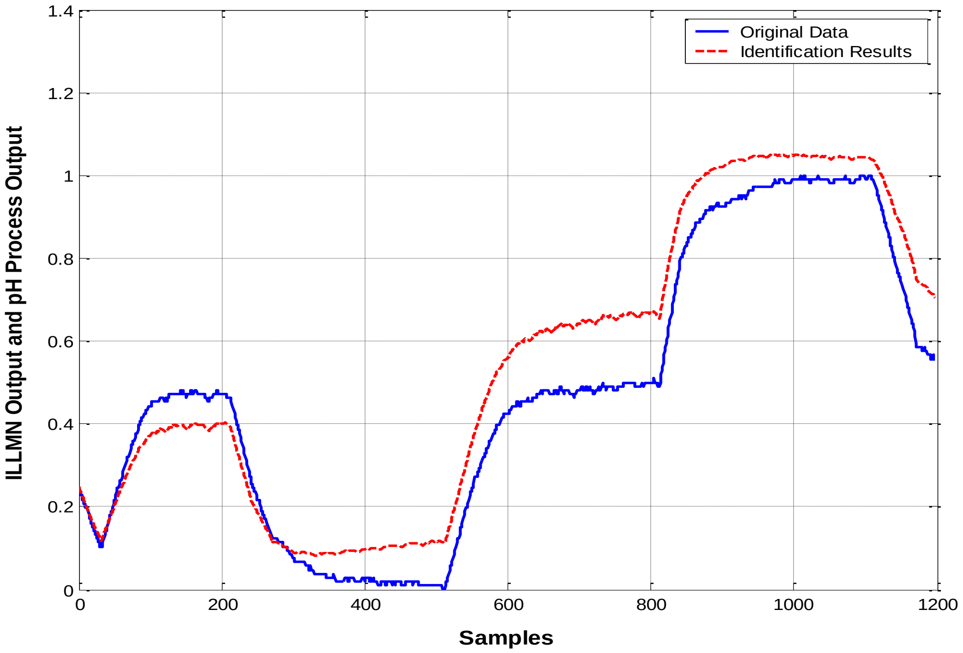


Figure 3-30 Independent LLMN results on training data for first order *pH* model. (Scaled data, sample time=10 sec)

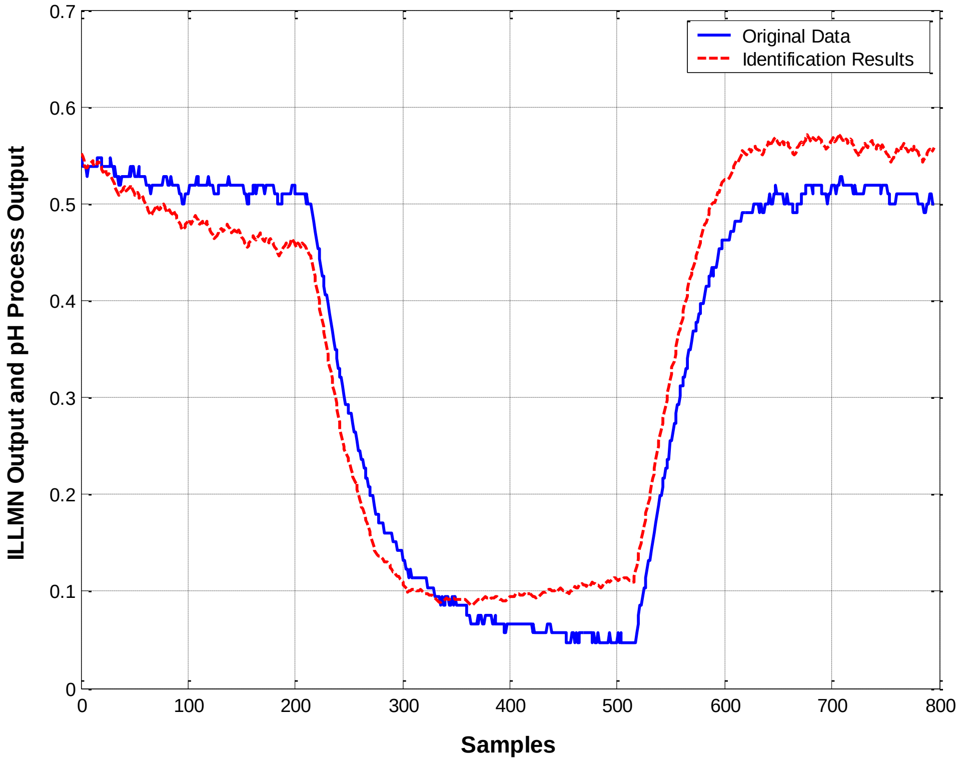


Figure 3-31 Independent LLMN results on test data first order *pH* model. (Scaled data, sample time=10 sec)

3.5 Investigation of Adaptive Neuro-Fuzzy Inference System

The adaptive neuro-fuzzy inference system ANFIS was first introduced by (Jang, 1993). ANFIS is working similarly to artificial neural networks. It has the capability to learn from data training and work as the basis from which to construct a set of fuzzy if-then rules with suitable membership functions to generate the stipulated input-output.

MATLAB tool box is used for this study, by using the input-output data set; a fuzzy inference system (FIS) was constructed by the ANFIS toolbox function. The membership function parameters are tuned using a back-propagation algorithm either alone or in combination with a least squares type of method. This allows fuzzy systems to learn from the data they are modelling (MathWorks, 2013; Talpur *et al.*, 2017).

3.5.1 ANFIS Training Procedure

This section presents system identification for single input and output using ANFIS modelling which can be found in the MATLAB toolbox. The model here is first order model for training and validation data and the results are presented in section 3.5.2. The system identification steps for training and test data are illustrated as follows:

1. Generate the training data which is called in MATLAB (trnData)
2. Select the number of membership functions which are (5)
3. Select the mfType to be 'gbellmf'
4. Use genfis1 to generate initial membership functions in_fis=genfis1(trnData,numMFs,mfType)
5. generate the anfis output using yan=evalfis(x,out_fis)

Where out_fis is calculated by anfis(trnData,in_fis,20) and x is the first order model.

These steps are achieved using MATLAB (see Appendix C) and the comparisons of MSE between RBFNN, LLMNs and ANFIS are presented in Table.3.3.

3.5.2 Real Data Identification Results

Figure 3.32 illustrates the ANFIS system identification results for the training data of first order *pH* model, while identification results for the test data is illustrated in figure 3.33. It can be seen from the figures that acceptable results are achieved using ANFIS method. The mean square error for training is 0.000049957and for test is 0.000068840.

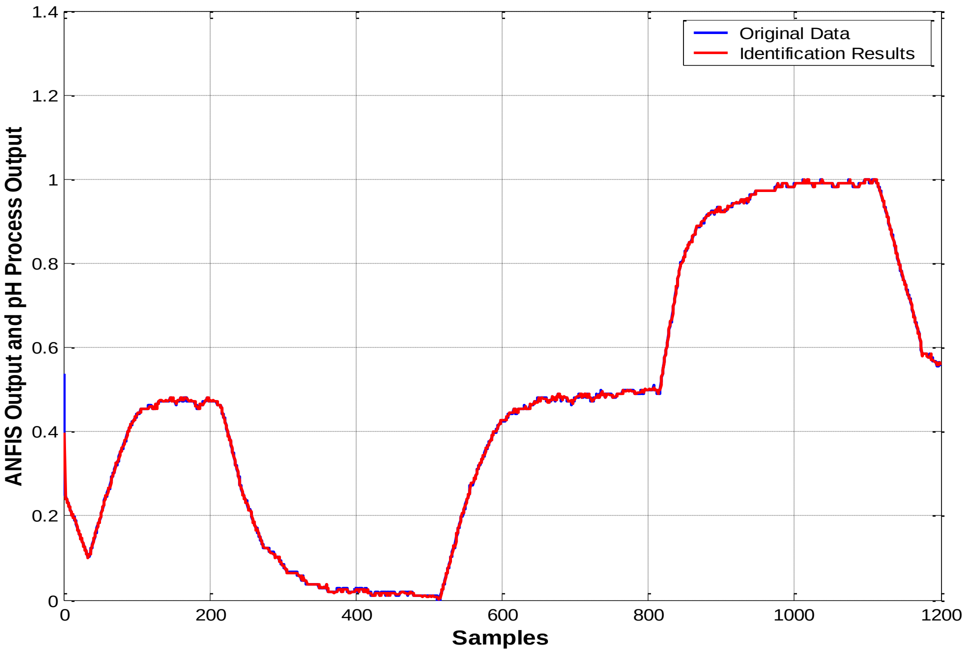


Figure 3-32 ANFIS identification results on training data for first order *pH* model. (Scaled data, sample time=10 sec)

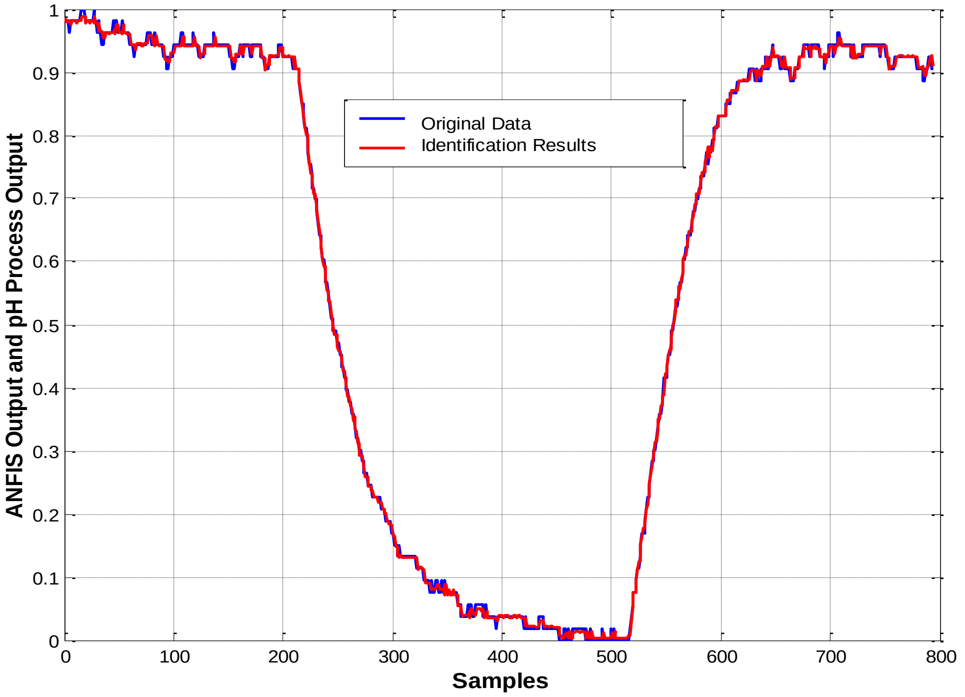


Figure 3-33 ANFIS identification results on test data for first order *pH* model. (Scaled data, sample time=10 sec)

3.6 Comparison of the LLMN, RBFNN and ANFIS

Applications of local linear model networks (LLMN), radial basis function networks (RBFNN) and Adaptive Neuro-Fuzzy Inference System (ANFIS) for system identification of real *pH* data have been investigated. The results reveal the ability of these networks to accurately represent the process when suitable choices and optimisations are made for various network parameters. However, local linear models networks (LLMN) have possibility and are powerful for approximating nonlinear dynamic systems. Their behaviour gave improved results with reduced MSE values compared with those of RBFNN and ANFIS for the identified *pH* system. The results of local linear model networks have been achieved with less number of parameters which is 16 compared with 24 parameters used in RBFNN. In other words, the LLMN structure is smaller than the RBFNN in this application. Table 3.3 lists the Comparison of MSE between RBFNN, LLMNs and ANFIS.

Table 3.3 Comparison of MSE between RBFNN, LLMNs and ANFIS

Method	MSE on train data	MSE on test data
RBFNN	0.000012341	0.0000077384
LLMNs unnormalised	0.000010414	0.0000075046
ANFIS	0.000049957	0.000068840
ILLMNs	0.0106	0.0022
LLMNs Normalised	0.000010344	0.00000746255

3.7 Development and Investigation of Control Based on Local Linear Model Networks for *pH* Process

3.7.1 Overview and Purpose

As section 3.4 explained, the system identification of the *pH* process using local linear model networks for the real *pH* data is achieved. This section describes the controller design for the *pH* chemical process. PI controller using the direct design method and local linear model controllers are applied to the identified process and are described in the following section.

3.7.2 Direct Design Control Procedure

The important of control nonlinear systems has become well known in most modern engineering systems, where controllers are required to have a good steady-state performance and satisfactory transient. In this section a direct control design will be presented to control a nonlinear *pH* process. The controller transfer function $G_c(z)$ for the process can be calculated using MATLAB from the desired closed loop transfer function $G_T(z)$ and this closed loop control system should be stable. The closed-loop transfer function $G_T(z)$ must have the same pole-zero deficit as the process $G_S(z)$, while the delay in $G_T(z)$ must be at least as long as that in $G_S(z)$. The direct design equation is presented in equation (3.23).

$$G_c(z) = \frac{1}{G_S(Z)} \times \frac{G_T(Z)}{1-G_T(Z)} \tag{3.23}$$

The block diagram of direct design feedback controller is illustrated in Figure.3.34 (Fadali and Visioli, 2012). In terms of when unstable pole-zero cancellation occurs, the system is input-output stable but not asymptotically stable. The reason for this is that the response is due to the initial conditions that are not affected by the zeros and are affected by the unstable poles, even if they cancel with a zero. Therefore, the closed loop control systems should be designed carefully to avoid unstable pole-zero cancellations. This indicates that the set of zeros of $G_T(z)$ must include all the zeros of $G_S(z)$ that are outside the unit circle. There is a common challenge that is facing the direct design method is the selection of a suitable closed-loop transfer function.

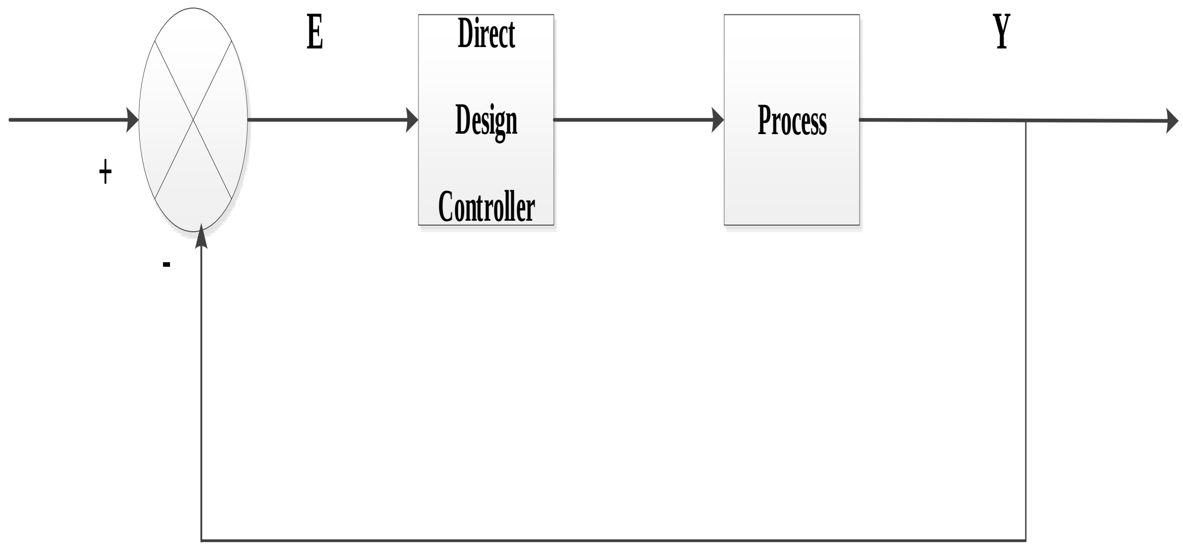


Figure 3-34 Direct Design Control Diagram for Nonlinear Process

3.7.3 PI Controller Design For First Order *pH* Model

Direct design controller procedure is used in this section to design PI controller for first order *pH* model. From a LLMN property of the transfer function $G_S(z)$ can be computed by the sum of multiplying the basis function for each local model with its weights to find out the transfer function parameters (a_1) and (b_1). The basis functions are evaluated at a parameter operating point to obtain the transfer function at that point. The identified first order LLMN of the *pH* process in section 3.4 has the form:

$$y(k) = [y(k-1) \dots u(k-1)] \qquad (3.24)$$

Where $y(k) = pH(k)$ and $(k) = f_b(k)$. This corresponds to a local first order difference equation,

$$y(k) + a_1 y(k-1) = b_1 u(k-1) \qquad (3.25)$$

The first order transfer function can then be obtained from

$$Y(z) + a_1 z^{-1} Y(z) = b_1 z^{-1} U(z) \qquad (3.26)$$

Giving

$$\frac{Y(z)}{U(z)} = \frac{b_1 z^{-1}}{1+a_1 z^{-1}} = G_S(z) \tag{3.27}$$

The transfer function parameters of the first order equation are calculated from the LLMN by multiplying the Gaussian functions with the weights for each operating point.

$$a_1 = \sum \emptyset_i \times w_i \tag{3.28}$$

$$b_1 = \sum \emptyset_i \times w_i \tag{3.29}$$

Here w_i are the LLMN weights which are calculated using recursive least squares (RLS) algorithm, and ϕ_i is the basis function for each local model.

To design the target of the desired closed loop transfer function $G_T(z)$ requires time constant to be determined for $5\tau = 100$ samples time then the transfer function can be designed when $A = e^{-\Delta t/\lambda} = 0.95$. λ here is the desired closed loop time constant. See equation (3.30). (Seborg *et al.*, 2004).

Since the identified *pH* model is first order, a first order desired closed loop transfer function $G_T(z)$ was also chosen:

$$G_T(z) = \frac{(1-A)z^{-1}}{1-Az^{-1}} \tag{3.30}$$

The above $G_T(z)$ has unity steady state gain to achieve zero steady state offset in the control. The pole A was chosen to give a desired closed loop time constant $\tau = 20\ Sec.$

By substituting equations (3.27) and (3.30) into equation, (3.23) then the controller transfer function became as:

$$\frac{U(z)}{E(z)} = \frac{g_0 z - g_1}{z-1} = G_c(z) \tag{3.31}$$

Where $E(z)$ is the error between reference set point and the output $e(k) = r(k) - y(k)$. $G_c(z)$ in equation (3.31) is a PI controller transfer function, then the PI controller applied to the process is calculated from the following difference equation:

$$u(k) = u(k-1) + g_0 e(k) + g_1 e(k-1) \tag{3.32}$$

3.7.3.1 Simulation Results

This section describes the possibility of designing a PI controller for the first order *pH* model. The controller was implemented for centre number 4, which is the scaled *pH* value of 0.4950 in the process. The first order model transfer function $G_S(z)$ was solved using equation (3.27) and the desired closed loop transfer function $G_T(z)$ was solved using equation (3.30) and became as:

$$G_S(z) = \frac{0.02046}{\mathrm{z} - 0.9829} \quad (3.33) \qquad\qquad G_T(z) = \frac{0.05}{\mathrm{z} - 0.95} \quad (3.34)$$

These equations are substituted into equation (3.23) to obtain the controller transfer function became as:

$$G_c(z) = \frac{2.444\,\mathrm{z} - 2.402}{\mathrm{z} - 1} \tag{3.35}$$

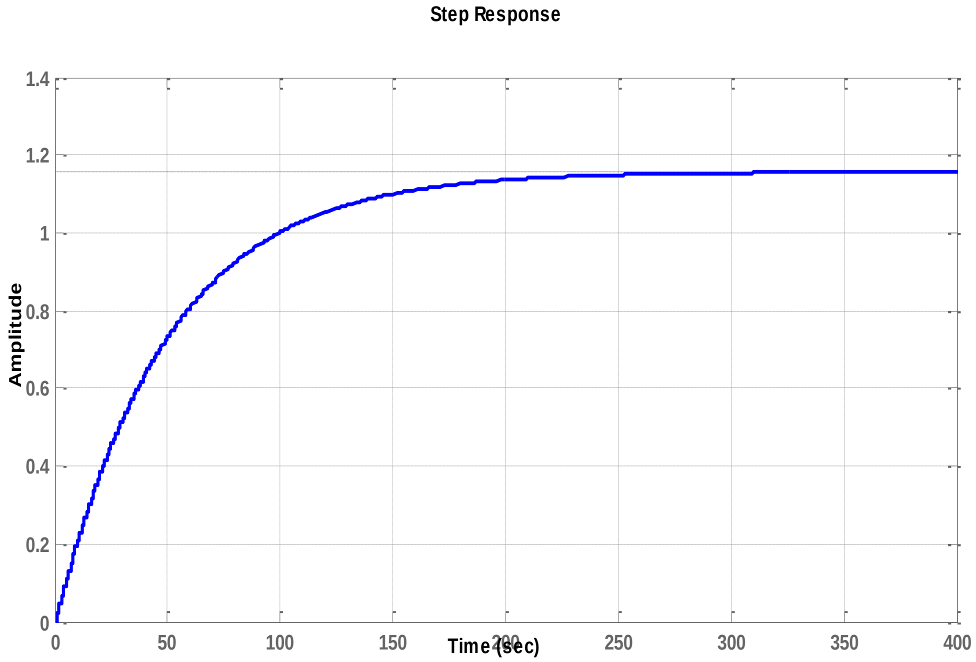


Figure 3-35 Step response of closed loop transfer function

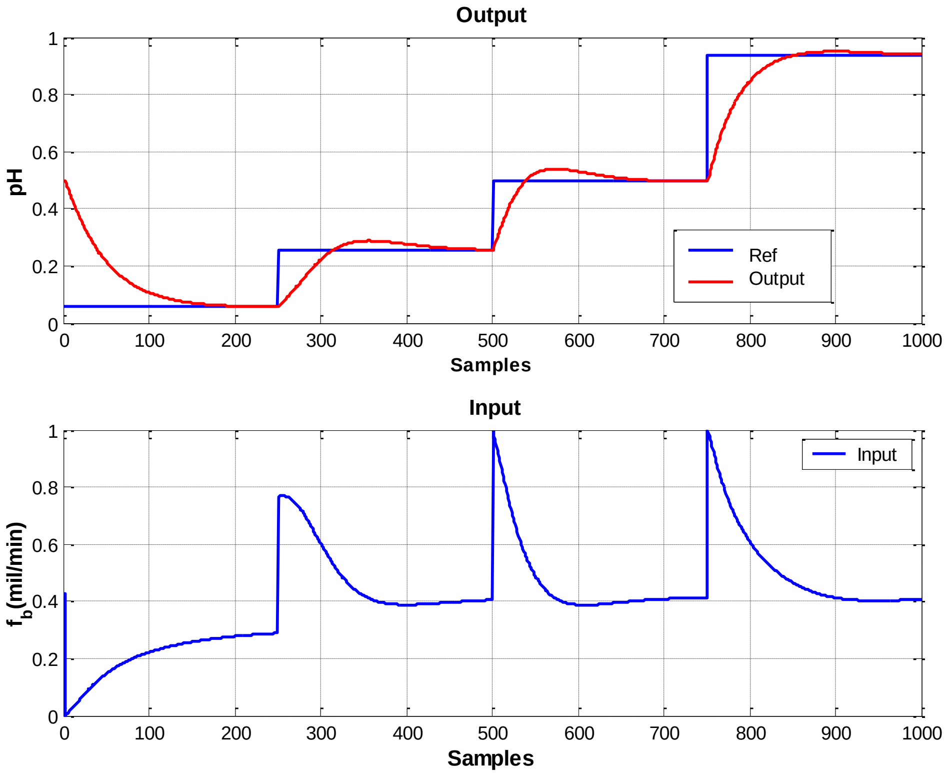


Figure 3-36 Process input and output of *pH* for single PI controller. (Scaled data, sample time=10 sec)

Figure 3.35 illustrates the step response of closed loop transfer function, while figure 3.36 presents the process input and output of *pH* for a single PI controller.

3.7.4 Local Linear Model Controllers Based on The First Order *pH* Model

The direct design procedure is used to design a controller for each local model identified at the network centres (4 in this case). These controllers are combined as below to form local linear model controllers. The Gaussian functions for the four local linear models on first order are combined to form a matrix ϕ as:

$$\phi = \begin{bmatrix} \phi_{11} & \phi_{21} & \phi_{31} & \phi_{41} \\ \phi_{12} & \phi_{22} & \phi_{32} & \phi_{42} \\ \phi_{13} & \phi_{23} & \phi_{33} & \phi_{43} \\ \phi_{14} & \phi_{24} & \phi_{34} & \phi_{44} \end{bmatrix} \tag{3.36}$$

The coefficients of the controller transfer functions for each local model are also combined to form the matrix G as:

$$G = \begin{bmatrix} g_{11} & g_{21} & 1 \\ g_{12} & g_{22} & 1 \\ g_{13} & g_{23} & 1 \\ g_{14} & g_{24} & 1 \end{bmatrix} \tag{3.37}$$

Then, by multiplying matrix (G) and (ϕ) for the 4 local models, then the controller weights (W) for the LLMN controller can be obtained from:

$$W = Inv(\emptyset) \times \mathrm{G} \tag{3.38}$$

The LLMN controller structure for the first order *pH* model is then as follows:

$$u(k) = [\emptyset\,]^T \times \begin{bmatrix} W \times \begin{bmatrix} e(k) \\ e(k-1) \\ u(k-1) \end{bmatrix} \end{bmatrix} \tag{3.39}$$

Where $\emptyset$ are the activation functions for the local linear model controller.

