FORMULATION OF MODEL PREDICTIVE CONTROL ALGORITHM FOR NONLINEAR PROCESSES

(FORMULASI ALGORITMA KAWALAN RAMALAN MODEL UNTUK PROSES TIDAK LINEAR)

BOO CHIN ENG
HONG MEI KWEE
AMY TAN SIEW WERN
KHAIRIYAH MOHD YUSOF

RESEARCH VOTE NO:
71992

Jabatan Kejuruteraan Kimia
Fakulti Kejuruteraan Kimia & Kejuruteraan Sumber Asli
Universiti Teknologi Malaysia

2006

UNIVERSITI TEKNOLOGI MALAYSIA

BORANG PENGESAHAN
LAPORAN AKHIR PENYELIDIKAN

TAJUK PROJEK : **FORMULATION OF MODEL PREDICTIVE CONTROL ALGORITHM FOR NONLINEAR PROCESSES**

Saya **KHAIRIYAH MOHD YUSOF**
(HURUF BESAR)

Mengaku membenarkan **Laporan Akhir Penyelidikan** ini disimpan di Perpustakaan Universiti Teknologi Malaysia dengan syarat-syarat kegunaan seperti berikut :

1. Laporan Akhir Penyelidikan ini adalah hakmilik Universiti Teknologi Malaysia.
2. Perpustakaan Universiti Teknologi Malaysia dibenarkan membuat salinan untuk tujuan rujukan sahaja.
3. Perpustakaan dibenarkan membuat penjualan salinan Laporan Akhir Penyelidikan ini bagi kategori TIDAK TERHAD.
4. * Sila tandakan (/)

[ ]	SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau Kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972).
[ ]	TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh Organisasi/badan di mana penyelidikan dijalankan).
[/]	TIDAK TERHAD	

TANDATANGAN KETUA PENYELIDIK

Nama & Cop Ketua Penyelidik

Tarikh : 26 DISEMBER 2006

CATATAN : ***** *Jika Laporan Akhir Penyelidikan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh laporan ini perlu dikelaskan sebagai SULIT dan TERHAD.*

FORMULATION OF MODEL PREDICTIVE CONTROL ALGORITHM FOR NONLINEAR PROCESSES

(Keywords: Model Predictive Control (MPC), process control, high purity distillation, highly non-linear process)

Process control is essential in any chemical plant. For the past forty years, the conventional PID controller has governed the process control industry. It is the sole selection although many other sophisticated control algorithms have been developed largely because it is able to deliver satisfactory performance for most control problems when properly tuned and installed. However, with faster computing technology, the industry is now demanding a tighter advanced control strategy. To fulfil all these objectives, Model Predictive Control (MPC), an optimal model based control algorithm is definitely the best choice among all the advanced control algorithms available to date. The most significant feature that distinguishes MPC from other control algorithms is its long range prediction concept. MPC will perform the prediction over the future horizon and this will enable current computations to consider future dynamic events and hence allow it to overcome the limitation from the process dead-time, nonminimum phase and slow process dynamic. This research explores the capability of MPC in controlling a highly nonlinear, iterative process. Two case studies are explored. For the first case study, linear MPC is applied on a continuous solution copolymerization reactor with promising results. For the second case study, linear and Nonlinear MPC is applied on a high purity distillation column. This is to determine if there is superiority of one over the other. An unconstrained MIMO DMC and nonlinear MPC (NNMPC) algorithms were developed using a step response model and two feedforward neural networks respectively. Additionally, the comparison between DMC, NNMPC and PI controller based on IAE tuning rules was conducted. Overall, NNMPC control scheme shows a superior performance over the DMC and PI controllers by presenting a smaller overshoot and shorter settling time.

Key researchers :
Assoc. Prof. Dr.Khairiyah Mohd. Yusof (Head)
Boo Chin Eng
Hong Mei Kwee
Amy Tan Siew Wern

E-mail : k-khairi@fkkksa.utm.my
Tel. No. : 07-5535512
Vote No. : 71992

FORMULASI ALGORITMA KAWALAN RAMALAN MODEL UNTUK PROSES TIDAK LINEAR

ABSTRAK

(Keywords: Model Predictive Control (MPC), process control, high purity distillation, highly non-linear process)

Kawalan proses penting di dalam industri kimia. Pengawal PID biasa digunakan di industri kawalan proses selama empat puluh tahun yang lepas. Walaupun banyak algoritma kawalan yang canggih dibina, namun pengawal PID masih biasa digunakan disebabkan prestasi pengawal PID yang baik sekiranya dipasang dengan baik. Walau bagaimanapun, industri kini berminat dengan kawalan termaju yang lebih ketat. Kawalan ramalan modal (MPC) sebagai algoritma kawalan optima berdasarkan modal, adalah pilihan yang terbaik antara algoritma kawalan termaju kini. Keistimewaan MPC adalah dalam konsep ramalan jangka panjang. MPC akan melaksanakan ramalan untuk masa hadapan dan ini membolehkan komputasi masa sekarang untuk mempertimbangkan peristiwa dinamik masa hadapan. Ini akan membolehkan MPC untuk mengatasi batasan masa mati proses, fasa tidak minima and dinamik proses yang perlahan. Kajian ini mendalami keupayaan MPC untuk mengawal proses yang amat tidak linear. Dua kajian kes akan didalami. Dalam kajian kes yang pertama, MPC linear diaplikasi untuk mengawal reaktor 'solution copolymerisation' berterusan dengan keputusan yang baik. Untuk kajian kes yang kedua, MPC linear and MPC tidak linear diaplikasi untuk mengawal 'distillation column' ketulenan tinggi. Ini bertujuan menentukan prestasi kedua-dua algoritma. Algoritma MIMO DMC dan MPC tidak linear dibangunkan dengan menggunakan model 'step response' dan dua buah 'neural network' suap balik. Perbandingan antara pengawal DMC, NNMPC dan PI berdasarkan IAE dilaksanakan. Secara keseluruhannya, kawalan NNMPC menunjukkan prestasi yang cemerlang dibandingkan dengan pengawal DMC dan PI dengan 'overshoot' yang kecil dan masa redaman yang kecil.

Key researchers :
Assoc. Prof. Dr.Khairiyah Mohd. Yusof (Head)
Boo Chin Eng
Hong Mei Kwee
Amy Tan Siew Wern

E-mail : k-khairi@fkkksa.utm.my
Tel. No. : 07-5535512
Vote No. : 71992

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF NOMENCLATURE

A	-	Degree of polynomial for transfer function model
B	-	Degree of polynomial for transfer function model, Bottom product
b_1	-	Bias vector
b_2	-	Scalar bias in the output layer
D	-	Additive variable vector
h_i	-	Impulse response coefficients
I	-	Observer gain
K_c	-	Controller gain matrix
F	-	Diagonal matrix in filtering, Feed rate
L	-	Reflux
L_i	-	Liquid flow rate from stage i
m	-	Control horizon, number of manipulated variables
N	-	Model horizon where the response becomes stable, Nonlinearity, number of variables
P	-	Prediction horizon
rj(k)	-	Setpoint for the jth component of controlled variable vector at time k
S_N	-	Step response coefficients
S_u	-	Dynamic matrix in DMC
U	-	Manipulated variable vector
μ(k)	-	input to the system
V	-	Boilup rate
V_i	-	Vapour flow rate from stage i
W_1	-	Weight matrices

W_2	-	Weight matrices
W(t+j)	-	Pre-specified set-point trajectory
x(t)	-	State variable
Y	-	Controlled variable vector
y	-	Observed data
y(k)	-	Output of the system
XD	-	Distillate composition
XB	-	Bottom product composition

Greek Letters

θ	-	Unknown parameter
φ	-	Known regression variables
α	-	Nonnegative scalar in Ridge method, relative volatility
σ	-	Standard deviation for data
σ1	-	Activation function of the hidden layer
σ2	-	Activation function of the output layer
Λ	-	Move suppression coefficients
Γ	-	Penalty for output in DMC control algorithm

CHAPTER 1

INTRODUCTION

1.1 Background

Process control is essential in any chemical plant. For the past forty years, the conventional PID controller has governed the process control industry. It is the sole selection although many other sophisticated control algorithms have been developed. This is largely because the standard PID controller is able to deliver satisfactory performance for most control problem when properly tuned and installed. Additionally, for most of the advanced control algorithms, a fast computation machine is required to execute their complex and time consuming calculation in real time. Consequently this is a big problem in the past as such computers were very costly and not easy to get. However, the computing technology has progressed much. It is now possible for the process engineer to implement many sophisticate control methods with the availability of faster and cheaper computers. There are also many changes in the chemical process industry, with the tougher environment regulations, rigorous safety codes, and rapidly changing economic situation. As a result, the demand and requirement for process control system has become more stringent. The industry is now demanding a tighter advanced control strategy with the ability to integrate all the requirements to reduce operating costs, improve product quality, better use of energy resources and reduce environmental emission.

To fulfill all these objectives, Model Predictive Control (MPC), an optimal model based control algorithm is definitely the best choice among all the advanced control algorithms available to date. MPC refers to a wide class of optimal control

based algorithm that makes use of explicit process model to predict the future behavior of a plant. MPC is "optimal" in the sense that it minimizes the pre-specified objective function. MPC was originally developed in late 70's to meet the specialized control needs of power plants and petroleum refineries. Over the past decade, MPC has established itself as standard control especially in petrochemical and refinery industries (Qin and Bagwell, 1997) largely due to its advantages over traditional controllers (Garcia *et al*., 1989). The most significant feature that distinguishes MPC from other control algorithms is its long range prediction concept. MPC will perform the prediction over the future horizon and this will enable current computations to consider future dynamic events and hence allow it to overcome the limitation from the process dead-time, nonminimum phase and slow process dynamic. In addition, the superior performance of MPC in handling constraints violation in systematic way (through incorporating the constraints directly into objective function) also makes it theoretically a perfect real-time optimal control paradigm equipped with process integration ability (Camacho and Bordons, 1998).

This research explores the capability of MPC in controlling a highly nonlinear, iterative process. Both linear and Nonlinear MPC is applied to determine if there is superiority of one over the after. A high purity distillation column is chosen as the process of interest. The distillation column is chosen because it is a common yet critical unit operation in chemical and petroleum industries. In addition, the distillation column is also very often to be the final separation process which will determines the quality of the product. It is also one of the highest energy consuming unit operations in the chemical industries. The high purity distillation column is selected due to its highly nonlinearity characteristic which has made it as the focus of many research activities (Fruzzetti *et al*., 1997; Georgiou, *et al*., 1988; Kyoung, 1995; Ravi Sriniwas *et al*., 1995).

Theoretically, there are some difficulties to control a high purity column by means of MPC. This is because to most of the popular and commercial MPC packages available today are based on linear models (Qin and Bagwell, 1997), while the high purity column is well-known for its highly nonlinear characteristic. This will definitely limit the performance of Linear Model Predictive Controller (LMPC). A Nonlinear Model Predictive Controller (NMPC), which employs a more accurate

nonlinear model, is also implemented in this thesis to control the high purity column. The NMPC is used with the expectation to provide a better performance compare to LMPC. Neural Network (NNs) is chosen to be the nonlinear process model used in NMPC formulation due to its proven mapping ability as a global approximator (Hornick et *al*., 1989). In addition, the literature review also revealed that it is the most popular nonlinear empirical process model and applications of Neural Network based Model Predictive Control (NNMPC) have been reported (Proll, 1993; Pottmann and Seborg, 1997; Shaw and Doyle III, 1997).

1.2 Model Predictive Control

MPC is not a specific control strategy but a wide class of optimal control based algorithms that use an explicit process model to predict the behavior of a plant. There is a wide variety of MPC algorithms that have been developed over past 30 years. For example, the Model Predictive Heuristic Control (MPHC) algorithm reported by Richalet *et* al. in 1976 which is used an impulse response model as its linear model. In addition, the most industrially popular LMPC algorithm, the Dynamic Matrix Control (DMC) presented by Cutler and Ramaker (1979) , the Generalized Predictive Control (GPC) by Clarke *et al.* (1987) which was intended to provide a new adaptive control alternative and lastly the Internal Model Control (IMC) reported by Garcia and Morari (1982a). The main differences for all these MPC algorithms are the types of models used to represent the plant dynamic and the cost function to be minimized (Soeterboek, 1992).

However, the fundamental framework of MPC algorithms is in common for any kinds of MPC schemes. The basic elements of MPC are illustrated in Figure 1.1 and can be defined as follows:

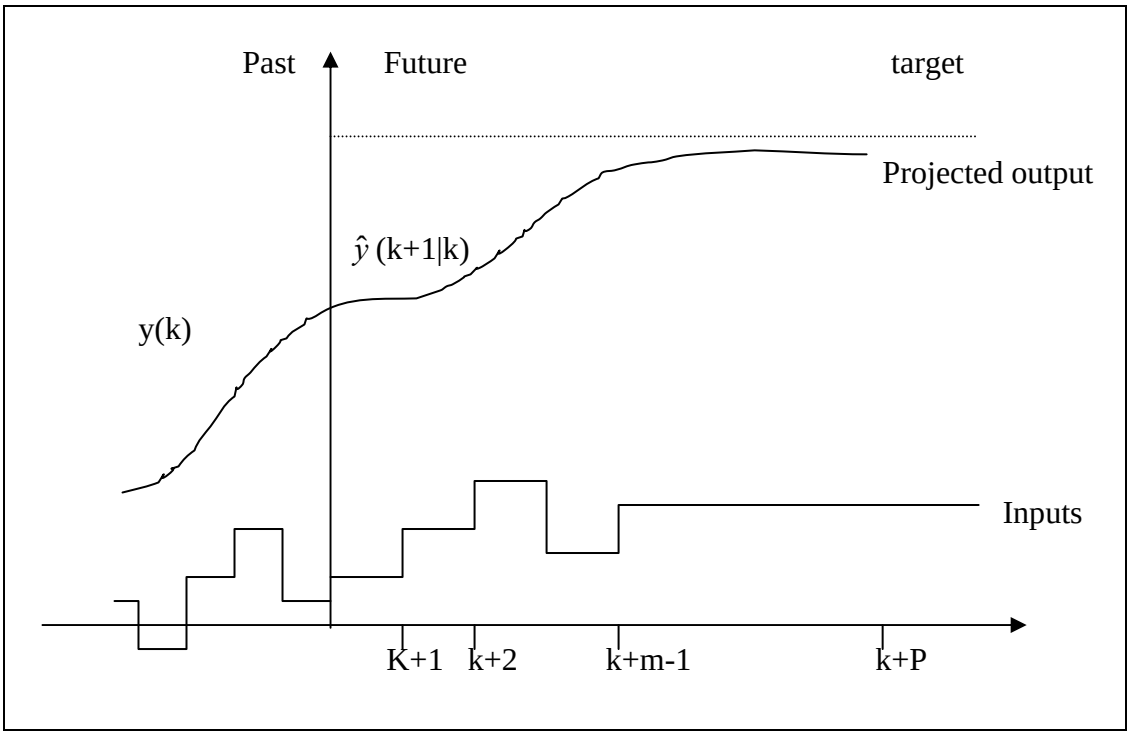

Figure 1.1 MPC strategy

1) An appropriate model is used to predict the output behavior of a plant over a future time interval or normally known as the prediction horizon (P). For a discrete time model this means it predicts the plant output from $\hat{y}(k+1)$ to $\hat{y}(k+H_p)$ based on all actual past control inputs $u(k), u(k-1), ..., u(k-j)$ and the available current information $y(k)$.

2) A sequence of control actions adjustments *(*$\Delta u(k|k\text{-}1) \ldots \Delta u(k+m|k\text{-}1)$*)* to be implemented over a specified future time interval, which is known as the control horizon (m) is calculated by minimizing some specified objectives such as the deviation of predicted output from setpoint over the prediction horizon and the size of control action adjustments in driving the process output to target plus some operating constraints. However, only the first move of computed control action sequence is implemented while the other moves are discarded. The entire process step is repeated at the subsequent sampling time. This theory is known as the receding horizon theory.

3) A nominal MPC is impossible, or in other words that no model can constitute a perfect representation of the real plant. Thus, the prediction error, $\varepsilon(k)$ between the plant measurement $y_m(k)$ and the model prediction $\hat{y}(k)$ will always occur. The $\varepsilon(k)$ obtained is normally used to update the future prediction. The Figure 1.2 illustrated the error feedback of MPC.

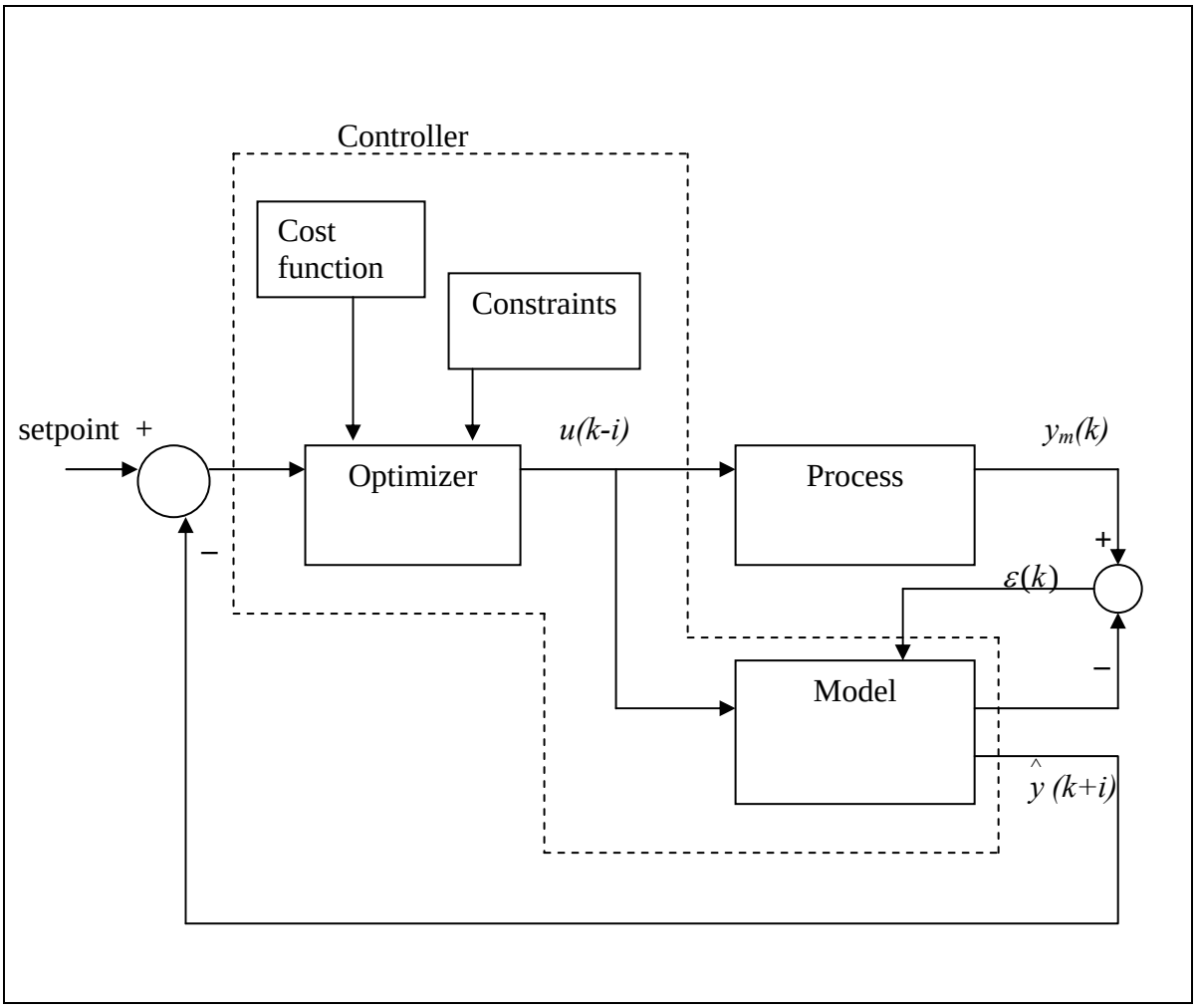

Figure 1.2 The MPC block diagram

1.3 Research Objective

The primary objective of this research is to assess the performance of LMPC and NMPC. In the first case study, LMPC is applied to a solution copolymerization continuous reactor. The second case study explores the application of LMPC and

NMPC in controlling a chosen high purity distillation column both in disturbance rejection and setpoint tracking aspects. The LMPC chosen is DMC, and the NMPC chosen is NNMPC.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The interest in MPC started to surge after the first successfully implementation of MPC algorithm was reported in 1978 with the application of Identification and Command (IDCOM) by Richalet *et al.* (1978). Richalet *et al.* (1978) reported a successfully implementation of Model Predictive Heuristic Control (MPHC) algorithm on a Fluid Catalytic Cracking Unit (FCCU) main fractionator column in a poly-Vinyl Chloride (PVC) plant. However, the thought of ideas for model predictive control (MPC) had started since the 1960's (Garcia *et al.*, 1989). Propoi (1963) had suggested the core of all MPC algorithms, the moving horizon approach in 1963 and Lee and Markus (1967) had anticipated current MPC practice in their optimal control textbook.

After almost 30 years since the first implementation of MPC in industry had been reported, the MPC has now become a standard advanced control technique in many process industries. The application area for MPC has now spread wide and covers not only the petrochemicals and refining fields but also in food processing, automotive, metallurgy, pulp and paper and aerospace and defense industries (Qin and Bagwell, 1997). Qin and Bagwell (1997) showed that at of the end of 1995, there were over 2200 reported applications of using MPC in United State. In Asia, Yamamoto and Hashimoto (1991) showed that MPC were one of the most popular advanced control strategies in industry. They reported that out of 139 Japanese companies in their survey, 25.4% of them have already applied the MPC to their

plants and another 21.1% are considering the possibility. In 1995, Ohshima *et al.* (1995) likewise conducted a survey on the MPC application in Japan and the results showed that number of MPC application has increased steadily in Japan from 1990 to 1995.

MPC became popular for the past 20 years especially in petrochemicals and refining industrial largely due to its superior performance over conventional controllers. One of the advantageous features of this strategy is that it is ideally suited for multivariable control operations where all interaction between manipulated variables (MVs) and control variables (CVs) is taken into account. In conventional controllers, pairing is often done between CVs and MVs and if there is strong interaction, a decoupler is applied. Moreover, MPC is an optimal control strategy that provides constraints handling abilities which had been ignored by most of the conventional controller. As a model based control strategy, the MPC is also able to tackle the long delay time and non-minimum phase problems intrinsically.

However, there are also some limitations or drawbacks associated with MPC, such as the complexity in derivation of control law, the lack of systematic analysis of stability and robustness properties, the lack of clearly defined tuning procedure since the effect of certain parameters upon closed-loop behavior is example dependent (Camacho and Bordons, 1998), and highly dependence on the accuracy of model used. Any discrepancy that arises between the real process and the model will deteriorate the performance severely. Lundstrom *et* al. (1994) reported a few limitations of DMC including it may perform poorly for multivariable plants with strong interaction. On the other hand, Hugo (2000) questioned the benefits that can be brought by MPC and argues that the claims of improved performance boosted by the MPC vendor are often not justified by rigorous statistical analysis such as Harris Performance Index and the benefit actually might be generated by the installation of new instrumentation, change in feedstock or improvement to the regulatory layer before the implementation of MPC. He also presented his ideas about some other practical limitations of MPC such as the difficulties with operation, the high maintenance cost, and lack of flexibility of MPC which may result in a fragile controller.

2.2 Linear Model Predictive Control

Until recently, the MPC is actually a synonym to Linear Model Predictive Control (LMPC). Most of the MPC software available in the market nowadays used linear models even though most processes are nonlinear. For example, the DMC™ from Aspen and HIECON™ from Adersa used the step convolution model and Finite Impulse model respectively (Qin and Bagwell, 1997). Actually, there are several reasons behind this. First, a linear model can be identified in a straightforward manner from the process test data whereas it is very difficult in developing a generic nonlinear model from empirical data. Second, the computational problem in NMPC. Nonlinear Programming (NP) resulting from nonlinear model would make the NMPC's computational problem become very complex, time consuming and sometimes non-convex. On the other hand, the solutions for LMPC algorithm are much easier and sometimes can easily be solved analytically. For example, the solution of DMC algorithm can be done analytically by performing simple least square method. Even for the second generation of DMC, the Quadratic Dynamic Matrix Control (QDMC) algorithm where the problem is the form of Quadratic Program (QP), it is still a very highly structured convex problem where enormous number of reliable solution can be found. For example, many LP and QP problem solution can be found in Fletcher (1987). From the practical point of view, the conventional linear model predictive controller is acceptable in industry because most of the applications of MPC to date are in refinery processing, where the process operates at a single setpoint and the primary use of controllers is for disturbance rejection (Qin and Bagwell, 2000). In this term, a carefully identified linear model is accurate enough because the MPC only have to operate at a certain single operating region.

Even though it has been a long time that since the MPC becomes standard control in industry, the MPC is still being the focus or subject of many researches. Many researches are still carried out with the aims of improving the performance of MPC algorithms or strengthen the weakness of MPC such as the stability and robustness issues. As mention before, traditional LMPC is only able to perform well for the process which characteristic doesn't change significantly (one setpoint operation) along the operating region. However, many chemical processes such as

the polymerization reactor doesn't operate at a single setpoint, and may operate at different setpoint depending on the grade of product to be produced. Hence, the performance of LMPC may severely deteriorate and need to be detuned and redesigned. Thus, some researchers have come out with a new predictive controller strategy which is able to accommodate the changing process conditions and at the same time retain the solution in the LP or QP form for ease of computation. The common solution is to perform an Adaptive Model Predictive Control (AMPC) in which the internal process model is updated as it needed. This can be done in several ways include the multi-model method or by linearizing a suitable nonlinear model near the current operating condition. In general, multiple model approaches is normally done by using a bank of linear models to describe the dynamic behavior over a wide operating region. A recursive Bayesian scheme may be used to assign weights to each linear model and then the combined weighted model is used to do the prediction similar to conventional MPC scheme (Aufderheide and Bequette, 2003). In addition, many literatures about linearization of a nonlinear model for predictive control also have been reported (Garcia, 1984; Gattu and Zafrizou, 1992; Lee and Ricker, 1994). In 1984, Garcia proposed a Nonlinear-QDMC (NQDMC) where the linear model is updated as the state of the process changes and used to obtain the step response model coefficients and the nonlinear model is used to do the prediction. This strategy was able to solve the nonlinearity problem of the process and at the same time retain the controller solution in simple QP form. However, the standard constant output disturbance assumption inherited from DMC algorithm has limited the NQDMC be applied only for open loop stable system.

Analysis of stability and robustness properties cannot be easily performed in MPC algorithm because of the nature of MPC formulation where the implicit definition of MPC control law through a quadratic program with explicitly incorporated input and output constraints. Nevertheless, the MPC stability analysis problem has attracted a considerable number of research activity and a several encouraging results had been reported for the last decade. In fact, closed loop stability for MPC is not easy to ensure because the MPC is a feedback controller as a result of the receding horizon policy. Rawling and Muske (1993) published the first useful result on stability issues in MPC by using the infinite prediction horizon. This theory had proven that an asymptotic stability is able to retain even in the presence of

constraints by imposing infinite horizon and with the criterion where the model used is perfect (nominal stability). However, it is hard to handle constraints with infinite horizon, and hence another way to ensuring stability is to impose a terminal constraint which will force the state to take a particular value at the end of prediction horizon. This method can be performed in many ways such as terminal equality constraint (Meadow *et al*., 1995) or terminal inequality constraints (Polak and Yang, 1993).

Perhaps, the principal shortcoming of existing MPC algorithms is their inability to effectively deal with the difference between the model used and the actual plant. However, compared to the extensive amount of literature on other MPC issues, the number of research activities on the MPC design in the presence of plant-model uncertainty is much less. For the analysis of robustness properties of MPC, Garcia and Morari (1989) reported a robustness analysis of an unconstrained MPC by using a new MPC framework which is later known as Internal Model Control (IMC). They also developed a tuning guideline for IMC filter to ensure robust stability. Zafiriou (1990) have used the contraction properties of MPC to develop necessary / sufficient condition for robust stability of input output constraint. In addition, Polak and Yang (1993) also contributed their idea in robust stability analysis of a continuous time linear system by imposing a contraction constraint on the state. In fact, the conventional way to deal with the plant–model uncertainty or robustness problem is to detune the controller. Nevertheless, in recent years, the subject of the research in this issue is to incorporate the plant-model uncertainty explicitly into the MPC problem formulation. A min-max method (Campo and Morari, 1987) which modifies the online constrained minimization problem to a min-max problem (minimizing the worst case value where worst case is taken over the set of uncertain plant) is a relevant reference in this area.

For further details about MPC, several excellent technical reviews of MPC that provide more detail about MPC formulation and its future exploration direction from the an academic perspective (Garcia *et* al., 1989; Morari and Lee, 1999) and from an industrial perspective (Camacho and Bordon, 1995; Qin and Bagwell, 1997; Qin and Bagwell, 2000; Maciejowski; 2002) are available.

2.2.1 Dynamic Matrix Control

In this research, DMC is chosen as the LMPC algorithm used to control the chosen high purity distillation column because the DMC represent the chemical process industry's standard for MPC. According to a survey by Qin and Bagwell (1997), DMC is the most industrially popular MPC scheme at the end of 1995 in the United State. A large part of DMC's appeal comes from its effectiveness in handling multivariable constrained problem and its theoretical simplicity compare to other MPC schemes. This is because by using Finite Step Response (FSR), the need to specify the model order and dead time has been eliminated. Moreover, the FSR model also can be used to represent multivariable plants easily by superposition of linear models. In addition, the constant output disturbance assumption also has greatly simplified the DMC algorithm. However, the use of FSR model had limited the DMC algorithm only for open-loop stable process and the constant output disturbance assumption has limited DMC's feedback performance.

2.2.2 Application of LMPC on Distillation Columns

There is an abundance of LMPC applications on various kinds of distillation columns in the past 20 years. In a simulated column model application, McDonald and McAvoy (1987) applied a dual point composition control by using a DMC controller to control a moderate-high purity column (a benzene-toulene column with product purities 0.994/0.006). They suggested the use of gain and time constant scheduling method to overcome the nonlinearity problem and also found that it is difficult to obtain a representative process model because the gain and time constants are highly dependent on the size and direction of input step used. Georgiou *et* al. (1988) likewise studied the possibility to control a high purity distillation column by means of DMC. However, they deal with the nonlinearity by performing a logarithmic transformation on the output results. They found that DMC performed well for moderate purity column (10,000 ppm) but for the high purity column (1000 ppm), it displayed a worse performance compared to PI diagonal controller especially when a large load disturbance is imposed. Simple output logarithmic

transformation method improves significantly the performance of DMC for high purity column. For the very high purity column (10 ppm), the transformation method failed to remove the nonlinearity and thus PI diagonal perform a better bad performance compare to DMC. Gokhale *et* al. (1994) likewise applied DMC to a propylene propane splitter (C3 splitter) and compared its performance with the PI controller. However, they did not observe a significant difference in the performance between PI and DMC controller for servo and regulatory problem.

Other than common binary distillation column, the application of DMC can also be found in other types of distillation column. For example, Serra *e*t al. (2001) applied the DMC to a Divided Wall Column (DWC), which was used to separate three different ternary mixtures up to purity 0.99 molar fraction. The result showed that DMC present a longer response time compare to PI controller in both setpoint tracking and disturbance rejection problems. For real industrial applications (pilot scale column), Abou-Jeyab *et al.* (2001) implemented a Simplified Model Predictive Control (SMPC) algorithm, developed by Gupta (1996) to an Imperial Oil Ltd's distillation column. The application significantly improved the column performance. The cycling in the product composition which occurred with the SISO controller was eliminated and there was 2.5% increase in production rate and 0.5 increase in product recovery. Hovd and Michaelsen (1997) likewise applied the D-MPC (*Fantoft* Process Package) on a vacuum distillation column at Nynashamn Refinery of Company Nynas. The use of MPC successfully increased the yield of the most desirable product fraction (110K USD per year) and the feed rate (120k USD) while at the same time decreasing the energy consumption by 20K USD annually.

2.3 Nonlinear Model Predictive Control

Nonlinear Model Predictive Control refers to the MPC algorithm that employs a more accurate nonlinear model in doing prediction and optimization (Henson, 1998). Theoretically, NMPC is able to deliver a better performance compared to LMPC because many chemical processes are highly nonlinear and have strong interactions such as the high-purity distillation column and multi-grade

polymerization reactor. Although the need of NMPC is well recognized and various kinds of NMPC strategy have been developed, whether from academic researchers or commercial companies, LMPC is still much more popular than NMPC in industry (Qin and Bagwell, 1996 & 2000). There were only 86 of NMPC commercial applications that had been reported at the end of 1999 in the United State (Qin and Bagwell, 2000). This is largely due to two main reasons: 1) the difficulty and sometime inability in developing an accurate nonlinear process model and 2) the computational problem associated with the Non-Linear Programming (NLP).

Many kinds of strategies have been proposed over the past 20 years in developing and incorporating a nonlinear model into a MPC algorithm. Overall, the relevant nonlinear modeling methods can be divided into two main groups:

1. Fundamental or first principles modeling method: This method is performed through the analysis of the system at fundamental level such as analysis of system's physical relationships like the conservation laws (mass, energy and momentum), phenomenological laws, state equations and boundary conditions. It is normally in the form of differential and algebraic equations such as the ordinary differential equations (ODE) or partial differential equation (PDE). This kind of model is globally valid due to its natural characteristic, and thus makes it suitable for optimization and control task which often required extrapolation beyond the range of data. However, the derivation of first principles model is normally expensive and difficult to maintain (Piche *et* al., 2000) and often yield a model of very high order due to rigorous modeling (Lee, 1998). Many of NMPC studies based on the fundamental model had been reported within last decade (Patwardhan and Edgar, 1990; Chen and Allgower, 1997; Ricker and Lee, 1995; Zheng, 1997). However, Henson (1998) pointed out that most of them used a very simple dynamic model except Ricker and Lee (1995) that used a model with 10 x 23 (10 MVs and 23 CVs). In NMPC, online solution to NLP or at least nonlinear integration Jacobian matrix calculation is required and hence it is good to keep the model order low. Therefore, order reduction technique such as Orthogonal Collocation method (Patwardhan *et* al. 1992; Proll, 1993; Kawatheka, 2004) is normally applied to ease the computation.

