

GROUPED NEURAL NETWORK MODEL-
PREDICTIVE CONTROL AND ITS EXPERIMENTAL
DISTILLATION APPLICATION

By

JING OU

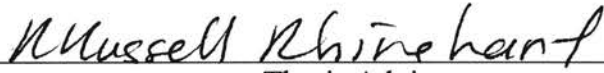
Bachelor of Engineering
Zhejiang University
Hangzhou, People's Republic of China
1992

Doctor of Philosophy
Zhejiang University
Hangzhou, People's Republic of China
1997


Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILISOPHY
August, 2001

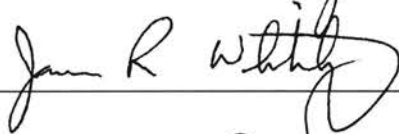
GROUPED NEURAL NETWORK MODEL-
PREDICTIVE CONTROL AND ITS EXPERIMENTAL
DISTILLATION APPLICATION

Thesis Approved:



Thesis Adviser









Dean of the Graduate College

ACKNOWLEDGEMENTS

I am extremely grateful to my research advisor, Dr. R. Russell Rhinehart. In my eyes, he is a perfect advisor. He gave me full independence and freedom during the entire period of this research while directed, advised and helped me promptly whenever needed. His open-mindedness, his flexibility, his knowledgeableness, his enthusiasm in the research, his encouraging and helping attitude towards everybody, all make my research journey a very enlightening, rewarding, and pleasant one.

I thank my committee members, Dr. James Robert Whiteley, Dr. Arland "AJ" Johannes, Dr. Martin Hagan, and Dr. Gary Yen for their support and encouragement. I thank Dr. AJ for fixing all the headaches caused by my computer. Thanks also go to Mr. Charles Baker for his continuous support of the experimental work.

I am thankful to Mr. Yong Hu, Mr. Qing Li, Dr. Sharad Bhartiya, Mr. John Szela, and Ms. Phoebe Katterhenry for their generous help and helpful discussions.

Thanks are extended to the Edward E. and Helen Turner Bartlett Foundation for its financial support.

I dedicate this thesis to my parents, who give me unselfish, endless, and perfect love all the time.

TABLE OF CONTENTS

Chapter	Page
I. INTRODUCTION	1
II. LITERATURE SURVEY	7
2.1 Overview	7
2.2 Recent Development of NMPC	12
2.2.1 NMPC with First-Principle Model	12
2.2.2 NMPC with Semi-Empirical Model	13
2.2.3 NMPC with Empirical Model	14
2.2.4 Linearized NMPC	16
III. GROUPED NEURAL NETWORK MPC (GNNMPC)	18
3.1 Background – NN Dynamic Model for Control	18
3.2 Grouped Neural Network (GNN) for MPC	22
3.3 Process Model Mismatch (PMM) Compensation in GNNMPC	26
3.4 Objective Function in GNNMPC	28
3.5 Summary	29
IV. EXPERIMENTAL SETUP	31
4.1 Distillation Unit	31
4.2 Instrumentation	35
4.3 Data Acquisition and Control (DAC) System	36
4.4 GNNMPC System for the Distillation Column	40
V. GNNMPC AND THE SIMULATED DISTILLATION	41
5.1 First-Principles Dynamic Model of Methanol-Water Distillation Column	41
5.2 Operating range	45
5.3 Open-Loop Responses	46
5.4 GNN Modeling	57
5.5 Process Model Mismatch (PMM)	68
5.6 Control Performance of GNNMPC	71
5.6.1 Setpoint Tracking and Controller Aggressiveness	72
5.6.2 Disturbance Rejection	79
5.6.3 Constraint Handling	85

Chapter	Page
VI. GNNMPC AND THE EXPERIMENTAL DISTILLATION PROCESS	93
6.1 Operating Range	93
6.2 Open Loop Responses	94
6.3 GNN Modeling	102
6.4 “Goodness” of GNN	107
6.5 Configuration of GNNMPC	113
6.6 Control Performance of GNNMPC	116
VII. SUMMARY AND CONCLUSIONS	126
7.1 Summary	126
7.2. Conclusions.....	127
VIII. DISCUSSION AND RECOMMENDATIONS	129
BIBLIOGRAPHY	135
APPENDIXES	141
A. CALIBRATIONS FOR THE DISTILLATION UNIT	141
A.1 Heating Power for the Reboiler	141
A.2 Flow Rates	142
A.3 Levels	144
A.4 Column Pressure	146
A.5 Refractometer	146
B. CORRELATIONS OF THERMODYNAMIC PROPERTIES IN THE METHANOL-WATER SYSTEM	147
B.1 Vapor-Liquid Equilibrium (VLE)	147
B.2 Enthalpy	149
B.3 Density	149
C. MATLAB CODES FOR THE TRAY-TO-TRAY DYNAMIC MODEL OF THE METHANOL-WATER DISTILLATION COLUMN	152
C.1 Initialization	152
C.2 Main Calculation Procedures	154
C.3 Correlations of Thermodynamic Properties	156
D. GNNMPC MATLAB CODES	159
D.1 GNNMPC for Distillation Simulator	160
D.1.1 Main Subroutine	160
D.1.2 GNNMPC Initialization	161
D.1.3 GNN Model	162
D.1.4 PMM	162

Chapter	Page
D.1.5 Objective Function	163
D.1.6 Auxiliary Subroutines	164
D.2 GNNMPC for Experimental Distillation	166
D.2.1 Main Subroutine	166
D.2.2 GNNMPC Model	169
D.2.3 PMM	170
D.2.4 Objective Function	172
D.2.5 Interfaces with Camile	173
D.2.6 Auxiliary Subroutines	177
E. TUNING OF THE CONTROLLER – CV DAMPING VERSUS MOVE SUPPRESSION	179
E.1 CV Damping and Move Suppression for Tuning MPC	179
E.2 GNNMPC for the Distillation Process Using CV Damping	180
E.2.1 The CV Damping Approach	180
E.2.2 Setpoint Tracking	182
E.2.3 Disturbance Rejection	185
E.3 Summary	190

LIST OF TABLES

Table	Page
5.1 Operating range and nominal point of the simulated distillation process	46
5.2 Steady state analysis in the local open-loop study	47
5.3 Steady state analysis in the wide range open-loop study	56
5.4 GNN structure parameters	57
5.5 Training results of the GNN model	62
5.6 Predicted step response versus the actual step response	64
5.7 Predicted step response versus the actual step response	64
5.8 Tuning parameter for the GNNMPC	72
5.9 Setpoint tracking case studies	73
5.10 Disturbance rejection case studies	80
5.11 Effect of disturbances on the process – Open-loop studies of Cases 5 to 8	85
5.12 Constraint handling case studies	86
5.13 Constraint on the rate-of-change of MVs	88
6.1 Operating range of the experimental distillation column	93
6.2 Steady state values under step input studies	99
6.3 Model structure and training results of the GNN	104
6.4 Tuning parameters of GNNMPC for the experimental distillation process	115
6.5 Experimental demonstration cases for GNNMPC	116
A.1 Calibration results for feed flowrate and reflux flowrate under the structure	143

Table	Page
B.1 VLE for methanol-water system at standard pressure (1 atm)	148
B.2 Enthalpy data for the methanol-water system under standard pressure (1 atm)	149
B.3 Density data for the methanol-water system	150
D.1 Default stopping criteria of the Matlab optimization algorithm	159

LIST OF FIGURES

Figure	Page
1.1 General structure of MPC	2
3.1 Selective predicted CV points and future MV points in GNNMPC	24
3.2. The GNN model structure	26
4.1 Front view of the distillation unit	32
4.2 Back view of the distillation unit	33
4.3 Schematic diagram of the distillation process	35
4.4 Camile [®] interfaces	38
5.1 Open-loop response to step change of reflux flowrate	49
5.2 Open-loop response to step change of heating power	50
5.3 Open-loop response to step change of feed flowrate	51
5.4 Step responses at the neighborhood of $\{R, H, F\} = \{90, 60, 250\}$	53
5.5 Step responses at the neighborhood of $\{R, H, F\} = \{120, 50, 280\}$	54
5.6 Step responses at the neighborhood of $\{R, H, F\} = \{150, 45, 310\}$	55
5.7 Input-ouput series for training the GNN	59
5.8 Input-output series (explosion of 5.7)	60
5.9 Training results for the testing set (performance of NN3)	65
5.10 Predicted step response versus the actual step response	66
5.11 Predicted step response versus the actual step response	66
5.12 Discrepancy of GNN outputs	67

Figure	Page
5.13 Modeling error	69
5.14 Predicted step response versus the actual step response (w/ modeling error)	70
5.15 Predicted step response versus the actual step response (w/ modeling error)	70
5.16 GNNMPC for setpoint tracking (Case 1)	74
5.17 GNNMPC for setpoint tracking (Case 2)	75
5.18 GNNMPC for setpoint tracking (Case 3)	77
5.19 GNNMPC for setpoint tracking (Case 4)	78
5.20 GNNMPC for disturbance rejection (Case 5)	81
5.21 GNNMPC for disturbance rejection (Case 6)	82
5.22 GNNMPC for disturbance rejection (Case 7)	83
5.23 GNNMPC for disturbance rejection (Case 8)	84
5.24 MV constraint due to setpoint change (Case 9)	87
5.25 MV constraint due to feed disturbance (Case 10)	89
5.26 MV constraint due to feed composition disturbance (Case 11)	90
5.27 Constraint on the MV movement (Case 12)	91
6.1 Step response Case 1	95
6.2 Step response Case 2	96
6.3 Step response Case 3	97
6.4 Noise level of top and bottom compositions (explosion of 6.2)	98
6.5 Effect of entraining on the process characteristics	101
6.6. Training performance of GNN	105
6.7 Training performance for the column pressure	107

Figure	Page
6.8 Comparison of GNN prediction with the running results for Case 1	109
6.9 Comparison of GNN prediction with the running results for Case 2	110
6.10 Comparison of GNN prediction with the running results for Case 3	111
6.11 Steady state PMM for cases listed in 6.2	112
6.12 GNNMPC performance for setpoint tracking with sluggish tuning parameter	117
6.13 GNNMPC performance for setpoint tracking with aggressive tuning parameter ..	119
6.14 GNNMPC performance for disturbance rejection	121
6.15 Open loop response to feed flowrate change	122
6.16 GNNMPC performance for handling MV constraint	124
6.17 GNNMPC performance for state variable constraint handling	125
A.1 Calibration curve of the refractometer for methanol-water system	146
B.1 VLE plot of the methanol-water system under standard pressure (1 atm)	148
E.1 GNNMPC for setpoint tracking	183
E.2 GNNMPC for setpoint tracking	184
E.3 GNNMPC for feedforward disturbance rejection	186
E.4 GNNMPC for feedforward disturbance rejection	187
E.5 GNNMPC for unmeasured disturbance rejection	188
E.6 GNNMPC for unmeasured disturbance rejection	189

CHAPTER 1

INTRODUCTION

The complexity of the behavior of a chemical process is multi-dimensional. Nonlinear process behavior (statically and/or dynamically), multivariable process interactions, and external disturbances are typical characteristics of a chemical process. Surprisingly, for decades, such complex processes were controlled, decently, by a simple Proportional-Integral-Derivative (PID) controller. The PID controller is a linear controller, so it does not “understand” the nonlinearity of the process. The PID controller is a single-loop controller, so it is “bad” at handling interactions. The PID controller is a feedback controller, so it is slow at responding to external disturbances. Achievement of decent PID control on a complex process relies heavily on both successful design of the process and clever arrangements of controllers.

Fortunately, many processes can be very well approximated linearly near the nominal operating points, allowing linear control schemes such as PID to be functional. But still, there are processes that are too challenging for a PID control scheme. For example, when a process is highly nonlinear or interactive, the PID controller can rarely do a good job. If there were no other alternatives and the PID control scheme has to be used, control performance is sacrificed.

Made possible by the development of the digital control technology, the concepts of process control have undergone a significant change. Availability of inexpensive and speedy digital computers provides a platform to develop and implement more complex control schemes. Among a vast variety of these advanced (or modern) control schemes,

this work focuses on one type of controller whose concepts have been very well accepted in both academia and industry – Model Predictive Control (MPC).

MPC is an optimal-based control scheme. An explicit predictive model (also referred to as a “dynamic model” in some literature) is needed to predict future process behavior. The prediction model is then used to evaluate and optimize a control-performance-related objective function to produce the desired control actions. Commonly, a MPC strategy consists of three parts, the prediction model, the model adjustment, and the constrained optimizer. Figure 1 shows the general structure of MPC in the discrete time domain.

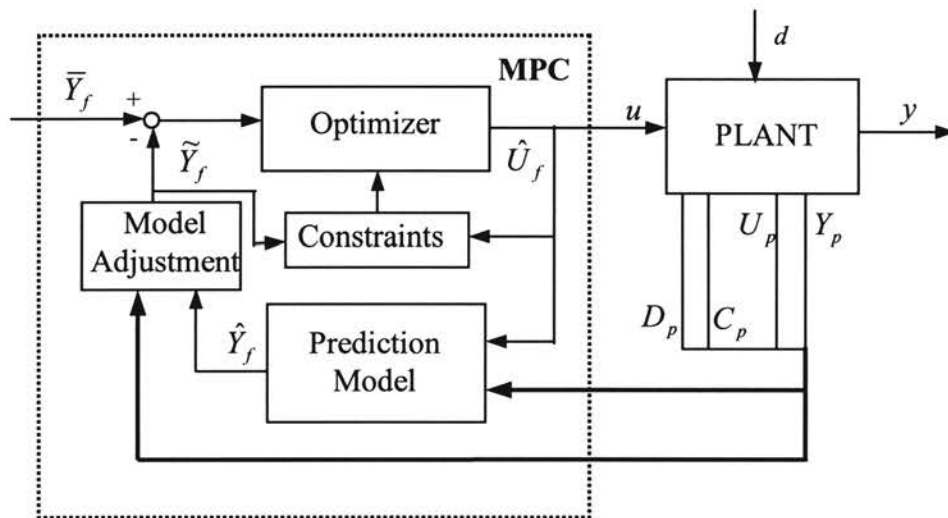


Figure 1.1 General structure of MPC

An upper case letter in Figure 1.1 indicates a series of variable values along a continuous discrete time scale. It is a vector in the case of a SISO system and a matrix in the case of a MIMO system. Subscripts p and f represent *past* (including present values) and *future* respectively. A lower case letter represents the *present* value, which is

a scalar for a SISO system and a vector for a MIMO system. The letter “ u ” represents Manipulated Variables (MV’s), “ y ” Controlled Variables (CV’s), “ d ” disturbances and “ c ” plant parameters. The optimizer of MPC proposes a series of future control actions (control sequence), \hat{U}_f , so that the objective function, which is chosen to meet certain operation objectives (for example, the quadratic error criterion for a future period in the case of regulation to a setpoint), is optimized. The optimizer is subject to constraints imposed by the plant characteristics (physical limitations of the control and process equipment) or plant operating strategies. Only the first control signal of the control sequence, $\hat{U}_f(1)$, is utilized to manipulate the plant. Since it is the actual input to the plant, it is denoted as u in Figure 1.1. At the next control instant, a new optimization problem with updated parameters based on currently available plant information is formulated and solved. This is called receding strategy and it is the reason for another name of MPC, Receding Horizon Predictive Control (RHPC). Available plant information, variables Y_p , U_p , D_p , and C_p are used to adjust the prediction model as partial feedback (referred to as model adjustment, residual, or Process Model Mismatch (PMM) compensation in this work).

The advantages of the MPC scheme are obvious. It handles interactions and constraints directly, and the structure is very flexible to accommodate most process characteristics. Also as obvious are the essential role of the predictive model, as well as the computation issue as the optimization is involved. The predictive model is an integral part of the optimization algorithm, therefore, model accuracy and simplicity play an important role on the capability and performance of the controller. The optimizer, instead of optimizing only the next control action, has to optimize *a series of* future control

actions subject to a variety of constraints; therefore the dimension of the optimization problem grows very fast as the dimension of the controller grows. Hence the concern of the speed of the optimization algorithm rises for online implementation.

Linear MPC (LMPC) uses a linear predictive model, and therefore leads to a Linear Programming (LP) or Linear Quadratic Programming (LQP) method for the optimization part. LMPC products such as DMC[®] and IDCOM[®] (Qin and Badgwell, 1997), have been successfully applied and well accepted in industry. MPC has established a reputation of reducing costs and saving money in industry since late 1970s when the first papers on the applications of DMC[®] (Cutler & Ramaker, 1979) and IDCOM[®] (Richalet, Rault, Testud & Papon, 1978) to industrial problems were reported. This success contributes to the much interest and surging research work in Nonlinear MPC (NMPC), a direct extension of LMPC to nonlinear processes, from both academia and industry.

Linear models are very well suited for MPC because they may be solved quickly, and the optimization problem may be posed as linear or quadratic programming problems, for which robust and reliable software is available. Nonlinear models better represent the process characteristics and are therefore expected to produce better control performance. However, nonlinear equation processing adds much complexity to the MPC scheme. Robust modeling approaches and speed of the optimization algorithm become major concerns for practicing NMPC.

The predictive model plays a critical role in MPC. It expresses the controller's "knowledge" of the process. It is also an integral part of the optimization algorithm, and therefore influences the formulation and solution of the optimization problem. Since there

is no universal method for representing a nonlinear process, a variety of modeling methods ranging from first-principles model, semi-empirical model (gray-box model or hybrid model), to empirical model (black-box model) have been under research, for better representation of the process and for better incorporation into the MPC scheme.

In our work, an empirical modeling approach, the Neural Network (NN) modeling method, is used in a unique way, and is referred to as Grouped-Neural-Network (GNN) model. The proposed predictive model, instead of predicting continuously in the future time span, only provides the prediction at selected future discrete time instants. What the proposed work tries to achieve is to eliminate all the assumptions made on the process characteristics while alleviating as much as possible the computational burden on the controller due to the complexity of the predictive model.

This new strategy, incorporating some heuristic choices, is demonstrated to work effectively both by simulations and experiments. The experimental setup is a lab-scale distillation column. The column is operated with a binary methanol-water system at atmospheric pressure. Top and bottom compositions are controlled using reflux flowrate and reboiler's heating power as the manipulated variables. Process characteristics include nonlinear multivariable interactions, non-ideal thermodynamic behavior, significant unintentional disturbances, and operating constraints on both manipulated and auxiliary variables. Experiments would demonstrate effectiveness of the controller for handling constraint, interactions and nonlinearity of the process.

This study features experimental results. The process control literature is more focused on theoretical development and simulation studies of advanced control strategies, but there are few reported references regarding experimental implementation of these

techniques. More experimental control research and real process data are needed to quantitatively compare the wide variety of advanced control methods developed, to focus the theoretical work to solve implementation problems, and to establish credibility within the practice community.

CHAPTER 2

LITERATURE SURVEY

2.1 Overview

Model Predictive Control (MPC) does not designate a specific control strategy. It represents a wide range of control methods which make an explicit use of a dynamic model (referred to as predictive model or prediction model in MPC) of the process and which obtains the control signals by minimizing an objective function, which is often related to the control performance. For the scope of this work, the survey is on those MPC strategies that are similar to the DMC structure, which have deterministic predictive model. The MPC strategies that consider stochastic issues such as the Generalized Predictive Control (GPC) (Clarke, Mohtadi & Tuffs, 1987 a,b) are beyond this work's scope.

MPC grew up in the oil and chemical industries, where MPC was found very effective and successful in dealing with multivariable constrained problems. In less than 20 years since the first papers on MPC applications (Richalet, Rault, etc., 1978; Cutler & Ramaker, 1979) were published, the MPC technology has grown rapidly. Nowadays, MPC has become by far the most popular and successful one in industry among all kinds of advanced process control technologies. According to an industrial survey done in 1997 (Qin & Badgwell), there are over 2,000 applications of MPC covering refining, petrochemicals, chemicals, pulp and paper, food processing, automotive, aerospace, etc.. Among many reasons for the success of MPC including availability of cost effective high performance computer facilities, a key advantage of MPC is that it is particularly

attractive to the industrial personnel because the concepts are very intuitive and at the same time the tuning is relatively easy.

Both the success of MPC in industry and issues which arose from practicing MPC have been driving theoreticians busy in understanding and providing theoretical insights and support to the MPC technology. MPC started from linear system, which is called LMPC. In LMPC, a linear predictive model is used to represent the process dynamics, and the optimization problem is formed as a Linear Programming (LP) or Linear Quadratic Programming (LQP) problem. For LMPC, while there are still many issues under research such as the stability analysis of constrained finite horizon systems (Primbs & Nevistic, 2000a, 2000b; Bergounioux & Kunisch, 2000; Lee, Kwon, & Choi, 1998), issues such as feasibility of the on-line optimization, closed-loop stability and performance are largely understood. Two review papers are referred on such theoretical issues (Garcia, Prett, & Morari, 1989; Morari & Lee, 1999). The paper by Lundstrom, Lee, Morari and Skogestad (1995) discussing the limitations of DMC, the most popular MPC strategy in industry, provides deep insights into LMPC.

While theoreticians were working on theoretical analysis of LMPC, based on success of LMPC in industry, practitioners have been exploring applications of nonlinear MPC (NMPC), which has a nonlinear predictive model and is a direct and intuitive extension of LMPC. The interest, of course, is due to the argument that many real processes can not be well represented by a linear model, and it is expected that with a nonlinear model, which can better represent the process, better control performance will result.

Among the so called nonlinear MPC, two general approaches are observed in open literature. One is referred by the author as a linearized NMPC. It uses Linear MPC (LMPC) techniques, especially the linear optimization technique used in LMPC. However, the model can represent nonlinear characteristics of the process. The other one is a “thorough” NMPC, nonlinear model is introduced into the MPC structure, the optimization problem becomes non-convex, new techniques are needed to deal with the problems.

The most commonly adopted idea behind a variety of linearized NMPC schemes is the piecewise linear model (multiple model, multi-model, or multi-mode) approach. This approach is usually suitable for a process with moderate nonlinearity. There are many processes that operate in a wide range, locally, it can be very well represented by a linear model, however, in the whole operating range, one single linear model is not representative. In the piecewise model approach, the operating range is divided into several regions, each region can be well represented by a linear model. For this approach, how to divide the operating range and how to transit between sub-regions are the major issues.

As to so called “thorough” NMPC, the modeling method can be classified to 3 approaches, the first-principles model, the semi-empirical model, and the empirical model.

The first-principles model is derived from first-principles, i.e., material balance, energy balance, momentum balance, together with information on the hydraulic and thermodynamic properties of the process. GMC (Lee & Sullivan, 1988) and PMBC (Rhinehart and Riggs, 1991) apply such modeling approach. The first-principles model

provides insight into the characteristics of the process, and provides additional help to process owners by providing deep understanding of the mechanics of the process. However, a first-principles model is not always available. The process may be so complicated that deriving a first-principles model is either not feasible or very cost inefficient. Incorporating the first-principles model into MPC may also encounter computation problems. Internal convergence loops may fail to converge. The model may be so complicated that it is not convenient to be incorporated into the optimizer of MPC. Or, the computation burden of solving the model becomes so heavy as to hinder the online implementation of MPC.

The empirical model, on the other hand, applies available process data (online measurements) to identify the relations between the process variables of interest. The empirical model is often called data-oriented modeling technique. The modeling approach assumes that the process characteristics are well embedded in the data and can be extracted from the data by using a proper technique. Neural Networks (NNs) is one of the most popular empirical modeling method (Hussain, 1999). The empirical model requires least knowledge of the process mechanics. Comparing to a first-principles model, an empirical model usually does not provide information on the mechanics of the process, its internal structure and parameters can not be interpreted in terms of physical effect or process parameters. Further, the empirical model can only be used to represent the process in the operating region that the model has been identified, and has unreliable extrapolation capability. For incorporation of an empirical model into MPC, the model often poses as a nonlinear constraint for the optimizer. Good examples of MPC using NN

modeling technique can be found in papers of Bhat and McAvoy (1990), Hernandez and Arkun (1992), Temeng, Schnelle, and McAvoy (1995).

Between the first-principles model and the empirical model is the semi-empirical model. This modeling approach attempts to take advantage of both the knowledge of the process mechanics and available process data. For example, in a first-principles model, there are some parameters whose values are not directly known but can be obtained from an empirical modeling method. Or, vice versa, in an empirical model, some parameters in the model can be determined by a prior knowledge of the process, such as the order of the process, the structure of the model, etc. Strictly saying, most models are by some extend a semi-empirical model. For this survey, a modeling method is identified as a semi-empirical modeling method when knowledge of the process mechanics is used to determine the model structure and values of the parameters in the model structure are then found using an empirical modeling method. This approach is in some literature called parametric modeling.

Because nonlinear processes can not be represented by a universal method, which modeling approach to use for a nonlinear process is case-oriented. All the above three approaches are abundant in the open literature. The review papers by Henson (1998) and Rawlings (2000) cover modeling issues for incorporation into MPC.

Theoretical results for NMMC are by far very limited, as for other nonlinear control technologies (Mayne, Rawlings, Rao & Scokaert, 2000). Morari & Lee (1999) pointed out some ideas in NMPC analysis for future exploration. Nevertheless, a limit of theoretical support does not prevent practitioners from exploring the results and advantages brought by using a nonlinear model. Henson (1998) provided a review on

some issues faced in practicing NMPC, including modeling, optimization, calculation, and tuning. After a survey of industrial applications of LMPC in 1997 (Qin & Badgewell, 1997), a survey of industrial applications of NMPC appeared in 1998 (Qin & Badgewell, 1998), indicating the thriving interest in NMPC.

2.2 Recent Development of NMPC

In this section, a survey of the recent practice (after 1998) on NMPC is provided. Most of the surveyed papers can be classified according to their modeling approach. Other issues in MPC are closely related to the modeling approach. Therefore, this survey is divided into several parts according to the modeling approach.

2.2.1 NMPC with First-Principles Model

Roffel, Betlem and Ruijter (2000) developed a rigorous first principles dynamic model for a heat-integrated cryogenic distillation process. However, the first-principles model is not directly incorporated into the MPC scheme, instead, the first-principles model is used to generate data from which classical linear time series identification is used to develop the conventional DMC type LMPC. MPC was shown in their work to be very effective in handling the strong interaction of the process. It was also concluded that due to the strong interactive nature of the process variables, process changes have to be made slowly, since otherwise manipulated variables easily saturate and process output targets can not be maintained.

Muske, Howse, Hansen and Cagliostro (2000) describe the MPC technique which also uses a detailed heat transfer model in an industrial steel facility. Batch nonlinear

least squares estimation is used to update the predicted temperature profile of the hot blast stoves and heat transfer coefficients. These estimated parameters are then used by MPC to determine the minimum fuel required for the subsequent regenerative cycle.

Pickhardt (2000) presents the application of NMPC to the distributed collector field of a solar power plant using a first-principles model. The parameters of the model are estimated on-line in order to compensate for time-varying effects and modeling errors. Experimental results are presented.

2.2.2 NMPC with Semi-Empirical Model

Zhu, Henson and Ogunnaike (2000) proposed an approach to combine LMPC and NMPC. Some processes can be decomposed into approximately linear subsystems and highly nonlinear subsystems. The linear subsystem and nonlinear subsystem interact via mass and energy flows. LMPC is applied to the linear subsystems and NMPC is applied to the nonlinear subsystems. A simple controller coordination strategy that counteracts interaction effects is proposed for the case of one linear subsystem and one nonlinear subsystem. A reactor/separators process with recycle is used to compare this hybrid method with conventional LMPC and NMPC techniques.

Norquay, Palazoglu and Romagnoli (1999) found that a Wiener model, consisting of a linear dynamic element followed in series by a static nonlinear element, is ideal for representing a high-purity distillation column. The Wiener model is relatively simple model requiring little more effort in development than a standard linear step response model, yet offer superior characterization of systems with highly nonlinear gains. The Wiener model may be incorporated into MPC in a unique way that effectively removes

the nonlinearity from the control problem, preserving many of the favorable properties of LMPC, especially in the analysis of stability. The paper describes the application of Wiener MPC (WMPC) to an industrial C2-splitter at the Orica Olefines plant. The same idea was also implemented on an experimental pH neutralization apparatus and the results was compared with benchmark proportional integral derivative (PID) and LMPC strategies considering the effects of output constraints and modeling error (Norquay, Palazoglu and Romagnoli, 1999).

2.2.3 NMPC with Empirical Model

Kambhampati, Mason and K. Warwick (2000) applied a Radial Basis Function Neural Network modeling approach to represent a simulated continuous-stirred tank reactor (CSTR). The focus on this paper, however, is to investigate the stability of one-step ahead predictive controllers based on nonlinear models. The paper showed that, under conditions which can be fulfilled by most industrial plants, the closed-loop system is robustly stable in the presence of plant uncertainties and input-output constraints. There is no requirement that the plant should be open-loop stable and the analysis is valid for general forms of nonlinear system representation.

Zamarreño and Vega (1999) derived a recurrent neural network model and used in NMPC for a simulated sulphitation tank taken from the sugar industry. The neural network model is able to represent the system in the state-space form. It thus is able to deal with constraints in the system variables as well as highly nonlinear dynamics.

An extended Kalman filter (EKF)-based NMPC is developed (Ahn, Park & Rhee, 1999) for the property control of an experimental continuous polymerization reactor with

the availability of a first-principles model. The NMPC controls both the conversion and the weight-average molecular weight of the polymer product using jacket inlet temperature and the feed flow rate.

Berenguel, Arahál and Camacho (1998) described an industrial application of NMPC using neural networks to a solar power plant. The neural network is a feedforward (static) neural network in an autoregressive configuration. Modeling issues such as the selection of the input variables, the input/output vectors for training, and the neural network structure, are discussed. In particular, an algorithm is proposed to obtain the number of past values of the measured variables needed to feed the neural network.

Galván and Zaldívar (1998) practiced real-time nonlinear inverse and predictive control using a recurrent neural network. The controlled process is the heat transfer fluid temperature in a pilot chemical reactor. The training of the inverse control system is carried out using both generalized and specialized learning, which allows the preparation of weights of the controller to act in real-time, and appropriate performances of inverse neural controller to be achieved. The predictive control system makes use of a neural network to calculate the control action. Thus, the problems related to the high computational effort involved in NMPC systems are reduced.

Daoping and Cauwenberghé (1998) considered modeling error in their approach of NMPC using a neural network model. Several on-line corrections of long-range predictive outputs and neural-network-based multiple feedback predictive control for nonlinear cascade industrial processes are proposed. A variable correction coefficient and its design approaches are presented.

Balasubramhanya and Doyle (2000) used traveling wave phenomena to develop low order models for a multicomponent reactive distillation columns and demonstrated the inherent trade off between model accuracy and computational tractability for MPC applications.

Roubos, Molloy, Babuka and Verbruggen (1999) developed a NMPC scheme using fuzzy model. The process model is derived from input-output data by means of product-space fuzzy clustering. Two methods are also proposed to deal with the nonlinear optimization problem. One is to use a branch-and-bound searching method with iterative grid-size reduction. The other method is to use LMPC locally. The system that are under study is a simulated MIMO liquid level process with two inputs and four outputs.

Fischer, Nelles and Isermann (1998) also proposed a NMPC scheme based on fuzzy model. The process is an industrial-scale cross-flow water/air heat exchanger with the temperature under control. A fuzzy model of the process is identified from measurement data. Guidelines for the generation of proper excitation signals are given.

2.2.4 Linearized NMPC

Ozkan, Kothare and Georgakopoulos (2000) modeled a nonlinear system as a set of piecewise linear or affine systems. The paper provides a brief review on such systems. Using techniques from the theory of linear matrix inequalities (LMIs), a multiple model MPC technique is developed, which utilizes a single quadratic Lyapunov function and multiple local state-feedback matrices, cast the optimization problem as a convex problem involving LMIs. Bemporad, Ferrari-Trecate and Morari (2000), investigated the observability and controllability of such piecewise affine systems.

A piecewise linear model for MPC was also developed by Banerjee and Arkun (1998). The plant operates in several distinct operating regimes, linear models were built for the individual regimes, and then nonlinear models in between these local models (called transition models) were interpolated to match plant dynamics during transitions.

Piche, Sayyar-Rodsari, Johnson, and Gerules (2000) described a commercial version of NMPC by Pavilion Technologies Corporation. An empirical nonlinear model is developed using step test data and historical data. The step test data is used to create a linear dynamic model, and the historical data is used to build a nonlinear steady-state model. The steady-state model is implemented by a neural network. A parsimonious nonlinear dynamic model is then created by combining the linear dynamic model and the nonlinear steady-state model using an advanced form of gain scheduling. Therefore, their NMPC uses LMPC techniques while the process gain varies over the control horizon of the MPC algorithm.

The strategy by Prasad, Irwin, Swidenbank, Hogg (2000) for a constrained, nonlinear thermal power plant is to make use of successive linearization and recursive state estimation using extended Kalman filtering to obtain a linear state-space model with the availability of a physical model for the process.

A feedback linearization was proposed in NMPC using neural network model (Botto and Costa, 1998). The process is modeled with an affine combination of multilayer feedforward neural networks, whose structures are suitable to be further integrated into feedback linearization schemes with mild assumptions, thus the general constrained nonlinear optimization problem can be transformed into a convex optimization problem. A simulated highly nonlinear titration process is used for test.

CHAPTER 3

GROUPED NEURAL NETWORK MODEL PREDICTIVE CONTROL (GNNMPC)

The novel Nonlinear Model Predictive Control (NMPC) strategy developed in this work is called Grouped Neural Network Model Predictive Control (GNNMPC), whose novelty lies mainly in its modeling part. The predictive model, the GNN, is a group of separately trained Neural Networks (NNs), each predicting process outputs at a specific step in the future, given historical input-output data series. This modeling idea is introduced to relieve the burden of computation under the MPC structure while maintaining decent prediction accuracy for desired control performance. The following sections describe the development of the GNN and general issues when incorporating the GNN model into the MPC structure.

3.1. Background – NN Dynamic Model for Control

NN modeling is a subset of empirical modeling. It has become popular in NMPC due to the difficulty of either obtaining a first-principles model or incorporating a first-principles model into the predictive control scheme (Bequette, 1991; Henson, 1998). In contrast to a first-principles model, an empirical model treats the process as a black box and fits itself to the input-output data obtained from the process, on the assumption that the input-output data can fully, or to a large extent, represent the process's dynamic response. Therefore, much less knowledge on the mechanics of the process is needed. As

with most empirical models, after training, a NN model usually only requires simple algebraic calculations, which makes it more ready for online implementation.

In general, there are two NN forms commonly used for representation of the process dynamics. One is the Feed Forward Neural Network (FFNN) (Werbos, 1990), the other is the Recurrent Neural Network (RNN) (Narendra 1990). The FFNN is static in its structure -- NN inputs propagate forward layer by layer till the output layer is reached, where the NN outputs are obtained. The RNN, on the other hand, is intrinsically dynamic -- outputs of all or part of the nodes are fed back as inputs to all or part of the nodes in the RNN.

Representation of FFNN for dynamics is realized by the NARMAX (Nonlinear AutoRegressive Moving Average model with eXogenous inputs) model structure (Billings and Voon, 1986), which describes process output at the next sampling instant as a function of the process's historical input-output series:

$$\hat{y}(k+1) = f(y(k), y(k-1), \dots, y(k-p), u(k), u(k-1), \dots, u(k-q)) \quad (3.1)$$

A variable name with a hat represents output from a model instead of directly available values (measured process values). In Equation (3.1), y indicates output(s) of the process and u indicates input(s) to the process. The term k is the discrete-time instant counter. The value of $k = 0$ indicates the time instant, 'Now', separating 'Past' and 'Future'. The time instant prior to k by a discrete time period of i is depicted as time instant $k - i$. Time instant j discrete time periods in the future is depicted as time instant $k + j$. The series $y(k), y(k-1), \dots, y(k-p)$ is a reverse-ordered sequence of process outputs from discrete time instant $k-p$ to k . And $u(k), u(k-1), \dots, u(k-q)$ is a reverse-ordered sequence of process inputs from discrete time instant $k-q$ to k . Based on this structure,

An FFNN, is then constructed to have past process input and output series as the inputs to the FFNN and the next-step process output(s) as the desired output(s) from the FFNN. Given enough well-represented input-output data series, the FFNN is trained to best approximate the relationship between its inputs and outputs as embedded in the process's input-output data series and mathematically modeled as the structure expressed in Equation (3.1).

By using the model structure as described in Equation (3.1), however, the FFNN model can only have prediction one-step ahead. This is insufficient for the MPC scheme, because MPC needs a long-range prediction of the process to be able to: 1) determine the distance between the desired future trajectory and the estimated future trajectory, 2) avoid control moves “now” that would create constraint violations in the “future”, and 3) ensure control stability. One way to provide long-range prediction using a FFNN is to 'cascade' the FFNNs, i.e., to use the prediction of one-step ahead, which is generated by one FFNN, as an input to the next FFNN for prediction of two-steps ahead, and so on (Werbos, 1990). Therefore, the prediction can stretch to the prediction range as required by the control scheme. However, model error of one NN will propagate to the following NNs, which will cause degradation of the long-range prediction precision.

An RNN is trained so that, given an initial state (e.g., the present process outputs) and input (e.g., the inputs to the process), the RNN model can produce a best estimation of the future sequence of states, i.e., the dynamics, of the process. An example of the RNN model based on the NARMAX model structure is to feedback the output from the output layer back to the input layer, the NARMAX model structure becomes:

$$\hat{y}(k+1) = f(\hat{y}(k), \hat{y}(k-1), \dots, \hat{y}(k-p), u(k), u(k-1), \dots, u(k-q)) \quad (3.2)$$

Notice that the plant output, y , is not used as the input of the model, but instead, the predicted output, \hat{y} , is used. RNN actually iterates to evolve a sequence of outputs, which approximates the dynamics of the process (Su, McAvoy, & Werbos, 1992). RNN, heuristically and shown by many studies, has considerably greater temporal representational capabilities than FFNN (Puskorius and Feldkamp, 1994). Despite considerable contributions and progress in the development, application, and analysis on RNN (Delgado,1995; Dimopoulos,1995; Tsoi,1997), practical learning algorithms lack for RNN. In the RNN training, the calculation of dynamic derivatives of its outputs with respect to its weights is computationally expensive and often prohibitive. Also, the gradient-based training algorithm is not only slow but also ineffective at finding a good solution for RNN. These are the main reasons of preferring the implementation of FFNN in the framework of NARMAX model to approximate the dynamics of the process. Many commercialized reliable training tools for FFNN are available, such as the Neural Network Toolbox in the Matlab[®] package, which is used in this thesis work to obtain the predictive model.

Targeting at keeping the computation both simple and speedy while remaining accuracy of the model for the MPC scheme, a long-range prediction model, named Grouped Neural Network (GNN) model, is proposed. The GNN comprises a group of independently-trained FFNNs, each predicting a process output at a particular time instant into the future. The GNN approach retains easy access to reliable and speedy training algorithms of FFNN while eliminating the problem of error propagation because the GNN model predicts n -steps in the future without the need of “cascading”. The following section explains how this is realized.

3.2 Grouped Neural Network (GNN) for MPC

Consider a Single-Input-Single-output (SISO) first-order nonlinear system described by: $\dot{y} = f(y, u, w, p)$, where y is the process output, u is the process input, w is the load, and p is the vector of the process parameters. The letter “ f ” in the equation represents a general nonlinear function. For convenience, in the following part, we will not differentiate between two different nonlinear functions, rather, all nonlinear functions are denoted with the letter “ f ”. The Euler form of the discrete time description of the system is $y(k+1) = y(k) + Tf(y(k), u(k), w(k), p(k))$, where T is the time interval for discretization. This equation can be reformulated as $y(k+1) = f(y(k), u(k), w(k), p(k))$. Assume that both the load disturbance, w , and the process parameters, p , are slowly changing variables so that they can be taken as constants within the prediction horizon, the system can be simplified as $y(k+1) = f(y(k), u(k))$, which is the format of a NARMAX model.

It is easy to get the equation for $y(k+2)$ as:

$$y(k+2) = f(y(k+1), u(k+1)) = f(f(y(k), u(k)), u(k+1)) = f(y(k), u(k), u(k+1))$$

Similarly, a general equation for $y(k+i)$ is given as:

$$y(k+i) = f(y(k), u(k), \dots, u(k+i-1)) \quad (3.3)$$

Similar results can be easily obtained for higher order system, for q -th order system (a process described by a q -th order ordinary differential equation), we have

$$y(k+i) = f(y(k-q), \dots, y(k), u(k-q), \dots, u(k), u(k+1), \dots, u(k+i-1)) \quad (3.4)$$

Comparing to the model structure in Equation (3.1), while Equation (3.1) provides a model structure for one-step prediction based on available (past) values of process input(s) and output(s), Equation (3.4) provides a model structure that allows multi-step

direct prediction (long-range prediction) given the information of future process input(s).

The model structure as expressed in Equation (3.4) can be conceptually interpreted as:

$$y_future = function(y_past, u_past, u_future) \quad (3.5)$$

This model structure assumes that with additional information of future inputs to the process, the model can have long-range prediction without cascading NNs. Direct long-range prediction eliminates error propagation caused by cascading NNs. The use of parallel, not cascaded structure accommodates parallel processors for a computational speed advantage. Future inputs to the process are known, under the MPC control scheme, because the future input series are the decision variables of the optimized function (objective function).

In the common MPC practice, the prediction model produces a series of estimated future process outputs at each sampling step in a discrete time scale. This is not necessarily needed. The purpose of the predictive model is to provide the dynamics of the process for comparison with the desired dynamics. Based on the argument that the distance between the desired and predicted trajectories is measured in the Least Mean Square (LMS) sense, prediction of process outputs at each step within the prediction horizon is not necessary, selected points that can represent the trend of the process outputs are enough. This will lead to further relief on computation burden.

This NN modeling approach for MPC, which applies model structure as expressed in Equation (3.5) and predicts only selected points in the future for the usage of the MPC optimizer, is referred to as GNN.

Computation burden can be further decreased. In the common MPC practice, the optimizer optimizes the MVs at each future control interval within the control horizon.

Heuristically, this is also not necessary. MPC uses a receding strategy, which means that although the MVs at each future control interval within the control horizon are optimized, only the first optimized control signal (the MVs at discrete time $k=0$) is utilized to manipulate the plant. It is believed that skipping some points between the next MVs and the MVs at the control horizon will decrease the computation burden of the optimization algorithm due to the decrease of the number of its decision variables (the future MV series) without sacrificing the control performance.