3.7.4.1 Simulation Results

The local model controller is designed after obtaining the local linear model controller weights(W). Subsequently, it is substituted into equation (3.39) to design the local model controller that is applied to the *pH* process. The matrix G and $\emptyset$ for the first order model became as follows:

$$\mathrm{G} = \begin{bmatrix} 7.960 & -7.714 & 1 \\ 3.649 & -3.627 & 1 \\ 3.288 & -3.208 & 1 \\ 2.444 & -2.402 & 1 \end{bmatrix} \tag{3.40}$$

$$\emptyset = \begin{bmatrix} 1 & 0.9902 & 0.9975 & 0.9917 \\ 0.9508 & 1 & 0.9699 & 0.9915 \\ 0.9975 & 0.9940 & 1 & 0.9975 \\ 0.9876 & 0.9975 & 0.9962 & 1 \end{bmatrix} \tag{3.41}$$

$$W = 1e^5 * \begin{bmatrix} 1.2784 & -1.2302 & 0.000032 \\ -1.0103 & 0.9721 & -0.000006 \\ -2.6670 & 2.5665 & -0.00006 \\ 2.4021 & -2.3115 & 0.00004 \end{bmatrix} \tag{3.42}$$

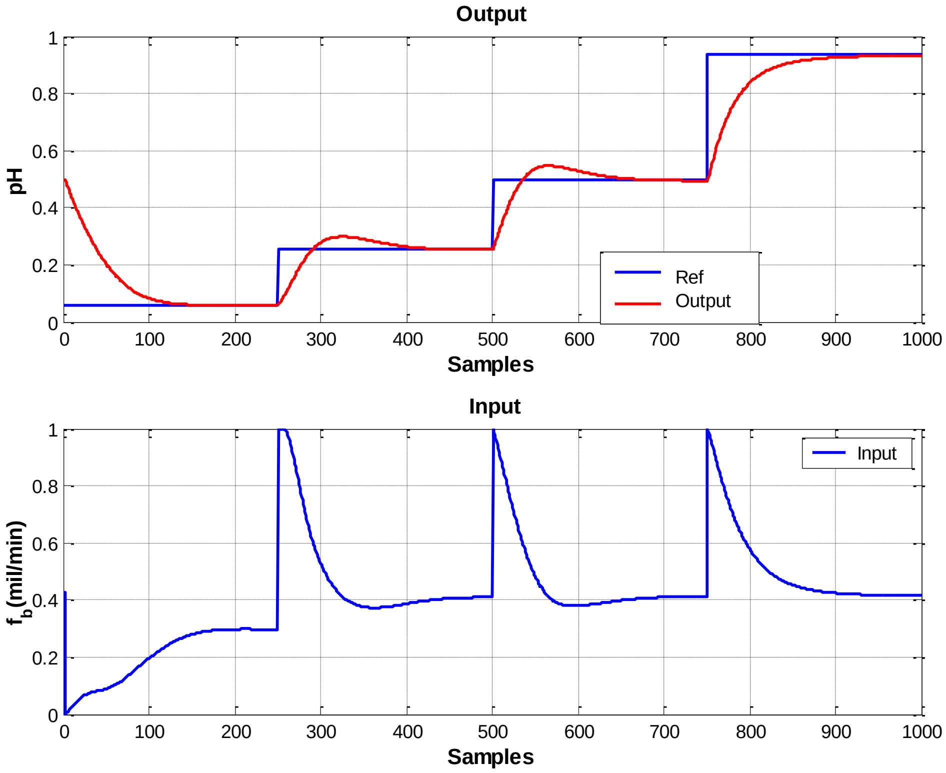


Figure 3-37 Process input and local linear model controller output of first *pH* model. (Scaled data, sample time=10 sec)

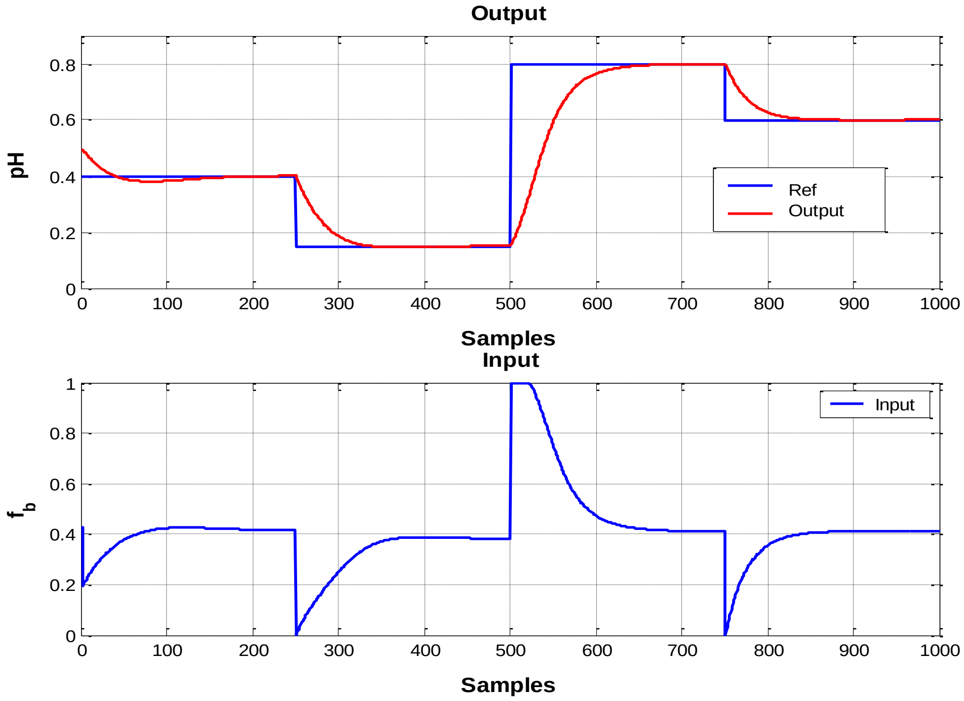


Figure 3-38 Local linear model controller output of *pH* for different points. (Scaled data, sample time=10 sec)

3.7.5 Discussion

In this section, local linear controllers were implemented using the direct design method, and the weights of the local linear controller were calculated by multiplying the inverse of basis function (3.40) for all local models with the control coefficients (3.41). Then, the error between the set point and the output is fed back to the process and the final controllers for all models are determined using equation (3.39) before being applied to the process input. The step response of closed loop direct design for first order model at centre number 4 is given in Figure. 3.35. Figure 3.36 illustrates the PI controller output of first order *pH* model and the input response. The controller output is tracking the reference set points and it can observed that it takes about 200 seconds to get the steady state for each set point, with a mean square error 0.00929. While Figure.3.37 illustrates the local linear model controller result, whereby the good match between the controller output (red) and set points (Blue) with mean square error 0.00838 is observed. Additionally, another simulation was also performed for different set points and also given a good result, as presented in Figure.3.38, with a mean square error of 0.01254. Table 3.4 illustrates the comparison of MSE between single PI and local linear controllers. Meanwhile, Table.3.5 details the comparison of MSE between a single PI controller and local linear controller for normalised Gaussian function.

Table 3.4 Comparison of MSE between single PI controller and local linear controller with un-normalised Gaussian function

Controller	MSE
Single PI controller	0.00929
Local linear controller for the 4 centres	0.00838
Local linear controller for different points	0.01254

Table 3.5 Comparison of MSE between single PI controller and local linear controller with normalised Gaussian function

Controller	MSE
Single PI controller	0.00954
Local linear controller for the 4 centres	0.00909
Local linear controller for different points	0.01260

3.8 Summary

This chapter has investigated system identification for a nonlinear *pH* process by using local linear model networks. This technique was compared with radial basis function networks and adaptive neuro-fuzzy inference system and from the results it is found that the application of local linear model networks gave smaller mean square error than radial basis function and ANFIS. After achieving system identification, a PI controller and a local linear model network controller were designed and implemented using direct design method to the process to control and maintain the *pH* output. The results support the use of local linear model networks to control *pH*.

Chapter 4 | Development of the Methods for Designing Local Linear Models for System Identification and Control of Real *pO_2* Data

4.1 Introduction

Dissolved oxygen concentration (*pO_2*) is a complete indicator to measure water quality and an significant indicator of water pollution. *pO_2* concentration was considered as the most important water quality variable such as in fish culture (Liu *et al.*, 2009).

The concentration of dissolved oxygen *pO_2* plays an important role in chemical industries such as water treatment. The amount of gaseous oxygen dissolved in the water is called dissolved oxygen and can get in the water from the atmosphere. There are some factors that could affect dissolved oxygen levels, such as water temperature. For example, in cold water oxygen could be dissolved more easily than in warmer water.

As dissolved oxygen in water is a critical issue, consequently, a good controller should be considered. This chapter describes the ability to use the local linear model networks technique (LLMNs) for system identification and control of the dissolved oxygen process and the use of direct design control. In this chapter the dissolved oxygen process contains input which is the flow rate of air (f_a) and the output is dissolved oxygen *pO_2* and the temperature will be selected to be as input in activation function and its procedure and results will be discussed later in this chapter.

There are many types of controller that could be used to control nonlinear process systems such as PI, PD and PID. In this chapter the direct design method is designed to control and maintain the *pO_2* in the process at a certain operation regime. The system identification and controller design are performed using MATLAB software.

4.2 Investigation of Local Linear Model Networks for System Identification of Real pO_2 Data

The comparisons between RBFNN, ANFIS and local linear model networks are discussed in chapter 3. This chapter describes system identification and control of the pO2 process using local linear model networks. Its training network is used to optimise the hidden layer centres and the widths in Gaussian functions and update weights, to minimise the model prediction error. The results are compared with a gradient descent algorithm which is used for system identification to update the width in the network. Then PI and local model controllers are designed and applied to the process and the results are discussed in this chapter.

4.2.1 Process Description

Figure 4.1 illustrates the dissolved oxygen process which contains airflow rate f_a $(0-100l/min)$ adjusted by a mass flow-meter connected to a compressing air network to regulate the percentage of the dissolved oxygen (pO_2) in the liquid. Moreover, the tank is equipped with an electric heating system to adjust the liquid temperature between $(35-50°\text{C})$ (Yu and Gomm, 2003). The data was collected with the process under closed loop PID control to drive the pO_2 over the operating range between (10–80%). It was found difficult to tune the PID controller to cover the nonlinear operating range but the data collected was considered suitable for examining nonlinear identification. The liquid level in the tank (and hence the liquid volume) is maintained at a constant value. It was noted from the experiment that since the dissolved oxygen responds quickly to changes in the airflow rate, especially at high temperature, therefore consideration should be given to the rise times for different variables. The liquid in the tank is stirred continuously to make sure the dissolved oxygen is consistent throughout the tank. A personal computer with analogue I/O is connected to the process to sample the measurements and issue the control output. The 2000 data points collected are presented in Figure 4.2. The process was affected by some problems such as the rise time for the temperature is very long due to the limitation of the heating power, whereas that of the dissolved oxygen is quite short. All these effects cause the process to be non-linear in both dynamic and static behaviour and this non-linearity of the process is demonstrated in the fixed parameter PID controller responses. Suitable sample time for all variables was selected to be 10 seconds. Figure 4.3 illustrates the scaled *pO2* and temperature data. The temperature is used as input in the Gaussian function network. The process input u and output y are described in equation 4.1:

$$u = [f_a], y = [pO_2] \tag{4.1}$$

Where f_a denotes air flow rate and *pO2* is the percentage of dissolved oxygen.

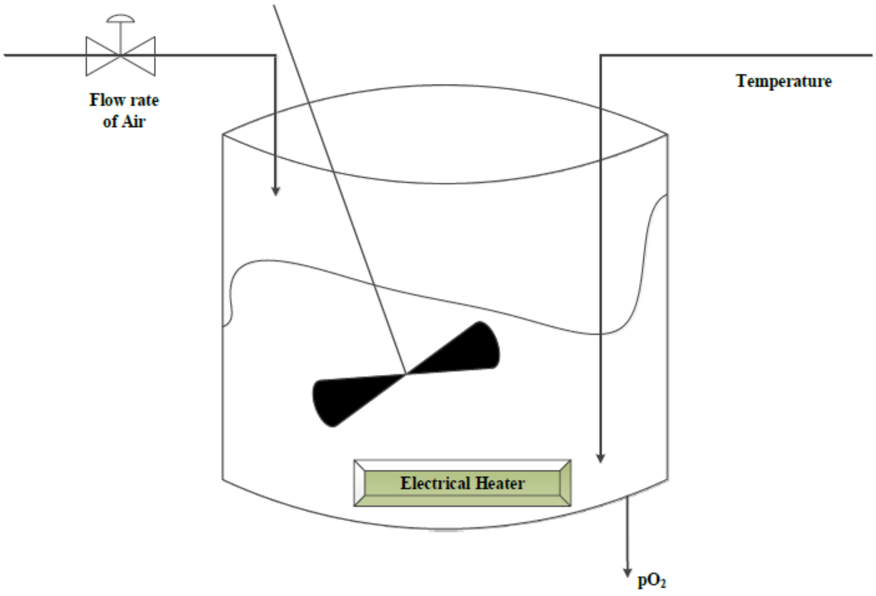


Figure 4-1 *pO2* process

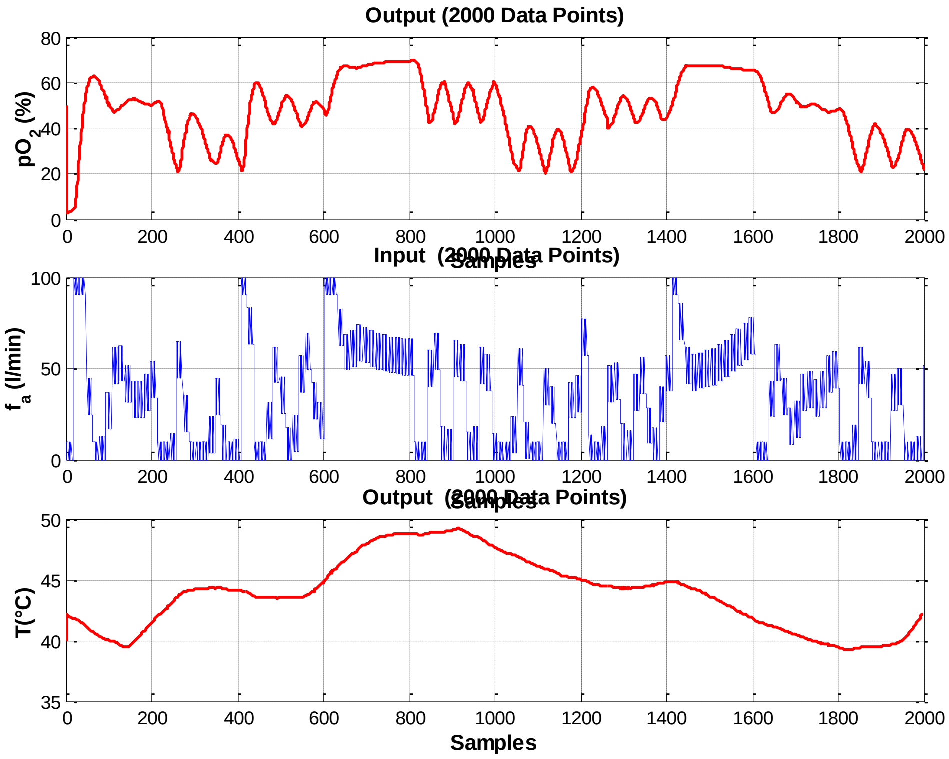


Figure 4-2 Measured real *pO*$_2$ and temperature data for network training and validation (sample time=10 sec)

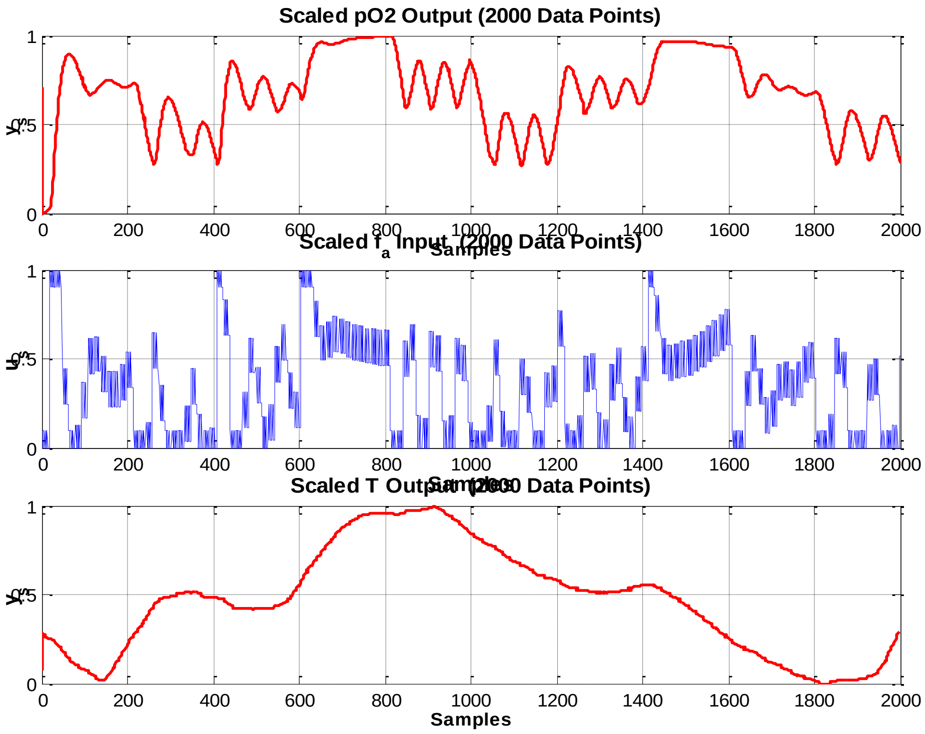


Figure 4-3 Scaled real *pO*$_2$ and temperature data for network training and validation (sample time=10 sec)

4.2.2 Local Linear Model Network Training and Evaluation for *pO₂* Data

System identification of complex nonlinear systems has been achieved using local linear model networks. The structure of dynamic local model networks, as illustrated in Figure. 3.20 in Chapter 3, will be used in this chapter to identify the dissolved oxygen process.

By using equation 3.19, where the input denotes the flow rate of air and the output is the dissolved oxygen, hence the model inputs becomes as:

$$x(k) = [pO_2(k-1)...pO_2(k-n)\ \ f_a(k-1)...f_a(k-n)\ ]^T \tag{4.2}$$

As the side effects of the normalisation of the validity function (equation 3.21) were discussed in Chapter 3, section (3.4.1), in this chapter un-normalised validity function ϕ_i will be used in the local linear model networks (equation 4.3). In chemical processes, the concentration of dissolved oxygen is dependent on the temperature behaviour to maintain the required oxygen level (Ogata, 2003). In this chapter $\tilde{x}(k) = T(k-1)$ was selected for validity function calculation.

$$\phi_i(k) = \phi_i(\tilde{x}(k)) = \exp\left(-\frac{\left\|\tilde{x}(k) - c_i\right\|^2}{(\alpha\sigma_i)^2}\right) \tag{4.3}$$

4.2.2.1 Real Data Identification Results and Discussion

In Chapter 3 equations, (3.15) and (3.16) were used to scale pH data. The same equations will be used in this chapter to scale the *pO₂* data. Then LLMNs are used to predict *pO₂* output which is scaled $y_S(k)$ for system identification. The 2000 real data samples were divided into two parts, the first 1200 data samples were used for training and the other 800 data samples for testing and validation.

The centres and widths in the validity functions of the LLMN were calculated using K-means algorithm and P-nearest neighbours method respectively. Initial model parameters are set as $w = 1.0 \times 10^{-6} \times U_{m\times 2n}$, $P(0) = 1.0 \times 10^{8} \times I_{m\times 2n}$, and λ is chosen to be 0.999. Then, recursive least square algorithm is used to update the weights for model output validation. The results presented in Figure. 4.4 indicate first order model, n = 1, M=3 and scaling factor $\alpha = 10$ achieves small mean square errors in one step ahead prediction on train is 0.000047930 and for test is 0.000026697. However, after the LLMN output is fed back to the input as an independent model, the results became better on first order independent model, M=3 and scaling factor $\alpha = 20$ the mean square errors for train and test data are 0.0155 and 0.0296 compared to the first order independent model, M=3 and scaling factor $\alpha = 10$. The mean square errors for training and test data are 0.0154 and 0.0308. Figures (4.5) and (4.6) illustrate the results for LLMN independent model results on training and test data first order *pO_2* model, M=3, α=20 respectively. Figures (4.7) and (4.8) illustrate the LLMN independent model results on training and test data first order *pO_2* model, M=3,$\alpha = 10$ respectively. Because of these results a single PI and local linear model controllers are designed on first *pO_2* model, M=3 and $\alpha = 20$.

Another experiment was done when n = 2, M=3 and scaling factor $\alpha = 20$ to investigate the behaviour of second order *pO_2* model and mean square errors for training and test data are 0.0003610 and 0.00015026 respectively. Meanwhile, the mean square errors in training and test data for second order independent LLMNs were 0.0941 and 0.0181 respectively. It can be observed that increasing the number of model order will increase the mean square error. While Figure.4.9 illustrates comparison between mean square error for LLMN training and test with a different number of local models and the same model order n=1 and width scaling $\alpha = 10$. The effect of varying the widths in the hidden nodes by multiplying by a scaling factor (σ×α) is presented in Figure.4.10 for networks with n=1 and M= 3. But as discussed before the best choice giving minimum prediction error when the widths in the local models by multiplying by a scaling factor is $\alpha = 20$ $(\sigma \times 20)$ for networks with n=1 and M=3. And more details and to clarify the selection of the model order, width scaling factor and number of local models for the local linear model network training for system identification, comparison for the mean square error can be found in table (4.1). The performance of the final trained network $(n = 1, \alpha = 20, \mathrm{M} = 3)$ on the training and test data are shown in Figures.4.11 and 4.13. Figures (4.12 and 4.14) show the errors for LLMN on training and test data.

Further experiments were conducted to investigate the behaviour of the first order model with M=3, , $\alpha = 10$ and when the activation function input depends on dissolved oxygen and the results reveal that the mean square errors for the training data and test data are 0.000052255 and 0.000023562 respectively.

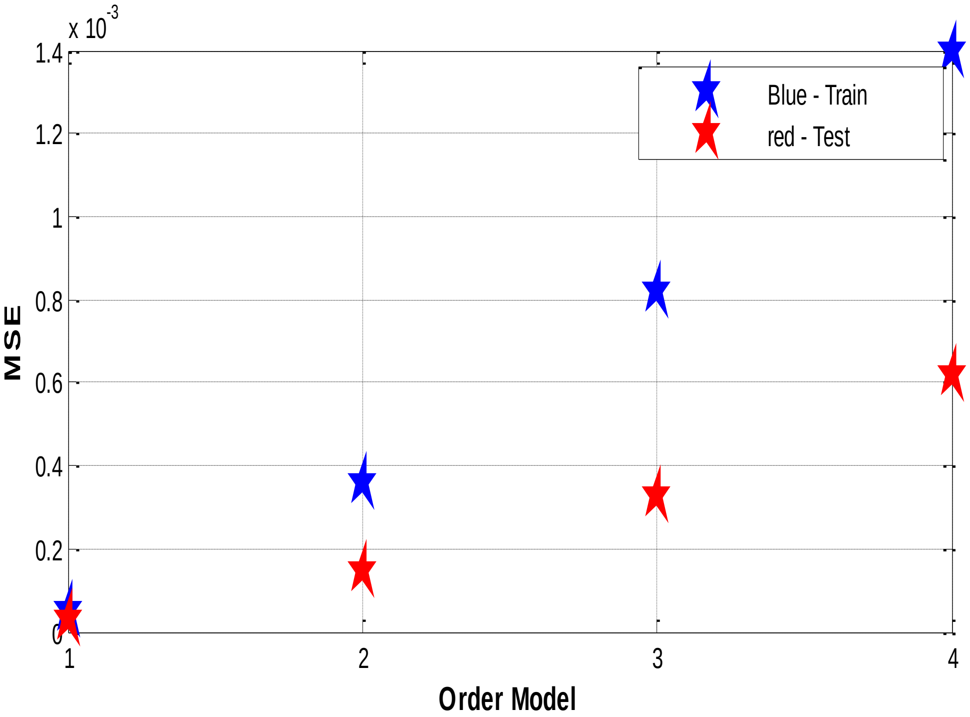


Figure 4-4 Comparison between MSE for LLMN on training and test data with different model order, M3, α=10

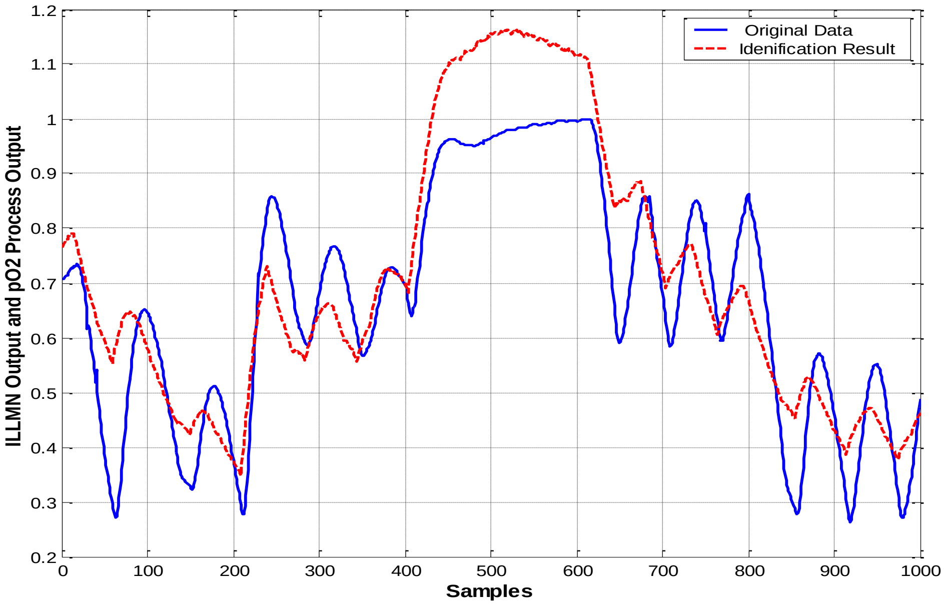


Figure 4-5 LLMN Independent model results for LLMN on training data first order *pO_2* model, M=3, α=20. (Scaled data, sample time=10 sec)

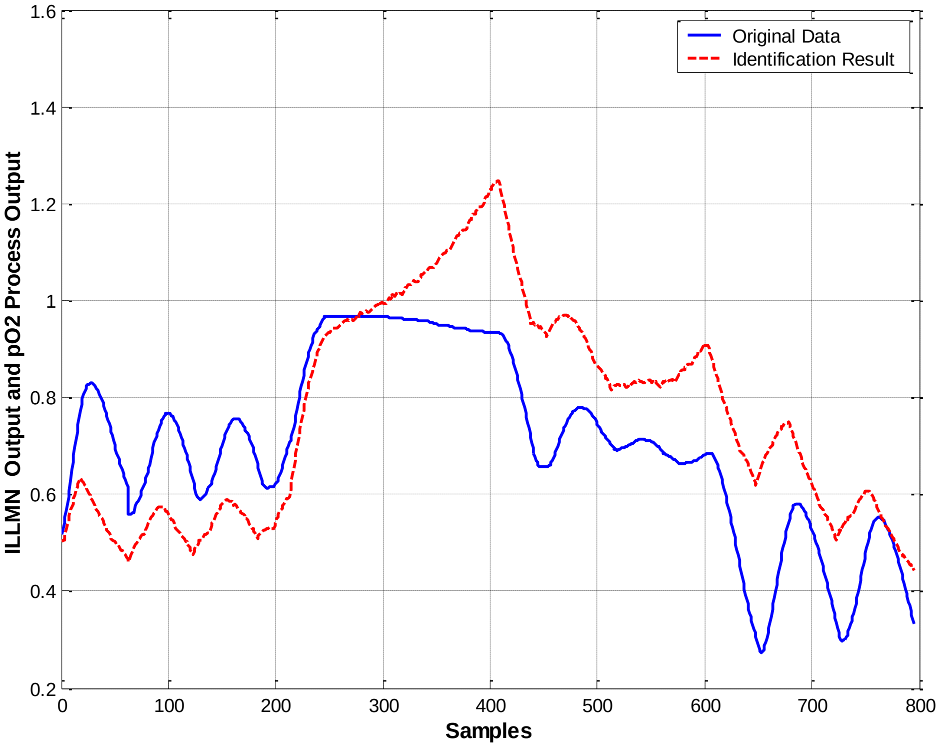


Figure 4-6 LLMN Independent model results for LLMN on test data first order *pO_2* model, M=3. α=20. (Scaled data, sample time=10 sec)