2. Empirical modeling method: This method relies solely on the process data available and requires no understanding of underlying physical phenomena of the system and hence is also known as black-box method. This modeling approach is based on the assumption that the process characteristics are well embedded in the data and can be extracted by appropriate methods and hence the application of this modeling method is limited to the operating region where the model had been identified. In other words, it has unreliable extrapolation capability, which is often required in optimization and control problems. Various kinds of empirical models have been utilized in NMPC design. These include Hammerstein model (Fruzzetti, *et* al., 1997), Volterra model (Maner *et* al., 1996), collocation model (Jang and Wang, 1997) and the most popular one, the neural network model (Asohi, 1995; Doherty, 1999).

Computational problem is another obstacle that precludes successful application of NMPC and a large part of NMPC computational problem is stemmed from the NLP problem. The solution procedure of NMPC basically consists of two main parts: 1) solving optimization problem 2) integrating the system model equation. These two solution procedures can be implemented either sequentially or simultaneously.

1. Sequential solution: In this method, the optimization problem and the differential equation is solved separately. Normally, the optimization algorithm serves as outer the loop to iteratively select new sets of manipulated variables to minimize the objective function, while the ODE solver will be used to integrate the dynamic equations to obtain the controlled variable profile in order to determine the objective function. The availability of accurate and efficient integration and optimization packages largely ease the implementation in this method. However, there are some drawbacks associated with this method. There are difficult to incorporate state/output constraints in this approach and this method requires the solution of differential equation at each iteration of optimization and this has made the implementation very computationally expensive, especially for large system. In addition, the gradients information required for optimization procedure is often obtained through the finite differences based on small changes in the manipulated variables and are commonly done by differencing the output of an integration routine

with adaptive step. However, the integration error is unpredictable and hence differencing output of an integration routine would greatly degrade the quality of the finite difference derivative (Gill *et al*., 1998)

2. Simultaneous Solution: In this approach, the system dynamic (ODE) is reduced to algebraic equations using a weighted residual method (Finlayson, 1980). The algebraic equation is then solved as equality constraints in a nonlinear program. Sequential Quadratic Programming (SQP) is often used to solve the optimization problem for this approach. This solution approach results in more decision variables (the manipulated inputs are the only decision variables in sequential solution) since the values of the state variables at each collocation point are included as decision variables. The advantage of this solution is that the state variable constraints can be handled easily and the fast convergence speed of SQP whereas its disadvantage is that only at the end of the iteration a valid state for the system is available and if the optimization cannot finish in time, nothing can say about the feasibility.

NMPC's computational problem has been an active research topic for the last 20 years and a numbers of alternatives have been developed and reported. One of them is the model order reduction approach. This approach has been proven to reduce the computational burden of numerically integrating differential equations and this helps especially when dealing with a large system. A famous model order reduction example is the Orthogonal Collocation method (Patwardhan *et* al. 1992; Proll, 1994; Kawatheka, 2004). This approach is able to reduce the model order by converting a differential equation in the time domain into an approximating set of algebraic equations. Instead of reducing the model order, some researchers chose to increase the speed of popular local optimization method by tailoring them to take advantage of the specific structure of the MPC formulation. The interior-point approach is a good example for these. This method was successful in easing the computational burden in NMPC and come into favor with many academic researchers lately (Albuquerque *et* al., 1999). In addition, the non-convex problem is also another popular issue in NMPC. Many of them focused on global optimization especially the genetic algorithms (Onnen, *et al*. 1997; Wasif, 2001). However, this approach tends to be too slow and therefore not applicable. Another popular research direction is to simplify the optimization problem in order to decrease the

computation time of each calculation. Zheng (1997) proposed to optimize only the first move of the prediction horizon instead of perform the optimization all along the control horizon. However, this approach does not yield favorable results as it is closely related to a finite horizon optimization with one step ahead prediction. Additionally, Cannon *et* al., (2001) suggested the Triple mode MPC algorithm, which is done by splitting the prediction horizon into three distinct parts to ease the computation.

For further details about NMPC, there are several excellent technical reviews of NMPC that provide more details about NMPC formulation and its future exploration direction from the academic perspective (Henson, 1998; Lee, 1998; Findeisen and Allgower, 2002) and from an industrial perspective (Qin and Bagwell, 2000; Piche *et* al., 2000).

2.3.1 Application of NMPC on Distillation Columns

Unlike the LMPC, the inherent computational and nonlinear modeling problems in NMPC have precluded the popularity of NMPC. However, there are still several literatures on the application of NMPC to control distillation especially the high purity columns, which posses highly nonlinearity and strong interaction. For example, Norquay *et* al. (1999) applied an incorporation of Wiener Model in a unique way into MPC to control a C2-splitter at Orica Olefines plant. The Wiener model actually consisted of a linear dynamic element in series with a static nonlinear element. The results showed that the nonlinearity of the control problem can be removed effectively and at the same time retain the favorable properties of LMPC. However, there is a limitation for this approach as there are processes in which the dominant nonlinearities cannot be separated as a distinct static element.

Brizuele *et* al. (1996) reported an application of Neural Network based Model Predictive Control (NNMPC) to control a Multi-component distillation column. They used the static feedforward neural networks (FNN) to model the distillation column and employed two different single loops to perform the two point

composition control. The results showed that NNMPC displays a significant improvement in performance compared to PI controller in regulatory problem whereas the difference is negligible in setpoint tracking. Moreover, Shaw and Doyle III (1997) proposed an efficient way to model a 2x2 high purity distillation column by using Recurrent Dynamic Neural Network (RDNN). The model was able to capture actual fundamental underlying column dynamics which are second order in nature and hence this model able to account for both external and internal flow in open loop simulation. Similarly the model was able to represent both high and low frequency in closed loop simulation. The RDNN model was then incorporated into MPC scheme to control the column and simulation results showed that the RDNN based MPC outperform the normal IMC and Input-Output Linearization based controller.

For the real time application, Findeisen and Allgower (2002) reported an application of Nonlinear Model Predictive Control (NMPC) to control a 40 tray high purity distillation column for the separation of Methanol and n-Propanol. They applied the Quasi-Infinite Horizon MPC algorithm (QIH-MPC) which is found more real-time feasible compare to the optimization toolbox in MATLAB. The former requires only 0.89s for the solution of optimal control problem for 42^{nd} order model whereas the latter needs in average 620s and hence is not real time implementable.

2.4 Neural Networks for System Identification and Control

The interest of employing the NNs in nonlinear system identification and control has increased and grown over the last decade. This is largely due to the proven superiority of NNs, or more specifically, the ability of certain NNs architecture, such as the Multi-Layered Perceptron (MLP) in arbitrary non-linear mapping (Hornik *et al.*, 1989). Hunt *et al.* (1992) and Nareda and Parthasarathy (1990) have reported a review paper on the system identification and control using neural network. According to Hunt *et al.* (1992), typical application of NNs to nonlinear control is based on internal model control (IMC) strategy where the control structure uses both a forward and an inverse NNs model to control a system. This

approach have been applied widely especially in the robotic control system. However, in this thesis, the interest is on the NNMPC in which the NNs is used as forward model for the prediction of process output.

In recent years, after the successful application of MPC in industry, the drive in refining the current MPC schemes has motivated the growth and development of many nonlinear modeling approaches. Among them, the NNs is the most popular empirical modeling method and numbers of literature review on the incorporation of various kinds NNs into MPC schemes have been reported. For example, Zhan and Ishida (1994) reported an application of Feedforward Neural Networks based Model Predictive Control scheme to control an unstable nonlinear Constant Stirred Tank Reactor (CSTR). The input data are fed to the trained FNN in recursive way to in order to perform the multi-step prediction. However, their multi-step prediction algorithm is limited to Single Input Single Output (SISO) process only. Pottmann and Seborg (1997) employed a Radial Basis Function (RBF) neural network. The RBF model is developed using stepwise regression and least squares estimation. An implementation of MPC by using the RBF neural network is also demonstrated and the results showed that RBF based MPC outperform the PI controller in controlling a stirred tank in a pH neutralization process. Shaw *et al.* (1995) applied the Recurrent Neural Network (RNN) in modeling and control the Constant Stirred Tank Reactor (CSTR) and Cycol-pentenol production process. They found that RNN outperform the FNN and linear model in performing the prediction. They claimed that the FNN is unable to capture the behavior of the system in this case because the fact the FNN is essentially an Auto-Regressive Moving Average (ARMA) model which is not able to perform well in doing multi-step prediction. The control results also showed that the RNN based MPC outperform the FNN based MPC.

2.4.1 Neural Networks and Feedforward Neural Networks

Neural Networks (NNs) basically comprised of interconnected simulated neurons. A neuron is the smallest unit in NNs and is used to receive and send signals. Normally, each neuron receives signals from other neurons, sums these signals and

transforms this sum by means of an activation function, which is monotonic continuously differentiable, bounded function. Frequent used activation function including logistic sigmoid and hyperbolic tangent functions. In addition, there are weights associated with each connection that scale the input to target and training process is to determine optimal weight. The neuron can be arranged into multi-layers which are normally known as Multi-Layer Perceptron (MLP). The Figure 2.1 illustrated the basic structure of a neuron.

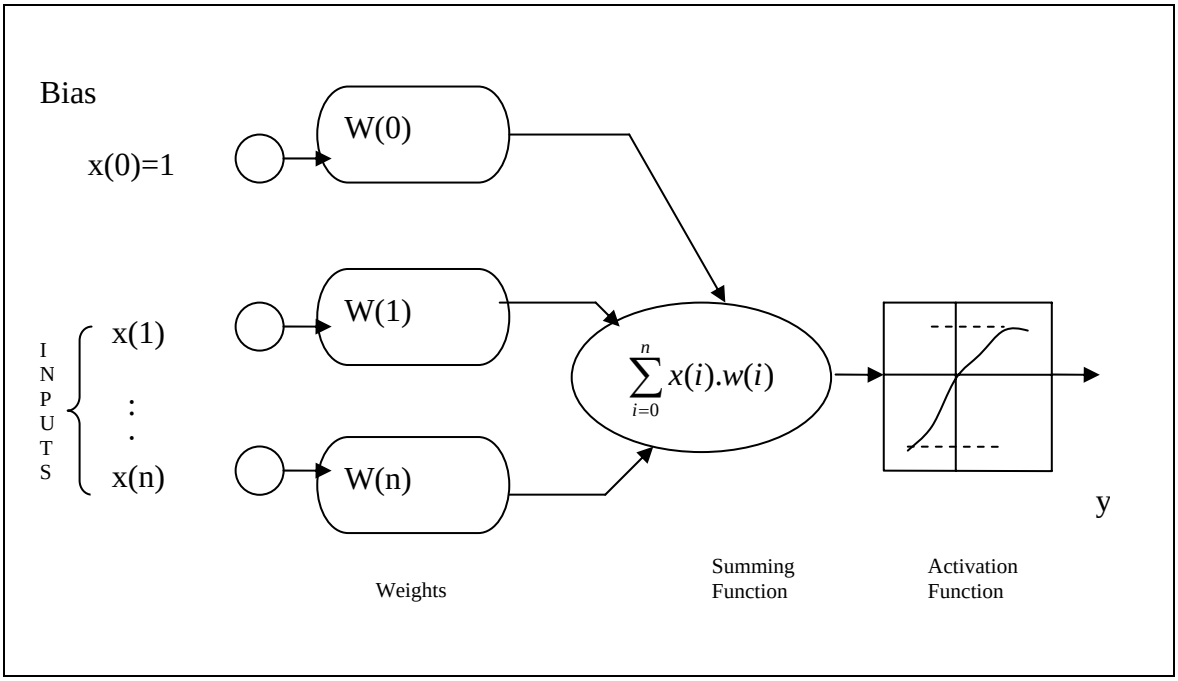

Figure 2.1 A Basic Architecture of A Neuron

In chemical engineering application to date the most widely used neural network is the Feedforward Neural Network (FNN). This is large due its simplicity compared to other networks and its ability to learn the implicit governing relation between the inputs and outputs if sufficient training data is supplied. Feedforward networks is network structure in which the information or signals will propagates only in one direction on contrary to the recurrent networks in which the time-delayed neural net outputs will feed back in to the neural networks as inputs. The FNN typically consist of three or four layers including input layer, hidden layer and output layer (as depicted in Figure 2.2).

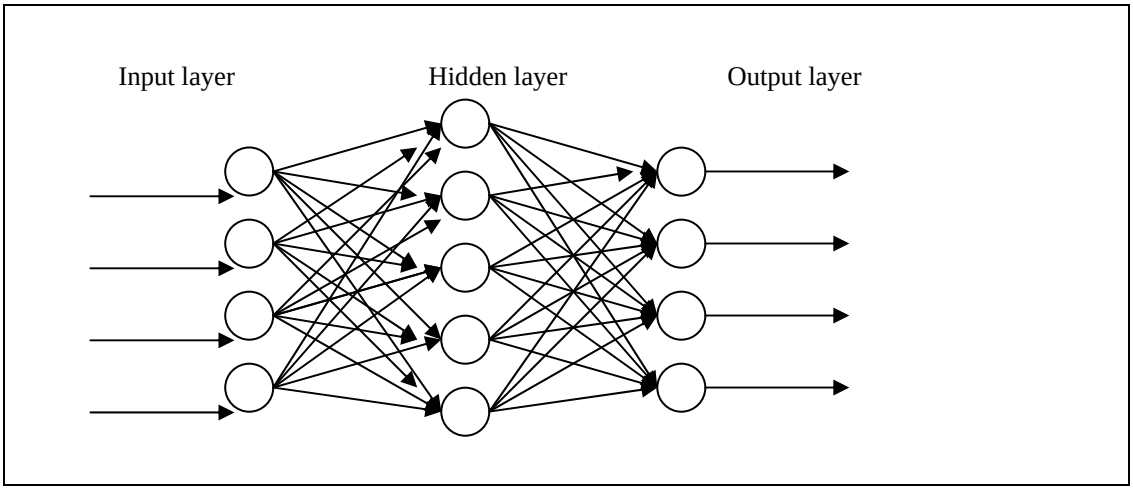

Figure 2.2 A Feedforward Artificial Neural Network Structure

It is possible to have more than one hidden layer but a single layer is sufficient to approximate any function to a desired degree of accuracy (Hornik *et al.*, 1989). The numbers of neurons in the input layer and output layer are normally determined by the problem. However, the number of neurons in the hidden layer has to be specified; optimal number of neurons has to be determined in order to obtain a good identified network. If there are too few neurons in the hidden layer, the network may be unable to describe the underlying function because it has insufficient parameters to map all point in the training data. On the contrary, if there are too many neurons, the network would have too many parameters and might overfit the data and results losing of the ability to generalize. For most cases to date, the best way to determine the optimal number of neurons is done by systematic trial and error.

2.4.2 Training

Training is basically a systematic adjustment of weights to get a chosen neural network to predict a desired output data set (training set) and it can be done in either supervised or unsupervised way. The training for FNN is supervised. In the supervised training, the connection weights for each processing element are randomly initialized. As the training begins, the training algorithm will start to compare NNs's predicted outputs to the desired outputs (from training data set), and

any error will be used to correct the network. The correction is done by adjusting the set of connection weights of each processing element (neuron) and this will continues until the algorithm meet the pre-specified convergence criteria. The frequent used criteria including the limit of error and the numbers of iteration. However, care must taken to ensure that the network does not overfit or overfamiliarize with the training data set and hence lose its generalization ability. Various approaches can be used to avoid this problem including regularization theory which attempt to smooth the network mapping (Larsen and Hansen, 1994) and cross-validation which using as independent test data.

2.4.3 Backpropagation

In the majority of studies, the FNN will employ backpropagation as its training algorithm. Backpropagation get its name from the fact that, during training, the output error is propagated backward to the connections in the previous layers, where it is used to update the connection weights in order to achieve a desired output. Typical backpropagation is a gradient descent optimization method, which is executed iteratively with implicit bounds on the distance moved in the search direction in the weight space fixed via learning rate, which is equivalent to step size. The backpropagation technique adjusts each variable (weight) individually according to the size along the path of the steepest descent to minimize the objective function.

For example, given a set of input-output training data in which (Koivisto, 1995):

$$Z^N = \{u(k), y(k)\}_{k=1,\ldots,N} \tag{2.1}$$

and a set of candidate models:

$$\hat{y}(t \mid \theta) = g(\varphi(t), \theta) \tag{2.2}$$

Training a neural network implies fitting the network parameter θ (which is the weights in NNs) such that the network learns to approximate the output sequence

{y(k)} by giving the input sequence *{u(k)}*. Normally, the Prediction Error Method (PEM) would be used and the estimated parameter θ can be found such that to minimize an objective function which is typically a Mean Square Error (MSE).

$$\theta = \min_{\theta} V_N(\theta, Z^N) \tag{2.3}$$

Where $V_N(\theta, Z^N) = \frac{1}{2N}\sum_{i=1}^{N}[y(t) - \hat{y}(t \mid \theta)]^2$ (2.4)

For the backpropagation algorithm, the way to find θ is done in iterative way. The backpropagation often start with an initial parameter vector θ and then the training would iteratively decrease the MSE in equation 2.4 by incrementally update the θ along the negative gradient of MSE as follow:

$$\theta^{i+1} = \theta^i - \eta\delta_k \tag{2.5}$$

where the η is the learning rate, and δ_k is the gradient of the objective function

$$\delta_k = \frac{\partial V_N(\theta)}{\partial \theta} \tag{2.6}$$

This procedure of updating the θ or weights using only the gradient information often requires a small step size (learning rate) to attain stability. Thus, the backpropagation method has to be repeated many times to achieve the minimum value of the objective function. Small step size able to ensure convergence but would increases the number of iteration and calculation time. In addition, a local minimum solution also being a problem for this method and normally, a trial and error procedure (start with a different set of initial weights) would be employ and try to get a global minimum convergence.

In recent years, other than classical backpropagation training algorithm, numbers of other optimization alternatives have been developed and applied for training neural network. For example, the Levenberg Marquardt, Quasi-Newton and Conjugate gradient approaches. All these algorithms might display a more reliable

and fast convergence ability compare to backpropagation under certain circumstances.

2.4.4 Neural Networks based Multi-Step Predictor

In most of the NNMPC scheme, NNs is employed as the Multi-Step Predictor (MSP) whose require to predict the process output over the prediction horizon. Literature review had shown that there are three main approaches in performing the MSP using NNs:

1. Recursive Prediction: In this approach, NNs is used to do one step ahead prediction and the predicted output plus current input measurement is feedback to same NNs iteratively to do following multi step ahead prediction. For example, assume there is a one step ahead NARX predictor (Koivisto, 1995):

$$\hat{y}(t+u \mid i) = f(\varphi(t+i), \theta), i = 1...N_2 \tag{2.7}$$

where the input vector,

$$\varphi(t,\theta) = [y^T(t\text{-}1),\ldots, y^T(t\text{-}m_y), u^T(t\text{-}d),\ldots,u^T(k\text{-}d\text{-}mu\text{+}1)]^T \tag{2.8}$$

The future measurements are unknown and thus they are replaced with the predicted one. The Figure 2.3 illustrated an example of neural network that acts as multi-step predictor under the assumption of m_y= 2, m_u=1 and d=1 and prediction horizon, P =3.

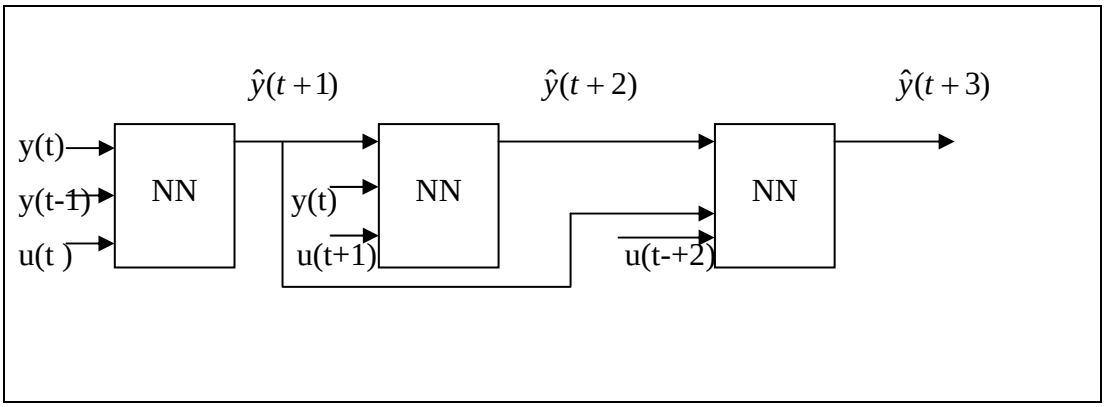

Figure 2.3 Neural Network-based Multi-Step Predictor Employing Recursive Prediction Method

However, there is a drawback for this method where the small single step (SS) prediction errors at the beginning of the horizon would accumulate and propagate and this often resulting in poor prediction accuracy. To reduce the prediction error, data conditioning method is used where each network input and output variable is normalized and then the normalized data is applied to each single network nodes. Moreover, Spread Encoding method proposed by Gomm *et al.* (1996) by utilizing some fuzzy logic knowledge has been proven able to enhance the performance of MLP especially in doing the long range prediction.

2. Grouped neural network strategy (Ou, 2001)**:** In this method, several of separate direct *i*-step ahead predictors or grouped neural network is used to so the multi step prediction. For example, consider a prediction horizon of three and the 3 step ahead prediction can be done as follow:

$$\hat{y}_{k+1|K} = NN_1(y_k, u_k) \tag{2.9}$$

$$\hat{y}_{k+2|K} = NN_2(y_{k+1}, u_{k+1}) \tag{2.10}$$

$$\hat{y}_{k+2|K} = NN_3(y_{k+2}, u_{k+2}) \tag{2.11}$$

To do the first prediction (as described by equation 2.9), the present output, y_k and future manipulated move, u_k is required and since both of them is known, the equation can used to evaluate $\hat{y}_{k+1|K}$. The second prediction is depend on y_{k+1} and u_{k+1}, and for this, u_{k+1}, the manipulated move to be made at time k+1 is known but

the output at the next instant yk+1 is unknown at time k. However, y_{k+1} can be assume to be equal to $\hat{y}_{k+1|K}$ and by combining the equation 2.9 to equation 2.10,

$$\hat{y}_{k+2|K} = NN_2(y_k, u_{k+1}, u_k) \tag{2.12}$$

By extending this approach to third prediction,

$$\hat{y}_{k+3|K} = NN_3(y_k, u_{k+2}, u_{k+1}, u_k) \tag{2.13}$$

In short, for this approach, three neural networks are required for three predictions. There are two inputs to the NNs for the one step prediction whereas there are four inputs for three step predictions. The limitation of this approach is that it is not suitable to do a large prediction due to the fact that the number and size of NNs as well as the data required to process will increase significantly as the number prediction increase.

3. Employing a dynamic recurrent neural network (Su *et* al., 1992; Parlos *et* al., 2000): Instead of using the feedforward network as what happen in two previous methods, the latest approach is using a single time lag recurrent network to perform the Multi Step Prediction. This approach is done by employing a single multi-step prediction recurrent network and then trains it with some dynamic optimization method such as dynamic gradient descent method (Parlos *et* al., 2000). The recurrent network showed a better prediction accuracy compare to FNN especially in performing a Multi Step Prediction due to its inherent dynamic (local hidden layer fed back/data feedback via tapped delay).

2.5 Summary

In summary, a complete and updated literature review of MPC technology is presented in this chapter. They includes the historical background of MPC; the basic ideas of MPC; the reported implementation of LMPC and NMPC in industry; the reason of popularity of LMPC; the advantages and limitations of LMPC; the ideas of AMPC; the stability and robustness issues in LMPC; the reasons why we need

NMPC; the basic elements of NMPC; the difficulties associated in performing the NMPC and some of the alternative solutions in counteract it. Additionally, the literature review in studying the application of MPC in controlling a high purity distillation column (interest of this research) had shown that the comparison between the performance of DMC and NNMPC particularly in controlling a high purity distillation column remains an unexplored issue (similar research hasn't been found) and this has reinforced again the need of conducting this research. Lastly, the description for FNN based system identification and three common ways to perform the multi step prediction were also shown in this chapter.

CHAPTER 3

CASE STUDY 1: LMPC ON CONTINUOUS SOLUTION COPOLYMERIZATION REACTOR

In this chapter, a control system is designed for a copolymerization reactor using the feedback control to regulate polymer production rate, copolymer composition, molecular weight and reactor temperature. A model is developed to illustrate the behaviour of the copolymerization process. Relative Gain Analysis (RGA) is used to investigate input/output control pairings in order to identify fundamental nature of the solution copolymerization control problem and to determine the best control system structure. PID control and LMPC control techniques are applied to compare the results. A flowchart of methodology is illustrated in Figure 3.1.

3.1 Process Description and Model Development

The solution polymerization of methyl methacrylate and vinyl acetate in a continuous stirred tank reactor is described in Figure 3.2 (Congladis *et al*., 1989). Monomers A (methyl methacrylate) and B (vinyl acetate) are continuously added with initiator, solvent and chain transfer agent. A coolant flows through the jacket to remove the heat of polymerization. Polymer, solvent, unreacted monomers, initiator and chain transfer agent flow out of the reactor to the separator for further processing. In this study, initiator used is azobisisobutyronitrile (AIBN) and the chain transfer agent is

acetaldehyde. This system is interesting as methyl methacrylate (MMA) is much more reactive than vinyl acetate (VAc) in copolymerization, as indicated by their respective reactivity ratios of 26 and 0.03.

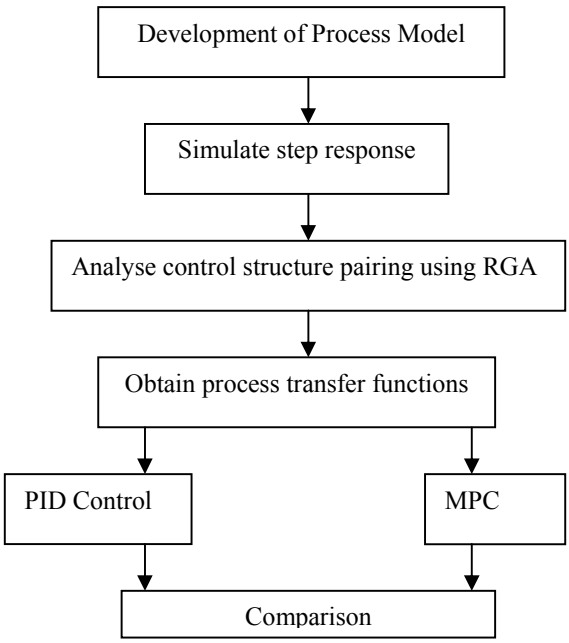

Figure 3.1 Methodology for Case Study 1

The level of liquid in the reactor is assumed constant. The steady-state operating conditions are shown in Table 3.1.

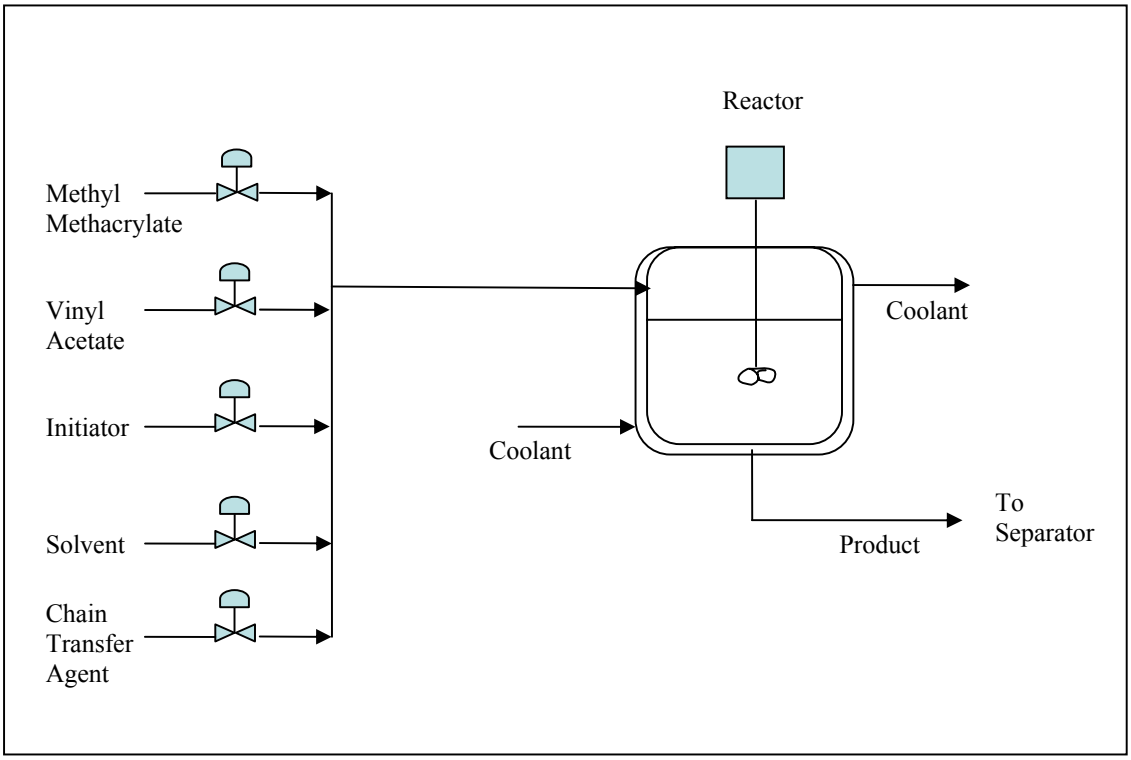

Figure 3.2 Solution Copolymerization in Continuous-Stirred Tank Reactor

3.2 Step Response Simulation

The flow of work for developing step response is shown below:

a) Define input variables
b) Steady-state operations
   - Define initial values
   - Run the ODE function from t=0 to t= 55 hours
   - Use the ODE values to calculate other output variables
c) Plot output variable versus time
d) Step change each variable
e) Run ODE functions and obtain output variables
f) Plot step response

Table 3.1: Steady-state Operating Conditions

Inputs	**Value**
Monomer A (MMA)	18 kg/hr
Monomer B (VAc)	90 kg/hr
Initiator (AIBN)	0.18 kg/hr
Solvent (Benzene)	36 kg/hr
Chain Transfer Agent (Acetaldehyde)	2.7 kg/hr
Inihibitor (m-DNB)	0 kg/hr
Reactor Jacket Temperature	336.15 K
Reactor Feed Temperature	353.15 K
Reactor Parameters	**Value**
Residence Time	6 hr
Volume	1 m^3
Heat Transfer Area	4.6 m^3
Density	879 kg/m^3
Outputs	**Value**
Polymer Production Rate	23.3 kg/hr
Mole fraction of A in Polymer	0.56
Weight Average Molecular Weight	35 000
Reactor Temperature	353.01 K

3.3 Relative Gain Array Analysis

Relative gain method is a measure of the influence a selected manipulated variable has over a particular controlled variable relative to that of other manipulated variables acting over the process. The relative gain of a controlled variable, i to a manipulated variable, j is defined as

$$\lambda_{ij} = \frac{(\partial c_i / \partial m_i)_m}{(\partial c_i / \partial m_i)c} \tag{3.1}$$

For the copolymerization process, it is convenient to arrange the RGA in an array.

$$\mathrm{A} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccc} m1 & m2 & m3 & m4 \end{array} \\ \begin{bmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} & \lambda_{14} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} & \lambda_{24} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} & \lambda_{34} \\ \lambda_{41} & \lambda_{42} & \lambda_{43} & \lambda_{44} \end{bmatrix} \end{array} \tag{3.2}$$

One property of the array Λ is that the relative gain in each column and row add up to unity. The λ_{ij} that has the nearest value to unity will be paired with that particular I controlled variable and j manipulated variable. If any lies outside 0 and 1, it means that there is a substantial difference between those processes. The matrix method has been used to pair up the variables for this copolymerization process. Results of RGA analysis for four sets of selected manipulated variables are shown in Table 3.2.