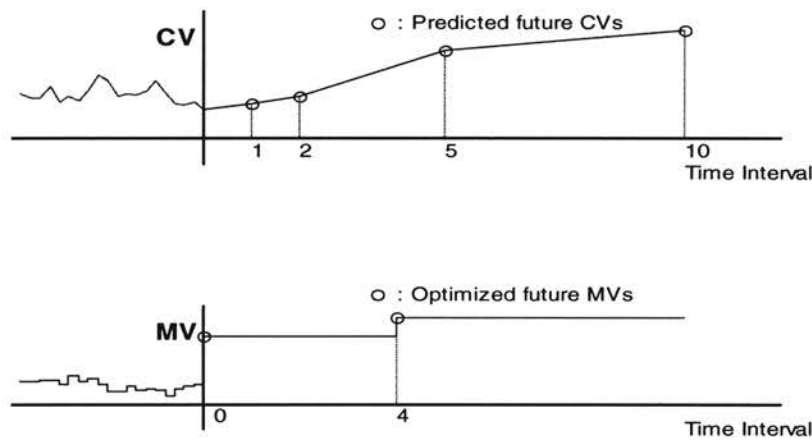


Figure 3.1 Selective predicted CV points and future MV points in GNNMPC

Figure 3.1 illustrates the idea of selective predicted CV points and future MV points (decision variables in the optimization problem) in GNNMPC. “CV” represents Controlled Variables (process outputs). “MV” represents Manipulated Variables (process inputs). The vertical axes, representing time instant “now”, separate the “past” and the “future”. The predictive model produces estimated values at selected future discrete time instants. In the illustration these are at future discrete time intervals 1, 2, 5, and 10, but any appropriate prediction points may be chosen. “Appropriate” means that there are

enough points, at the right places to be able to “see” the process dynamic behavior, yet no more than needed to retain computational ease. Individual NNs are used for each prediction. The line connecting these separately obtained predictions represent the generally predicted trend of the process behavior (but there is no implication that we believe that the process will exactly follow the line). The future input series to the process, needed by the NNs, are actually “guesses” provided by the optimizer of the MPC at $k = 0$, as shown by the lower part of the figure. Illustrated here, the “guessed” process inputs, i.e., the decision variables for the optimizer, change at future time intervals of 0 (“now”) and 4. Circles are values of the decision variables of the optimizer. Inputs are to be kept unchanged between the points by choice.

Note that in the “past”, the CV path is shown by “connecting the dots” which represent CV values at the time intervals. The “connect the dots” lines are angled to make a “saw-tooth” trace. By contrast, the past MV trace shows a square wave pattern. These differences indicates that the CV are discrete measurements with values only at the sampling points, and that the process inputs are held at the value throughout the time interval. Note also that the MV changes at each past interval. Even though the predictive models assume MV changes only at selected future points, at each control interval the optimizer responds to the latest information, calculates a new future sequence, and the new first MV is always implemented, as commonly practiced in MPC.

Figure 3.2 demonstrates the GNN model structure. In this example, the model provides prediction at future steps $l=1, 2, 5, 10$. In Figure 3.2, $\Gamma = [y(k), y(k-1), \dots, y(k-p), u(k), u(k-1), \dots, u(k-q)]^T$, which includes past input and output series of the process. Notice that present input and output values are also included.

This part is the common input of all the NNs inside the dynamic predictive model (the GNN model). For prediction into the future, an additional future input series, $U_{1-l-1} = [u(k+1), u(k+2), \dots, u(k+l-1)] (l > 1)$, is needed.

These NNs are independently trained. This creates the two training benefits of elimination of error propagation within the NN model, and a less intensive training procedure.

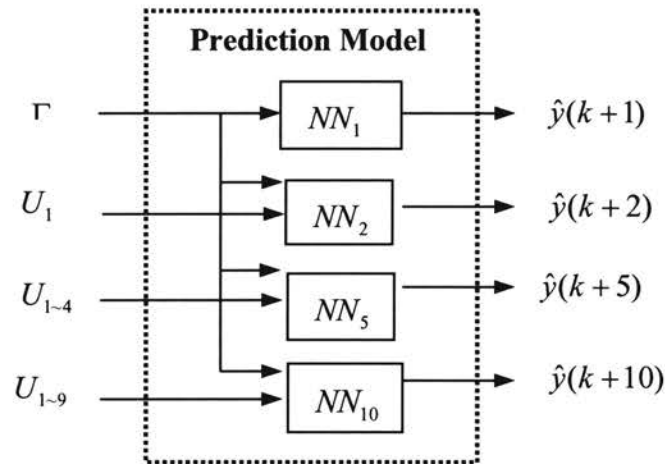


Figure 3.2. The GNN model structure

3.3. Process Model Mismatch (PMM) Compensation in GNNMPC

As briefly described in Chapter 1 and as illustrated in Figure 1.1, MPC composes of three major components: the prediction model which provides long-range estimation of the future process behavior; the model adjustment (PMM compensation) part which uses online available measurements to adjust the predicted future process behavior as a partial feedback; and the constrained optimizer which produce optimal future control action series.

PMM compensation is necessary for the MPC system to provide feedback for stability of the control loop and for removal of the steady state offset. DMC[®]'s (Cutler

and Ramaker, 1979) PMM compensation approach, which is simple yet effective, is widely accepted and practiced for both LMPC and NMPC. In DMC[®], the difference, d , between the present output, y , and the prediction of the present output at last discrete time instant, $\hat{y}(k|k-1)$, is calculated and added to all the future prediction values as adjustment for the predictive model. This approach assumes that differences observed between the process output and the model prediction are due to a step disturbance in the output which persists throughout the prediction horizon. This approach can accurately model setpoint changes which often enter feedback loops as step disturbances, thus provides zero offset for step changes in setpoint. It also approximates slowly varying disturbances. Since PMM can appear as slowly varying output disturbances, the DMC[®] approach provides robustness to PMM.

Unfortunately, directly adoption of the DMC[®]'s PMM compensation approach in the GNNMPC scheme fails to provide zero steady state offset due to the separate training approach for each NN of the GNN model. Since the prediction value at a chosen discrete time instant is from NN models which are separately trained, a discrepancy between the prediction values, the outputs from different NNs exists even when the NNs are given identical input data. We state that the NNs are not "consistent", because given the same input values (as if the process had been at steady state) the NNs would produce slightly different output values. This means that if the process was at steady state with no disturbances or input changes, the several NN models would predict slightly different CV values. This would cause the controller to take action. If we adopt the common feedback practice, the adjusted predictions would still be inconsistent, and control action would lead to steady state offset.

In accordance with the separately-trained approach, a separate PMM compensation approach is proposed. The difference between the measured outputs and the predicted outputs for each NN is separately calculated, and used to adjust each prediction output at each prediction instant, separately. For the example shown in Figure 3.1, the model adjustment procedure is as follows:

$$d_l = y - \hat{y}(k|k-l), \quad l = 1,2,5,10 \quad (3.6)$$

Here, $\hat{y}(k|k-l)$ indicates the process output prediction of time instant k , based on the process information available at time instant $k-l$ and MV information available at time instant k . Then, the differences are added separately to each prediction output:

$$\tilde{y}(k+l) = \hat{y}(k+l) + d_l, \quad l = 1,2,5,10 \quad (3.7)$$

to produce the adjusted prediction, \tilde{y} .

It should be noted that this approach retains the classic and well-accepted MPC assumption that the model error is due to additive step disturbances in the output that persist throughout the prediction horizon.

3.4. Objective Function in GNNMPC

Introduction of the GNN modeling approach does not introduce a constraint on formatting the objective function of the MPC's optimizer. As in the general MPC scheme, the objective function can be freely chosen to meet a variety of process objectives including maximizing profits, minimizing operating costs or energy usage, or minimizing deviation of the process output from a reference trajectory. In this work, we confine the objective function to be of quadratic error criterion, Equation (3.8) describes a most commonly accepted and applied objective function for a SISO system:

$$J = W_{CV} \sum_i (\tilde{y}(k+i|k) - y_{ref}(i))^2 + W_{MV} \sum_j (\Delta u(k+j|k))^2 \quad (3.8)$$

The objective function consists of two parts, the deviation (distance) of the predicted future trajectory from the reference trajectory, and the movements of the future control actions as a punitive factor. i and j are elements of the selected future prediction points and MV movement points, respectively. For the example illustrated in Figure 3.1, $i = 1, 2, 5, 10$ and $j = 0, 4$. $\tilde{y}(k+i|k)$ is the adjusted predicted value of the process at future step i , the value of the reference trajectory at future step i is referred to as $y_{ref}(i)$. $\Delta u(k+j|k)$ is the change of control action, $\Delta u(k+j|k) \equiv u(k+j|k) - u(k+j-1|k)$. The presence of $\Delta u(k+j|k)$ indicates rate of change limit of the control actions for most chemical processes. W_{CV} and W_{MV} are weighting factors of CV deviation and MV movements. The prediction horizon, P, and the control horizon, M, are inferred by the selected prediction points and the future MV movement points: the last prediction point and MV movement point indicates the prediction horizon and the control horizon, respectively. For the example in Figure 3.1, P=10 and M=4.

3.5 Summary

Based on the assumptions that the NARMAX model structure can adequately represent the process dynamics, that load disturbances and process parameters are either constants or slowly changing, and that a least-square distance criteria is used for performance of the controlled variables, the GNN modeling approach is developed for incorporation in the MPC control scheme.

GNN provides prediction at selected future time instants, instead of a continuous consequence of values into the future, which decreases the model structure complexity and the training intensiveness. While using FFNN, no NN-cascading is needed for long-range prediction, prediction is carried out in parallel mode, which accommodates parallel processors for computational speed advantage.

CHAPTER 4

EXPERIMENTAL SETUP

The experimental work presented here is carried out on the fully automated fractional distillation unit in the Unit Operation Lab of School of Chemical Engineering at Oklahoma State University. This chapter describes the experimental setup including instrumentation and Data Acquisition and Control (DAC) system.

4.1 Distillation Unit

The distillation unit is a Technovate[®] Model 9079 fractional distillation system designed for variety of experimental investigations as they pertain to fractional distillation column performance. Figures 4.1 and 4.2 picture the front view and the back view of the unit. The front view pictures the six-tray distillation column, the distillate condenser, the distillate drum, and the computer for monitoring and control. While the column is un-insulated Pyrex glass as shown in the picture, it was insulated for the experimental study of this work to widen the operating range. The back view pictures the feed tank, the feed and reflux pumps, the flowrate transmitters, the control valves, and the SCR (Silicon Rectifier) heating units. Figure 4.3 shows the schematic flow diagram of the distillation unit. Components used for the distillation process are methanol (reagent grade ACS USP/NF) and water (from municipal utility water pipelines).

The column consists of 6 sieve trays, each assembled within a 0.13 m. (5 in.) long, 0.076 m. (3 in.) I.D. Pyrex glass pipe sections. Each section contains process fitting for feed, liquid/vapor sampling, and weir downcomer adjustment. The column when

assembled is approximately 0.76 m. (30 in.) in height, bolted to the reboiler at its bottom and to the vapor output line at its upper end by means of a bell reducing coupling section and a flexible Teflon expansion joint. The sieve trays are 0.0031 m. (1/8 in.) thick with 36, 0.0038 m. (0.15 in.) holes on each and with weirs and downcomers adjusted for 0.0063 m. (1/4 in.) ideal liquid holdup on each tray (Actual holdup ranges from zero at weeping to 6 in at flooding).

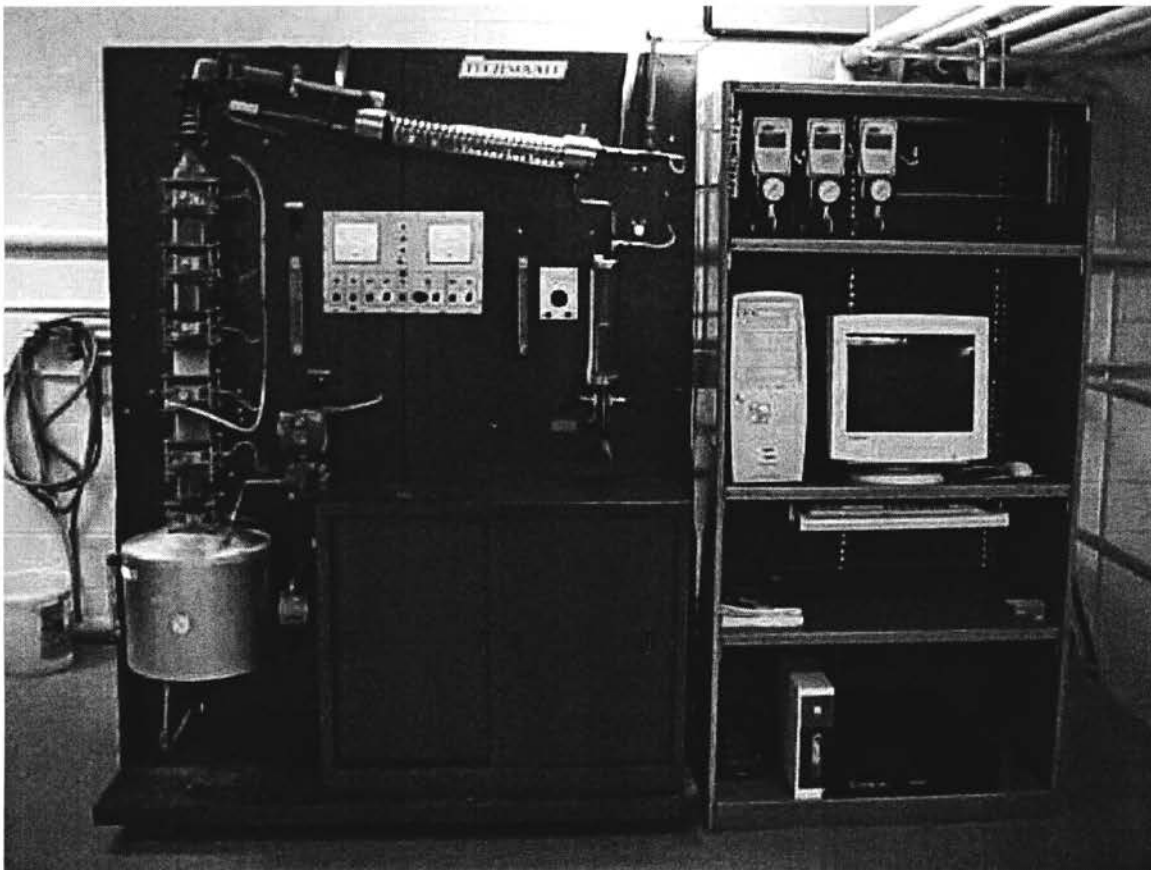


Figure 4.1 Front view of the distillation unit

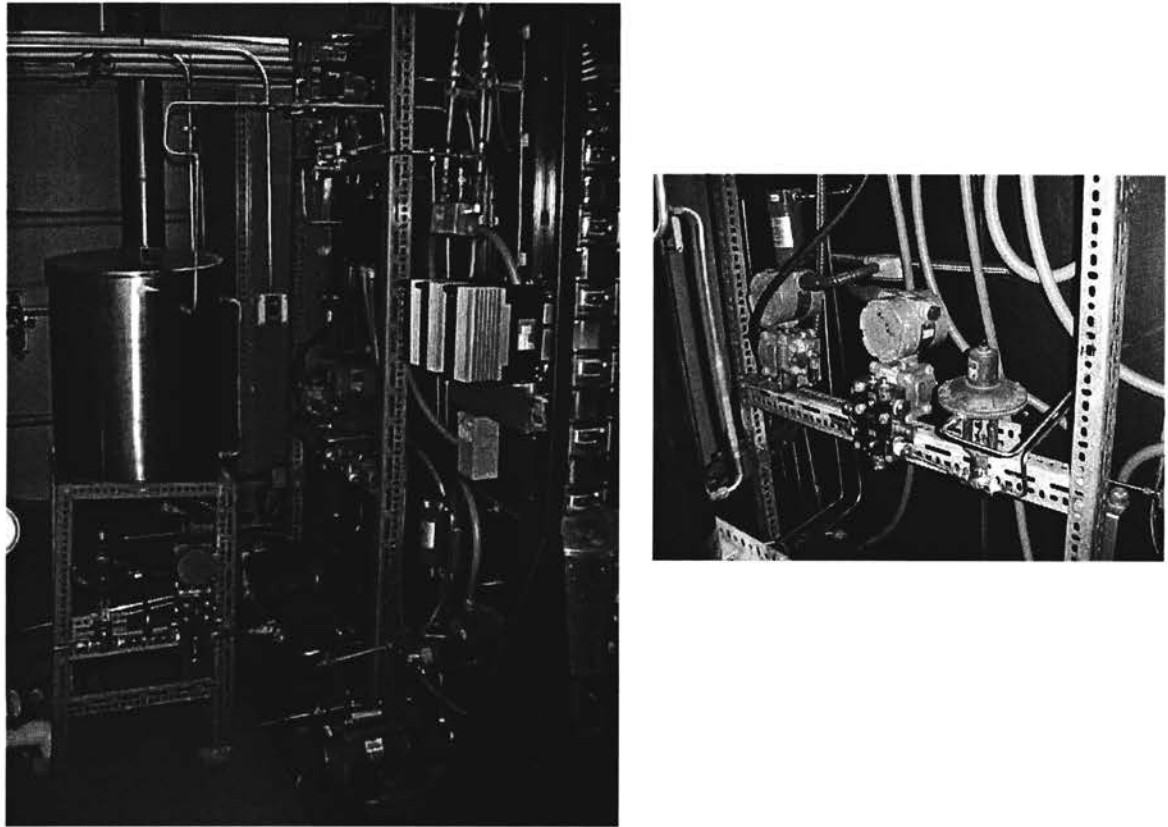


Figure 4.2 Back view of the distillation unit

The reboiler is a cylindrically welded, stainless steel tank with a capacity of approximately 0.015 m^3 (3.86 Gallons). The heating element in the reboiler is a stainless steel sheathed-bayonet-type 3-cartridge heating element with explosion-proof electrical fittings and is rated at 4.0 KW. Calibration of the heating power for the reboiler is described in Appendix A. Liquid level control in the reboiler is obtained by means of a float-type control element that actuates a solenoid-operated valve, which transfers excess liquid (bottom product in the experiments) from the reboiler, through a double pipe heat exchanger (to cool the bottom product), to the bottom (product) tank (a polypropylene container of 5-gallon capacity).

The overhead condenser is a Pyrex and stainless steel shell-and-tube type heat exchanger which contains the equivalent of 0.14 m^2 (1.5 ft^2) of spiral heat exchanger surface. The spiral condensing tube is 3.34 m . (132 in.) long coiled to a length of 0.46 m . (18 in.). The tube has 0.016 m . ($5/8 \text{ in.}$) O.D. and 0.00048 m . (0.019 in.) wall. The shell is a 0.076 m . (3 in.) diameter, 0.6 m . (23.6 in.) long Pyrex glass tube, and is open to the atmosphere at the distillate exit end through a relief valve to keep the column operated at atmosphere pressure.

The reflux drum (distillate receiver) is a 0.076 m . (3 in.) O.D. by 0.304 m . (12 in.) high Pyrex glass tube, which is flanged at the top and bottom. Inside the tube is a 0.0063 m . ($1/4 \text{ in.}$) O.D. tube with a length of 0.152 m . (6 in.) to overflow excess condensed vapor (top product in the experiments) to the top (product) tank (a polypropylene container of 5-gallon capacity).

The feed pump is a centrifugal pump driven by a $1/2$ horsepower DC motor with 3450 RPM speed. The reflux pump is a centrifugal pump driven by a $1/3$ horsepower DC motor with 3450 RPM speed. The feed and reflux supply lines have cartridge type immersion preheaters right before they enter the column rated at 0.25 KW . Transfer lines are stainless steel tubing of 0.016 m . ($5/8 \text{ in.}$) O.D.

The coolant used for the condenser is the water from the municipal water supply pipeline. It is not only used to condense the vapor from the distillation column but also used to cool the bottom product through a double pipe heat exchanger, as illustrated in Figure 4.3.

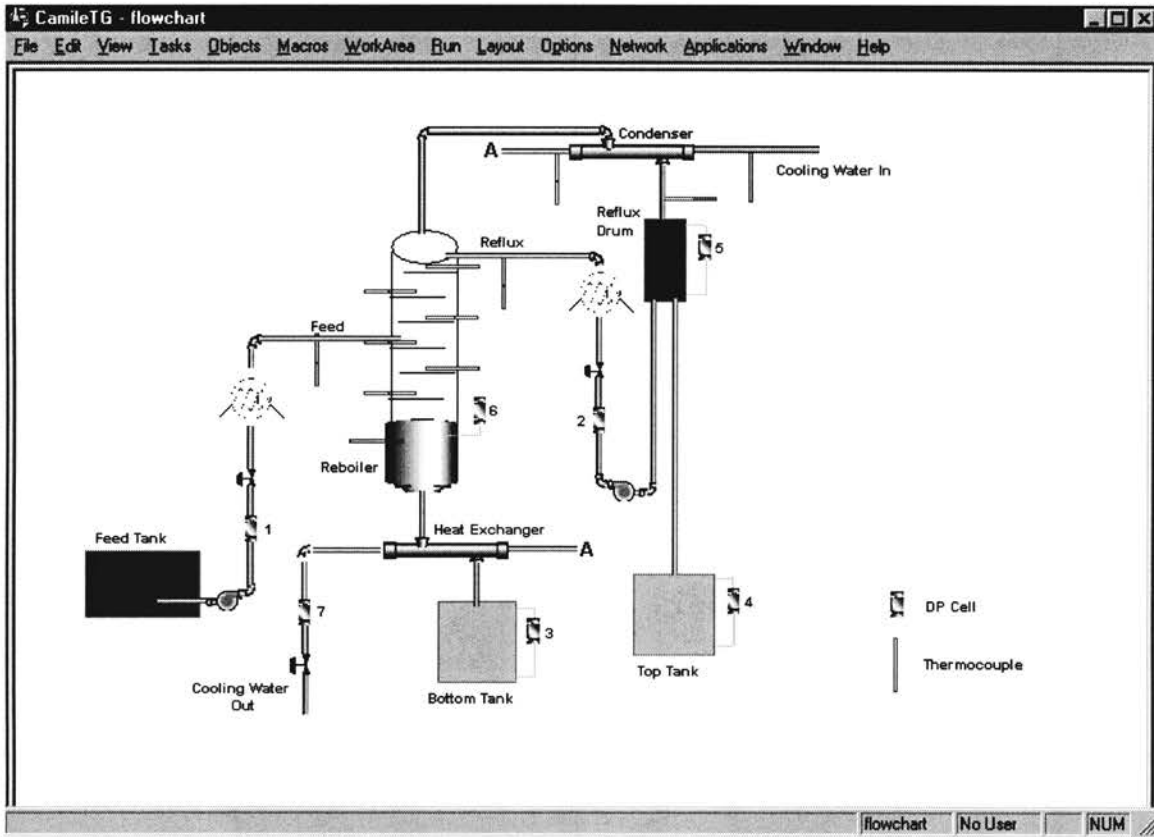


Figure 4.3 Schematic diagram of the distillation process

4.2 Instrumentation

The apparatus is equipped with 12 thermocouples connected at strategic points throughout the system. These include one in the reboiler (immersed in the liquid), six in the column (one for each tray, located about $\frac{1}{4}$ inches over the tray), two on the condenser's inlet and outlet lines (cooling water inlet and outlet temperatures), one for the distillate line (distillate temperature from the outlet of the condenser), one for the feed after the preheater, and one for reflux after its preheater. These thermocouples are of chromel-alumel type and each is epoxy-sealed within a stainless steel tube.

There are seven differential pressure (DP) cells used to signal the feed and reflux flow rates, the levels of the top and bottom product tanks (to monitor overflow of the

product tanks and to inferentially measure the flow rates of the top and bottom products), the cooling water flow rate through the overhead condenser, the liquid level in the reflux drum, and the column pressure (differential pressure between atmosphere and the reboiler's vapor space, which indicates the accumulated liquid level of the column. This is a good index for flooding/entraining and is used as a constraint in the experimental runs). A fractional refractometer is used to provide off-line measurement of the composition of the mixture. Calibrations of the DP cells and the refractometer are described in Appendix A.

4.3 Data Acquisition and Control (DAC) System

The original Technovate[®] unit had manual controls. To apply computer control, the manual controls as shown on the board in Figure 4.1 are bypassed, the back view in Figure 4.2 shows some newly installed devices for full automation such as differential pressure transmitters, control valves, and SCR heating control units. A Camile[®] 2200 DAC system together with a Pentium[®] II (with 333 MHz CPU) PC is equipped with the distillation process for full automation.

There are six computer boards in the Camile[®] 2200 DAC system, five installed inside the Camile[®] 2200 chassis (the black box in Figure 4.1), one (PN566) plugged in the host computer. Control board PN 564 performs control action calculation, scheduling data collection and transmission. Terminator board PN554 minimizes system noise by damping resonant frequencies in the communication bus. Communication board PN565 communicates with the host PC. Thermocouple board PN525 takes thermocouple input signals. Analog input board PN525AI reads standard (4~20mA current in our case)

analog input signals. Analog output board PN522 accepts analog output signals (0~20 mA current in our case) and actuates the control elements (control valves and SCRs). Communication board PN566 is plugged inside the host computer to communicate with other Camile[®] boards. The 12 thermocouple input signals, the readings from the 7 DP cells are read into the computer through the Camile[®] DAC system (each signal is filtered by a first order moving average filter configurable in the Camile[®] interface). There are 6 analog output signals. Three for the feed, reflux and cooling water control valves, and three for the feed, reflux, and reboiler SCRs.

The Camile[®] 4.0.5 software under Windows[®] NT[®] operating system provides user-friendly interfaces for monitoring and controlling the process. Although Camile[®] provides a platform to write customized codes for complicated applications, for our work, we chose to use a more advanced language, the Matlab[®] language in the Matlab[®] application to carry out the proposed GNNMPC scheme. Both Camile[®] and Matlab[®] run in real time within the same PC and communication between the two applications are through an intermediate text file. Camile[®] reads measurement signals from the process, provides interface for monitoring the process, and communicates with the Matlab codes. The Matlab reads data from the Camile[®], makes control decisions, and sends calculation results (control decisions) back to the Camile[®] for its execution. Figure 4.4 shows some human machine interfaces created within the Camile[®] application.

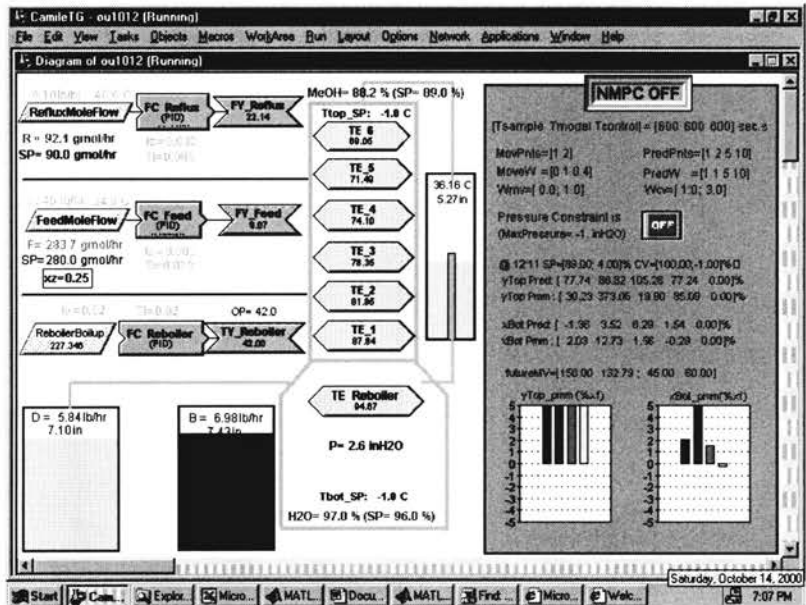


Figure 4.4 (a) Camile interface – flow chart and control parameter values

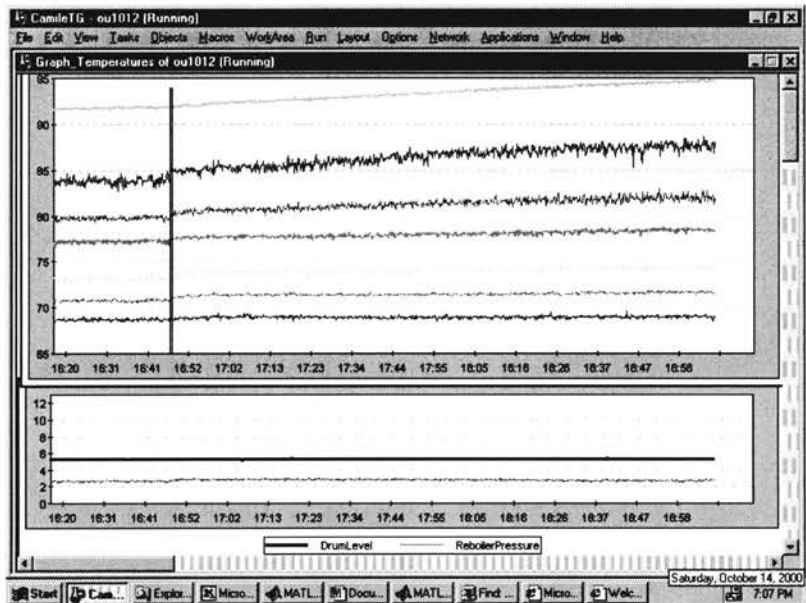


Figure 4.4 (b) Camile interface – graphic records of temperatures, drum level, and column differential pressure

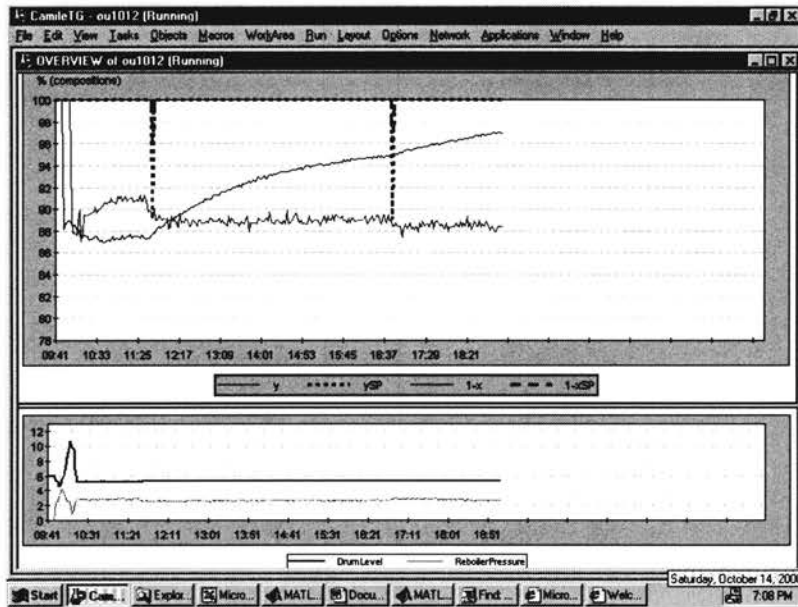


Figure 4.4 (c) Camile interface – graphic records of product compositions, drum level, and column differential pressure

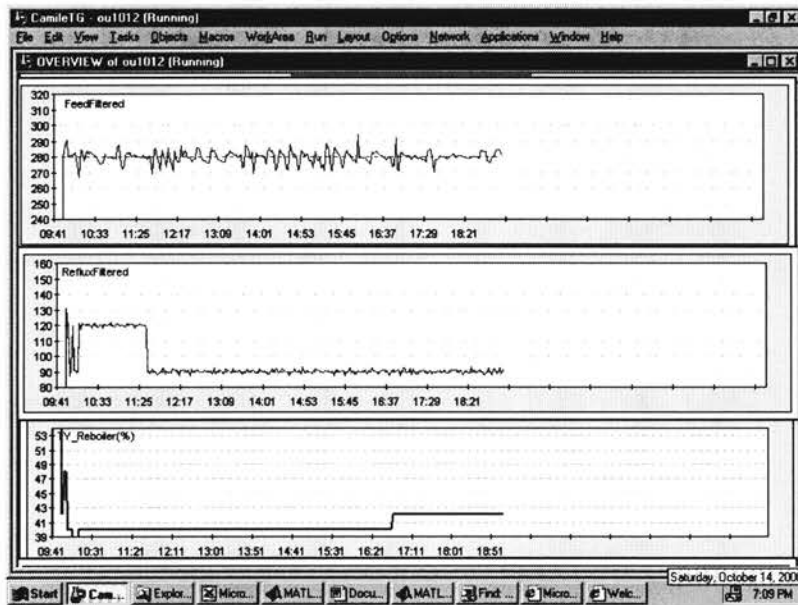


Figure 4.4 (d) Camile interface – graphic records of process inputs, feed flowrate, reflux flowrate, heating power to the reboiler

Figure 4.4 Camile[®] interfaces

4.4 GNNMPC System for the Distillation Column

The GNNMPC control system is a 2*2 MIMO control system. The manipulated variables are the reflux mole flowrate, R , and the heating power to the reboiler as expressed by percentage of the full power, H . The controlled variables are the mole fraction of methanol of the top product, y , which is inferred by temperature on the top tray, and the mole fraction of methanol of the bottom product, x , which is inferred by temperature in the reboiler.

The feed molar flowrate, F , is considered as a measured feedforward load disturbance. The composition of the feed, z , is offline measurable using the refractometer. Temperatures of the feed and reflux, $FeedT$ and $RefluxT$ are measured and controlled by PID controllers. z , $FeedT$, and $RefluxT$ are taken as unknown disturbances for the GNNMPC controller. Under normal operations, z , $FeedT$, and $RefluxT$ are fixed at 0.25 mole fraction methanol, 35 °C, and 50 °C, respectively.

CHAPTER 5

GNNMPC AND THE SIMULATED DISTILLATION

A tray-to-tray dynamic distillation simulator representing a similar experimental setup at Texas Tech University was originally developed by Pandit (1991) . Codes in Basic language (Pandit, 1991) and in C language (Ramchandran, 1994; Dutta, 1997) are available. The model was revised and recoded in Matlab[®] (see codes in Appendix C) to represent the experimental setup at Oklahoma State University. This simulator was used in this work for the following purposes:

- Facilitate better understanding of the dynamic behavior of the experimental distillation process
- Provide insights into the nature of the interactions between the inputs and the outputs
- Carry out preliminary test on the GNNMPC
- Facilitate deep investigation of the GNNMPC

We will describe the modeling method briefly and then discuss implementation of the GNNMPC for the simulated distillation process.

5.1 First-Principles Dynamic Model of Methanol-Water Distillation Column

Material balance, energy balance, plus thermodynamic and hydraulic properties of the components, methanol and water, are used to create the dynamic model of the distillation process. Assumptions made by the simulation are:

- (1) The two components, methanol and water, are pure.

- (2) The column is operated under standard pressure (1 atm). Correlations of thermodynamic properties are obtained from data under standard pressure.
- (3) The dynamics of the heating element in the reboiler is negligible compared to the dynamics of the bottom composition.
- (4) The dynamics of the internal energies on the trays are negligible comparing to the dynamics of the composition, therefore, energy balances on each tray are just algebraic ($\frac{d(M_i h_i)}{dt} = 0$ in Equation (5.3) below).
- (5) The condenser is a total condenser.
- (6) Perfect level control in the reflux drum and the reboiler allows a constant (volume) holdup in the reflux drum and the reboiler by instantaneously changing flowrates of the bottoms product and the liquid distillate product.

The material and energy balance equations for each equilibrium stage are:

The overall material balance:

$$\frac{dM_i}{dt} = L_{i+1} + V_{i-1} - L_i - V_i + F_i \quad (5.1)$$

The component material balance:

$$\frac{dM_i x_i}{dt} = L_{i+1} x_{i+1} + V_{i-1} y_{i-1} - L_i x_i - V_i y_i + F_i z_i \quad (5.2)$$

The energy balance:

$$\frac{d(M_i h_i)}{dt} = L_{i+1} h_{i+1} + V_{i-1} H_{i-1} - L_i h_i - V_i H_i + F_i h_i^F \quad (5.3)$$

In Equations (5.1) to (5.3), i represents an equilibrium stage. We start the numbering of the equilibrium stage from the reboiler (note that the preferred convention of numbering a distillation column starts from the top, this bottom-up convention is

inherited from the coding/modeling legacy in the previous work), which is an equilibrium stage with an efficiency of unity. Then we go up to the bottom tray as stage 2, the top tray is numbered as stage 7. So $i=1\sim 7$. L_i , x_i , and h_i denote mole flow rate, mole composition and enthalpy of the liquid leaving equilibrium stage i (going downwards to equilibrium stage $i-1$). V_i , y_i and H_i denote mole flow rate, composition, and enthalpy of the vapor flow leaving equilibrium stage i (going upwards to equilibrium stage $i+1$). M_i is liquid hold up on stage i . F_i , z_i denote side-stream's mole flow rate and its mole composition entering stage i . For this work, F_i and z_i are feed mole flowrate F and its mole composition z if i equals 4 (feed tray), and it is reflux mole flowrate R and its mole composition x_D if i equals 7 (top tray). For this binary distillation process, component composition refer to the composition of the light component, i.e., methanol.

Correlations of thermodynamic properties of the methanol-water system such as vapor-liquid equilibrium (VLE), enthalpies, densities are given in Appendix B.

A Murphree vapor-phase efficiency is used to describe the departure from equilibrium for calculation of the vapor composition, y_i . The Murphree efficiency on each tray (equilibrium stage) is predetermined from equipment manual and operating experience. Then, from the definition of the Murphree tray efficiency,

$$\varepsilon_i = \frac{y_i - y_{i-1}}{y_i^* - y_{i-1}} \quad (5.4)$$

the actual vapor composition, y_i , can be calculated, given the ideal vapor composition, y_i^* , which is in VLE with the liquid composition x_i , and the composition of the vapor from the previous tray, y_{i-1} . For this work, the Murphree efficiency on each tray was

roughly determined according to observations on the bubbling condition on each tray during experiments. Good bubbling on the tray (e.g. ½” to 2” bubbling liquid) was interpreted as good mass transfer and therefore high Murphree efficiency, refer to the codes Appendix C.1 for the values used for the simulator.

The hydraulic dynamic response of each tray is modeled using Francis weir formula (Luyben,1990) to relate the liquid holdup to the liquid flowrate leaving the tray.

The Francis weir formula is given as

$$Q_L = 3.33l_w(h_{ow})^{1.5} \quad (5.5)$$

where Q_L is liquid flowrate over the weir in ft^3/s , l_w is the length of the weir in ft , and h_{ow} the height of the liquid over the weir in ft . Additionally,

$$L_i = \rho_i Q_{L,i} \quad (5.6)$$

$$M_i = \rho_i \frac{\pi}{4} D^2 (h_{ow,i} - h_w) \quad (5.7)$$

where D is the diameter of the column, h_w is the height of the overflow weir.

Combining Equations (5.5) to (5.7), L_i and M_i in the material and energy balance equations (Equation (5.1) to (5.3)) are correlated.

Given the information of thermodynamic and hydraulic properties and using Euler’s method, the ordinary differential equation (ODE) Set (5.1) to (5.3) can be solved.

Procedures are:

Step 1 Initialization. Either initialize all x_i to be the feed composition, z , and all L_i to be the reflux flowrate, R ; calculate holdup M_i from x_i and L_i by the Francis weir formula; set tray efficiencies, or initialize the column to be at a steady state that is already obtained from a previous simulation.

- Step 2 Get the ideal vapor composition, y_i^* , in VLE with x_i from y - x correlation, then get y_i by using the definition expression of Murphree's tray efficiency (Equation (5.4)).
- Step 3 Get the saturated vapor and liquid phase enthalpies H_i and h_i from H - y and h - x correlations.
- Step 4 Get the vapor flowrate, V_i , for the next time increment, from the energy balance equation on each tray (Equation (5.3)).
- Step 5 Get total liquid holdup M_i and component liquid holdup $M_i x_i$ for the next time increment from total mass balance equation (5.1) and component mass balance equation (5.2) using Euler's discretization method.
- Step 6 Update x_i by $x_i = (M_i x_i / M_i)$.
- Step 7 Update L_i by the Francis weir formula (Equations (5.5) to (5.7)).
- Step 8 Go back to Step 2 if the simulation time is not reached.

The correlations of the thermodynamic properties of methanol-water system is attached in Appendix B. The Matlab[®] codes of the distillation process is attached in Appendix C. In the simulator, the volume of the reboiler is set to be 1.42 liters, instead of 14.2 liters in the experimental setup to speed up the simulation.

5.2 Operating Range

For the simulation, the feed composition is normally at 0.25 mole fraction. The feed temperature is 35 °C and the reflux temperature 50 °C. the operating range are listed in Table 5.1.

5.3 Open-Loop Responses

Open-loop response investigations were carried out to provide insights into the characteristics of the process. Characteristics such as nonlinearity, interaction, deadtime, ill-dynamics can raise control difficulties and should be closely investigated and identified.

For a distillation column with medium product purity, the characteristics that raise control difficulties are usually the interactions and the nonlinearity over a wide operating range. In the following, step responses both in a narrow operating range (locally) and in a wide operating range (globally) were carried out for investigation of the process's characteristics.

Table 5.1 Operating range and nominal point of the simulated distillation process

	RANGE	NOMINAL
Feed flowrate (F) (gmol/hr)	250-310	280
Reflux flowrate (R) (gmol/hr)	90-150	120
Heating power (H) (% full power)	45-60	50
Top product composition (y) (mole fraction of methanol)	0.6-0.95	0.8712
Bottom product composition (x) (mole fraction of methanol)	0-0.2	0.0528

“Local” step responses are carried out around the nominal operating point of the simulator ($\{R,H,F\}=\{120,50,280\}$), with positive and negative deviations of the process inputs (MVs and LDs) from their nominal values. The responses are shown in Figure 5.1 to 5.3. Figure 5.1 shows the process response to a change in the reflux mole flowrate, R ,

(a 10% of both positive (to 132 gmol/hr) and negative (to 108 gmol/hr) deviation from the nominal point) while keeping all other process inputs unchanged. Figure 5.2 shows the process response to a change in the heating power, H , (a 10% of both positive (to 55%) and negative (to 45%) deviation from the nominal point) while keeping all other process inputs unchanged. Figure 5.3 shows the process response to a change in the feed mole flowrate, F , (a 10% of both positive (to 308 gmol/hr) and negative (to 252 gmol/hr) deviation from the nominal point) while keeping all other process inputs unchanged. Table 5.2 (a) lists the steady state deviation of the process outputs, top composition, y and bottom composition, x , from their nominal points under deviated process inputs, Table 5.2(b) presents the local gains calculated from the steady state values in Table 5.2(a). In Table 5.2(b), local static gain, is calculated as $K_{i,o} = \frac{\Delta o}{\Delta i}$, where “ Δo ” represents the steady state change of the process output (y or x), and “ Δi ” represents the steady state change of the process input (R , H , or F).