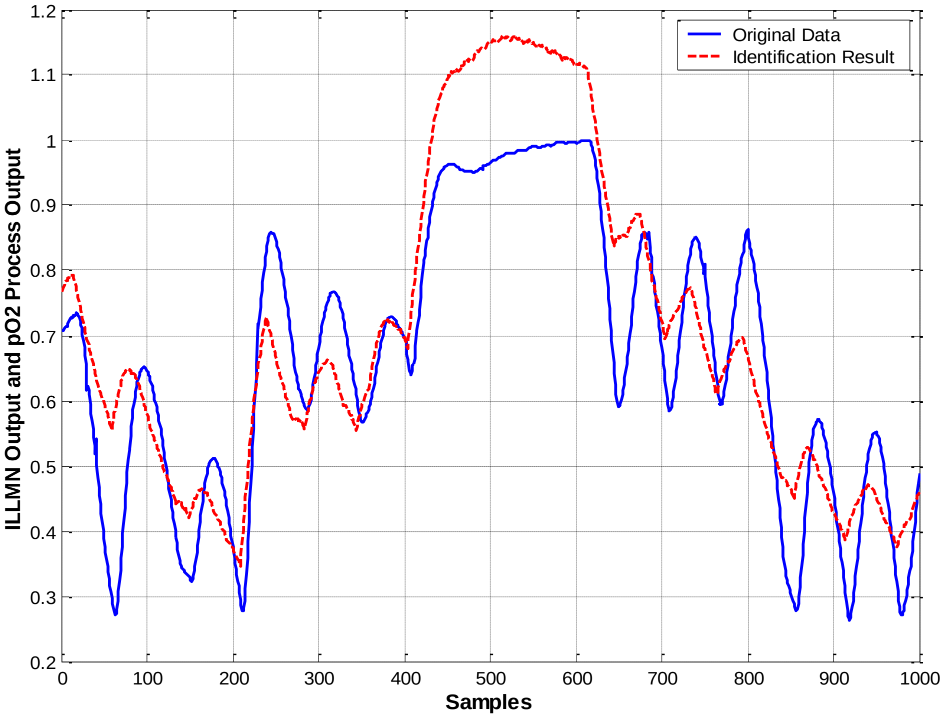


Figure 4-7 LLMN Independent model results for LLMN on training data first order *pO_2* model, M=3,α=10. (Scaled data, sample time=10 sec)

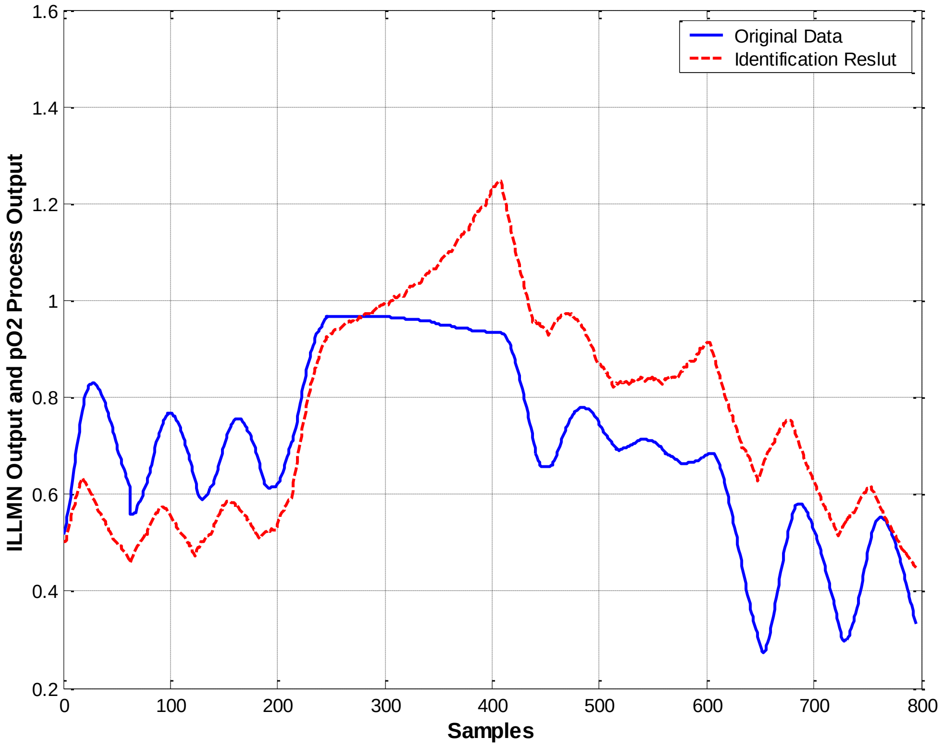


Figure 4-8 LLMN Independent model results for LLMN on testing data first order *pO_2* model, M=3, α=10. (Scaled data, sample time=10 sec)

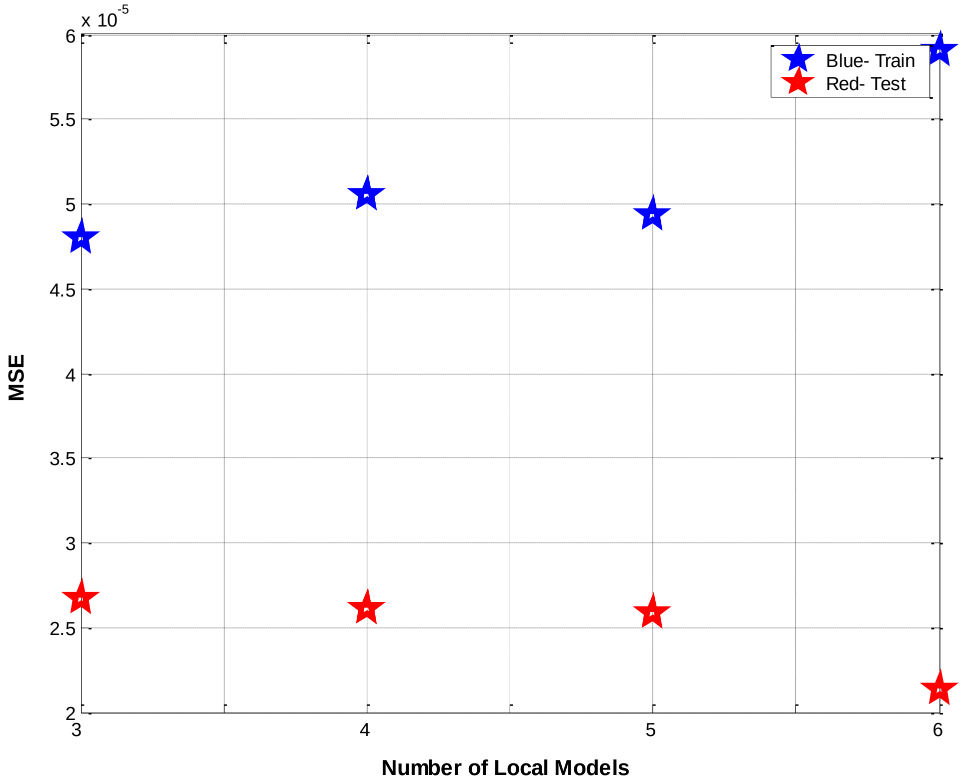


Figure 4-9 Comparison between MSE for LLMN on training and test data with different number of local models, n=1, α=10

Table 4.1 comparison for the mean square error for pO_2 with different model order, scaling factor and number of local models

Model order	MSE on train network $\sigma \times 20$	MSE on test network $\sigma \times 20$
First order M3	0.000056886	0.000026837
First order M4	0.00004992	0.00002677

Independent model	MSE on train network $\sigma \times 20$	MSE on test network $\sigma \times 20$
First order M3	0.0155	0.0296
First order M4	0.0136	0.0481
First order M5	0.0139	0.0436
First order M6	0.0132	0.0502

Model order	MSE on train network $\sigma \times 20$	MSE on test network $\sigma \times 20$
Second order M3	0.00036100	0.00015026
Second order M4	0.00036026	0.00013369

Independent model	MSE on train network $\sigma \times 20$	MSE on test network $\sigma \times 20$
Second order M3	0.0941	0.0181
Second order M4	0.0559	0.0419

Model order	MSE on train network $\sigma \times 10$	MSE on test network $\sigma \times 10$
First order M3	0.000047930	0.000026697
First order M4	0.000050550	0.000026182

Independent model	MSE on train network $\sigma \times 10$	MSE on test network $\sigma \times 10$
First order M3	0.0154	0.0308
First order M4	0.0132	0.0507
First order M5	0.0134	0.0502
First order M6	0.0118	0.1119

Model order	MSE on train network $\sigma \times 10$	MSE on test network $\sigma \times 10$
Second order M3	0.00035860	0.00014836
Second order M4	0.00036094	0.00013318

Independent model	MSE on train network $\sigma \times 10$	MSE on test network $\sigma \times 10$
Second order M3	0.0902	0.0196
Second order M4	0.0521	0.0454

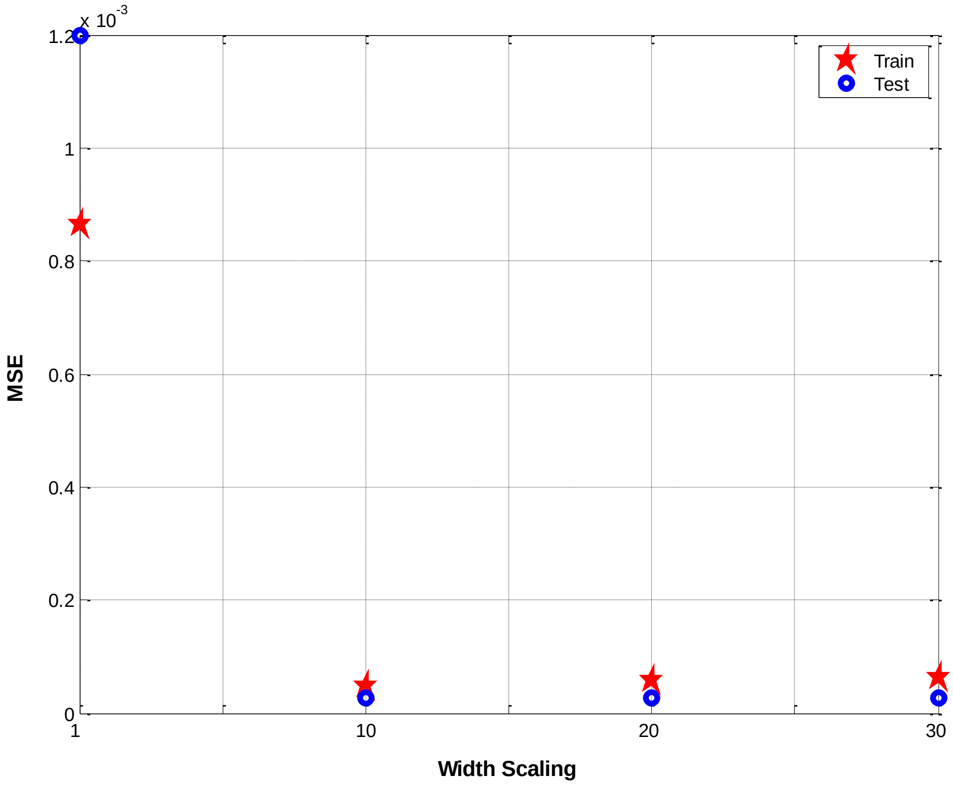


Figure 4-10 Comparison between MSE for LLMN on training and test data with different width scaling factor, n=1,M=3

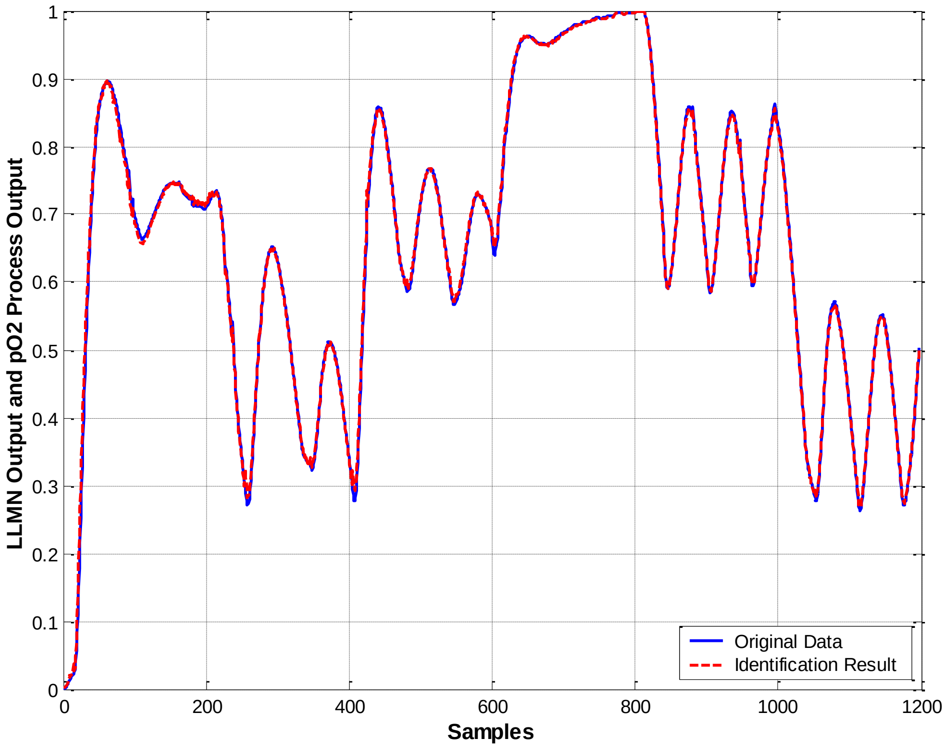


Figure 4-11 One step ahead prediction output for LLMN on training data for *p*O_2, n=1, M=3, α=20. (Scaled data, sample time=10 sec)

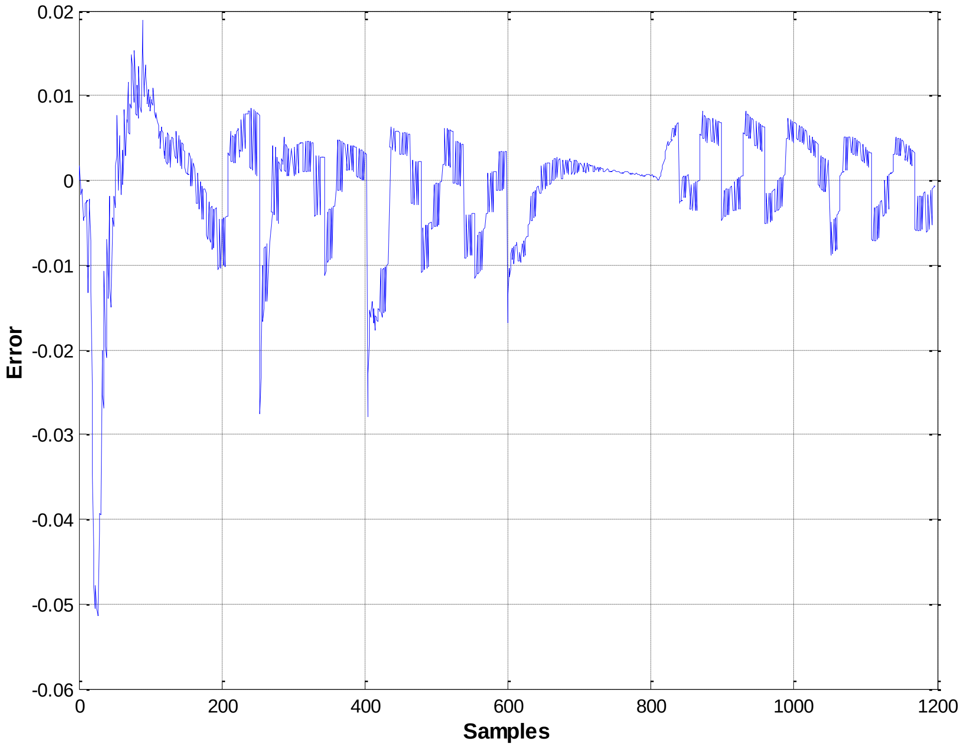


Figure 4-12 Error for LLMN on training data

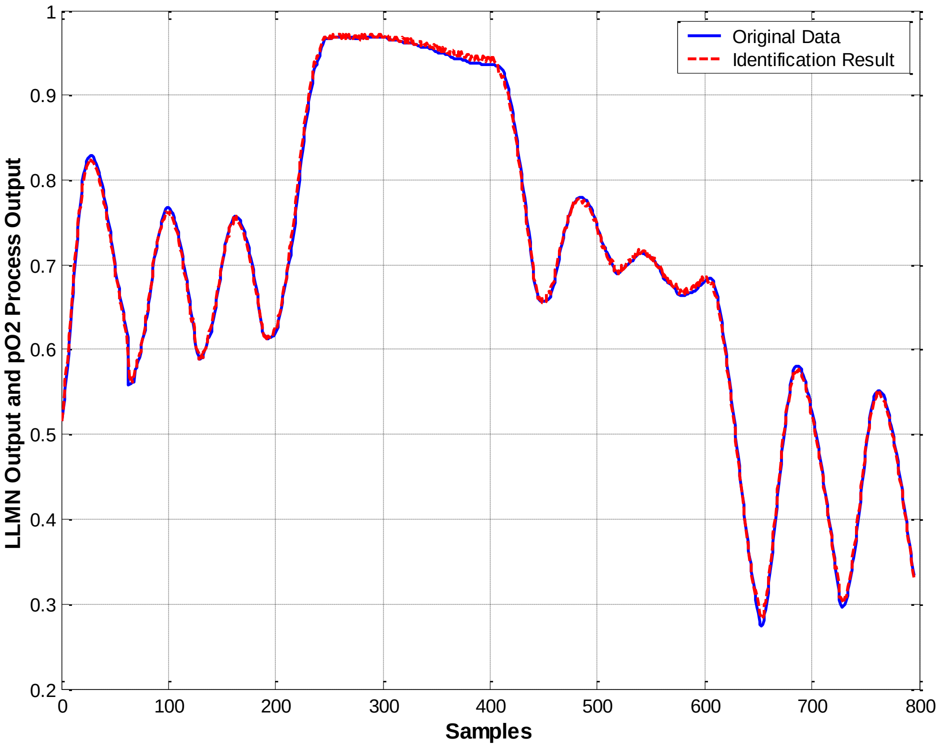


Figure 4-13 One step ahead prediction output for LLMN on test data for pO_2, n=1, M=3, α=20. (Scaled data, sample time=10 sec)

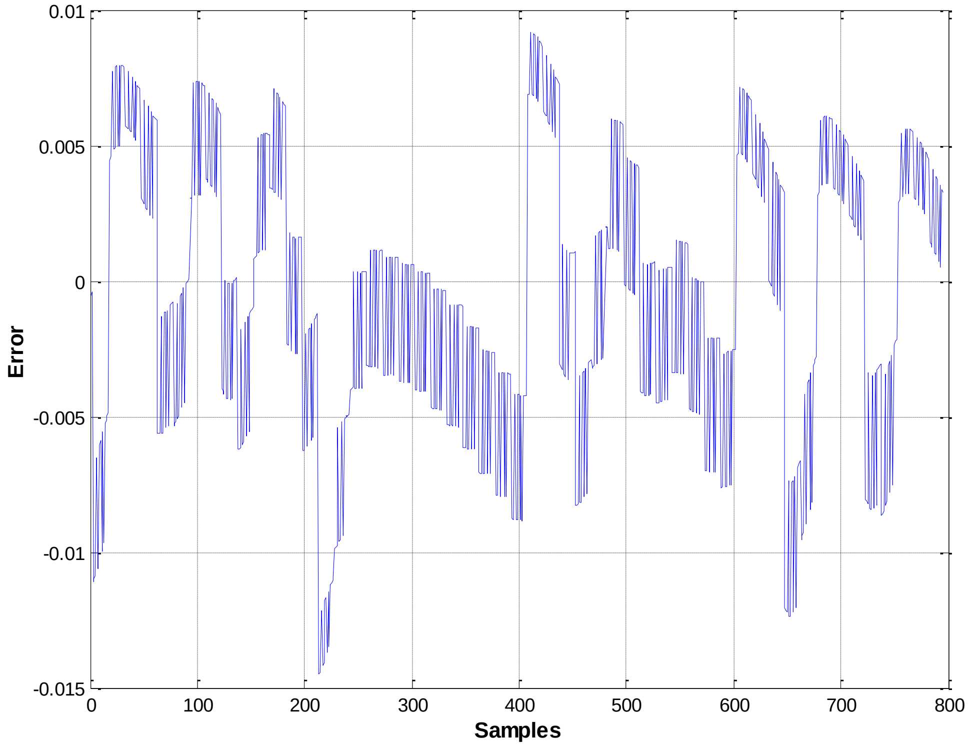


Figure 4-14 Error for LLMN on test data

4.3 Learning Algorithm Based on Gradient Descent

As the widths are one of the main factors that should be considered in the network, this section investigates several experiments as possible structures for testing the behaviour between basis function width adjustment and network performance. The gradient descent based learning algorithm tries to train the network to implement a desired input-output mapping by creating incremental updates of the weights of the network (Karayiannis, 1997; Karayiannis, 1999). It is suggested in this section a procedure for computing the Gaussian function widths. The purpose is to demonstrate the importance of optimising of Gaussian widths. Here, the weights are optimised via recursive least square algorithm, while the receptive field widths are updated using gradient descent algorithm. The width adjustment is explained by the following equations:

$$\frac{d\tilde{y}(k)}{d\alpha(k)} = \sum_{i=1}^{M}(\frac{d\varphi_i(k)}{d\alpha(k)} \times \tilde{y}_i(k)) \tag{4.4}$$

$$\alpha(k) = \alpha(k-1) - \eta \times \frac{d\tilde{y}(k)}{d\alpha(k)} \times E \tag{4.5a}$$

Where E is the error between the actual and prediction outputs:

$$E = y_s(k) - \hat{y}(k) \tag{4.5b}$$

Now activation function can be calculated as follows:

$$\varphi_i(k) = exp\left(-\alpha(k) * \frac{\|\tilde{x}(k)-C_i\|^2}{\sigma_i{}^2}\right) \tag{4.6}$$

4.3.1 Real Data Identification Results and Discussion with Gradient Descent

This section aims to demonstrate the importance of the optimisation of Gaussian widths. Model order is selected following the conducting of various experiments to be n = 1 in equation (4.2). Initial model parameters are set as $w = 1.0 \times 10^{-6} \times U_{m\times 2n}$, $P(0) = 1.0 \times 10^{8} \times I_{m\times 2n}$, and λ is chosen to be 0.999. The initial value for alpha is chosen after some initial experiments to be $\alpha\ (0) =\ 0.05$ and η is the learning rate and is set to be 0:05 in equation (4.5a) which gives acceptable results. Then the updated scaling factor $\alpha(k)$ is used in an activation function in equation (4.6). the mean square error when (n= 1; $\alpha =\ 0.05$; M=3) on training data is 0.000076027 and mean square error MSE for test data is 0.00026517. This is achieved by controlling the weight update during the learning process. The performance of the trained network (n= 1; $\alpha =\ 0.05$; M=3) on the training and test data are presented in Figures.4.15 and 4.16. Another experiment was also done when n= 2; $\alpha =\ 0.05$; M=3, the mean square error on training data is 0.00039774 and mean square error MSE for test data is 0.00027620 and are updated using equations (4.4) and (4.5). Figures.4.17 and 4.18 illustrate the variation of width activation during training with gradient on first and second order respectively. The results of the gradient descent method reveal it can provide the equivalent performance. However, the results presented in section (4.2.3) that manual selection in the LLMN gives smaller error compared with an automatic width updating, as it can be observed from the results in section 4.2.2.1 when n = 1, M=3 and scaling factor $\alpha = 10$ achieves small mean square errors on train data is 0.000047930 and for test is 0.000026697. And mean square error for training and test data when n = 2, M=3 and scaling factor $\alpha = 20$ data in manual selection are 0.00036100 and 0.00015026 respectively.