Control variables :

Gpi, yap, Mpw, Tr

Manipulated variables:

Set 1: Gif/Gbf, Gaf/Gbf, Gbf, Tj

Set 2: Gtf/Gbf, Gaf/Gbf, Gbf, Tj

Set 3: Gif, Gaf, Gbf, Tj

Set 4: Gtf, Gaf, Gbf, Tj

From the four sets, the best set of manipulated variables is Set 2 with the pairing as below:

Gtf/Gbf	:	Gpi
Gaf/Gbf	:	yap
Gbf	:	Mpw
Tj	:	Tr

3.4 Results and Discussion

3.4.1 Digital PID Feedback Control

Figure 3.3 shows the block diagram for digital PID feedback control for copolymerization system. For multivariable input multivariable output system (MIMO), there will still be interactions between the manipulated variables with the controlled variables as shown in Figure 3.4. In controlling the copolymerization process, there will be four PID controllers for a 4 x 4 input-output variable system. The equation for the digital PID controller in its velocity form are shown below.

$$\Delta p_n = p_n - p_{n-1} = K_c\left[\left(e_n - e_{n-1}\right) + \frac{\Delta t}{\tau_I} e_n + \frac{\tau_D}{\Delta t}\left(e_n - 2e_{n-1} + e_{n-2}\right)\right] \qquad (3.3)$$

Where Δt =the sampling period

p_n = controller output at the *t* sampling instant.

e_n = error at the *t* sampling instant

Kc = Ultimate controller gain

τ_I = Integral time

τ_D = derivative time

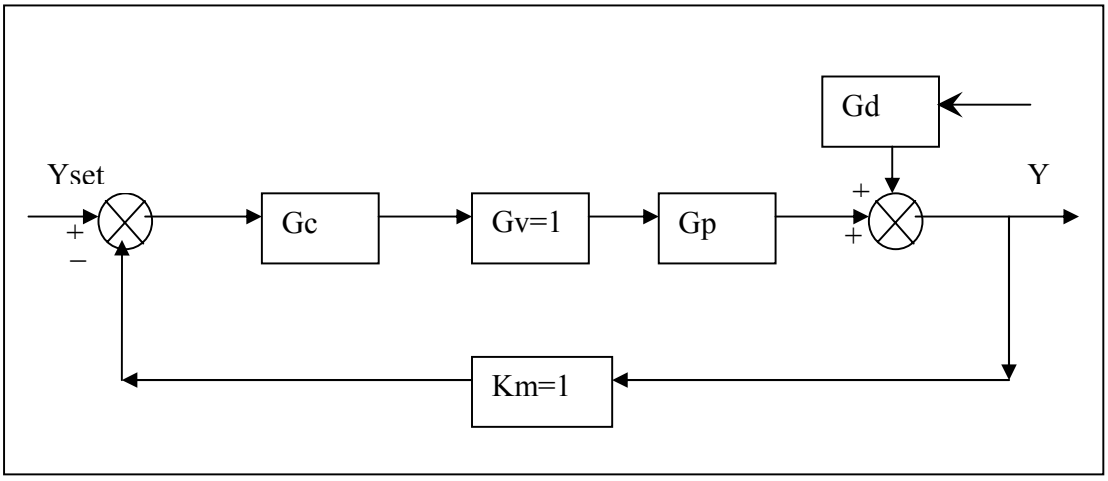

Figure 3.3 Block Diagram for PID Control System

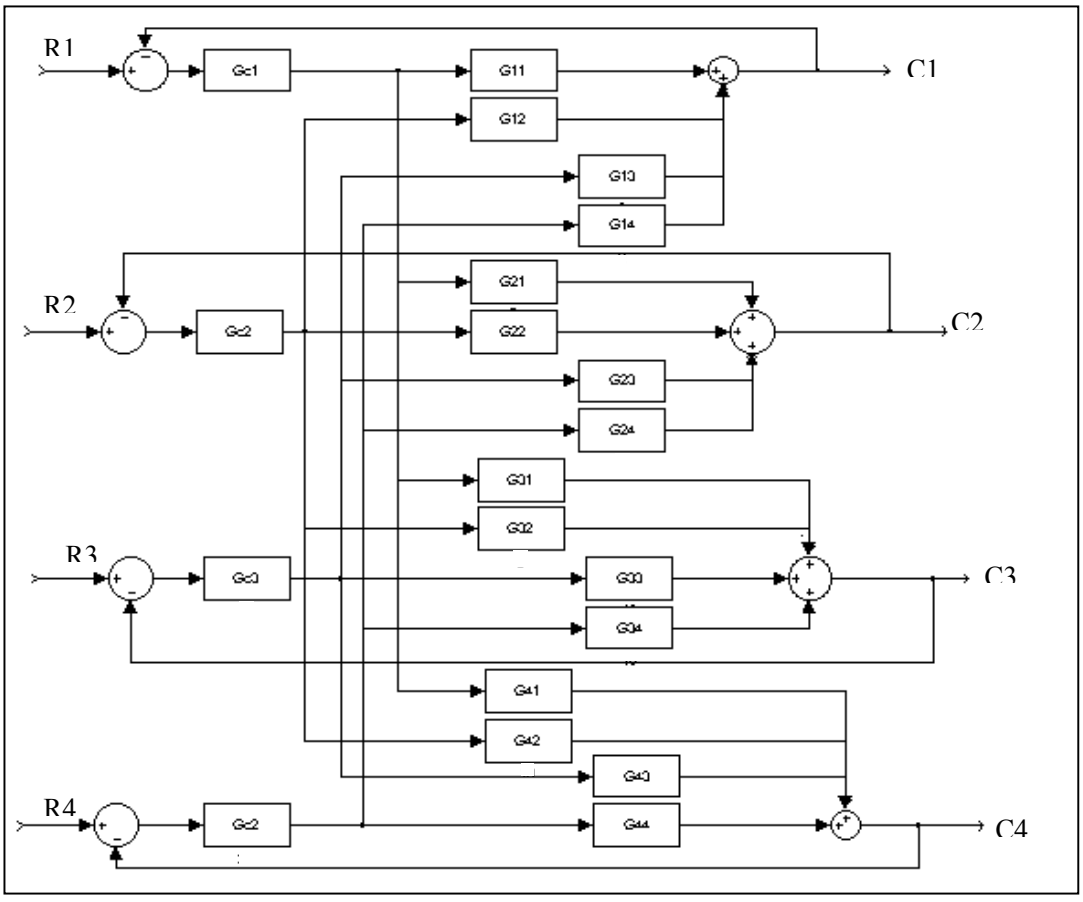

Figure 3.4 Block Diagram for a MIMO Control System

The results for the simulation are shown in Figures 3.5 to 3.10. In Figure 3.5, the response after a disturbance of 10% ratio between Gtf to Gbf was seen to be negligible. There was hardly any change in the outputs. However, there was a significant change in polymer composition and molecular weight after a 10% change in the ratio of Gaf to Gbf. The nomenclature for the symbols are as follows:

Input 1: Gt/Gbf (ratio of transfer agent feed rate to the monomer B feed rate)
Input 2: Gaf/Gbf (ratio of monomer A feed rate to monomer B feed rate)
Input 3: Gbf (monomer B feed rate)
Input 4: Jacket temperature

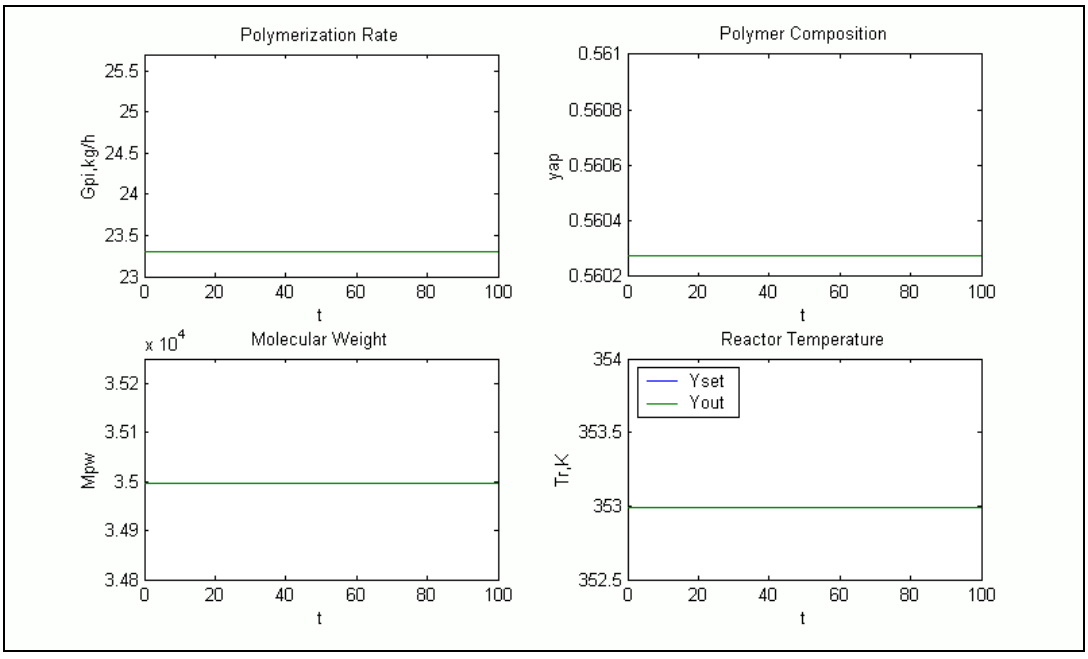

Figure 3.5 Result After Adding 10% Of Gtf/Gbf as Disturbance. (Very Small Change Observed)

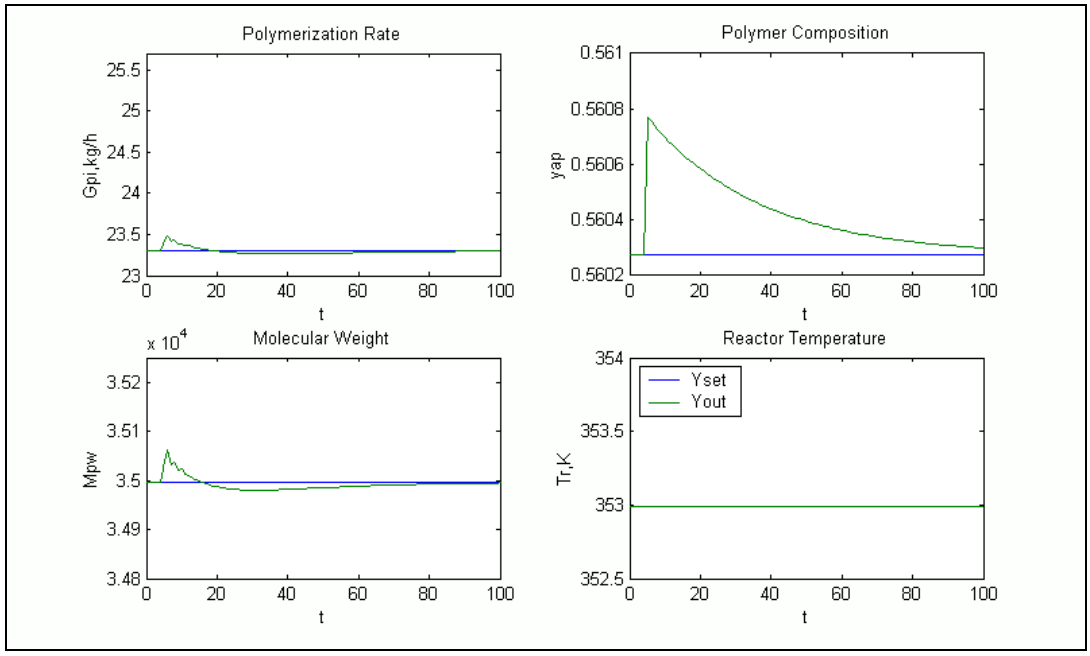

Figure 3.6 Result After Adding 10% Of Gaf/Gbf as Disturbance.

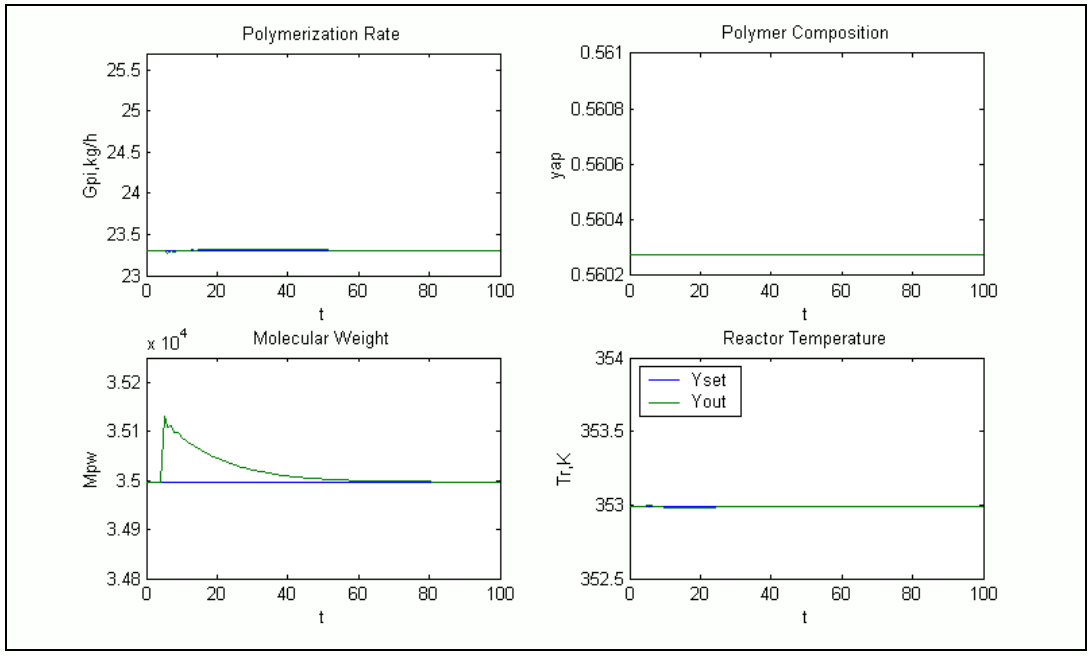

Figure 3.7 Result After Adding The 10% Of Gbf as Disturbance.

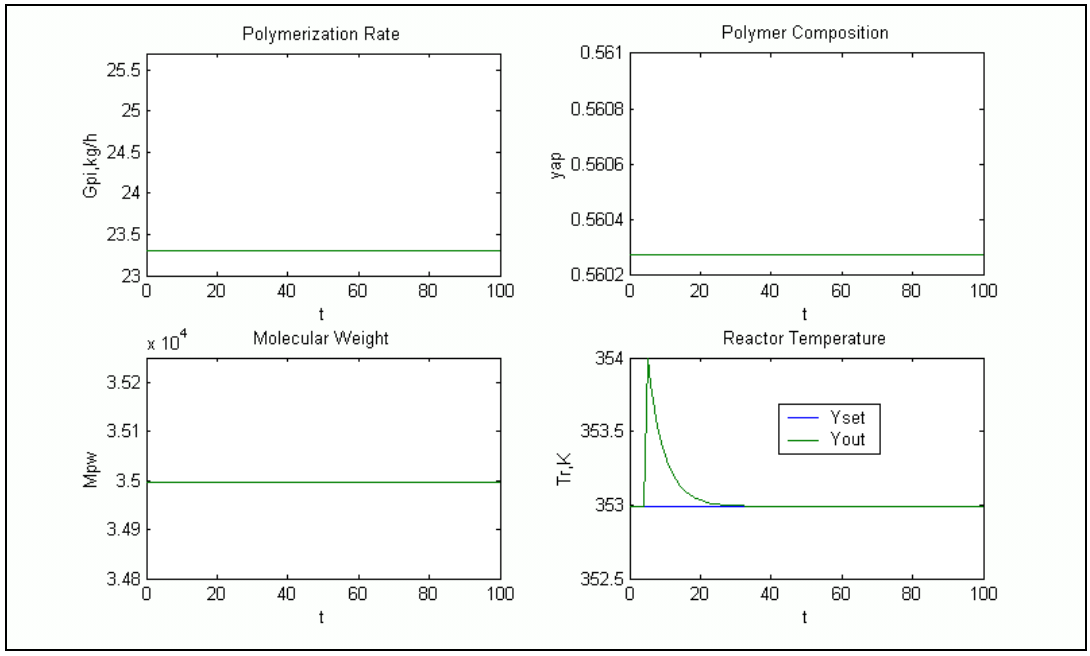

Figure 3.8 Result After Adding The 10% Of Tj as Disturbance.

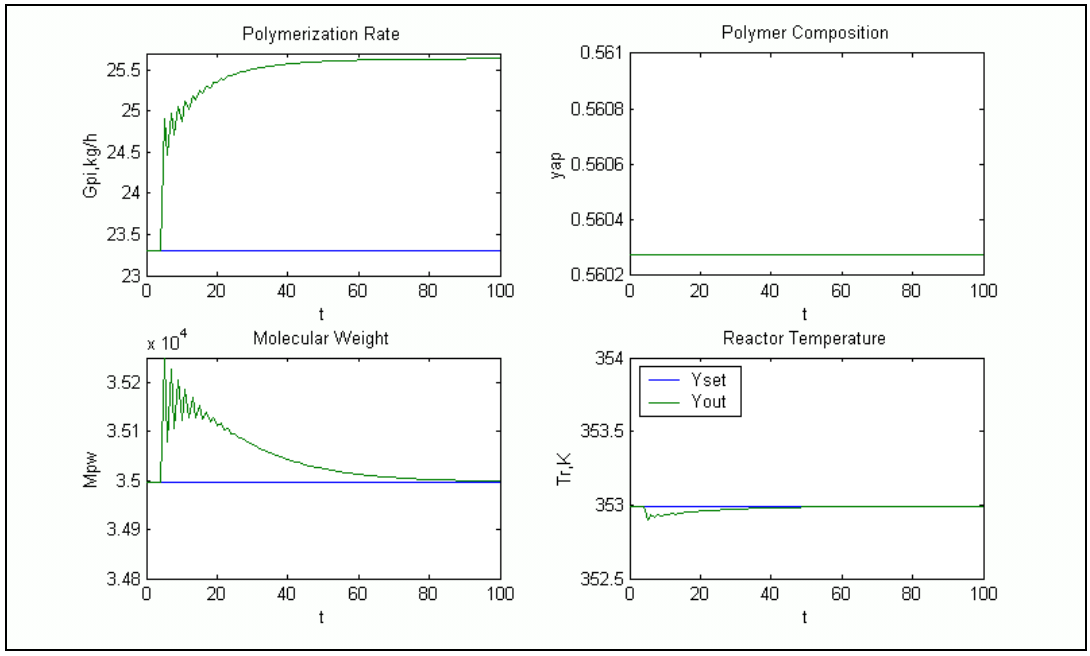

Figure 3.9 Result After Adding The 10% Of Gpi as New Set-Point

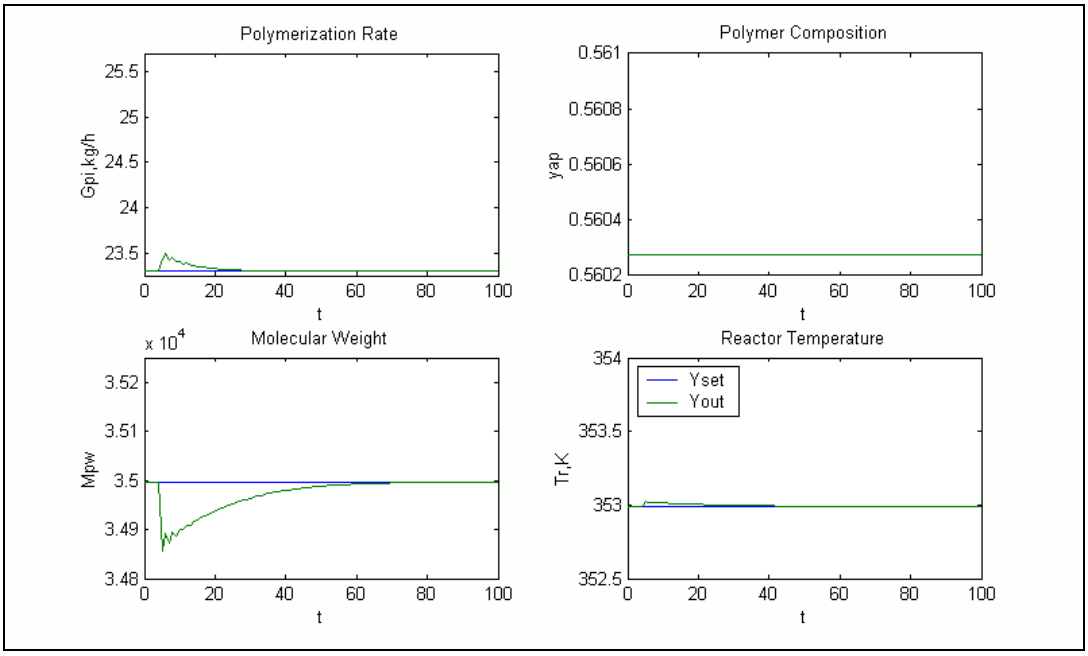

Figure 3.10 Result After Adding 10% To Gif as Disturbance.

3.4.2 Linear MPC

In this case study, LMPC based on state-space model is used. The general discrete-time linear time invariant (LTI) state-space representation used in the MPC toolbox in MATLAB is as follows.

$$x(k+l) = \Phi x(k) + \Gamma_u u(k) + \Gamma_d d(k) + \Gamma_w w(k) \tag{3.4}$$

$$y(k) = Cx(k) + D_u u(k) + D_d d(k) + D_w w(k) \tag{3.5}$$

where x = vector of n state variables

u = manipulated variables

d= measured but freely-varying inputs

w= immeasurable disturbances

z= measurement noise

The results from the simulation are shown in Figures 3.11 to 3.22. Results for each run is shown in two figures which represent changes in inputs and outputs. The inputs and outputs are as follows:

Input 1: Gt/Gbf (ratio of transfer agent feed rate to the monomer B feed rate)
Input 2: Gaf/Gbf (ratio of monomer A feed rate to monomer B feed rate)
Input 3: Gbf (monomer B feed rate)
Input 4: jacket temperature

Output 1: Gpi (polymerization rate)
Output 2: yap (composition of monomer A in the product polymer)
Output 3: Mpw (molecular weight of polymer)
Output 4: Tr (reactor temperature)

d(1): Gif (initiator feed rate)
d(2): Gsf (solvent feed rate)

Steady state operation values:

Input	=	[0.03	0.2	90	336.15]
Output	=	[23.3	0.56	35000	353.01]
d	=	[0.18	36]		

Conditions given:

P	=	6	number of prediction horizon
M	=	[3]	blocking factor
ywt	=	[1 3 1 1]	output weight
uwt	=	[1 0 0 0]	input weight

Figures 3.11 to 3.18 show the inputs and outputs with MPC when 4 different set point changes were applied. As seen from the figures, the MPC were able to perform

the set point changes smoothly. Figures 3.19 to 3.22 illustrate the inputs and outputs with MPC when 2 different disturbances occurred. As for the servo control, the MPC were able to correct the controlled variables back to the set point smoothly.

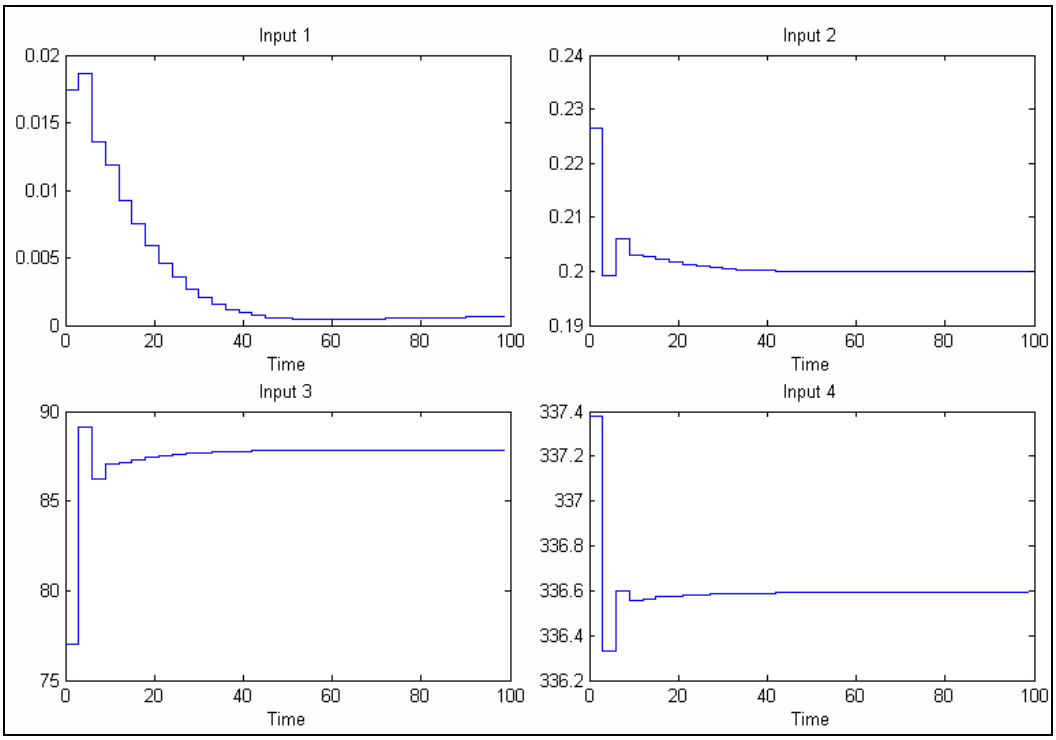

Figure 3.11 Input Changes For The New Set-Point (Gpi = 24 kg/h)

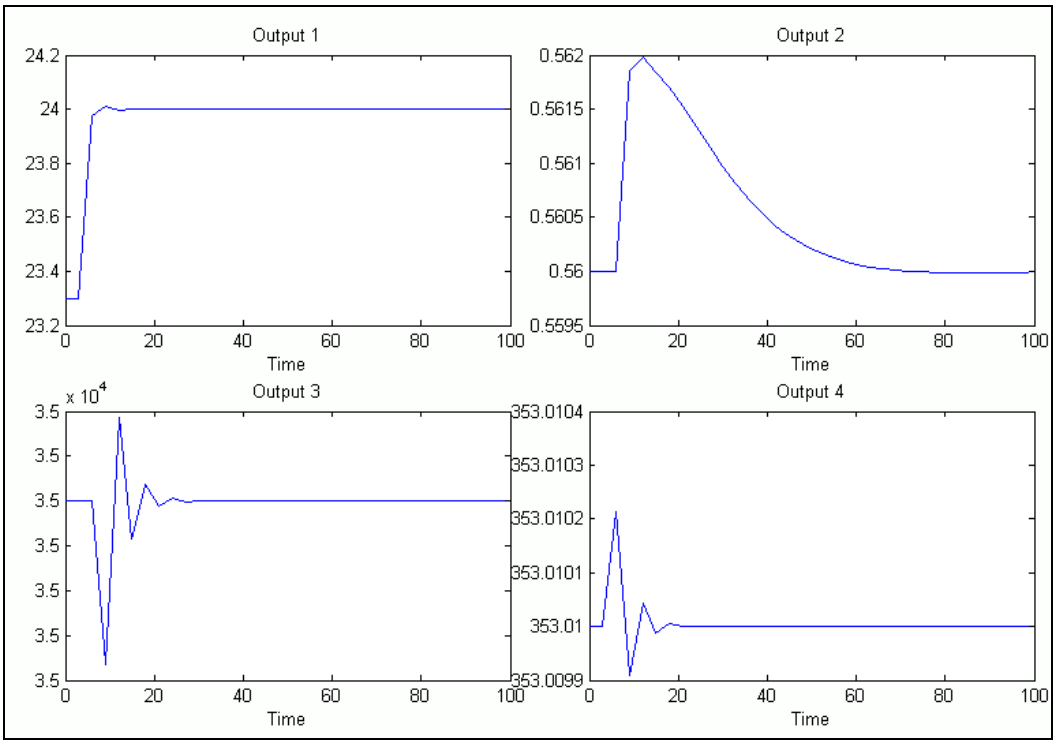

Figure 3.12 Output Changes For The New Set-Point (Gpi = 24 kg/h)

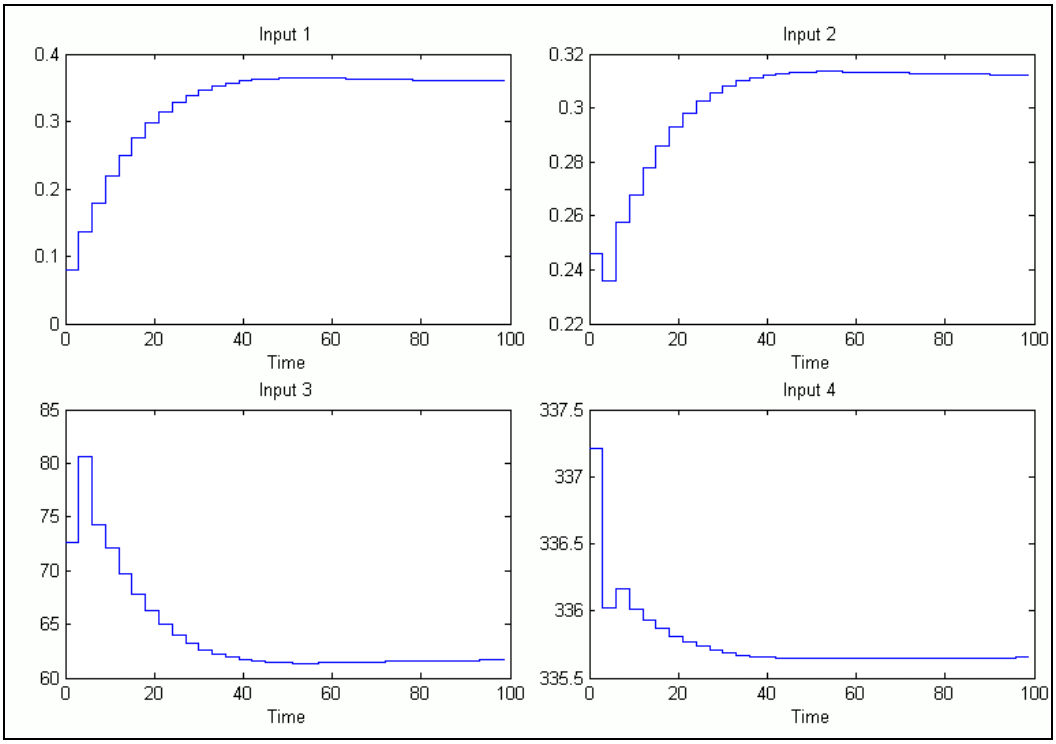

Figure 3.13 Input Changes For The New Set-Point (Gpi = 24 kg/h, yap = 0.6)

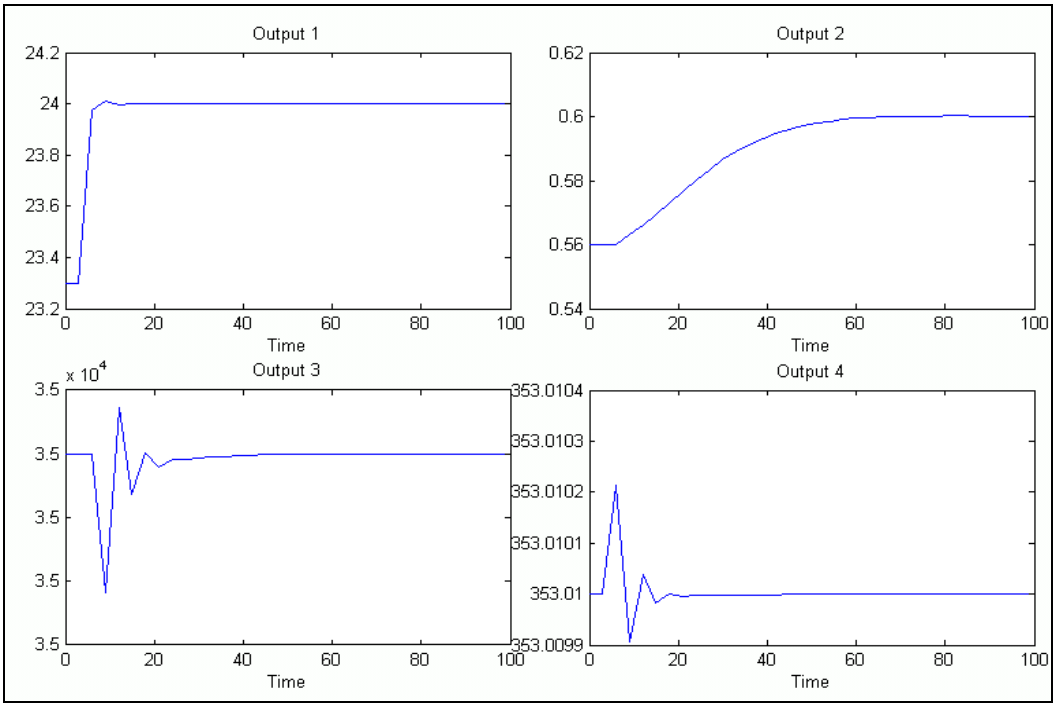

Figure 3.14 Output Changes For The New Set-Point (Gpi = 24 kg/h, yap = 0.6)

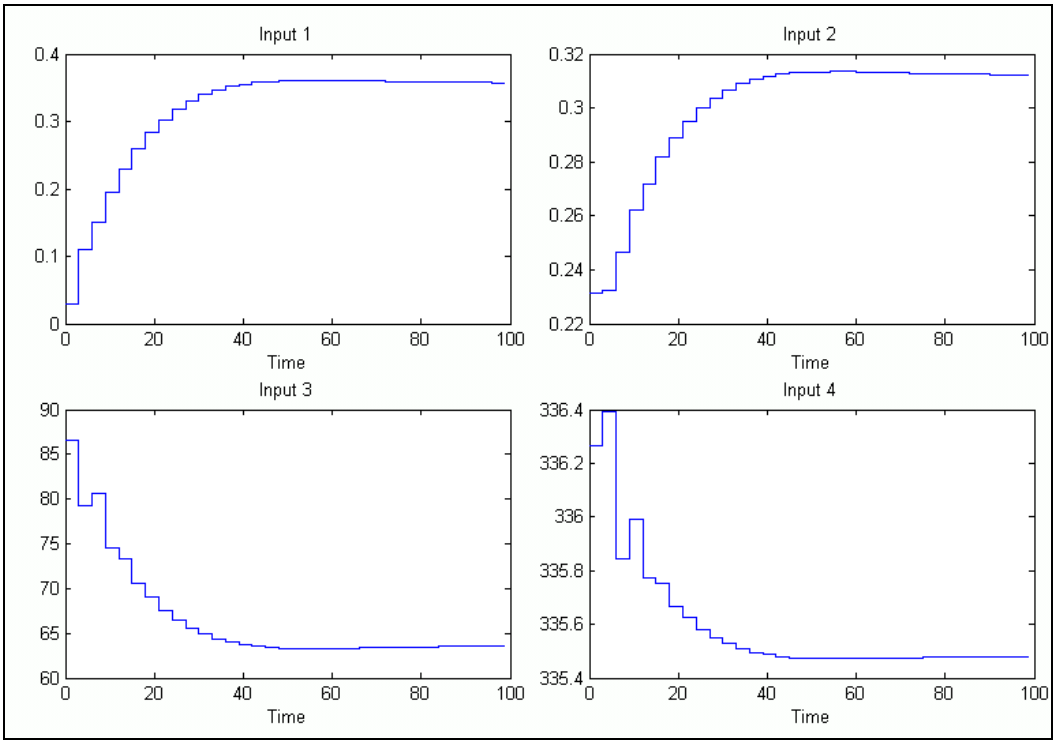

Figure 3.15 Input Changes For New Set-Point (Gpi = 24kg/h, yap = 0.6, Mpw = 35500)