Table 5.2 Steady state analysis in the local open-loop study

(a) Steady state values of inputs and outputs

	Deviation	y (0.871 mf)	x (0.053 mf)
Reflux flowrate (120 gmol/hr)	+10%	+0.0232 (+2.66%)	+0.0299 (+56.6%)
	-10%	-0.0372 (-4.27%)	-0.0221 (-41.9%)
Heating power (50% full power)	+10%	-0.0686 (-7.87%)	-0.0370 (-70.08%)
	-10%	+0.0305 (+3.50%)	+0.0614 (+116.28%)
Feed flowrate (280 gmol/hr)	+10%	+0.0097 (+1.1%)	+0.0237 (+44.9%)
	-10%	-0.0193 (-2.2%)	-0.0232 (-43.9%)

(b) Local static gains at $\{R,H,F\}=\{120,50, 280\}$

	Deviation				
Reflux flowrate (120 gmol/hr)	+10%	$K_{R,y}$	1.93e-03	$K_{R,x}$	2.49e-03
	-10%		3.10e-03		1.84e-03
Heating power (50% full power)	+10%	$K_{H,y}$	-1.37e-02	$K_{H,x}$	-7.40e-03
	-10%		-6.10e-03		-1.23e-02
Feed flowrate (280 gmol/hr)	+10%	$K_{F,y}$	3.46e-04	$K_{F,x}$	8.46e-04
	-10%		6.89e-04		8.29e-04

As seen in Table 5.2 (b), local gains calculated from positive and negative deviation are different (as much as about a ratio of 2 for $K_{H,y}$ and $K_{H,x}$), indicating static nonlinearity.

Graphs in Figure 5.1 to 5.3 show that the process is of first-orderish dynamics, with essentially the same response time for the top composition and the bottom composition. These graphs also visually show moderate static nonlinearity and slight dynamic nonlinearity. If the step responses under the positive and negative input deviations are symmetric to the steady state response at the nominal point, the process is locally linear. Otherwise, there is either steady state nonlinearity or/and dynamic nonlinearity. In Figure 5.1, it is observed that the settling times are about the same. For the top composition, the change due to the positive deviation is smaller than the change due to the negative deviation, indicating static nonlinearity. This is more obvious from the numeric comparison in Table 5.2, which shows that the same amount of change in the process input does not lead to the same amount of change in the output. The same analyses are done for deviation of the heating power and the feed flowrate. In Figures 5.2 and 5.3, the settling times are essentially the same, and static nonlinearity is observable from the graphs and from numerical values in Table 5.2.

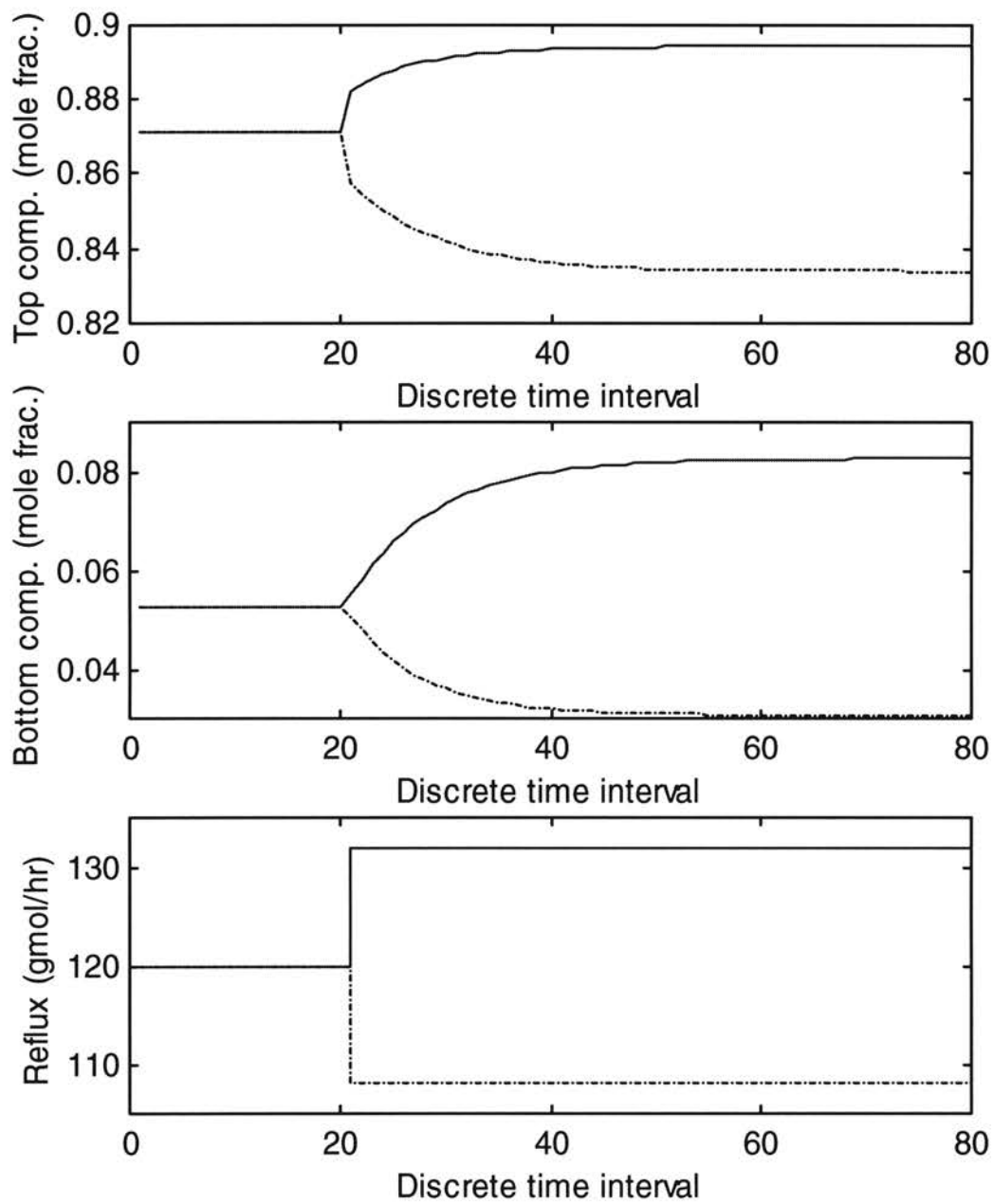


Figure 5.1 Open-loop response to step change of reflux flowrate
(solid line: +10% change; dash-dot line: -10% change)

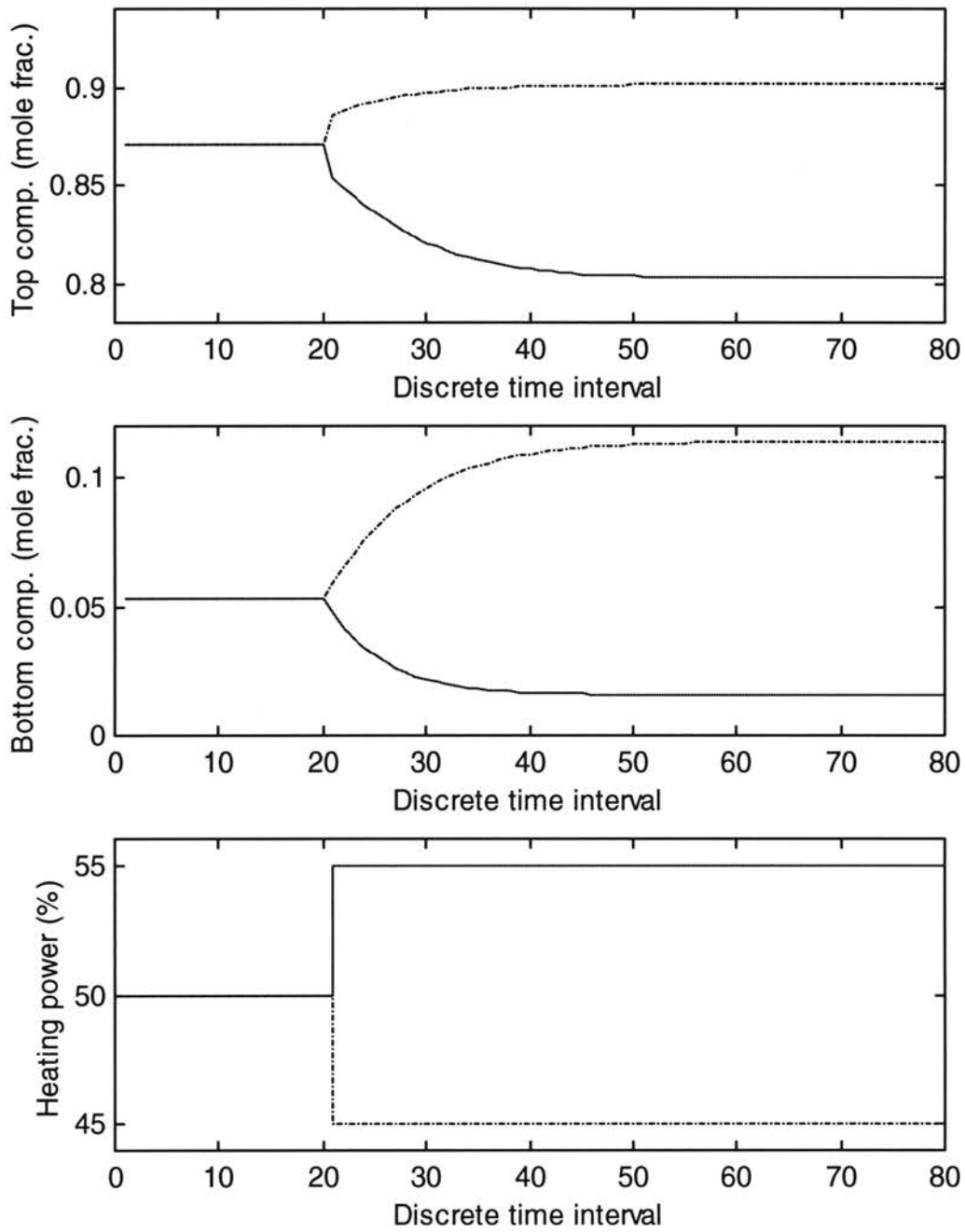


Figure 5.2 Open-loop response to step change of heating power
(solid line: +10% change; dash-dot line: -10% change)

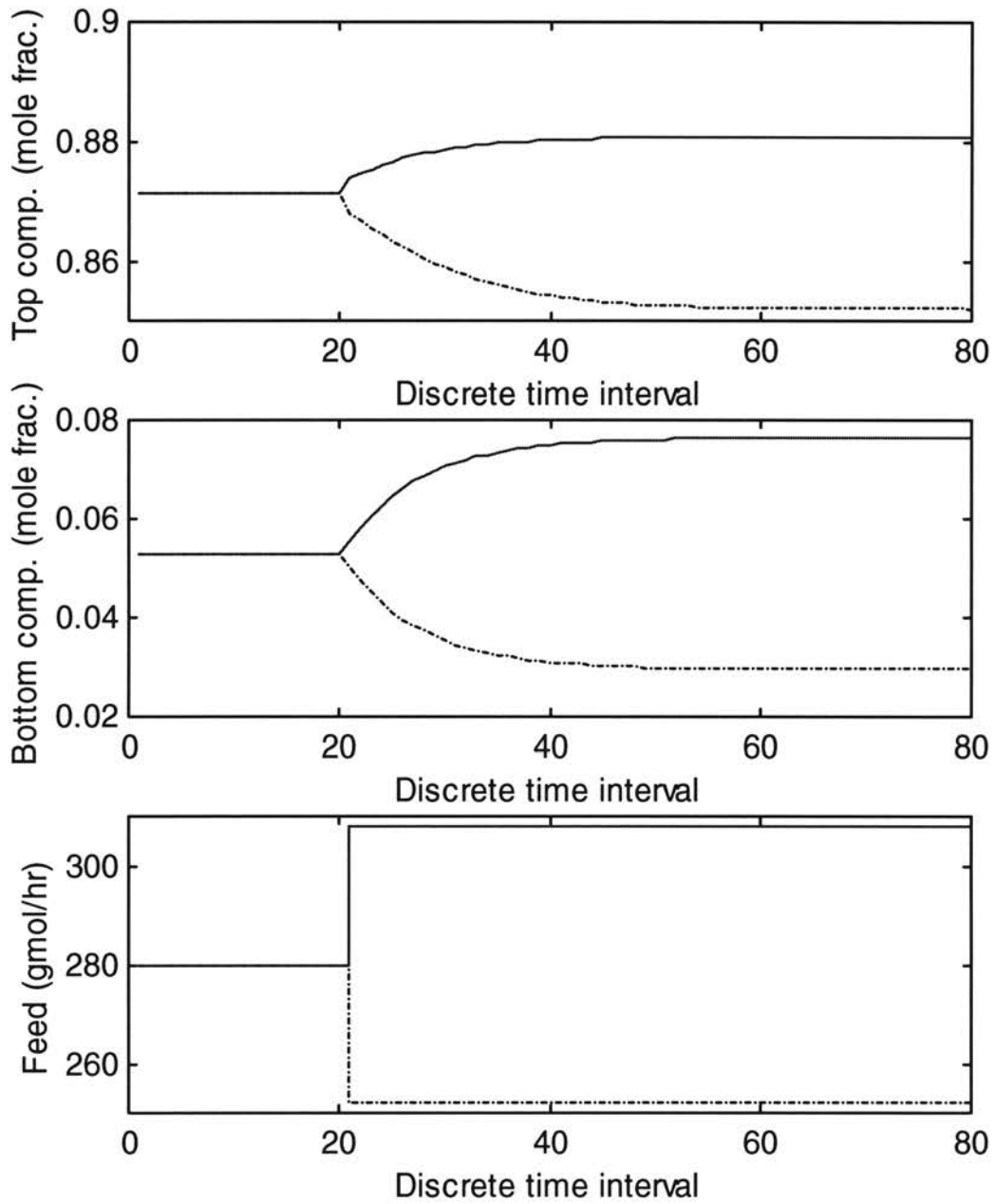


Figure 5.3 Open-loop response to step change of feed flowrate
(solid line: +10% change; dashdot line: -10% change)

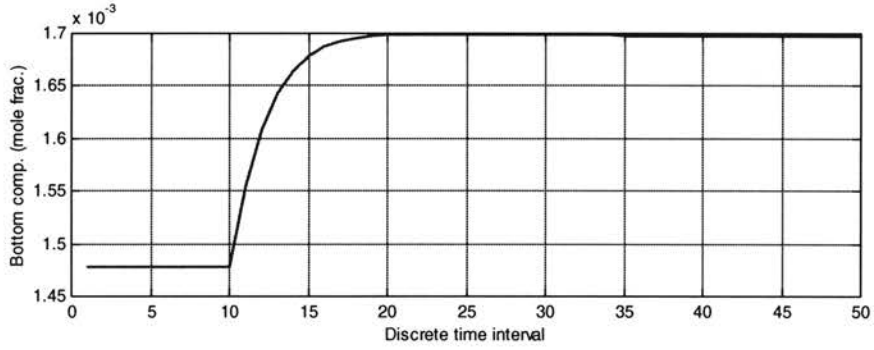
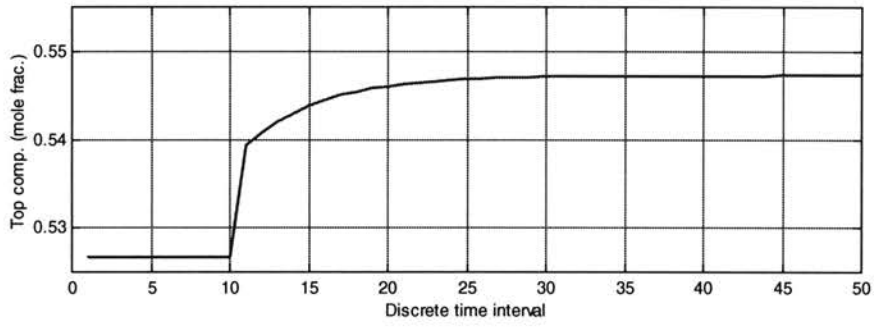
Next, the “global” characteristics were studied over a wide operating range.

Figure 5.4 shows the step responses at one corner of the operating range (Region 1), where the feed flowrate and the reflux flowrate are low and the heating power is high. Figure 5.5 shows the step responses around the nominal point (Region 2), and Figure 5.6 shows the step responses at another corner of the operating range (Region 3), where the feed flowrate and the reflux flowrate are high and the heating power is low. Region 1 and Region 3 can be considered as the two ends of the diagonal line of a cube defined by the operating range of the process inputs (the reflux flowrate, the feed flowrate, and the heating power), while Region 2 is the middle point on the diagonal line.

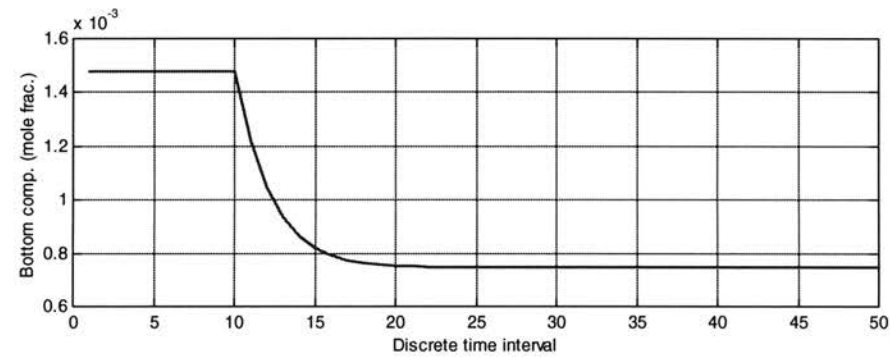
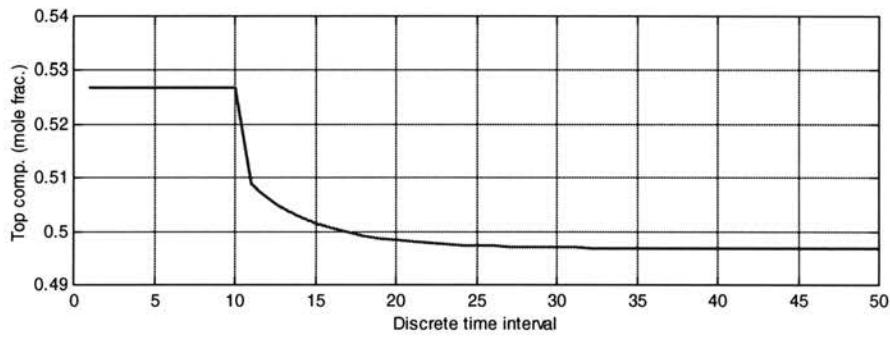
Because the step change is small enough, local gains can be calculated numerically from the steady state values of the inputs and output as, $K_{i,o} = \frac{\Delta o}{\Delta i}$, where “ Δo ” represents the steady state change of the process output, and “ Δi ” represents the steady state change of the process input. As an example, $K_{R,y}$ in Table 5.3 denotes gain of the top composition with respect to the reflux flowrate. Further, Bristol’s relative gain (Seborg, Edgar & Mellichamp, 1989), λ , can be calculated from the local gains as:

$$\lambda = \frac{1}{1 - \frac{K_{R,x}K_{H,y}}{K_{R,y}K_{H,x}}} \quad (5.8)$$

for this distillation process. The steady state values, the local gains as well as the relative gains are listed in Table 5.3. Also listed is the estimated settling time ($t_{i,o}$) from observation of the step responses.

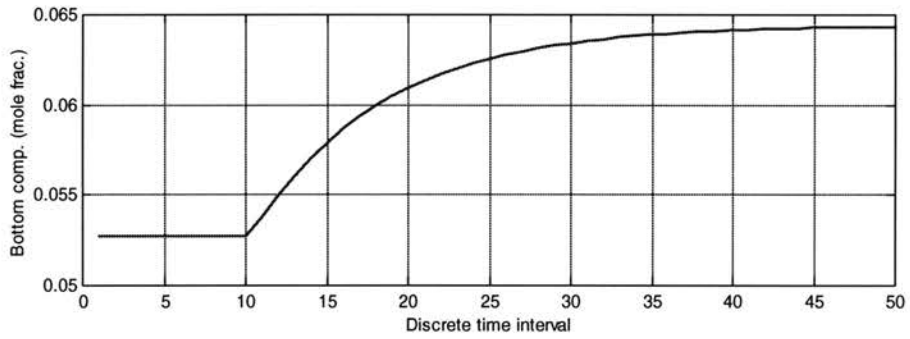
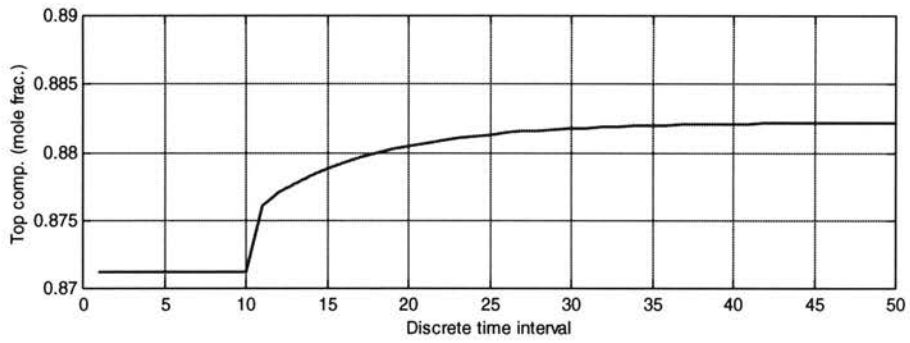


(a) Response to step change of reflux flowrate (+5 mole/hr)

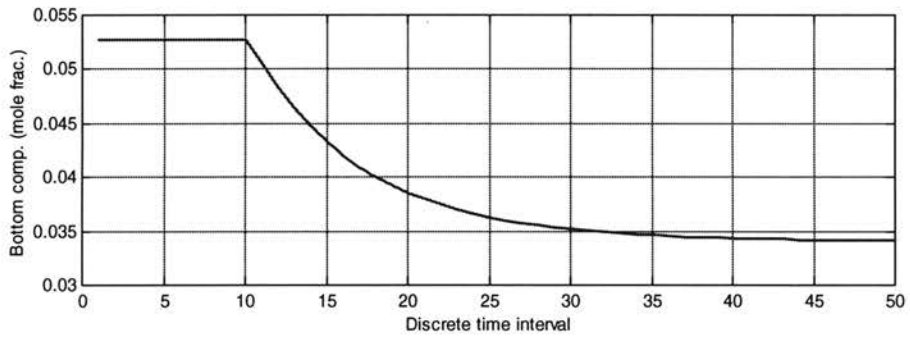
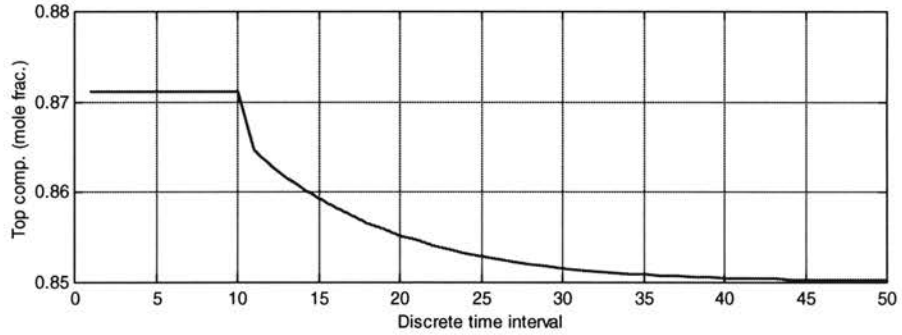


(b) Response to step change of heating power (+2 % full power)

Figure 5.4 Step responses at the neighborhood of $\{R, H, F\} = \{90, 58, 250\}$

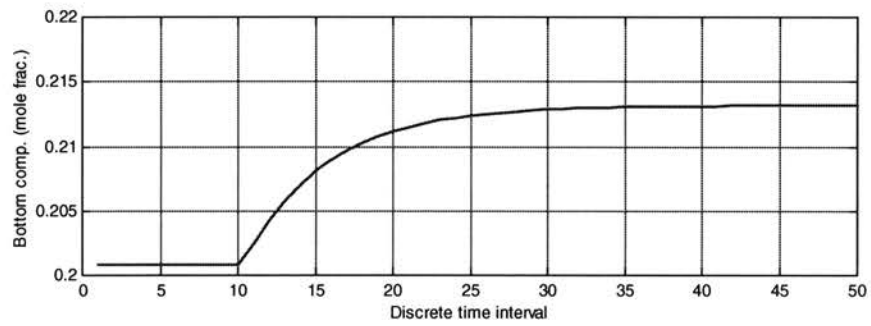
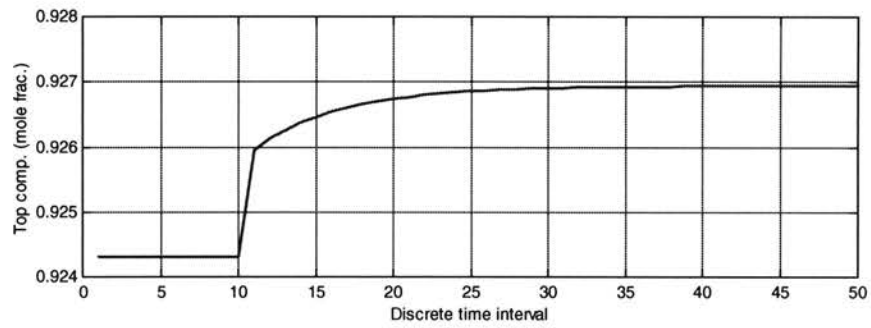


(a) Response to step change of reflux flowrate (+5 mole/hr)

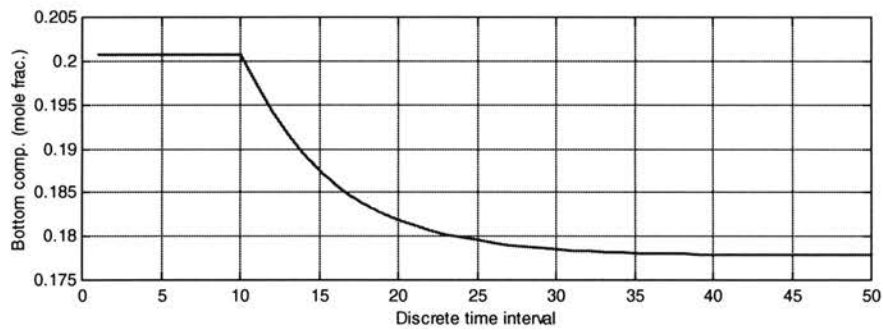
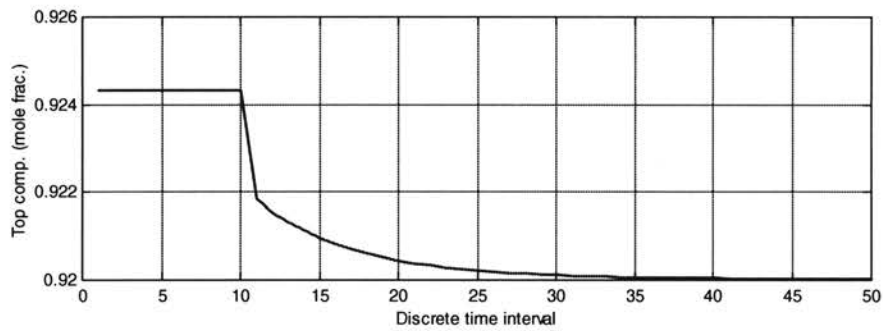


(b) Response to step change of heating power (+2 % full power)

Figure 5.5 Step responses at the neighborhood of $\{R, H, F\} = \{120, 50, 280\}$



(a) Response to step change of reflux flowrate (+5 mole/hr)



(b) Response to step change of heating power (+2 % full power)

Figure 5.6 Step responses at the neighborhood of $\{R, H, F\} = \{145, 45, 310\}$

Table 5.3 shows that local gains can change magnitude by a factor of over 5 times for the top composition and over 10 times for the bottom composition from one region to another region, denoting significant nonlinearity in the wide operating range. The relative gain changes from 8.16 in Region 3 to -5.35 in Region 2, indicating not only significant interactions of the process but also drastic change of the interaction effects at different regions. There is also dynamic nonlinearity as shown by the change of the settling times magnitude of as much as 3 times in different operating regions.

Table 5.3 Steady state analysis in the wide range open-loop study

	Region 1			Region 2			Region 3		
Reflux flowrate (gmol/hr)	90	95	90	120	125	120	145	150	145
Heating power (% full power)	58	60	60	50	50	52	45	45	47
Feed flowrate (gmol/hr)	250	250	250	280	280	280	310	310	310
Top comp. (mole frac.)	.5267	.5473	.4969	.8712	.8822	.8502	.9243	.9269	.9200
Bottom comp. (mole frac.)	.0015	.0017	.0007	.0528	.0643	.0342	.2008	.2132	.1778
$K_{R,y}$	4.12e-03			2.20e-03			5.30e-4		
$K_{H,y}$	-1.49e-02			1.05e-02			-2.14e-3		
$K_{R,x}$	4.40e-05			2.32e-03			2.48e-3		
$K_{H,x}$	-3.65e-04			-9.30e-03			-1.15e-2		
λ	1.77			-5.35			8.16		
$t_{R,y}$	20			30			25		
$t_{H,y}$	15			30			25		
$t_{R,x}$	10			35			25		
$t_{H,x}$	10			30			25		

5.4 GNN Modeling

For the control system, the MVs are the reflux mole flowrate and the heating power to the reboiler, the feed flow rate is taken as a slowly changing measured feedforward load disturbance, the feed composition is taken as an unmeasured load disturbance. The CVs are the top composition and the bottom composition. Parameters chosen for the GNN modeling are shown in Table 5.4.

Table 5.4 Parameters for GNN modeling
(sample interval is 2 minutes, numbers are with respect to discrete sample interval)

Past Window Length (q)	5
Prediction Points (Y_p)	1,2,3,5,10

The model structure of each NN in GNN is described as

$$Y(k+i) = f(Y(k-q), \dots, Y(k), U(k-q), \dots, U(k-1), F(k), U(k), \dots, U(k+i-1)), i \in Y_p \quad (5.9)$$

Where Y is the vector of CVs, U is the vector of MVs, and F is the feed flowrate, the feedforward load disturbance. Note that for this distillation process, the feed flowrate is taken as a slowly changing measurable disturbance, only the “current” feed flowrate is taken as the GNN’s input.

Each NN in the GNN model is a 3 layer FeedForward Neural Network (FFNN) with sigmoidal node transfer function. the inputs and outputs are determined by the model structure as shown in (5.9). The number of hidden units for each NN, which

determine the approximation performance of the NN, is determined by trial-and-error, a common practice in NN modeling as will be described in detail later in this section.

The input-output series that are used to construct the training data set for the GNN are obtained by open-loop responses to randomly distributed input series $\{Reflux, Heating\ power, Feed\}$. The magnitudes and frequencies (distribution) of the input series must be such designed that the dynamics of the process is sufficiently stimulated and fully developed. For this work, the magnitudes of all the process inputs are uniformly distributed in the operating range. For the MVs, the time period that one magnitude lasts is a random integer that is uniformly distributed between 15 to 60 discrete time intervals (30 to 120 minutes). For the feed flowrate (the feedforward disturbance), the time period that one magnitude lasts is a random integer that is uniformly distributed between 30 to 80 discrete time intervals (60 to 160 minutes). The principles in choosing the above design parameters are that there are enough points representing sufficiently developed dynamics as well as enough points representing steady state values. For the simulated distillation process, the response time of the process is about 30 discrete time intervals, thus change at time interval 15 would provide dynamic stimulation and change at time interval 60 would allow the process to develop enough steady state points. Because the feed flowrate is treated as a slowly changing feedforward load disturbance, the change frequency is low. The minimum lasting period is set to be about the settling time of the process (30 discrete time interval). The generated input-output series is shown in Figure 5.7, where about 4500 samplings are available. Figure 5.8 is an exploded view of a small section of Figure 5.7.

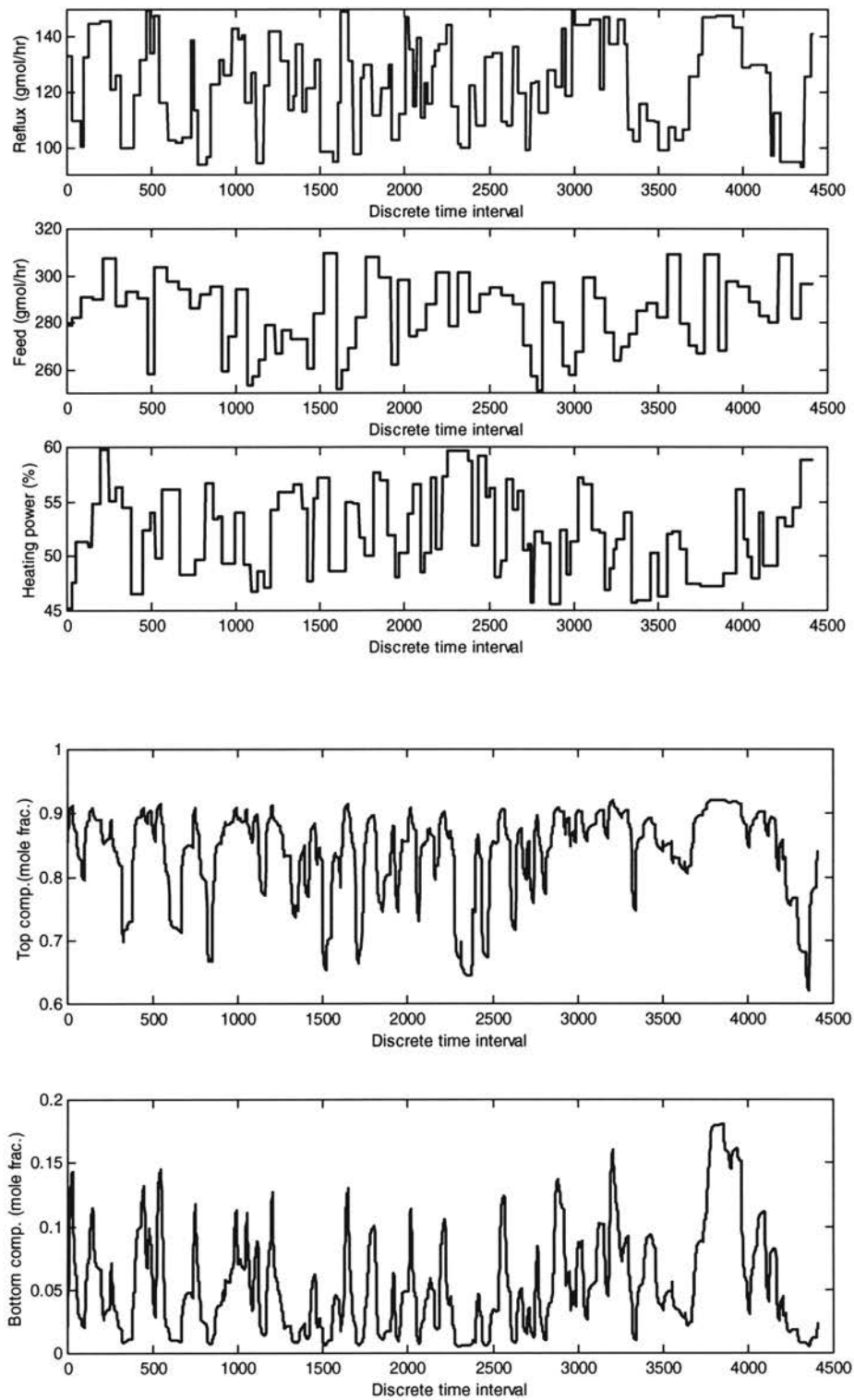


Figure 5.7 Input-output series for training the GNN

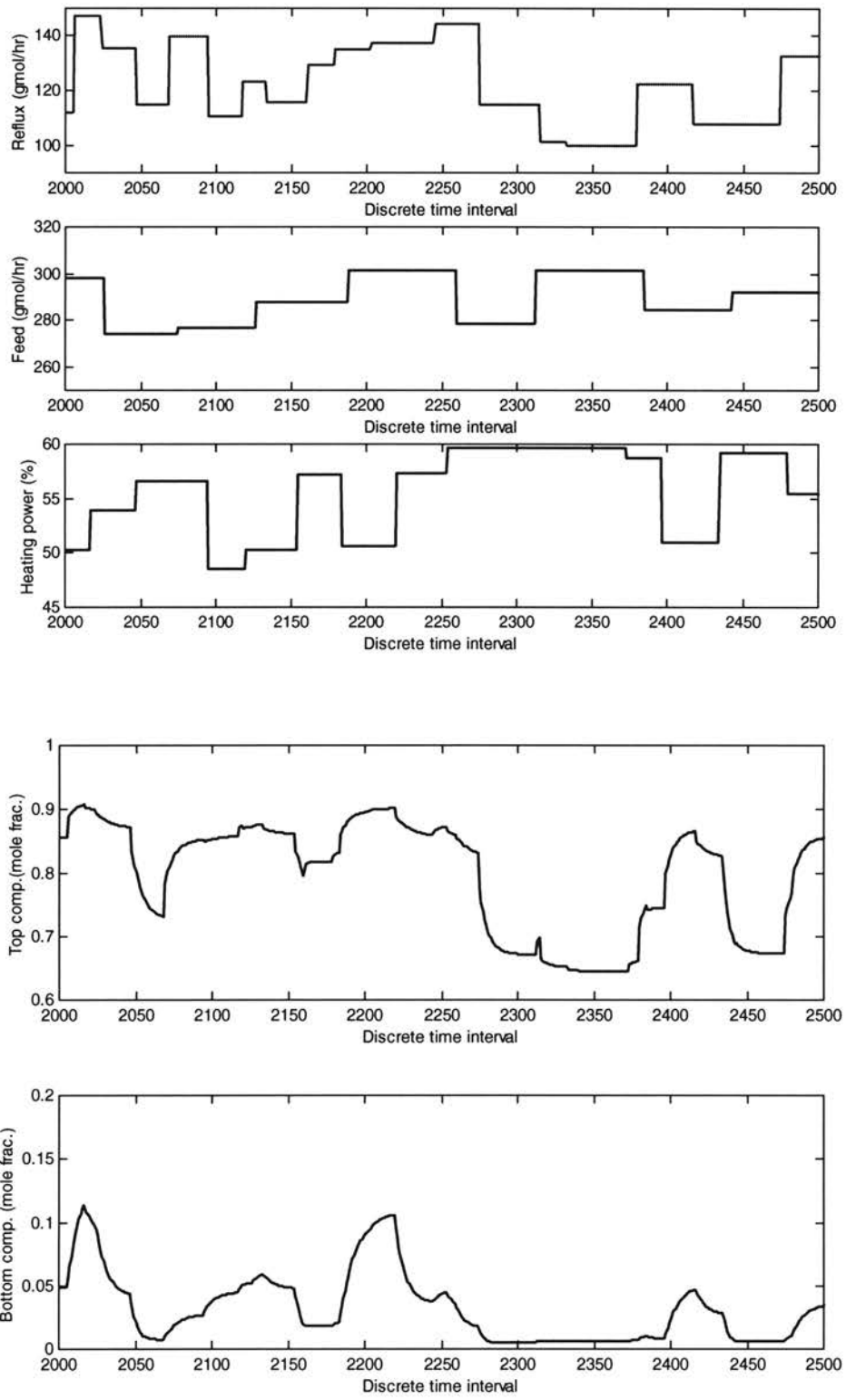


Figure 5.8 Input-output series (explosion of Figure 5.7)

As shown in Figures 5.7 and 5.8, both dynamic points and steady state points are generated. The values in the input-output series are then scaled (linearly compressed to a range of 0 to 1 based on the operating range of each variables as defined in Table 5.1) and constructed to produce 4396 training patterns (NN's inputs and target outputs) for the NNs. Among the patterns, the first 4000 patterns are used as the training set, and the remaining 396 patterns compose the testing set.

The trial-and-error training procedure is as follows, the number of hidden units and a maximum training epochs are set, then the NN is trained till the maximum training epochs (100 as default in the Matlab toolbox) is reached. The performance (Mean Square Error (MSE)) of both the training set and the testing set is recorded. When the two criteria that the MSE of both the training set and the testing set is less than 10^{-3} and that the MSE of the testing set is not increasing are met, the training is considered to be of good performance and adopted. Otherwise, the number of hidden units is increased (if the MSE of the training set is greater than 10^{-3}) or decreased (if the MSE of the testing set increases, indicating overfitting of the training patterns) and the training process is carried out again. Graphical results of comparing the NN output and the target output are also used as guidelines for determining the “goodness” of the NN training. The threshold of 10^{-3} for MSE is a subjective choice based on the justification that 10^{-3} responses to an average relative error of about 3% (the average error is the square root of 10^{-3} , which is about 0.03, since the range is 0 to 1, this indicates an average relative error of about 3%).

Training results of the GNN are shown in Table 5.5. Three criteria are used to test the training performance. The Root Mean Square Error (RMSE), the Maximum Absolute Error (MAE), and the Maximum Relative Error (MRE). Each NN outputs two values, the

scaled top composition and the scaled bottom composition at a future moment. RMSE of a magnitude of less than 0.02 indicates a mean modeling error of 2%. The values of the MAE and MRE do not look good. The MRE for the scaled bottom composition can be as high as 99.03% and that for the scaled top composition can be as high as 121.59%. It was observed, however, from the time series comparison (as partly shown in Figure 5.9) that the MAE (MRE) occurred at points where the initial state change occurred. The trend that the NNs learned tracked the process very well. Therefore, it is nevertheless claimed that the GNN has good approximating (modeling) performance.

Table 5.5 Training results of the GNN model

(RMSE: Root Mean Square Error; MAE: Maximum Absolute Error; MRE: Maximum Relative Error.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2}; MAE = \max(|\hat{y}_i - y_i|, i = 1, \dots, N); MRE = \max\left(\left|\frac{\hat{y}_i - y_i}{y_i}\right|, i = 1, \dots, N\right)$$

Where \hat{y}_i is the output from the NN, y_i is the target output, and N is the number of patterns)

		NN ₁	NN ₂	NN ₃	NN ₅	NN ₁₀
NN Structure		25-6-2	27-6-2	29-6-2	35-6-2	43-6-2
RMSE of training set		0.0110	0.0113	0.0118	0.0130	0.0173
RMSE of testing set		0.0117	0.0120	0.0128	0.0140	0.0206
Scaled top composition	MAE (training set)	0.1929	0.1918	0.1930	0.1903	0.1941
	MAE (testing set)	0.2257	0.2176	0.2192	0.2042	0.1735
	MRE (training set)	48.05%	47.20%	46.17%	46.29%	45.25%
	MRE (testing set)	80.50%	77.59%	78.15%	76.86%	99.03%
Scaled bottom composition	MAE (training set)	0.0598	0.0585	0.0667	0.1136	0.1285
	MAE (testing set)	0.0424	0.0408	0.0508	0.0492	0.0618
	MRE (training set)	30.17%	35.16%	47.96%	86.57%	121.59%
	MRE (testing set)	15.87%	22.32%	32.62%	54.60%	76.46%

Figure 5.9 compares the NN output with the target output using the testing set for $Y(k+3)$ (scaled top and bottom compositions). The dotted line, which is the output from the NN, mostly overlapped with the solid line, the target output for the NN, indicating that the NN catches the dynamics well. The deviation plot shows that most NN outputs clustered very close to the diagonal line, where the NN output equals the target output, also indicating good training results. The lack of data where the scaled bottom composition is greater than 0.6 as shown in Figure 5.9 (b) implies that data in the region are not available from the generated data.