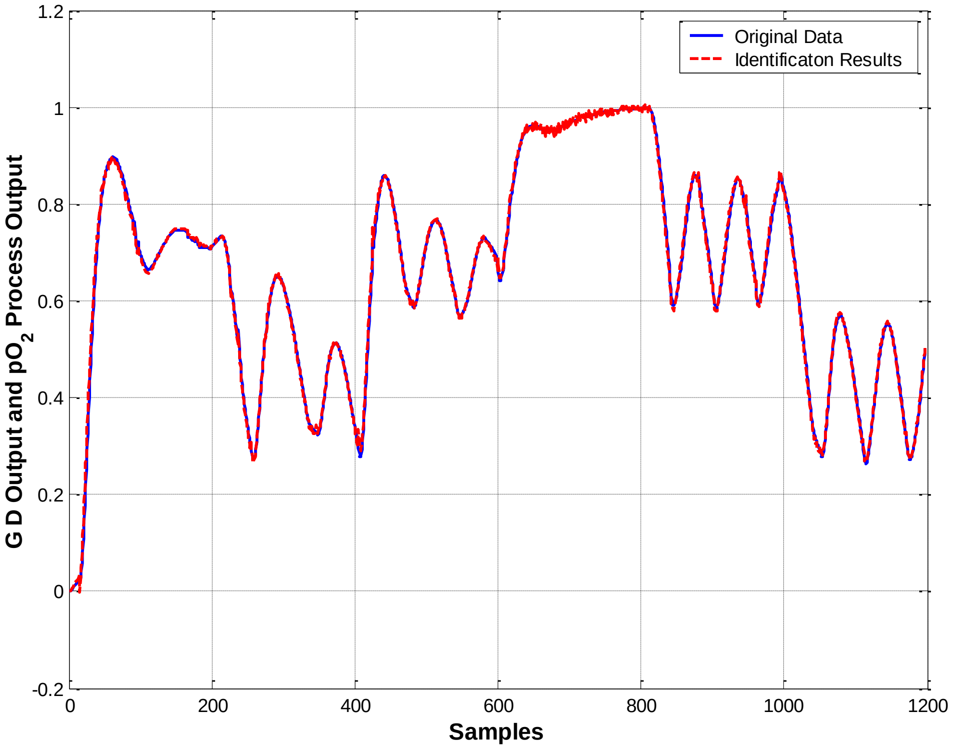


Figure 4-15 LLMN identification results on training data of pO_2 train data with gradient decent. (Scaled data, sample time=10 sec)

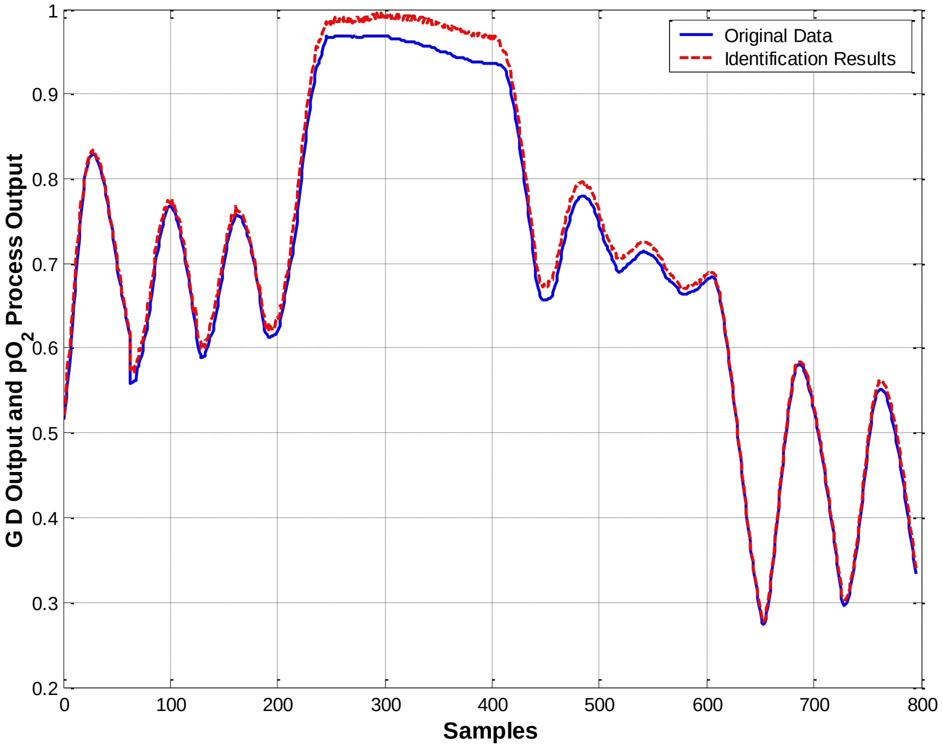


Figure 4-16 LLMN identification results on test data of *pO*₂ test with gradient decent. (Scaled data, sample time=10 sec)

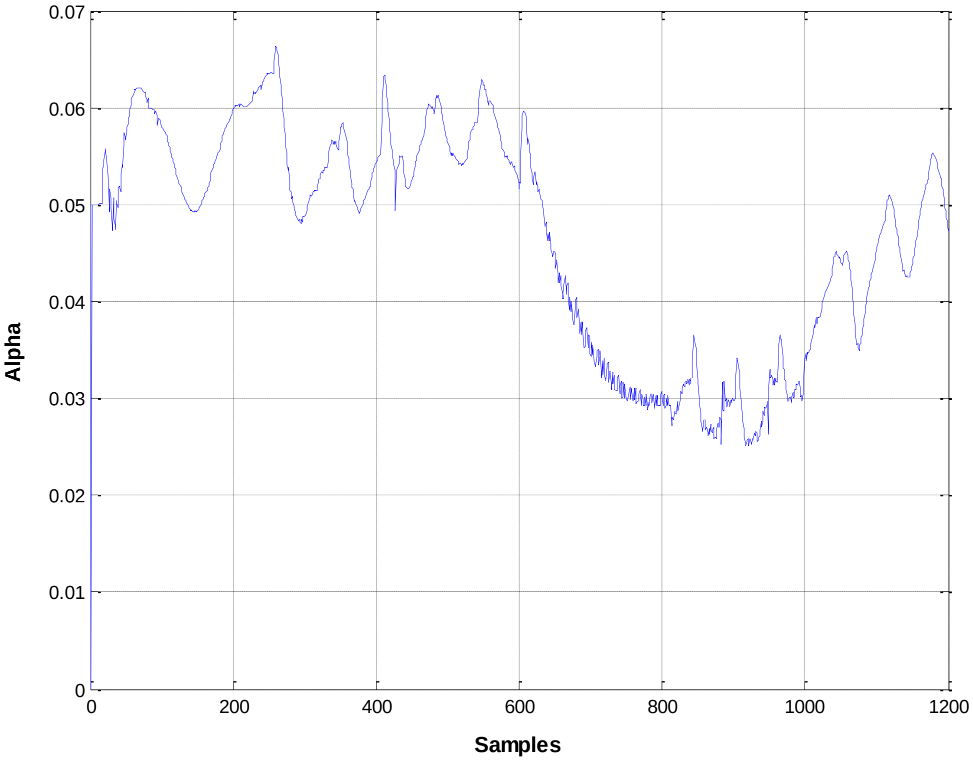


Figure 4-17 Variation of width activation during training with gradient decent for first order *pO*₂ model

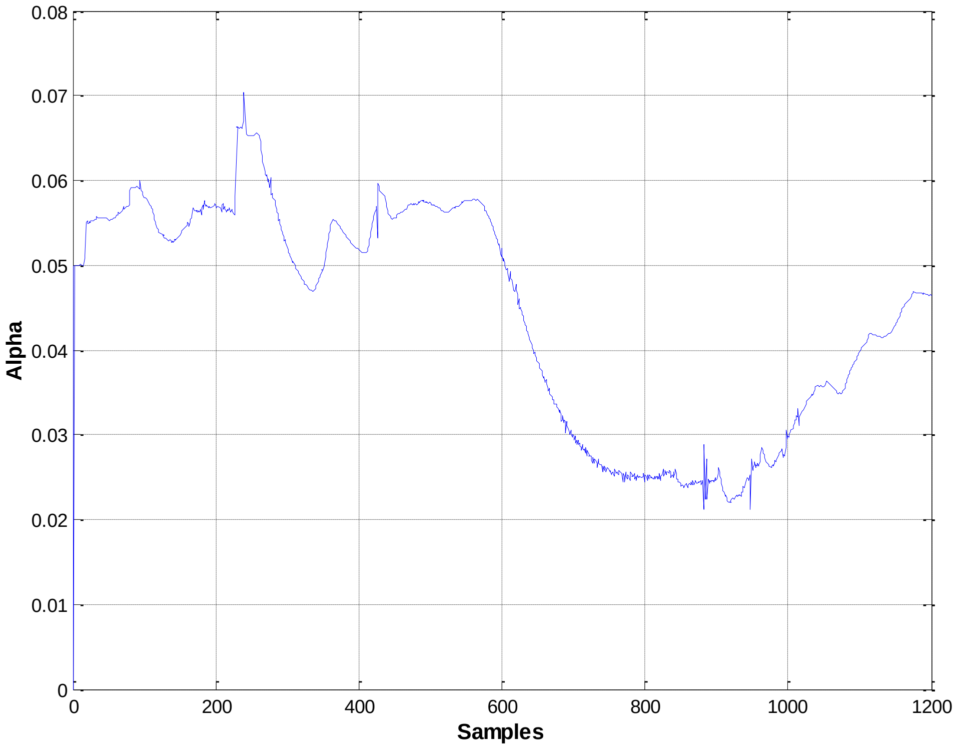


Figure 4-18 Variation of width activation during training with gradient decent for second order *pO_2* model

4.4 Development and Investigation of Control Based on Local Linear Model Networks for *pO_2* Process

4.4.1 Direct Design Control Procedure

Chapter 3 demonstrated that the equation (3.23) was used to design the controller transfer function. This chapter applies the same procedure to design PI and local linear model controllers for the dissolved oxygen *pO_2* process.

4.4.2 PI Controller Design For First Order *pO_2* Model

Direct design controller procedure is used in this section to design PI controller for first order *pO_2* Model. The local linear model network was trained for first order *pO_2* model with M=3. We considered calculating the first order transfer function to facilitate implementation of the transfer function controller around the operation point which is around second centre at 0.5010 in this case. The first order model transfer function $G_S(z)$was solved using equation (3.27) and $G_T(z)$ using equation (3.30) and became as:

$$G_S(z) = \frac{0.0215}{\mathrm{z} - 0.9903} \quad (4.7) \qquad G_T(z) = \frac{0.05}{\mathrm{z} - 0.95} \quad (4.8)$$

4.4.2.1 Simulation Results

In this section, the single PI controller for first order with M=3 was designed at second centre which is 0.5010, while parameters a_1 and b_1 are calculated using equations (3.28)and (3.29) and they are $0.9903, 0.0215$ then these parameters were used to calculate g_0 and g_1 for the controller transfer function and they are 2.318 and -2.296 respectively. The controller transfer function for first order model became as

$$G_c(z) = \frac{2.318z-2.296}{z-1} \quad (4.9)$$

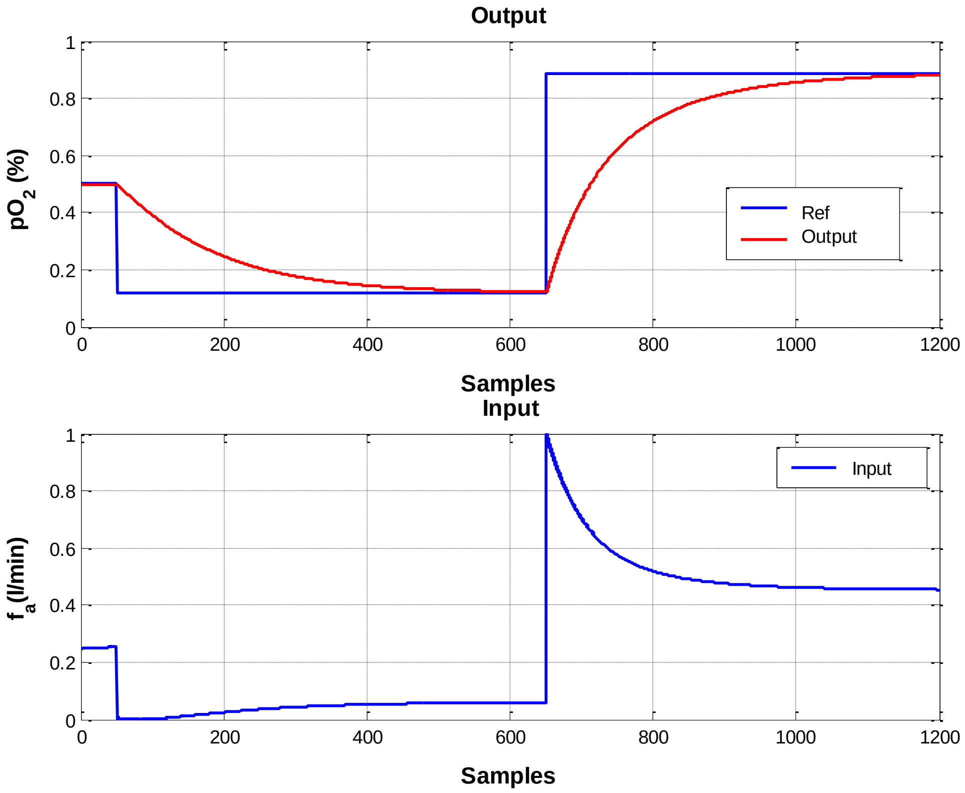


Figure 4-19 Single PI controller for first order pO_2 model around second centre. M=3, scaling factor=20. (Scaled data, sample time=10 sec)

4.4.3 Local Linear Model Controllers Based On First Order *pO₂* model

The direct design procedure is repeated to design controllers for each of the local models identified at the network centres (three in this case). These controllers are combined to form the LLMN controller structure. Where the activation function for the local linear model networks were combined using equation (4.10) and the control coefficients were combined using equation (4.11) then the LLMN controller weights can be calculated using equation (3.38). The final controller equation was calculated using equation (3.39).

$$\emptyset = \begin{bmatrix} \emptyset_{11} & \emptyset_{21} & \emptyset_{31} \\ \emptyset_{12} & \emptyset_{22} & \emptyset_{32} \\ \emptyset_{13} & \emptyset_{23} & \emptyset_{33} \end{bmatrix} \qquad (4.10) \qquad \emptyset = \begin{bmatrix} 1 & 0.9975 & 0.9901 \\ 0.9974 & 1 & 0.9975 \\ 0.9899 & 0.9975 & 1 \end{bmatrix}$$

$$G = \begin{bmatrix} g_{11} & g_{21} & 1 \\ g_{12} & g_{22} & 1 \\ g_{13} & g_{23} & 1 \end{bmatrix} \qquad (4.11) \qquad G = \begin{bmatrix} 2.559 & -2.533 & 1 \\ 2.317 & -2.94 & 1 \\ 2.08 & -2.59 & 1 \end{bmatrix}$$

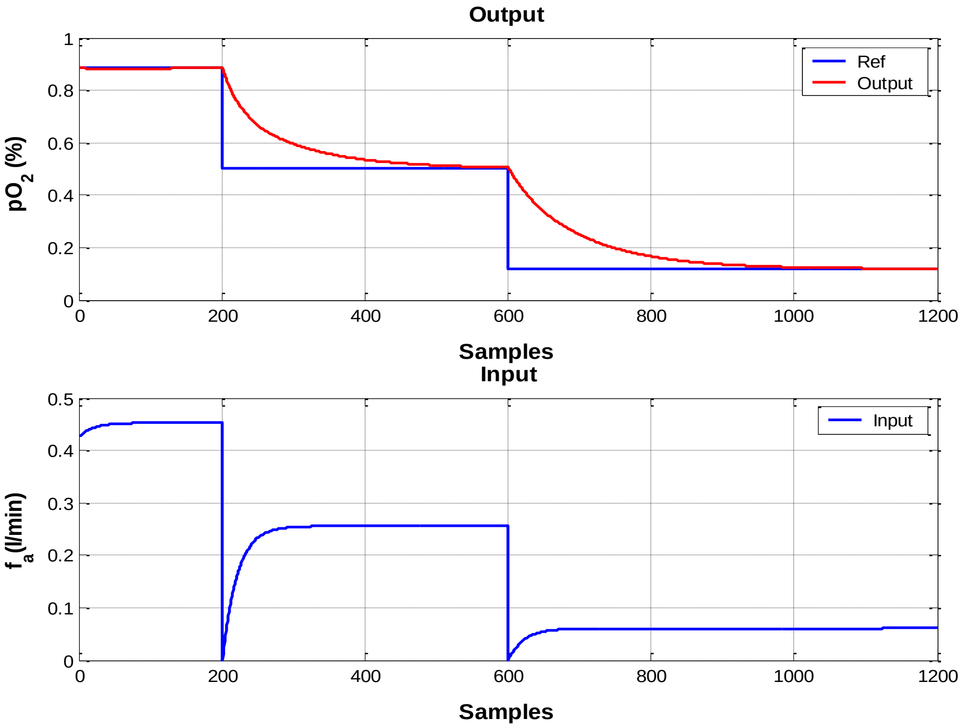


Figure 4-20 LLMN controller for first order *pO₂* model. M=3, scaling factor=20. (Scaled data, sample time=10 sec)

4.4.4 Discussion

Figure 4.19 illustrates the PI controller output response at the second centre, which is 0.5010. The output tracks smoothly with the reference set point changes at (0.5) and remains stable for approximately 50 seconds before dropping to reach the second set point around 0.11. However, it takes time to reach steady state before beginning to rise smoothly to reach the third operation point, which is first centre. These results prove that the first order model can lead to system stability.

While Figure 4.20 illustrates that the output response for the local model controllers and the input, the output response tracking the reference set point around 0.88 for about 200 seconds then dropped down to get steady state at second point which is second centre after approximately 300 seconds. Finally the output response get steady state at the third set point after about 500 seconds.

4.4.5 PI Controller Design For Second Order *pO₂* Model

In this section the possibility of designing a PI controller using a second order model is described (Seborg *et al.*, 2004). The identification order LLMN of the *pO₂* process in section 4.2 has the form:

$$x(k) = [y(k-1) \ldots .\, y(k-n) \ldots ..\, u(k-1) \ldots u(k-n)]^T \tag{4.12}$$

Where $y(k)$=pO_2 , $u(k)$=f_a and n is the number of order model which is here second order. Consider general second order difference equation,

$$a_0 y(k) + a_1(k-1) + \cdots + a_m y(k-m) = b_0 u(k) + b_1 u(k-1) \ldots b_n u(k-n) \tag{4.13}$$

Taking the z transform of both sides of equation (4.13) gives

$$a_0 Y(z) + a_1 z^{-1} Y(z) + \cdots + a_m z^{-m} Y(z) = b_0 U(z) + b_1 z^{-1} U(z) + \cdots + b_n z^{-n} U(z) \tag{4.14}$$

By rearranging equation (4.14) gives the transfer function form:

$$\frac{Y(z)}{U(z)} = \frac{b_0+b_1z^{-1}+\cdots+b_nz^{-n}}{a_0+a_1z^{-1}+\cdots+a_nz^{-n}} = G_S(z) \tag{4.15}$$

Then controller input for second order difference equation for the controller became as:

$$u(k) = -f_1 \times u(k-1) - f_2 \times u(k-2) + g_0 \times e(k) + g_1 \times e(k-1) + g_2 \times e(k-2) \tag{4.16}$$

The controller was implemented for first centre, which is 0.8871 in the process. The second order model transfer function $G_S(z)$was solved using equation (4.17) and $G_T(z)$ using equation (4.18) and became as:

$$G_S(z) = \frac{b_1-b_2z^{-1}}{1-a_1-a_2z^{-1}} = \frac{0.002959\,z+0.002306}{z^2-1.855\,z+0.8573} \tag{4.17}$$

$$G_T(z) = \frac{(1-A)z^{-1}}{1-Az^{-1}} \tag{4.18}$$

Where A in equation (4.18) is set to be 0.95 and then the controller transfer function for the first centre became as

$$G_C(z) = \frac{16.88z^2-31.31z+14.47}{z^2-0.2217z-0.7783} \tag{4.19}$$

It was noted that from the controller transfer function in equation (4.19) that it has pole out the unite circle at $z = -0.7783$. This leads to an oscillation in the input and leads to unstable control. This problem was solved by zero cancellation using equation (4.20) and then the final controller transfer function became as equation (4.21).

$$G_T(z) = \frac{(z+0.7783)\times G_T(z)}{1.7783z} \quad (4.20) \quad G_C(z) = \frac{9.514z^2-17.65z+8.158}{z^2-0.9781z-0.02188} \tag{4.21}$$

4.4.5.1 Simulation Results

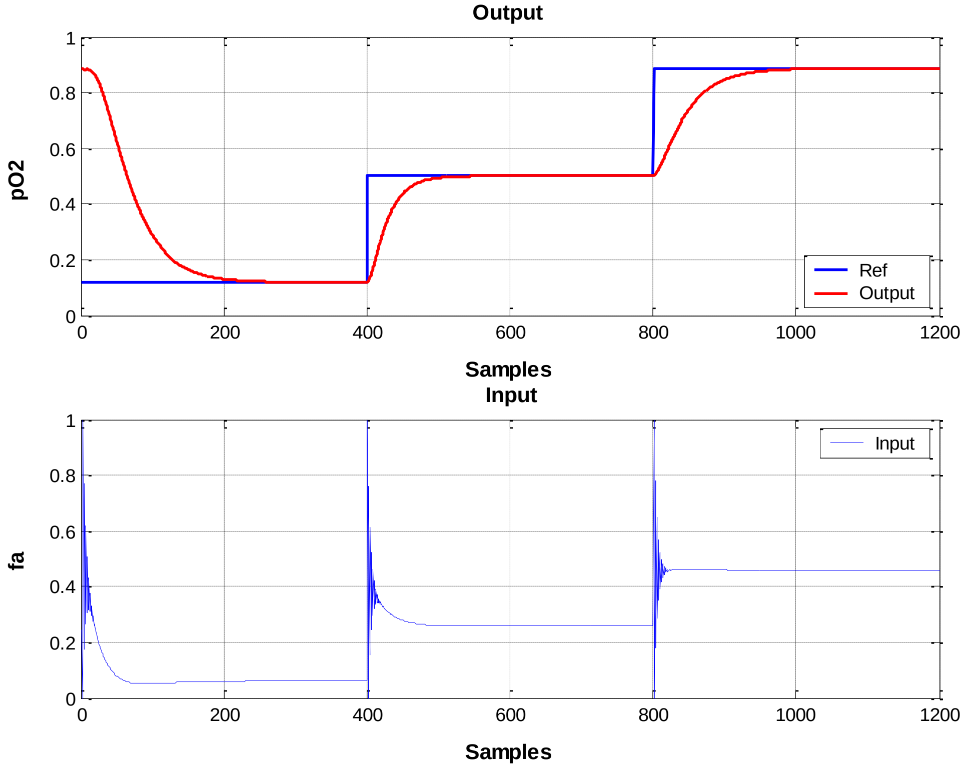


Figure 4-21 Single PI controller on second order *pO_2* model. M=3, scaling factor=20 with ringing. (Scaled data, sample time=10 sec)

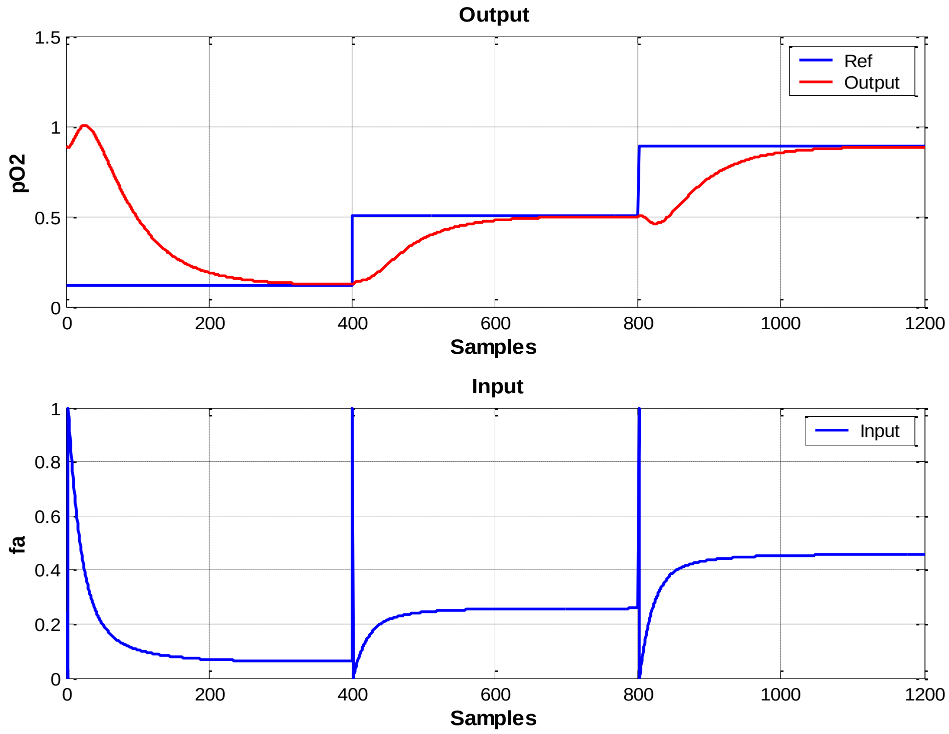


Figure 4-22 Single PI controller for second order *pO_2* model. M=3, scaling factor=20 after zero cancellation. (Scaled data, sample time=10 sec)

4.4.6 Local Linear Models Controller On Second Order *pO₂* Model

In this section local linear controllers are designed and implemented on second order *pO₂* model, M=3. These controllers are combined together and applied to the process to control and maintain *pO₂* level. Where the control coefficients were combined using equation (4.22) and the activation function for the local linear model networks were combined using equation (4.23) then the LLMN controller weights can be calculated using equation (3.38) and the final controller equation was calculated using equation (3.39).

$$G = \begin{bmatrix} g_{11} & g_{21} & g_{31} \\ g_{12} & g_{22} & g_{32} \\ g_{13} & g_{23} & g_{33} \\ -f_{11} & -f_{21} & -f_{31} \\ -f_{12} & -f_{22} & -f_{32} \end{bmatrix}, (4.22) \quad \text{and} \quad \emptyset = \begin{bmatrix} \emptyset_{11} & \emptyset_{21} & \emptyset_{31} \\ \emptyset_{12} & \emptyset_{22} & \emptyset_{32} \\ \emptyset_{13} & \emptyset_{23} & \emptyset_{33} \end{bmatrix} \qquad (4.23)$$

And giving,

$$G = \begin{bmatrix} 16.92 & 23.91 & 24.07 \\ -31.37 & -41.34 & -38.50 \\ 14.51 & 17.53 & 14.55 \\ 0.22 & -1.88 & -3.16 \\ 0.77 & 2.88 & 4.16 \end{bmatrix} (4.24) \quad \phi = \begin{bmatrix} 1 & 0.99 & 0.98 \\ 0.99 & 1 & 0.99 \\ 0.99 & 0.99 & 1 \end{bmatrix} \qquad (4.25)$$

4.4.6.1 Simulation Results

The results local linear controllers for second order *pO₂* model with M=3 have an oscillation in the input. This leads to unstable and unacceptable output, as illustrated in the Figures (4.23) and (4.24).