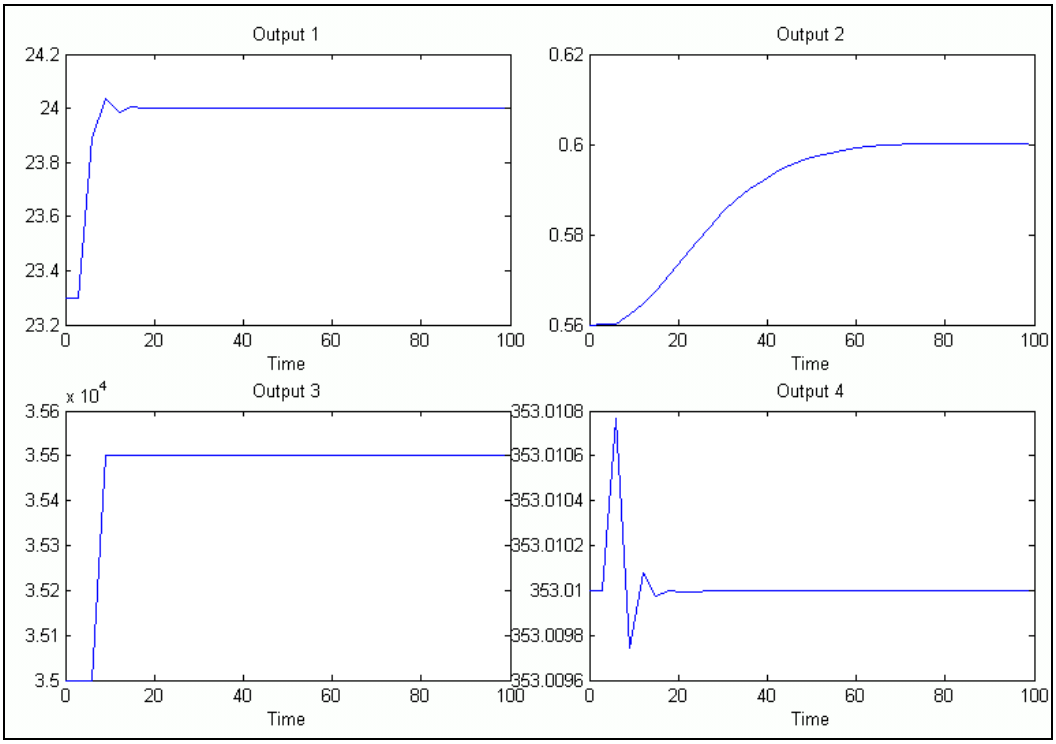

Figure 3.16 Output Changes For New Set-Points (Gpi = 24kg/h, yap = 0.6, Mpw =35500)

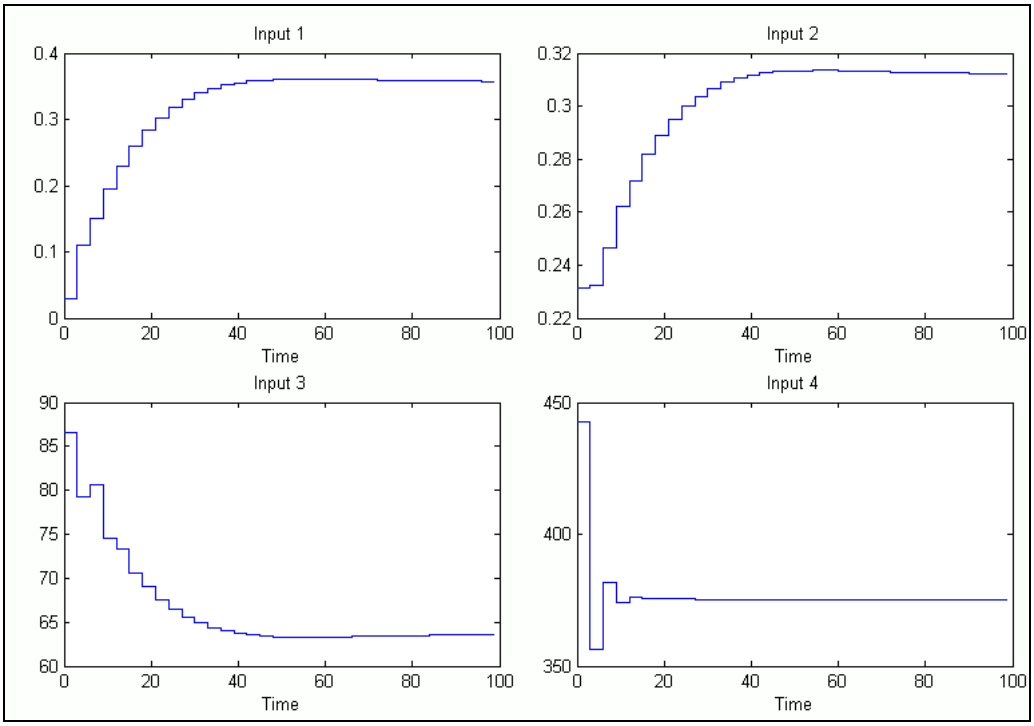

Figure 3.17 Input Changes For New Set-Points (Gpi=24kg/h, yap=0.6, Mpw=35500, Tr=363K)

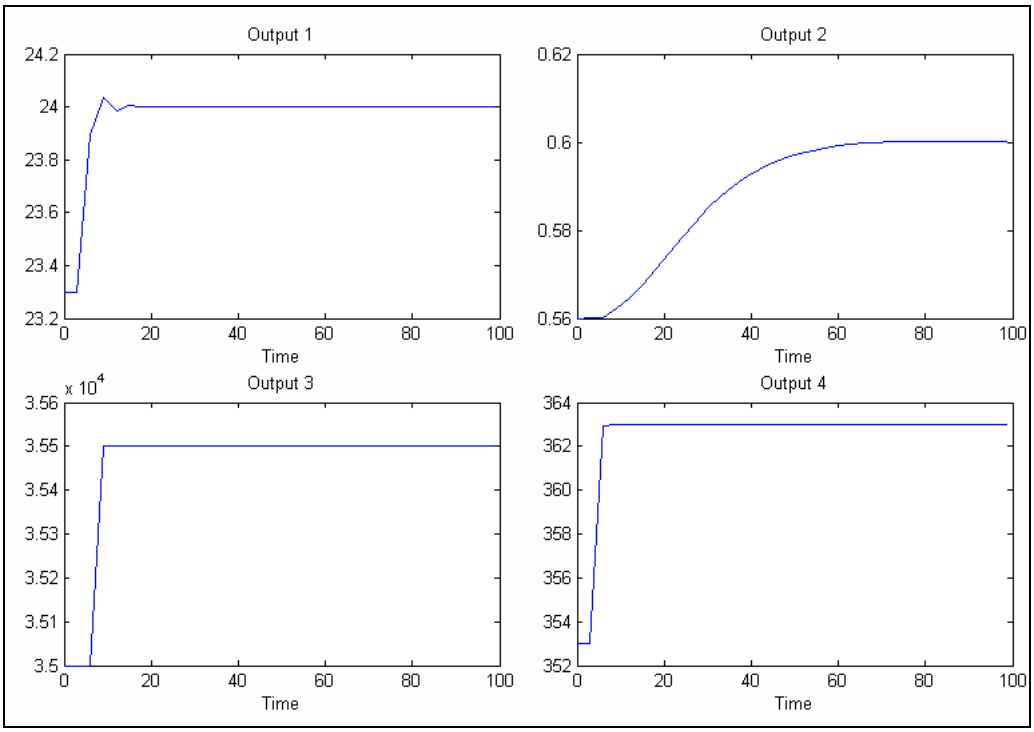

Figure 3.18 Output Changes For New Set-Points (Gpi=24kg/h, yap=0.6, Mpw=35500, Tr=363K)

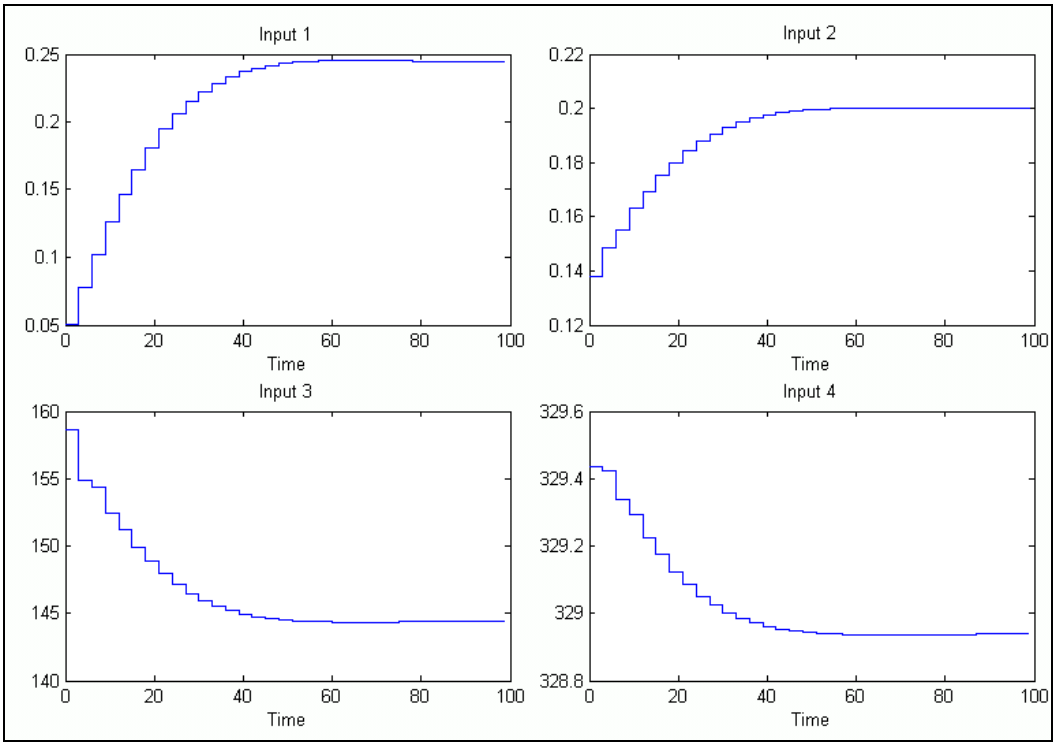

Figure 3.19 Input Changes When Disturbance Loaded (d(1) = 0.28)

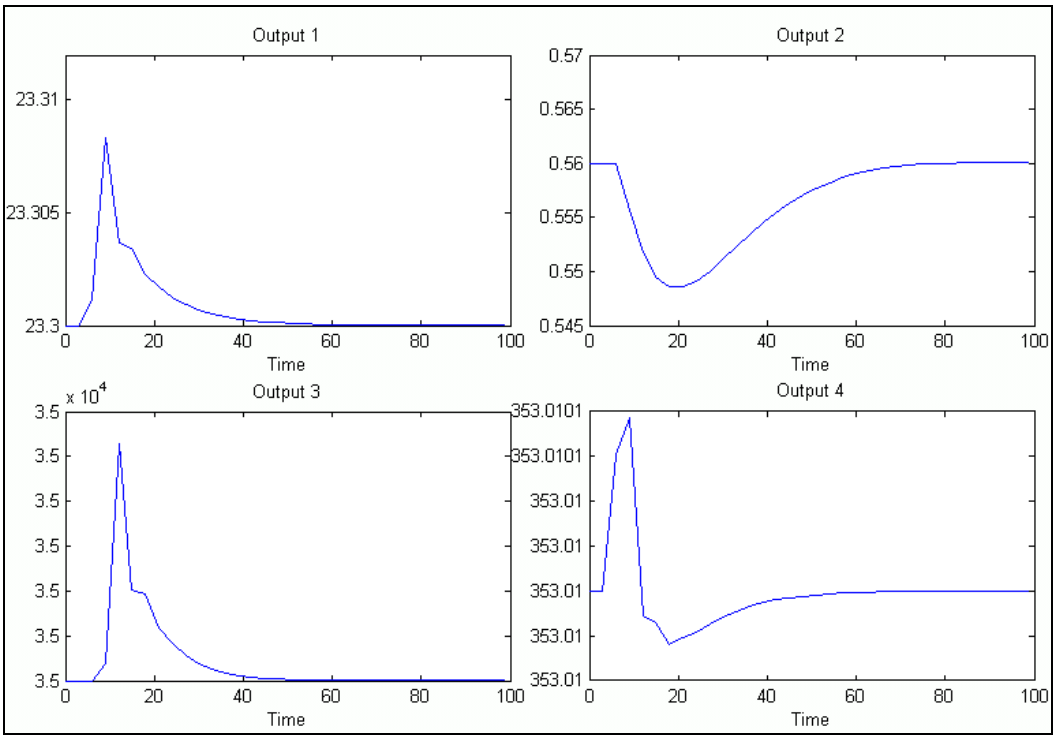

Figure 3.20 Output Changes When Disturbance Loaded (d(1) = 0.28)

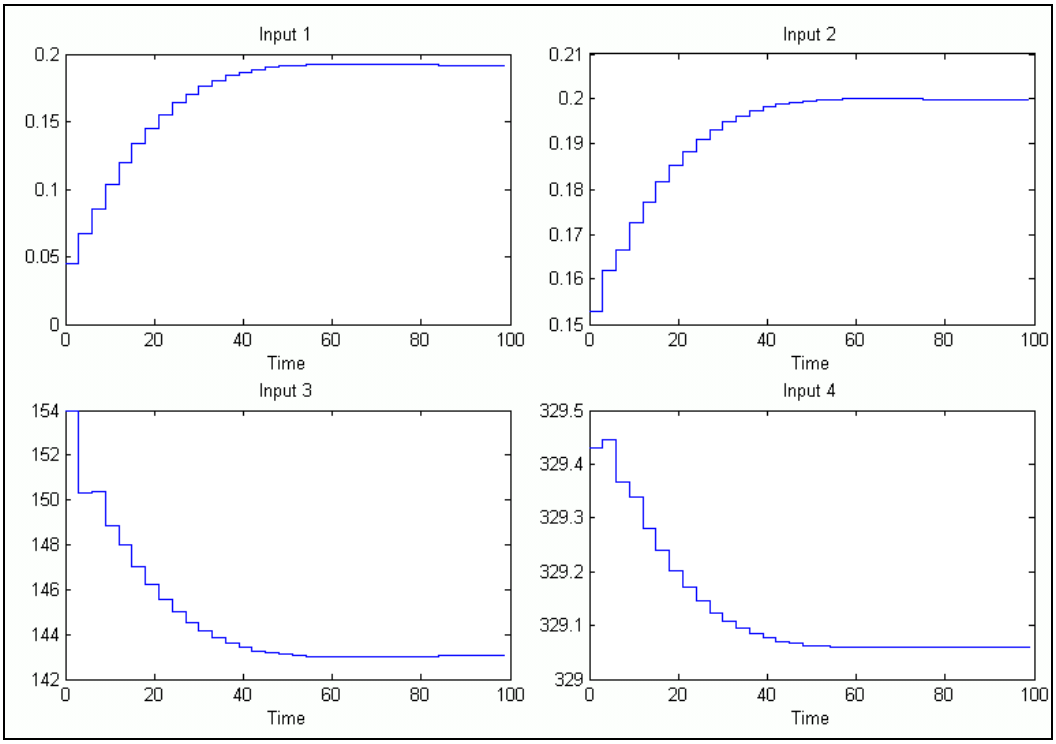

Figure 3.21 Input Changes When Disturbance Loaded (d(1) = 0.28, d(2) = 41)

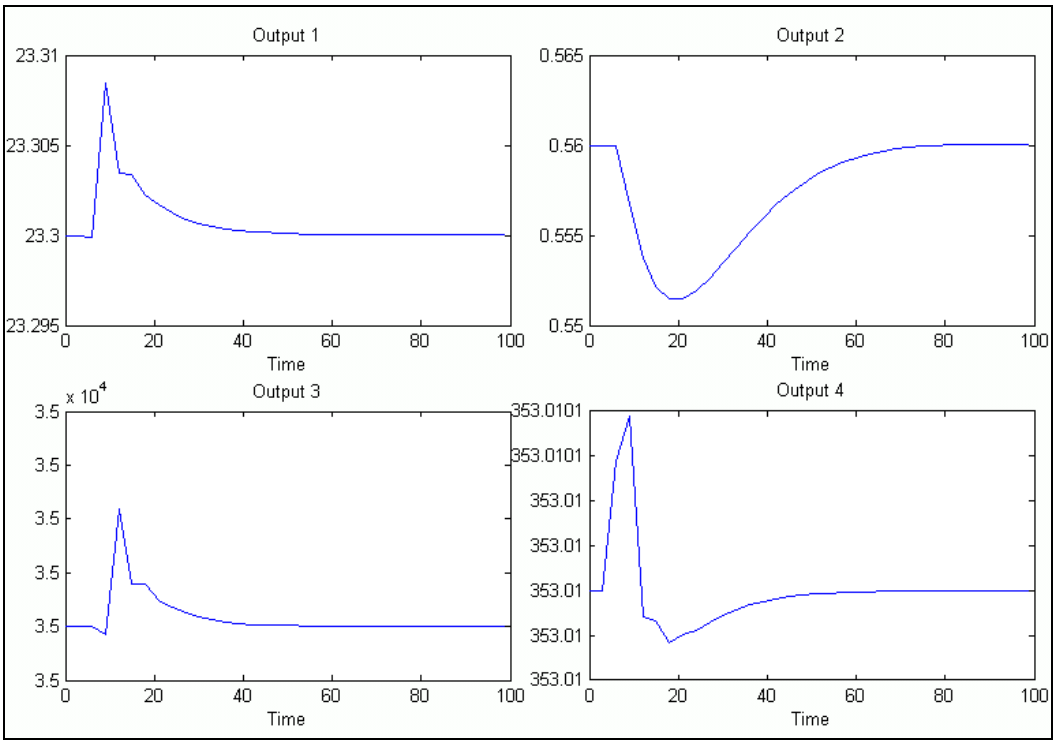

Figure 3.22 Output Changes When Disturbance Loaded (d(1) = 0.28, d(2) = 41)

CHAPTER 4

APPLICATION OF LMPC AND NNPC ON HIGH PURITY DISTILLATION COLUMN

In this chapter, the methodology is comprised of three main elements. There are the distillation column simulated with PI Control, Linear Model Predictive Control and Nonlinear Model Predictive Control. First part in this chapter describes the detail of the chosen distillation column model and the dual composition control approach used. Section two focuses on the detail of the Step Response Model derivation and the design of the DMC controller. Final section discusses mainly on the nonlinear model fitting using the FNN and the way to incorporate the FNN into MPC paradigm. A flowchart of methodology is illustrated in Figure 4.1.

4.1. Distillation Column and PI Controller

The high purity distillation column used in this work is taken form Skogestad (1997) and it is the extension of the distillation model derived form Skogestad and Morari (1988). The main reason this model is chosen instead of many other available distillation models (Luyben, 1987; Chen and Ogunnaike, 1993) is largely due to this model has been proven able to represent of a large class of moderately high purity distillation column and has been tested for numbers of researchers under various kinds of control based research activities (Chou *et* al., 2000; Skogestad *et* al., 1990; Skogestad *et* al., 1998). Skogestad provided an ODE based model which is better than the transfer function model (TF model). TF models are unable to model the highly nonlinear response of minimum reflux distillation column especially when

the operation is shifted towards high purity limit (McDonald *et* al., 1988). In addition, Skogestad performed many works on the configuration selection and controllability issues based on this model. For example, the Closed Loop Disturbance Gain analysis (CLDG) which able to define the effect of the disturbances for decentralized control when interaction taken into account has been done by Skogestad (1997) and according to him this is a better alternatives for decentralized control to deal with the effect of disturbance (Skogestad and Hovd, 1990).

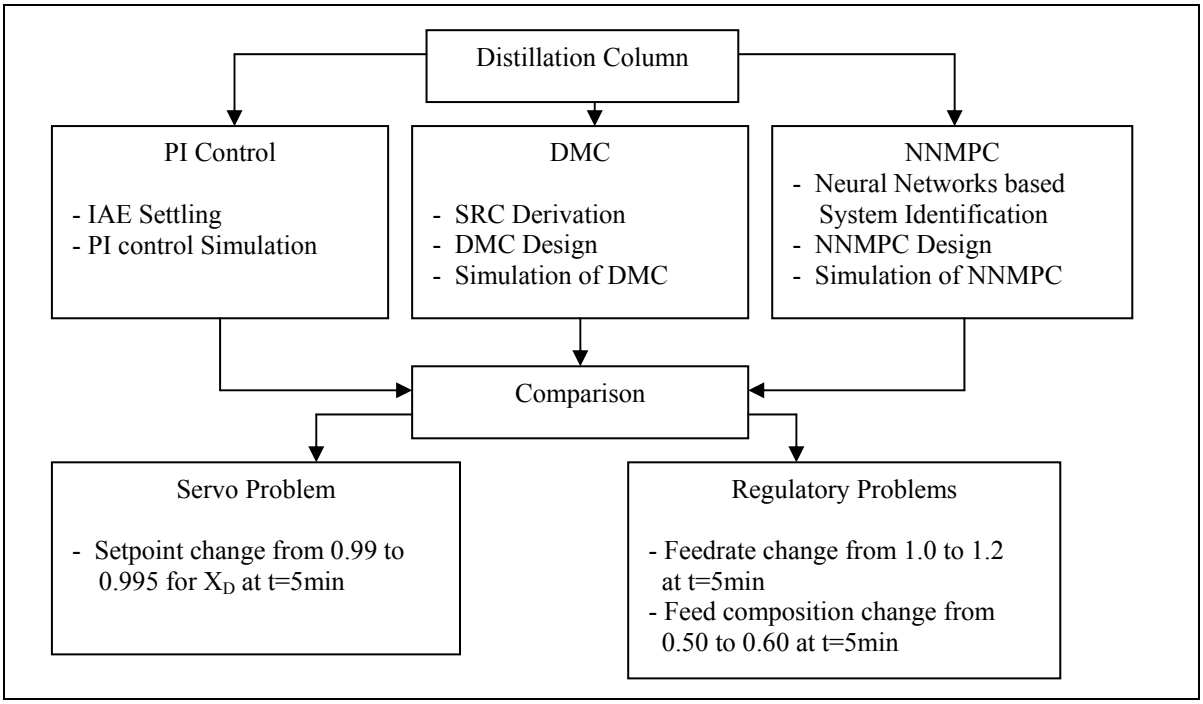

Figure 4.1 Methodology for Case Study 2

4.1.1 Formulation of Skogestad's Distillation Model

Skogestad's distillation column (1997) is an ideal, binary, 40-tray distillation column with a single feed and a total condenser. The specifications and assumptions made for this model are: i) it is a binary mixture column ii) constant pressures change over the entire tray iii) constant relative volatility iv) constant molar over flows and constant liquid holdup on all trays v) linear liquid dynamics and vapor liquid equilibrium (VLE) vi) perfect mixing on all stages vii) no vapor holdup as

well (immediate vapor response). The overall material balance based on Skogestad (1997) for all stages is described as follows:

For the feed stage: $i = N_F$

The overall material balance:

$$\frac{dM_i}{dt} = L(i+1) - L(i) + V(i-1) - V(i) + F \tag{4.1}$$

The component material balance:

$$\frac{d(M_i X_i)}{dt} = L(i+1)x(i+1) + V(i-1)y(i-1) - L(i)x(i) - V(i)y(i) + Fz_F \tag{4.2}$$

The total condenser:

The overall material balance:

$$\frac{dM_i}{dt} = V(i-1) - L(i) - D \tag{4.3}$$

The component material balance:

$$\frac{dM_i X_i}{dt} = V(i-1)y(i-1) - L(i)x(i) - Dx(i) \tag{4.4}$$

Reboiler:

The overall material balance:

$$\frac{dM_i}{dt} = L(i-1) - V(i) - B \tag{4.5}$$

The component material balance:

$$\frac{dM_i X_i}{dt} = L(i+1)x(i+1) - V(i)y(i) - Bx(i) \tag{4.6}$$

The Rest Stages:

The overall material balance:

$$\frac{dM_i}{dt} = L(i+1) - L(i) \tag{4.7}$$

The component material balance:

$$\frac{dM_i X_i}{dt} = L(i+1)x(i+1) - Vy(i-1) - L_i x_i - Vy(i) \tag{4.8}$$

The steady state data for this model is described in the Table 4.1.

Table 4.1: The steady-state data for Skogestad's column

Column Data	Operating Variables
Relative volatility, α =1.5 Number of theoretical trays, N=40 Feed tray position, N_F = 21 Feed composition Z_F = 0.5; y_D =0.99; x_B=0.01	Feed rate = 1 kmol/min Reflux flow = 2.706 kmol/min Boilup rate= 3.206 kmol/min Nominal liquid holdup = 0.5 kmol Liquid holdup for condenser = 32.1 kmol Liquid holdup for reboiler = 10 kmol Distillate flow, D = 0.5 kmol/min Bottom product flow, B = 0.5 kmol/min

4.1.2 Two Point Composition Control

The distillation column is a 5 x 5 system (five inputs flow that can be adjusted: L, V, V_T, D, B). However, at steady-state, the assumption of constant pressure and perfect level control in the condenser and reboiler reduces it into a 2 x 2 control system. A well accepted industrially control scheme, the LV control configuration is employed (Georgiou, *et* al., 1988) and this means the top product composition X_D is regulated by adjusting reflux flow, L whereas the bottom product composition is regulated directly by adjusting vapor flow V. A schematic of this control scheme is illustrated in Figure 4.2. Two single Proportional-Integral (PI) controllers would be used to perform the two point composition control. In this work, the digital PI controller is used and the digital PI controller equation in the velocity form is as follows:

$$u_t = u_{t-1} + K_C[e_t + \frac{\Delta t}{\tau_I}\sum_{k=1}^{t-1} e_k] \quad (4.9)$$

Where Δt =the sampling period

u_t = controller output at the *t* sampling instant.

e_n = error at the *t* sampling unit

Kc = Ultimate controller gain

τ_I = Integral time

The PI controller was tested and tuned for a load disturbance and a set point change. Nevertheless, a tuning procedure should be done before taking any control action. In this work, due to the nonuniqueness of Quarter Decay Ratio tuning parameter, the Integral of Absolute Value of the Error (IAE) tuning approach is used (Chiu *et* al., 1973). The IAE is a tuning approach in the aim to minimize the integral error as follow:

$$\text{IAE} = \int_0^\infty |e(t)|\, dt \quad (4.11)$$

The final tuning parameters used in testing the PI control method were obtained using the fitting method proposed by Smith (1972) and the results were as follow:

For disturbance rejection:

$$K_D=25.66 \quad \tau_D=15.80$$
$$K_B=-0.6767 \quad \tau_B=37.17$$

For setpoint change:

$$K_D= 18.56 \quad \tau_D=16.62$$
$$K_B=-0.4307 \quad \tau_B=31.32$$

After tuning the controller, the next step was to test the PI control for a load disturbance and a set point tracking. Independent tests were conducted for a 20% step change in feed composition (Z_F), a 20% step change in feed rate (F), a distillate set point step change from 0.99 to 0.995 at t=5min.

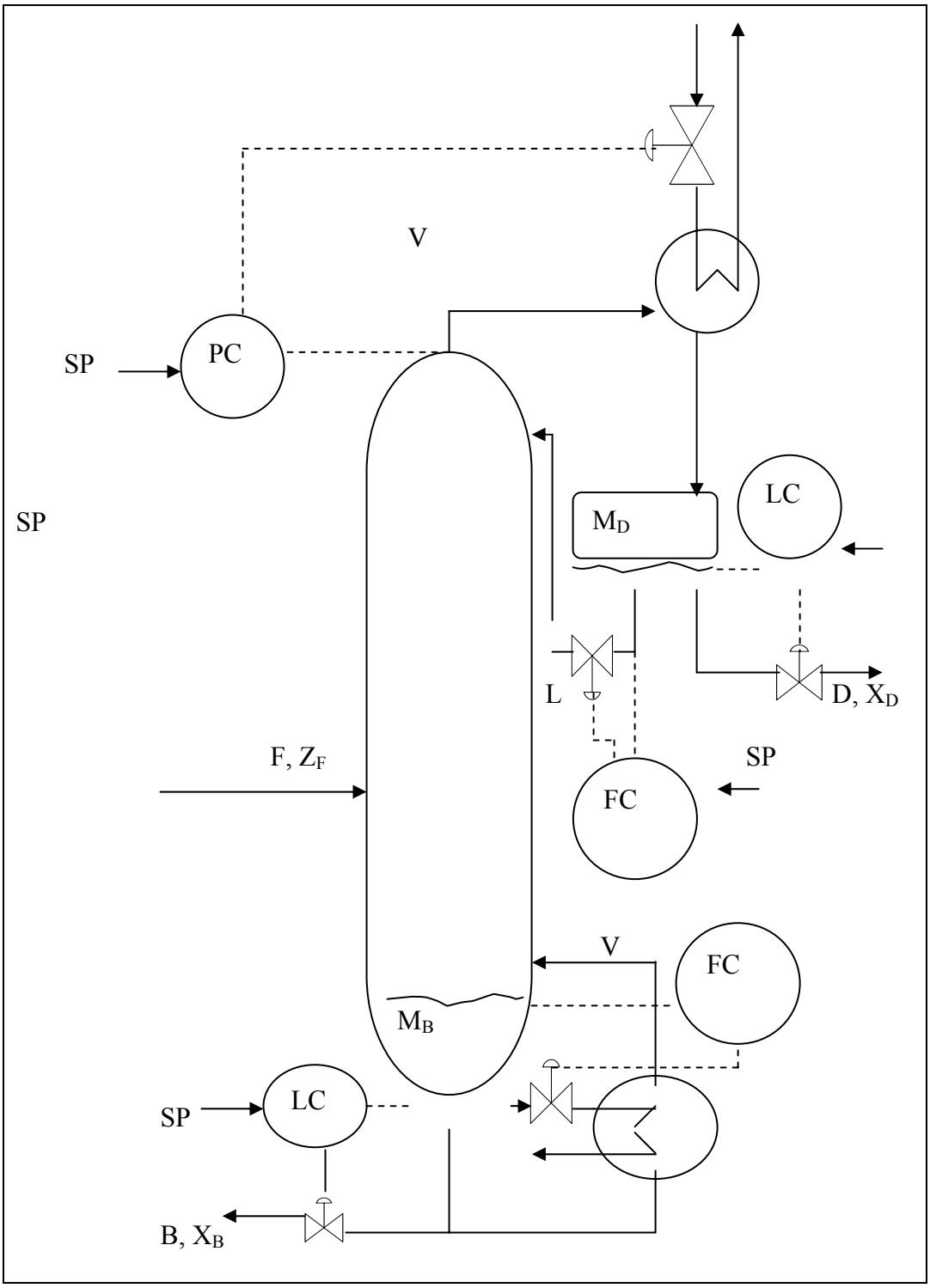

Figure 4.2 LV control scheme

4.2 LINEAR MODEL PREDICTIVE CONTROL

The LMPC algorithm chosen to be applied in this research is the DMC algorithm from Cutler and Ramaker (1979) and hence a Finite Step Response model (FSR) is required to do the prediction. Therefore, a FSR system identification procedure was performed before starting the DMC algorithm. There are three common ways to obtain the FSR including: 1) Straightforward form the step response data by the step test. 2) Indirectly form the impulse response coefficient by pulse testing. 3) Indirectly using the Orthogonal Least Square Method (OLS) form the Input Output data. In this work, a step test was employed to derive the FSR. After obtaining the Step Response Coefficients (SRC) and obtaining the dynamic matrix by arranging the SRC in specify form of matrix, the DMC was carried out. In short, the procedure the DMC algorithm is as follows:

0. Initialize the controller: calculate controller gain, *Kc*; measure the plant output y(t); set all elements of predicted output $\hat{y}(t)$ equal to y(t) (start at steady state); set the set point equal to y(t) (no set point changes)

1. Increment the sample time by one. Start to test the controller (set point tracking or disturbance rejection problems). Measure the plant output and calculate the error of the future projection of the controlled variable (CV) using:

$$e(t+1) = y_{setpo\,\mathrm{int}} - \hat{y}(t+1) \tag{4.12}$$

2. Solve the objective function, which is the minimization of the sum of error squared plus weighting for the CVs and the solution

$$\Delta U(k) = [A^{u\,T}\Gamma^T\Gamma A^u + \Lambda^T\Lambda]^{-1} A^{u\,T}\Gamma^T\Gamma e(k) \tag{4.13}$$

where Γ : diagonal weighting matrix

Λ : Diagonal move suppression matrix

k : Sampling instant

A^u : Dynamic matrix

$\Delta U(k)$: Resultant control actions

3. Implement the computed changes in the MV. This is done by adding the first change of the MV to the current MV.

4. Calculate the plant/model mismatch, *d(t)*.

5. Update the vector of the past changes in the MV and also correction of prediction by adding *d(t)* from step 4. The vector update is done by swap the values down the array and then adds new inputs changes.

6. Repeat the step from 1 to 5 to the end of simulation time.

On the other hand, the tuning parameters such as the prediction horizon, control horizon, Γ and Λ must be initialized first before starting the DMC algorithm. Moreover, in this research, the constraints for the MVs are set and being evaluated after step 6 and before step 7 (not include the constraint consideration directly into the objective function). The details in FSR derivation and DMC algorithm will be explained in next section.

4.2.1 Finite Step Response Model

The step response model is the integral of the impulse response model. It is so-named as its coefficients are the changes in the sampled process output value from its initial value in response to a unit step change in the process input. For example, in a SISO, stable process, the step response model can be presented as:

$$y_{k+j} = \sum_{i=1}^{N-1} a_i \Delta u_{k+j-1} + a_N u_{k+j-N} \qquad (4.14)$$

Where a_i is the step response coefficients and element after a_N is assumed to be constant where N is the model horizon (when process settle down).

The control system is in 2x2 and thus step test was performed twice by imposing a +5% step input from steady state in reflux, L_T and boilup rate, V_B separately to obtain 2 set of input-output data (each set of I/O data contains L_T, V_B, X_D and X_B data). The step response coefficients that relate the L_T and X_D, L_T and X_B are derived from first set data (+5% L_T change) whereas the relationship of V_B and X_D and V_B and X_B are obtained through second set data (+5% V_B change). Their relationship can be shown in Figure 4.3 below where G_{11}, G_{12}, G_{21}, and G_{22} are the relationship associated with L_T and X_D, L_T and X_B, V_B and X_D and V_B and X_B respectively.