Another approach of validating the GNN model is to compare the step response of the process with the GNN prediction. To do this, the process was initially at steady state, then a step change was made to both MVs (denoted as time instant '0'), and the GNN made its prediction of the step response. The process was then allowed to developed till it reached beyond the prediction horizon, the actual step response was then compared to the GNN prediction done at time instant '0'. Figure 5.10 shows the comparison for a step response to the step change of the MVs, $\{R, H\}$, from $\{120, 50\}$ to $\{130, 46\}$. Table 5.6 shows the difference between the GNN output and the actual output . Figure 5.11 shows the comparison for a step response to the step change of the process inputs, $\{R, H, F\}$, from $\{120, 50, 280\}$ to $\{90, 55, 260\}$. NN outputs are scaled back to their operating range to be compared with the actual process outputs. Table 5.7 shows the difference between the GNN output and the actual output. From Table 5.6 and Table 5.7, the maximum absolute error of the top composition prediction is 0.0279 mole fraction, which is 7.97% of the top composition's operating range (0.6 to 0.95). The maximum absolute error of the bottom composition prediction is 0.0119 mole fraction, which is 5.95% of the bottom

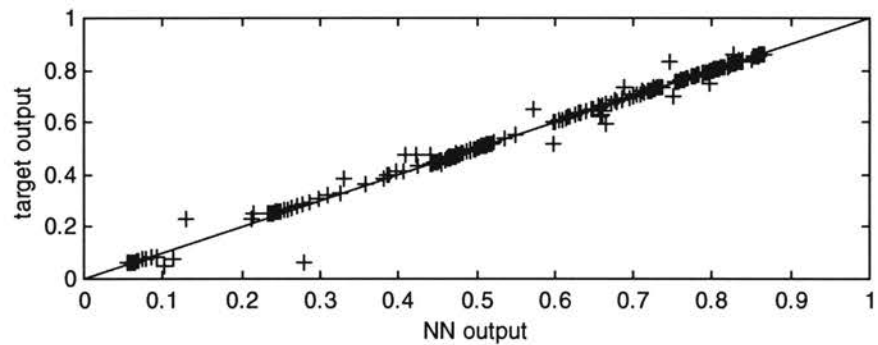
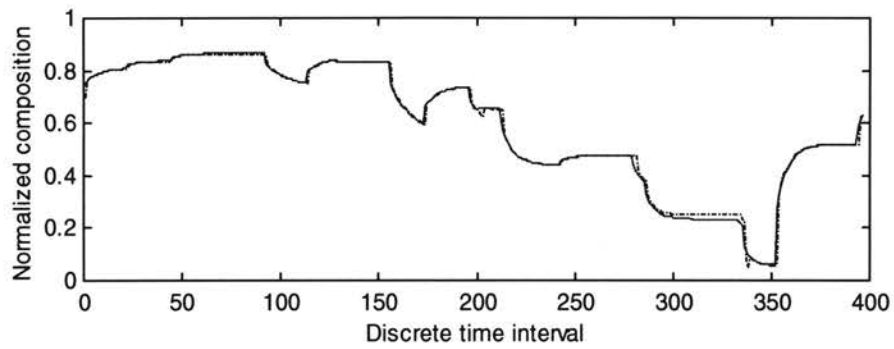
composition's operating range (0 to 0.2). Figures 5.10 and 5.11 also show that the GNN provides a good representation of the process dynamics, because the predicted behavior is relatively close to the actual behavior.

Table 5.6 Predicted step response versus the actual step response
 $\{R,H\}$ changes from $\{120, 50\}$ to $\{130, 46\}$

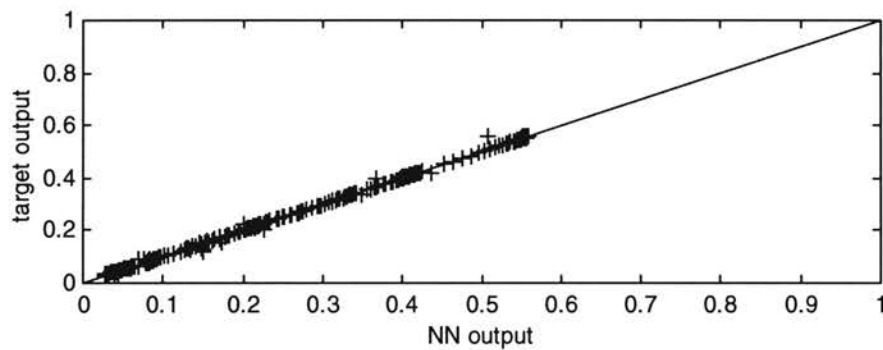
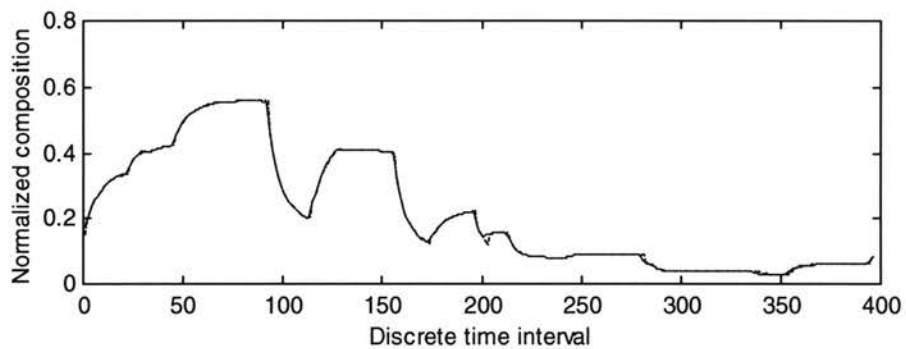
Future points		1	2	3	5	10
Top composition (mole frac.)	Actual	.8908	.8932	.8961	.8995	.9046
	Prediction	.8932	.8951	.8976	.9017	.9058
	Error	.0024	.0019	.0005	.0022	.0012
Bottom composition (mole frac.)	Actual	.0598	.0676	.0746	.0865	.1066
	Prediction	.0670	.0742	.0804	.0904	.1105
	Error	.0072	.0066	.0058	.0039	.0039

Table 5.7 Predicted step response versus the actual step response
 $\{R,H,F\}$ changes from $\{120, 50, 280\}$ to $\{90, 55, 260\}$

Future points		1	2	3	5	10
Top composition (mole frac.)	Actual	.7988	.7696	.7440	.6983	.6395
	Prediction	.7709	.7454	.7252	.6852	.6543
	Error	-.0279	-.0242	-.0188	-.0131	.0184
Bottom composition (mole frac.)	Actual	.0417	.0312	.0232	.0132	.0053
	Prediction	.0298	.0233	.0161	.0079	.0058
	Error	-.0119	-.0079	-.0071	-.0053	.0005



(a) Top composition



(b) Bottom composition

Figure 5.9 Training results for the testing set (performance of NN3)

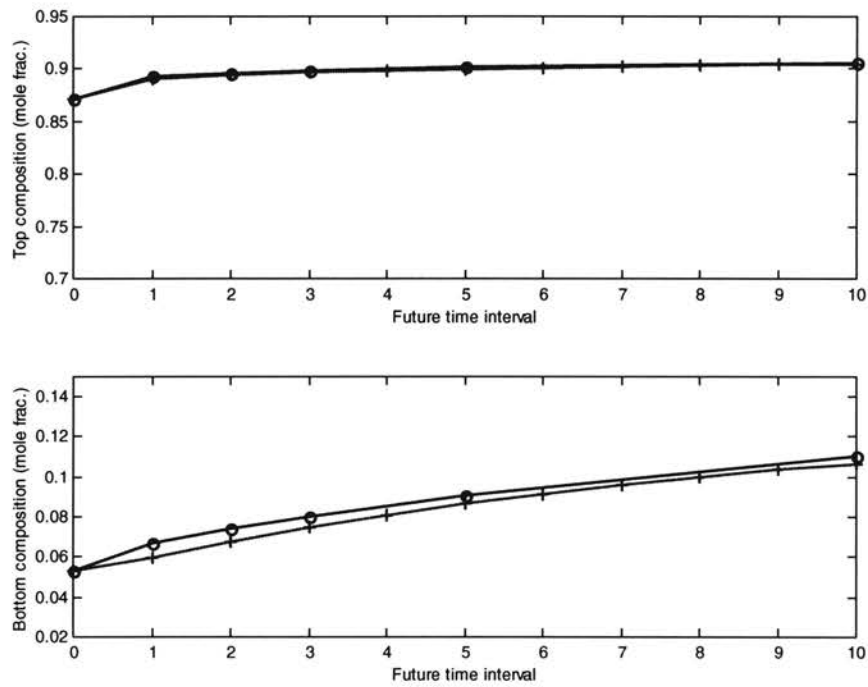


Figure 5.10 Predicted step response versus the actual step response $\{R,H\}$ changes from $\{120, 50\}$ to $\{130, 46\}$, 'o' is prediction, '+' is actual value

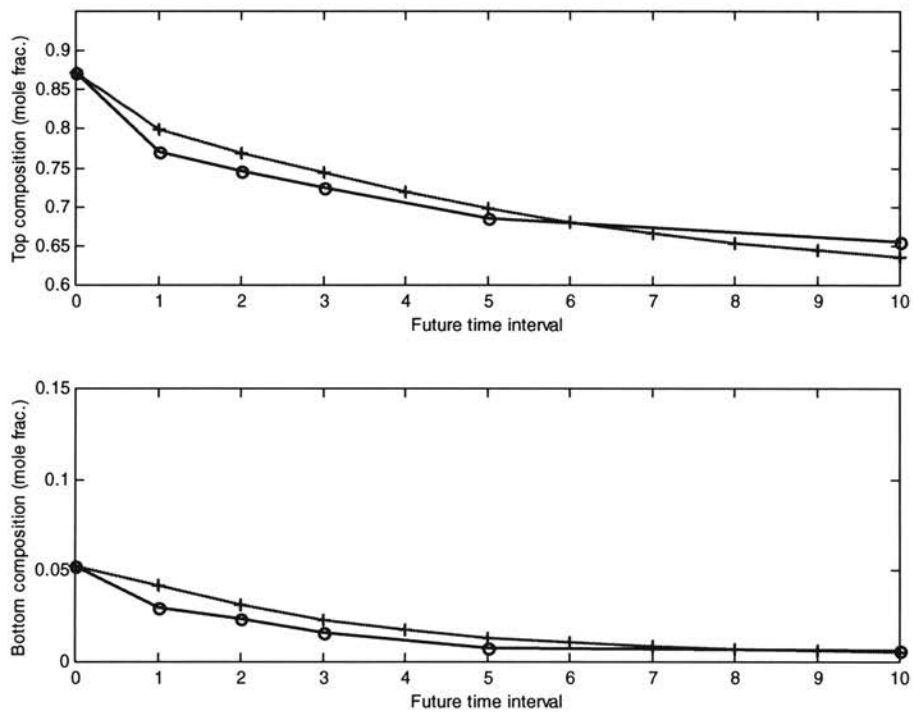


Figure 5.11 Predicted step response versus the actual step response $\{R,H,F\}$ changes from $\{120, 50, 280\}$ to $\{90, 55, 260\}$. 'o' is prediction, '+' is actual value

Figure 5.12 demonstrates the lack of “consistency” of the GNN outputs, which requires individual NN model corrections. The process is at steady state under $\{R,H,F\}=\{120,50,280\}$. The GNN made prediction when there are no changes to the process inputs. Prediction from each NN in the GNN, although produces close numbers, these numbers are not exactly the same, indicating transient state instead of steady state. This discrepancy was the results of the separate training methodology that is used for the GNN modeling, and was the motivation for the proposal of a new Process Model Mismatch (PMM) compensation strategy as described in Chapter 3.

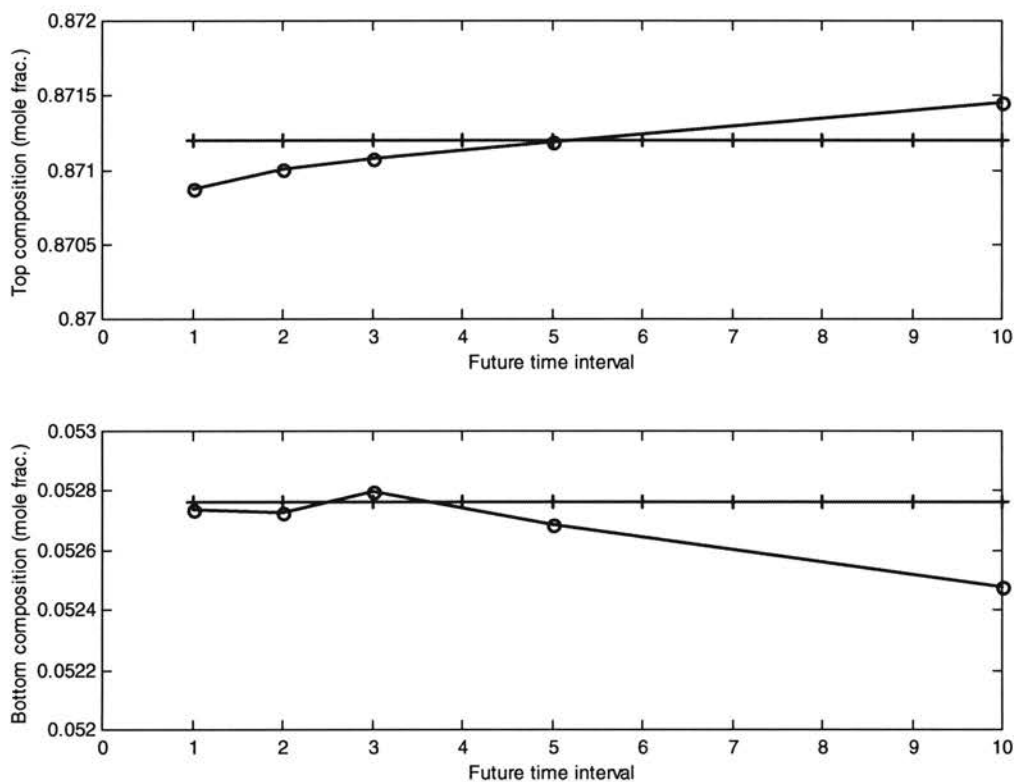


Figure 5.12 Discrepancy of GNN outputs
('o' is prediction, '+' is actual value)

5.5 Process Model Mismatch (PMM)

The NN modeling method is a data-dependent method, for which a large amount of data points are often necessary for the NN to extract the process characteristics efficiently. It is often the case that a dynamic simulator of a real process is available for deep investigation of the process. Because it is much quicker and more convenient and cost efficient to generate data from the simulator, it would be desired that the data used for the NN training is obtained from the simulator rather than from the real process. However, the simulation model is often a simplified one, and often does not fully represent the real process. We call this mismatch between the simulation model and the real process modeling error. In the controller simulation results that follow (Section 5.6), the simulator which generated training data is intentionally different from the simulator which represent the process. The situation where modeling error exists is simulated below. The simulated real process has different tray efficiencies and reboiler correlations from the simulation model used for the NN training. The mismatch between the NN model and the simulated real process is called Process Model Mismatch (PMM).

Figure 5.13 demonstrates the modeling error in the simulation work. Under the same process inputs ($\{R, H, F\} = \{120, 50, 280\}$), the model and the real process reached significantly different steady state values ($\{y, x\}$ of $\{0.8712, 0.0528\}$ versus $\{0.8520, 0.0284\}$). When the process inputs made a step change at time instant 60 to $\{R, H, F\} = \{90, 55, 260\}$, the amount of the output steady state values are also different ($\{y, x\}$ of $\{0.5999, 0.0039\}$ versus $\{0.5537, 0.0021\}$). Figure 5.14 and Figure 5.15 show the effect of *modeling error* on the PMM. Exactly the same step response tests as shown in Figure 5.10 and Figure 5.11 were taken, with the existence of modeling error. Figure 5.14

shows the same tests as done in Figure 5.10, where the process inputs, $\{R, H, F\}$, made a step change from $\{120, 50, 280\}$ to $\{130, 46, 280\}$. Figure 5.15 shows the same tests as done in Figure 5.10, where the process inputs, $\{R, H, F\}$, made a step change from $\{120, 50, 280\}$ to $\{90, 55, 260\}$. It is obvious that with the modeling error, the PMM becomes bigger.

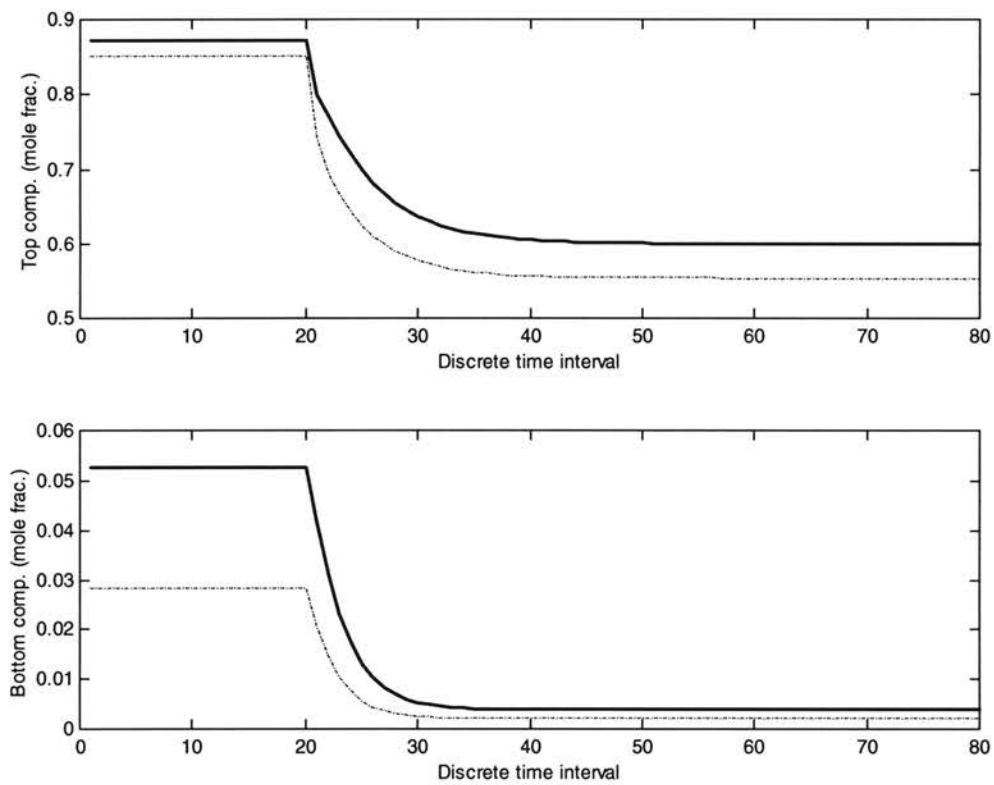


Figure 5.13 Modeling error

Solid line is response of the simulated real process, dash-dot line is response of the simulation model

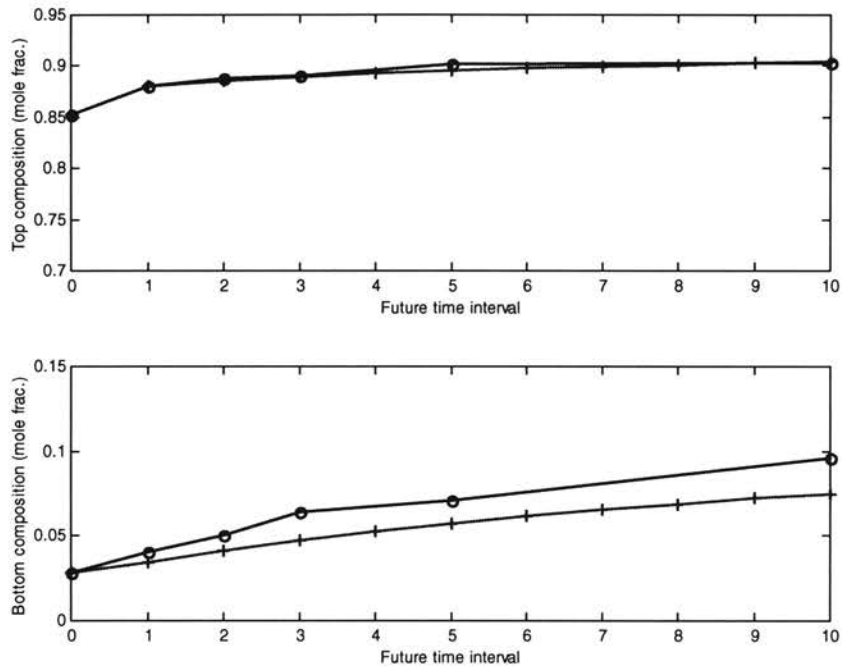


Figure 5.14 Predicted step response versus the actual step response (with modeling error)
 $\{R,H\}$ changes from $\{120, 50\}$ to $\{130, 46\}$. 'o' is prediction, '+' is actual value

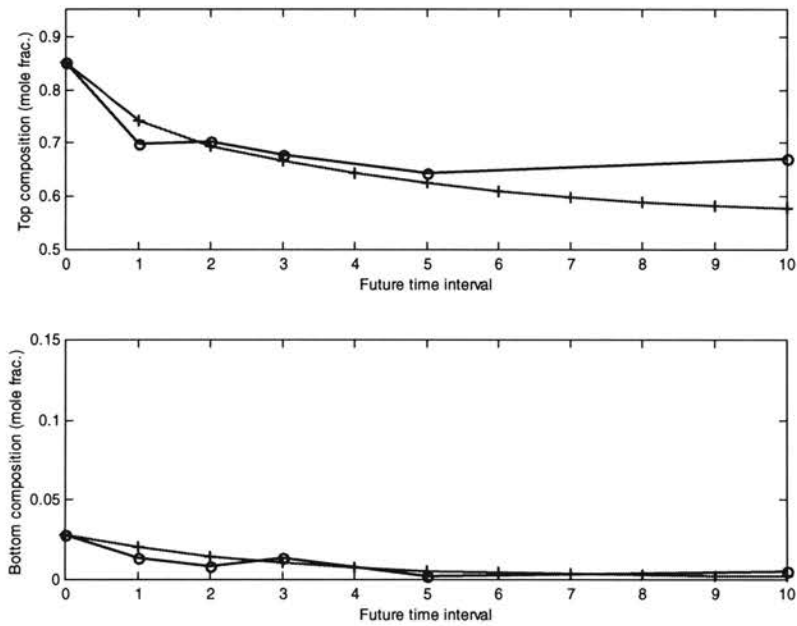


Figure 5.15 Predicted step response versus the actual step response (with modeling error)
 $\{R,H,F\}$ changes from $\{120, 50, 280\}$ to $\{90, 55, 260\}$. 'o' is prediction, '+' is actual value

5.6 Control Performance of GNNMPC

GNNMPC is formulated as a nonlinear constrained optimization problem, which is solved using the Sequential Quadratic Programming (SQP) technique available in the Matlab[®] optimization toolbox (version 1.5.2). The optimization problem is:

$$\underset{R_f, H_f}{Min} = \sum_{i \in y_p} (\tilde{y}(k+i) - y_{sp})^2 + \sum_{i \in x_p} (\tilde{x}(k+i) - x_{sp})^2 + w_{MV} \left(\sum_{j \in R_p} \Delta R(k+j)^2 + \sum_{j \in H_p} \Delta H(k+j)^2 \right)$$

subject to:

$$[\tilde{y}(k+i), \tilde{x}(k+i)] = GNN(y(k-5), \dots, y(k), x(k-5), \dots, x(k), \dots, F(k), R(k-5), \dots, R(k+i-1), H(k-5), \dots, H(k+i-1)) + [dy(k+i), dx(k+i)]$$

$$[dy(k+i), dx(k+i)] = [y(k), x(k)] - GNN(y(k-5-i), \dots, y(k-i), x(k-5-i), \dots, x(k-i), \dots, F(k-i), R(k-5-i), R(k-1), H(k-5-i), H(k-1))$$

$$\Delta R(k+j) = R(k+j) - R(k+j-1)$$

$$\Delta H(k+j) = H(k+j) - H(k+j-1)$$

$$0 \leq \tilde{y}(k+i) \leq 1$$

$$0 \leq \tilde{x}(k+i) \leq 1$$

$$0 \leq R(k+j) \leq 1$$

$$0 \leq H(k+j) \leq 1 \tag{5.10}$$

Equation Set (5.10) includes the objective function, the GNN model, the model adjustment, and the hard constraints on the MVs, CVs, and MV movements. \tilde{y} and \tilde{x} are the predicted top and bottom compositions after adjustment. dy and dx are the adjustment for top and bottom compositions, respectively. It should be noted that although the same nomenclature such as R , H , F , y , x are used as before to represent the process variables, in Equation Set (5.10), these variables are all scaled to their operating

range to be numbers that are in the range of 0 to 1. R_f and H_f are future MVs, i.e., the optimized variables. Parameters for the optimization problems are listed in Table 5.8.

Table 5.8 Tuning parameter for GNNMPC in simulation tests

w_{MV}	y_p	x_p	R_p	H_p
0.1	{1,2,3,5,10}	{1,2,3,5,10}	{0,4}	{0,4}

For this work, w_{MV} , the MV suppression factor, is used as the sole tuning parameter of GNNMPC. Though other parameters such as R_p or H_p are also tuning parameters, they are not discussed here. References are available for choosing the prediction and the control horizons (Garcia & Morari, 1989; Rawlings & Muske, 1993; Scattolini & Bittanti, 1990). The Matlab codes for GNNMPC (in the form of the optimization problem as shown in Equation Set (5.10)) are presented in Appendix D.1.

In the following section, the capabilities of GNNMPC of tracking setpoints, tuning aggressiveness, rejecting disturbances and handling constraints will be demonstrated.

5.6.1 Setpoint Tracking and Controller Aggressiveness

Table 5.9 lists the cases studied for the servo mode (setpoint tracking). In all cases, the process was initially at closed-loop steady state under GNNMPC with w_{mv} at 0.1. The MVs were kept at $\{R, H\}=\{114.13, 48.70\}$, and the steady state CVs were kept at the setpoint $\{y, x\}=\{0.85, 0.03\}$.

Table 5.9 Setpoint tracking case studies
 (for all cases, the process is initially at closed-loop steady state $\{y,x\}=\{0.85, 0.03\}$
 under GNNMPC with $w_{mv}=0.1$, $\{R,H\}=\{114.13, 48.70\}$)

	New setpoint for $\{y,x\}$	w_{mv}	New MVs ($\{R,H\}$)
Case 1	$\{0.65, 0.005\}$	0.1	N/A
Case 2	$\{0.65, 0.005\}$	0.5	$\{96.40, 53.72\}$
Case 3	$\{0.9, 0.08\}$	0.1	$\{120.06, 45.21\}$
Case 4	$\{0.9,0.08\}$	0.02	$\{119.98, 45.19\}$

In Case 1 and Case 2, the setpoint, $\{y_{sp}, x_{sp}\}$ was changed to $\{0.65, 0.005\}$ at 20 minutes. In Case 1, w_{mv} is set to be 0.1. In Case 2, w_{mv} is set to be 0.5. The control results are shown in Figure 5.16 for Case 1 and in Figure 5.17 for Case 2.

Figure 5.16 (Case 1) shows that the GNNMPC system is unstable when w_{mv} equals 0.1. Figure 5.17 (Case 2) shows that the GNNMPC system performs well when w_{mv} equals 0.5. In Case 1, GNNMPC responds immediately after the setpoint change is introduced. The reflux flowrate is pushed to its lower limit, 90 mole/hr and the heating power to about 58% full power (near the upper limit of 60% full power), which causes a rapid decrease of the top composition and the bottom composition, toward their setpoints. The top composition and the bottom composition hit their setpoint within 15 minutes comparing to the process's settling time of 60 minutes. Unfortunately, GNNMPC does not settle down, instead, both MVs oscillate. The hard constraints on MVs make MVs

oscillate between their limits (the lower limit of 90 mole/hr for the reflux flowrate and the upper limit of 60% full power for the heating power) and some mediate values.

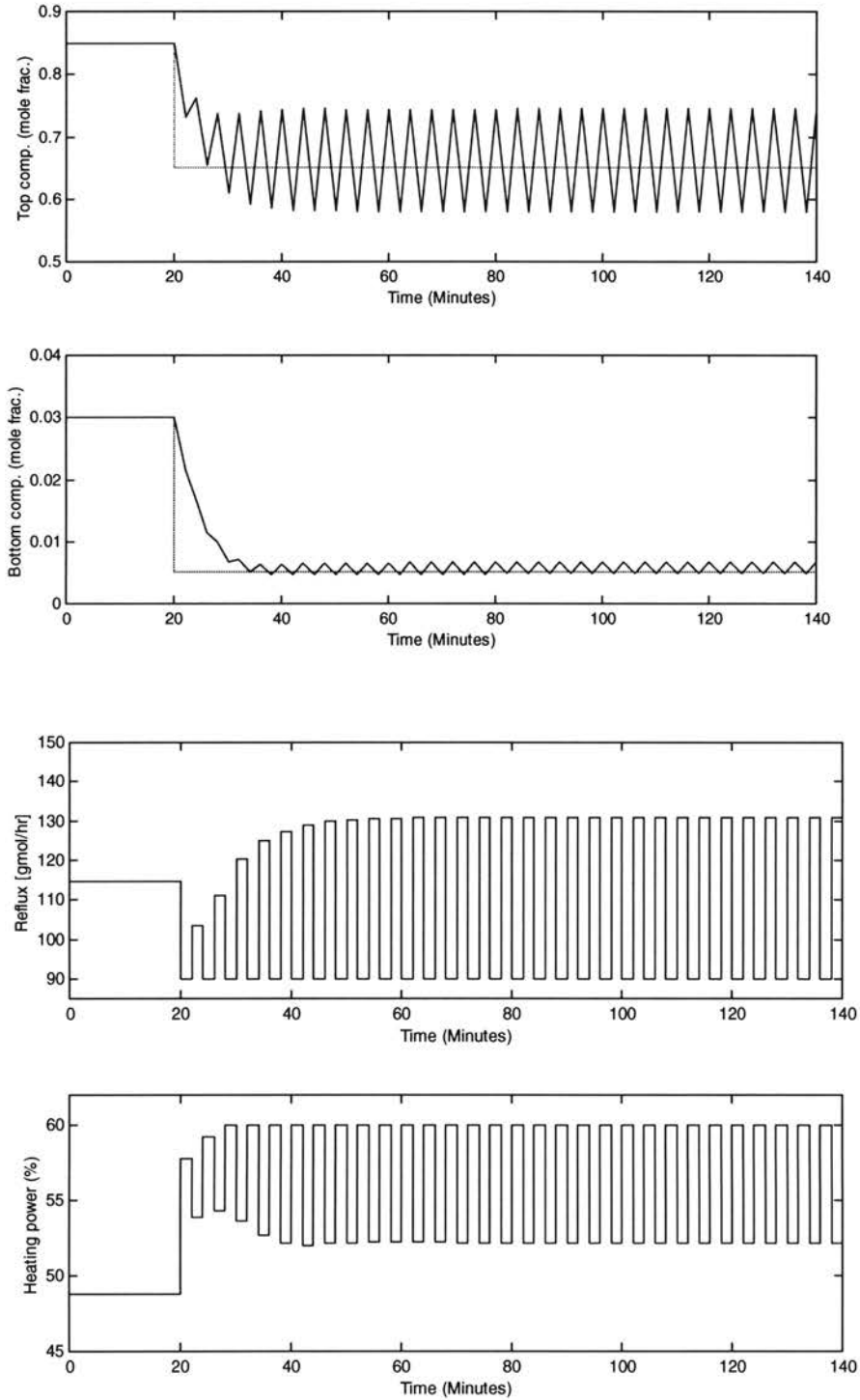


Figure 5.16 GNNMPC for setpoint tracking (Case 1)

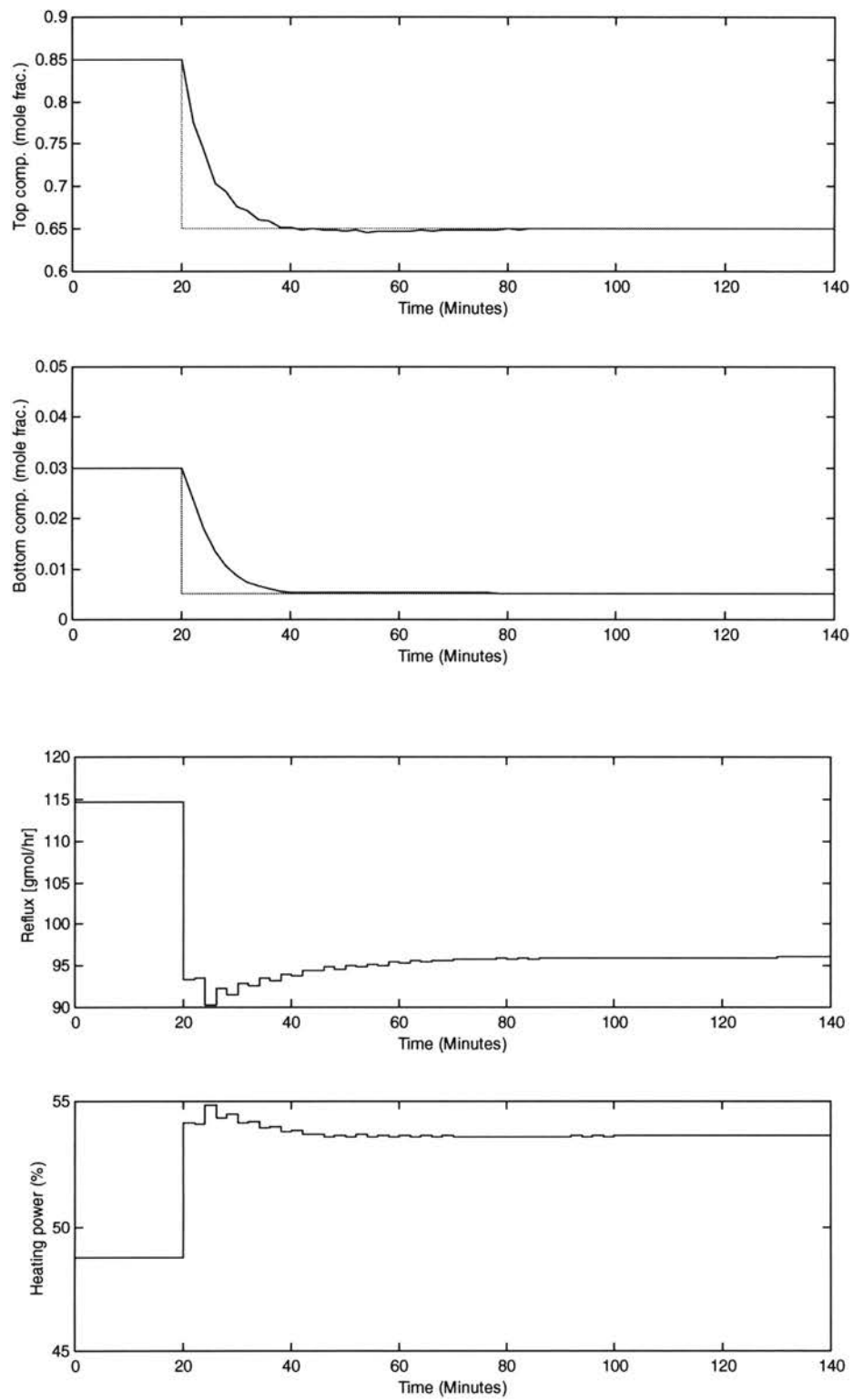


Figure 5.17 GNNMPC for setpoint tracking (Case 2)

In Case 2, the MVs also took actions immediately when the setpoint changed, after a small “excessive action”, settled down at values that keep the CVs at its setpoint. The reflux flowrate is pushed to about 93 mole/hr and the heating power to about 54.4% full power, which is less aggressive than the GNNMPC in Case 1. The MVs do not hit constraints but bring the CVs to their new setpoints with 20 minutes (as opposed to 60 minutes of the process’s settling time). Because the only difference of design in Case 1 and Case 2 is w_{mv} , it can be concluded that w_{mv} equals 0.1 too aggressive while w_{mv} equals 0.5 brings good control performance.

In Case 3 and Case 4, the setpoint, $\{y_{sp}, x_{sp}\}$ was changed to $\{0.90, 0.08\}$ at 20 minutes. In Case 3, w_{mv} is set to be 0.1. In Case 4, w_{mv} is set to be 0.02. The control results are shown in Figure 5.18 for Case 3 and in Figure 5.19 for Case 4.

In Case 3 (Figure 5.18), GNNMPC responds immediately to the setpoint change, pushing the reflux flowrate to about 130 mole/hr and the heating power to its lower limit of 45% full power. However, the MVs then are very sluggish to settle down. After the settling time of the process (about 60 minutes) at about 80 minutes, the MVs are still slowing moving, and the top composition has not settled at its new setpoint yet.

In Case 4 (Figure 5.19), with a smaller value of w_{mv} than that in Case 3, the MVs settle down at about 80 minutes and the CVs reach its new setpoint at about 70 minutes. GNNMPC first pushes the reflux flowrate to about 145 mole/hr and the heating power to 45% full power. Afterwards, the MVs oscillate with a decreased magnitude till they settle down.

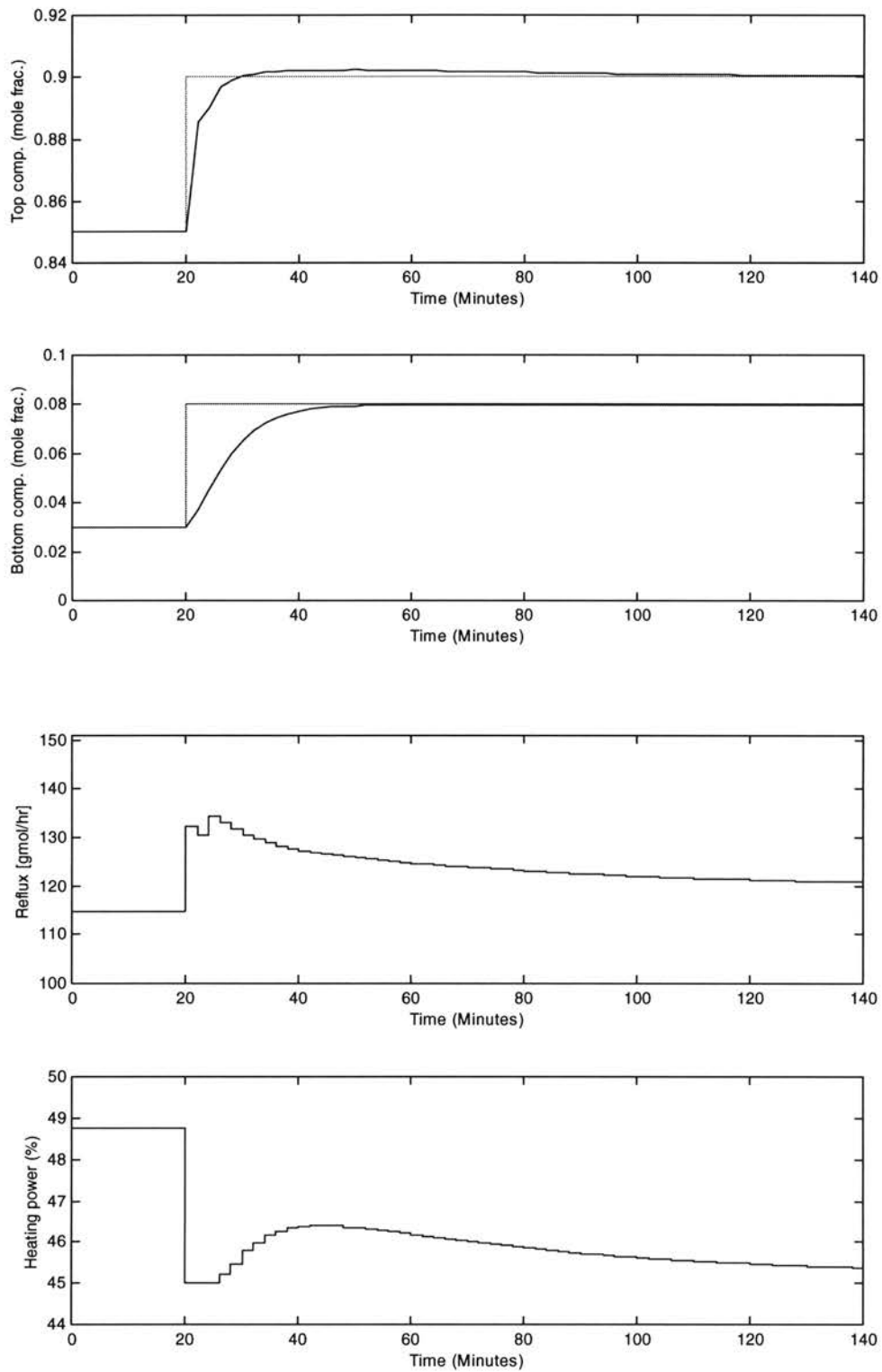


Figure 5.18 GNNMPC for setpoint tracking (Case 3)

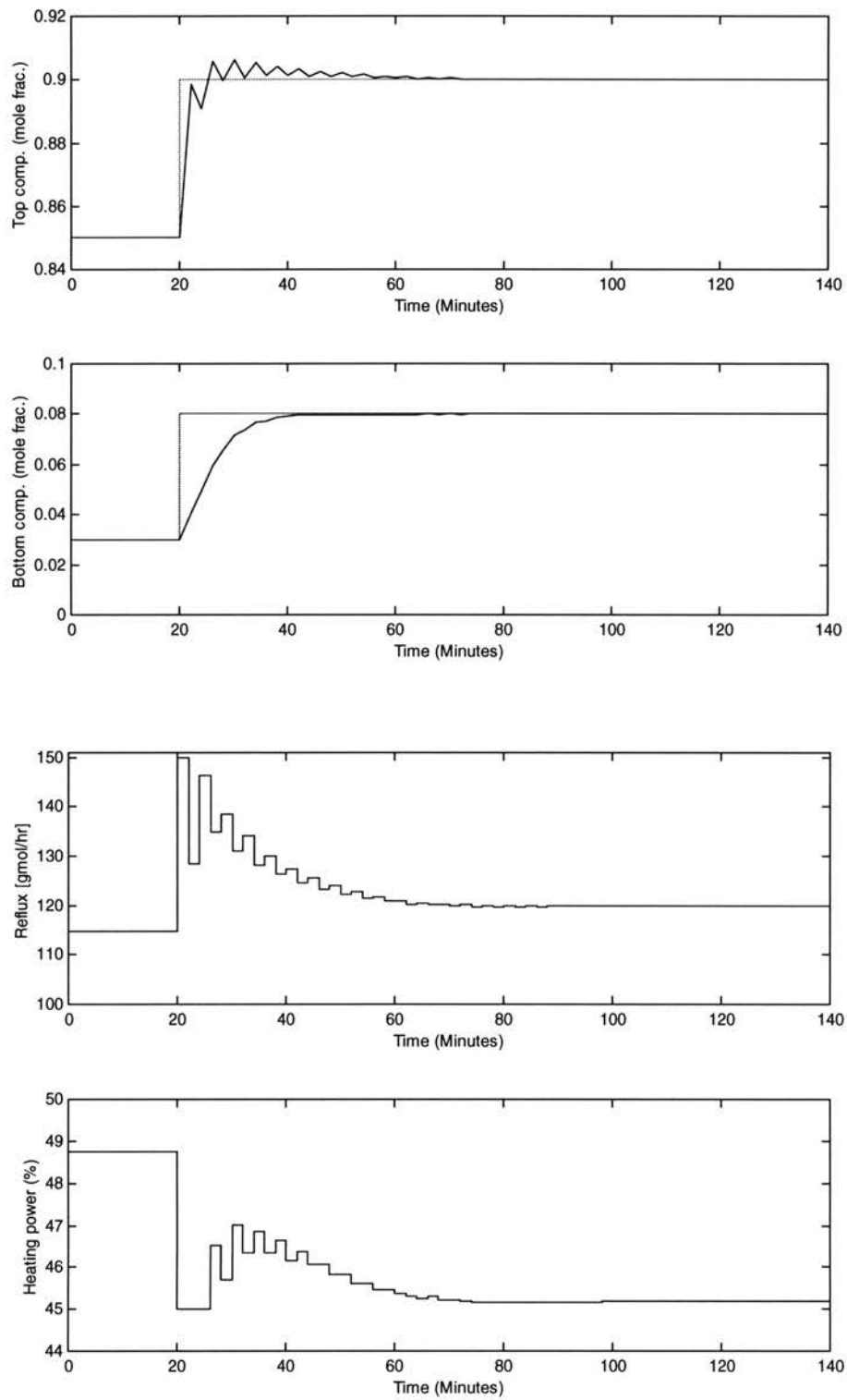


Figure 5.19 GNNMPC for setpoint tracking (Case 4)

The setpoint tracking case studies use different setpoint and MV suppression factor, w_{mv} . Control performance is sensitive to the value of w_{mv} . Further, the optimal value of w_{mv} is not fixed throughout the operating range. While w_{mv} equals 0.5 is a good value in Case 2, it would be too sluggish to be used in Case 3. Vice versa, while w_{mv} equals 0.1 leads to sluggish control in Case 3, it is so aggressive as to cause instability in Case 1. This reflects the nonlinearity of the process. The two new setpoints are chosen to be at two opposite operating regions ($\{y, x\}=\{0.65, 0.005\}$ requires low reflux flowrate and high heating power, and $\{y,x\}=\{0.9,0.08\}$ requires high reflux flowrate and low heating power). Recall Table 5.3 where the nonlinearity of the global operating was demonstrated, $\{0.65,0.005\}$ is near Region 1 in Table 5.3 and $\{0.9, 0.08\}$ is near Region 3 in Table 5.3. These two regions have distinctively different static and dynamic characteristics. While GNNMPC can understand the nonlinearity and is able to track the setpoint in different regions, it requires different suppression factor on w_{mv} to achieve good control performance. A discussion on the tuning issue is presented in Appendix E, where the comparisons between using move suppression (MV movement is part of the objective function as a penalty) and using CV damping (instead of comparing the future dynamics to the setpoint, the future dynamics is compared to a tunable first-order trajectory from the present state to the setpoint) are provided as a reference for future work.