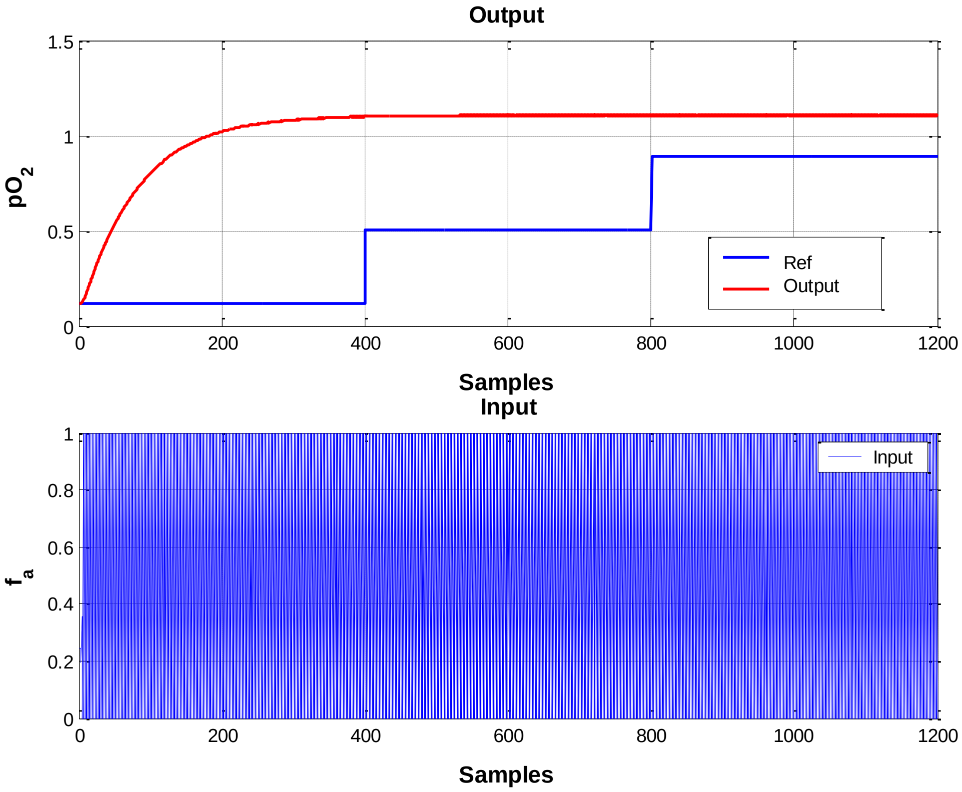


Figure 4-23 LLMN controller for second order *pO*$_2$ model. M=3, scaling factor=20 with ringing. (Scaled data, sample time=10 sec)

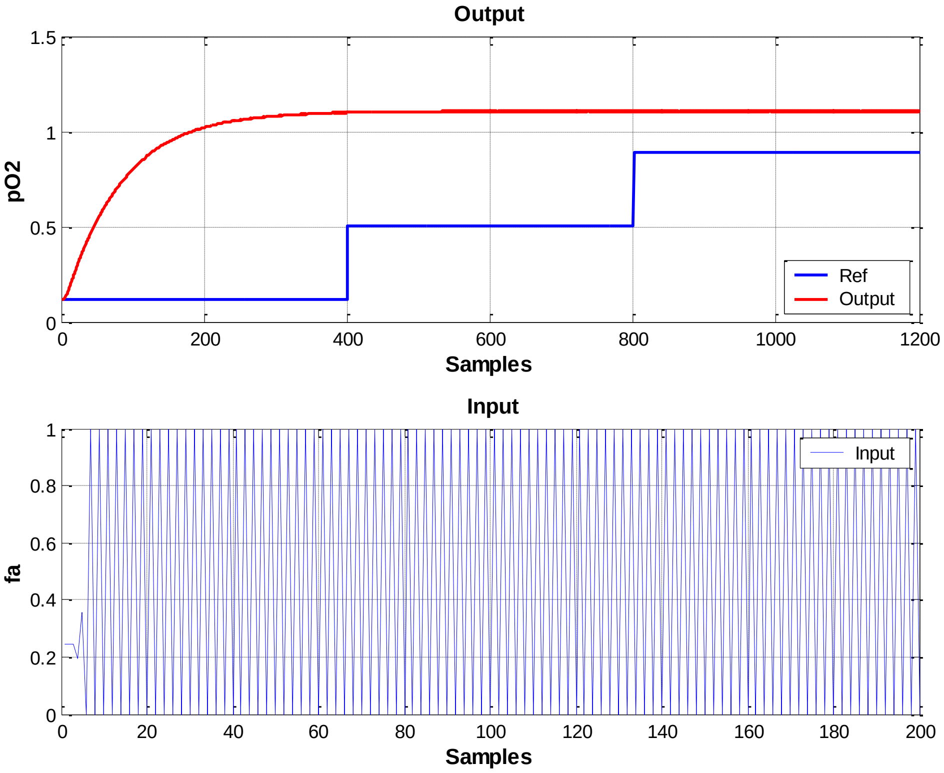


Figure 4-24 LLMN controller for second order *pO*$_2$ model. M=3, scaling factor=20 with ringing. (Scaled data, sample time=10 sec)

The matrix G and ϕ of the local linear model controller after zero cancellation became as:

$$G = \begin{bmatrix} 9.49 & 6.16 & 4.66 \\ -17.60 & -10.66 & -7.45 \\ 8.13 & 4.51 & 2.81 \\ 0.97 & 0.96 & 0.95 \\ 0.021 & 0.03 & 0.04 \end{bmatrix} \quad (4.26) \quad \phi = \begin{bmatrix} 1 & 0.99 & 0.98 \\ 0.99 & 1 & 0.99 \\ 0.99 & 0.99 & 1 \end{bmatrix} (4.27)$$

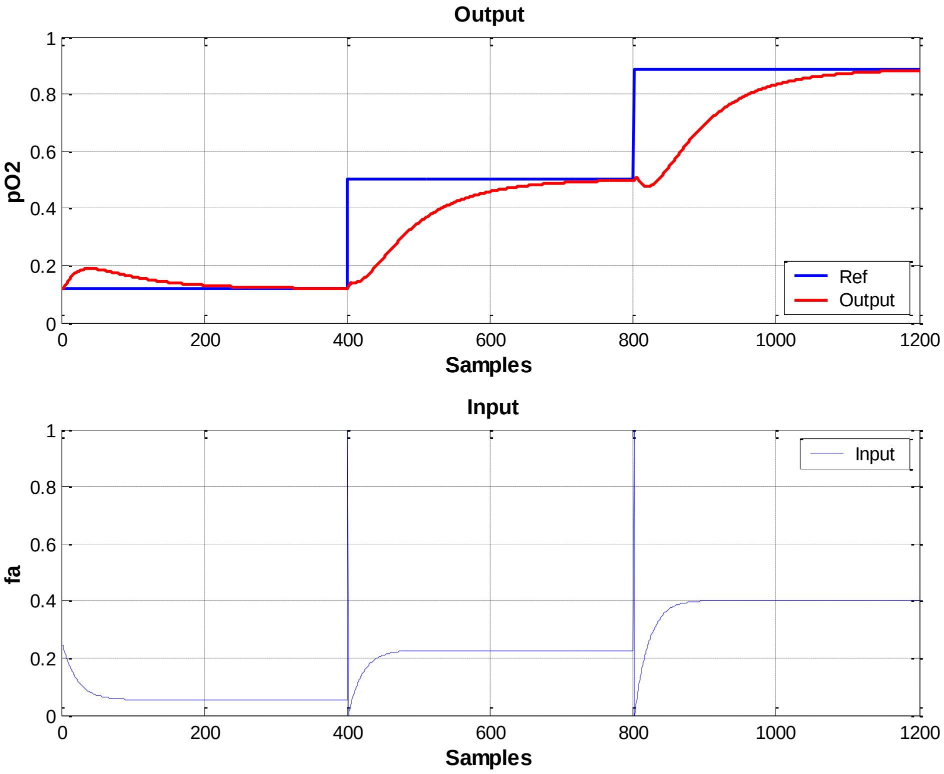


Figure 4-25 LLMN controller for second order *pO₂* model. M=3, scaling factor=20 after zero cancellation. (Scaled data, sample time=10 sec)

4.4.7 Discussion

Figure 4.21 demonstrates the performance of PI controller for one operating point, which is first centre on second order *pO₂* model. The output response started from 0.8871 and takes more than 200 seconds to match the reference set point at 0.11 before rising smoothly and taking approximately 200 seconds to reach steady state at 0.5. At the second 800, the response rises again smoothly like the first order curve to get steady state at 0.8871 after 200 seconds.

However, this result is not very encouraging as there is an oscillation on the input response. It can be seen from the figure that this is caused by ringing poles when the transfer function has $(z = -0.7783)$ in equation (4.19), strong oscillation is often considered as unsatisfactory, despite the process being controlled as intended. For this reason, the controller for the second order should be investigated. This problem is solved by zero cancellation using equation (4.20) and the result is improved as can be seen in Figure 4.22. Figures 4.23 and 4.24 show the local linear model controller performance for second order pO_2 model. Input at the bottom and output at the top of the figure. The particular interest in this figure is that there is an oscillation in the input response and the controller output does not track the reference set point. This is because of ringing poles and this is solved by zero cancellation. The results presented in Figure 4.25 highlight significant improvement in the response of the controller as it can be seen the local linear controller is able to react and track to the three reference set points properly and it takes about 350 seconds to get steady state for each set point. Comparison of MSE between PI and local linear controllers for second order pO_2 model with ringing poles and after zero cancelation for second order pO_2 model are presented in Tables. 4.2 and 4.3.

Table 4.2 Comparison of MSE between single PI controller and local linear controller with ringing

Controller	MSE
Single PI controller	0.03162
Local linear controller for the 3centres	0.3604

Table 4.3 Comparison of MSE between single PI controller and local linear controller after zero cancellation

Controller	MSE
Single PI controller	0.06147
Local linear controller for the 3centres	0.01819

4.5 Summary

This chapter has investigated, system identification for a nonlinear pO_2 process by using local linear model networks. The results were also compared with a gradient descent algorithm and observed that the application of local linear model networks gave smaller mean square error. Following this achievement, PI and local linear model controllers were designed and implemented to the process to control and maintain the dissolved oxygen pO_2 output. It can be concluded from the results that a local linear model controller can maintain and control pO_2 process with the first order model. However, increasing the order of the assumed process model has a major influence on the occurrence of ringing. As can be shown from the results, a controller designed using the second order model can lead to ringing behaviour. Therefore, care must be taken when a higher-order model is chosen to represent the process. This problem was solved by zero cancellation and the results improved.

Chapter 5 | Development of the Methods for Designing Local Linear Models for System Identification and Control of Real a Temperature Data

5.1 Introduction

The control of temperature is a very important element in chemical industries such as in water utilities. Nowadays the demand for temperature control has become more popular for industry evolution, as temperature has some characteristics, such as big inertia and difficulty developing accurate mathematical models, this can result in the poor performance of the control system (Yu and Hu, 2016).

Water treatment processes involve heat treatment and demand the stability of the water temperature, which must be maintained at a certain level. There are some factors that affect the temperature in water such as the weather. Therefore, a good controller should be considered to keep the temperature in the water at the required level.

A good controller should be considered to achieve desirable behaviour for the dynamic systems. This chapter explains the ability to use the local linear model networks technique (LLMNs) for system identification and control of a temperature process and the use of direct design control. In this chapter the temperature process contains input which is heating power (Q), while the output is temperature that keeps changes affected by the input. A good controller of water temperature is needed for domestic consumers.

There are many types of controller that could be used to control nonlinear process systems such as PI, PD and PID. In this chapter, the direct design method is designed to control and maintain the temperature in the process at certain operation regimes. The system identification and controller designed were done using MATLAB software.

5.2 Investigation of Local Linear Model Networks for Identification of Real Temperature Data.

This section begins by providing a brief description of the temperature process. It will then present an explanation of system identification of the temperature process by using local linear model networks technique (LLMNs). Finally, the results will be discussed.

5.2.1 Process Description

The temperature process is presented in Figure.5.1, where Q denotes the heating power as input which is between (0-100 W) and the data was collected with the process was under closed loop PID control to drive the temperature over the operating range between $(38 - 50°\mathrm{C})$. It was found difficult to tune the PID controller to cover the nonlinear operating range but the data collected was considered suitable for investigating nonlinear identification. The liquid level in the tank (and hence the liquid volume) is maintained at a constant value. The liquid in the tank is stirred continuously to make sure the temperature is consistent throughout the tank. The process was affected by some problems, such as the long rise time for the temperature due to the limitation of the heating power. These issues cause the process to be non-linear in both dynamic and static behaviour and this non-linearity of the process is demonstrated in the fixed parameter PID controller responses (Yu and Gomm, 2003). Suitable sample time for all variables was selected to be 10 seconds. The 2000 data points collected are presented in Figure.5.2. The process input u and output y are described as follows:

$$u = [Q], y = [T] \tag{5.1}$$

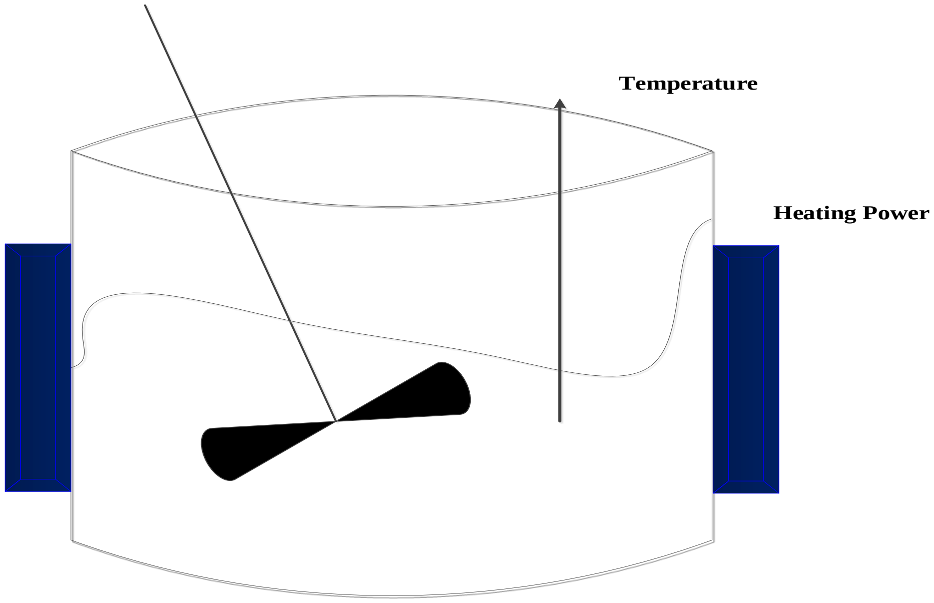


Figure 5-1Temperature process

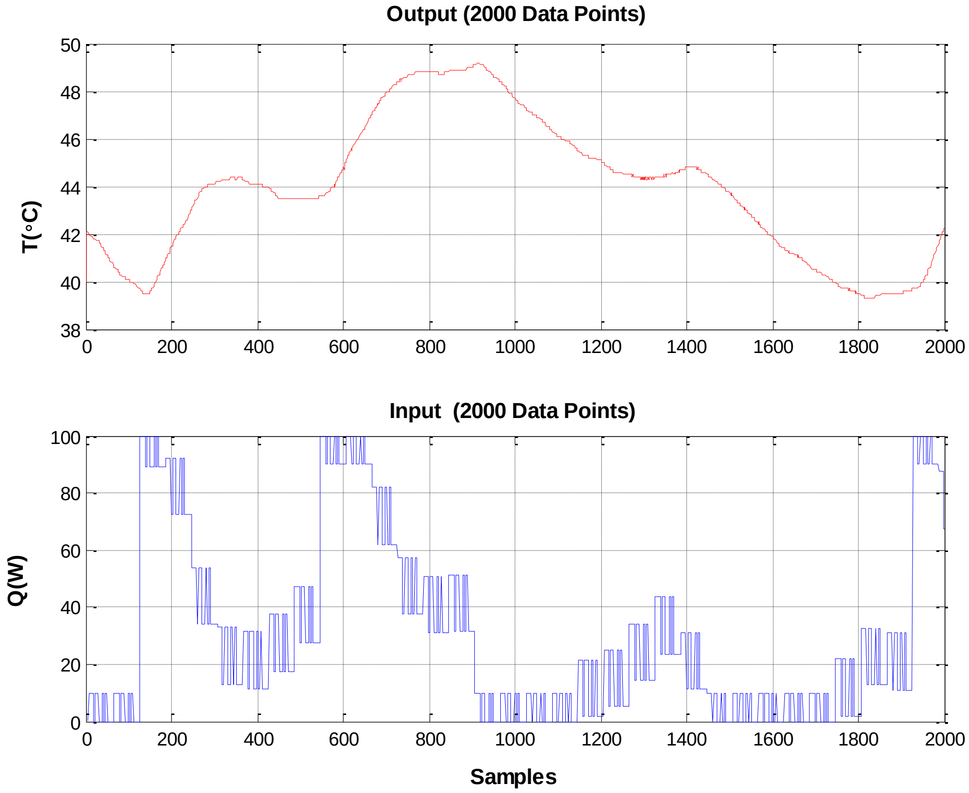


Figure 5-2 Measured real Temperature data for network training and validation (Sample time=10 sec)

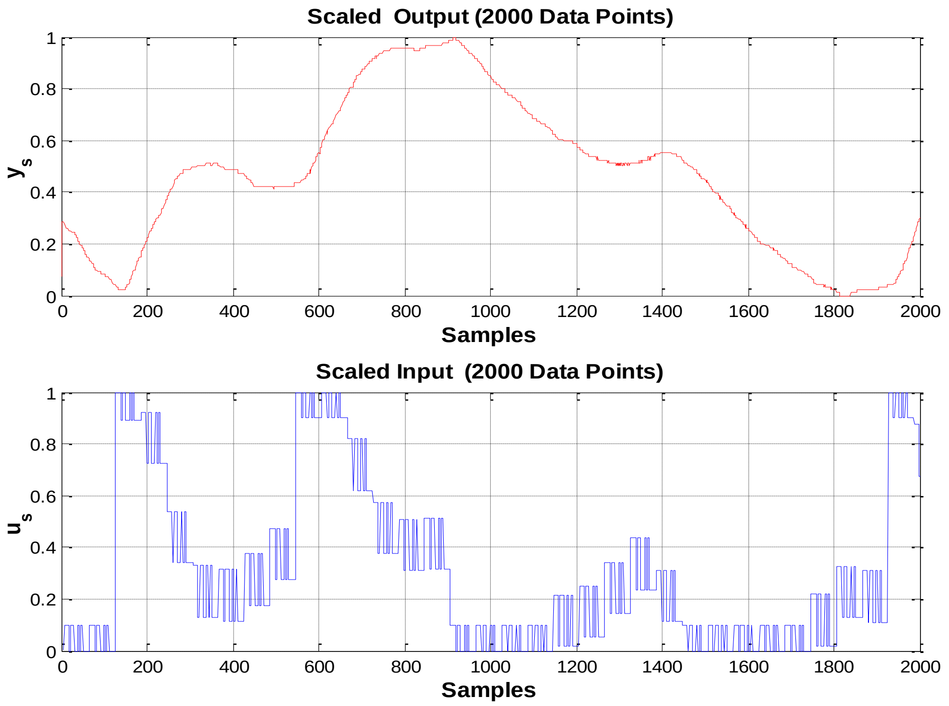


Figure 5-3 Scaled real Temperature data for network training and validation (Sample time=10 sec)

5.2.2 Local Linear Model Network Training and Evaluation for Temperature Data

This section describes the development of LLMN applied to the temperature data for system identification and results will be presented to predict the output. As explained in Chapter 3 the local linear model networks were used for system identification after making a comparison with the radial basis function neural networks (RBFNN) and ANFIS model, this method will be used again in this chapter to identify and control the temperature model.

By using the equation (3.19) in which in this case the input denotes the heating power Q and the output is the temperature T, hence the model become as:

$$x(k) = [T(k-1)...T(k-n)\ Q(k-1)...Q(k-n)\ \ ]^T \tag{5.2}$$

Equation (3.21) in Chapter 3 illustrated the activation function ($\emptyset_i$) for the network which is un-normalised. This equation is also used in this chapter, with $\tilde{x}(k) = T(k-1)$ chosen in the activation function.

In this research, the collected data are 2000 samples which are scaled for network training and test, and are presented in Figure 5.3. The data are split into two parts: 1200 samples for training and 800 samples for test. When data is large will be most straightforward method which is there is no difficulty. On the other hand, small training data samples can lead to a poor quality model. When the training missed data from some regimes, this could lead to the model not performing perfect in these regimes. Moreover, if significant data is lost in the test data, the evaluation of the model performance becomes unreliable (Nelles., 2002). A different number of model order is investigated to select the best model order on the training and validation data. It was noticed that the higher model order leads to increased mean square error.

5.2.3 Real Data Identification Results and Discussion

In this section LLMN structure as presented in Chapter 3 Figure 3.20, is used to predict the temperature output $y_s(k)$, which is scaled here using equation (3.16) and between [0 to 1]. The 2000 real data samples were scaled using equations (3.15 and 3.16) in section 3.2.2 before network training and split into two groups. The first 1200 data samples were used for training and the other 800 data samples for testing and validation.

In this section, different numbers of local models and different orders of network model inputs have been used in training experiments and equation (5.2) is selected when model order $n = 1$. The centres and widths in the validity functions of the LLMN were calculated using K-means algorithm and P-nearest neighbours method respectively. Initial model parameters are set as,$w = 1.0 \times 10^{-6} \times U_{M\times 2n}$, $P(0) = 1.0 \times 10^{8} \times I_{M\times 2n}$ and λ is selected as 0.999. Then weights are updated using recursive least square algorithm for model output validation.

In this section, after various experiments varying the widths scaling in the hidden nodes, a scaling factor of $\alpha = 10$ $(\sigma \times 10)$ achieves small train and test errors in one step ahead prediction. The mean square error for LLMN training and test with different model order (n=1, 2, 3, 4), different local model numbers and different width scaling are shown in Figures 5.4, 5.5, and 5.6. The LLMN here trained as on one step ahead prediction, and the mean square errors for first order model with 3 local linear models were 0.00000366 and 0.000001098 for trained and validation networks, while the mean square errors for first order with 4 local linear models were 0.0000033662 and 0.0000014197 for trained and validation networks respectively. On the other hand, to train the local linear model network independently, the model output is fed back to the network and used as input. The dependent model is called one step ahead prediction and the independent model is called multi step ahead prediction. The LLMN independent model was done for first order model with M=3 and M=4 and the mean square error became less when M=4 and the prediction results are illustrated in Figures 5.7, 5.8, 5.9 and 5.10. The mean square errors for the first order model with M=3 for test and train are 0.0028 and 0.0032 respectively. While the mean square errors for the first order model with M=4 are 0.0023 and 0.00038097. This means the first order temperature model with 4 local models is still the better choice by reducing the mean square error. Figures 5.11 and 5.12 demonstrate the final trained and testing one step ahead prediction results for the first order temperature model with scaling factor $\alpha = 10$ and number of local models M=4. However, in the controller section, the implementation of local linear models controller on the first order model has been achieved and the results reveal that when the network has 4 local models on the first order model the controller output response does not take as long time as that with an increased number of local linear models to get steady state.

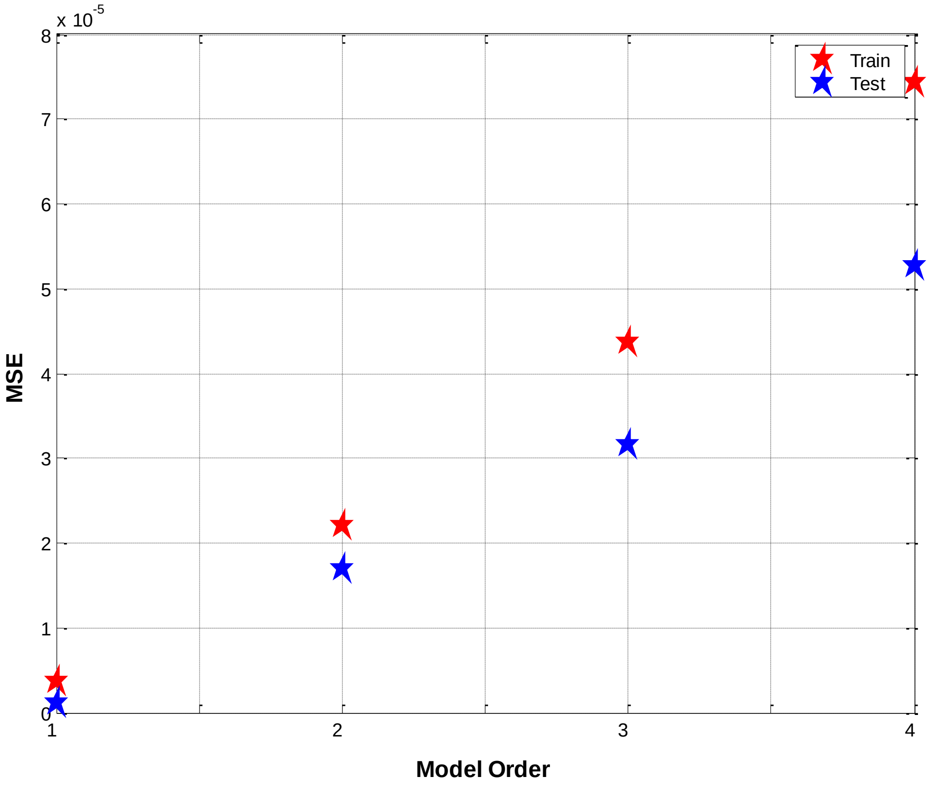


Figure 5-4 Comparison between MSE for LLMN training and test with different model orders for 3 local models

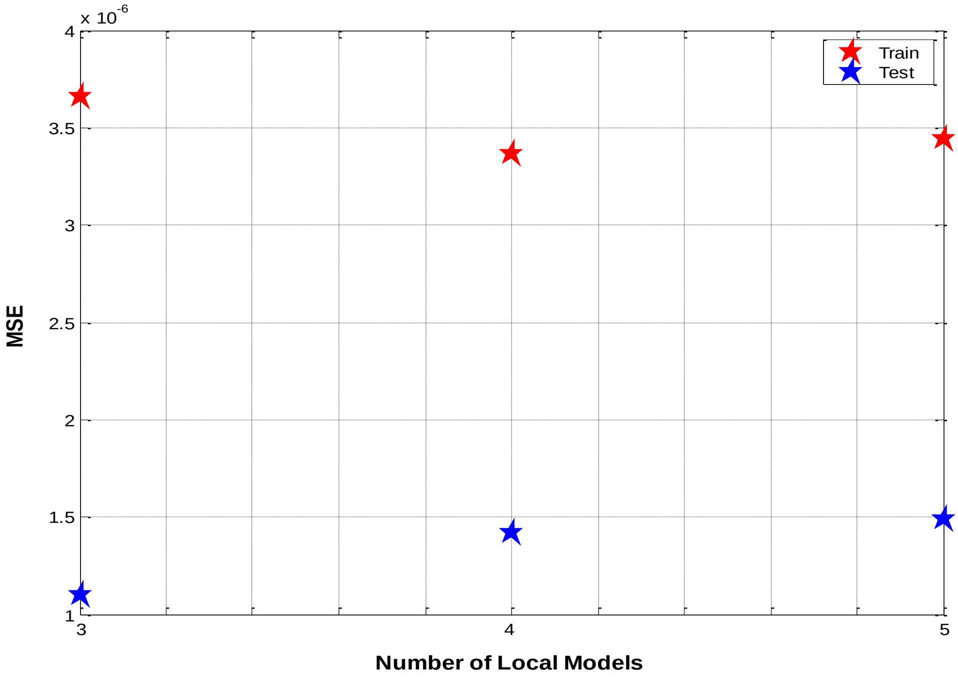


Figure 5-5 Comparison between MSE for LLMN training and test with different number of local models on first order