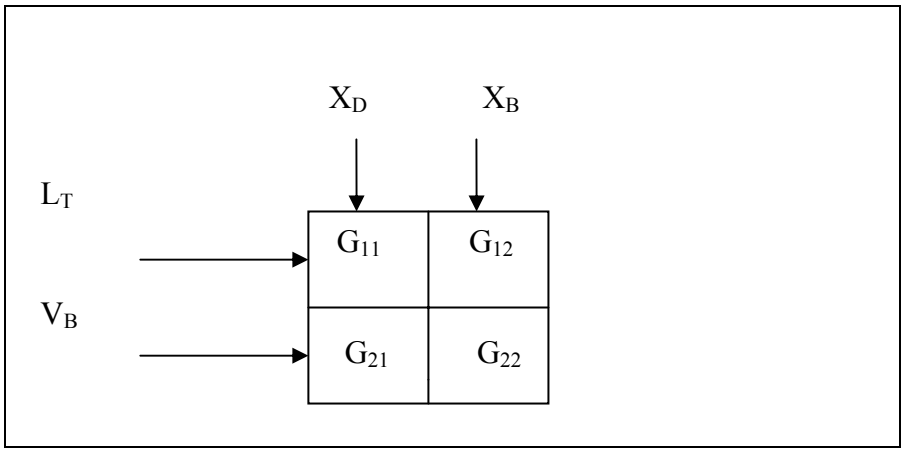

Figure 4.3 The relationship between I/O data in step test

The procedure of step test can be summarized as below:

1. Make a step change in u of a specified magnitude, Δu

$$u(t) = u_o + \Delta u \text{ for } t \geq t_o \tag{4.15}$$

2. Measure y(t) at regular intervals:

$$y_k = y(t_o + kh_s) \text{ for } k = 1,2,\ldots,N \tag{4.16}$$

where h_s : The sampling interval

Nh: Approximate time to reach steady state

3. Calculate the step response coefficient from data

$$a_k = \frac{y_k - y_0}{\Delta u} \tag{4.17}$$

4.2.2 Design of DMC Algorithm

In this research, the simulated distillation column employed does not have the constraints information. Thus, an unconstrained DMC is actually implemented in this research, instead of performing the constrained DMC like what Garcia and Morshedi (1985) did in their QDMC algorithm in which the constraints of both the MVs and CVs are taken into account directly in their objective function. In this case, only the constraints of MVs are imposed and considered after the derivation of inputs from equation 4.13. For the tuning procedures, due to the large number of adjustable parameter available in DMC, many of these parameters have overlapping effects on the closed loop performance (Shridhar, 1998). Thus, the appropriate control horizon (m) and prediction horizon (P) would be found and fixed first by trial and error. Then, the effects of moves suppression (Λ) and weighting matrix (Γ) on closed loop performance would be investigated to determine which one is more appropriate to be used as the principal (active) tuning parameters.

As a model predictive control approach, the core of the DMC is the FSR that used to compute the predicted process variable profile, $\hat{y}_r(k+n)$ for each of the R process variables with n sampling instant ahead of current time instant, k:

$$\hat{y}_r(k+n) = y_{r,0} + \underbrace{\sum_{s=1}^{S}\sum_{i=1}^{n}\{a_{rs,i}\Delta u_s(k+n-i)\}}_{\textit{Effect of current \& future moves}} + \underbrace{\sum_{s=1}^{S}\sum_{i=n+1}^{N}\{a_{rs,i}\Delta u_s(k+n-i)\}}_{\textit{Effect of past moves}} \tag{4.18}$$

where r = 1, 2, 3, …, R

$y_{r,0}$ = initial condition of r^{th} process variable

$\Delta u_s(i)$ = changes in s^{th} manipulated variable at the i^{th} sampling instant

$a_{rs,i}$ = i^{th} unit of step response coefficient of the s^{th} manipulated variable to the r^{th} process variable

N = number of samples of the past manipulated variable moves required to predict the future process behavior (normally equal to sampling time required reach steady state)

However, since the second term of equation 4.18 is undetermined yet, the equation used to do prediction is reduced to:

$$\hat{y}_r(k+n) = y_{r,0} + \sum_{s=1}^{S}\sum_{i=n+1}^{N}\{a_{rs,i}\Delta u_s(k+n-i)\} + d_r(k+n) \qquad (4.19)$$

where $d_r(k+n)$ is computed from the prediction error from the r^{th} process variables.

The assumption of $d_r(k+n)$ is also a significant feature in DMC, in which this term is used to conclude all the effect of unmeasured disturbances and plant model mismatch and normally a constant step disturbance assumption is made. For example, at the current time instant, k where the $n=0$; the prediction error can be calculated as:

$$d_r(k) = y_r(k) - \hat{y}(n) = y_r(k) - y_{r,0} + \sum_{s=1}^{S}\sum_{i=1}^{N-1}\{a_{rs,i}\Delta u_s(k-i)\} \qquad (4.20)$$

and since the future values of $d_r(k+n)$ required in equation 4.19 are not available, thus an assumption is made by estimating the prediction error to be equal over the future sampling instant or:

$$d_r(k+n) \approx d_r(k) \qquad (4.21)$$

According to Meadow and Rawling (1997), this choice of disturbance model may offers a few practical advantages including:

a) Accurately models setpoint changes that often enter feedback loop as step disturbances.
b) Effectively approximates slowly varying disturbances and this would add robustness to model error since model error can appear as slowly varying output disturbance.
c) Provides zero offset for step changes in setpoint.

For the derivation of the control variables or inputs (Garcia and Morshedi, 1986), normally the objective function used is formed by a minimization of the square of deviation of process variables set point and the predicted process output over the prediction horizon plus some control variables deviation effect as shown in Equation 4.22.

$$\min_{\Delta u} J = [r(t) - \hat{y}(t)]^T \Gamma^T \Gamma [r(t) - \hat{y}(t)] + \Delta u^T(t) \Lambda^T \Lambda \Delta u(t) \qquad (4.22)$$

where

$$r(t) = [r(t+1), r(t+2), \ldots, r(t+P)]^T$$
$$\hat{y}(t) = [\hat{y}(t+1), \ldots, \hat{y}(t+P)]^T$$
$$\Delta u(t) = [\Delta u(t), \ldots, \Delta u(t+m-1) \qquad (4.23)$$

and the weighting matrix, Γ is in $P \times P$ matrix whereas move suppression matrix, Λ is in $m \times m$ matrix and r is the defined setpoint block.

For the expression of $\hat{y}(t)$:

$$\hat{y}(t) = A\Delta u(t) + M\hat{y}(t-1) \qquad (4.24)$$

or

$$\hat{y}(t) = A\Delta u(t) + \hat{y}^P(t) \qquad (4.25)$$

and in simple matrix form

$$\begin{bmatrix} y(k+1) \\ y(k+2) \\ \text{M} \\ y(k+N) \\ y(k+N+1) \end{bmatrix} = M_I \begin{bmatrix} \tilde{y}(k-1\,| \\ \tilde{y}(k) \\ \text{M} \\ \tilde{y}(k+N-2) \\ \tilde{y}(k+N-1) \end{bmatrix} + \begin{bmatrix} S1 \\ S2 \\ \text{M} \\ S_N \\ S_{N+1} \end{bmatrix} \Delta u(k) \tag{4.26}$$

$$\text{where } M_I = \begin{bmatrix} 0 & I & 0 & \Lambda & \Lambda & 0 & 0 \\ 0 & 0 & I & 0 & \Lambda & 0 & 0 \\ \text{M} & \text{M} & \text{M} & \text{M} & \text{M} & \text{M} & \text{M} \\ 0 & 0 & \Lambda & \Lambda & \Lambda & 0 & I \\ 0 & 0 & \Lambda & \Lambda & \Lambda & 0 & I \end{bmatrix} \tag{4.27}$$

$$\text{and dynamic matrix, } A^u = \begin{bmatrix} A_1 & 0 & \ldots & 0 \\ A_2 & A_1 & 0 & \text{M} \\ \text{M} & \text{M} & A_1 & 0 \\ A_p & A_{p-1} & \ldots & A_{p-m+1} \end{bmatrix}_{Pxm} \tag{4.28}$$

where I is an identity matrix but it is merely used to represent the shift operation where such matrix does not need to be created in reality.

By substituting equation 4.25 into the equation 4.22, the objective function becomes:

$$J = [y^{ref}(t) - A^u \Delta u - \hat{y}^P(t)]^T \Gamma^T \Gamma [y^{ref}(t) - A^u \Delta u - \hat{y}^P(t)] + \Delta u^T(t) \Lambda^T \Lambda \Delta u(t) \tag{4.29}$$

or written into error function:

$$J = [E(t) - A^u \Delta u(t)]^T \Gamma^T \Gamma [E(t) - A^u \Delta u(t)] + \Delta u^T(t) \Lambda^T \Lambda \Delta u(t) \tag{4.30}$$

where $E(t) = y^{ref}(t) - \hat{y}^P(t)$

Thus, objective function is minimized by taking the derivative of equation 4.30 with respect to $\Delta u(t)$ and setting the result equal to zero:

$$0 = -2A^{u^T} \Gamma^T \Gamma E(t) + 2A^{u^T} \Gamma^T \Gamma A^u \Delta u(t) + 2\Lambda^T \Lambda \Delta u(t) \tag{4.31}$$

Solving the equation 4.31 for $\Delta u(t)$ yield:

$$\Delta u(t) = (A^{u^T} \Gamma^T \Gamma A^u + \Lambda^T \Lambda)^{-1} A^{u^T} \Gamma^T \Gamma E(t) \tag{4.32}$$

or equivalently:

$$\Delta u(t) = K_C[y^{ref}(t) - \hat{y}^P(t)] = K_C E(t) \tag{4.33}$$

Where K_c is the controller gain matrix:

$$K_c = (V^T \Gamma^T \Gamma A^u + \Lambda^T \Lambda)^{-1} A^{u^T} \Gamma^T \Gamma \tag{4.34}$$

In this work, however the system is in 2x2. Therefore, a superposition of matrix is required to be done. In the equation 4.32, the dynamic matrix, A^u would become $P \cdot 2 \times m \cdot 2$ matrix form and $\Delta u(t)$ would be in $m \cdot 2 \times 1$ matrix form; *E(t)* would be in $P \cdot 2 \times 1$ matrix form*;* $\Lambda^T \Lambda$ would be in $m \cdot 2 \times m \cdot 2$ matrix form and $\Gamma^T \Gamma$ would be in $P \cdot 2 \times P \cdot 2$ matrix form.

4.3 NONLINEAR MODEL PREDICTIVE CONTROL

Neural Network is used as nonlinear model in the Nonlinear Model Predictive Control (NMPC) scheme. The methodology of NMPC can be decomposed into two main sections: 1) Neural Network Model Fitting 2) Design of Neural Network based Model Predictive Control (NNMPC). Neural Network Model Fitting procedures include the generation of input-output (I/O) data, the determination of the neural network structure, the training of network, and validation. The ideas for the automation of distillation column studied using NNMPC originated from Brizuela *et* al. (1996), where two separate NNMPC based control loops are employed. Just like in the decentralized PI controller system, reflux ratio, L and boilup rate, V_B are the manipulated variables to control the top product composition, X_D and bottom product composition, X_B. Figure 4.4 illustrated the schematic diagram of the proposed

NNMPC strategy. The overall procedure of NNMPC algorithm is summarized as follows (same for both control loops):

0. Initial the NNMPC tuning parameters and optimization termination criterion.

1. Sample the process output, $y(t)$, and calculate the process/model mismatch, $d(t)$ between $y(t)$ and $\hat{y}(t \mid t\text{-}1)$.

2. Using the NNs process model to predict the values of the process output $\hat{y}(t \mid t+k)$ for next prediction horizon. $(k=n_1 \ldots n_2)$. Correct the prediction by adding the $d(t)$ from step 1. Calculate the cost function based on the current prediction output to provide a reference cost function that can be subtracted in calculating the control actions.

3. Calculate the sequence of manipulated variables over the control horizon, $U(t) = [u(t),\ldots,u(t + m\text{-}1)]^T$ by minimization of the defined cost function.

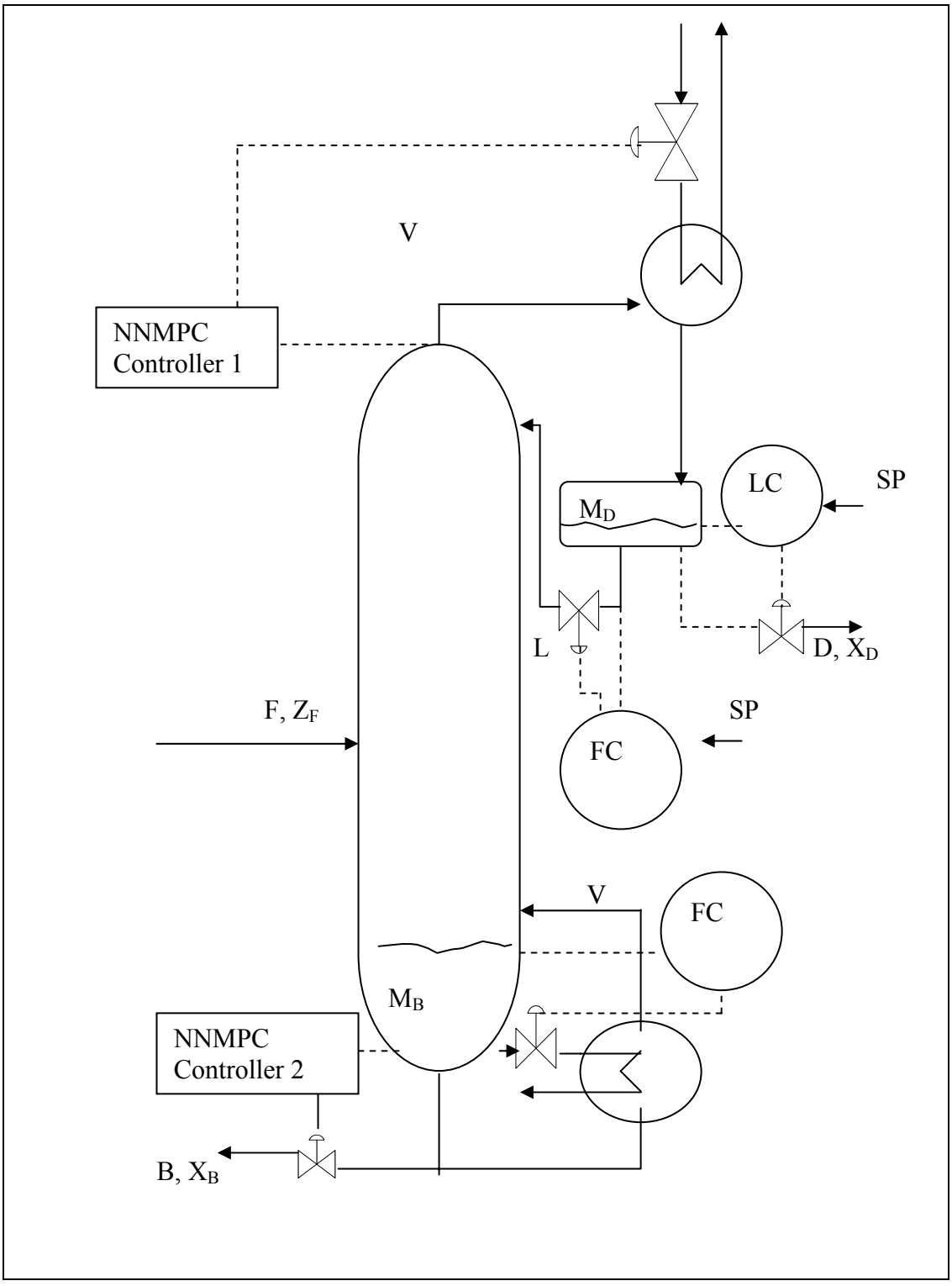

Figure 4.4 Proposed NNMPC control scheme

The cost function used in the NNMPC simulation is:

$$J(\Delta U) = \sum_{i=n_1}^{n_2} [y^{setpoint}(t+i) - \hat{y}(t+i)]^2 + \sum_{j=1}^{n_u} \rho_j [u(t+j-1) - u(t+j-2)]^2 \quad (4.35)$$

where $\Delta U = [\Delta u(t+1), ..., \Delta u(t+m-1)]^T$

n_1, $n2$ = upper limit and lower limit for Prediction Horizon

n_u = Control Horizon

ρ_j = Weighting vectors for manipulated variables

The optimization algorithm used is the Modified Levenberg Marquardt algorithm (Fletcher, 1987) and it will terminate when one of the following criterion is satisfied.

a) The iteration number reaches a pre-specified value (5 iterations). A iteration is counted when there is an improvement in new cost function ($J_{new} < J_{init}$)
b) The step size in the updating of the optimization parameters reaches a pre-specified tolerance value (1x 10^{-5}).

4. Implement the u(t) to the process model.

5. Repeat the step from 1 to 4 to the end of simulation time.

In addition, the selection of initial guess for the optimization parameters, *u(t)* is very important and in this work, and the values of *u(t)* is set by taking some manipulated variables measurement from PI control result.

For initial guesses of u_1: [2.70629, 2.80, 2.98, 3.03, 3.042, 3.049, 3.25, 3.30]
For initial guesses of u_2: [3.20629, 3.30, 3.47, 3.50, 3.540, 3.546, 3.85, 3.90]

It must be stressed here that in this NNMPC, because two separate NNMPC control loops are employed, the correction of the predicted model output to account for process/model mismatch and unmeasured disturbance can only be made in step 2 while we initialize the cost function. Thus a small process/model mismatch is

unavoidable in this control scheme. The correction is done by comparing the measured output of the process to the model prediction at *k* to generate a disturbance estimate $\hat{d} = y_k - y_k^m$, in which y_k and y_k^m represent the process measurement and model prediction respectively. Consequently, this disturbance term is added to the output prediction over the entire prediction horizon.

4.3.1 Neural Networks Model Fitting

Just like in any other nonlinear empirical model fitting approaches, the neural network model fitting is generally a procedure of determining the parameters (θ) or be more specific the weight vector that is able to map the regressor vector (φ), to output space correctly. Consider a nonlinear system,

$$\hat{y}(t \mid \theta) = g(\varphi(t), \theta) \tag{4.36}$$

where $\varphi(t) = [\ \varphi_1, \varphi_2, ...\varphi_z\ ]$ is the vector of regressor, θ is a vector containing adjustable parameter or the weights in neural network and g is the nonlinear function (function realized by neural network). In this thesis, the neural network model fitting procedures are decomposed into four main steps:

1. Data generation and data pretreatment. In the data generation, the design of input sequence is very important because the model would display different characteristics depend on how the input energy is distributed over different frequency and direction (Ljung, 1999). The Pseudo-Random Binary Sequence (PBRS) type signal is used in this research as an input signal to excite the plant. This is due to the PBRS- type signals are more persistent exciting (PE) and able to provide more information about the process dynamic compare to other common-used input signals. For example, the step input signal emphasizes too much on the low-frequency excitation whereas the excitation of pulse signal is too widely spread over the whole frequency ranges. The PBRS signal is generated by imposing a + 5% deviation from its steady-state values in inputs signal (2.70629 and 3.20629 for u_1 and u_2

respectively) and 200 set of I/O data is logged with the sampling time of 2 min. In addition, the data pre-treatment is also necessary before the starting of network training. In this work, each element of an I/O data is scaled between 1 to 0 to avoid the saturation of the activation function in every neuron and the scaled I/O data vector, v_{norm} is presented as:

$$v_{norm} = \left(\frac{v - v_{\min}}{v_{\max} - v_{\min}} \right) \tag{4.37}$$

where v_{max} and v_{min} are the declared maximum and minimum of the training vector.

2. Network structure selection. A two-layer perceptron NNs with hyperbolic tangent hidden units and linear output is used in this work due to it has proven that it is good enough to approximate all the continuous functions to any desired accuracy (Ramchandram and Rhinehart, 1995; Hornik *et* al., 1989). Figure 4.5 illustrate the proposed MLP structure *d* inputs, *M* hidden neurons and *c* output neuron.

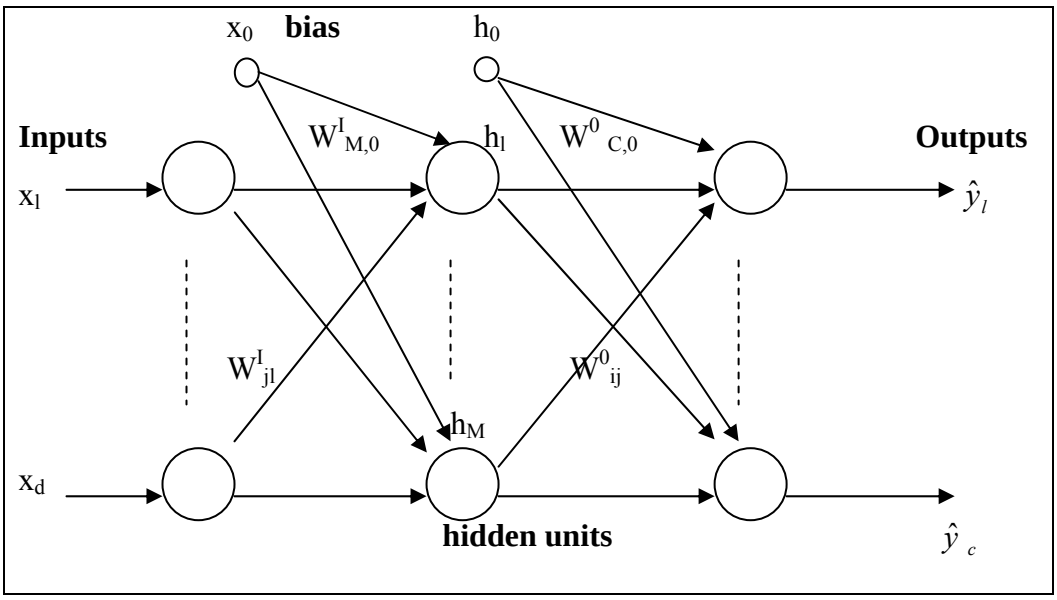

Figure 4.5 The architecture of MLP network

where the processing in the network can be expressed as (Baughman and Liu, 1995):

$$h_j(x) = \psi(\sum_{l=1}^{d} w^I{}_{jl} x_l + w^I{}_{j,0}) \qquad \text{(for hidden layer)} \qquad (4.38)$$

$$\hat{y}_i(x) = \psi(\sum_{j=1}^{M} w^0{}_{ij} h_j + w^0{}_{i,0}) \qquad \text{(for output layer)} \qquad (4.39)$$

The $\boldsymbol{x} = [1, x_1, ..., x_d]$ is the input vector and $\psi(u)$ is a nonlinear activation function**.** In other words, the activation function with respect to the input to hidden weights, $\boldsymbol{\psi(u_i^0(k))}$ is a hyperbolic tangent and the activation function with respect to the hidden to output weights, $\boldsymbol{\psi(u_j^I(k))}$ is an linear function The weights form input l to hidden neuron j and the weights form the hidden neurons j to the output i is denoted as w^I_{jl} and w^0_{ij} respectively. The biases weights are denoted as $w^i_{j,0}$ and $w^0_{i,0}$.

To express it in simpler matrix form, where:

$$\mathbf{h} = \boldsymbol{\psi}\,(\mathbf{W}^{\mathbf{I}} \cdot \mathbf{x}) \qquad (4.40)$$

and $\mathbf{y^{pred}} = \boldsymbol{\psi}\,(\mathbf{W}^0 \cdot \mathbf{h}) = \mathbf{f(x,w)}$ (4.41)

where $\boldsymbol{W}^I$ is the (M, d+1) input-hidden weight matrix and $\boldsymbol{W}^0$ is the (c, M+1) hidden output weight matrix and $\mathbf{x} = \{x_l\}$ is the (d+1, 1) input vector and $\mathbf{h} = \{h_l\}$ is the (M+1,1) hidden vector with $h_0 \equiv 1$ and $\mathbf{y^{pred}} = \{\hat{y}_i\}$ is the (c, 1) output vector and the element by element vector activation is given by $\boldsymbol{\psi}$ **(u)** = $[\psi(u_1),\ldots,\psi(u_n)]$. In short, this network structure used can be viewed as a nonlinear function **f(x, w)** of the input vector and weight vector **w**, which contains all the weights.

On the other hand, for the network's input vector (**x**) or regressor vector selection, a Nonlinear Auto-Regressive with eXogenous input (NARX) model is employed and the regressor vector can be expressed as (Leontaritis and Billings, 1985):

$$\varphi(t,\theta) = [(y(t-1),...,y(t-n_y).u(t-k),..,u(t-k-n_u+1)]^T \qquad (4.42)$$

where the $y(t)$ and $u(t)$ are the system output and input sampled at time t, $\varphi(t,\theta)$ is the regressor vector and θ is the adjustable parameter or networks weight in this case,

k is the system dead time and n_u and n_y are the number of the delayed system inputs and output included in the model. The NARX model is chosen due to it has shown to perform well for predicting system outputs and has been successfully used for controlling nonlinear system (Ravi Sriniwas *et* al., 1995; Pottmann and Seborg, 1992). In this research, both of the values for n_u and n_y are set at 2 as the chosen distillation column can be defined well under approximation of 2nd order system. The schematic diagram of NARX structure is shown in Figure 4.6 below.

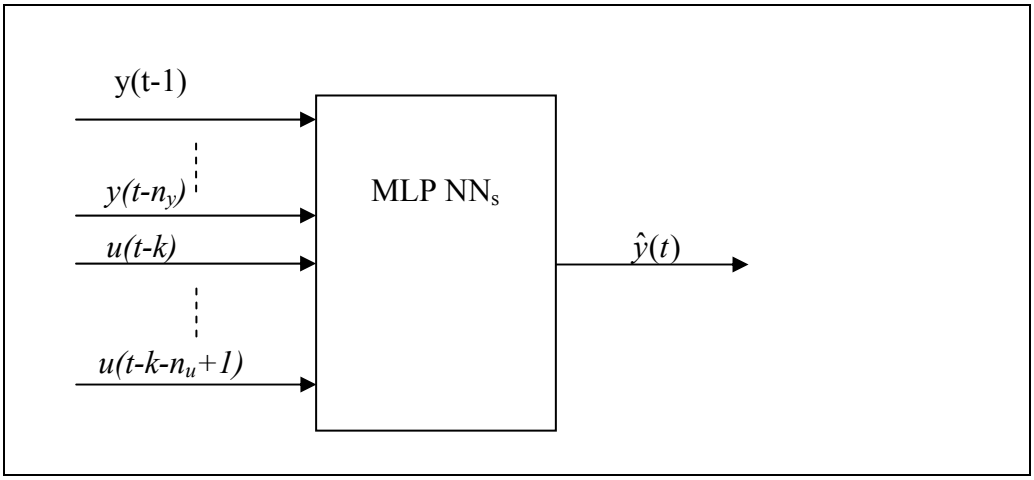

Figure 4.6 The NARX structure

3. Network Training. Again, the NN_S training is actually a process to determine the optimal weights (the vector or matrices W^I and W^0) for a prediction problem and it is done by performing the data mapping from a set of training data $Z^N = \{[u(t), y(t)] \mid t=1,..,N_{train}\}$, to a set of possible weights: $Z^N \rightarrow \hat{\theta}$ with the objective to produce the prediction $\hat{y}(t)$ which is close to the true outputs $y(t)$. Normally, a cost function is pre-defined as a measure of the quality of the output prediction. In this thesis, a classic Prediction Error Method (PEM) is used (Ljung, 1999) and the cost function is as usual, the sum of the squared differences between the networks predicted output $\hat{y}(t)$ and the expected output $y(t)$:

$$J(\theta) = \frac{1}{2}\sum_{t=1}^{N_{train}}[y(t) - \hat{y}(t \mid \theta)]^2 = \frac{1}{2}\sum_{t=1}^{N_{Train}}(\varepsilon^i(\theta))^2 \qquad (4.43)$$

However, the ultimate cost function used in this thesis is augmented with a regularization term as follow:

$$V(\theta) = \frac{1}{2N_{Train}} \sum_{t=1}^{N_{train}} [y(t) - \hat{y}(t \mid \theta)]^2 + \frac{1}{2N_{Train}} \theta^T D \theta \tag{4.44}$$

where the matrix D is a diagonal matrix which is commonly selected to $D = \alpha I$ and in this thesis the α is set to $1x10^{-5}$. This approach is known as weight decay regularization method which is used to smooth the cost function and improving the generalization error by eliminate over-fitting and ensure numerical stability (Sjoberg and Ljung, 1995). Finally, the optimal weights are found as:

$$\theta = \arg\min_{\theta} V(\theta) \tag{4.45}$$

by employing some iterative optimization algorithm:

$$\theta_{i+1} = \theta_i - \eta \nabla J_{ac}(V) \tag{4.46}$$

where η is the step size and $\nabla Jac(V)$ is the gradient respective to cost function.

The Gauss-Newton based Levenberg-Marquardt algorithm (Fletcher, 1987) is used as the network training algorithm in this thesis due to its rapid convergence properties and robustness. As usual, the terminal criterions are required to pre-define before starting the training algorithm. These include:

a) The maximum iteration number (500 iterations).
b) The minimum acceptable criterion value ($1x\ 10^{-10}$)
c) The minimum of criterion difference (previous criterion value minus current criterion value) ($1x\ 10^{-7}$).

The network training method used in this thesis is originated from Norgaad (1997), Gaiceanu *et* al., (2000) and Xia (2003). The summary of the procedure is as follow:

0. Set the necessary termination criterions and weight decay diagonal matrix, D.
1. Initial the weight vector, W^1 and W^0 (starting point). Run the simulation with the initial weight vector and calculated the information required like the predicted output, $\hat{y}(t)$ error vector, ε and the value of pre-defined cost function, $V(\theta)$.
2. set iteration index, k=0, Let LM parameter, $\lambda^0 = 10^{-3}$
3. Calculate J_{ac}, the Jacobian matrix in respect to the defined criterion in Eq (4.44).
4. Calculate the search direction, f from $[R(\theta^{(i)}) + \lambda^{(i)} I] f^{(i)} = -G(\theta^{(i)})$, where I being a unit matrix, R and G first and second partial derivatives of $V(\theta)$, λ is a small positive diagonal element added to matrix of second partial derivative or so-called Hessian matrix to overcome the non-positive definite problem arising in Hessian matrix.
5. Calculate the $W(\theta^{(i)} + f^{(i)}, Z^N)$
6. Calculate the term $r = \dfrac{W(\theta^{(i)}, Z^N) - W(\theta^{(i)} + f^{(i)}, Z^N)}{W(\theta^{(i)}, Z^N) - L^{(i)}(\theta^{(i)} + f^{(i)})}$

   where $W(\theta^{(i)}, Z^N) - L^{(i)}(\theta^{(i)} + f^{(i)}) = \dfrac{1}{2N_{Train}}[(-f^{(k)})^T G(\theta^{(k)}) + \lambda^{(k)} \left\| f^{(k)} \right\|^2 + D \left\| f^{(k)} \right\|^2]$
7. Set $\lambda^{(i)} = \lambda^{(i)} / 2$ if $r^{(i)} > 0.75$ and Set $\lambda^{(i)} = 2\lambda^{(i)}$ if $r^{(i)} > 0.25$
8. If $W(\theta^{(i)} + f^{(i)}, Z^N) - W(\theta^{(i)}, Z^N)$ then accept $\theta_{i+1} = \theta_i + f^{(i)}$ as new iterate and let $\lambda^{(i+1)} = \lambda^{(i)}$ and increment k by one, $k=k+1$
9. Calculate the stooping criterion, if satisfy, terminate. If no, go to step 4.

For updating the weight vector (**w**) at each iteration as in Equation 4.46, the first derivative of cost function with respect to the weight, $\dfrac{\partial J_T(w)}{\partial w_i}$ in each iteration

is required. In this thesis, it is calculated by updating the weight vector, **w** through the back propagation techniques in which the error signal, *ε(k)* is propagated backward through the network via the error every individual neurons.

The gradient of the cost function can be written in the form of:

$$\nabla J_T = \frac{\partial J_T(w)}{\partial w_i} = \frac{\partial V(w)}{\partial \varepsilon(k)} \frac{\partial \varepsilon(k)}{\partial \hat{y}_i(k)} \frac{\partial \hat{y}_i(k)}{\partial w_i} \tag{4.47}$$

Where $\frac{\partial V(w)}{\partial \varepsilon(k)} = \frac{1}{N_{Train}} \sum_{k=1}^{N_{tarin}} \varepsilon(k)$ and $\frac{\partial \varepsilon(k)}{\partial \hat{y}_i(k)} = -1$ (4.48)

By differential with respect to the hidden to output weight gives:

$$\frac{\partial \hat{y}_i(k)}{\partial w_{ij}^{\,0}} = \boldsymbol{\psi}'(\mathbf{u_i^0(k)})\ \mathbf{h_j(k)} \tag{4.49}$$

This lead to

$$\frac{\partial J_T(w)}{\partial w_i} = -\frac{1}{N_{Train}} \sum_{k=1}^{N_{tarin}} \varepsilon(k)\, \boldsymbol{\psi}'(\mathbf{u_i^0(k)})\ \mathbf{h_j(k)} \tag{4.50}$$

Whereas the derivatives with respect to the input to hidden weights are found by using the chain rule:

$$\frac{\partial \hat{y}_i(k)}{\partial w_{ij}^{\,0}} = \frac{\partial \hat{y}(k)}{\partial h_j(k)} \frac{\partial h_j(k)}{\partial w_{ji}^{\,I}} = \boldsymbol{\psi}'(\mathbf{u_i^0(k)})\ \mathbf{w_{ij}^0}\, \boldsymbol{\psi}'(\mathbf{u_j^I(k)})\ \mathbf{x_I(k)} \tag{4.51}$$

And this has eventually yield the ∇J_T wanted in the form of:

$$\frac{\partial J_T(w)}{\partial w_i} = -\frac{1}{N_{Train}} \sum_{k=1}^{N_{tarin}} \varepsilon(k)\, \boldsymbol{\psi}'(\mathbf{u_i^0(k)})\ \mathbf{w_{ij}^0}\, \boldsymbol{\psi}'(\mathbf{u_j^I(k)})\ \mathbf{x_I(k)} \tag{4.52}$$

Due to there are two separate NNMPC controllers used (as mentioned in section 4.3), the NNs model fitting procedure also has to be done in two times (L and X_D; V and X_B) or in others words the SISO system identification is employed. Nevertheless, the I/O data used is taken from MIMO PBRS tests in which both inputs variables are changing its values simultaneously.