In the following case studies, we will fix w_{mv} at 0.1 if not otherwise stated.

5.6.2 Disturbance Rejection

Table 5.10 lists the cases studies for the regulatory mode (disturbance rejection). Both feedforward load disturbance (the feed flowrate, F) and the unmeasured load disturbance (the feed composition, z) are studied.

Table 5.10 Disturbance rejection case studies
(for all cases, initially at close-loop steady state $\{y,x\}=\{0.85, 0.03\}$
under GNNMPC with $W_{mv}=0.1$, $\{R,H,F\}=\{114.13, 48.70, 280\}$)

	Disturbances	New MVs ($\{R,H\}$)
Case 5	Feed flowrate from 280 mole/hr to 300 mole/hr	{122.82, 52.12}
Case 6	Feed flowrate from 280 mole/hr to 260 mole/hr	{106.48, 45.41}
Case 7	Feed comp. from 0.25 mole frac. To 0.28 mole frac.	{120.16, 52.02}
Case 8	Feed comp. from 0.25 mole frac. To 0.22 mole frac.	{108.72,45.42}

Figures 5.20 to 5.23 show the GNNMPC performance for Cases 5 to 8, respectively. In all cases, disturbances were introduced at 20 minutes, and were rejected within the settling time (60 minutes). It is observed that the dynamics of Case 5 and Case 6 as well as the dynamics of Case 7 and Case 8 are almost mirrored with respect to the original steady state, indicating the nearly linear characteristics locally. However, even though the peak deviation in feedforward disturbance rejection cases (Case 5 and Case 6) is smaller than that in the unmeasured disturbance rejection cases (Case 7 and Case 8), since the feed composition disturbance causes more deviation as shown in Table 5.11, no conclusion can be made that there is improvement of the controller's performance for

feedforward performance over that for unmeasured disturbance. This issue will be discussed further in Chapter 8.

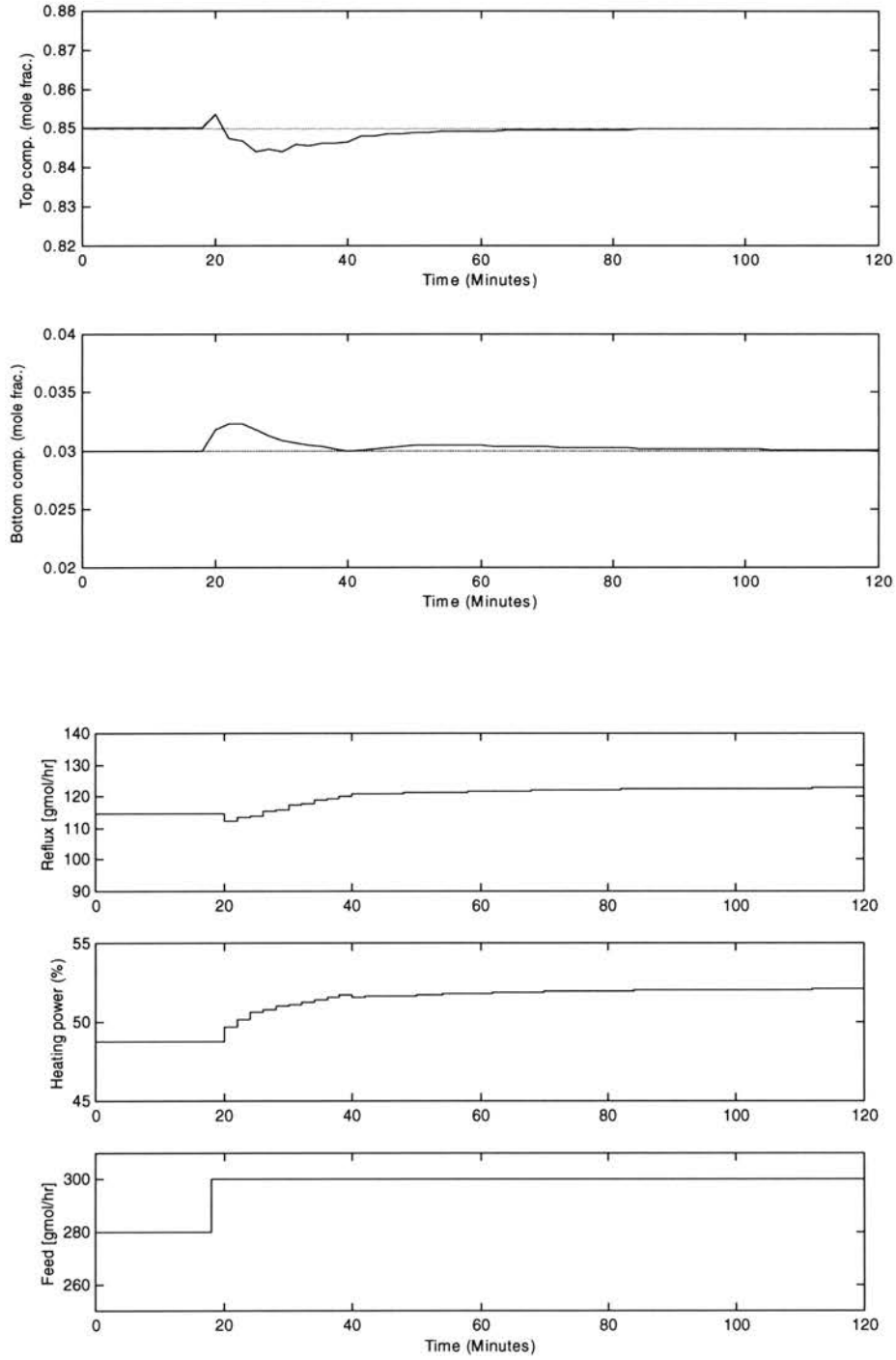


Figure 5.20 GNNMPC for disturbance rejection (Case 5)

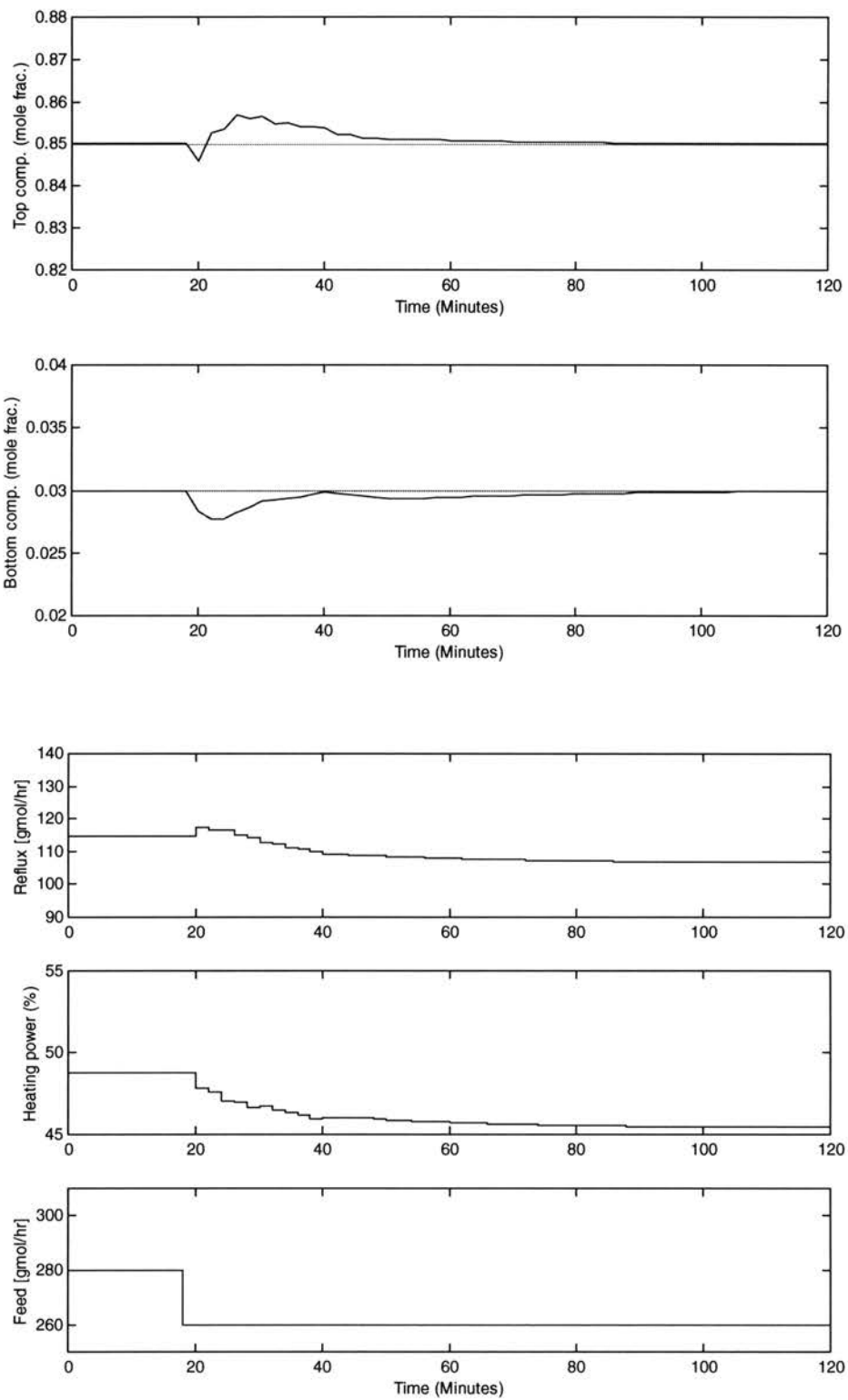


Figure 5.21 GNNMPC for disturbance rejection (Case 6)

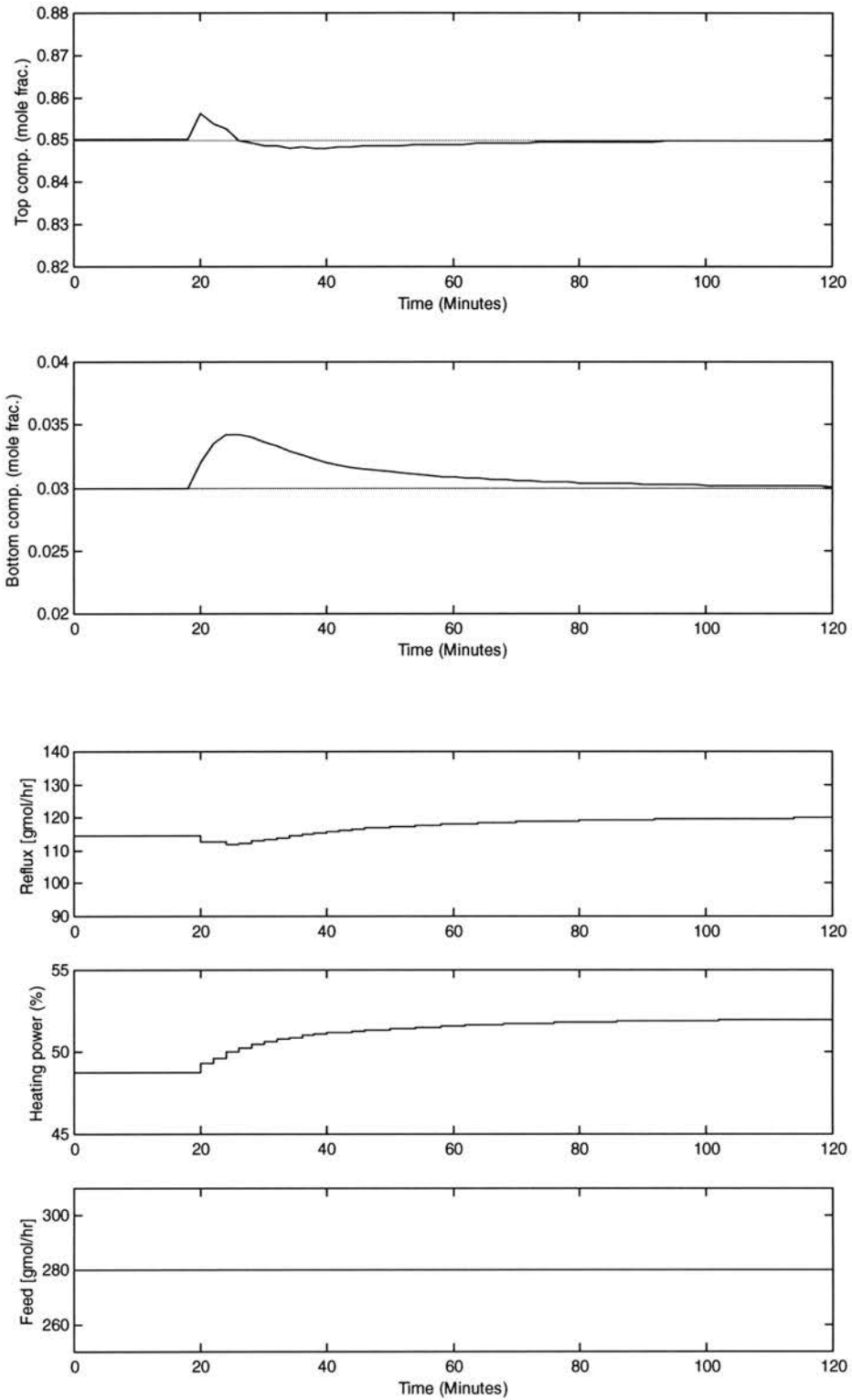


Figure 5.22 GNNMPC for disturbance rejection (Case 7)
 (feed composition changes from 0.25 to 0.28 at 20 minutes)

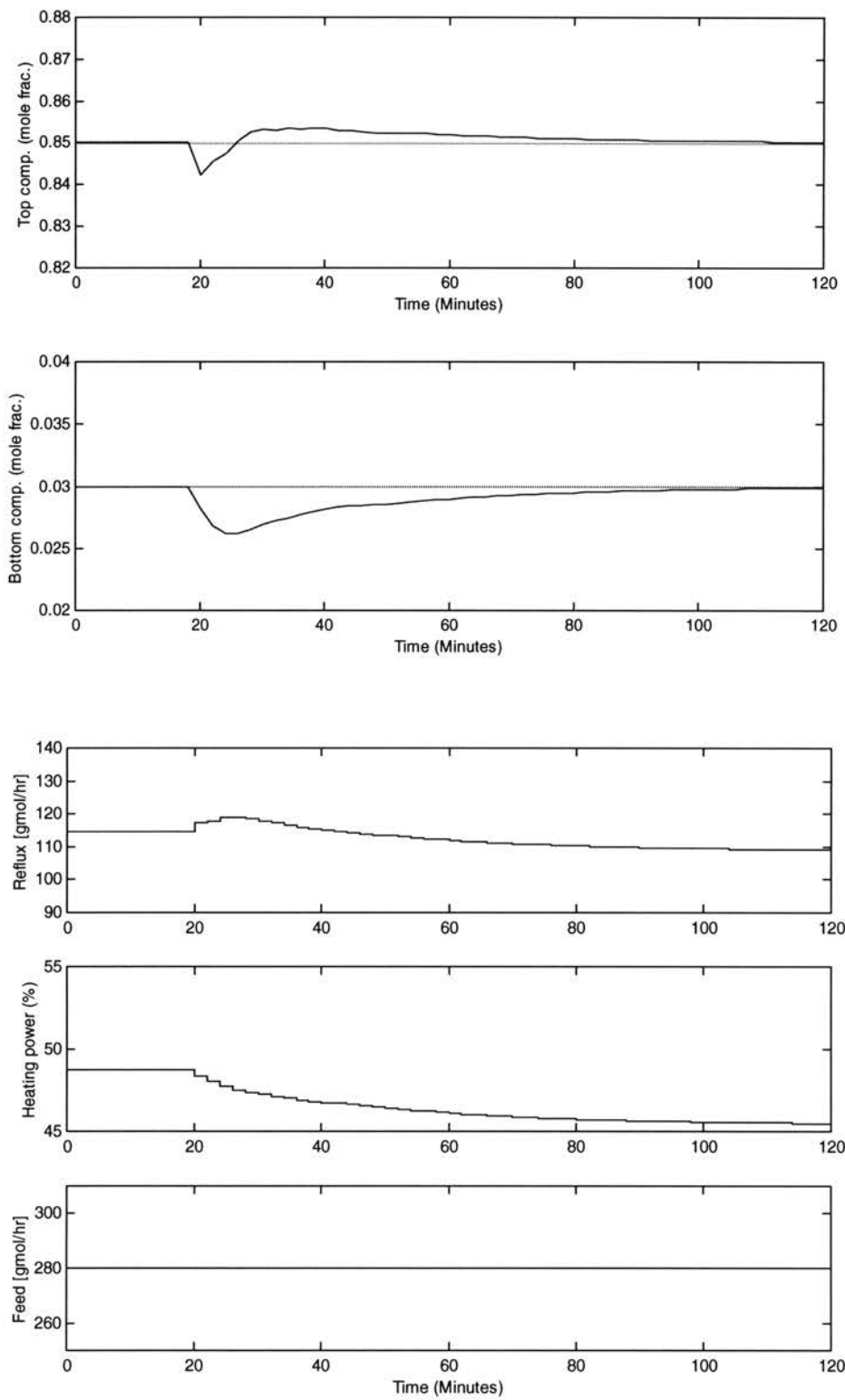


Figure 5.23 GNNMPC for disturbance rejection (Case 8)
 (feed composition changes from 0.25 to 0.22 at 20 minutes)

Table 5.11 Effect of disturbances on the process – Open-loop studies of Cases 5 to 8

	Case 5	Case 6	Case 7	Case 8
$\{y,x\}_{ss}$	{0.864, 0.045}	{0.827, 0.017}	{0.871, 0.051}	{0.816, 0.016}
$\{\Delta y, \Delta x\}_{ss}$	{0.014, 0.015}	{-0.023, -0.013}	{0.021, 0.021}	{-0.034, -0.014}

5.6.3 Constraint Handling

MPC has explicit constraint handling capability by including the constraints in the optimization problem as shown in Equation Set (5.10). Equation Set (5.10) includes most often encountered constraints in process control problems, constraints on the input and output variables. The input constraints can arise due to actuator limitations such as saturation and rate-of-change restrictions. The output constraints usually are associated with operational limitations such as specifications and safety considerations. Therefore, the constraints on the input and output variables are often expressed as simple bounds on the variables, which is the case in Equation Set (5.10). These constraints are hard constraints in the sense that they can not be violated. The corresponding other kind of constraint is called soft constraint, which can be violated but there will be a penalty for the violation. Only hard constraints are considered in this work.

Table 5.12 lists the cases studied for the performance of GNNMPC on handling MV constraints.

Table 5.12 Constraint handling case studies

	Initial steady state	Changes made	Constraint hit
Case 9	$\{R,H,F\}=\{122.53, 52.05, 300\}$ $\{y,x\}=\{0.85, 0.03\}$	Setpoint from $\{0.85, 0.03\}$ to $\{0.94, 0.09\}$	R=150 (upper limit)
Case 10	$\{R,H,F\}=\{125.31, 46.65, 280\}$ $\{y,x\}=\{0.9, 0.075\}$	Feed flowrate from 280 mole/hr to 260 mole/hr	H=45 (low limit)
Case 11	$\{R,H,F\}=\{131.26, 48.25, 280\}$ $\{y,x\}=\{0.9, 0.07\}$	Feed comp. from 0.25 mole frac. to 0.2 mole frac.	H=45 (low limit)

In Case 9 (shown in Figure 5.24), when the setpoint is changed to $\{0.94, 0.09\}$ at 20 minutes, the reflux flowrate increases and the heating power decreases, both cause the CVs to approach their setpoints. However, the heating power soon increases back, realizing that the future bottom composition would surpass its setpoint with the constrained reflux flowrate. The reflux flowrate is increased all the way to its upper limit of 150 mole/hr, trying to reach the setpoint of the top composition. The system reaches a static point where deviations of the top composition and the bottom composition are “best” balanced as defined by the objective function. If the reflux flowrate is decreased, the bottom composition would be closer to its setpoint while the top composition would be further away from its setpoint. If the heating power is increased to bring the bottom composition closer to its setpoint, it will cause the top composition to deviate more from its setpoint. Vice versa, if the heating power is decreased to bring the top composition closer to its setpoint, it will cause the bottom composition to increase further and thus deviate more from its setpoint. When the setpoint changes back to $\{0.85, 0.03\}$, GNNMPC immediately responds and relieves the reflux flowrate from its upper limit

constraint, and the CVs track their setpoints within desired time, expectedly there is no wind-up.

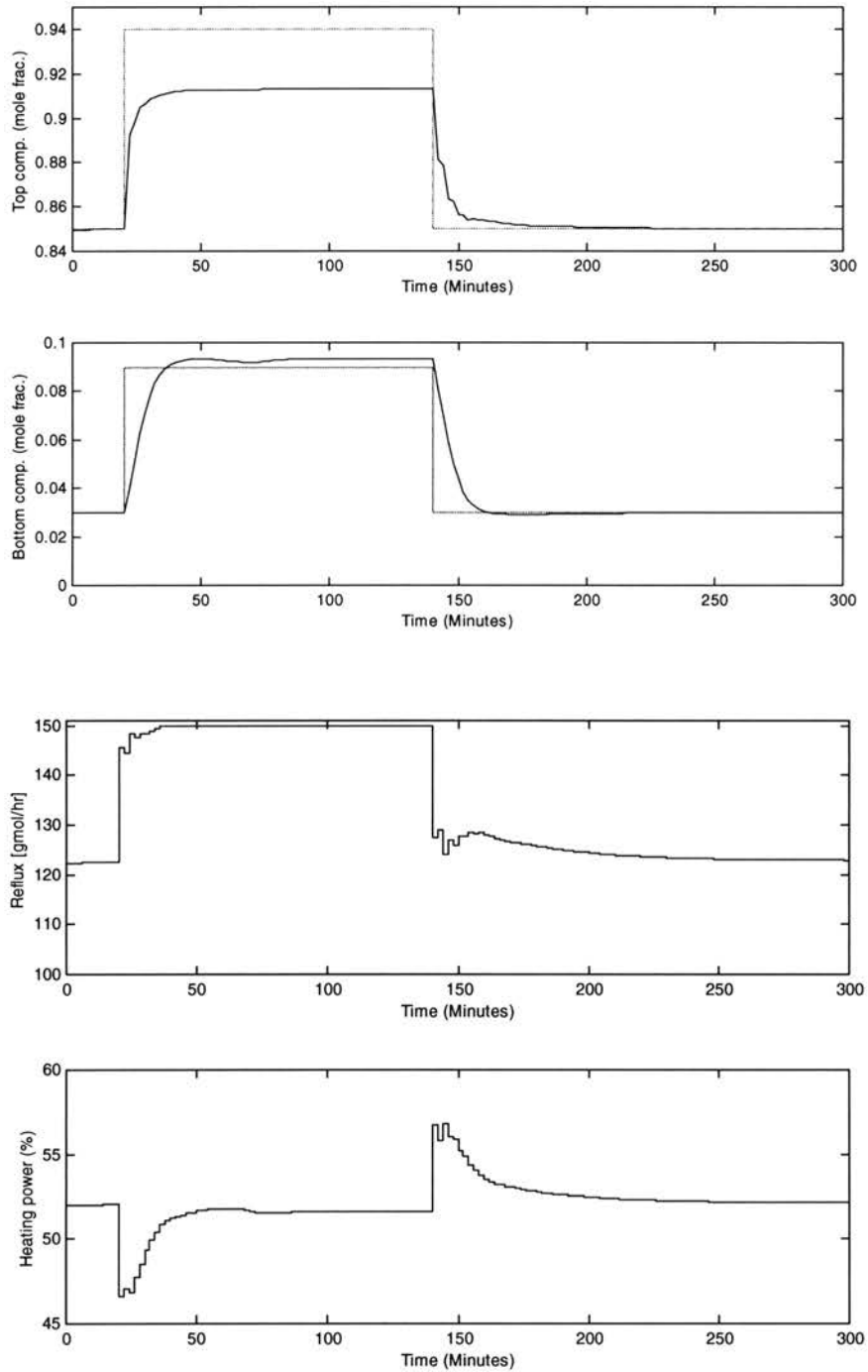


Figure 5.24 MV constraint due to setpoint change (Case 9)

In Case 10 and Case 11, shown in Figure 5.25 and Figure 26 respectively, where disturbances (feed flowrate disturbance in Case 10 and feed composition disturbance in Case 11) cause the heating power to reach its lower limit of 45% full power, similar phenomena happen to that observed in Case 9. The CVs stay at points where the best balance between the deviation of the top composition and the deviation of the bottom composition is achieved. When the constraint is relieved, GNNMPC is able to respond immediately and bring the CVs to their setpoints.

Table 5.13 describes the case studied for constraint on the rate-of-change of the MVs. Note that the constraints are expressed in their physical units (mole flowrate for the reflux, R, and percentage of full power for the heating power, H).

Table 5.13 Constraint on the rate-of-change of MVs

	Changes made	w_{mv}	Constraint
Case 12	Setpoint from {0.85 0.03} to {0.9, 0.08}	0.02	$\Delta R \leq 6$ & $\Delta H \leq 1.5$

Case 12 is designed to be exactly the same as Case 4 except that constraints on the rate-of-change of MVs are enforced such that for each control action (step), the reflux flowrate is not allowed to change by more than 6 mole/hr (10% of its range of 60 mole/hr) and the heating power is not allowed to change by more than 1.5% full power (10% of its range of 15% full power) both at the next step and at steps in the future. The control performance is shown in Figure 5.27.

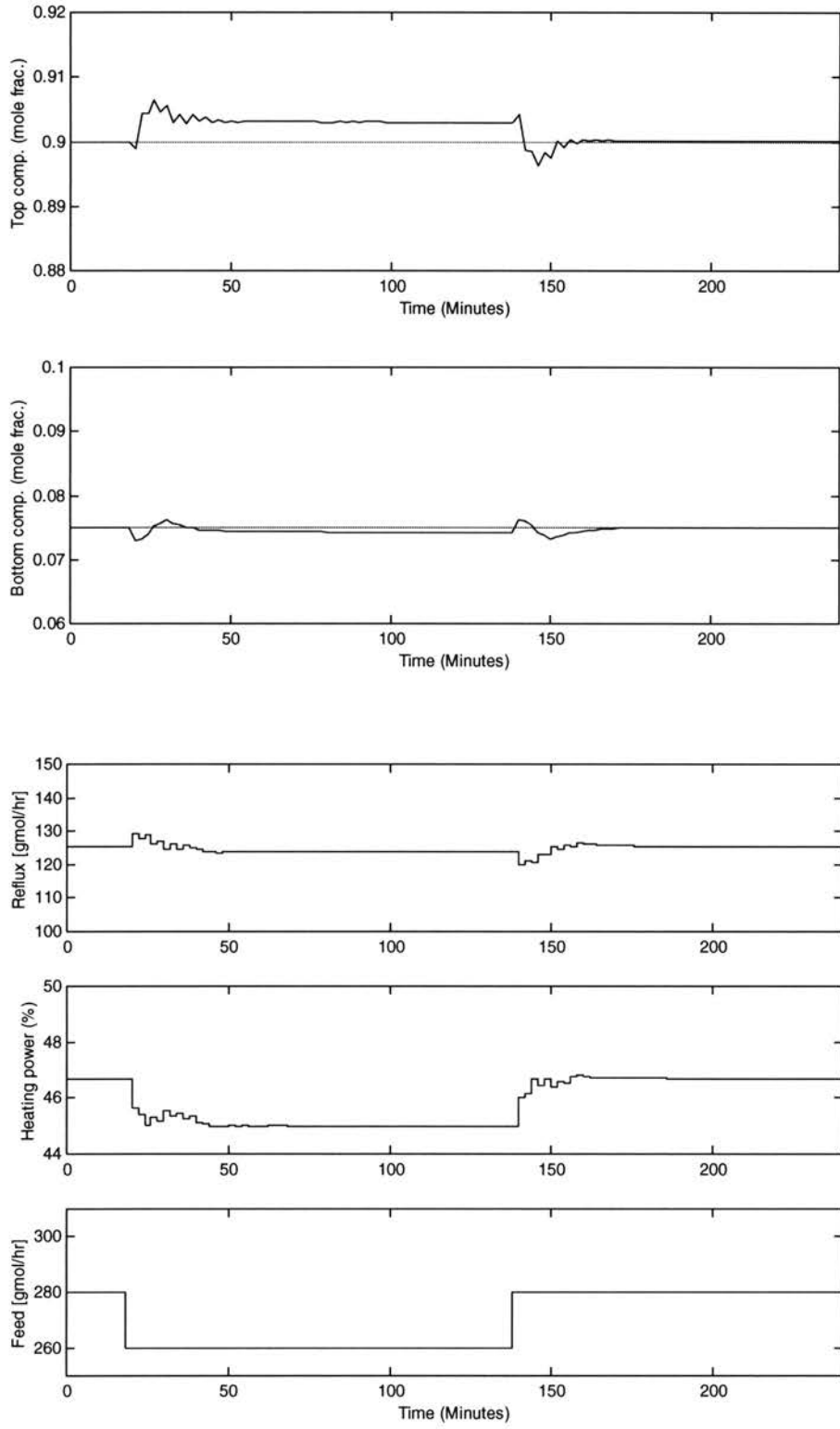


Figure 5.25 MV constraint due to feed disturbance (Case 10)

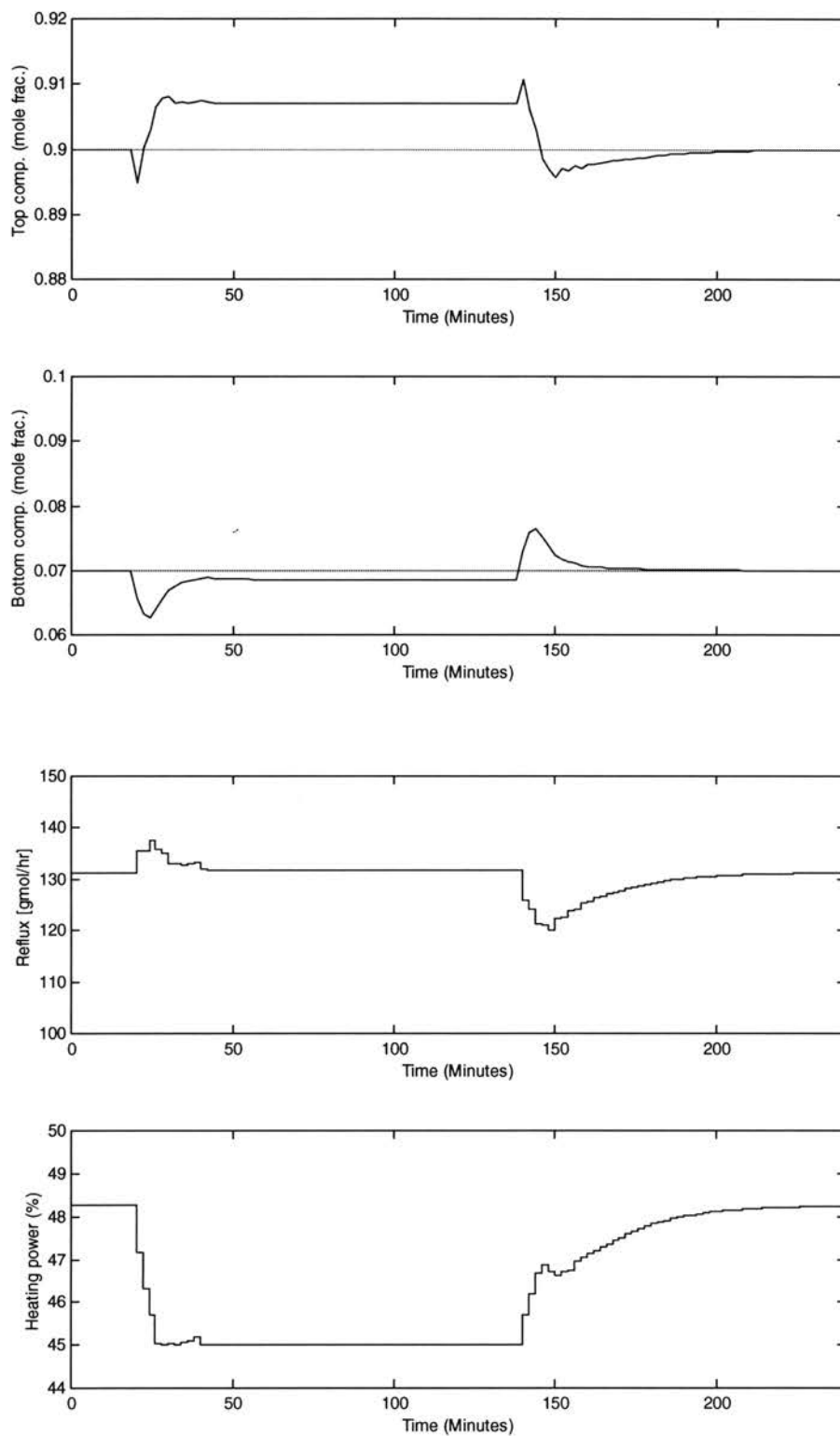


Figure 5.26 MV constraint due to feed composition disturbance (Case 11)

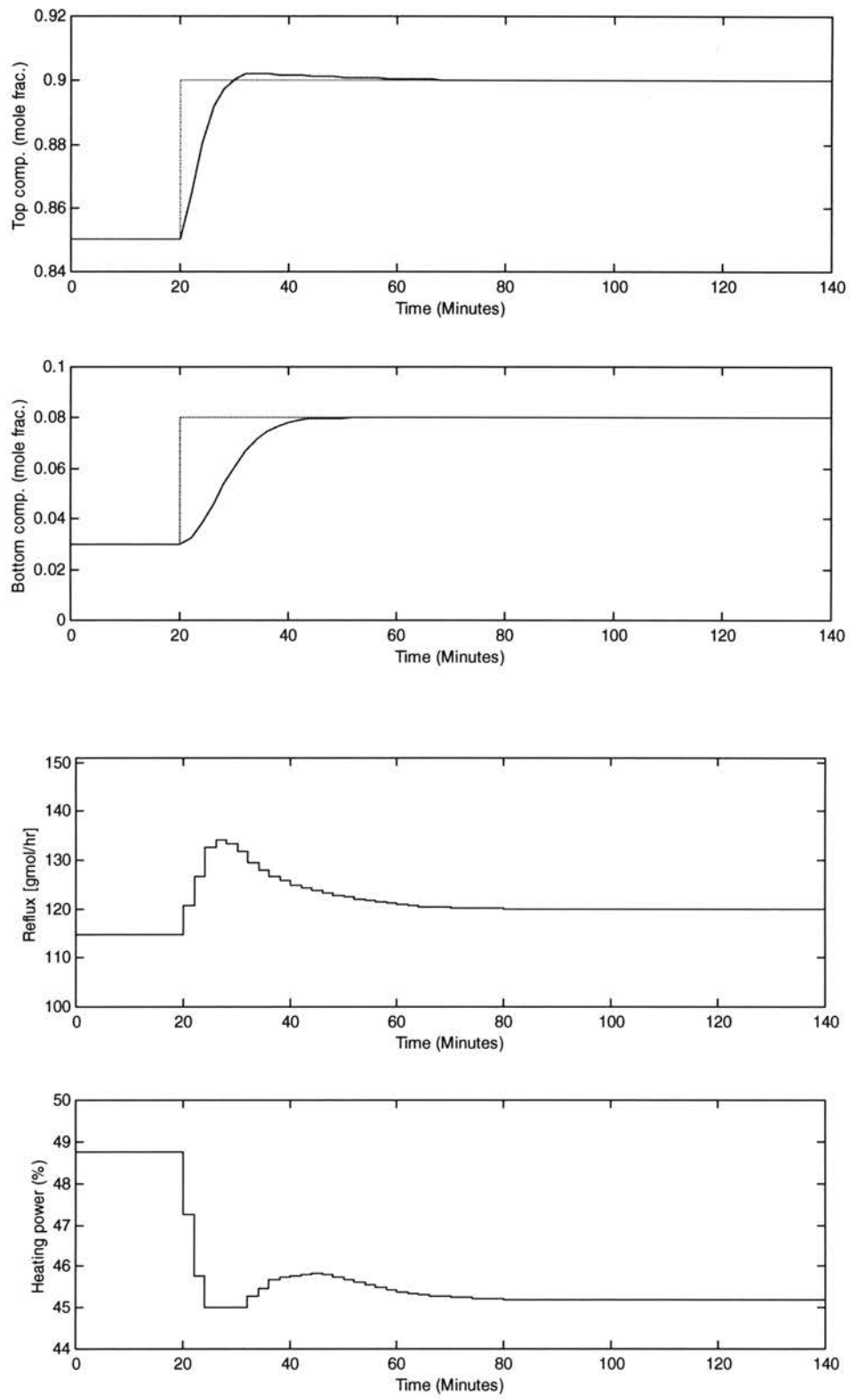


Figure 5.27 Constraint on the MV movement (Case 12)

Comparing Case 12 to Case 4 as shown in Figure 5.27 and Figure 5.19 respectively, when the MV movement constraint is enforced (Figure 5.27), the dynamics of the MVs have been “smoothed out”, which leads to much smoother transient period for the CVs. Notice that the MV constraint also leads to less overshoot of the MVs, with the peak values of {Reflux, Heating power} at about {135, 45} compared to {145, 45} in Figure 5.19.

CHAPTER 6

GNNMPC AND THE EXPERIMENTAL DISTILLATION PROCESS

There is a need for both simulation and experimental testing. Simulations are clearer, go faster, and can be used for detailed analysis. But, they don't establish the credibility of a real world application, because simulations only express what the designer knows. In this chapter, experimental runs are used to confirm and establish credibility of the simulation findings. The experimental setup has been described in Chapter 4.

6.1 Operating Range

The operating range of the experimental distillation process is shown in Table 6.1. This operating range ensures avoidance of flooding and weeping operating conditions.

Table 6.1 Operating range of the experimental distillation column

	RANGE
Feed flowrate (gmol/hr)	250-310
Reflux flowrate(gmol/hr)	90-150
Heating power (% full power)	40-60
Top product composition (mole fraction of methanol)	0.6-0.95
Bottom product composition (mole fraction of methanol)	0-0.2

6.2 Open Loop Responses

The open loop response runs show the response of the system to disturbances (intentional and unintentional), the noise and drift levels in the system, the response time to the process inputs, and the operating constraint on the system.