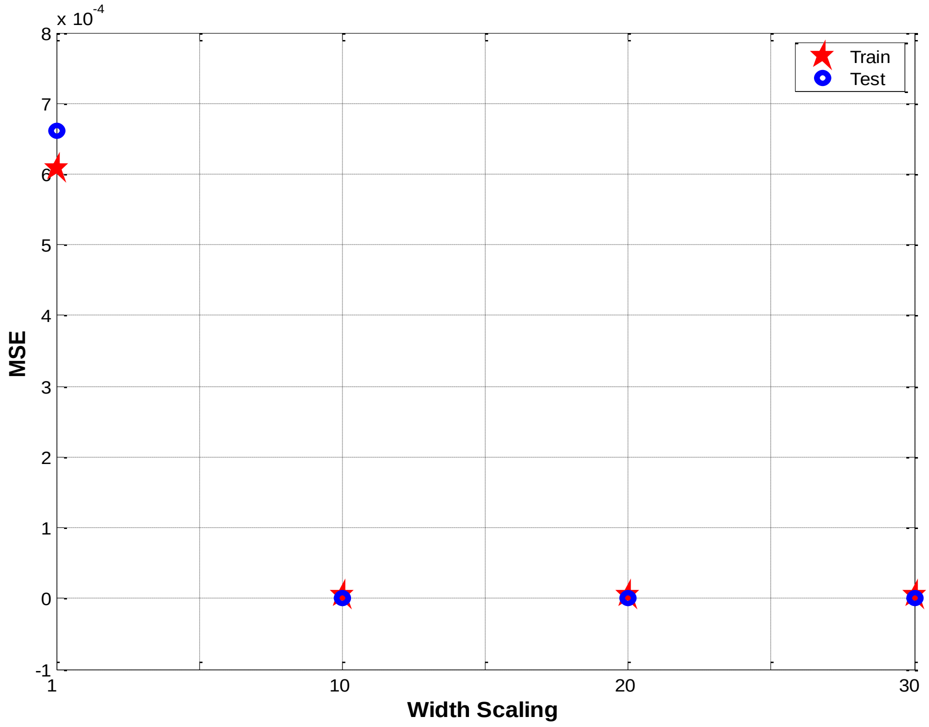


Figure 5-6 Comparison between MSE for LLMN training and test with different width scaling on first order, M=3

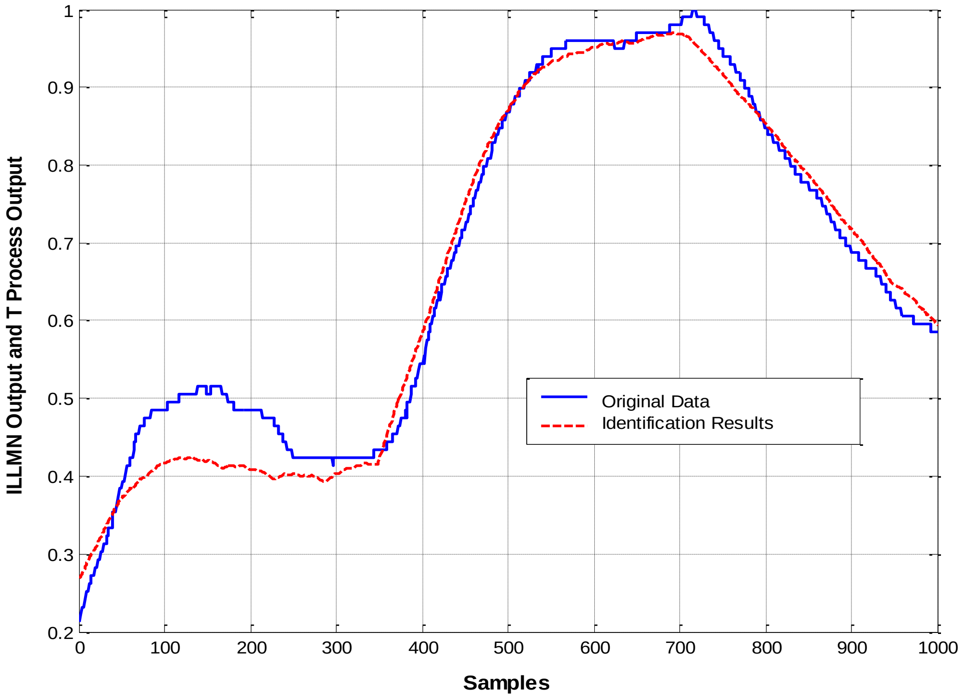


Figure 5-7 LLMN Independent model results for training data first order temperature model. M=3.scaling factor=10. (Scaled data, sample time=10 sec)

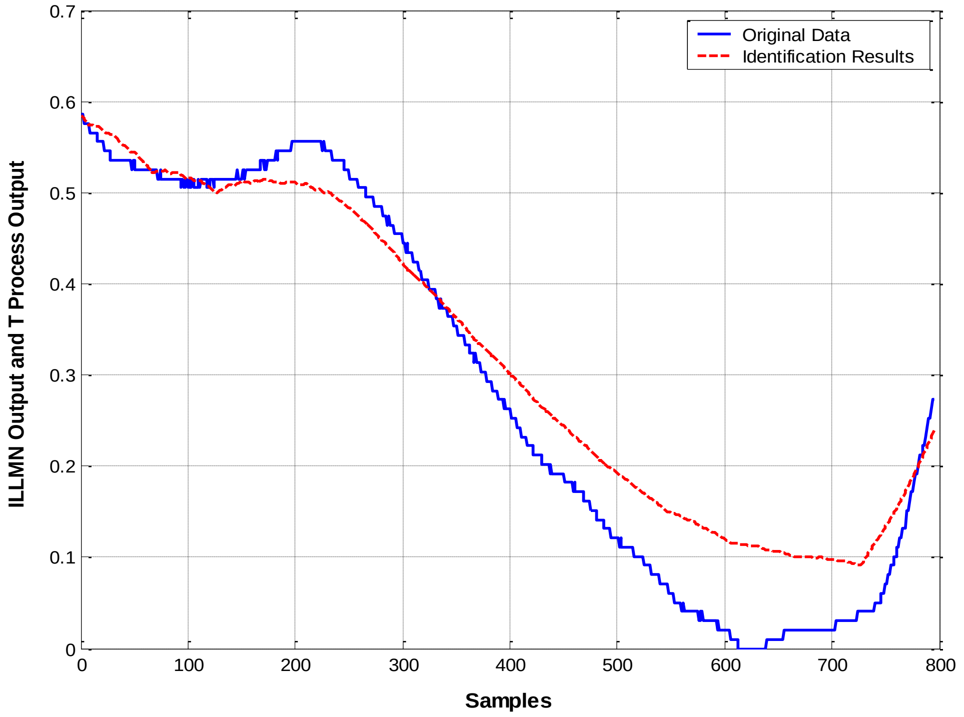


Figure 5-8 LLMN Independent model results for testing data first order Temperature model. M=3.scaling factor=10. (Scaled data, sample time=10 sec)

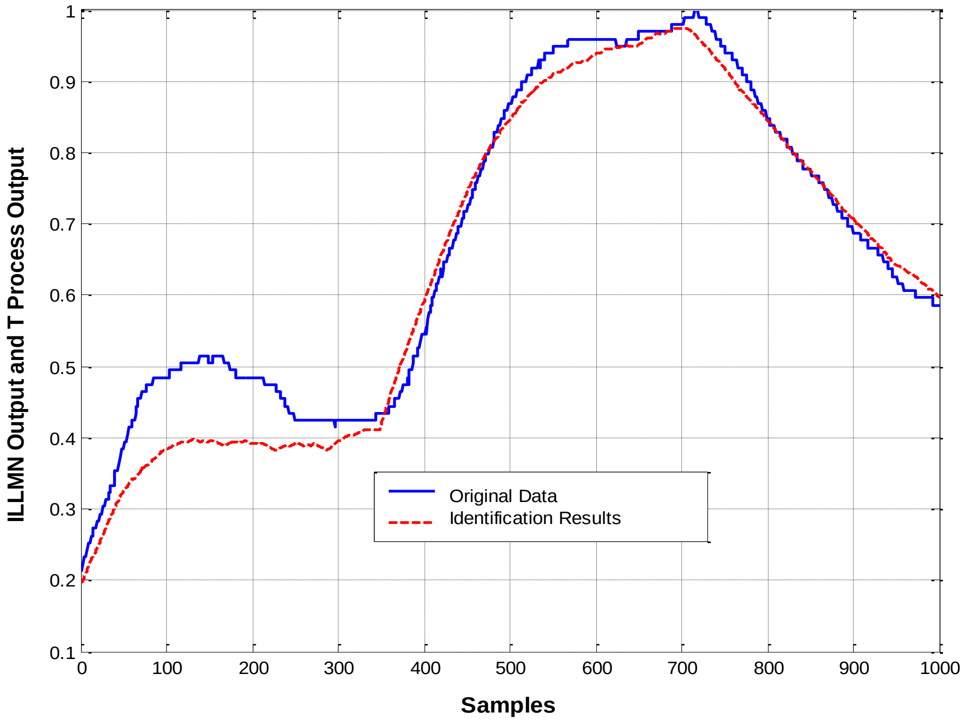


Figure 5-9 LLMN Independent model results for training data first order Temperature model. M=4.scaling factor=10. (Scaled data, sample time=10 sec)

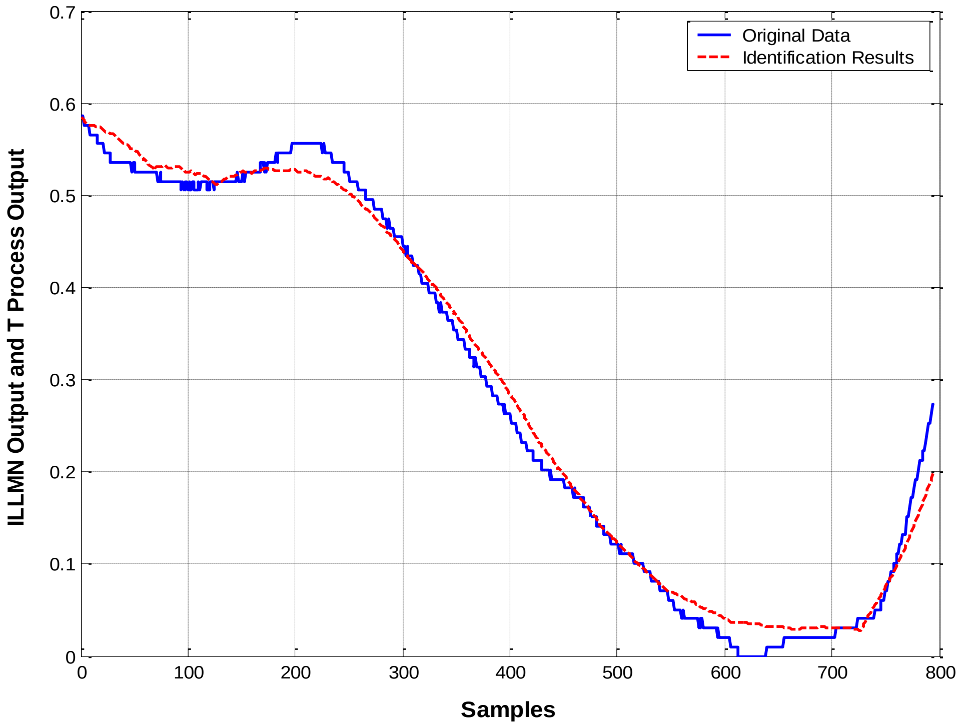


Figure 5-10 LLMN Independent model results for testing data first order Temperature model. M=4.scaling factor=10. (Scaled data, sample time=10 sec)

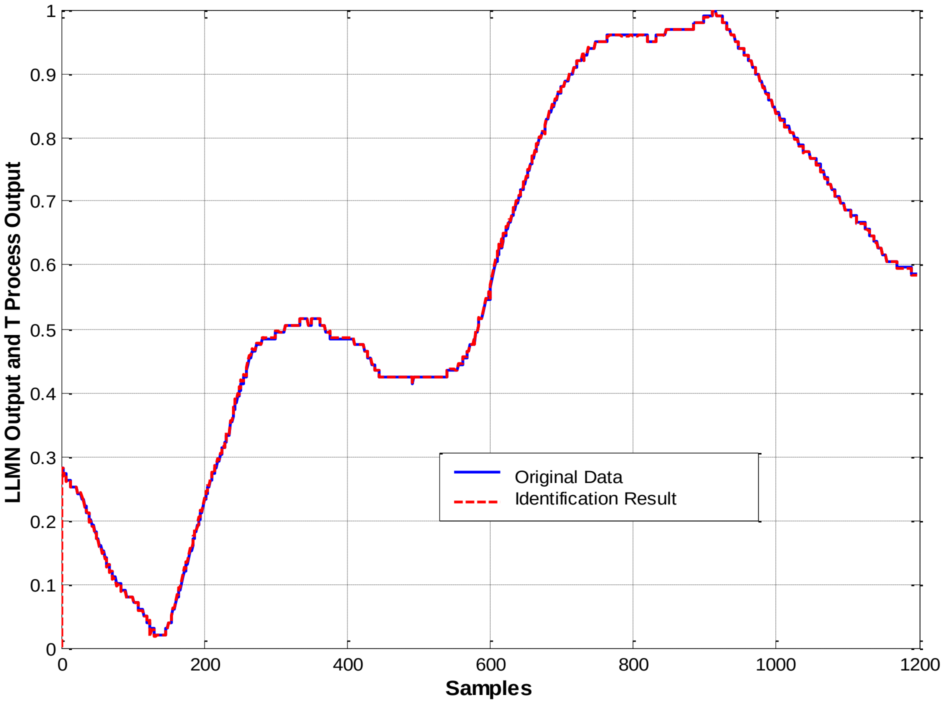


Figure 5-11 LLMN trained network for first order Temperature model. M=4, scaling factor=10. (Scaled data, sample time=10 sec)

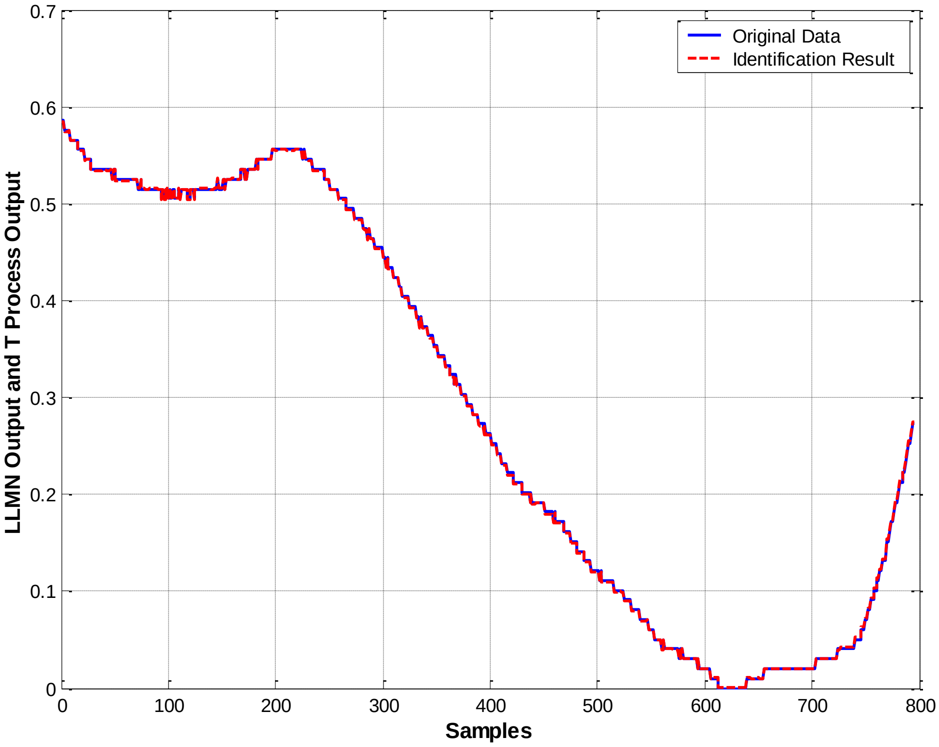


Figure 5-12 LLMN testing network for first order Temperature model. M=4, scaling factor=10. (Scaled data, sample time=10 sec)

5.3 Development and Investigation of Control Based on Local Linear Model Networks for Temperature Process

As discussed in the previous section, system identification of temperature process was achieved using local linear model networks (LLMNs). The first order model with scaling factor $\alpha = 10$ was selected after performing a variety of experiments that gave acceptable results with less mean square error. Because of these results PI and local linear model controllers are applied to the identified temperature process to control temperature and the results are compared in the following section.

5.3.1 Direct Design Control Procedure

As demonstrated in Chapter 3, the equation (3.23) was used to design the transfer function controller $G_C(z)$. The same equation will be used in this chapter to design PI and local linear controllers for the temperature process:

$$G_C(z) = \frac{1}{G_S(z)} \times \frac{G_T(z)}{1-G_T(z)} \tag{5.3}$$

5.3.2 PI Controller for First Order Temperature Model

5.3.2.1 PI Controller for First Order Temperature Model, M=3, scaling factor=10

The local linear model network (LLMNs) was trained for the first order temperature model with three local models M=3 and scaling factor $\alpha = 10$ and the mean square error is 0.00000366, Next, consideration is given to calculating the first order transfer function to enable us to implement the controller transfer function around an operating point which is around centre two 0.501 in this case. The first order model transfer function $G_S(z)$was solved using equation (3.27) and $G_T(z)$ using equation (3.30) and became as:

$$G_S(z) = \frac{0.00518}{z-0.9968} \quad (5.4) \qquad G_T(z) = \frac{0.05}{z-0.9968} \quad (5.5)$$

These equations were substituted into equation (5.3), then the controller transfer function became as:

$$G_c(z) = \frac{9.653z-9.622}{z-1} \tag{5.6}$$

Where g_0 = 9.653 and g_1 = -9.622

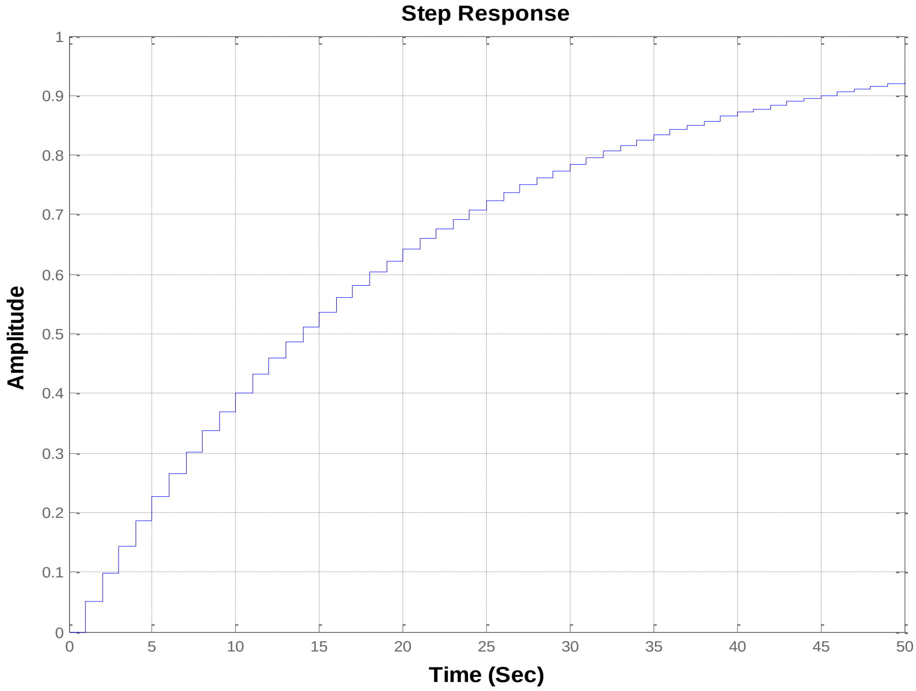


Figure 5-13 Step response of closed loop transfer function

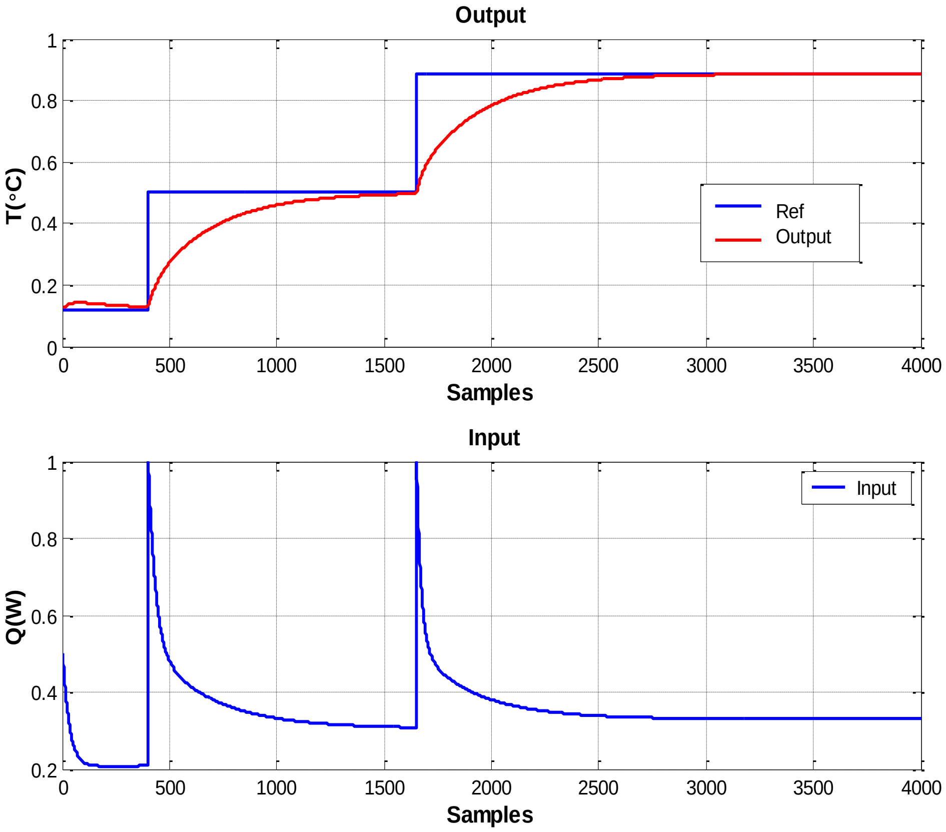


Figure 5-14 PI controller for first order Temperature model around centre two. M=3, scaling factor=10. (Scaled data, sample time=10 sec)

5.3.2.2 PI Controller for First Order Temperature Model, M=4, scaling factor=10

The local linear model network was trained for the first order temperature model with four local models and M=4 and scaling factor $\alpha = 10$, the mean square error is 0.0000033662. Next, consideration is given to calculating the first order transfer function to enable us to implement the controller transfer function around the operation point which is around centre three 0.5477 in this case. The first order model transfer function $G_S(z)$was solved using equation (3.27) and $G_T(z)$ using equation (3.30) and became as:

$$G_S(z) = \frac{0.005143}{\mathrm{z} - 0.9973} \quad (5.7) \qquad\qquad G_T(z) = \frac{0.05}{\mathrm{z} - 0.95} \quad (5.8)$$

$$G_c(z) = \frac{9.721\ \mathrm{z} - 9.695}{\mathrm{z} - 1} \quad (5.9)$$

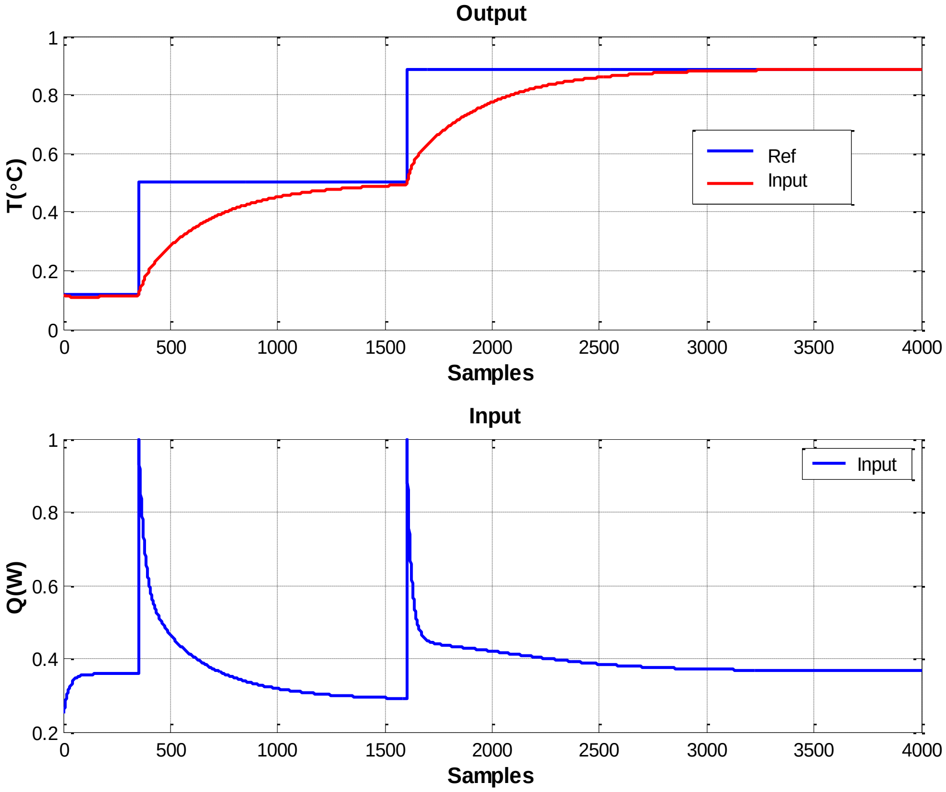


Figure 5-15 PI controller for first order Temperature model. M=4, scaling factor=10. (Scaled data, sample time=10 sec)

5.3.2.3 PI Controller for First Order Temperature Model, M=5, scaling factor=10

After system identification for temperature process on the first order model with five local models and scaling factor $\alpha = 10,$ we found that the mean square error in this case for training data is 0.00000344, which is still good. Then the PI controller is designed around the network centre 1 which is 0.488 to investigate the behaviour of the process under the controller.

$$G_S(z) = \frac{0.005054}{\mathrm{z} - 0.9971} \quad (5.10) \qquad\qquad G_T(z) = \frac{0.05}{\mathrm{z} - 0.95} \quad (5.11)$$

Firstly, we must calculate the controller transfer function around centre 1 which is 0.4886 by using equation (3.23). This is becoming as

$$G_c(z) = \frac{9.893\ \mathrm{z} - 9.864}{\mathrm{z} - 1} \qquad (5.12)$$

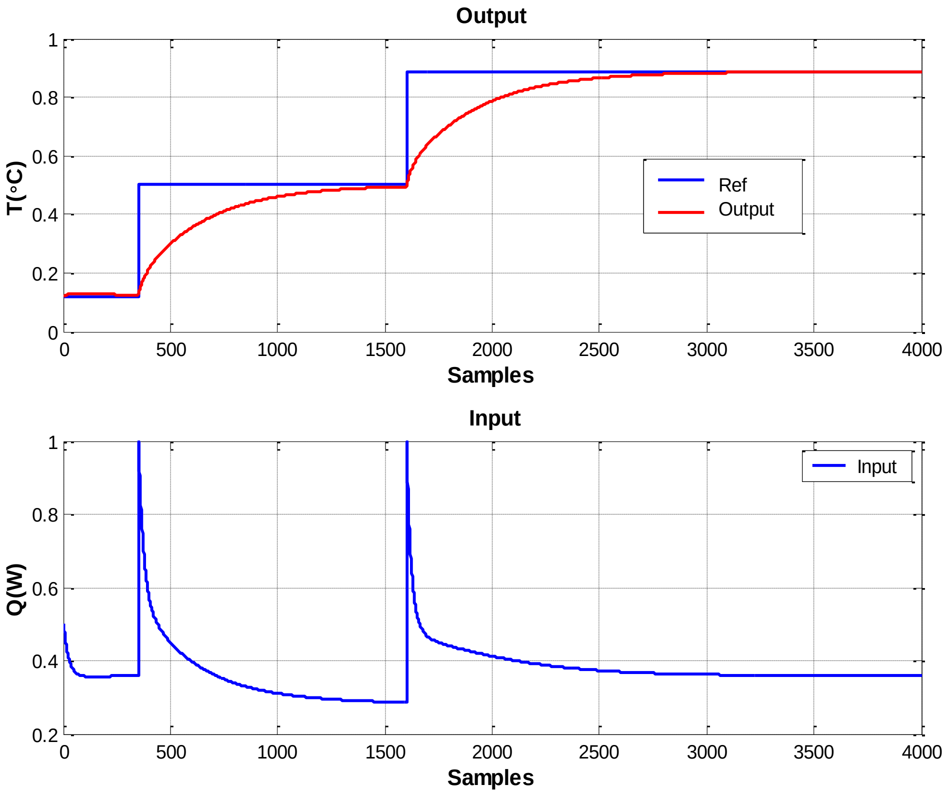


Figure 5-16 PI single controller for first order temperature model around centre one. M=5, scaling factor=10. (Scaled data, sample time=10 sec)

5.3.2.4 Discussion

Figure 5.13 represents a step response of the closed loop first order transfer function, while Figure.5.14 illustrates the simulation response of the PI controller for the first order temperature model, M=3 in the (upper part). As can be observed, it started from first set point at 0.11 and remains for about 350 seconds before rose to reach the second set point around 0.5 after about 1250 seconds then rose again to get steady state at the third set point after about 1250 seconds. While the (bottom part) shows the input response. Figure 5.15 illustrates the PI controller response for first order temperature model, M=4 at around centre 3. , it started from first set point at 0.11 and remains for about 350 seconds before rose to get the second set point around 0.5 after about 1200 seconds then rose again to get steady state at the third set point after about 1150 seconds. Figure. 5.16 illustrates the simulation response of the PI controller for the first order temperature model, M=5. , it started from first set point at 0.11 and remains for about 350 seconds before rose to get the second set point around 0.5 after about 1150 seconds then rose again to get steady state at the third set point after approximately 1200 seconds.