4. Validation. In this step, the available data (from data generation) irrespective of input and output is divided into two subsets. The first subset is the training set, which is used for computing the gradient and updating the network weights and biases. The second subset is used as the validation set. Because the principal role of identified NNs is to predict the process output over the prediction horizon, thus, the accuracy of NNs is validated by evaluating its performance in doing the multi-step prediction. Five samples of test are conducted including 1, 2 3, 4 and 5 step ahead prediction and the performance index used is the Sum of Squared Error (SSE) between predicted output, $\hat{y}$ and true output, y.

$$SSE = \sum_{t=1}^{N} \hat{y}(t) - y(t) \tag{4.53}$$

where N is the overall samples used for cross validation which is 200 in this work.

4.3.2 The Design of Neural Network based Model Predictive Control

The Neural Network based Model Predictive Control (NNMPC) algorithm implemented here is originated from the work of Sorensen *et* al. (1999). They had introduced a new idea of developing a controller with extended control horizon based on a neural network. For the predictor, they suggested the k-step prediction by using a NARX model:

$$\hat{y}(t+k) \equiv \hat{y}(t+k) = g(\hat{y}(t+k-1),...,\hat{y}(t+k-\min(k,n_y)),$$
$$y(t-1),...,y(t-\max(n_y-k,0))$$
$$u(t-d+k),...,u(t-d-n_u+k)) \quad (4.54)$$

where the n_u and n_y are the number of the delayed system inputs and output and the d is the system dead time. In other words, the k-step ahead prediction is calculated by shifting the expression forward in time while substituting prediction for actual measurement which does not exist.

As denoted in Equation 4.35, for the NNMPC scheme, the criterion or cost function to be minimized is defined as:

$$J(t,U(t)) = \sum_{i=N_1}^{N_2}[r(t+i)-\hat{y}(t+i)]^2 + \rho\sum_{i=1}^{N_u}\Delta u(t+i-1)^2 \quad (4.55)$$

or in the matrix form:

$$J(t,U(t))[R(t)-\hat{Y}(t)]^T[R(t)-\hat{Y}(t)] + \rho\tilde{U}(t)^T\tilde{U}(t) = E(t)^T E(t) + \rho\tilde{U}(t)^T\tilde{U}(t) \quad (4.56)$$

where $R(t) = [r(t+N_1)...r(t+N_2)]^T$

$$\hat{Y}(t) = [\hat{y}(t+N_1)...\hat{y}(t+N_2)]^T$$
$$E(t) = [e(t+N_1)...e(t+N_2)]^T$$
$$\tilde{U}(t) = [\Delta u(t)...\Delta u(t+N_u-1)]^T \quad (4.57)$$

and $e(t+k) = r(t+k) - \hat{y}(t+k_1)$ for $k=N_1,...,N2$ (4.58)

Just like the neural network model fitting problem, the quasi-Newton optimization algorithm or more specifically the Gauss-Newton based Levenberg-Marquardt algorithm is employed here to determine the control law by minimizing the NNMPC cost function (Equation 4.55). To solve this problem, the calculation of the gradient is necessary and this has caused the most difficult part in this NNMPC algorithm. The gradient is given by:

$$G(U^{(i)}(t)) = \frac{\partial J(t,U(t))}{\partial U(t)} = \left[-2\frac{\partial \hat{Y}(t)}{\partial U(t)} E(t) + 2\rho \frac{\partial \widetilde{U}(t)^T}{\partial U(t)} \widetilde{U}(t) \right] \tag{4.59}$$

where the partial derivatives of $\frac{\partial \widetilde{U}(t)^T}{\partial U(t)}$ is given by:

$$\frac{\partial \widetilde{U}(t+k)^T}{\partial U(t+k)} = \begin{bmatrix} \frac{\partial \Delta U(k)}{\partial U(k)} & \frac{\partial \Delta U(k)}{\partial U(k+1)} & \Lambda & \frac{\partial \Delta U(k)}{\partial U(k+N_U-1)} \\ \frac{\partial \Delta U(k+1)}{\partial U(k)} & \frac{\partial \Delta U(k+1)}{\partial U(k+1)} & \Lambda & \frac{\partial \Delta U(k+1)}{\partial U(k+N_u-1)} \\ \text{M} & \text{M} & \text{O} & \text{M} \\ \frac{\partial \Delta U(k+N_u-1)}{\partial U(k)} & \frac{\partial \Delta U(k+N_u-1)}{\partial U(k+1)} & \Lambda & \frac{\partial \Delta U(k+N_U-1)}{\partial U(k+N_U-1)} \end{bmatrix} \tag{4.60}$$

and since the $\widetilde{U}(t) = \Delta u(t) = u(t) - u(t-1)$

$$\frac{\partial \widetilde{U}(t+k)^T}{\partial U(t+k)} = \begin{bmatrix} 1 & 0 & \Lambda & \Lambda & 0 \\ -1 & 1 & \Lambda & \Lambda & \text{M} \\ \text{M} & \text{O} & \text{O} & \text{O} & \text{M} \\ \text{M} & \text{O} & \text{O} & \text{O} & \text{M} \\ 0 & \Lambda & \Lambda & -1 & 1 \end{bmatrix} \tag{4.61}$$

The $\frac{\partial \widetilde{U}(t+k)}{\partial U(t+k)}$ is a matrix of dimension *Nu x Nu* and this derivative is independent of time and constructed beforehand. So, the unsolved term left now is only the partial derivative of $\frac{\partial \hat{Y}(t)}{\partial U(t)}$ which is a matrix of dimension $N_U \,\text{x}\, (N_2 - N_1 + 1)$:

$$\frac{\partial \hat{Y}(t)^T}{\partial U(t)} = \begin{bmatrix} \frac{\partial \hat{y}(t+N_1)}{\partial u(t)} & \Lambda & \frac{\partial \hat{y}(t+N_2)}{\partial u(t+N_u-1)} \\ \text{M} & \text{O} & \text{M} \\ \frac{\partial \hat{y}(t+N_2)}{\partial u(t)} & \Lambda & \frac{\partial \hat{y}(t+N_2)}{\partial u(t+N_u-1)} \end{bmatrix} \tag{4.62}$$

To calculate the partial derivative of $\frac{\partial \hat{Y}(t)}{\partial U(t)}$ in where:

$$\frac{\partial \hat{Y}(t)}{\partial U(t)} = \frac{\partial \hat{Y}(t)}{\partial h(t)} \frac{\partial h(t)}{\partial U(t)} \tag{4.63}$$

The hidden output is calculated by taking into account for the past and future control inputs terms (first three sums depend on future control input and the remaining three on past control input):

$$\begin{aligned} \widetilde{h}(k,j) = & \sum_{i=1}^{\min(k-d,n)} w_{ji}\hat{y}(t+k-i) + \sum_{i=1}^{\min(k-d-N_u+2,m+1)} w_{j,n+i}u(t+N_u-1) \\ & + \sum_{i=k-d-N_u+2}^{\min(k-d,m)} w_{j,n+i+1}u(t-d+k-i) + \sum_{i=k-d+1}^{\min(k,n)} w_{ji}\hat{y}(t+k-i) \\ & + \sum_{i=k+1}^{n} w_{ji}y(t+k-i) + \sum_{i=k-d+1}^{m} w_{j,n+i+1}u(t-d+k-i) + w_{j0} \end{aligned} \tag{4.64}$$

For $\forall k \in [N_1, N_2]$ and $\forall l \in [0, \min(k-d, N_u-2)]$, assume

$$\frac{\partial \hat{y}(t+k)}{\partial u(t+l)} = \sum_{j=1}^{n_h} W_j f'(\widetilde{h}(k,j)) \frac{\partial \widetilde{h}(k,j)}{\partial u(t+l)}$$

$$= \sum_{j=1}^{n_h} W_j f'(\widetilde{h}(k,j)) h(k,l,j) \tag{4.65}$$

$$\text{Where } f(x) = \tanh(x) \rightarrow f'(x) = 1 - (f(x))^2 \tag{4.66}$$

$$\text{and } h(k,l,j) = \sum_{i=1}^{\min(k-d,n)} w_{ji} \frac{\partial \hat{y}(t+k-i)}{\partial u(t+l)} + \sum_{i=1}^{\min(k-d-N_u+2,m+1)} w_{ji} \frac{\partial u(t+N_u-1)}{\partial u(t+l)}$$

$$+ \sum_{i=k-d-N_u+2}^{\min(k-d,m)} w_{j,n+i+1} \frac{\partial u(t-d+k-i)}{\partial u(t+l)} \tag{4.67}$$

$$\text{since } \frac{\partial u(t+N_u-1)}{\partial u(t+l)} = \begin{cases} 1 & l = N_u - 1 \\ 0 & otherwise \end{cases} \tag{4.68}$$

$$\frac{\partial u(t-d+k-i)}{\partial u(t+l)} = \begin{cases} 1 & l = k-d-i \\ 0 & otherwise \end{cases} \tag{4.69}$$

$$\frac{\partial \hat{y}(t+k-i)}{\partial u(t+l)} = 0 \quad l \geq k-d-i+1 \tag{4.70}$$

The expression for *h(k,l,j)* can be reduced to

$$h(k,l,j) =$$

$$\begin{cases} \sum_{i=1}^{\min(k-d-l,n)} w_{ji} \frac{\partial \hat{y}(t+k-i)}{\partial u(t+l)} + \sum_{i=1}^{\min(k-d-N_u+2,m+1)} w_{j,n+i} & l = N_u \\ \sum_{i=1}^{\min(k-d-l,n)} w_{ji} \frac{\partial \hat{y}(t+k-i)}{\partial u(t+l)} + w_{j,n+k-d-1+1} & \max(0,k-d-m+1) \leq l \leq N_u - 1 \\ \sum_{i=1}^{\min(k-d-l,n)} w_{ji} \frac{\partial \hat{y}(t+k-i)}{\partial u(t+l)} & 0 \leq l \leq \max(0,k-d-m+1) \end{cases} \tag{4.71}$$

After obtaining the $\frac{\partial \hat{Y}(t)}{\partial U(t)}$ from equation 4.65, the Gauss-approximate Hessian and Gradient is calculated and the Gauss-Newton Leverberg Marquardt approach is applied to calculate the control action by minimizing the cost function following the same procedure as denoted in Section 4.31.

CHAPTER 5

RESULTS AND DISCUSSION

Results are presented and discussed in this chapter. The first section reports the system identification results for both the DMC and NNMPC. For DMC, the step response coefficients derived from step test is presented whereas for the NNMPC, the neural network system identification results for both the control loops; X_D and L_T as well as X_B and V_B is presented respectively. Second section illustrated and discussed the closed loop simulation results for PI, DMC and NNMPC controller in dealing with servo and regulatory problems. Additionally, the selection of tuning parameters in DMC and NNMPC is discussed in this section. Finally, the comparison between PI, DMC and NNMPC controller is shown.

5.1 System Identification

To obtain the SRC, two step tests are performed with 5% step change in L_T and V_B respectively to obtain a_{11} and a_{12} (step response coefficient associated with G_{11} and G_{12}) and a_{21} and a_{22} (step response coefficient associated with G_{21} and G_{22}). By using equation 3.17, the step response coefficients for a_{11}, a_{12}, a_{21}, and a_{22} are calculated.

For the NNMPC, two separate feed-forward neural networks are employed to represent the nonlinear systems (X_D and L_T and X_B and V_B). These two NN_S are trained as expressed in section 3.31. The training I/O data used is generated by using the PBRS type input signal as illustrated in Figures 5.1 and 5.2. By varying the

hidden nodes used in both NN_S and cross validating the trained NN_S with multi-step prediction test, the NN_S training results are obtained.

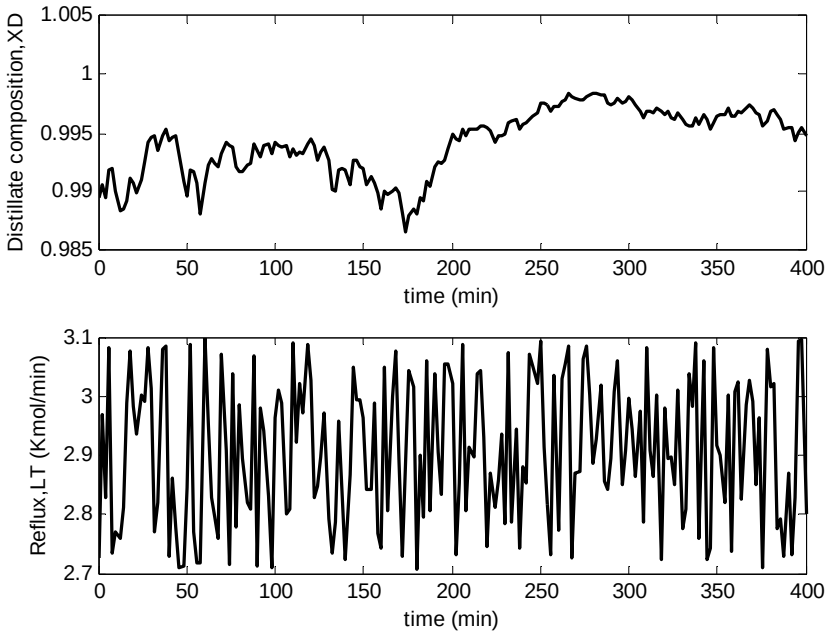

Figure 5.1 Training data used to model the nonlinear system which correlates L_T and X_D

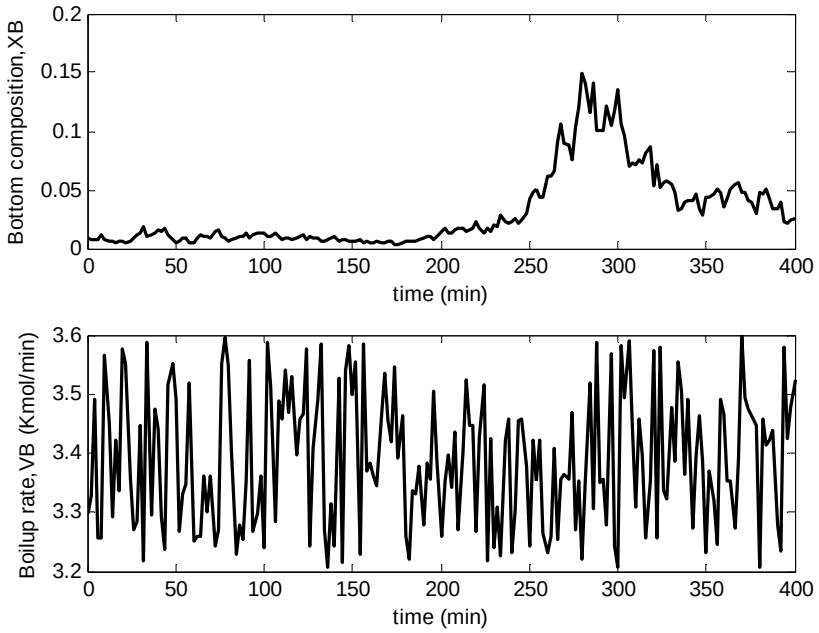

Figure 5.2 Training data used to model the nonlinear system which correlates V_B and X_B

5.2 Closed Loop Simulation Results

In this section, the performance of PI, DMC and NNMPC controller is assessed by testing them through the servo and regulatory problem. For the servo problem, a step setpoint change for X_D was imposed at *t=5 min* from 0.99 to 0.995. Two types of disturbances were investigated for regulatory problem: a 20% step change in feed rate, F (from 1.00 kmol/min to 1.20 kmol/min) and a 20% step change in feed composition, Z_F (from 0.50 to 0.60). In addition, to make the simulated distillation column model closer to real column, a 1 min measurement delay for X_D and X_B is included in the simulation. The procedure and rationale used for tuning in DMC and NNMPC are discussed in this section.

5.2.1 PI Control Result

The two-point composition control strategy as described earlier in section 3.1.2 is implemented to control the distillation column. PI controller parameters are taken from IAE tuning approach (Chiu *et* al., 1973). Figure 5.3 show the control result for PI controller based on IAE tuning rules for the set point tracking whereas Figures 5.5 and 5.6 illustrated the control results for unmeasured disturbances in feed rate and feed composition respectively.

For the setpoint tracking, as can be seen from Figure 5.3, PI controller based IAE setting show a rapid responds but very sluggish control actions. The Reflux rate (L_T) and Boilup rate (V_B) were increased once the setpoint change is introduced with a smooth but small magnitude profile. As a consequence, the control results for IAE setting were evident lead to a long settling time. This can be seen clearly in particularly for the bottom composition, X_B from Figure 5.3 in which at the end of the simulation (t=600 min), the X_B from the IAE tuning rules is still unable to settle down to its setpoint at 0.01.

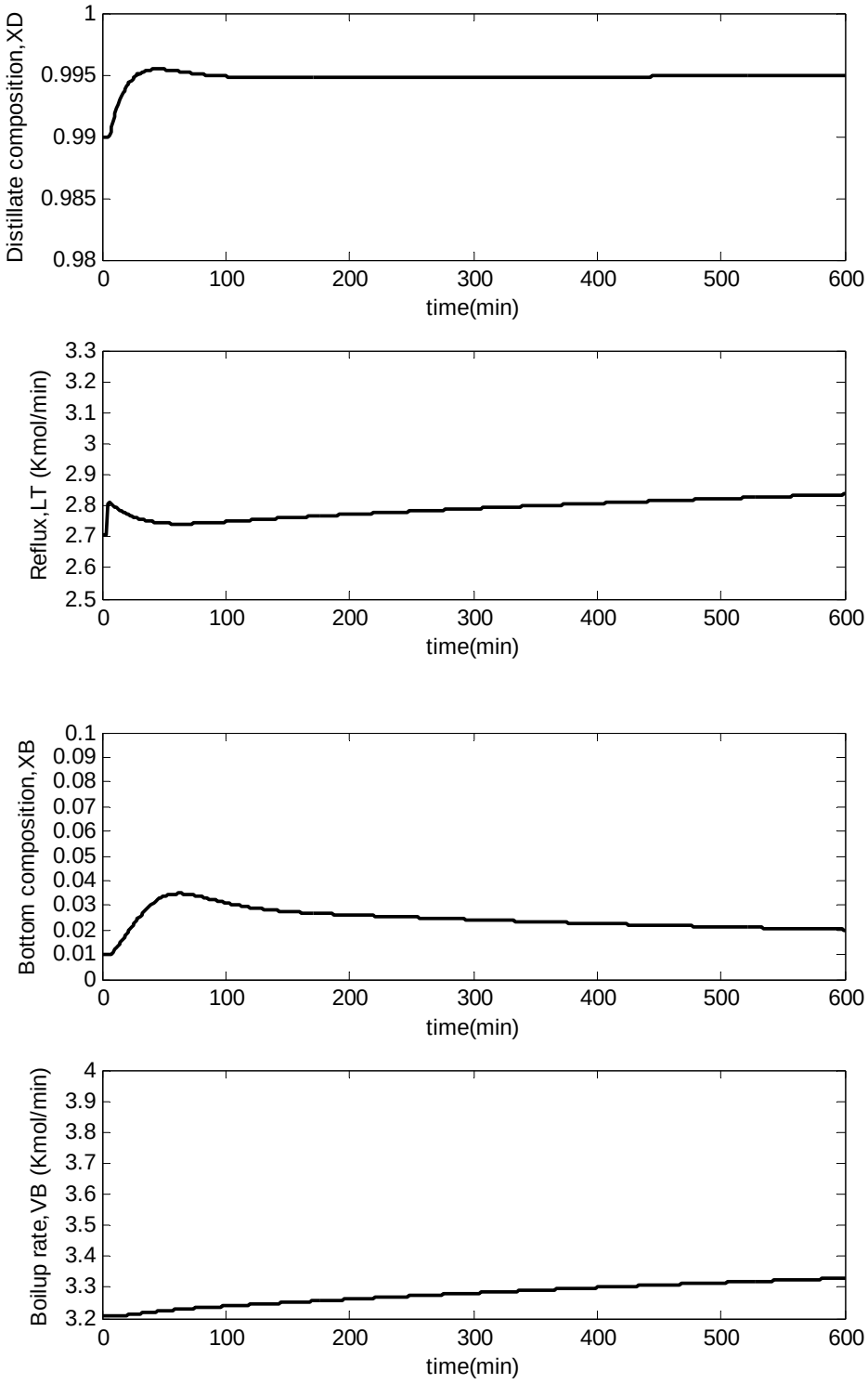

Figure 5.3 PI controller (IAE setting) for setpoint change (step setpoint change in XD at t=5min from 0.99 to 0.995)

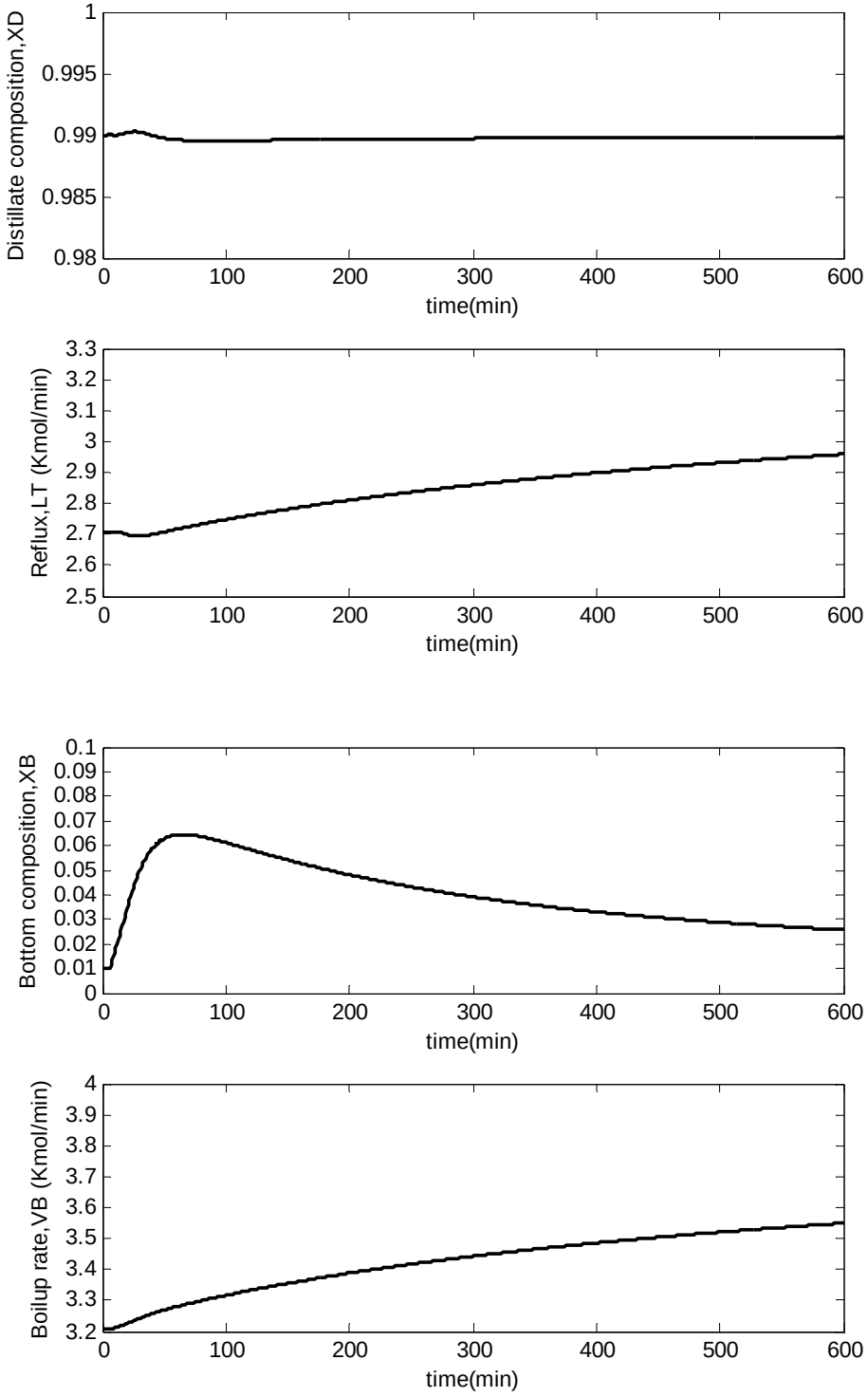

Figure 5.4 PI controller (IAE) for disturbance rejection (feed rate change at t=5min from 1.00 to 1.20)

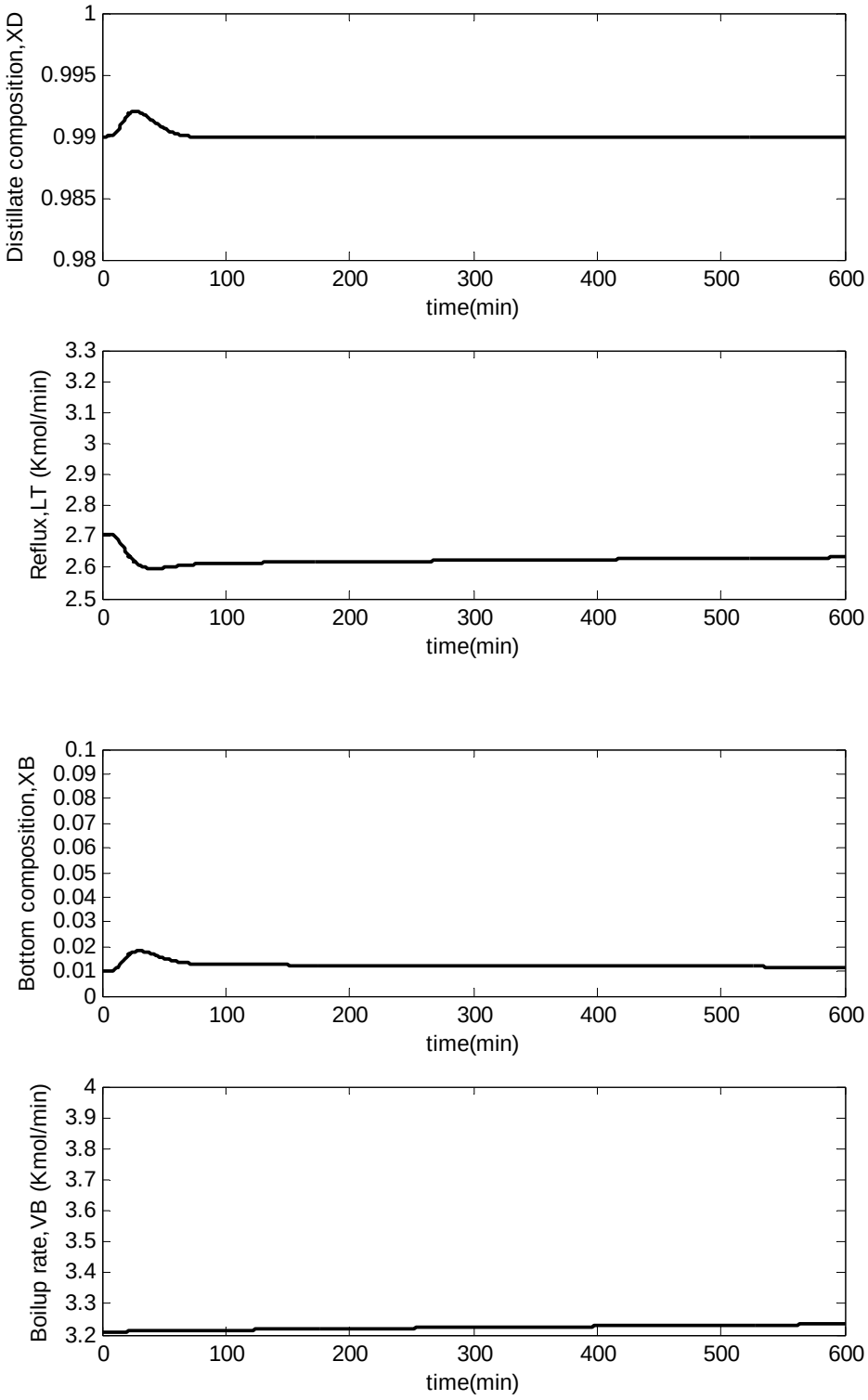

Figure 5.5 PI controller (IAE) for disturbance rejection (feed composition change at t=5min from 0.50 to 0.60)

For the regulatory control, it can be seen clearly from Figures 5.4 and 5.5 that the step change in feed composition, ΔZ_F does not produce a significant deviation in X_D and X_B compare to step change in feed rate, ΔF . This happens because the interactions for the LV configuration (two point control) has strongly amplified the effect of feed rate *F*, and reduce the effect of the feed composition, Z_F. Also, it can be observed that the disturbance in Z_F is more easily rejected compared to the disturbance in *F* that has more effect on X_D and X_B. An increased feed rate goes down to the bottom of the column, and this result, through the action of the bottom level controller, in a corresponding increase in the bottom flow. This would significantly affect the material balance which in turn affected the product compositions. The effect of *F* is exaggerated particularly when the controller can not make fast control in both composition loops as can be seen in the Figure 5.4 where the PI controller form IAE setting was unable to provide fast control action and this has resulted that the bottom composition still quite far away from its setpoint even at the end of simulation time (*t*=600min).

In overall, as can be seen from Figures 5.3 to 5.5, the X_B composition loop obviously took a longer settling time compare to X_D composition loop in this PI control scheme. The sluggish control movement shown by PI control also resulted a long settling time for both the X_D and X_B and this has caused them unable to settle down in setpoint change and feed rate change problem. Thus, it can be concluded that the PI control is able to perform better in a less interactive and nonlinear problem. As can be seen in Figures 5.4 and 5.5, PI control is evident able to deliver a better control results in counteracting the feed composition disturbance than in feed rate change.

5.2.2 DMC Control Results

For DMC**,** a selection and determination of the values of tuning parameter such as the prediction horizon (*P*), control horizon (*m*), move suppression weighting (Λ), weighting for controlled variables (Γ) is very important and has to be done before starting the closed loop control simulation. Although many detailed studies of

DMC tuning parameter has been done (Garcia and Morshedi, 1986; Shridhar, 1998), a well-established DMC tuning parameter selection theory has not yet been found and the choice of these parameters are generally strongly dependent on the sample time and the nature of the process.

In this work, the selection procedure is performed by testing them through a servo problem as mentioned in earlier section. As can be seen from Figure 5.6, as the prediction horizon (*P*) increases, the control actions became more vigorous and thus the corresponding output respond became more oscillatory and can eventually become unstable. In addition, it is also found that the value of prediction horizon (*P*) must be less than 3. This can be seen through the Figures 5.7 and 5.8, if the P is bigger than 3, the closed loop control actions either highly oscillatory, or very sluggish. Therefore, the *P* is not the final tuning parameter used in this work and they are set at [3 3] (the first 3 is for X_D-L_T control system whereas the second 3 is for X_B-V_B). For a fixed prediction horizon (*P*), a bigger control horizon would yield more aggressive output response (Henson, 1998). In this case, the maximum values available for *P* is 3 which is not a huge number (computational problem doesn't exist) and thus the values of control horizon, *m* is set as big as possible in this research and it is finally set to [3 3] and [2 2] for setpoint tracking and disturbance rejection problems respectively.

Next, Λ is varied in four different runs with Λ = [1 1], [3 3], [5 5] and [10 10] respectively in each run to see its impact on the closed loop performance (1 in Λ actually equal to 1*I* in which *I* is the *m* x *m* identity matrix). As can be seen from Figure 5.9, the value of Λ has to be higher than a certain limit (> 3 in this work) to give enough suppression onto the amplitude of the input moves in order to prevent it from behave unstable and oscillatory as what can be seen when Λ =1. Therefore, in conclusion, the candidate that best suits as the final DMC tuning parameter in this work is the weighting for controlled variables (Γ).