The Step responses of the three cases are shown in Figures 6.1 to 6.3. The process starts at steady state, then a step change of the process input is made, the process runs till it reaches steady state (due to the long running time, the bottom composition was not fully developed to its steady state when the run was stopped).

It is obvious from the step responses that the response time of the bottom composition is much longer than that of the top composition. It takes about 70 minutes for the top composition to settle down while it takes more than 300 minutes for the bottom composition to settle down. This response time discrepancy is due to the large volume of the reboiler (14.2 L) in comparison to the tray size and the volume of the reflux drum (0.535 L).

As also shown in Figures 6.1 to 6.3, the noise level of the top composition is much larger than that of the bottom composition. A detailed look at the upper plot in Figure 6.2 is shown in Figure 6.4 to compare the noise level. As shown in Figure 6.4, the noise level of the top composition is about 0.012 mole fraction (3.43% of its operating range) and that of the bottom composition is about 0.001 (0.5% of its operating range) mole fraction.

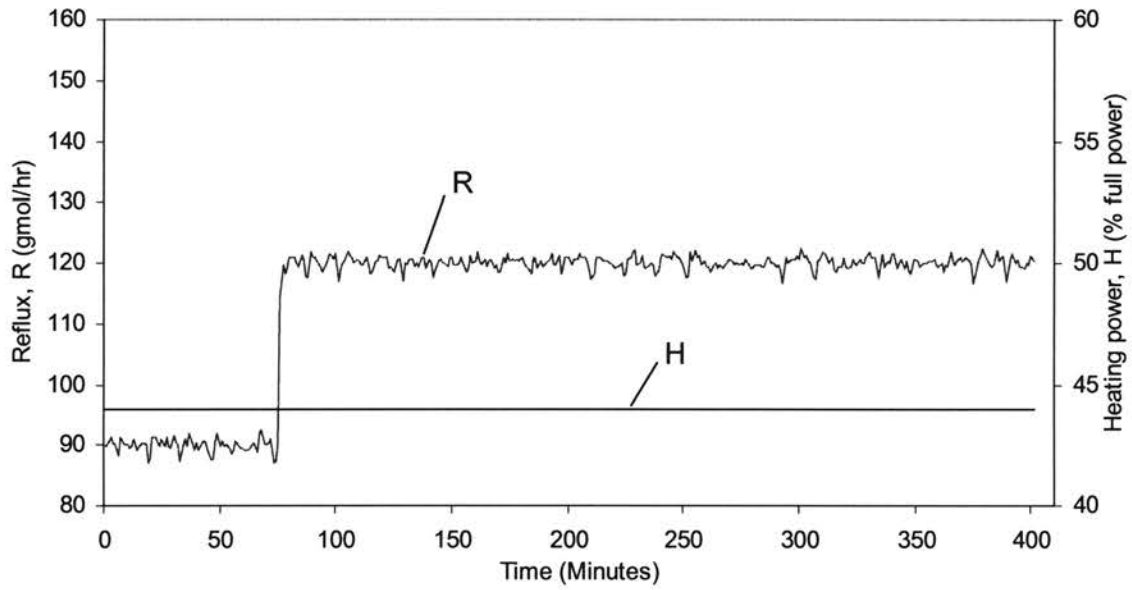
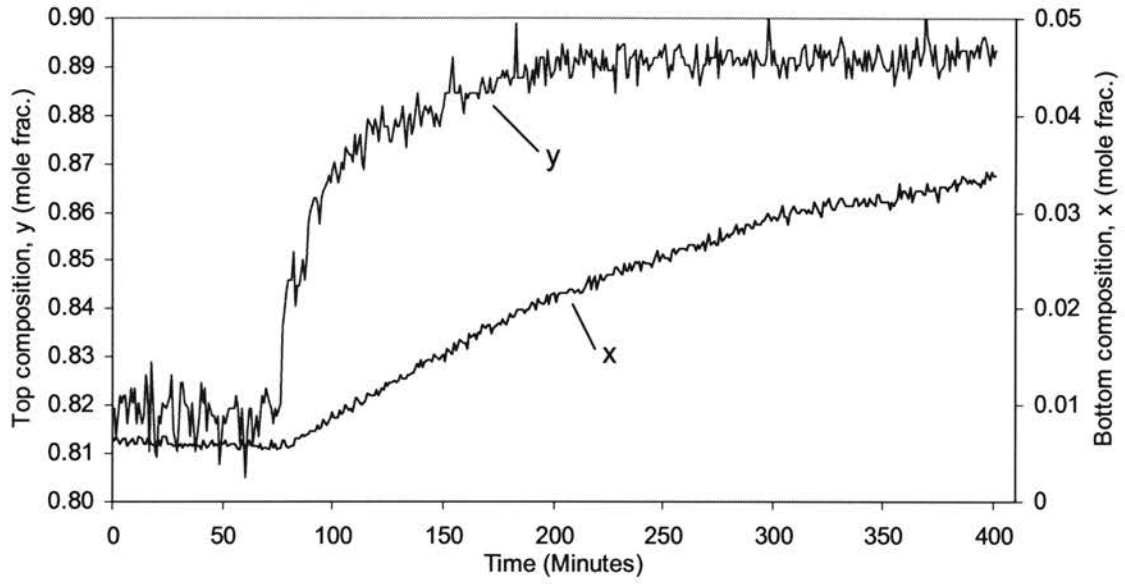


Figure 6.1 Step response Case 1

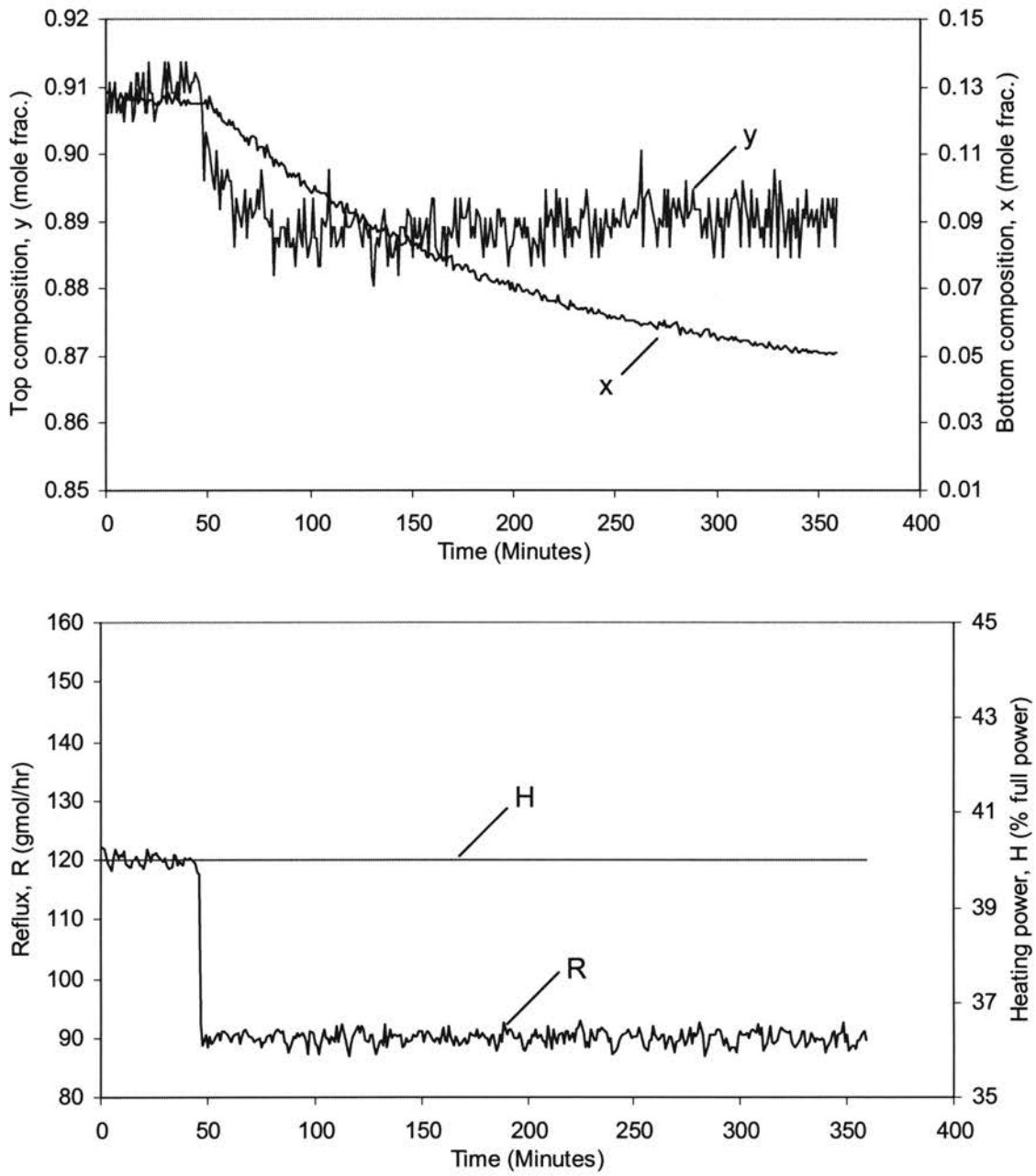


Figure 6.2 Step response Case 2

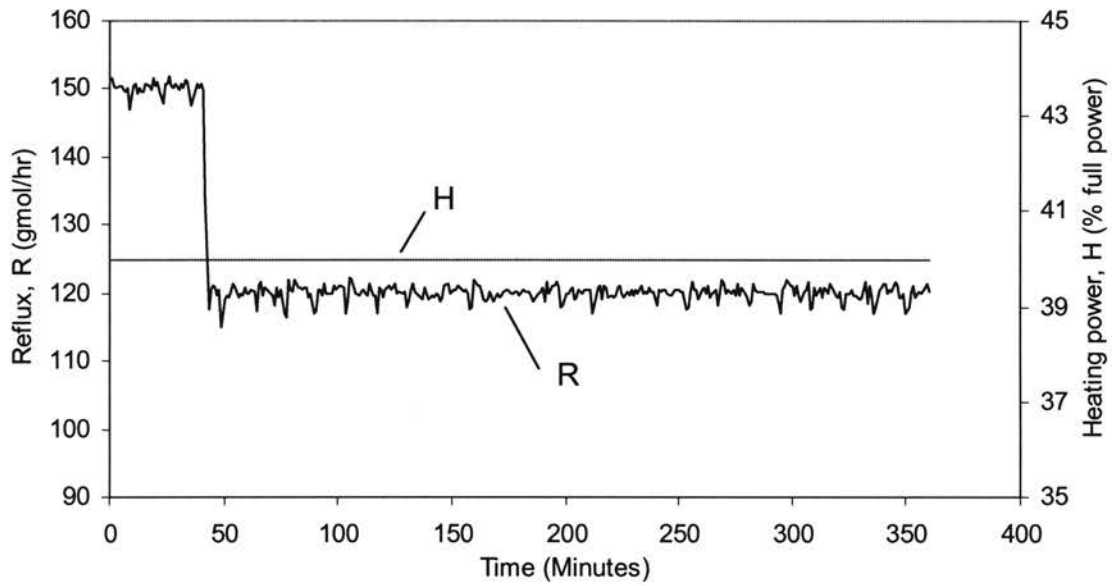
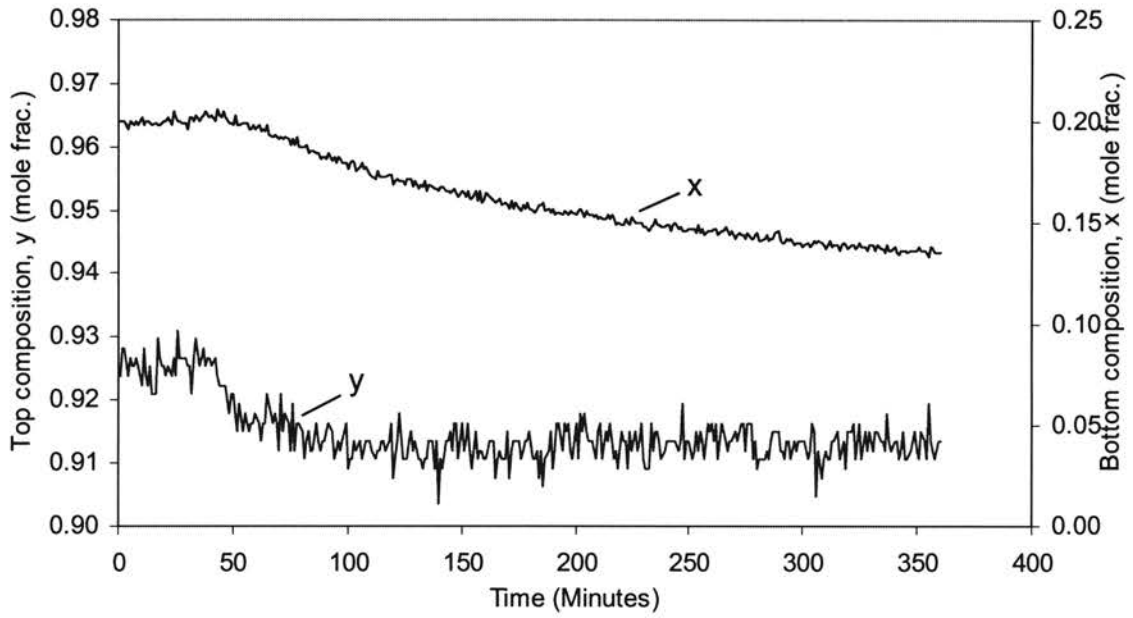


Figure 6.3 Step response Case 3

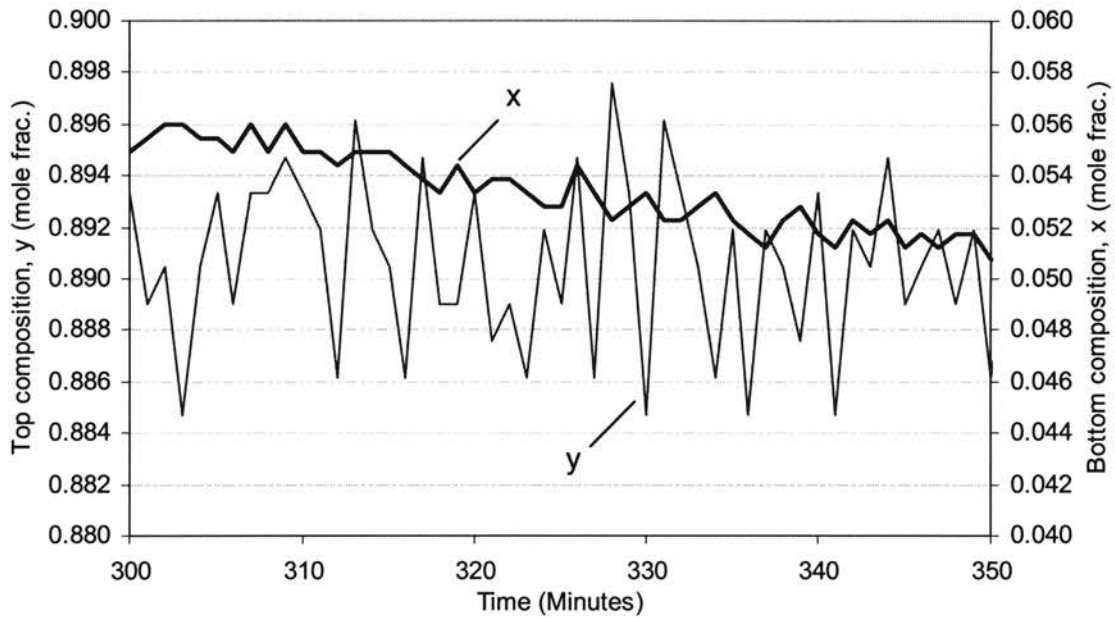


Figure 6.4 Noise level of top and bottom compositions (explosion of Figure 6.2)

Table 6.2 lists the steady state values in three step response case studies. Due to the noise of the process, the steady state values of the CVs shown in Table 6.2 are averaged measurements over a period of time when the process is considered to be at steady state (for the top compositions, sampled values of the last 60 minutes were averaged, for the bottom compositions, sampled values of the last 5 minutes were averaged, the sampling rate was 10 seconds).

It can be noticed that, the same process inputs of $\{120, 40\}$ in the start of Case 2 and the end of Case 3 do not lead to the same steady state process outputs. The difference between the top composition of 0.9083 mole fraction in Case 2 and that of 0.9128 mole fraction in Case 3 is 0.0045 mole fraction, about 1% of the top composition's operating range. The difference between the bottom composition of 0.1265 mole fraction in Case 2

and that of 0.1352 mole fraction in Case 3 is 0.0087 mole fraction, about 4.5% of the bottom composition's operating range. Since the steady state values are filtered (averaged) values, the inconsistency is believed due to the natural disturbances from run to run, such as ambient heat loss, feed composition, and atmosphere pressure. Beside this, the much larger discrepancy in the bottom composition is because the bottom composition in Case 3 has not fully reached steady state when the run was stopped, as will be shown in Figure 6.3.

Table 6.2 Steady state values in step response case studies
(in all cases, the feed flowrate is 280 gmol/hr)

		Reflux (gmol/hr)	Heating power (%)	Top comp. (mole frac.)	Bottom comp. (mole frac.)
Case 1	start	90	44	0.8187	0.0061
	end	120	44	0.8929	0.0340
Case 2	start	120	40	0.9083	0.1265
	end	90	40	0.8904	0.0510
Case 3	start	150	40	0.9254	0.1997
	end	120	40	0.9128	0.1352

The extent of static nonlinearity is not observed in the case studies. In Case 2, a 30 gmol/hr step change of the reflux flowrate leads to a top composition change of 0.0179 mole fraction and a bottom composition change of 0.0755 mole fraction. The same amount of step change, but in a different operating region of the reflux flowrate in Case 3 leads to a top composition change of 0.0126 mole fraction and the bottom composition change of 0.0645 mole fraction. The gain only changed by a factor of about 1.5:1.

Investigation on Case 1 and Case 2, however, shows severe interaction of the process. In Case 1, when the heating power is at 44% full power, a change of the reflux flowrate from 90 gmol/hr to 120 gmol/hr results in an increase of 0.0742 mole fraction of the top composition and an increase of 0.0279 mole fraction of the bottom composition. In Case 2, the same change of the reflux flowrate from 90 gmol/hr to 120 gmol/hr but with a different heating power of 40% full power results in an increase of 0.0179 mole fraction of the top composition and an increase of 0.0755 mole fraction of the bottom composition. The different amount of change in both the top composition (differ by about 3 times) and the bottom composition (differ by about 2 times) is due to different heating power, indicating interaction of the process.

It was observed that severe entraining on the top tray can occur. The top tray section was connect to the vapor feed line by a flexible Teflon expansion joint (See the photo shown in Figure 4.1). At certain operating conditions, the vapor would carry the entrained top tray liquid up through the connection joint and into the vapor feed line. In this case, the “distillate” material becomes dominated by liquid, not vapor, of the tray. Since it appears that the trays above the feed tray flood at about the same time, during this flooding/entraining event, liquid from several trays dominates composition. An online measurable value was added to monitor this phenomenon. This variable is the pressure drop in the column, ΔP (the differential pressure between atmosphere and the reboiler’s vapor space), which was observed sensitive to the entraining phenomenon. Figure 6.5 demonstrates the influence of the entraining on the process characteristics. Around points number 150, 180, 230, ΔP increases rapidly (within several minutes) to a

high value (above 5.5 inH₂O), and the top composition drops drastically (as much as 0.08 mole fraction) and immediately (within several minutes).

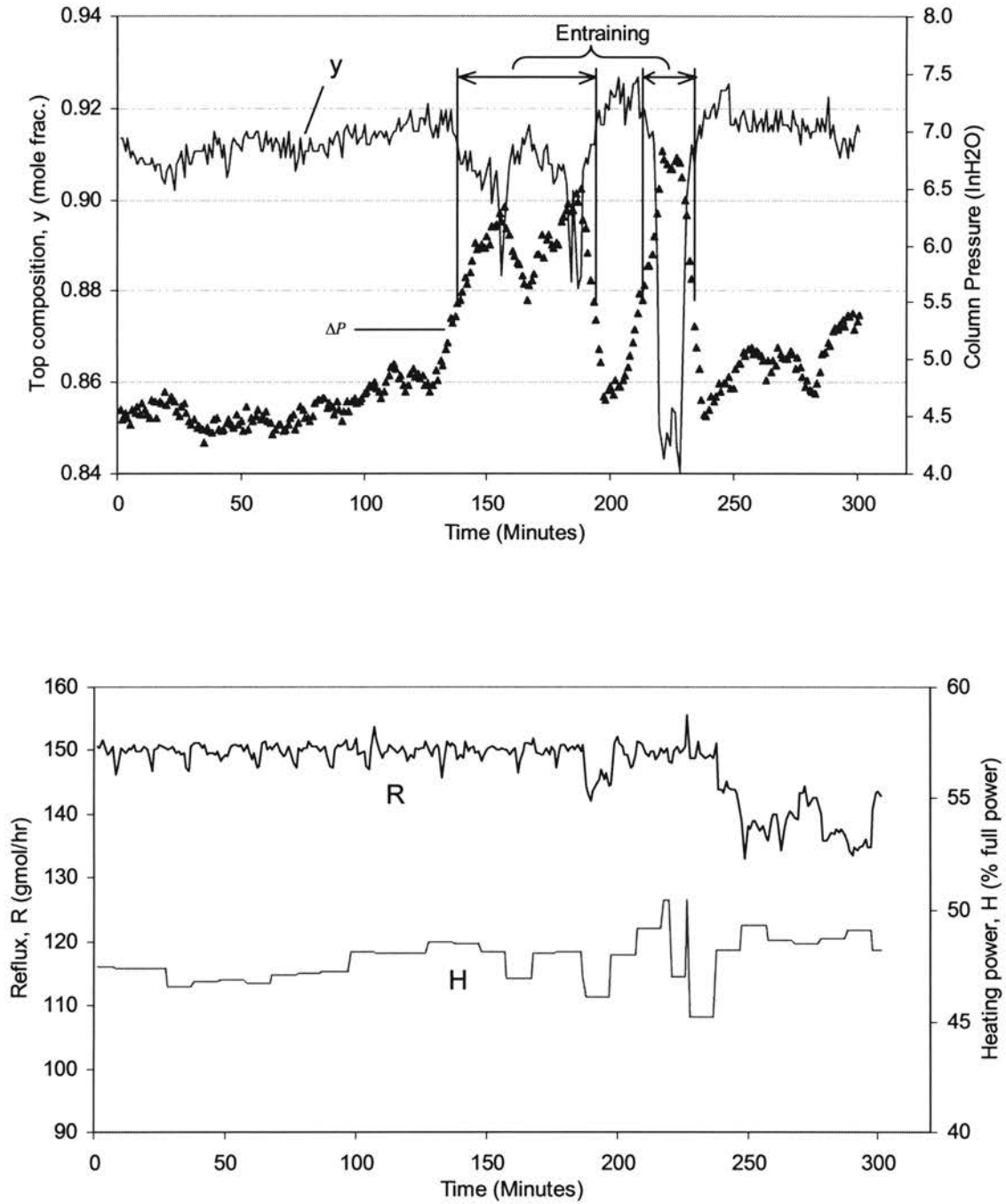


Figure 6.5 Effect of entraining on the process characteristics

6.3 GNN Modeling

The large discrepancy in the response time of the top and bottom compositions (about 70 minutes for the top composition and about 300 minutes for the bottom composition) requires further consideration on the GNN structure. Under the MPC scheme, the predictive model (dynamic model) must predict into the future far enough, otherwise, instability will probably occur (Scattolini & Bittanti, 1990). However, if the common practice is accepted that the prediction horizon is chosen to be the longer one of those for the top and the bottom compositions, separately, redundant information appears. If the bottom transient defines the prediction horizon, then for the top composition prediction, a much longer prediction horizon than necessary is used. Then with this horizon, to see the top transient, a small sample time would be required, but for the bottom composition prediction, the sample interval would be too short and unnecessary samples are taken. In turn, the model structure would be much more complicated than necessary and thus violates the principles of GNNMPC to minimize computation burden.

The approach adopted in this work is to predict the top composition and the bottom composition separately, with different sample interval and prediction horizon. The sample interval for the top composition is 10 minutes, and that for the bottom composition is chosen to be 30 minutes. Predictions with respect to sample interval is 1,2,3,4,5,6,7 for the top composition and 1,3,5,7,10 for the bottom composition, which implies that the prediction horizon for the top composition is 70 minutes and that for the bottom is composition 300 minutes.

The length of historical input-output series is selected to be 50 minutes (5 top composition sample intervals) for the top composition and 150 minutes (5 bottom

composition sample intervals) for the bottom composition. The control horizon is with respect to a sample interval of 10 minutes.

Because enough experimental operating data points are available, experimental data was used to train the GNN model (See discussions in Chapter 8 on why the simulator was not used to generate data for training). The structure of each NN in GNN and the training results are shown in Table 6.3. Note that the inputs and outputs of each NN are normalized to be in a range of 0 to 1 using each variables operating range listed in Table 6.1. Detailed description of the training procedures can be found in Chapter 5.

The RMSE is of the magnitude of 10^{-2} for the normalized top composition and of the magnitude of 10^{-3} for the normalized bottom composition. Comparing to the scaled noise level of 0.0343 for the top composition and 0.005 for the bottom composition as shown in Figure 6.4, the training results are good using the RMSE indices. However, the MAE for the normalized top composition can go as large as 0.2580 (25.80% of its operating range). The MAE for the normalized bottom composition is around 0.02 (2% of its operating range), which indicates good training performance. The MRE for the normalized top and the bottom compositions looks bad, with number over 90% for the top composition and over 50% for the bottom composition. However, as will be shown by figures, the MAE and MRE often happen at points where transient process starts. In general, the model follows the trend of the process well.

Table 6.3 Model structure and training results of the GNN

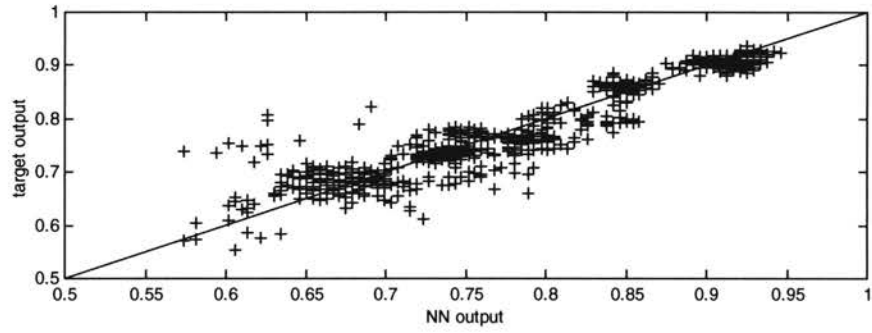
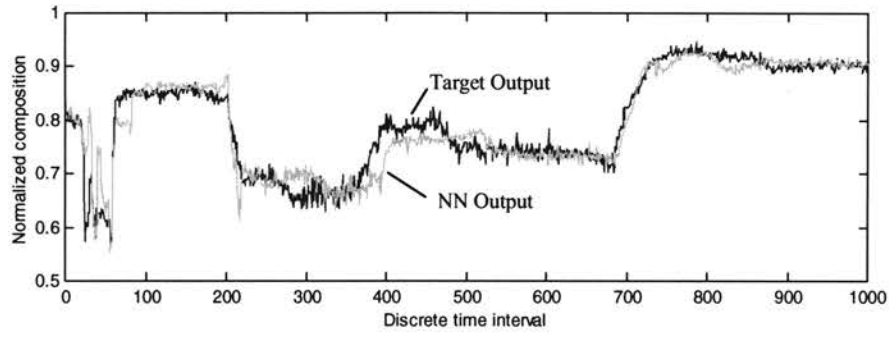
(RMSE: Root Mean Square Error; MAE: Maximum Absolute Error; MRE: Maximum Relative Error.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2}; MAE = \max(|\hat{y}_i - y_i|, i = 1, \dots, N); MRE = \max\left(\left|\frac{\hat{y}_i - y_i}{y_i}\right|, i = 1, \dots, N\right)$$

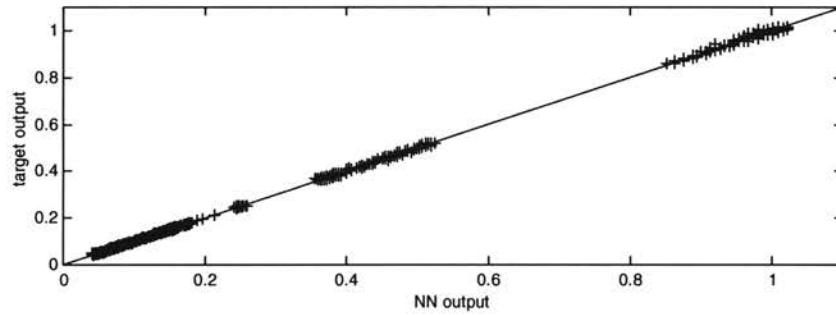
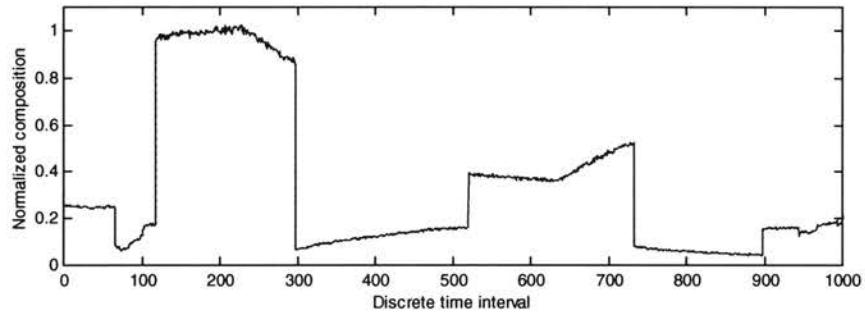
Where \hat{y}_i is the output from the NN, y_i is the target output, and N is the number of patterns)

Top Comp. Model (training patterns=7582)	NN₁	NN₂	NN₃	NN₄	NN₅	NN₆	NN₇
NN Structure	14-8-1	16-8-1	18-8-1	20-8-1	22-8-1	24-8-1	26-8-1
RMSE	0.0134	0.0210	0.0305	0.0293	0.0248	0.0307	0.0277
MAE	0.1439	0.2376	0.2408	0.2181	0.2580	0.2281	0.2341
MRE	26.17%	89.62%	60.93%	51.73%	92.70%	42.48%	87.98%
Bottom Comp. Model (training patterns=2505)	NN₁	NN₃	NN₅	NN₇	NN₁₀		
NN Structure	24-8-1	36-8-1	48-8-1	60-8-1	78-8-1		
RMSE	3.58e-3	3.52e-3	3.19e-3	3.12e-3	5.33e-3		
MAE	0.0232	0.0240	0.0228	0.0233	0.0291		
MRE	17.81%	11.79%	17.18%	12.92%	58.51%		

Figure 6.6 shows part of the training results, the training results for NN₅ of the top composition and NN₅ of the bottom composition (again, because the sample interval for the top composition is 10 minutes, NN₅ of the top composition predicts 50 minutes ahead. Because the sample interval for the bottom composition is 30 minutes, NN₅ of the bottom composition predicts 150 minutes ahead). The performance of the NN₅ for the top composition on 1000 patterns out of the 7582 patterns in the training set are shown in Figure 6.6(a). The performance of the NN₅ for the bottom composition on 1000 patterns



(a) Top composition (NN5)



(b) Bottom composition (NN5)

Figure 6.6. Training performance of GNN

out of the 2505 patterns in the training set are shown in Figure 6.6(b). Both the time series comparison and target value deviation plot show that the NN model outputs are close enough to the target outputs and thereby indicates good training performance. It can also be seen that the bottom dynamics and changes are almost perfectly tracked throughout the entire range, while the dynamics of the top composition is tracked well with constant biased periods.

A NN model was also obtained for the column pressure drop, ΔP , which is used as a state variable constraint in the GNNMPC performance studies in Section 6.6. The model uses the past ΔP (10 minutes ago), the current ΔP , the current feed flowrate, and the next step MVs to predict ΔP 10 minutes into the future. The structure of the NN is 5-10-1. The number of the training patterns for ΔP is 8634. Again, all variables are normalized linearly into a range of 0 to 1 within their operating range as listed in Table 6.1. The range for normalization of ΔP is 0 to 10 inH₂O. The performance of the ΔP model on 1000 patterns out of the 8634 patterns in the training set are shown in Figure 6.7, with both time series comparison and target value deviation plot. Though deviation can be large from points to points, the NN tracks the dynamics of ΔP very well.

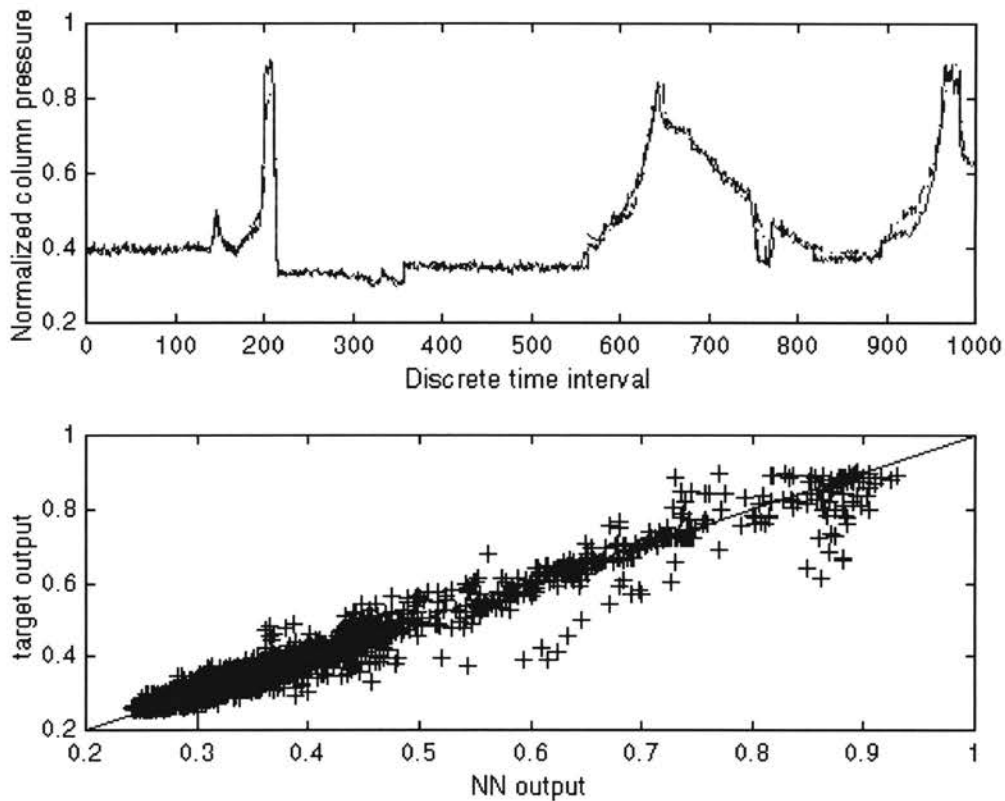


Figure 6.7 Training performance for the column pressure (10 minute ahead)

6.4 “Goodness” of GNN

Validation of the GNN model for the experimental distillation process was carried out by comparing the step response from the experimental run with the predicted step response from the GNN model. The three open loop response case studies in Table 6.2 were used for comparison.

For all three cases, the process was originally at steady state, the GNN model made prediction at the instant when a step change of the inputs was made. The adjusted prediction was also calculated according to the PMM obtained when at the original steady state. The prediction and the adjusted prediction were then compared with the

actual process response. Figures 6.8 to 6.10 show the comparison results. In the figures, the top composition was predicted at discrete time interval (with a sampling interval of 10 minutes) 1, 2, 3, 4, 5, 6, 7. The bottom composition was predicted at discrete time interval (with a sampling interval of 30 minutes) 1, 3, 5, 7, 10. Prediction point “0” in the figures indicates the time instant when step change of the process input was made.

For all three cases, it is observed that the adjusted prediction did not sufficiently compensate the PMM and bring the GNN output closer to the actual responses. This implies that the PMM obtained at steady state could not compensate the PMM during the transient period. Values of PMM are plotted in Figure 6.11. It is seen that the magnitude of the PMM for the top composition is of 10^{-3} mole fraction, the same magnitude of the noise level for the top composition. Therefore, it is reasonable draw the conclusion that the top composition prediction at steady state is good. The PMM for the bottom composition is of 10^{-2} mole fraction, much larger than the noise level (10^{-3}) for the bottom composition. This implies that systematic mismatch exists between the GNN model and the process. While from Table 6.3, it seems that the bottom composition has a better training performance than the top composition, observation here leads to an opposite conclusion. This implies that the bottom composition may not have enough training patterns to be fully trained (2505 training patterns for the bottom composition in comparison to 7582 training patterns for the top composition).