5.3.3 Local Linear Model Controllers for First Order Temperature model

As discussed Chapter 3, section (3.8.3), the local linear model controllers have been designed. The same procedure is applied in this section to design local linear controllers on the first order temperature model for the three operating regions (3 centres in this case). The activation function for the local linear model networks were combined using matrix (5.13) and the control coefficients were combined using matrix (5.14). Then the LLMN controller weights can be calculated using equation (3.38) and became as matrix (5.15) and the final controller equation was calculated using equation (3.39).

$$\emptyset = \begin{bmatrix} \emptyset_{11} & \emptyset_{21} & \emptyset_{31} \\ \emptyset_{12} & \emptyset_{22} & \emptyset_{32} \\ \emptyset_{13} & \emptyset_{23} & \emptyset_{33}1 \end{bmatrix} \qquad \emptyset = \begin{bmatrix} 1 & 0.98 & 0.96 \\ 0.99 & 1 & 0.99 \\ 0.96 & 0.99 & 1 \end{bmatrix} \qquad (5.13)$$

$$G = \begin{bmatrix} g_{11} & g_{21} & 1 \\ g_{12} & g_{22} & 1 \\ g_{13} & g_{23} & 1 \end{bmatrix} \qquad G = \begin{bmatrix} 10.94 & -10.92 & 1 \\ 9.65 & -9.62 & 1 \\ 15.42 & -15.33 & 1 \end{bmatrix} \tag{5.14}$$

$$W = 1e^4 * \begin{bmatrix} 0.6237 & -0.6198 & 0.0017 \\ -1.2397 & 1.2318 & -0.0033 \\ 0.6294 & -0.6253 & 0.0017 \end{bmatrix} \tag{5.15}$$

5.3.3.1 Simulation Results for First Order Temperature Model with different number of local models, M=3, M=4 and M=5, scaling factor=10

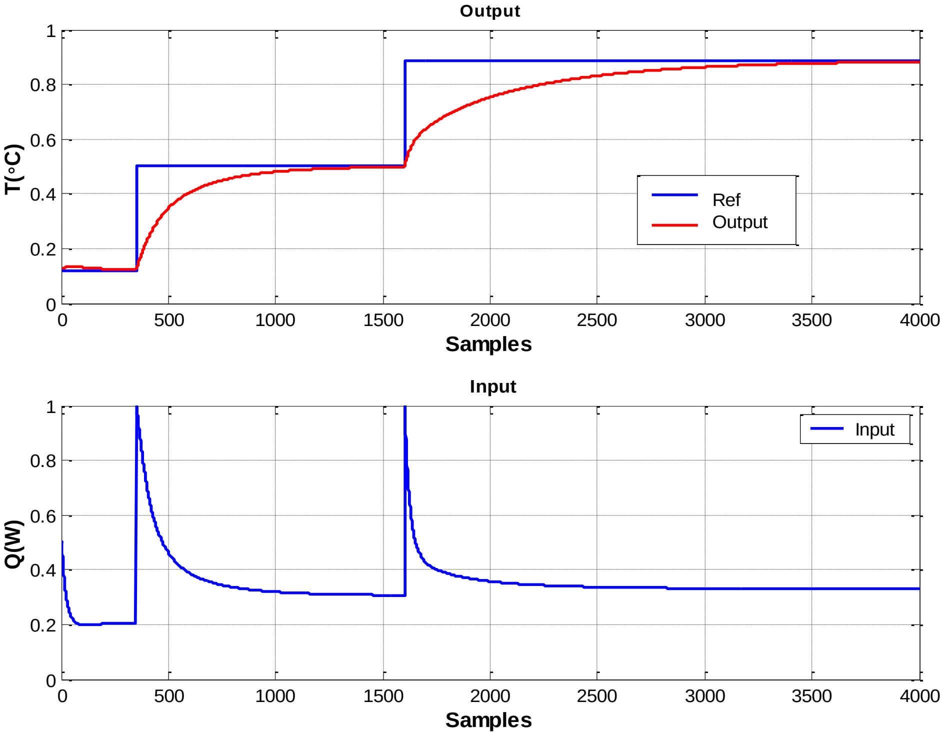


Figure 5-17 LLMN controller for first order temperature model. M=3, scaling factor=10. (Scaled data, sample time=10 sec)

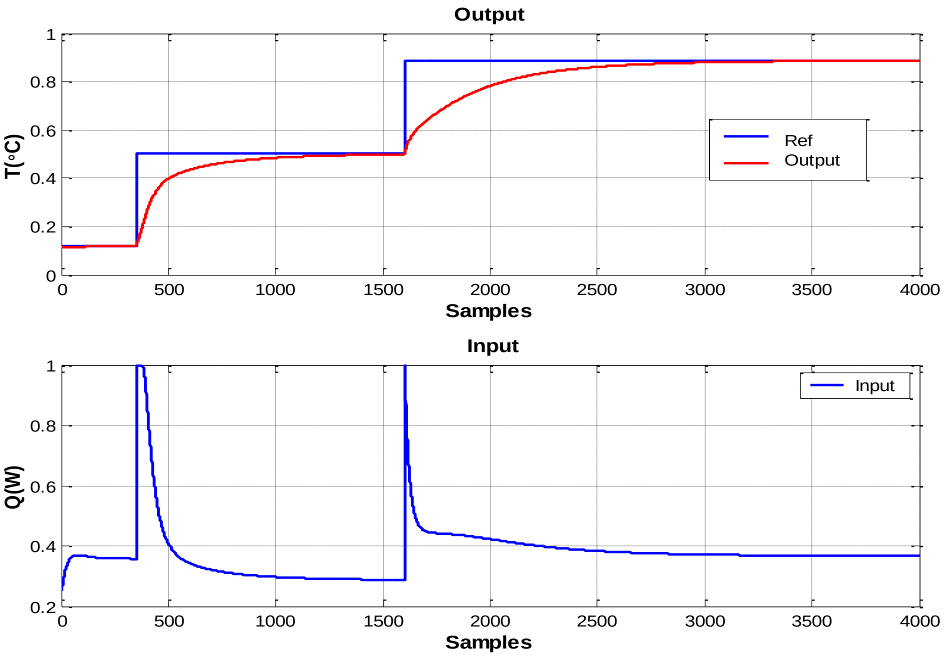


Figure 5-18 LLMN controller for first order temperature model. M=4, scaling factor=10. (Scaled data, sample time=10 sec)

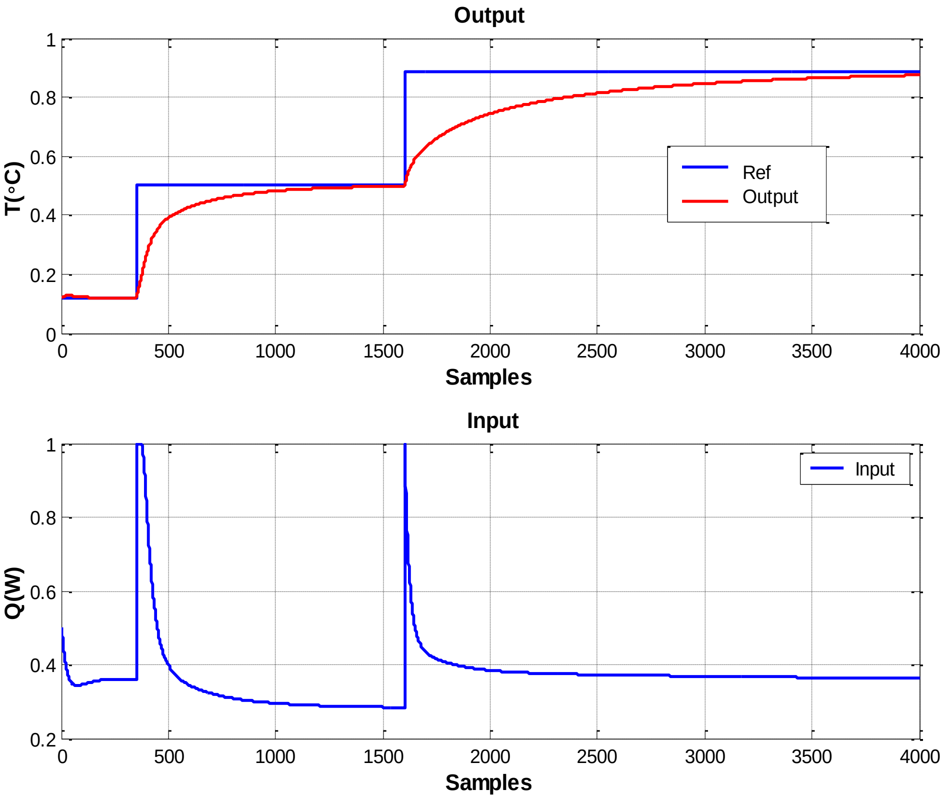


Figure 5-19 LLMN controller for first order temperature model. M=5, scaling factor=10. (Scaled data, sample time=10 sec)

5.3.3.2 Discussion

Figure.5.17 illustrates the LLMN controller response for the first order temperature model, M=3. It can be seen from the figure that temperature output (red line) is tracking the three set points changes (blue line). It began from around 0.11 for approximately 350 seconds before rising up to reach the second set point after about 1050 seconds, finally reaching the third point after about 1800 seconds. From the results, it is observed that the controller response needs more time to reach the steady state at the third set point compared with the other two points. While Figure 5.18 shows the LLMN controllers for first order temperature model, M4. It started from around 0.11 for approximately 350 seconds before rising up to reach the second set point after about 950 seconds, finally reaching the third point after about 1000 seconds. Figure.5.19 illustrates that the local linear model controllers output response (upper part) for first order model with 5 local models. It started from around 0.11 for about 350 seconds before rose to reach the second set point after about 1000 seconds, finally reaching the third point after about 2000 seconds. The results show the behaviour of five local linear models when combined together and the temperature output is tracking the three set points. However, from the results of the local linear model controllers, it can be observed that more time is required to reach the third set point when the number of local models M3 and M5 compared with M4. In addition, the local linear model controller for the first order temperature model, M=4 giving improved result with less mean square error, Table 5.1 shows the comparison of MSE of local linear controllers for first order temperature model with different number of local models.

Table 5.1 Comparison of MSE of local linear controllers for first order temperature model with different number of local models

Controller	MSE
Local linear controller for the 3 centres	0.00926454605409
Local linear controller for the 4 centres	0.00727019031171
Local linear controller for the 5 centres	0.00931683226467

5.4 Summary

This chapter has investigated system identification for a nonlinear temperature process (T) using local linear model networks (LLMNs). The model order, number of local models and width scaling factor were selected after the conducting of various experiments. The results presented that the first order with M=3 and scaling factor=10 gave less mean square error in one step ahead prediction. Moreover, the mean square error became less for first order model with M=4 and scaling factor=10 when the local linear model networks trained independently. In the controller section, the PI controller was designed using direct design method and implemented to the process to control and maintain the temperature. Moreover, local linear model controllers on first order temperature model with M=3, M=4 and M=5 were designed to control the temperature output. From the results, it can be observed that the first order model gives an acceptable output response. We also observe that when the LLM network has 4 local models the output response has less time to reach a steady state and achieved less mean square error compared with M=3 and M=5 local models.

Chapter 6 Conclusion and Future Work

6.1 Conclusion

The first objective of this research was to identify and control real pH data using the local linear model network (LLMN) strategy, as described in Chapter 3. First of all, the Simulink model of the pH reactor was established. The reactor contains acid, which is set as a constant and neutralised by varying the base. The PI controller for the simulation has been designed using internal model control (IMC). That was used to become familiar with the process dynamics behaviour and nonlinearity. Moreover, we examined the deficiencies of conventional control applied to this highly nonlinear process. The results for the PI controller are achieved for different pH values. Second, different algorithms for system identification were used in this research, such as radial basis function neural network (RBFNN), which is used for real pH data system identification. The hidden layer nodes of the RBFNN were selected following a range of experiments. The network centre was computed using the K-means clustering algorithm and the width (σ) was chosen using the p-nearest neighbours algorithm. The first order model was chosen after various experiments which gave less mean square error. The other two techniques called local linear model networks (LLMN) and adaptive neuro-fuzzy inference system were also investigated for system identification of real pH data. The results for these networks have been compared and show that the ability of these networks to accurately represent the process when suitable choices and optimisations are made for various network parameters. However, local linear model networks (LLMNs) have potential and are powerful for approximation of nonlinear dynamic systems. Their behaviour gave improved results with reduced mean square error values with those of radial basis function networks and adaptive neuro-fuzzy inference system for the identified pH system. The results of local linear model networks have been achieved with less number of parameters which is 16 compared with 24 parameters used in RBFNN. In other words, the LLMN structure is smaller than the RBFNN in this application. The simulation results for performance have been investigated and implemented in MATLAB/SIMULINK. These results give support to the application of the local linear model networks method for control of chemical processes.

Because of these results PI and local linear model controllers are designed on the first order pH model using the direct design method and applied to the process to control and maintain the pH output and the results show the capability of the LLMN method to control the nonlinear process.

In Chapter 4, the second objective investigated is system identification for a nonlinear pO_2 process by using local linear model networks. A width scaling factor is a significant issue in the network. A gradient descent algorithm was used to optimise the width scaling factor in the network automatically and the results were compared. It is observed that the application of local linear model networks gave smaller mean square error than that with gradient descent. After this achievement PI and local linear model controllers for first and second order pO_2 models were designed and implemented to the process to control and maintain the dissolved oxygen pO_2 output. The controllers on the first order model for 3 local models were achieved and gave acceptable results which gave support to using local linear model networks. On the other hand, the main finding in this chapter is that by increasing the number of model orders to be second order, the controller is faced with a challenge to obtain a stable output. To address this problem, the second order model had to be investigated, the problem that was causing an unstable response by ringing poles when the transfer function has $(z = -0.77)$ and that was solved by zero cancellation; moreover, the results improved and became better.

In Chapter 5, the third goal of this research was achieved in system identification for a nonlinear temperature process (T) using local linear model networks. The model order, number of local models and width scaling factor were chosen after various experiments and the results presented that the first order with M=3 and scaling factor=10 gave less mean square error which can be choice for the local linear model networks. However, when local linear model network was trained independently the mean square error became less for the first order model with M=4 and scaling factor=10. Because of these results PI and local linear model controllers were designed using the direct design method and implemented to the process to control and maintain the temperature output. Moreover, the control action for the output was behaving much better when M=4, compared with M=3 and M=5. From the results, it can be observed that first order model gives an acceptable output response compared with higher order models. Furthermore, the results revealed that the mean square error became less when the number of local models M=4 in the controller, compared with having number of local models M=3 and M=5.

6.2 Future Work

The developed local linear model networks application needs to be tested in a real chemical process laboratory to assess its power and effectiveness. For future work the following suggestions should be given due consideration:

1. The proposed local linear model networks application for system identification and control need to be tested in the chemical industry. This is because, in the industry, the conditions will be more complicated than in the simulation.

2. From this research, it is believed that width scaling presents a significant issue in local linear model network training by reducing the mean square error. Therefore, it would be interesting to discover a way to optimise the width scaling in the local linear model network training.

3. The pH, dissolved oxygen and temperature are important factors in the chemical reactor and fermentation process. As the fermentation process is complicated it would be more beneficial to consider using local linear model network structure to control the fermentation process.

4. As most industrial systems are multivariable, a local linear model controller should be investigated and applied to control multi-input and multi-output processes.

References

Abdelhadi, A., Gomm, J.B., Yu, D. and Rajarathinam, K., 2014, September. Nonlinear system identification and control of a pH process using Local Linear Model Networks strategy. *Proceedings of 20th International Conference on Automation and Computing.* Cranfield University, Bedfordshire, UK. pp. 254-259. IEEE.

Abdelhadi, A., Gomm, J.B., Yu, D. and Rajarathinam, K., 2014, July. Comparison of RBF and local linear model networks for nonlinear identification of a pH process. In *2014 UKACC International Conference on Control (CONTROL)*. pp. 361-366. IEEE.

Ahmad, F., Isa, N. A. M., Osman, M.K. and Hussain, Z., 2010, November. Performance comparison of gradient descent and genetic algorithm based artificial neural networks training. *10th International Conference on Intelligent Systems Design and Applications*. pp. 604-609.

Ahmad, Z., Rahim, N. A., Bahadori, A. and Zhang, J. 2017 . Improving water quality index prediction in Perak River basin Malaysia through a combination of multiple neural networks. *river basin management*, 15(1). pp. 79-87.

Allgower, F., and Zheng, A., 2000. *Nonlinear model predictive control*: Birkhauser Verlag, Berlin, pp. 31.

Allgöwer, F. and Zheng, A. eds., 2012. *Nonlinear model predictive control* (Vol. 26). Birkhäuser.pp.42

Arefi, M., Montazeri, A., Poshtan, J. and Jahed-Motlagh, M., 2006, December. Nonlinear model predictive control of chemical processes with a wiener identification approach. In *2006 IEEE International Conference on Industrial Technology* (pp. 1735-1740). IEEE.

Astrom, K. J., & Hagglund, H. 1995.PID controllers: Theory, design and tuning (2nd ed.). Research Triangle Park, NC: Instrument Societyof America.

Ay, M. and Kisi, O., 2011. Modeling of dissolved oxygen concentration using different neural network techniques in Foundation Creek, El Paso County, Colorado. *Environmental Engineering*, *138*(6), pp.654-662.

Bashivan, P., Fatehi, A. and Peymani, E., 2008, December. Multiple-model control of pH neutralization plant using the SOM neural networks. In *2008 Annual IEEE India Conference* (Vol. 1, pp. 115-119). IEEE.

Bhandare, D.S. and Kulkarni, N.R., 2015, October. Performances evaluation and comparison of PID controller and fuzzy logic controller for process liquid level control. In *2015 15th International Conference on Control, Automation and Systems (ICCAS)* (pp. 1347-1352). IEEE.

Bhat, N. and McAvoy, T. J., 1990, May. Use of neural nets for dynamic modelling and control of chemical process systems. *Computers and Chemical Engineering*, 14(4/5), pp. 573–582.

Bingi, K., Ibrahim, R., Karsiti, M.N., Chung, T.D. and Hassan, S.M., 2016, September. Optimal PID control of pH neutralization plant. In *2016 2nd IEEE International Symposium on Robotics and Manufacturing Automation (ROMA)* (pp. 1-6). IEEE.

Boobalan, S., Prabhu, K. and Bhaskaran, V.M., 2013. Fuzzy based temperature controller for continuous stirred tank reactor. *Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *2*(12), pp.5835-5842.

Chen, J. and Huang, T.C., 2004. March. Applying neural networks to on-line updated PID controllers for nonlinear process control. *Journal of Process Control*, Vol. 14. pp. 211-230.

Chien, I. L. and Fruehauf, P. S., 1990. Consider IMC tuning to improve controller performance, *Chem. Eng. Progress*, 86(10), pp. 33-41.

Darab, C., Hodrea, R., Crisan, R. and Nascu, I., 2012. Modeling and internal model control strategy of pH neutralization process. *20th Telecommunications Forum (TELFOR)*. pp.1579-1582). IEEE.

Degachi, H., Chagra, W. and Ksouri, M., 2018. Nonlinear model predictive control for pH neutralization process based on SOMA algorithm. *International Journal of Advanced Computer Science and Applications*, *9*(1), pp.391-398.

Desai, K.M., Survase, S.A., Saudagar, P.S., Lele, S.S. and Singhal, R.S., 2008. Comparison of artificial neural network (ANN) and response surface methodology (RSM) in fermentation media optimization: case study of fermentative production of scleroglucan. *Biochemical Engineering Journal*, *41*(3), pp.266-273.

Doherty, S.K., Gomm, J.B. and Williams, D., 1997, November. Experiment design considerations for non-linear system identification using neural networks. *Computers & chemical engineering*, *21*(3), pp.327-346.

Du, X., Wang, J., Jegatheesan, V. and Shi, G., 2018. Dissolved oxygen control in activated sludge process using a neural network-based adaptive PID algorithm. *Applied Sciences.* Vol. *8*, pp. 1-21.

Fadali, M.S. and Visioli, A., 2012. *Digital control engineering: analysis and design*. Academic Press. pp. 213-214.

Findeisen, R. and Allgöwer, F., 2002, March. An introduction to nonlinear model predictive control. In *21st Benelux meeting on systems and control* (Vol. 11, pp. 119-141). Eindhoven, The Netherlands: Technische Universiteit Eindhoven Veldhoven.

Fink, A. and Nelles, O., 2001. Nonlinear internal model control based on local linear neural networks. In *2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace (Cat. No. 01CH37236).*Vol. 1, pp. 117-122. IEEE.

Foss, B.A., Johansen, T.A. and Sørensen, A.V., 1995, March. Nonlinear predictive control using local models—applied to a batch fermentation process. *Control Engineering Practice*, *3*(3), pp.389-396.

Franklin, G.F., Powell, J.D. and Workman, M.L., 1998. *Digital control of dynamic systems*. Third Edition. pp. 24. Menlo Park, CA: Addison-wesley.

Gan, M., Peng, H., Peng, X., Chen, X., and Inoussa, G., 2010, November. .A locally linear RBF network-based state-dependent AR model for nonlinear time series modelling. *Journal of Information Science*, Vol. 180, pp. 4370-4383.

Gao, R., O'Dwyer, A. and Coyle., E., 2002. Model predictive control of CSTR based on local model networks. *Proceedings of the Irish Signals and Systems Conference*, pp. 397-402, University College Cork.

Garcia, C. E. and Morari, M., 1982. Internal model control-1: A unifying review and some new results. *Industrial & Engineering Chemistry Process Design and Development.* Vol. 21, no. 2, pp. 308-323.

Garcia, C. E. and Morari, M., 1985, April. Internal model control. 2. Design procedure for multivariable systems. *Industrial & Engineering Chemistry Process Design and Development* .24, pp. 472-484

Gaya, M.S., Wahab, N.A., Sam, Y.M. and Samsuddin, S.I., 2013, March. ANFIS inverse control of dissolved oxygen in an activated sludge process. *IEEE 9th International Colloquium on Signal Processing and its Applications.* pp. 146-150.

Ghavipanjeh, F. 2006. Multivariable modelling and control of a wastewater benchmark system by PIP control design. *Proceedings of the International UK Automatic Control Conference*, Glasgow, Scotland, United Kingdom.

Goldar, A., Revollar, S. R., Lamanna, R. and Vega, P., 2016, January. Neural NLMPC schemes for the control of the activated sludge process. *In Proceedings of the 11th IFAC Symposium on Dynamics and Control of Process Systems Including Biosystems,* Vol. 49, pp. 913-918.Trondheim, Norway.

Gomm., J. B. Doherty, S.K. and Williams, D., 1996. Control of pH in-line using a neural predictive strategy. *UKACC International Conference on Control*, pp. 1058 - 1063.

Gomm, J. B., Weerasinghe, M. and Williams, D., 2000**.** Diagnosis of process faults with neural networks and principal component analysis. *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, *214*(2), pp.131-143.