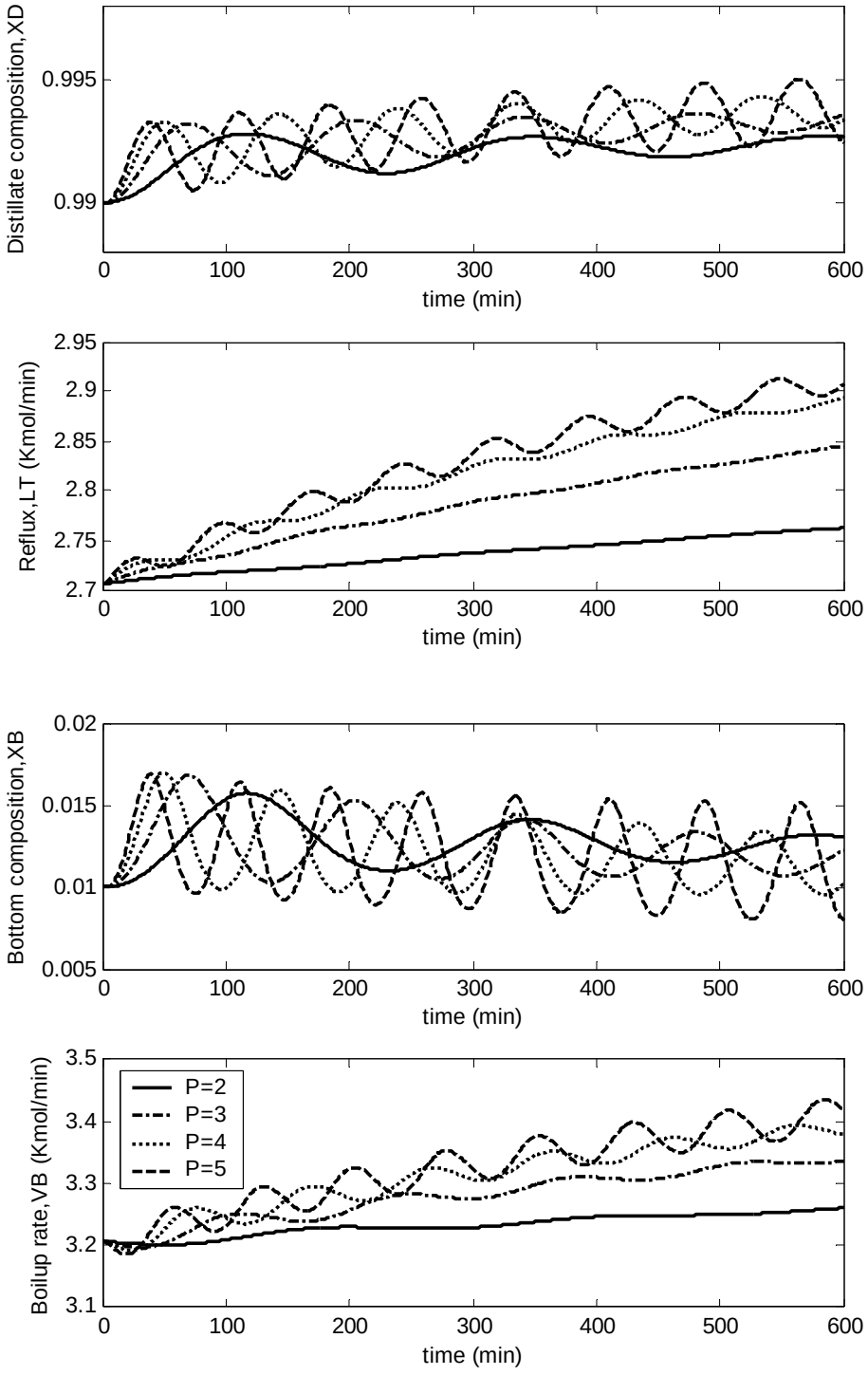

Figure 5.6 Effect of the prediction horizon on the closed loop performance

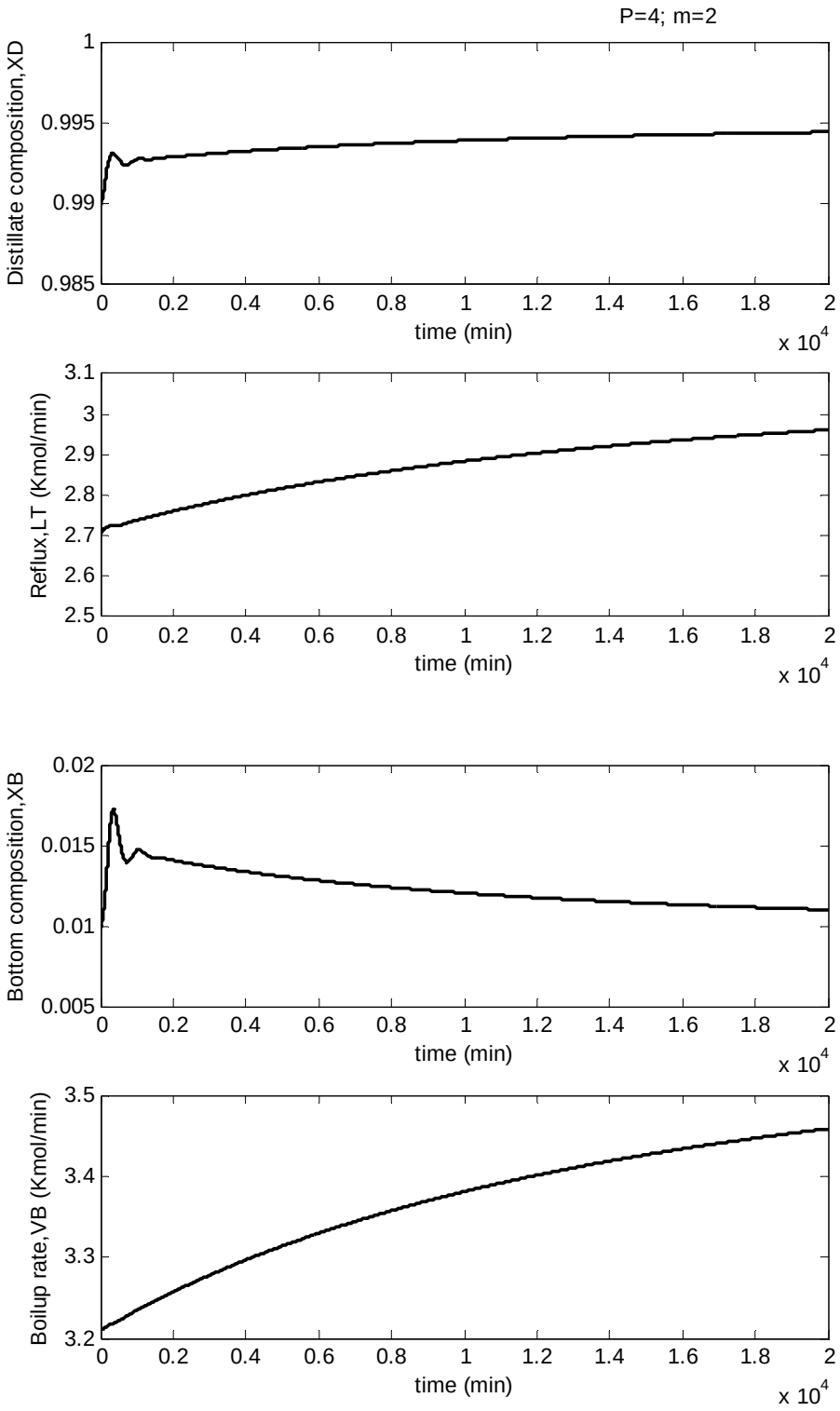

Figure 5.7 Closed loop response for P= [4 4]; m= [2 2]; Λ = [5 5] Γ = [5 5]

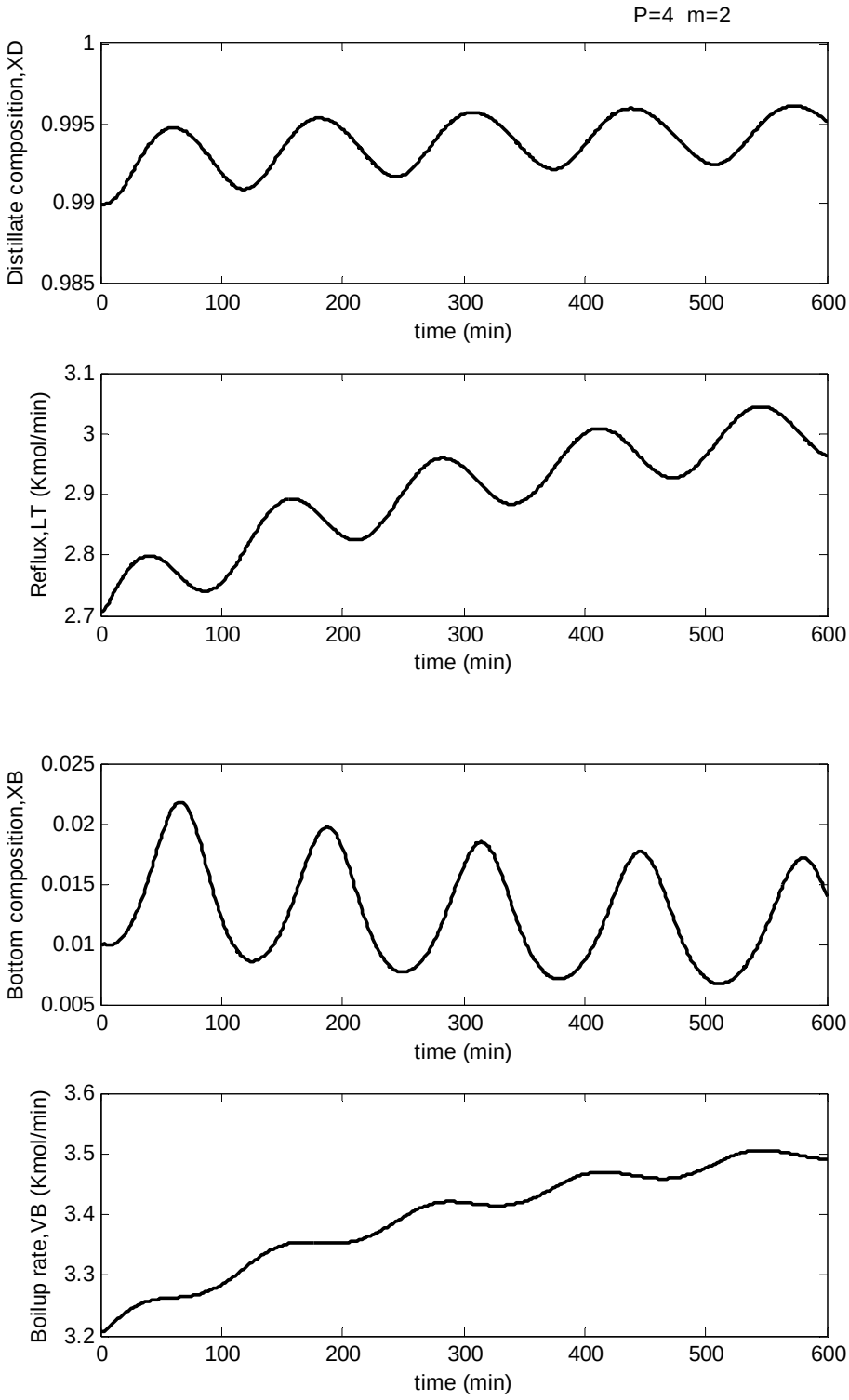

Figure 5.8 Closed loop response for P= [4 4]; m= [2 2]; Λ = [5 5] Γ = [20 20]

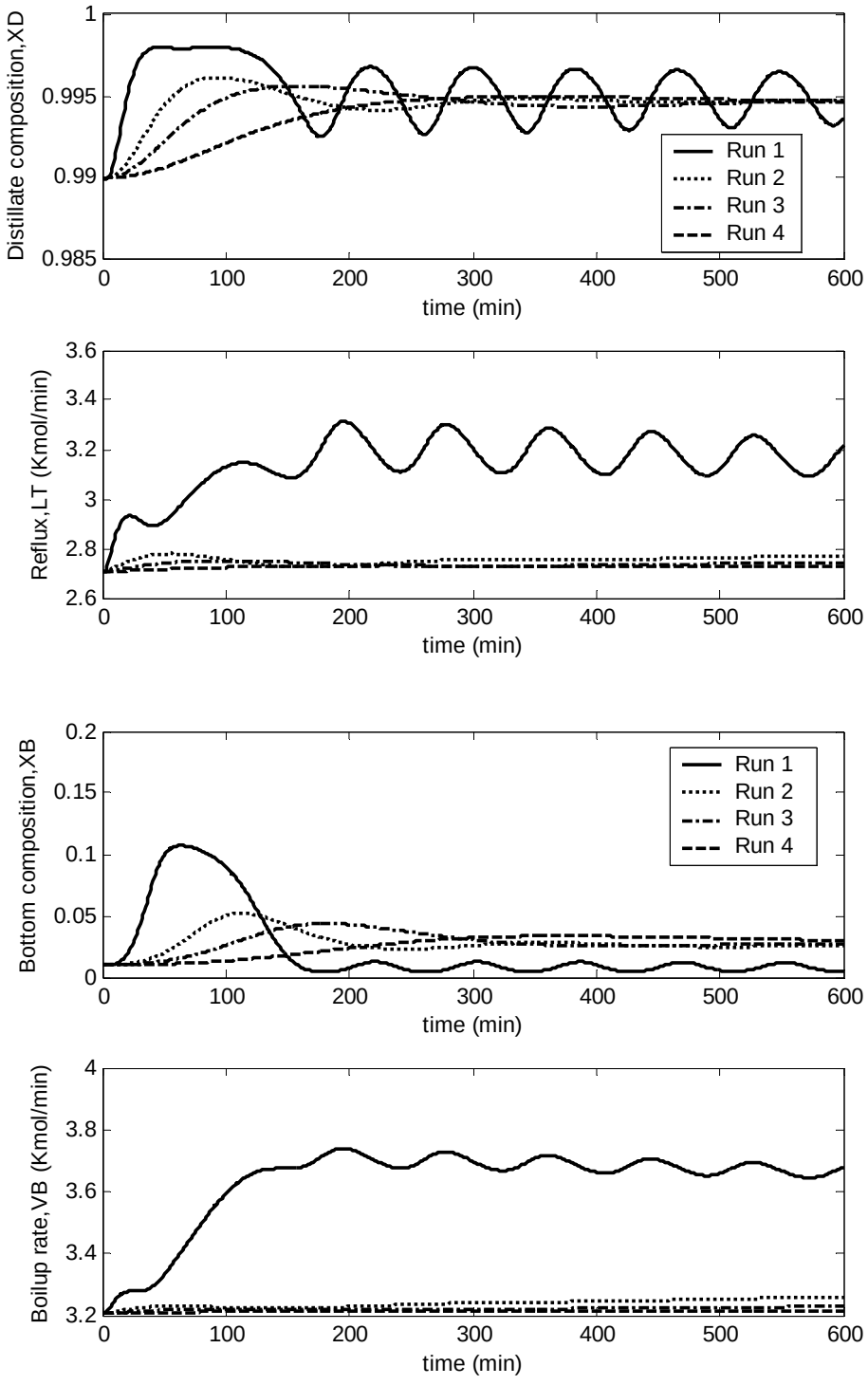

Figure 5.9 The impact of Λ on closed loop performance (P=3; m=2; Γ =20)

Table 5.1: Tuning parameters for DMC

Control Type	Setpoint Tracking	Feed Rate Change	Feed composition Change
Prediction Horizon, P	[3 3]	[3 3]	[3 3]
Control Horizon, m	[3 3]	[2 2]	[2 2]
Move Suppression Weighting (Λ)	[5 5]	[6 6]	[0.5 0.5]
Weighting for Controlled Variables (Γ)	[90 100]	[130 130]	[50 50]
Sampling time (min)	1	1	1

*****5 in Λ is actually equal to 5I in which I is the P x P identity matrix

With these final tuning parameters, the best servo and regulatory control performed by DMC can be seen from Figures 5.10, 5.11 and 5.12. As can be observed from Figure 5.15 for setpoint tracking, it is evident that DMC is able to cope with the interaction problem that occurred between the two composition loops by achieving their setpoint in a short time. As can be seen for Figure 5.10, the L_T and V_B were pushed to about 2.9 kmol/hr and 3.3kmol/hr after about 20min that the setpoint change is introduced. As a consequence, a large overshoot occurred in DMC especially in X_D loop (791.81%). DMC then took about 450min for X_D and 330min for X_B to reach a steady-state. On the other hand, for the regulatory mode, the control results shown by Figures 5.11 have indicated that the DMC is able to deliver a superior performance in unmeasured disturbance rejection. For the feed rate change, which is a more interactive and nonlinear process, the DMC took a very quick respond once the feed rate change is introduced and very smooth control actions are taken as well to counteract the effect of ΔF on top and bottom composition loops.

As a conclusion, the DMC shows a larger overshoot especially in X_D loop (791.81% for setpoint tracking, 717.58% for feedrate change and 56.736% for feed composition change) but a smaller settling time (settled down to steady-state in all experiments before the end of the simulation time) compare to PI control and this is a typical DMC characteristic.

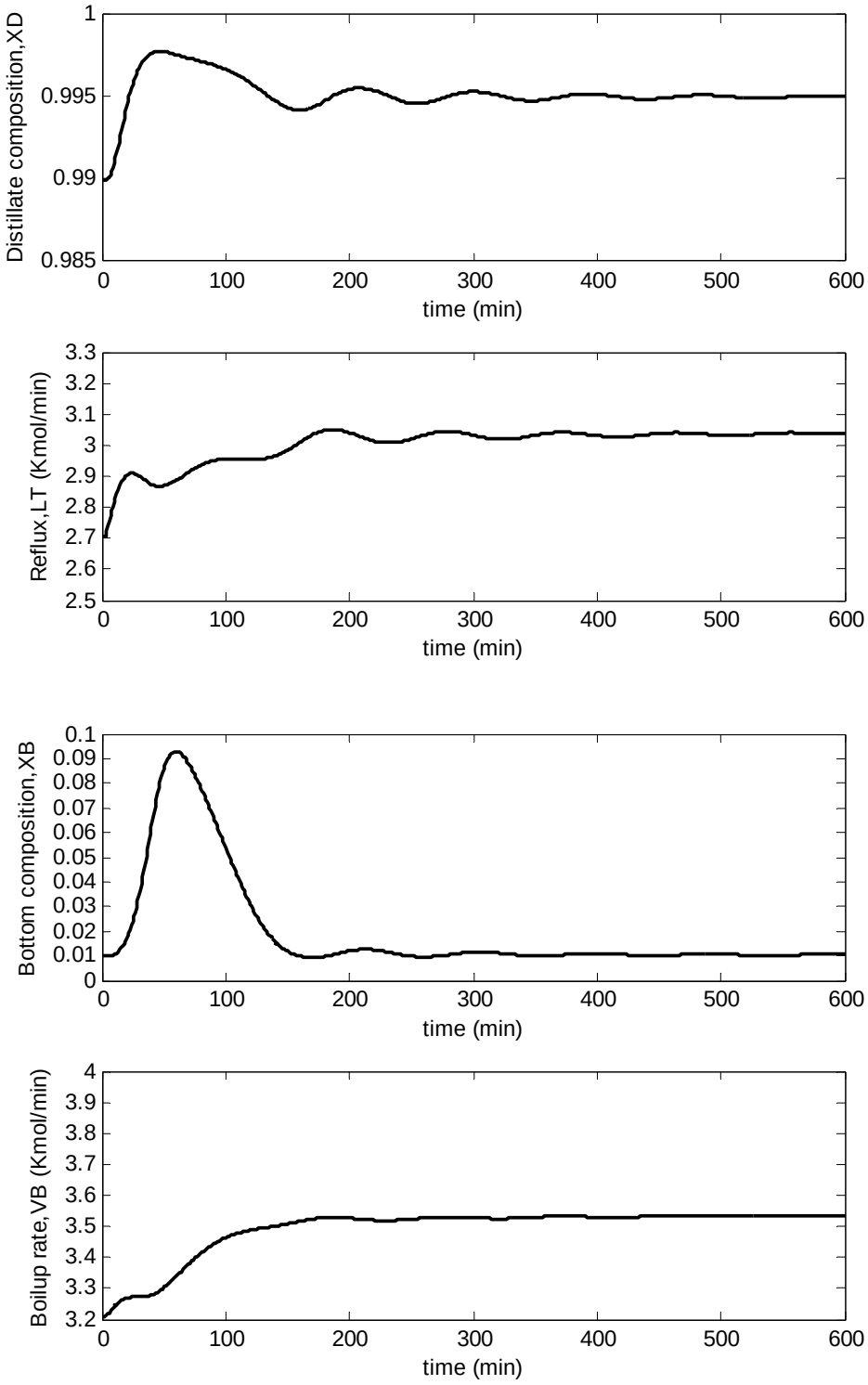

Figure 5.10 Illustration of the best servo control for DMC (step setpoint change in X_D at t=5min from 0.99 to 0.995)

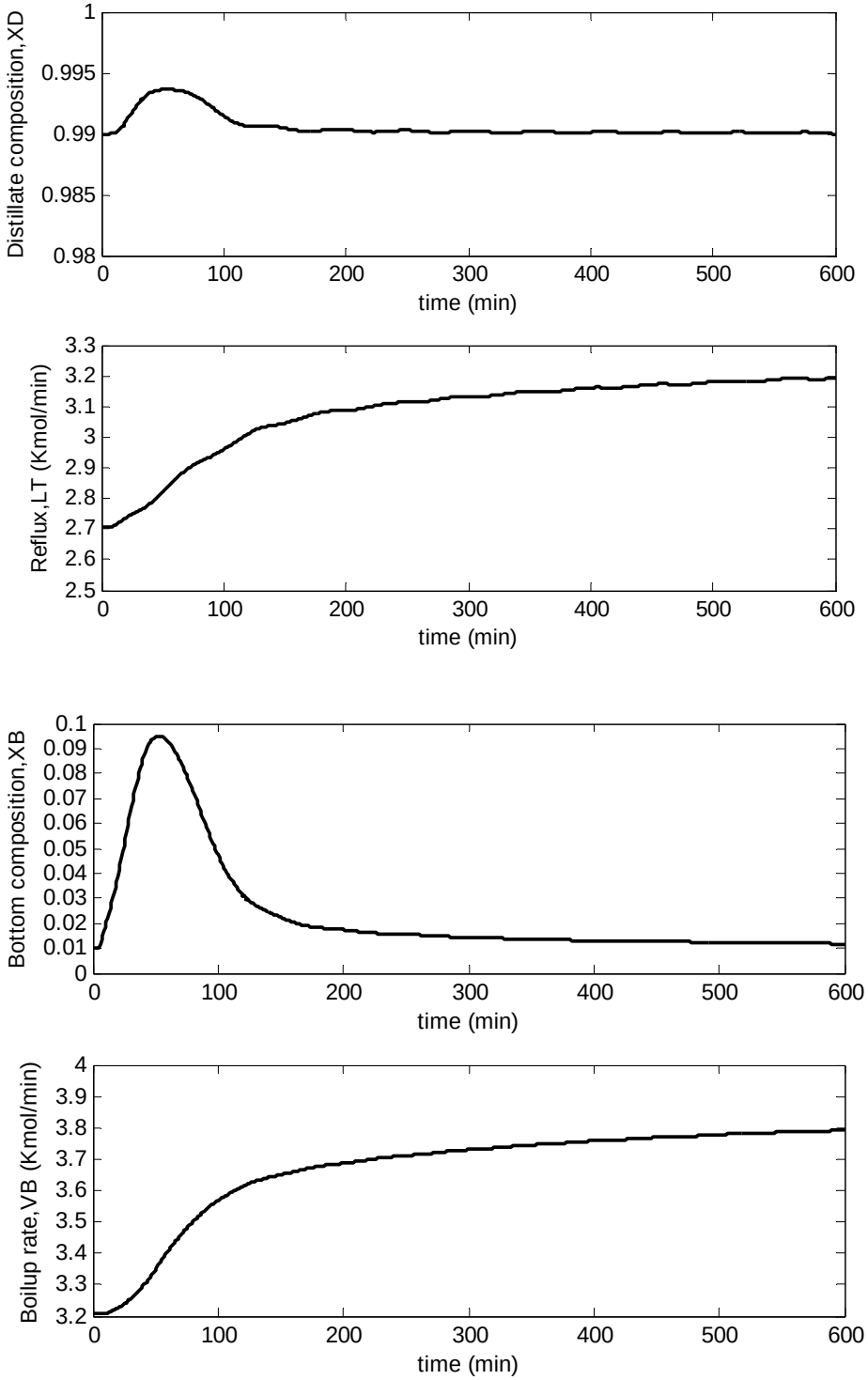

Figure 5.11 Illustration of the best regulatory control for DMC (feed rate change at t=5min from 1.00 to 1.20)

5.2.3 NNMPC Control Results

For the NNMPC, there are four available tuning parameters, which are the upper and lower limit of prediction horizon, n_1 and n_2, the control horizon, n_u and the manipulated variables weighting, ρ_j. Thus, to ensure the NNs are able to provide a promising model prediction, the maximum values of prediction horizon, n_1 is fixed at 2 and since 1 sample interval or 1 min time delay is included as a measurement delay for top and bottom composition, the lower limit of prediction horizon, n_2 is fixed at 1.

The initialization of n_1 and n_2 has indirectly fixed the value of control horizon n_u to 1. In most of the published NNMPC researches found, the n_u investigated has not greater than one except in Pottmann and Seborg (1997) research, in which the control horizon investigated was greater than 1, however it has caused a very aggressive control. In addition, since a maximum 2-step ahead prediction is required, the 2-node NNs is employed in both X_D - L_T and X_B - V_B control systems to control the distillation column although the 4-nodes NNs seen able to provide a more consistent performance in doing the multi-step prediction. In other words, manipulated variables weighting, ρ_j is the remaining and only choice for the final tuning parameter in NNMPC. After a series of simulation experiments performed to establish the appropriate values of ρ_j, the tuning parameters final results are shown as in Table 5.2.

Table 5.2: Tuning parameters for NNMPC

Control Type	Setpoint Tracking	Feed Rate Change	Feed composition Change
Upper limit of Prediction Horizon, n_1	[2 2]	[2 2]	[2 2]
Upper limit of Prediction Horizon, n_2	[2 2]	[2 2]	[2 2]
Control horizon, n_u	[1 1]	[1 1]	[1 1]
manipulated variables weighting, ρ_j	[0.50 0.30]	[1 0.5]	[2 0.03]
Sampling time (min)	1	1	1

For the NNMPC, the closed loop performance for setpoint tracking is illustrated by Figure 5.17 while Figures 5.13 and 5.14 shows the control results in coping with the feed rate and feed composition change disturbances. For the servo mode, as can be seen from Figure 5.13, NNMPC is obviously able to show a superior control performance in handling the setpoints change. NNMPC displayed a rapid respond with smooth control actions in which the L_T and V_B were increased gradually with a small magnitude of overshoot and shorter settling time. For the regulatory problem, the control results shown by Figures 5.14 and 5.15 also illustrated that the NNMPC is able to perform well in handling the feedrate and feed composition change. The superiority of NNMPC is evident particularly in handling the feed rate change, which is a more interactive and nonlinear process. Figure 5.13 shows that a rapid responds with a gradual increasing control actions were taken by NNMPC especially in X_B loop and this is has drive both the composition to settle down to their setpoints in a shorter settling time and with a smaller overshoot..

As a conclusion, NNMPC is obviously able to display a better control performance compared to the previous two control strategies. It exhibited more consistent control actions with a smaller magnitude of overshoot and settling time as can seen from Figures 5.12 to 5.14. However, the mismatch between the process/model in NNMPC does result a small offset in its control variables. The offset problem is significant particularly for the feed rate change problem, as can be seen from Figure 5.13 where approximately 0.008-0.009 deviation in X_B composition setpoint happens. A small offset also can be observed in X_D and X_B for both the setpoint tracking and feed composition change disturbance problem as can be seen in Figures 5.12 and 5.14. They are just not as significant as in the feed rate change problem.

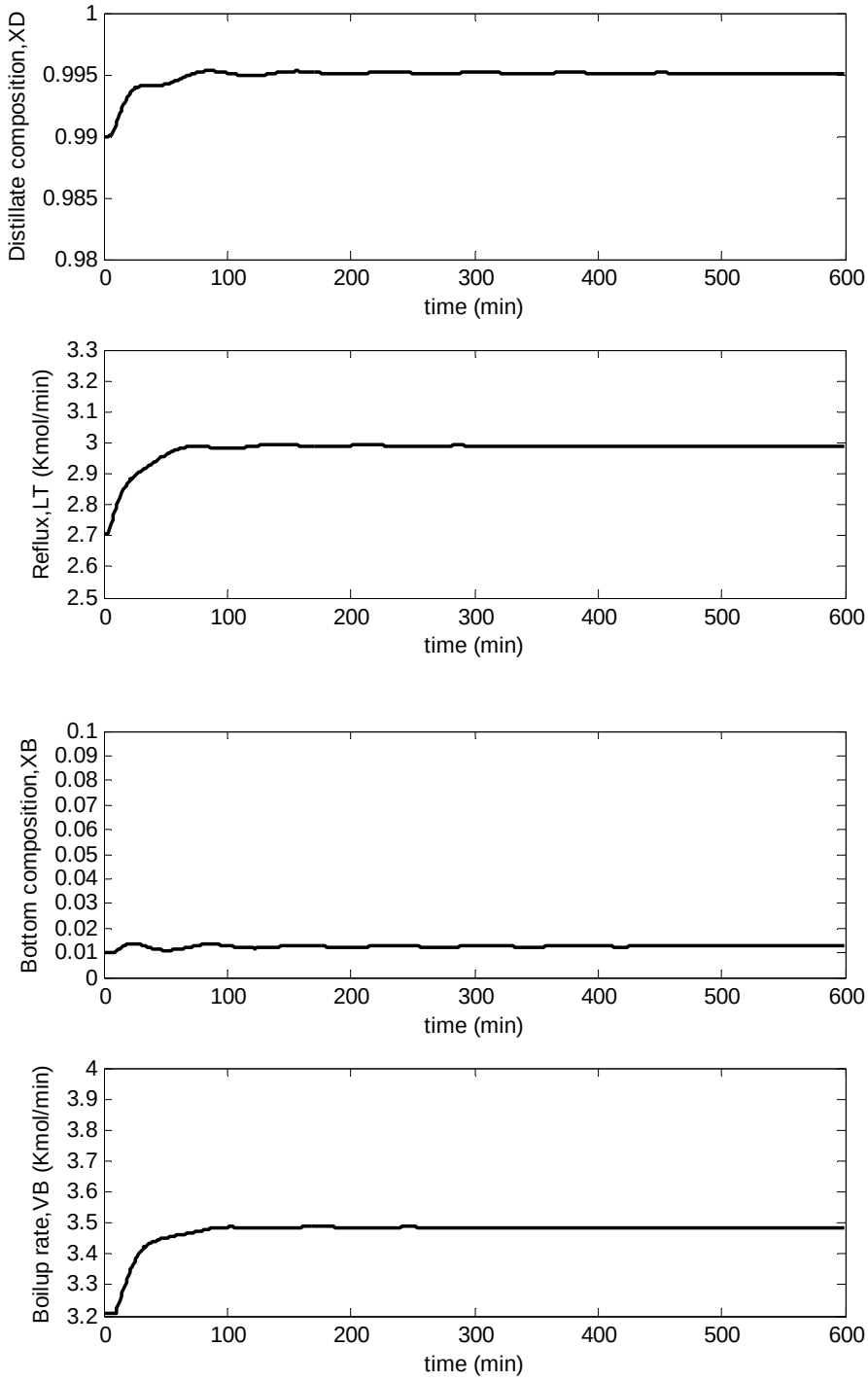

Figure 5.12 Illustration of the best servo control for NNMPC (step change in X_D at t= 5min from 0.99 to 0.995)

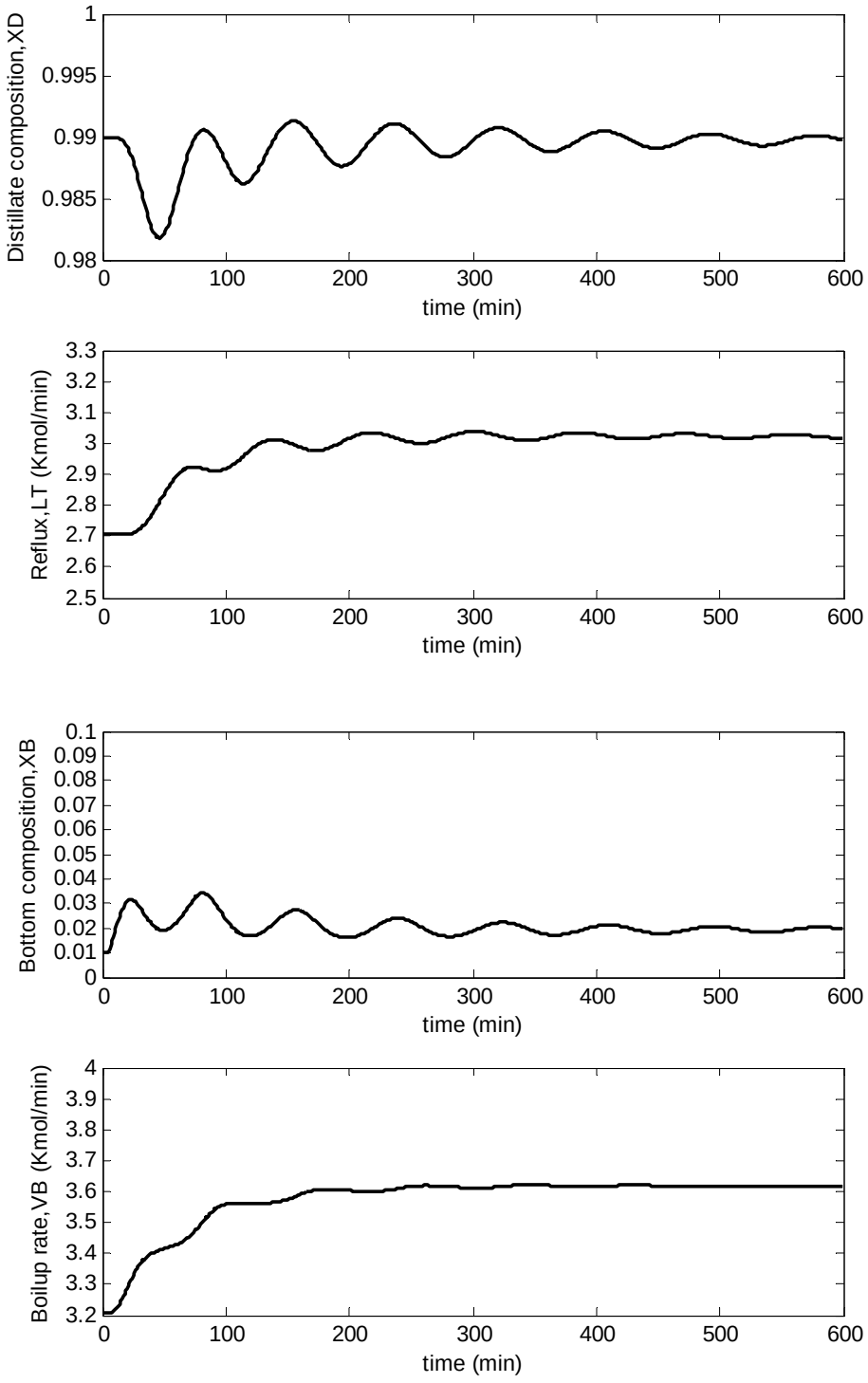

Figure 5.13 Illustration of the best regulatory control for NNMPC (feed rate change at t=5min from 1.00 to 1.20)

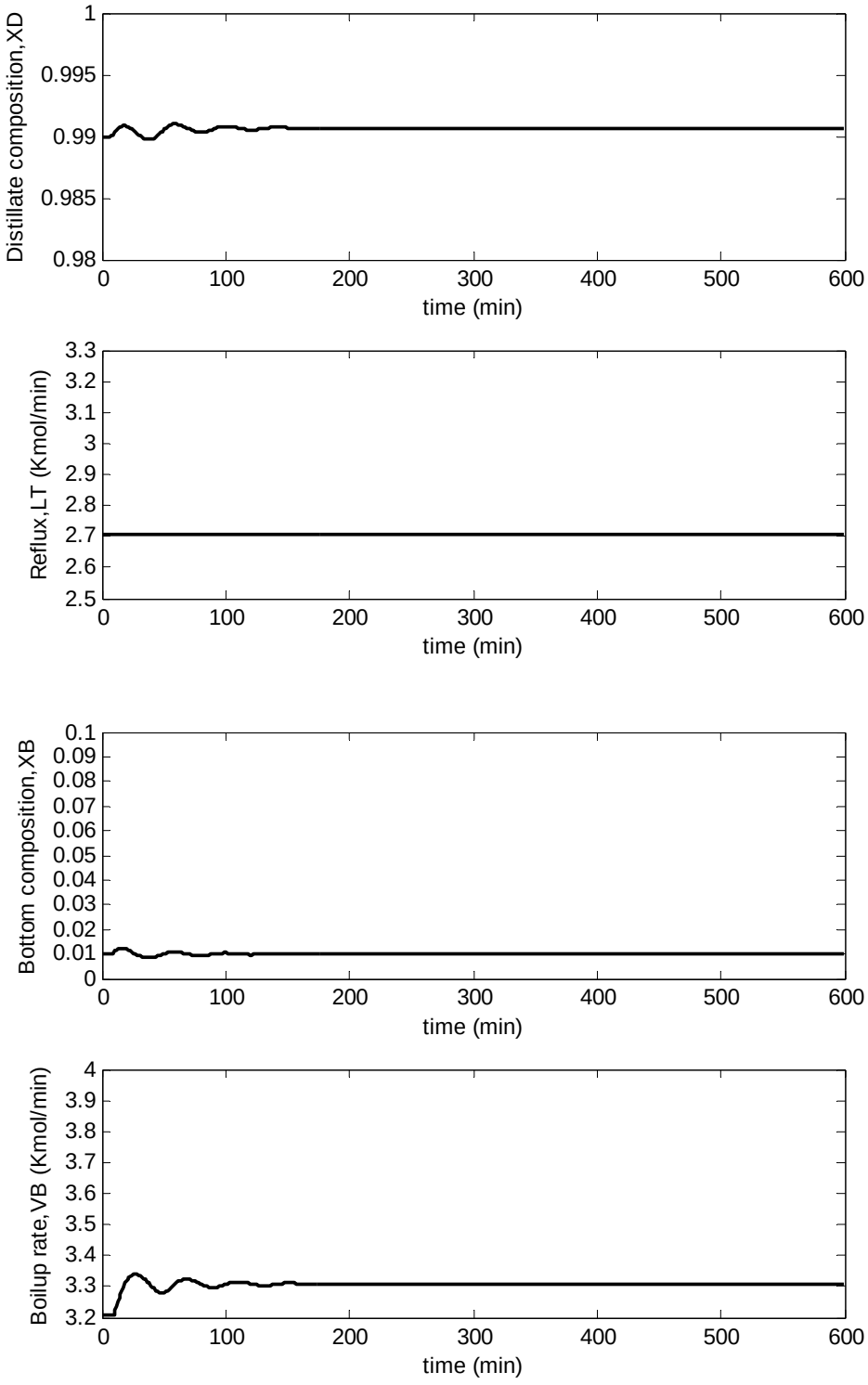

Figure 5.14 Illustration of the best regulatory control for NNMPC (feed) composition change at t=5min from 0.50 to 0.60)

5.3 Overall Comparison between PI, DMC and NNMPC

For the setpoint tracking, as can be seen from Figure 5.15, NNMPC is obviously outperformed over the PI control and DMC. NNMPC responds immediately once the setpoint change is introduced, this has enable both the composition settled down towards their setpoints with the comparatively shortest settling time and smallest overshoot among these three control strategies.