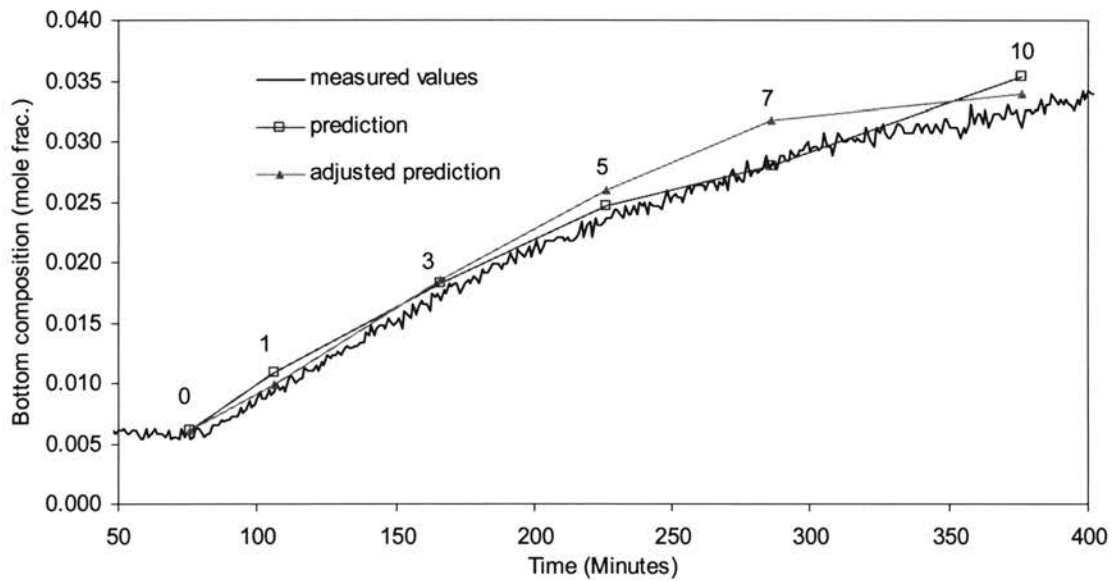
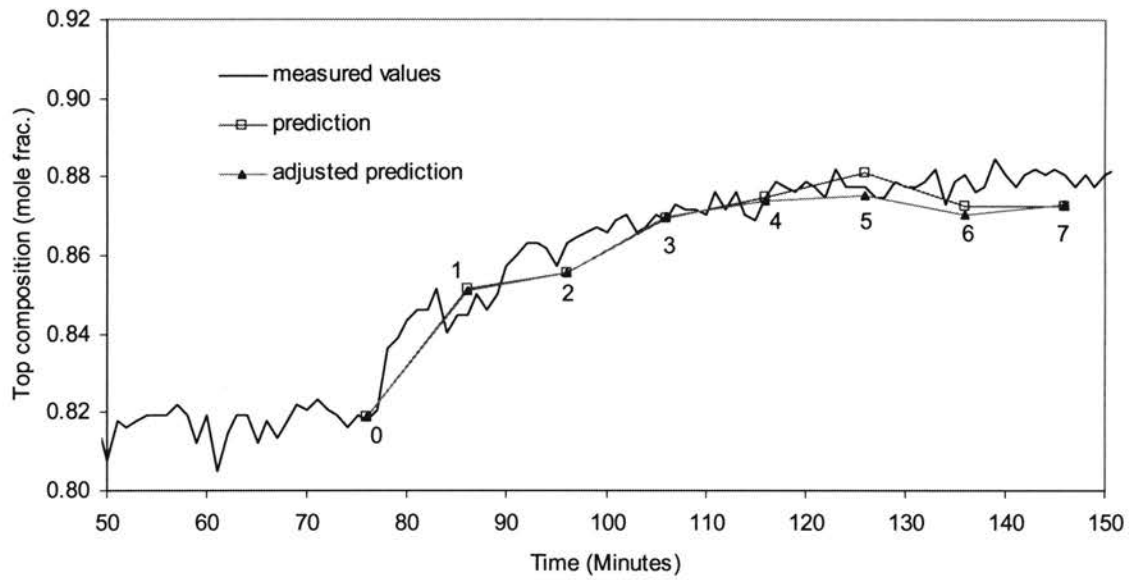


Figure 6.8 Comparison of GNN prediction with the running results for Case 1

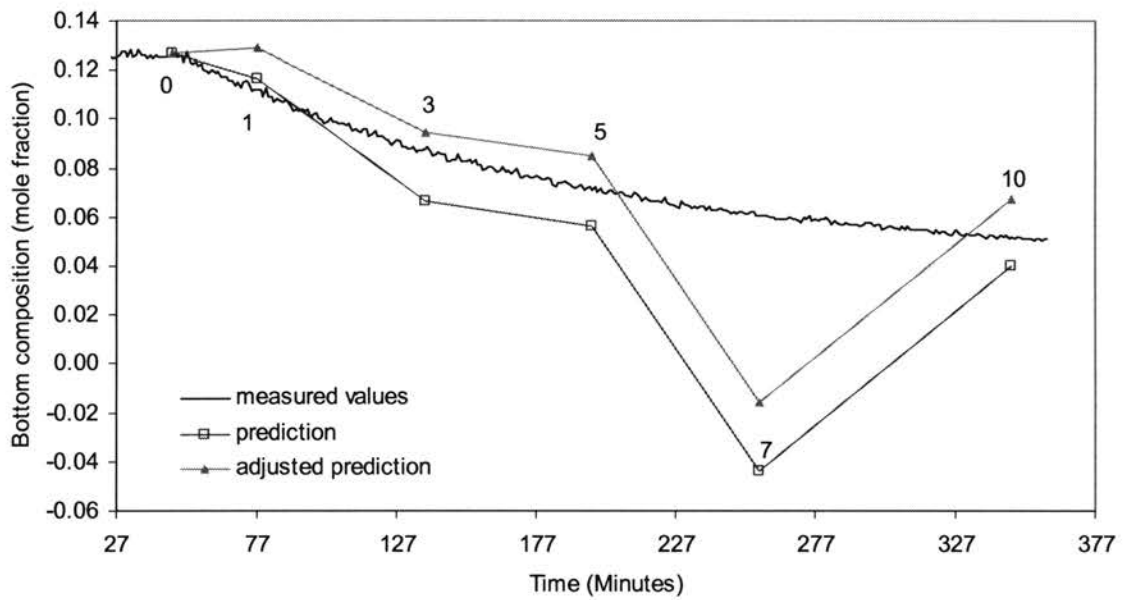
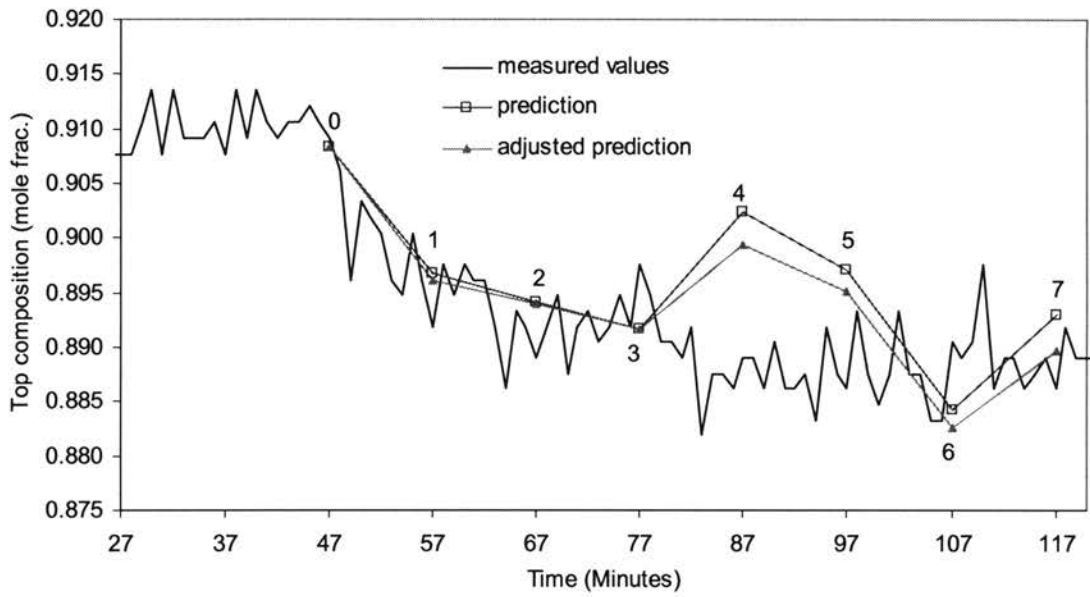


Figure 6.9 Comparison of GNN prediction with the running results for Case 2

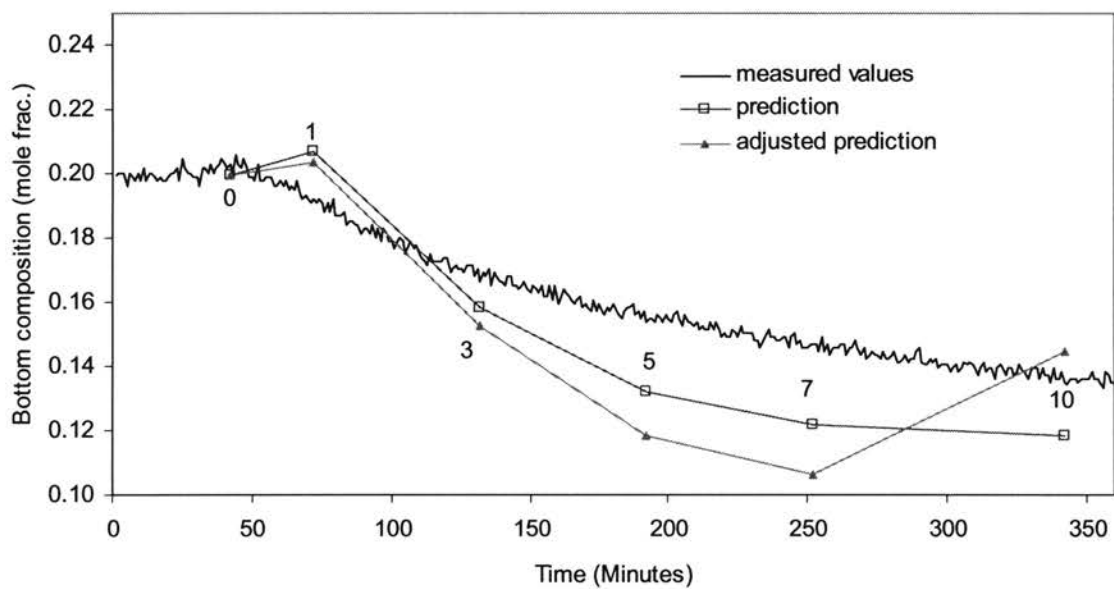
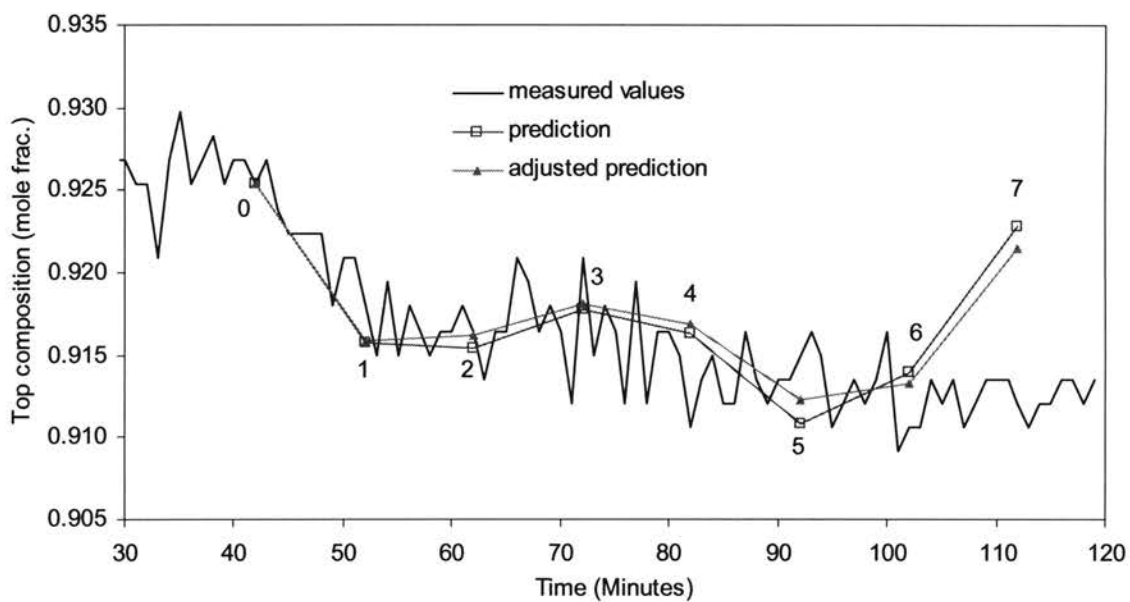


Figure 6.10 Comparison of GNN prediction with the running results for Case 3

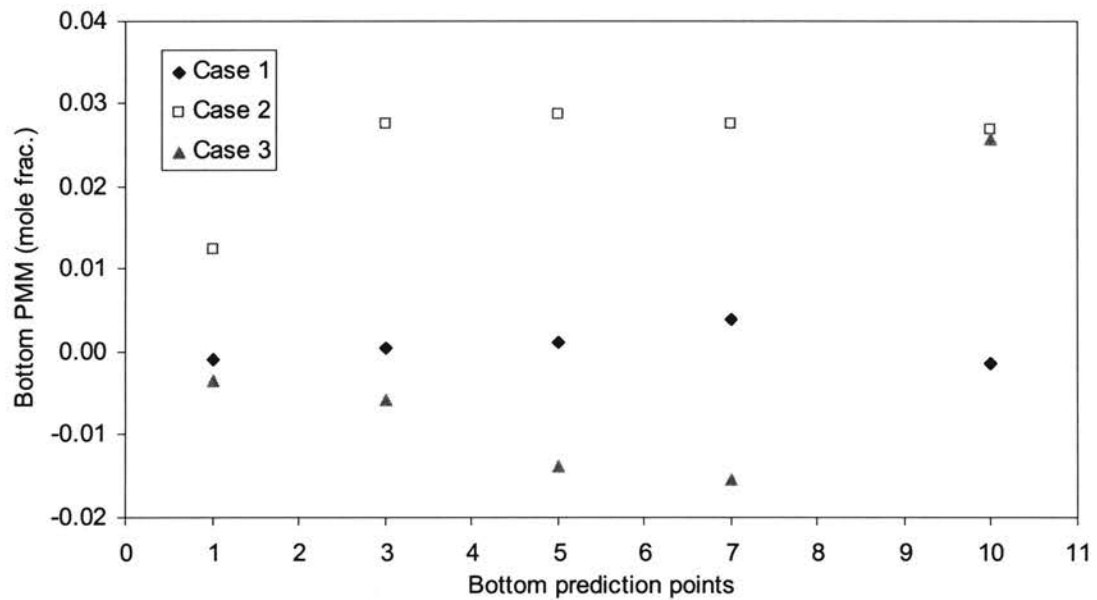
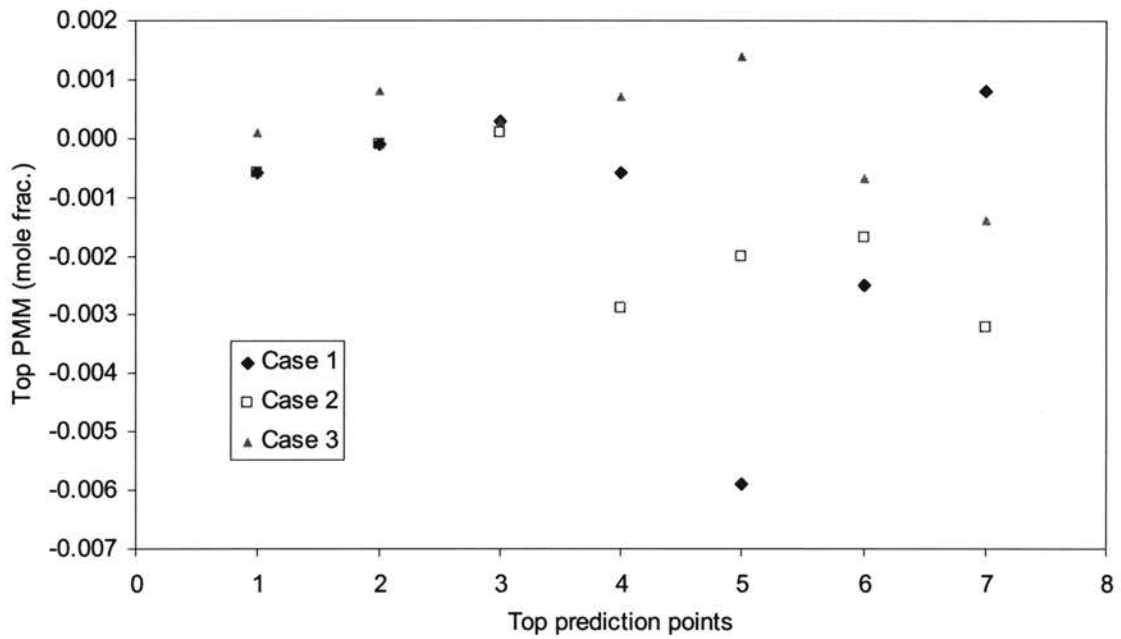


Figure 6.11 Steady state PMM for cases listed in Table 6.2

As shown in Figure 6.8, for Case 1, the line that connects the predicted points obtained from the GNN model shows that the predicted dynamics is very close to that of the actual response for both good prediction for the dynamics of both the top composition and the bottom composition.

Figure 6.9 shows the comparison for Case 2, the deviation of prediction points 4 and 5 for the top composition is about 0.015 mole fraction. Considering that the magnitude of steady state change for the top composition is about 0.025 mole fraction (from 0.910 mole fraction to 0.885 mole fraction), the prediction points 4 and 5 did not represent the real process well. For the bottom composition, while prediction points 1, 3, 5, 10 are all well modeled, the GNN model provides a negative value for the prediction point 7, which is unacceptable. This indicates that the NN for prediction point 7 was not trained well in this operating point.

Figure 6.10 shows the comparison for Case 3, the prediction point 7 of the top composition deviate from the actual value by about 0.05 mole fraction, a “bad” prediction. The bottom composition prediction essential follows the process dynamics.

The three step response case studies do not justify a claim of good modeling. However, it can be claimed that the model follows the “trend” of the process fairly well.

6.5 Configuration of GNNMPC

The GNNMPC is formed as a nonlinear constrained optimization problem, which is solved using the available optimization algorithm in the Matlab[®] toolbox. The optimization problem is:

$$\begin{aligned} \text{Min}_{R_f, H_f} = & w_y \sum_{i \in y_p} (\tilde{y}(k+i) - y_{sp})^2 + w_x \sum_{i \in x_p} (\tilde{x}(k+i) - x_{sp})^2 + \\ & w_{MV} (w_R \sum_{j \in R_p} \Delta R(k+j)^2 + w_H \sum_{j \in H_p} \Delta H(k+j)^2) \end{aligned}$$

subject to:

$$\tilde{y}(k+i) = \text{GNN}(y(k-5), \dots, y(k), F(k), R(k-5), \dots, R(k+i-1), H(k-5), \dots, H(k+i-1)) + dy(k+i)$$

$$dy(k+i) = y(k) - \text{GNN}(y(k-5-i), \dots, y(k-i), F(k-i), R(k-5-i), \dots, R(k-1), H(k-5-i), \dots, H(k-1))$$

$$\tilde{x}(k+i) = \text{GNN}(x(k-5), \dots, x(k), F(k), R(k-5), \dots, R(k+i-1), H(k-5), \dots, H(k+i-1)) + dx(k+i)$$

$$dx(k+i) = x(k) - \text{GNN}(x(k-5-i), \dots, x(k-i), F(k-i), R(k-5-i), \dots, R(k-1), H(k-5-i), \dots, H(k-1))$$

$$\Delta P(k+1) = \text{GNN}(\Delta P(k-1), \Delta P(k), F(k), R(k), H(k)) + d_{-} \Delta P(k+1)$$

$$d_{-} \Delta P(k+1) = \Delta P(k) - \text{GNN}(\Delta P(k-2), \Delta P(k-1), F(k-1), R(k-1), H(k-1))$$

$$\Delta R(k+j) = R(k+j) - R(k+j-1)$$

$$\Delta H(k+j) = H(k+j) - H(k+j-1)$$

$$0 \leq \tilde{y}(k+i) \leq 1$$

$$0 \leq \tilde{x}(k+i) \leq 1$$

$$0 \leq R(k+j) \leq 1$$

$$0 \leq H(k+j) \leq 1$$

$$\Delta P(k+1) \leq \Delta P_{\max}$$

(6.1)

Equation set (6.1) includes all three components in the MPC scheme, the objective function for constrained optimization, the predictive model as expressed by the GNN model, and the model adjustment part as expressed by the $dy(k+i)$, $dx(k+i)$, and

$d_{\Delta P}(k+1)$ items. Though using the same nomenclature as before, all variables are normalized variables in Equation set (6.1).

Table 6.4 Tuning parameters of GNNMPC for the experimental distillation process
(1: sample interval is 10 minutes; 2: sample interval is 30 minutes)

w_y	w_x	w_{MV}	w_R	w_H	y_P^1	x_P^2	R_P^1	H_P^1
1.5	1	0.02	1	3.5	{1,2,3,4,5,6,7}	{1,3,5,7,10}	{0,4}	{0,4}

Selected values of the tuning parameters in Equation set (6.1) for the experimental runs are listed in Table 6.4. Beyond the MV movement suppression factor w_{MV} , there are 4 other tuning weights in the objective function, w_y , w_x , w_R , and w_H . w_y and w_x are weights for the deviations of the CVs from their setpoints, w_R and w_H are weights for suppressing the MV movements. These weights are added so that mole fraction change of the top and the bottom compositions are treated equally important by the optimizer, and that the optimizer will not move one MV more than the other. w_y and w_x are chosen so that 1 mole fraction change of the top composition equally important to 1 mole fraction change of the bottom composition. Because the range of the top composition is 0.6 to 0.95 and that of the bottom composition is 0 to 0.2, the normalized top composition change is $\frac{1}{(0.95-0.6)}$, and the normalized bottom composition change is $\frac{1}{(0.2-0)}$. The

weighted normalized value should be equal to make the optimizer treat them equally, i.e.

$$w_y \frac{1}{(0.95-0.6)} = w_x \frac{1}{(0.2-0)}, \text{ therefore, } w_y : w_x = 1.75. \text{ Similar design routine can be}$$

applied to the weights w_R and w_H for the MVs. It is observed from the open loop

dynamics that a change of 10 gmol/hr in the reflux flowrate leads to about the same amount of change in the CVs as a change of 1% full power in heating power. Therefore, to make the controller treat the MVs equally, we let $w_R \frac{10}{(150-90)} = w_H \frac{1}{(60-40)}$, which results in $w_H : w_R = 10 : 3$. w_{MV} is the MV movement suppression factor, which penalized the MV movement for smooth dynamics and adjusting the aggressiveness of the controller. y_p and x_p are the prediction points for the top composition and the bottom composition, where predictions from the GNN model are available. R_p and H_p are the future points where the future MV movement are made. R_f and H_f are the optimized future MVs which make change at R_p and H_p respectively.

6.6 Control Performance of GNNMPC

Table 6.5 lists the experimental runs discussed in this section.

Table 6.5 Experimental demonstration cases for GNNMPC

Case Number	Control Mode	w_{MV}	Changes Made
1	Servo	0.1	From manual mode to auto mode with $\{y_{sp}, x_{sp}\} = \{0.9, 0.035\}$
2	Servo	0.01	From manual mode to auto mode with $\{y_{sp}, x_{sp}\} = \{0.9, 0.050\}$
3	Regulatory	0.02	$\{y_{sp}, x_{sp}\} = \{0.925, 0.0235\}$, feed flowrate change from 260 gmol/hr to 300 gmol/hr
4	MV constraint	0.02	$\{y_{sp}, x_{sp}\}$ change from $\{0.913, 0.045\}$ to $\{0.92, 0.127\}$ and then change back
5	State Constraint	0.02	Maximum column pressure changes from 4.5 inH ₂ O to 3 inH ₂ O and then changes back

Figure 6.12 shows the GNNMPC performance for setpoint tracking with a sluggishly tuned controller. The MV movement suppression factor, w_{MV} is set to be 0.1,

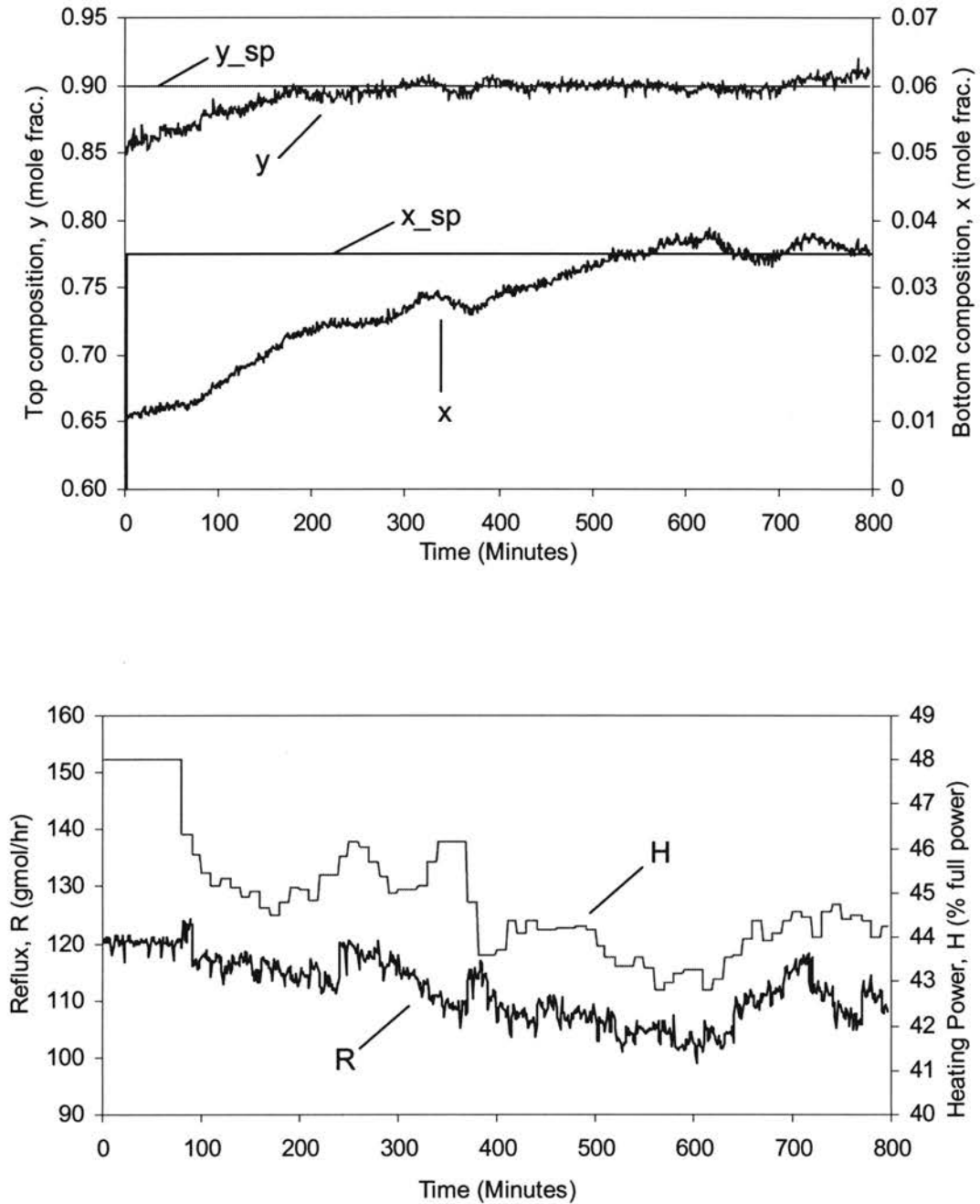


Figure 6.12 GNNMPC performance for setpoint tracking with sluggish tuning parameter (Case 1)

5 times of the nominal value of 0.02. The process was at first under open loop run and then transferred to auto mode at 80 minutes, while open loop steady state had not been reached. It took the top composition about 100 minutes to first hit its setpoint at about 180 minutes. And it took the bottom composition about 340 minutes to first hit its setpoint at about 520 minutes. Both took longer than their settling times (about 70 minutes for the top composition and about 300 minutes for the bottom composition). It can also be seen that the MVs progressed slowly to the settled values of about 105 mol/hr (the reflux flowrate) and 44% full power (the heating power).

Figure 6.13 shows the GNNMPC performance for setpoint tracking with a movement suppression factor of 0.01 (0.5 times of the nominal value of 0.02). The process was at first under open loop run at steady state and then transferred to auto mode at 67 minutes. It took the top composition about 100 minutes to first hit its setpoint at about 170 minutes. It took the bottom composition about 70 minutes to first hit its setpoint at about 140 minutes, then oscillate with a decreased magnitude of overshoot and was about to settle down at its setpoint when the run ended. much shorter than its settling time, but overshoot resulted. Comparing to the MV movements in Figure 6.12 (Case 1), the MV movements are much more aggressive. This demonstration and comparison shows that the MV movement suppression factor is an important tuning parameter of GNNMPC, as for other MPC strategies, which determines the aggressiveness of the controller.

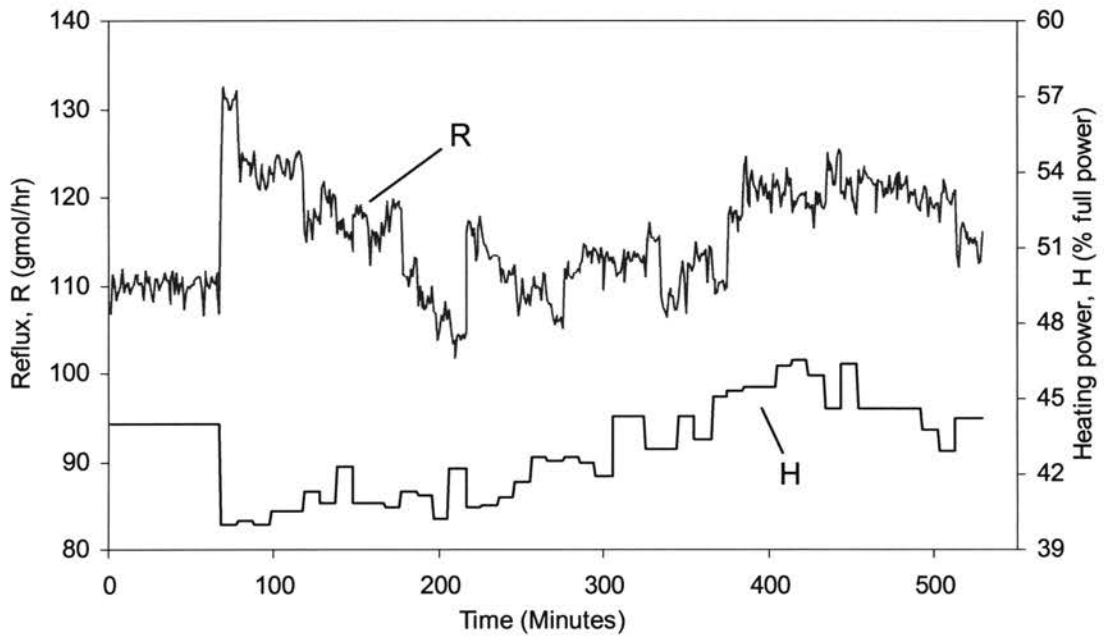
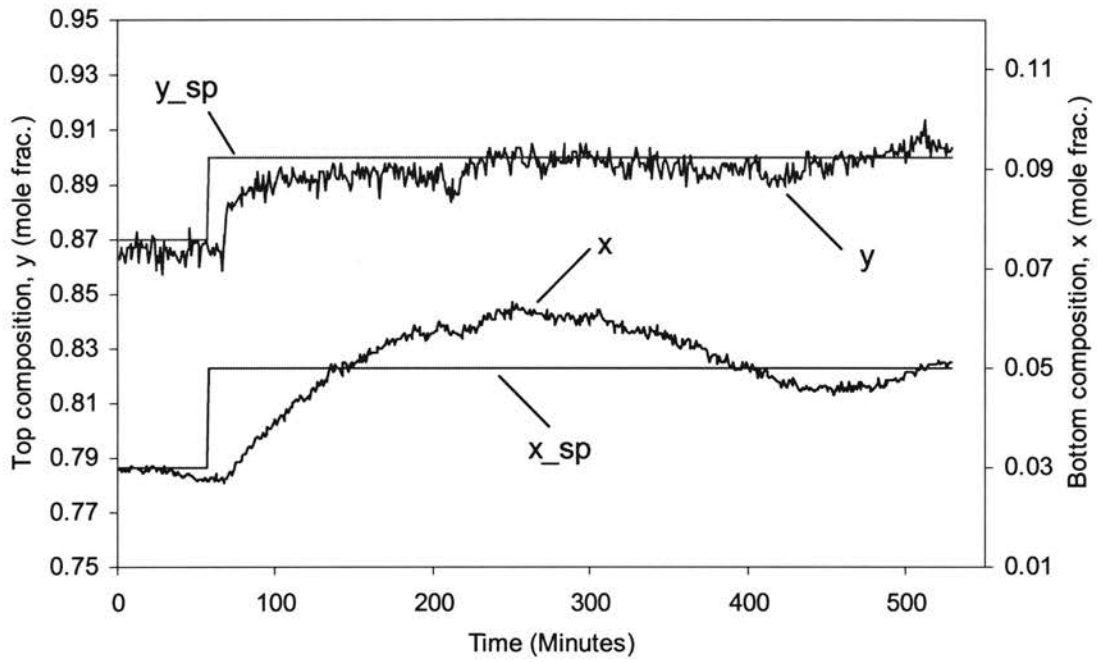


Figure 6.13 GNNMPC performance for setpoint tracking with aggressive tuning parameter (Case 2)

Figure 6.14 shows the GNNMPC's robustness to measured, feedforward load disturbance. The process was first at steady state under manual mode. At 100 minutes, the GNNMPC was transferred to automatic mode. At about 130 minutes, the feed flowrate was changed from 260 gmol/hr to 300 gmol/hr, which led to immediate drop of the reflux flowrate to compensate for the effect of load disturbance. Both the top and the bottom compositions were successfully regulated at their setpoints with a little deviation of the bottom composition from its setpoint at the beginning when the disturbance was introduced.

Figure 6.15 shows an open loop run which demonstrates the impact of the feed flowrate on the process. The process was first at steady state with the top composition at 0.93 mole fraction and the bottom composition at 0.028 mole fraction. The feed flowrate was changed from 260 gmol/hr to 300 gmol/hr at 240 minutes, which led to an increase of the top composition to 0.95 mole fraction and the bottom composition to 0.06 mole fraction when the process settle down. Figure 6.15 shows that the load disturbance has a significant effect on the process, and that the control action of Figure 6.14 had a significant effect.

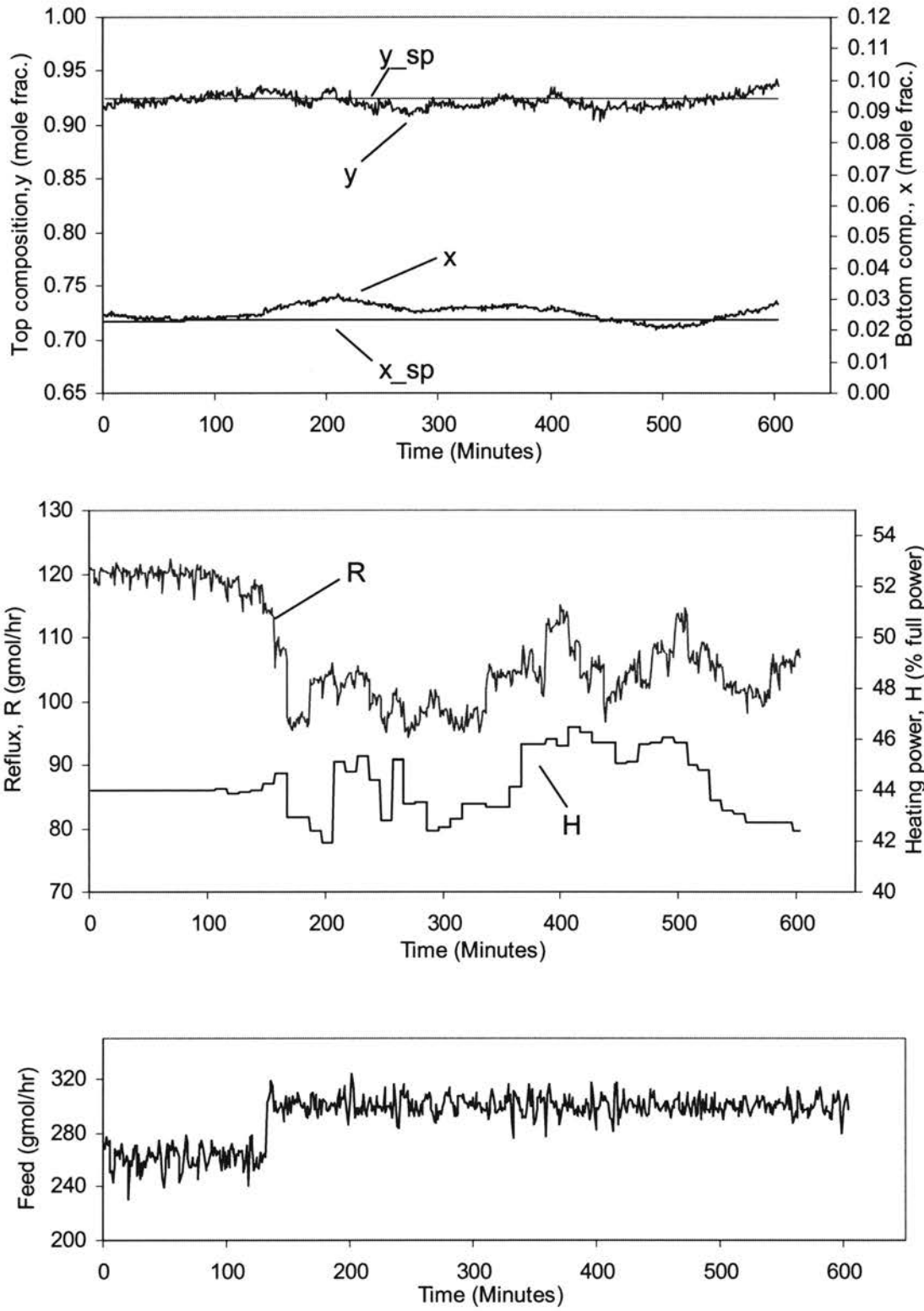


Figure 6.14 GNNMPC performance for disturbance rejection (Case 3)

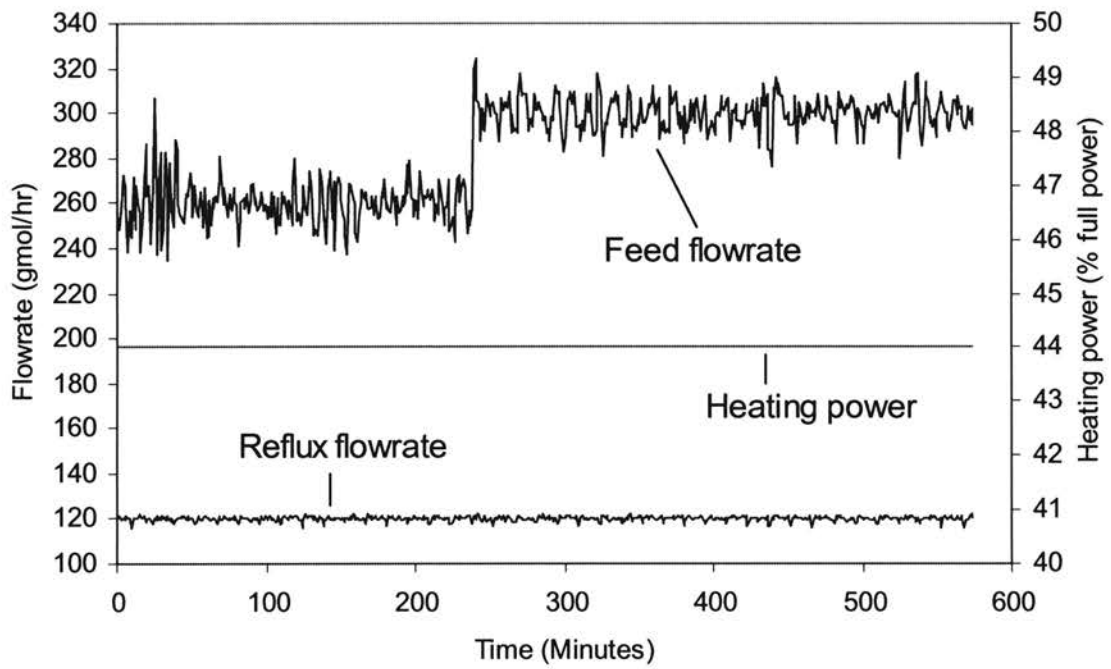
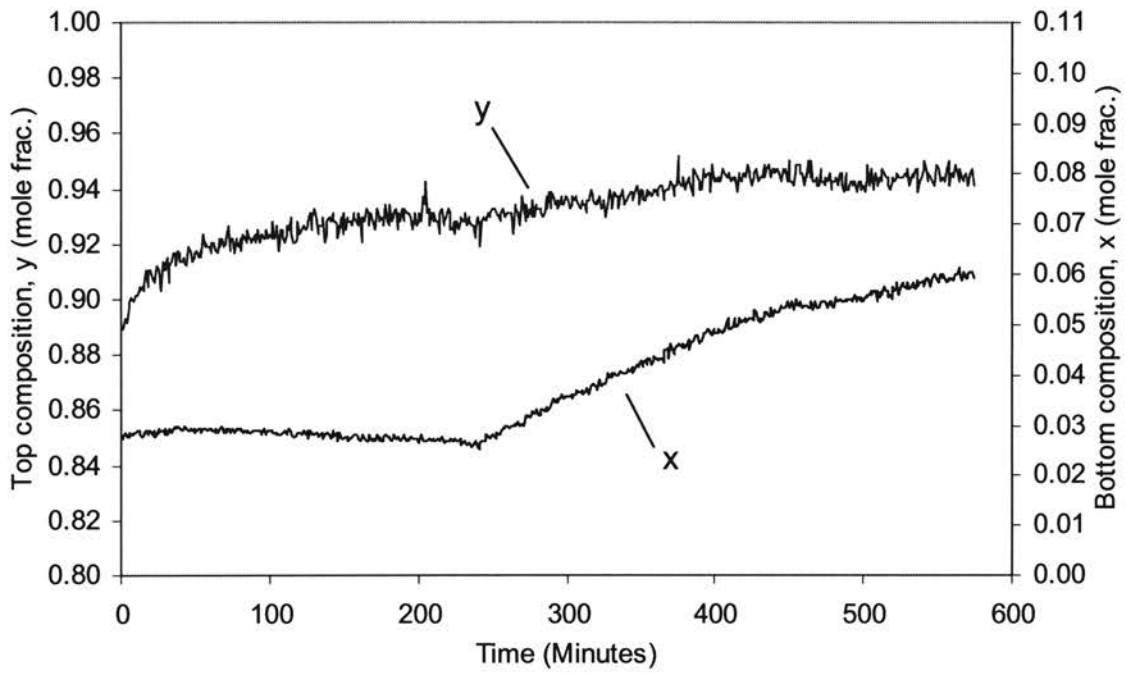


Figure 6.15 Open loop response to feed flowrate change

Figure 6.16 shows the GNNMPC's control performance when MV constraints were hit. The process was first open loop under constant process inputs. At about 240 minutes, when the top composition and the bottom composition are near steady state, The control loop is closed under GNNMPC. When the setpoints were changed to {0.92, 0.127} at about 320 minutes, the controller pushed the reflux flowrate to its upper limit (150 gmol/hr) and the heating power near its lower limit (45% for this case) trying to drive the CVs to their setpoints. However, within the operating range, the CVs could not meet the setpoints. The reflux flowrate stayed at its upper limit (the hard constraint on the reflux flowrate). When the setpoints changed back to the original ones ({0.913, 0.045}) at 655 minutes, the controller immediately relieved the MVs from their constraints and brought the CVs to the setpoints in desired time.

Figure 6.17 demonstrates successful handling of state variable constraint of GNNMPC. The process was at first at closed-loop steady state under GNNMPC. When the hard constraint on the column differential pressure, ΔP , was changed from an upper limit of 4.5 inH₂O to 3 inH₂O at 67 minutes, the constraint was hit and the controller sacrificed the performance of the CVs to meet the constraint. When the upper limit of the ΔP was changed back to 4.5 inH₂O at 176 minutes, the controller was relieved from the constraint and was able to bring the CVs back to their setpoints.