Gomm., J. B., Evans., J. T. and Williams., D., 1997, January. Development and performance of a neural-network predictive controller. *Control Engineering Practice*, Vol. 5, No. 1, pp. 49-59.

Gogoria, S., Parhar, T. and Pandian, J.B., 2015. Multi-Model Adaptive Fuzzy Controller for a CSTR Process. *Sensors & Transducers*, *192*(9), pp.96.

Gu, B. and Gupta, Y. P., 2008. Control of nonlinear processes by using linear model predictive control algorithms. *ISA Transactions*, Vol. 47, pp. 211-216.

Hamed, M.M., Khalafallah, M.G. and Hassanien, E.A., 2004. Prediction of wastewater treatment plant performance using artificial neural networks. *Environmental Modelling & Software*, *19*(10), pp.919-928.

Hametner, C. and Jakubek, S., 2010. Nonlinear system identification through local model approaches: Partitioning strategies and parameter estimation. In *Modelling, Simulation and Identification*. IntechOpen.

Hametner, C., and Jakubek, S., 2013, January. Local model network identification for online engine modelling. *Journal of Infromation Science,* Vol. 220, pp. 210-225.

Hermansson, A. W. and Syafiie, S., 2014, December. Control of pH neutralization system using nonlinear model predictive control with I-controller. *International conference on industrial engineering and engineering management*. pp. 853-857.

Himmelblau, D. M., 2000. Applications of artificial neural networks in chemical engineering. *Korean journal of chemical engineering*, *17*(4), pp.373-392.

Himmelblau, D. M., 2008. Accounts of experiences in the application of artificial neural networks in chemical engineering. *Industrial & Engineering Chemistry Research*, *47*(16), pp.5782-5796.

Hitit, Z.Y., Ozyurt, B. and Ertunc, S., 2017. The Application of System Identification and Advanced Process Control to Improve Fermentation Process of Baker's Yeast. *Yeast: Industrial Applications*, p.153.

Holenda, B., Domokos, E., Redey, A. and Fazakas, J., 2008. Dissolved oxygen control of the activated sludge wastewater treatment process using model predictive control. *Computers & Chemical Engineering*, *32*(6), pp.1270-1278.

Hunt, K.J., Sbarbaro, D., Żbikowski, R. and Gawthrop, P.J., 1992. Neural networks for control systems—a survey. *Automatica*, *28*(6), pp.1083-1112.

Jamali, B. and Jazayeri-Rad, H., 2010, November. Application of adaptive local linear model tree for nonlinear identification of heat recovery steam generator system based on experimental data. In *2010 Fourth UKSim European Symposium on Computer Modeling and Simulation*, pp. 16-20. IEEE.

Janert, P.K., 2013. *Feedback control for computer systems: introducing control theory to enterprise programmers*. " O'Reilly Media, Inc.".

Jang., J.S., 1993. ANFIS: adaptive network based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics*, *23*(3), pp.665-685.

Johansen., T.A. and Foss., B. A.,1992a. A NARMAX model representation for adaptive control based on local models. *Modelling, Identification, and Control*, Vol.13, pp. 25-39.

Johansen, T. A and Foss, B. A., 1992 b, February. Nonlinear local model representation for adaptive systems. *International Conference on Intelligent Control and Instrumentation,* Vol. 2, pp. 677-682.

Johansen, T.A. and Foss, B., 1993. Constructing NARMAX models using ARMAX models. *International Journal of Control*, *58*(5), pp.1125-1153.

Johansen., T.A. and Foss., B. A., 1995, February. Identification of nonlinear system structure and parameters using regime decomposition. Automatica, Vol. 31, No. 2, pp. 321-326.

Karayiannis, N.B., 1997, June. Gradient descent learning of radial basis neural networks. In *Proceedings of International Conference on Neural Networks (ICNN'97).* Vol. 3, pp. 1815-1820. IEEE.

Karayiannis, N.B., 1999. Reformulated radial basis neural networks trained by gradient descent. *IEEE transactions on neural networks*, *10*(3), pp.657-671.

Khare, Y. B. and Singh., Y,. 2010, October. PID control of heat exchanger system. *International Journal of Computer Applications (0975 – 8887).* Vol 8– No.6.pp. 22-27.

Kharaajoo, M.J., Rahmati, A. and Rashidi, F., 2003. Internal model control based on locally linear model tree (LOLIMOT) model with application to a PH neutralization process. In *IEEE. International Conference On Systems Man and Cybernetics.* Vol. 4, pp. 3051-3055.

Kittisupakorn, P., Thitiyasook, P., Hussain, M.A. and Daosud, W., 2009. Neural network based model predictive control for a steel pickling process. *Journal of Process Control*, *19*(4), pp.579-590.

Liang, W. and Wang, J., 2014, July. Application of internal model control in main steam temperature system. In *Proceedings of the 33rd Chinese Control Conference* (pp. 3511-3514). IEEE.

Liu, X., Liu, Y., Gong, X. and Zhang, H., 2009, April. Design of intelligent dissolved oxygen detecting system based on CAN bus and embedded USB host. In *2009 International Conference on Measuring Technology and Mechatronics Automation*. Vol. 1, pp. 85-88.

Luyben, W.L., 2007. *Chemical reactor design and control*. John Wiley & Sons. pp.2

Mahmoodi, S., Poshtan, J., Jahed-Motlagh, M.R. and Montazeri, A., 2009. Nonlinear model predictive control of a pH neutralization process based on Wiener–Laguerre model. *Chemical Engineering Journal*, *146*(3), pp.328-337.

Marzooghi, H., Raoofat, M., Dehghani, M., and Elahi, G. H., 2012, March. Dynamic modelling of solid oxide fuel cell stack based on local linear model tree algorithm. *International Journal of Hydrogen Energy*, Vol. 37, pp. 4367-4376.

McAvoy, T. J., Hsu, E., and Lowenthal, S., 1972, January. Dynamics of pH in controlled stirred tank reactor. *Industrial & Engineering Chemistry Process Design and Development.* Vol. 11, pp. 68–70.

McCulloch, W.S. and Pitts, W., 1943. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, Vol. 5. pp. 115.133.

Meng, Y., Zhiyun, Z., Fujian, R., Yusong, P. and Xijie, G., 2014, June. Application of adaptive PID based on RBF neural networks in temperature control. In *Intelligent Control and Automation (WCICA), 2014 11th World Congress on* (pp. 4302-4306). IEEE.

Mugisha, J. C., Munyazikwiye, B. and Karimi, H. R., 2015, November. Design of temperature control system using conventional PID and intelligent fuzzy logic controller. *Proceedings of 2015 International Conference on Fuzzy Theory and Its Applications (iFUZZY).*pp. 50-55.

Murray-Smith, R. and Johansen, T., 1997. *Multiple model approaches to modelling and control*. Taylor and Francis, London, UK.

Murray-Smith, R., 1994. Local model networks and local learning, in fuzzy duisburg, pp. 404-409.

Mwembeshi, M. M., Kent, C. A., and Salhi, S., 2004, August. A genetic algorithm based approach to intelligent modelling and control of pH in reactors. *Computers and Chemical Engineering*, 28(9), pp. 1743–1757.

Nagy, Z. K., 2007. Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks. *Journal of Chemical Engineering*, Vol. 127, pp. 95-109.

Narendra, K. S. and Parthasarathy, K.. 1990, March. Identification and control of dynamic systems using neural networks. *IEEE Transactions on Neural Networks*, Vol.1, pp. 4–27.

Nejjari, F., Benhammou, A., Dahhou, B. and Roux, G., 1997. On line estimation and multivariable predictive control of an activated sludge process. *IFAC Proceedings Volumes*, *30*(27), pp.207-212.

Nejjari, F., Dahhou, B., Benhammou, A. and Roux, G., 1999, August. Non-linear multivariable Adaptive control of an activated sludge wastewater treatment process, *Journal of Adaptive Control and Signal Processing*, Vol. 13, pp. 347-365.

Nelles, O., 1997, July. Orthonormal basis functions for nonlinear system identification with local linear model trees (LOLIMOT). *IFAC Proceedings Volumes*, *30*(11), pp.639-644.

Nelles, O., 2002. *Nonlinear system identification*. 1st edn, Springer Verlag.pp. 61

Nelles, O., Fink., A. and Isermann., R., 2000. Local linear model trees (LOLIMOT) toolbox for nonlinear system identification. *IFAC Proceedings Volumes*. Vol, 33. Pp. 845-850. Santa Barbara, USA.

Nelles, O. and Tomizuka, M., 2000, June. On the dynamics of local linear model networks with orthonormal basis functions. *IFAC Proceedings.* Vol. 33(15), pp.55-60.

Nelles, O., Sinsel, S. and Isermann, R., 1996, September. Local basis function networks for identification of a turbocharger, *UKACC International Conference on Control,* pp.7-12.

Nelles, O and Isermann, R., 1996, December. Basis function networks for interpolation of local linear models. *Proceedings of 35th IEEE Conference on Decision and Control*. Kobe, Japan, Vol. 1 pp. 470–475.

Nentwing, M. and Mercorelli, P., 2010, March. Matlab/Simulink toolbox for inversion of local linear model trees. *International Journal of Computer Science*, Vol. 37, pp. 19-26.

Norquay, S.J., Palazoglu, A. and Romagnoli, J.A., 1999. Application of Wiener model predictive control (WMPC) to a pH neutralization experiment. *IEEE Transactions on Control Systems Technology*, *7*(4), pp.437-445.

O'Dwyer, A., 2006. *Handbook of PI and PID controller tuning rules*. World Scientific

Ogata, K., 2003. *Modern Control Engineering* : Prentice Hall.

Patil, A. B. and Salunkhe, A. V., 2008. December. Adaptive neuro fuzzy controller for process control system. *The Third international Conference on Industrial and Information Systems,* Kharagpur, India, 8–10, pp. 1–5.

Pedram, A., Jamali, M.R., Pedram, T., Fakhraie, S.M. and Lucas, C., 2006, May. Local linear model tree (LOLIMOT) reconfigurable parallel hardware. In *International Symposium on Parallel Computing in Electrical Engineering*, Vol. 13, No. 17, pp. 198-201.

Petchinathan, G., Valarmathi, K., Devaraj, D. and Radhakrishnan, T.K., 2014, June. Local linear model tree and Neuro-Fuzzy system for modelling and control of an experimental pH neutralization process. *Brazilian Journal of Chemical Engineering*, *31*(2), pp.483-495.

Poongodi, P. and Sudhanan, R.M., 2015. Simulation of temperature control methodologies for chemical reactor. *Journal of Chemical and Pharmaceutical Research*, *7*(9), pp.682-689.

Prakash, J. and Senthil, R., 2008. Design of observer based nonlinear model predictive controller for a continuous stirred tank reactor. *Journal of Process control*, *18*(5), pp.504-514.

Rattanawaorahirunkul, R., Sanposh, P. and Panjapornpon, C., 2016, January. Nonlinear system identification of pH process using Hammerstein-Wiener model. In *2016 International Conference on Electronics, Information, and Communications (ICEIC)* (pp. 1-4). IEEE.

Regunath, S. and Kadirkamanathan, V. A., 2001, June. Design of a pH control system using fuzzy non-uniform grid scheduling and evolutionary programming. *Journal of Applied Soft Computing,* Vol. 1, pp. 91-104.

Sapuan, S.M. and Mujtaba, I.M. eds., 2009. *Composite materials technology: neural network applications*. CRC Press.

Seborg, D. E., Edgar,T. F. and Mellichamp, D. A., 2004. *Process dynamics and control*, Second Edition.

Shen, W., Guo, X., Wu, C. and Wu, D., 2011. Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. *Knowledge-Based Systems*, *24*(3), pp.378-385.

Shen, W., Zhu, Y. and Long, Z., 2016, July. Two PID-combined controllers for dissolved oxygen concentration in wastewater treatment process of a paper mill-a simulation study. In *2016 35th Chinese Control Conference (CCC)* (pp. 8533-8538). IEEE.

Shorten, R., and Murray-Smith, R., 1994, September. On normalising radial basis function networks. In *Proceedings of the fourth Irish Conference on Neural Networks, INNC*,Vol. 94, pp. 213-217.

Singh, P.K., Bhanot, S., Mohanta, H.K. and Bansal, V., 2015, September. Self-tuned fuzzy logic control of a pH neutralization process. In *2015 21st International Conference on Automation and Computing (ICAC)* (pp. 1-6). IEEE.

Sivakumaran, N., Radhakrishnan, T. K., and Babu, J.S.C., 2006, December. Identification and control of bioreactor using recurrent networks. *Journal of Instrumentation Science and Technology*, Vol. 34, pp. 635–651.

Steinkraus, K. H. 1998. *Bio-enrichment: production of vitamins in fermented foods*. In: Wood, B.J.B. (Ed.), Microbiology of Fermented Foods, 2nd Edition. Blackie Academic and Professional, London, pp. 603–621.

Sundararajan., N., Saratchandran., P. and Lu, Y. W . 1999. *Radial basis function neural networks with sequential learning. MRAN and Its Applications*. Vol. 11. Singapore: World Scientific.

Sung, S.W., Lee, I.B. and Yang, D.R., 1995. pH control using an identification reactor. *Industrial & engineering chemistry research*, *34*(7), pp.2418-2426.

Talpur, N., Salleh, M.N.M. and Hussain, K., 2017, August. An investigation of membership functions on performance of anfis for solving classification problems. In *IOP Conference Series: Materials Science and Engineering,* Vol. 226, No. 1,pp. 1-7. IOP Publishing.

te Braake, H.A., Van Can, E.J., Scherpen, J.M. and Verbruggen, H.B., 1998, July. Control of nonlinear chemical processes using neural models and feedback linearization. *Computer and Chemical Engineering*, Vol. 22, No. 7-8, pp. 1113-1127.

Tharakan, L. G., Benny, A., Jaffar, N. E. and Abdul Jaleel, J. A., 2013, March. Neural network based pH control of a weak acid-Strong base system. *International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s),* pp. 674-679.

The MathWorks, 2013. Adaptive neuro fuzzy inference systems. Available on line at: http://uk.mathworks.com/help/fuzzy/neuro-adaptive-learning-and-anfis.html

Vahdani, B., Iranmanesh, S. H., Mousavi, S. M. and Abdollahzade, M., 2012, October. A locally linear neuro-fuzzy model for supplier selection in cosmetics industry. *Journal of Applied Mathematical Modelling*. Vol. 36, pp. 4714–4727.

Valarmathi, K., Devaraj, D. and Radhakrishnan, T.K., 2009. Real-coded genetic algorithm for system identification and controller tuning. *Applied Mathematical Modelling*, *33*(8), pp.3392-3401.

Vukic, Z., Kuljace. L., Donlagic. D. and Tesnjak. S. 2003. February. *Nonlinear control systems*. United State of America: Marcel Dekker, pp.1-3.

Wang, Z. and Yuan, M., 2012, August. A self-tuning fuzzy PID control method of grate cooler pressure based on Kalman filter. In *2012 International Conference on Computer Science and Information Processing (CSIP)* (pp. 257-260). IEEE.

Williams, G. L., Rhinehart, R. R. and Riggs, J.B., 1990, July. In-line process-model-based control of wastewater pH using dual base injection. *Industrial and Engineering Chemistry Research,* American Chemical Society, Vol. 29, pp.1254-1259.

Xie, W. F. and Rad, A. B., 1999. November. Fuzzy on-line identification of SISO nonlinear systems. *Fuzzy Sets and Systems*, Vol. 107, pp.323-334.

Yang, Y. and Wu, Q., 2016, July. A neural network PID control for pH neutralization process. *Proceedings of the 35th Chinese Control Conference,* pp. 3480-3483, Chengdu, China.

Yoo, A., Lee, T. C. and Yang, D. R., 2004. July. Experimental simultaneous state and parameter identification of a pH neutralization process based on an Extended Kalman Filter. *Korean Journal of Chemical Engineering*, Vol. 21, no. 4, pp. 753-760.

Yu, D. W. and Yu, D. L., 2005. October. Modelling a multivariable reactor and on-line model predictive control. *ISA Transactions*, Vol. 44, pp. 539-365.

Yu, D.W. and Yu, D.L., 2003. Neural network control of multivariable processes with a fast optimisation algorithm. *Neural Computing & Applications*, *12*(3-4), pp.185-189.

Yu, D.W. and Yu, D.L., 2007. Multi-rate model predictive control of a chemical reactor based on three neural models. *Biochemical Engineering Journal*, *37*(1), pp.86-97.

Yu, D. L. and Gomm, J. B., 2003. November. Implementation of neural network predictive control to a multivariable chemical reactor. *Control Engineering Practice*, Vol. 11, pp. 1315-1323.

Yu, D.L., Gomm, J.B. and Williams, D., 1999. July. On-line predictive control of a chemical process using neural network models. *IFAC Proceedings Volumes*, *32*(2), pp.4347-4352.

Yu, E. and Hu, Y., 2016, May. A novel modified PID controller applied to temperature control with self-tuning ability. *In Chinese Control and Decision Conference*, pp. 7025–7029, Yinchuan, China.

Zhang, J. and Morris, J., 2001. Nonlinear model predictive control based on multiple local linear models. In *Proceedings of the 2001 American Control Conference,* Vol. 5, pp. 3503-3508.

Zhiyun, Z.O.U., Meng, Y.U., Zhizhen, W.A.N.G., Xinghong, L.I.U., Yuqing, G.U.O., ZHANG, F. and Ning, G.U.O., 2013. Nonlinear model algorithmic control of a pH neutralization process. *Chinese Journal of Chemical Engineering*, *21*(4), pp.395-400.

Zhu, G. Y., Zamamiri, A. M., Henson, M. A., and Hjortsϕ, M. A., 2000, December. Model predictive control of continuous yeast bioreactors using cell population balance models. *Chemical Engineering Science*, Vol. 55, pp. 6155-6167.

Appendix A

Published work

Abdelhadi, A., Gomm, J. B., DingLi, Yu. D. L. and Rajarathinam, K. 2014. Nonlinear system identification and control pf a pH process using local linear model networks strategy. *Proceedings of the 20th International Conference on Automation & Computing*, Cranfield University, Bedfordshire, UK.

Abdelhadi, A., Gomm, J. B., DingLi, Yu. and Rajarathinam, K. 2014 . Comparison of RBF and local linear model networks for nonlinear identification of a pH process. *UKACC International Conferenc*e. pp. 361-366.

Appendix B

This table illustrates the PID controller sitting using internal model control which were used for controller design in chapter 3.

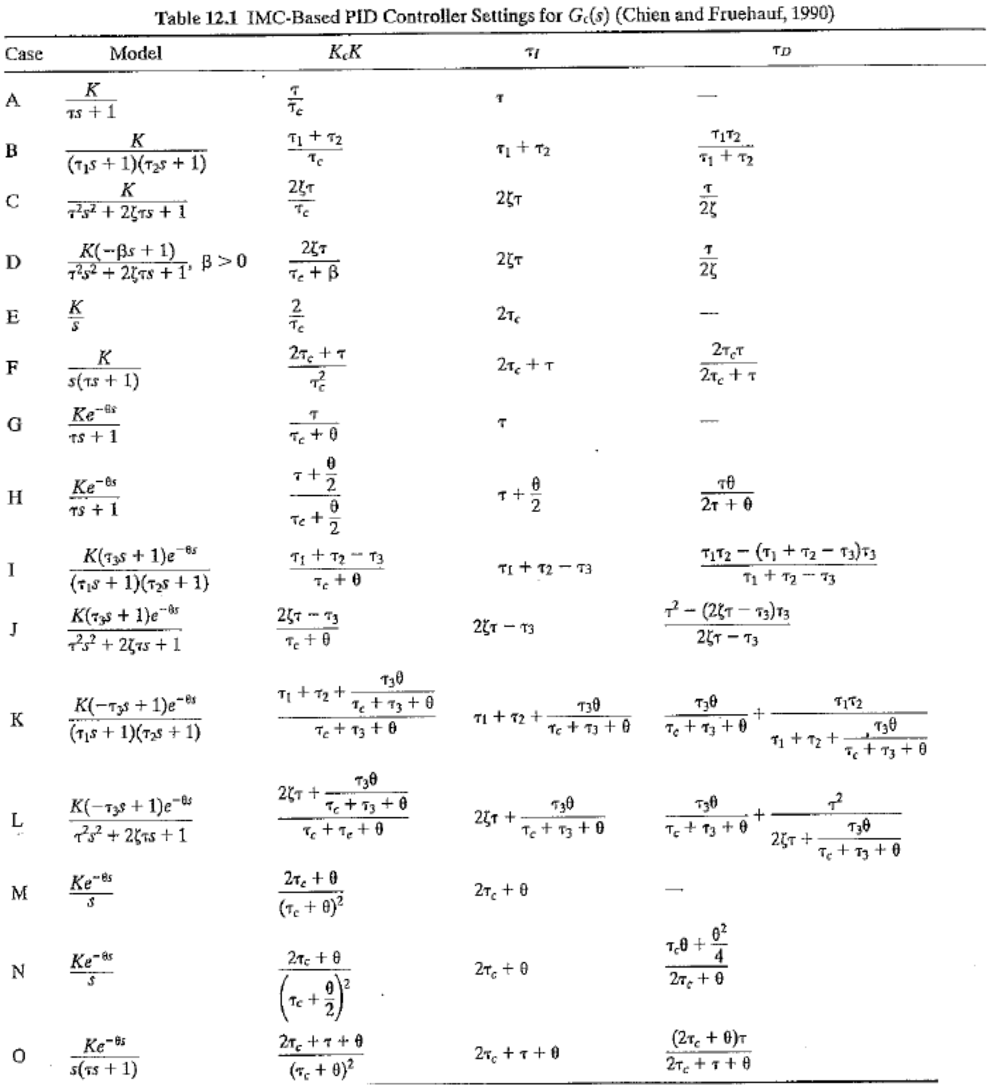


Table 12.1 IMC-Based PID Controller Settings for $G_c(s)$ (Chien and Fruehauf, 1990)

Case	Model	K_cK	τ_I	τ_D
A	$\frac{K}{\tau s + 1}$	$\frac{\tau}{\tau_c}$	τ	—
B	$\frac{K}{(\tau_1 s + 1)(\tau_2 s + 1)}$	$\frac{\tau_1 + \tau_2}{\tau_c}$	$\tau_1 + \tau_2$	$\frac{\tau_1 \tau_2}{\tau_1 + \tau_2}$
C	$\frac{K}{\tau^2 s^2 + 2\zeta\tau s + 1}$	$\frac{2\zeta\tau}{\tau_c}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$
D	$\frac{K(-\beta s + 1)}{\tau^2 s^2 + 2\zeta\tau s + 1}, \beta > 0$	$\frac{2\zeta\tau}{\tau_c + \beta}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$
E	$\frac{K}{s}$	$\frac{2}{\tau_c}$	$2\tau_c$	—
F	$\frac{K}{s(\tau s + 1)}$	$\frac{2\tau_c + \tau}{\tau_c^2}$	$2\tau_c + \tau$	$\frac{2\tau_c \tau}{2\tau_c + \tau}$
G	$\frac{Ke^{-\theta s}}{\tau s + 1}$	$\frac{\tau}{\tau_c + \theta}$	τ	—
H	$\frac{Ke^{-\theta s}}{\tau s + 1}$	$\frac{\tau + \frac{\theta}{2}}{\tau_c + \frac{\theta}{2}}$	$\tau + \frac{\theta}{2}$	$\frac{\tau\theta}{2\tau + \theta}$
I	$\frac{K(\tau_3 s + 1)e^{-\theta s}}{(\tau_1 s + 1)(\tau_2 s + 1)}$	$\frac{\tau_1 + \tau_2 - \tau_3}{\tau_c + \theta}$	$\tau_1 + \tau_2 - \tau_3$	$\frac{\tau_1\tau_2 - (\tau_1 + \tau_2 - \tau_3)\tau_3}{\tau_1 + \tau_2 - \tau_3}$
J	$\frac{K(\tau_3 s + 1)e^{-\theta s}}{\tau^2 s^2 + 2\zeta\tau s + 1}$	$\frac{2\zeta\tau - \tau_3}{\tau_c + \theta}$	$2\zeta\tau - \tau_3$	$\frac{\tau^2 - (2\zeta\tau - \tau_3)\tau_3}{2\zeta\tau - \tau_3}$
K	$\frac{K(-\tau_3 s + 1)e^{-\theta s}}{(\tau_1 s + 1)(\tau_2 s + 1)}$	$\frac{\tau_1 + \tau_2 + \frac{\tau_3\theta}{\tau_c + \tau_3 + \theta}}{\tau_c + \tau_3 + \theta}$	$\tau_1 + \tau_2 + \frac{\tau_3\theta}{\tau_c + \tau_3 + \theta}$	$\frac{\tau_3\theta}{\tau_c + \tau_3 + \theta} + \frac{\tau_1\tau_2}{\tau_1 + \tau_2 + \frac{\tau_3\theta}{\tau_c + \tau_3 + \theta}}$
L	$\frac{K(-\tau_3 s + 1)e^{-\theta s}}{\tau^2 s^2 + 2\zeta\tau s + 1}$	$\frac{2\zeta\tau + \frac{\tau_3\theta}{\tau_c + \tau_3 + \theta}}{\tau_c + \tau_e + \theta}$	$2\zeta\tau + \frac{\tau_3\theta}{\tau_c + \tau_3 + \theta}$	$\frac{\tau_3\theta}{\tau_c + \tau_3 + \theta} + \frac{\tau^2}{2\zeta\tau + \frac{\tau_3\theta}{\tau_c + \tau_3 + \theta}}$
M	$\frac{Ke^{-\theta s}}{s}$	$\frac{2\tau_c + \theta}{(\tau_c + \theta)^2}$	$2\tau_c + \theta$	—
N	$\frac{Ke^{-\theta s}}{s}$	$\frac{2\tau_c + \theta}{\left(\tau_c + \frac{\theta}{2}\right)^2}$	$2\tau_c + \theta$	$\frac{\tau_c\theta + \frac{\theta^2}{4}}{2\tau_c + \theta}$
O	$\frac{Ke^{-\theta s}}{s(\tau s + 1)}$	$\frac{2\tau_c + \tau + \theta}{(\tau_c + \theta)^2}$	$2\tau_c + \tau + \theta$	$\frac{(2\tau_c + \theta)\tau}{2\tau_c + \tau + \theta}$

Appendix C

The steps below are used to design and evaluate ANFIS model for system identification of pH process as discussed in chapter3.

```
trnData = [x Tar];
numMFs = 5;%number of membership functions
mfType = 'gbellmf';
epoch_n = 20;
in_fis = genfis1(trnData,numMFs,mfType);%use genfis1 to generate initial
membership functions.
out_fis = anfis(trnData,in_fis,20);
yan=evalfis(x,out_fis);
plot([Tar yan])
```