For the regulatory problems, as can be observed from Figures 5.16 and 5.17, the superiority of NNMPC over others two control strategies is still valid. Almost the same conclusion can be made especially in the feedrate change problem. NNMPC still able to deliver an outstanding performance by showing a shorter settling time and smaller overshoot control results. DMC also shows its natural characteristic by display a large overshoot while PI control keeps its sluggish control movements in coping with feedrate change disturbance. However, in the Figure 5.17 where a feed composition change disturbance is introduced, the difference in the control performance among PI control, DMC and NNMPC became smaller. This happen because the interaction effect for feed composition change is not as severe as in feedrate change where the increase in feedrate would in turn affects the material balance and consequently brings a significant change in product compositions.

As a conclusion, the overall comparison between the closed loop responses of the PI control, DMC and NNMPC has came out the fact in which the NNMPC is more superior over the DMC and PI control. However, the mismatch between the process/model in NNMPC caused by the implementation of two separately-trained neural networks in this research has resulted a small steady-state offset even the conventional DMC feedback strategy of adding the most current one step prediction error to all the future predictions was applied.

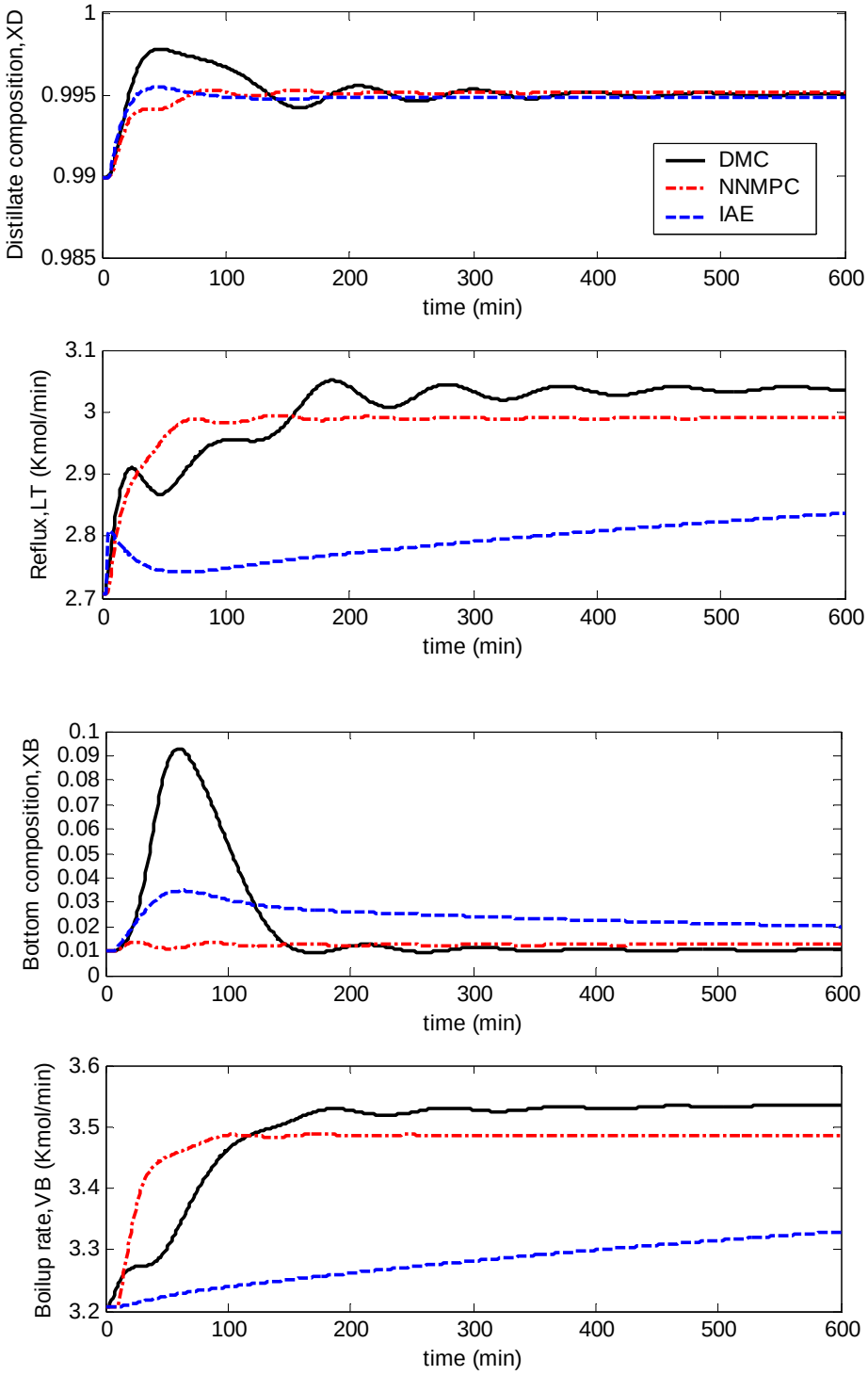

Figure 5.15 Comparison between the closed loop responses for the PI, DMC and NNMPC controllers (setpoint tracking problem)

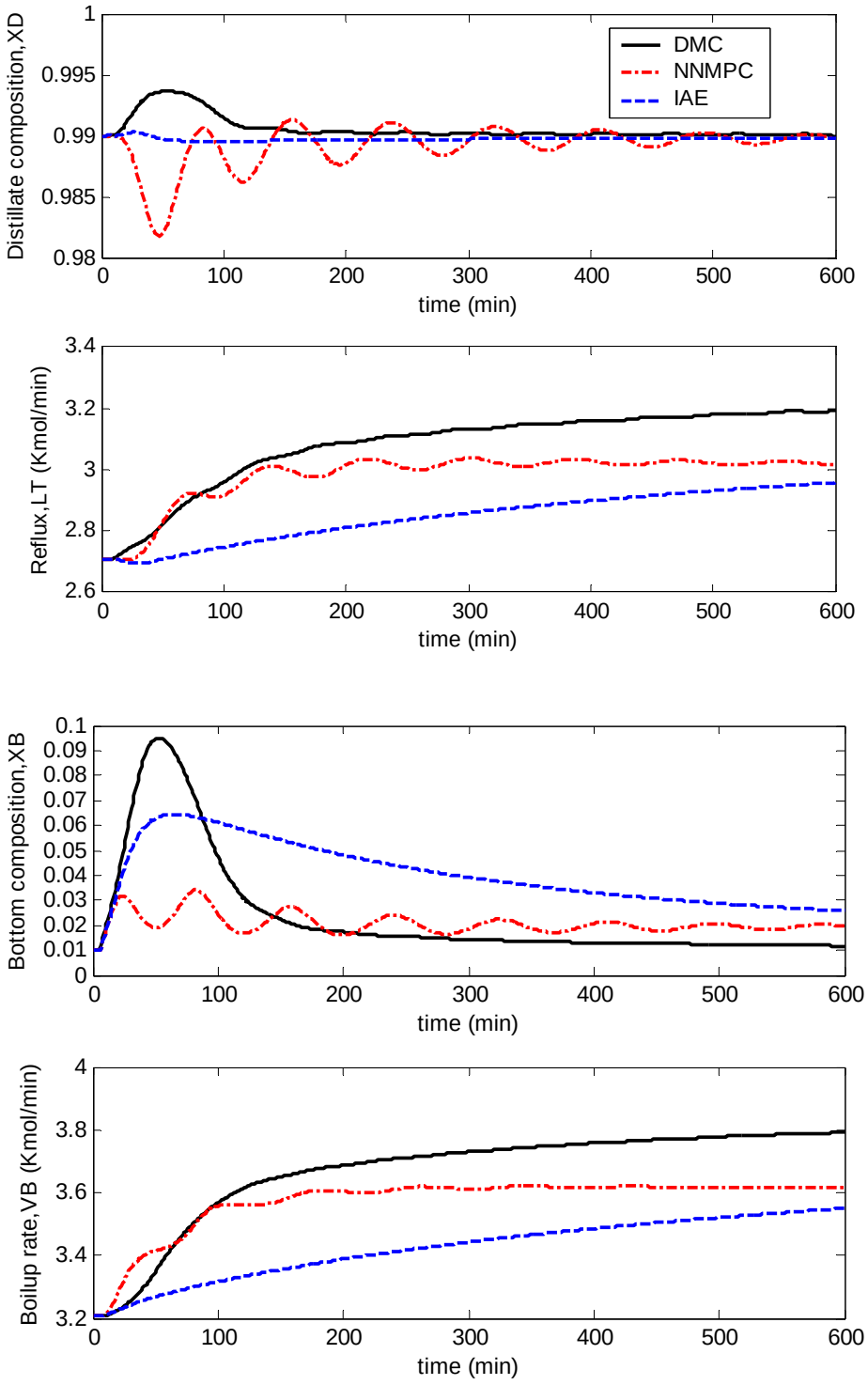

Figure 5.16 Comparison between the closed loop responses for the PI, DMC and NNMPC controllers (feed rate change disturbance rejection)

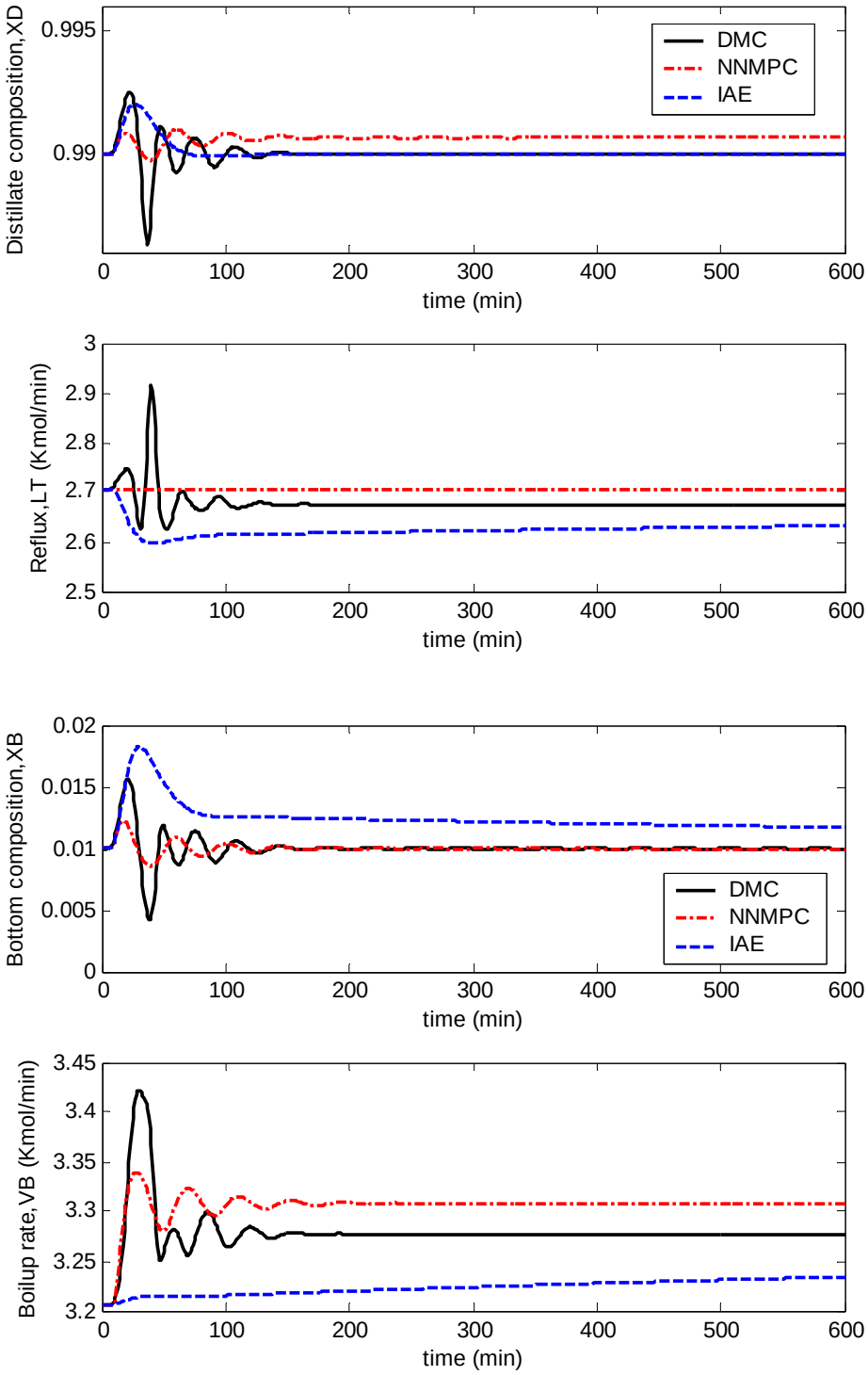

Figure 5.17 Comparison between the closed loop responses for the PI, DMC and NNMPC controllers (feed composition change disturbance rejection)

CHAPTER 6

CONCLUSION

The use of DMC and NNMPC to control a solution copolymerization reactor a high purity distillation column respectively, was investigated. An unconstrained MIMO DMC and NNMPC algorithms were developed using a step response model and two Feedforward Neural Networks respectively. Additionally, the comparison between DMC, NNMPC and PI controller based on IAE tuning rules was conducted. Thus, in overall, the primary research aims that were listed in Section 1.3 were achieved.

The control results illustrated in Chapter 5 by Figures 5.18 to 5.20 had shown that the DMC control scheme is well formulated and effectively implemented to control the high purity distillation column although only a simple 5% step input test was conducted to derive the step response coefficients. The outputs seem to converge well to the desired set point in both the composition loops for regulatory and servo problems.

For the NNMPC control scheme, the use of two separately-trained feedforward NNs and the Levenberg-Marquart optimization approach had demonstrated the training results that were considerably "applicable" up to 2^{nd} prediction step. Additionally, the use of the ideas originated from Sorensen *et* al. (1999) in performing the multi-step prediction and calculating the gradient of the cost function based on NNs was seen able to deliver a satisfactory control performance. In overall, NNMPC control scheme had shown a superior performance over the DMC and PI controllers by presenting a smaller overshoot, shorter settling

and smaller values of SSE and SCE. However, the use of two separately-trained neural networks in this research had caused a small offset in the controlled variable responds which can be considered as a small flaw in this control scheme.

Comparison of the NNMPC with the DMC and PI controller, both are the industrially popular and successful control strategies in this research had clarify the significant features of NNMPC.

REFERENCES

Abou-Jeyab, R.A., Gupta, Y. P., Gervais, J. R., Branchi, P. A. and Woo, S. S. (2001). Constrained Multivariable control of distillation column using a simplified model predictive control algorithm. *Journal of Process Control*. 11:509-517.

Albuquerque, J., Gopal, V. Staus, G. Biegler, L. T. Ydstie, E.B. (1999). Interior point SQP strategy for large-scale, structured process optimization problems. *Computer and Chemical Engineering.* 23(4): 543-554.

Aufderheide, B. and Bequette, B. W. (2003). Extension of matrix control to multiple models. *Computers and Chemical Engineering.* 27: 1079-1096.

Asohi, M. (1995). *Modeling and Control of a Continuous Crystallization Process Using Neural network and Model Predictive Control.* University of Saskatchewan: Thesis of PhD.

Baughman, D. R. and Liu, Y. A. (1995). *Neural networks in Bioprocessing and Chemical Engineering.* San Diego: Academic Press. 48-51.

Brizuela, E., Uria, M. and Lamanna, R. (1996). Predictive Control of a Multi-Component Distillation Column based on Neural Networks. *Proceedings of International Workshop on Neural Networks for Identification Control, Robotics and Signal/Image Processing.* August 1996. 270-278.

Camacho, E. F. and Bordons, C. (1998). *Model Predictive Control*. London: Springer. 67-80.

Campo, P. J. and Morari, M. (1987). Robust model predictive control. *Proceedings of the American control conference*. 1021-1026.

Cannon, M., Kouvaritakis, B., and Anthony Rossiter, J. (2001). Efficient active set optimization in triple mode MPC. *IEEE Transaction on Automatic Control*. 46(8): 1307-1312.

Chen, H. and Allgower, F. (1997). A quasi infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica.* 34(10):1205-1218.

Chen, I. L. and Ogunnaike, B. A. (1993). Modeling and control of High Purity Distillation Columns. *Technical report: E.I. du Pont de Nemours and Company.*

Chiu, K. C., Corripio, A. B. and Simith, C. L. (1973). Digital Control Algorithms. Part III. Tuning of PI and PID Controllers. *Instruments and Control Systems*. 46(12): 41-43.

Chou, C. T., Bloemen, H. H. J., Verdult, V., Van den Boom, T. T.J., Backs, T. and Verhaegen, M. (2000). *Nonlinear identification of High Purity Distillation Columns.* IFAC SYSID. California.

Clarke, D. W., Mohtadi, C. and Tuffs, P. S. (1987).Generalized predictive control-Part I. The basic algorithm. *Automatica (Journal of IFAC)*. 137 - 148.

Congladis, J. P., Richards, J. R. and Ray, W. H. (1989). Feedforward and Feedback Control of Solution Copolymerization Reactor. *AIChE Journal.* 35(6):891-907

Cutler, C. R. and Ramaker, B. L. (1979). *Dynamic matrix control: a computer control algorithm*. In Proceedings of AIChE 86th National Meeting. Texas.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems.* 2(4): 303-314.

Demuth, H. and Beale, M. (1998). *Neural Network Toolbox: User's Guide, Version 3.0.* The MathWorks, Inc., Natick, MA.

Doherty, S. K. (1999). *Control of pH in Chemical processes using Artificial Neural Network.* Liverpool John Moores University. Thesis of PhD.

Finlayson, B. A. (1980). *Nonlinear Analysis in Chemical Engineering*. New York:McGraw Hill.

Findeisen, R. and Allgower, F. (2002). An Introduction to Nonlinear Model Predictive Control. *21st Benelux Meeting on systems and Control*. 1-23.

Fletcher, R. (1987). *Practical Methods of Optimization.* 2nd edition. New York:Wiley & Sons.

Fruzzetti, K. P., Palazoglu, A. and MacDonald, K. A., (1997). Nonlinear model predictive control using Hammerstein models. *Journal of Process Control.* 7(1): 31-41.

Gaiceanu, M., Rosu, E. and Tataru, A. M. (2000). Neuro optimal controller for three phase induction motor based on Levenberg Marquardt training algorithm. *IEEE Journal*. 97-102.

Garcia, C. E., and Morari, M. (1982a). Internal Model Control 1: A unifying review and some new results. *Industrial Engineering Chemical Process Design and Development*. 21:308-323.

Garcia, C. E., and Morari, M. (1982b). Internal Model Control 2: A unifying review and some new results. *Industrial Engineering Chemical Process Design and Development*. 24:472.

Garcia, C. E., Prett, D. M. and Morari, M. (1989). Model Predictive Control: Theory and practice- a survey. *Automatica*. 25(3): 335-348.

Garcia, C. E. and Morshedi, A. M. (1986). Quadratic programming solution of Dynamic Matrix Control (QDMC). *Chemical Engineering Community*. 46:73-87.

Garcia, G. (1984). Quadratic dynamic matrix control of nonlinear processes: An application to a batch reactor process. *AIChE Annual Meetin*g. San Francisco.

Gattu, G. and Zaririou, E. (1992). Nonlinear quadratic dynamic matrix control with state estimation. *Industrial and Engineering Chemical Research*. 31:1091-1104.

Georgiou, A., Georgakis, C. and Lubyen, W. L. (1988). Nonlinear Dynamic Control for High Purity Distillation Column. *AlChE Journal*. 34(8): 1287-1299.

Gill, P. E., Murray, W. and Sauders, M. A. (1998). *User's Guide fro SNOPT 5.3/6.0: a Fortran Package fro large scale Nonlinear Programming*. System Optimization Laboratory. Stanford University.

Gokhale, V. B. (1994). *Control of a Propylene/propane splitter*. Texas Tech University: Thesis of Master in Science.

Gupta, Y. P. (1996). A simplified predictive control approach applied to FCCC units. *Proceeding of international Conference of Chemical Process Control*. 319-333.

Gomm, J. B., William, D., and Evans, J.T., Doherty, S. K. and Lisboa, P. J. G. (1996). Enhancing the nonlinear modeling capabilities of MLP neural networks using spread encoding. *Fuzzy Sets and Systems*. 79:113-126.

Henson, M. A. (1998). Nonlinear Model Predictive Control: Current status and Future direction. *Computer and Chemical Engineering*. 23(2): 187-202.

Hornick, K., Stinchcombe, M. and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*. 2:359-266.

Hunt, K. J., Sbarbaro, D., Zbikowski, R. and Gawthrop, P. J. (1992). Neural Network for control systems. *Automatica.* 28(6): 1083-1112.

Hovd, M. and Michaelsen, R. (1997). Model Predictive Control of a Crude Oil Distillation Column. *Computer and Chemical Engineering*. 21:S893-S897.

Hugo, A. (2000). Limitation of Model Predictive Controllers. *Hydrocarbon Proceeding*. 79: 83-88.

Jang, S. S. and Wang, L. S. (1997). Experimental study of rigorous nonlinear model predictive control for a packed distillation column. *Journal of Chinese Institute of Chemical Engineer*. 28(3): 151-162.

Kawathekar, R. (2004). *Nonlinear Model Predictive control of a Reactive Distillation Column*. Texas the University: Thesis of PhD.

Koivisto, H. (1995). A Practical Approach to Model Based Neural Network Control. Finland: Tampere University of Technology Publication.

Kyoung, S. J. (1995). *Control Relevant Identification methodology with Application to a High Purity Distillation Column*. Arizona State University: Thesis of PhD.

Larsen, J. and Hansen, L. K. (1994). Generalization Performance of Regularized Neural Network Models. *Proceedings of the IEEE workshop on Neural Networks for signal Processing IV*. New Jersey: Piscataway. 42-51.

Lee, E. B. and Markus, L. (1967). *Foundations of optimal Control Theory*. New York: John Wiley and Sons.

Lee, J. H. and Ricker, N. L. (1994). Extended Kalman filter based nonlinear model predictive control. *Industrial Engineering and Chemistry Research*. 33(6): 1530-1541.

Lee, J. H. (1998). Modeling and identification for nonlinear model predictive control: requirement, current status and future research needs. In: Allogower, F. and Zheng, A. *Nonlinear model predictive control* Birkhauser. 269-293.

Leontaritis, I. J. and Billings, B. A. (1985). Input-Output parameter models for nonlinear system. Part 1: Deterministic nonlinear systems. Part 2: Stochastic nonlinear systems. *International Journal of control*. 41: 303-344.

Ljung, L. (1999). *System Identification- Theory for the User*. 2nd edition. Eagle Cliffs, N. J.: Prentice Hall.

Lundstrom, P., Lee, J. H., Morari, M. and Skogestad, S. (1994). Limitation of Dynamic Matrix Control. *Computers and Chemical Engineering.* 19:409-421.

Maciejowski, J. M. (2002). *Predictive Control with Constraints*. England: Pearson Education Limited.

Maner, B. R., Doyle, F. J., Ogunnaike B. A. and Pearson R. K. (1996). Nonlinear model predictive control of a simulated multivariable polymerization reactor using second order Volterra models. *Automatica.* 32: 1285-1301.

McDonald, K. A, and McAvoy, T. J. (1987). Application of Dynamic Matrix Control to Moderate and High Purity Distillation Towers. *Industrial Engineering and Chemistry Research*. 26:1011.

McDonald, K. A., Palazoglu, A. and Bequette, B. W. (1988). Impact of Model Uncertainty Descriptions for High Purity Distillation Control. *AIChE Journal*. 34(12):14-17.

Meadows, E. S., Henson, M. A., Eaton, J. W. and Rawlings, J. B. (1995). Receding horizon control and discontinuous state feedback stabilization. *International Journal of Control*. 62(5):1217-1299.

Morari, M. and Lee, J. H. (1999). Model Predictive Control: Past, Present and Future. *Computer and Chemical Engineering.* 23:667-682

Narendra, K. S. and Parthasarathy, K. (1990). Identification and control of Dynamic Systems using Neural Networks. *IEEE Neural networks*. 1:4-27.

Norquay, S. J., Palazoglu, A., and Romagnoli, J.A. (1999). Application of Wiener Model Predictive Control (WMPC) to industrial C2-Slitter. *Journal of Process Control.* 9:461-473.

Norgaad, M. (1997). *Nueral Network based sysetm identification Toolbox.* Technical Report. 97-E-851. Department of Automation, technical university of Demark.

Onnen, C., Babuska, R., Kaymak, U., and Sousa, J. M. and Verbruggen, H. B. and Isermann, R. (1997). Genetic algorithm for optimization in predictive control. *Control Engineering Practice.* 5(10): 1363-1372.

Ohshima, M., Ohno, H. and Hashimoto, I. (1995). Model Predictive Control-experience in the university-industry joint projects and statistic on MPC application in Japan. *International al workshop on predictive and receding horizon control*. Seoul: university of Seoul, 1-16.

Ogunnaike B. A. and Ray W. H. (1994). *Process Dynamic, Modeling and Control.* New York: Oxford University Press.

Ou, J. (2001). *Grouped Neural Network Model Predictive Control and its Experimental Distillation Column*. Oklahoma State University. Thesis PhD

Parlos, A. S., Rais, O.T. and Atiya, A.F. (2000). Multi-step Prediction using Dynamic Recurrent neural Networks. *Neural networks* 13:765-786.

Patwardhan, A. A., and Edgar, T. F. (1990). Nonlinear model predictive control of a packed distillation column. *Industrial Engineering and Chemistry Research.* 32(10):2345-2356.

Padtwardhan, A. A., Wright, T. T. and Edgar, T. E. (1992). Nonlinear model predictive Control of distributed parameter systems. *Chemical Engineering Science.* 47(4): 721-735

Piche, S., Sayyar-Rodsari, B., Johnson, D. and Gerules. M. (2000). Nonlinear Model Predictive Control using Neural Network. *IEEE Control System Magazine.* 20(3): 53-62.

Polak, E. and Yang, T. H. (1993). Moving horizon of linear system with input saturation and plant uncertainty. *International Journal of Control.* 58(3): 613-663.

Pottmann, M. and Seborg, D. E. (1997). A nonlinear predictive control strategy based on radial basis function model. *Computer and Chemical Engineering.* 21: 965-980.

Proll, T. (1993). *Model Predictive Control based on Nonlinear Autoregressive and Neural Network Models*. Colorado State University: Thesis PhD.

Propoi, A.I. (1963). Use of linear programming methods for synthesizing sampled data automatic systems. *Automation and Remote Control.* 24: 837-844.

Qin, S. J. and Bagwell, T. A. (1997). An overview of industrial predictive control technology. In: Kantor, J. C., Garcia, C. E. and Carnahan, B. *Proceedings of 5th International Conference on chemical process control.* AICHE and CACHE. 155-171.

Qin, S. J. and Bagwell, T. A. (2000). An overview of nonlinear model predictive control applications. In: Allogower, F. and Zheng, A. *Nonlinear model predictive control.* Birkhauser. 369-392.

Rawlings, J. B. and Muske, K. R. (1993). Stability of constrained receding horizon control. *IEEE Transaction on Automatic Control*. 38(10): 1512-1516.

Richalet, J., Rault, A., Testud, J. L. and Papon, J. (1978). Model Predictive Heuristic Control: Application to industrial Process. *Automatica.* 14(5): 413-428

Ravi Sriniwas, G., Arkun, Y., Chien, I. L. and Ogunanike, B. A. (1995). Nonlinear Identification and Control of a High Purity Distillation Column. *Journal of Process Control*. 5:149-162.

Ricker, N. L. and Lee, J. H. (1995). Nonlinear Model predictive control of the Tennesse Eastman challenge process. *Computer and Chemical Engineering.* 19: 961-981.

Serra, M., Perrier, M., Espna, A. and Puigjaner, L. (2001). Analysis of different control possibilities for a divided wall column: feedback diagonal and dynamic matrix control. *Computers and Chemical Engineering.* 25:859-866.

Shaw, A. M. and Doyle III, F. J. (1997). Multivariable nonlinear control applications for a high purity distillation column using a recurrent dynamic neuron model. *Journal of Process Control*. 7(4): 255-268.

Shaw, A. M., Doyle III, F. J. and Schwaber, J. S. (1995). A Dynamic Neural Network Approach to Nonlinear Process Modeling. *Computers and Chemical Engineering.* 21(4):371-385.

Shiskey, F. G. (1984). *Distillation Column*. 2nd edition. New York: McGraw-Hill.

Sjoberg, J. and Ljung, L. (1995). Overtraining, regularization, and searching for minimum in neural networks. *Internatonal Journal of Control.* 62(6): 1391-1408.

Skogestad, S., Lundstrom, P. and Jacobsen, E. W. (1990). Selecting the Best Distillation Control Configuration. *AIChE Journal*. 36(5): 753-765.

Skogestad, S and Morari, M. (1988). Understanding the dynamic behavior of distillation columns. *Industrial Engineering and Chemistry Research*. 27:1848-1862.

Skogestad, S. and Morari, M. (1988). LV control of a High-Purity Distillation Column. *Chemical Engineering Science*. 43(1): 33-48.

Skogestad, S Morari M. and Doyle, J. C. (1988). Control of ill conditioned plant: High purity distillation column. *IEEE Transaction of Automatic Control.* 33(12):1092-1105.

Skogestad, S and Hovd, M. (1990). Use of frequency RGA for control structure selection. *Proceeding of American Control Conference*. May 1990. San Diego: 2133-2139.

Skogestad, S. (1997). Dynamics and control of Distillation Control. *Transaction of IChemE.* (75). Part A, 539-562.

Smith, C. L. (1972). *Digital Computer Process control*. Intex Educational Publishers. Scranton.

Sorensen, P. H., Norgaad, M., Ravn, O. and Poulsen, N. K. (1999). Implementation of neural network based nonlinear predictive control. *Neurocomputing*. 28: 37-51.

Su, H. T., McAvoy, T. J. and Werbos, P. (1992). Long Term Prediction of Chemical process using recurrent neural network: a parallel training approach. *Industrial Engineering and Chemistry Research*. 31:1338-1352.

Wasif, N (2001). *Nonlinear predictive control using Genetic Control Algorithm.* King Fahd University of Petroleum and Mineral: Thesis of Master.

Xia, P. Q. (2003). An inverse of MR damper using optimal neural network and system identification. *Journal of Sound and Vibration*. 266:1009-1023.

Yamamoto, S. and Hashimoto, I. (1991). Present status and future needs: the view from Japanese industry. *Proceeding of CPV-IV*. San Padre Island.

Zafiriou, E. (1990). Robust model predictive control of processes with hard constraints. *Computer and Chemical Engineering.* 14(4): 359-371.

Zhan, J. X. and Ishida, M. (1994). The Multi Step Predictive Control of Nonlinear SISO Processes with a Neural Model Predictive Control Method. *Computer and Chemical Engineering*. 21:201-210.

Zheng, A. (1997). A Computational efficient nonlinear linear model predictive control algorithm. *Proceeding of the American Control Conference*. Albuquerque, NM.