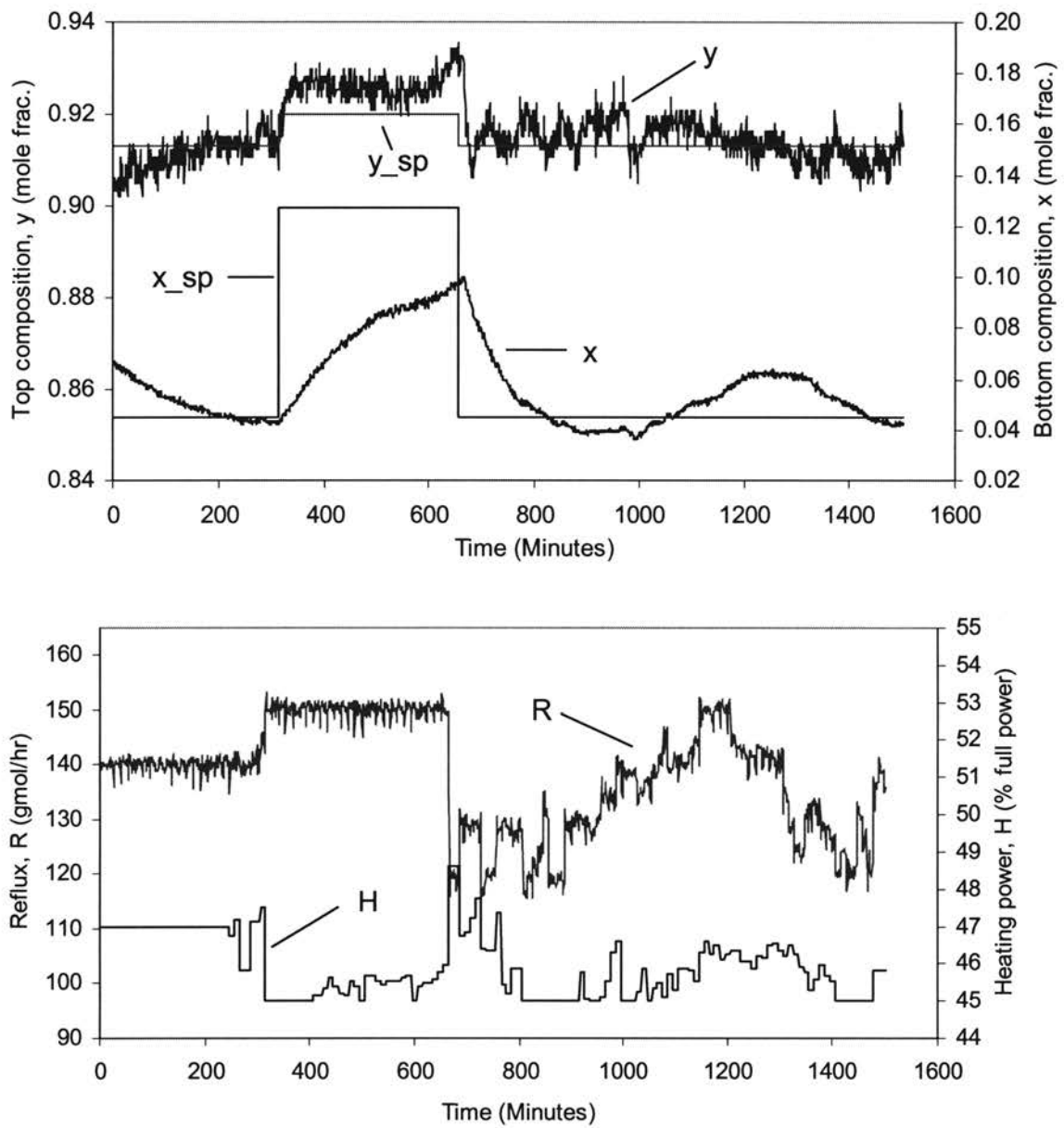


Figure 6.16 GNNMPC performance for handling MV constraint (Case 4)

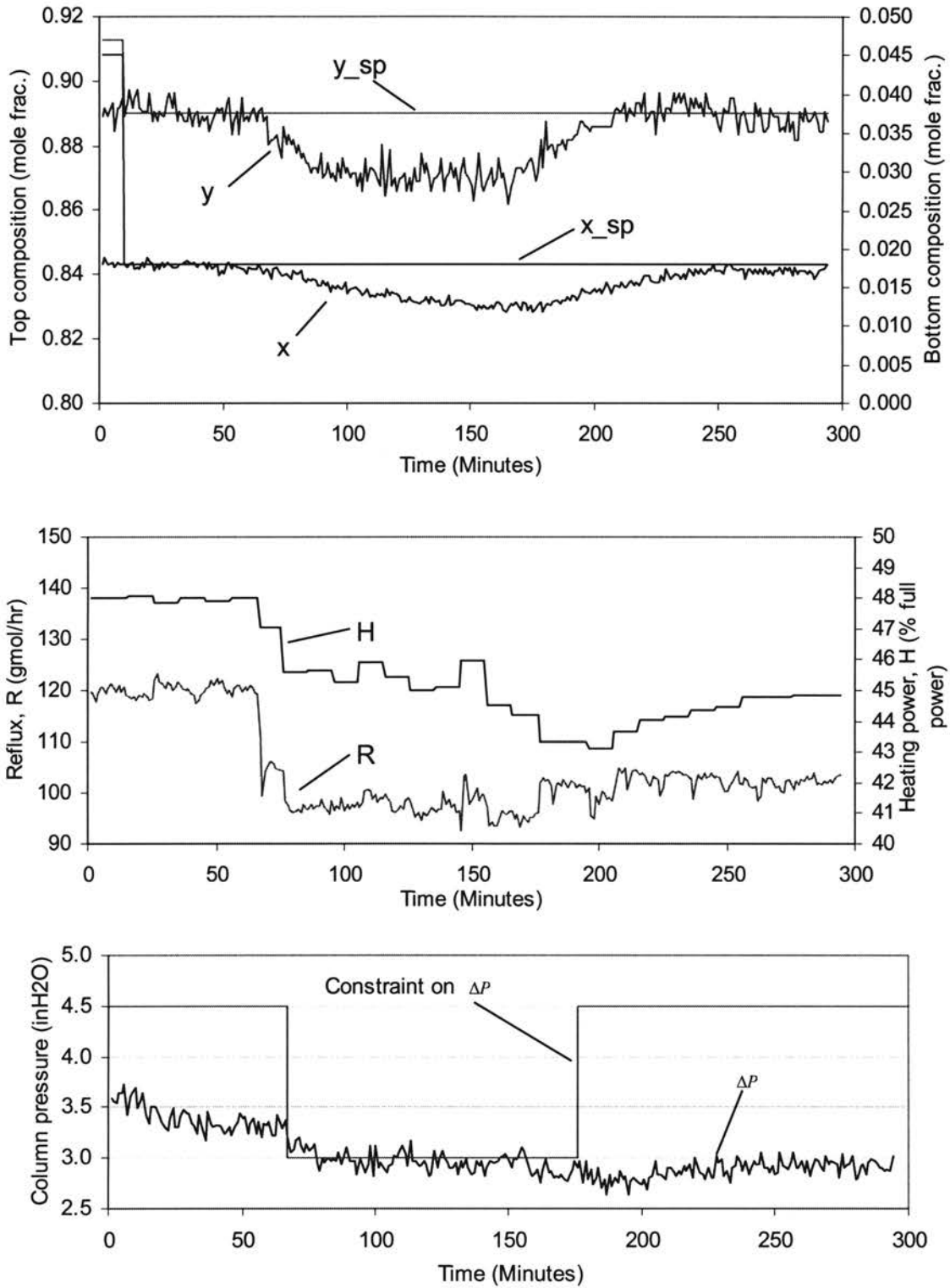


Figure 6.17 GNNMPC performance for state variable constraint handling (Case 5)

CHAPTER 7

SUMMARY AND CONCLUSIONS

7.1 Summary

In this work, a nonlinear model predictive control strategy based on neural network modeling technique which is called GNNMPC was proposed. The intention of GNNMPC is to develop a general structure for predictive modeling while relieving as much as possible the computational burden due to the model complexity. Effectiveness of GNNMPC was demonstrated by both simulations and experiments on a methanol-water distillation process.

Based on a tray-to-tray dynamic model (developed by Pandit (1991) and coded in C language by Dutta (1997)), a tray-to-tray dynamic model for the experimental methanol-water distillation unit in School of Chemical Engineering at Oklahoma State University was developed and coded in Matlab language.

GNNMPC was then tested using the simulated distillation process. Open-loop responses were first studied to investigate the characteristics of the process and determine the structure of the model. The process has a local gain change of about 2:1 ratio, and when operated in a wide operating range, the gain can change by a ratio of over 10 times in different regions, the settling time of the process outputs can change by a factor of 3. Index of the process interactions can range from about -5 in one region to about 8 in another region. The Grouped Neural Network (GNN) model was then obtained from pre-designed random input-output series. Afterwards, GNNMPC was implemented with modeling error considered. GNNMPC showed effectiveness in setpoint tracking,

disturbance rejection, handling interactions, handling hard constraints on the controller outputs, handling hard constraints on the control action movement, and easy tuning. The simulation results were presented in Chapter 5.

To set up the experimental apparatus for testing GNNMPC, the Technovate[®] Model 9079 fractional distillation system was converted to a fully automated system using a Camile[®] 2200 Data Acquisition and Control (DAC) system. For implementing GNNMPC, Camile[®] was configured to interface with the distillation process and the operator, Matlab[®] was configured to carry out all of the GNNMPC calculations. Camile[®] and Matlab[®] interface by an intermediate text file.

The structure of GNNMPC was determined from open loop step response testing. GNN was then trained from experimental data. With the presence of noise and drift, intentional and unintentional disturbances, process model mismatch, and operating constraints, five experimental runs shown here demonstrated the effectiveness of GNNMPC in tuning, tracking setpoint, rejecting disturbances, handling MV constraint, and handling state variable constraint on the experimental apparatus.

7.2 Conclusions

A novel modeling approach for general nonlinear MPC structure, called Grouped Neural Networks (GNN) model is proposed. The model uses a group of Neural Networks (NN) in a parallel structure to provide prediction values of the process outputs at subjectively chosen future time intervals within the prediction horizon as representation of the future process dynamics. During the implementation of GNNMPC, the number of decision variables for the optimization problem, i.e., the future control actions, is

significantly decreased by only making future control action change at subjectively selected control time intervals within the control horizon. Under this infrastructure, the computation burden for the general nonlinear optimization algorithm is decreased by the decreased number of future points to be calculated, the algebraic calculation of the prediction values, and the decreased number of the decision variables.

The proposed GNNMPC was demonstrated by both simulations and experiments to be capable of controlling a binary methanol-water distillation process, which has such control-challenging characteristics as modeling error, severe interactions, static and dynamic nonlinearity, measured and unmeasured disturbances, constraints on the process inputs as well as the operating conditions. GNNMPC has shown to be very effective in setpoint tracking, disturbance rejection, and constraint handling.

CHAPTER 8

DISCUSSION AND RECOMMENDATIONS

This work extensively explored and investigated the effectiveness of GNNMPC in simulation tests and demonstrated the effectiveness of GNNMPC on a real-world process (the distillation apparatus). Meanwhile, many new areas for further research were opened. In this chapter, discussions on some issues pertaining this work are presented, which would lead to the author's opinions on future work that can be done to make the investigation of GNNMPC more comprehensive and convincing.

Nonlinear model predictive control (NMPC) is more a practicing art than a science field. While theoretical results are by far very limited for NMPC, practitioners are eager to explore it due to the great success of its linear counterpart, linear model predictive control (LMPC), a great potential of benefits brought by NMPC, and the straight-forward approach of implementing NMPC. GNNMPC, as one of the numerous NMPC schemes, was proposed with the intention to relieve the computation complexity in a general NMPC scheme. With little theoretical support, heuristics, experiences of other practitioners were taken as good reference and extensive simulation tests were used to demonstrate its effectiveness as a general NMPC scheme. Experimental tests, as a significant step for deep investigation of the control strategy as well as for establishing credibility within the practice community, were also carried out. The work done in this thesis project built the framework of GNNMPC.

Some issues in GNNMPC are common issues encountered by the same type of NMPC, such as the choice of historical data length, generation of stimulating input

signals, optimization of the neural network structure, choice of the predictive horizon, control horizon and other tuning parameters, and solving the nonlinear optimization problem. Future researchers on GNNMPC are suggested to keep an eye on the advances of such issues in the open literature and be ready to take advantage of the advances.

There are some issues specific with GNNMPC that need further investigation.

One of the distinguished features of GNNMPC is to predict selected discontinuous (in discrete time domain) points, assuming that if well selected, these prediction points can well represent the “trend” of the process. The motivation of this strategy is to reduce the computation burden for GNNMPC, and it has been shown to work effectively throughout this work. However, the process that was studied (the distillation process) is a first-orderish process, or so called well-behaved process. While typical in manufacturing industry, there are many processes that are ill-behaved, such as a reactor with inverse dynamics. In principle, it is believed that as long as the selected points can represent the dynamics, GNNMPC should work. For an ill-behaved process, some interesting issues may come up that worth investigation. Therefore, it is recommended to explore GNNMPC on processes with ill-behaved dynamics.

GNNMPC uses separately-trained neural networks to get the predictive model, therefore, inconsistency of each NN output in GNN exists, which would cause steady state offset if the conventional DMC feedback strategy of adding the most current one step prediction error to all the future predictions was applied. To deal with this, a corresponding PMM adjustment strategy was proposed and shown effective in eliminating the steady state offset. The strategy is to compensate PMM of each NN separately applying currently and historically available process measurements. While not

found troublesome during this work, it might have caused sluggish or even misleading feedback in some cases. According to the feedback strategy of GNNMPC, a prediction l steps ahead would be adjusted by a value calculated based on the data l steps ago. If the process is operated in a distinctively different operating regime at l steps ahead from that at l steps ago, this additive strategy may be misleading, especially for a nonlinear process. Further, the more into the future predicted, the more delay in its corresponding feedback, which might make it more sluggish to disturbances than the conventional feedback strategy. This feedback issue is suggested to be investigated by comparing GNNMPC with conventional MPC such as DMC. It is also recommended that other strategies, either novel ones, or those adapted from literature, be explored and compared to the present GNNMPC feedback strategy. A crude idea of the author, as a recommendation to improve the feedback strategy of GNNMPC, is to apply the conventional feedback strategy (adding the prediction error of one-step-ahead prediction to all the future predictions) while handling the discrepancy of outputs from GNN separately. Because the discrepancy mainly affects the steady state offset, only steady state outputs from GNN are needed to be investigated for designing a proper strategy to compensate the discrepancy between the GNN outputs.

Among all the intentions of developing a dynamic distillation simulator for the experimental distillation process as described in Chapter 5, one original intention was to use the simulator to get a GNN model for the experimental work. It would have been much more time-and-cost-effective to get training data from the simulator rather than from the experimental runs. Unfortunately, this approach was not realized. Instead, a GNN model obtained from the experimental run data was used. The reason was that the

simulator was found not representative of the experimental apparatus. Reasons of the modeling error are multiple. The model assumes that the dynamics of the heating element in the reboiler is negligible. This assumption can not hold for the experimental apparatus. The reboiler has a overflow volume of about 14.2 liters, which is the standard volume of the liquid in the reboiler during experiments. The volume is about 25 times of the reflux drum's overflow standard volume, and is about the same ratio to the tray size. In such case, the time delay of heating in the reboiler can not be neglected. Ambient heat loss and the internal reflux was not considered in the simulator while it is significant in the experiments. The tray efficiencies that were determined by visual observation were very rough, therefore the tray efficiencies used in the simulator can not be expected to be the real tray efficiencies. The top and the bottom compositions in the experiment were inferred from the temperatures measured in the reboiler and on the top tray using the temperature-composition correlations from ideal vapor-liquid-equilibrium (VLE) of pure methanol-water system at standard pressure (1atm). The temperature about 0.25 inch above the top tray sensed by a thermocouple was found to deviate nonlinearly with significant noise from the temperature that is correlated from the composition offline measured by the refractometer. The thermocouple that senses the temperature of the liquid in the reboiler is located near the heating element of the reboiler. Since there is large volume time for the reboiler, the temperature does not represent the temperature at the vapor-liquid-equilibrium (VLE) point. The pressure of the reboiler's vapor, as measured by the column pressure, is above atmosphere pressure by several inch water. Inaccuracy of the flow rates (feed, reflux, boilup) due to the correlation errors also contribute to the modeling error. If the simulator can be modified (based on the present

one or using a new modeling approach) to well represent the process, it is recommended that the GNNMPC control performance based on the GNN model obtained from simulator-generated data. Or, if it is practically difficult to have a simulator that can represent the distillation process well, other experimental processes which can be simulated well be considered.

It is recommended that measures be taken to reduce the volume of the reboiler to save the experimental running time. Due to the large volume of the reboiler, the settling time of the reboiler is about 300 minutes, which made the experimental runs very long. If measures can be taken to reduce the liquid volume of the reboiler, the experimental work would be more time efficient. Flooding, weeping, and entraining constraint of the column also prevent the column from being operated in a wider range to allow significant nonlinearity of the process to occur.

As shown in Chapter 5 by simulation results, the move suppression factor, w_{MV} , is an important tuning parameter of GNNMPC and affects the control performance. For the distillation process that is both of nonlinearity and interaction, a fixed value of w_{MV} is found to be not effective in bringing optimal control performance throughout the operating range. Explore the use of an adaptive approach on w_{MV} would be very interesting and beneficial work. Previous work on this issue, though mainly pertinent to linear MPC (Rani & Unbehauen, 1997; Shridhar & Cooper, 1998; Al-Ghazzawi, Ali, Nouh, and Zafiriou, 2001), provide a good starting point for expanding to nonlinear MPC tuning. Another alternative is to use CV damping for tuning instead of using move suppression tuning approach. A preliminary investigation on the comparison shows

certain advantages of using CV damping over move suppression. The investigation results are presented in Appendix E.

The simulation results of disturbance rejection performance as shown in Cases 5 to 8 in Chapter 5 worth further investigation. Though the controller is shown to be of good disturbance rejection performance, it was observed that the controller did no better in rejecting the feedforward disturbance (the feed flowrate) than in rejecting the unmeasured disturbance (the feed composition). This is contradictory to the experience with a linear controller that a feed forward model leads to a much better disturbance rejection performance when the model is “good”. Note that in the GNN model, only the most recent feedforward disturbance is fed into the GNN model. Investigation from another angle on this issue as described in Appendix E shows that if CV damping approach for tuning GNNMPC is used rather than the MV movement suppression tuning approach, this issue seems to disappear.

It is always desirable that a newly-developed strategy is compared to ad-hoc similar strategies. Comparison of GNNMPC with DMC, and even auto-tuned PID, both popular and successful in industry, would clarify the significant features of GNNMPC.

BIBLIOGRAPHY

- Agarwal, M. (1997), Systematic classification of neural-network-based control, *IEEE Control Systems Magazine*, 17(2), 75-93
- Ahn, S. M., Park, M. J., Rhee, H. K. (1999). Extended Kalman filter-based nonlinear model predictive control for a continuous MMA polymerization reactor, *Industrial and Engineering Chemistry Research*, 38(10): 3942-3949
- Al-Ghazzawi, A. , Ali, E. , Nouh, A. , and Zafiriou, E. (2001). On-line tuning strategy for model predictive controllers. *Journal of Process Control*. 11(3): 265-284
- Balasubramhanya, L.S. & Doyle III, F. J. (2000). Nonlinear model-based control of a batch reactive distillation column, *Journal of Process Control*, 10(2-3):209-218
- Banerjee, A. & Arkun, Y. (1998). Model predictive control of plant transitions using a new identification technique for interpolating nonlinear models, *Journal of Process Control*, 8(5-6): 441-457
- Bemporad, A., Ferrari-Trecate, G., Morari, M. (2000). Observability and controllability of piecewise affine and hybrid systems, *IEEE Transactions on Automatic Control*, 45(10): 1864-1876
- Bequett, B. W. (1991). Nonlinear control of chemical processes: a review. *Industrial & Engineering Chemistry Research*. 30: 1391-1413.
- Berenguel, M., Arahall, M. R., and Camacho, E. F. (1998). Modeling the free response of a solar plant for predictive control, *Control Engineering Practice*, 6(10): 1257-1266
- Bergounioux, M. & Kunisch, K.(2000). Active set strategy for constrained optimal control problems: the finite dimensional case. *Lecture notes in economics and mathematical systems*. 481: 36-54
- Bhat, N., McAvoy, T. J. (1990). Use of Neural Nets for Dynamic Modeling and control of chemical process systems, *Computers & Chemical Engineering*, 14(5):573-583
- Billings, S.A. and Voon, W.S.F.(1986). Aprediction-error and tepwise regression algorithm for nonlinear systems. *International Journal of Control*, 44:803-822

- Botto, M. A. & da Costa, J. S. (1998). A comparison of nonlinear predictive control techniques using neural network models, *Journal of Systems Architecture*, 44(8): 597-616
- Camacho, E. F. and Bordons, C. (1999) Model Predictive Control. Springer-Verlag London
- Clarke, D. W., Mohtadi, C., Tuffs, P.S. (1987a) . Generalized Predictive Control – Part I: the basic algorithm. *Automatica*, 23, 137-148
- Clarke, D. W., Mohtadi, C., Tuffs, P.S. (1987b) . Generalized Predictive Control – Part I: Extensions and interpretations. *Automatica*, 23, 149-160
- Cutler, C.R. & Ramaker B. L. (1979). Dynamic matrix control – a computer control algorithm. *AICHE 86th National Meeting*. Houston, TX.
- Daoping, H. & van Cauwenberghe, A. R. (1998). Neural-network-based multiple feedback long-range predictive control, *Neurocomputing*, 18(1-3): 127-139
- Delgado, A., & Kambhampati, C. (1995). Dynamic recurrent neural network for system identification and control. *IEE proceedings. Control theory and applications*. 142(4): 307-315.
- Dimopoulos, N. J. & Neville, S. (1995). Asymptotically Stable Recurrent Neural Networks and their Use: An Overview. *IEEE Mediterranean Symposium on New Directions in Control and Automation*. 1: 307-314.
- Dutta, P. (1997). Application of neural network control to distillation. PhD Dissertation, Texas Tech University, Lubbock, TX
- Fischer, M., Nelles O., and Isermann R. (1998). Adaptive predictive control of a heat exchanger based on a fuzzy model, *Control Engineering Practice*, 6(2): 259-269
- Froisy, J. B. (1994). Model Predictive Control: Past, Present and Future, *ISA Transactions*, 33, 235-243
- Galván, I. M. & Zaldívar, J. M. (1998). Application of recurrent neural networks in batch reactors Part II: Nonlinear inverse and predictive control of the heat transfer fluid temperature, *Chemical Engineering and Processing*, 37(2): 149-161
- Garcia, C.E., Prett, D.M., Morari, M. (1989). Model Predictive Control: Theory and Practice –A Survey. *Automatica*, 25, 335-348
- Henson, M. A. (1998). Nonlinear Model Predictive Control: Current Status and Future Directions. *Computers & Chemical Engineering*. 23, 187-202

- Hernandez, E., & Arkun, Y. (1992). A Study of the Control relevant Properties of Backpropagation neural Net Models of Nonlinear Dynamic Systems. *Computers & Chemical Engineering*, 16(4): 227-240
- Hussain, M. A. (1999). Review of the applications of neural networks in chemical process control – simulation and online implementation. *Artificial Intelligence in Engineering*, 13:55-68
- Kambhampati, C., Mason, J. D., and Warwick, K. (2000). A stable one-step-ahead predictive control of non-linear systems, *Automatica*, 36(4): 485-495
- Lee, J.H. (1996). Recent advances in Model Predictive Control and Other Related Areas, *Proceedings of the 5th International Conference on Chemical Process Control (CPC-V)*, 201-216
- Lee, J. H., Morari, M., Garcia, C.E. (1994) State Space Interpretation of Model Predictive Control. *Automatica*, 30(4), 707-717
- Lee, J. W., Kwon, W. H., Choi, J. (1998). On Stability of Constrained Receding Horizon Control with Finite Terminal Weighting Matrix. *Automatica*. 34 (12): 1607-1612
- Lee, P.L., Sullivan, G. (1988). Generic Model Control (GMC), *Computers & Chemical Engineering*, 12(6):573-580
- Lundström, P., Lee, J. H., Morari, M., and Skogestad, S. (1995). Limitations of dynamic matrix control, *Computers & Chemical Engineering*, 19(4): 409-421
- Mayne, D. Q., Rawlings, J. B., Rao, C. V. and Sokaert, P. O. M. (2000). Constrained model predictive control: Stability and optimality, *Automatica*, 36(6): 789-814
- Morari, M. & Lee, J. H. (1999). Model Predictive Control: Past, Present, and Future. *Computers & Chemical Engineering*, 23(4-5), 667-682
- Muske, K. R., Howse, J. W., Hansen G. A., and Cagliostro D. J. (2000). Model-based control of a thermal regenerator. Part 2: control and estimation, *Computers & Chemical Engineering*, 24(11): 2507-2517
- Narendra, K. S. & K. Parthasarathy (1990). Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*. 1(1): 4-27.
- Norquay, S. J., Palazoglu A., and Romagnoli J. A. (1999). Application of Wiener model predictive control (WMPC) to an industrial C2-splitter, *Journal of Process Control*, 9(6), 461-473

- Norquay, S. J., Palazoglu, A., Romagnoli, J. A. (1999). Application of Wiener model predictive control (WMPC) to a pH neutralization experiment, *IEEE Transactions on Control Systems Technology*, 7(4), 437-445
- Özkan, L. , Kothare, M.V., and Georgasubstitutingkis, C. (2000). Model predictive control of nonlinear systems using piecewise linear models, *Computers & Chemical Engineering*, 24(2-7): 793-799
- Pandit, H.G. (1991). Experimental demonstration of model-based control techniques. PhD Dissertation, Texas Tech University, Lubbock, TX
- Pickhardt, R. (2000), Nonlinear modelling and adaptive predictive control of a solar power plant, *Control Engineering Practice*, 8(8), 937-947
- Piche, S., Sayyar-Rodsari, B., Johnson, D., Gerules, M. (2000). Nonlinear Model predictive Control Using Neural Network, *IEEE Control Systems Magazine*, 20(3):53-62
- Prasad, G., Irwin, G.W., Swidenbank, E., Hogg, B.W. (2000). Plant-wide predictive control for a thermal power plant based on a physical plant model, *IEE Proceedings: Control Theory and Applications*, 147(5), 523-537
- Primbs, J. A. & Nevistic, V. (2000a). Feasibility and stability of constrained finite receding horizon control, *Automatica*, 36(7): 965-971
- Primbs, J. A., Nevistic, V. (2000b). Papers - A Framework for Robustness Analysis of Constrained Finite Receding Horizon Control. *IEEE transactions on automatic control*. 45(10): 1828-1838
- Puskorius, G. V. & L. A. Feldkamp (1994). Neurocontrol of nonlinear dynamic systems with Kalman Filter trained recurrent networks. *IEEE Transactions on Neural Networks*. 5(2): 279-297.
- Qin, S. J. & Badgwell, T.A. Eds. (1997). An Overview of Industrial Model Predictive Control Technology. *Proceedings of the 5th International Conference on Chemical Process Control (CPC-V)*, 232-256
- Qin, S.J. & Badgwell, T.A. (1998). An Overview of Nonlinear Model Predictive Control Applications. *Proc. IFAC Workshop: Non-linear Model Predictive Control: Assessment and Future Directions*. Ascona, Switzerland
- Ramchandran, S. (1994). Neural network model-based control of distillation columns. PhD Dissertation, Texas Tech University, Lubbock, TX
- Rani, K. Y. & Unbehauen, H. (1997). Study of predictive controller tuning methods. *Automatica*, 33(12): 2243-2248

- Rao, C.V. & Rawlings, J.B. (2000). Linear programming and model predictive control, *Journal of Process Control*, 10(2-3):283-289
- Rawlings, J.B. (2000). Tutorial Overview of Model Predictive Control, *IEEE Control Systems Magazine*. 20(3): 38-52
- Rawlings, J.B. & Muske, K.R. (1993). Stability of constrained receding horizon control. *IEEE Transactions of Automatic Control*. 38(10):1512-1516
- Rhinehart, R. R., Riggs, J. B. (1991). Two Simple Methods for On-line Incremental Model Parameterization. *Computers & Chemical Engineering*, 15(3):181-189
- Richalet, J., Rault, A., Testud, J. L., Papon, J. (1979). Model predictive heuristic control: applications to industrial processes. *Automatica*, 14(5):413-428.
- Roffel, B., Betlem, B. H. L. and de Ruijter, J. A. F. (2000). First principles dynamic modeling and multivariable control of a cryogenic distillation process, *Computers & Chemical Engineering*, 24(1): 113-123
- Roubos, J. A., Mollov, S., Babuka R., and Verbruggen H. B. (1999). Fuzzy model-based predictive control using Takagi-Sugeno models, *International Journal of Approximate Reasoning*, 22(1-2): 3-30
- Scarselli, F. & Tsoi, A.C. (1998). Universal Approximation Using Feedforward Neural Networks: A Survey of Some Existing Methods, and Some New Results, *Neural Networks*, 11(1), 15-37
- Scattolini, R. & Bittanti, S. (1990). On the choice of the horizon in long-range predictive control – some simple criteria. *Automatica*, 26(5): 915-917.
- Seborg, D.E., Edgar, T.F., & Mellichamp, D.A. (1989). Process dynamics and control, New York : Wiley.
- Shridhar, R. & Cooper, D. J. (1998). Process design and control - A tuning strategy for unconstrained multivariable model predictive control. *Industrial & Engineering Chemistry Research*. 37(10): 4003-4017
- Su, H. T., McAvoy, T. J., & Werbos, T. J. (1992). Long-term predictions of chemical processes using recurrent neural networks: A parallel training approach. *Industrial & Engineering Chemistry Research*, 31: 1338-1352.
- Temeng, K. O. ; Schnelle, P. D. ; McAvoy, T. J. (1995). Model predictive control of an industrial packed bed reactor using neural networks, *Journal of process control*, 5(1):19

Tsoi, A. C. (1997). Discrete time recurrent neural network architectures: A unifying review. *Neurocomputing*, 15(3-4): 183-225.

Werbos, P. J. (1990). Backpropagation Through Time: What It Does and How to Do It. *Proceedings of the IEEE*. 78(10): 1550-1560.

Zamarreño J. M., & Vega, P. (1999). Neural predictive control. Application to a highly non-linear system, *Engineering Applications of Artificial Intelligence*, 12(2):149-158

Zhu, G.Y., Henson M. A., and Ogunnaike, B. A.(2000), A hybrid model predictive control strategy for nonlinear plant-wide control, *Journal of Process Control*, 10(5): 449-458

APPENDIX A

CALIBRATIONS FOR THE DISTILLATION UNIT

This section describes calibrations for carrying out experiments on the methanol-water distillation unit.

A.1 Heating Power for the Reboiler

Heating power for the reboiler is a manipulated variable for the GNNMPC system, it is also needed in the simulator to obtain the boilup rate -- vapor flowrate from the reboiler. The SCR heater control element accepts 4~20mA current signal from Camile analog output (A/O) board (which is expressed in its interface as a percentage value) and converts it into an on-off time ratio of the electrical heating source. Calibration for the reboiler is to correlate between the percentage value from Camile (H) and the real power to the reboiler as expressed by percentage of full power (η), while the full power P of the reboiler's heating element is measured to be $205V * 19.8A / \sqrt{2} = 2.8701kW$.

The calibration was carried out by filling the reboiler with certain amount of pure water (14.2 liters in the experiment, at which the reboiler is at its overflow level) and heating the water under certain values of H. The equivalent percentage of full power, η , is calculated from the following energy balance equation:

$$\eta P \Delta t = m C_p \Delta T \quad (\text{A.1})$$

where η Equivalent percentage of full power (%)

P Full power of the heating element (kW)

Δt Time period of heating, $\Delta t = t_1 - t_0$ (s)

m Mass of pure water being heated (kg)

C_p Heat capacity of the pure water (kJ/kg.K)

ΔT Temperature difference corresponding to Δt , $\Delta T = T_1 - T_0$ (K)

Temperature usually is made to change from room temperature to 50 °C, so that C_p can be taken as a constant (4.183 kJ/kg.K) and that evaporation effect is negligible. Equation (A.1) is based on the assumption that the heat transfer to the other parts of the reboiler (heating element, the insulation, and the reboiler shell) is negligible.

For this work, the range of H is 40% to 60%, the calibration result is:

$$\eta = 1.5525 * H - 2.6871, R^2 = 0.9825 \quad (\text{A.2})$$

The calibration showed that when H is greater than 70%, $\eta = 100\%$.

A.2 Flow Rates

The feed flowrate and the reflux flowrate are measured through differential pressure (DP) cells which are installed on the pipelines to sense the pressure drops across the orifices. The DP cells send 4~20 mA signals to Camile[®]. The calibration procedure is to use pure water under room temperature (about 20 °C) to get the correlation between the current signal, i (mA), and the standard volume flowrate, q (ml/sec). The correlation is in the structure of $q = \alpha(i - i_0)^a$, which means that the flowrate is proportional to a certain power of the differential pressure drop across the orifice. In correlation, the structure is converted to $\log_{10}(q) = a \log_{10}(i - i_0) + \log_{10}(\alpha)$ for convenient and accurate

correlating. The correlation determines constants a , b ($=\log_{10}(\alpha)$), and i_0 from experimental data.

Table A.1 list calibration results for the feed flowrate and the reflux flow rate. A low flowrate region is separately calibrated with more condensed data points and is used in the experimental work, because both the feed flowrate and the reflux flowrate fall into this low flowrate region in the experimental work.

Table A.1 Calibration results for feed flowrate and reflux flowrate under the structure $\log_{10}(q) = a \log_{10}(i - i_0) + b$

	Low Flowrate Region		Full Region	
Feed	$i_0 = 4.23$ $a = 0.4949$ $b = 0.5207$	$R^2 = 0.9962$	$i_0 = 4.23$ $a = 0.4973$ $b = 0.5221$	$R^2 = 0.9998$
Reflux	$i_0 = 4.235$ $a = 0.5213$ $b = 0.4546$	$R^2 = 0.9997$	$i_0 = 4.235$ $a = 0.5123$ $b = 0.4534$	$R^2 = 0.9997$

A DP cell was also installed for sensing the flowrate of the cooling water. For this work, it is only needed to ensure that the cooling water is flowing. Therefore, the current signal from the DP cell is used directly as the controlled variable instead of being converted to a flowrate value.

For the experimental work, methanol-water mixture flowrates should be the online measured variables. The mixture flowrates were obtained by converting the equivalent water flowrates to the mixture flowrates by doing a density compensation based on the mixture's composition, as described below.

The relationship between the volume flowrate, q , and the differential pressure, ΔP (as proportional to the differential current signal of the transmitter, $i - i_0$), is as

$q = C_1 \left(\frac{\Delta P}{\rho}\right)^a$ or $q = C_2 \left(\frac{i - i_0}{\rho}\right)^a$, where C_1 and C_2 are constant coefficients, ρ is the

density of the fluid, and a is a constant obtained from calibration as described above.

Therefore, for the same amount of differential pressure across the orifice, the relationship

between the volume flowrate of different fluids is as $\frac{q}{q_0} = \left(\frac{\rho}{\rho_0}\right)^{-a}$, where the subscript

“0” represents the base fluid (pure water here). Then the mass flowrate, m , can be

calculated as $m = \rho q$ and can be further converted to mole flowrate, n . The steps for

obtaining online feed mole flowrate and reflux mole flowrate are as follows:

Step1: Convert liquid mole fraction, x_n , to mass fraction, x_m :

$$x_m = \frac{w_{MeOH} x_n}{(w_{MeOH} x_n + w_{H_2O} (1 - x_n))}, \text{ where } w \text{ stands for molecular weight.}$$

Step 2: Get the density of the fluid from the correlation between the mole composition and the density as obtained in (B.7).

Step 3: Calculate the mass flowrate of the fluid as: $m = \rho q = \rho q_0 \left(\frac{\rho}{\rho_0}\right)^{-a} = \rho^{1-a} \rho_0^a q_0$.

Step 4: Convert mass flowrate to mole flowrate as: $n = mx / w_{MeOH} + m(1 - x) / w_{H_2O}$.

For the feed, the mole composition is measured offline by the refractometer. For the reflux, the mole composition is obtained through the correlation between the measured top tray temperature and its corresponding saturated vapor composition as in (B.2).

A.3 Levels

Three levels are needed for monitoring purpose, levels for the top (product) tank, the bottom (product) tank, and the reflux drum. The top product and the bottom product

flow to the top tank and the bottom tank respectively, which are both 5-gallon polypropylene containers in rectangular shape. Levels of the tanks should be monitored for overflow alarm. Additionally, flowrates of the top and bottom products can be estimated through level change in the tanks. Level of the reflux drum should be monitored so that under the normal operating condition, the level will keep at overflow level. Signals are sensed by DP cells.

Since the levels are for monitoring purpose only, composition effect is ignored by calibrating levels of the top tank and the reflux drum using pure methanol and level of the bottom tank using pure water.

$$\text{Level of top tank: Level [inch]}=1.0587*\text{dp [mA]} - 5.9575, R^2 =1 \quad (\text{A.3})$$

$$\text{Level of bottom tank: Level [inch]}=0.8797*\text{dp [mA]} - 7.1735, R^2 =1 \quad (\text{A.4})$$

$$\text{Level of reflux drum: Level [inch]}=0.8029*\text{dp [mA]} - 3.3115, R^2 =1 \quad (\text{A.5})$$

The levels of the top and bottom tanks can be further used to provide a rough estimation of the top and bottom product flowrates according to $m = \rho g \Delta h$, where m is the mass flowrate of the product, ρ the product density, and Δh the change rate of the liquid level. For the operating range of this work, it usually takes about 5 minutes for the product to bring observable change to the liquid level in the tank. Therefore, the calculated flowrate is the mean flowrate within this time period, further, the level signal needs to be filtered to remove the noise. A CUSUM type filter by Rhinehart (“A CUSUM Type On-Line Filter”, Process Control and Quality, 2, 169-176, 1992) was used.

A.4 Column Pressure

The column pressure refers to the pressure difference between atmosphere and the reboiler's vapor space. The signal indicates accumulated liquid level in the distillation column and found out to be a good index of flooding/entraining. The pressure (in inH₂O) is calibrated with respect to the current signal from the DP cell as:

$$\text{Pressure [inH}_2\text{O]} = 1.5368 * dp \text{ [mA]} - 7.8364, R^2 = 1 \quad (\text{A.6})$$

A.5 Refractometer

The refractometer measures composition of the methanol-water mixture from liquid sample. This offline measurement device is used to measure the composition of the feed, and give feedback for online temperature-composition correlations of the top and the bottom products. Calibration curve is plotted in Figure A.1. Diamond dots in Figure A.1 are points that were calibrated.

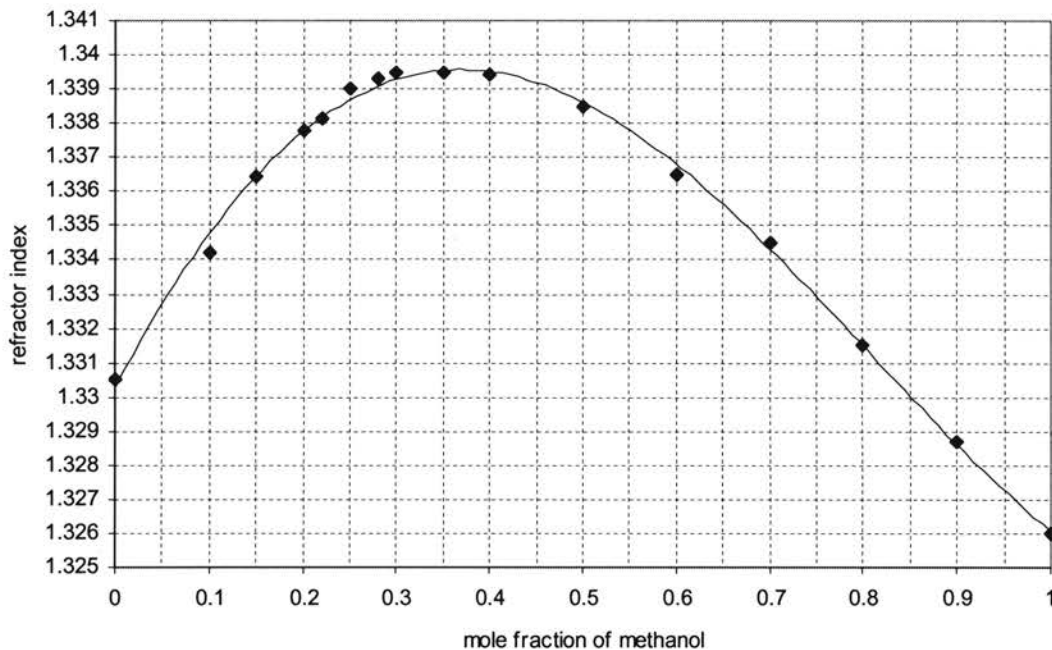


Figure A.1 Calibration curve of the refractometer for methanol-water system

APPENDIX B

CORRELATIONS OF THERMODYNAMIC PROPERTIES IN THE

METHANOL-WATER SYSTEM

Thermodynamic properties of the methanol-water system under standard pressure is available in Ranchandran's thesis (Ramchandran, S. Neural network model-based control of distillation columns. PhD Dissertation, Texas Tech University, Lubbock, TX, 1994). Here the original data points for correlations are listed in tables. Correlations are redone in the physical units applied for this work.

B.1 Vapor-Liquid Equilibrium (VLE)

The data points are listed in Table B.1 and plotted in Figure B.1. y - x , y - T , and x - T relationships were correlated. y - x correlation is used in the simulator for VLE calculation. y - T relationship is used during experiment to refer top product composition from temperature on the top tray, and x - T relationship is used during experiment to refer bottom product composition from temperature of the reboiler.

y - x correlation:

$$y = 11.2092x^5 - 33.4747x^4 + 37.8756x^3 - 20.2753x^2 + 5.6509x + 0.00207, \quad R^2=0.9991 \quad (\text{B.1})$$

y - T correlation:

$$y = -8.3487T'^3 + 18.426T'^2 - 15.927T' + 5.8427, \quad R^2=0.9998, T' = T[C]/100, 65^{\circ}C \leq T \leq 80^{\circ}C \quad (\text{B.2})$$

x - T correlation:

$$x = -16.587T'^3 + 50.69T'^2 - 52.009T' + 17.91, \quad R^2=0.9998, T' = T[C]/100, 80^{\circ}C \leq T \leq 100^{\circ}C \quad (\text{B.3})$$

Table B.1 VLE for methanol-water system at standard pressure (1 atm)
 (T in °C, x and y are mole fractions of methanol in liquid and vapor phase, respectively)

T	x	y
64.5	1.00	1.00
65.0	0.95	0.98
66.0	0.90	0.96
67.6	0.80	0.92
69.3	0.70	0.87
71.2	0.60	0.83
73.1	0.50	0.78
75.3	0.40	0.73
78.0	0.30	0.67
81.7	0.20	0.58
84.4	0.15	0.52
87.7	0.10	0.42
89.3	0.08	0.37
91.2	0.06	0.30
93.5	0.04	0.23
96.4	0.02	0.13
100.0	0.00	0.00

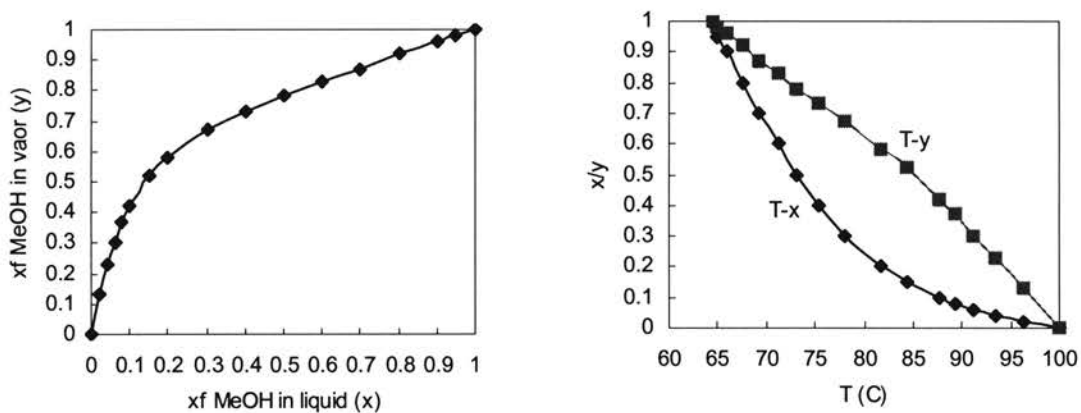


Figure B.1 VLE plot of the methanol-water system under standard pressure (1 atm)

B.2 Enthalpy

Enthalpy data for methanol-water system is shown in Table B.2. H represents enthalpy of the vapor phase and h is enthalpy of the liquid phase.

Table B.2 Enthalpy data for the methanol-water system under standard pressure (1 atm)

x or y (mf MeOH)	H (Btu/lbmol)	h (Btu/lbmol)
0.00	20720	3240
0.05	20520	3070
0.10	20340	2950
0.15	20160	2850
0.20	20000	2760
0.30	19640	2620
0.40	19310	2540
0.50	18970	2470
0.60	18650	2410
0.70	18310	2370
0.8	17980	2330
0.9	17680	2290
1.0	17390	2250

$$\text{H-y correlation: } H = -3338.3y + 20669.1, \quad R^2=0.9993 \quad (\text{B.4})$$

$$\text{h-x correlation: } h = -1692.5x^3 + 3631.7x^2 - 2918.9x + 3218.5, \quad R^2=0.9987 \quad (\text{B.5})$$

In the simulator, the enthalpy correlations are needed to solve the algebraic energy balance equations. In the experiment, they are used to calculate the latent heat of the reboiler to obtain the boilup rate for monitoring purpose.

B.3 Density

Table B.3 lists data points available for saturated mixture (ρ_{sat}) and subcooled (50 °C) mixture (ρ_{sc}).

Table B.3 Density data for the methanol-water system

x mf MeOH	ρ_{sat} lb/ft ³	ρ_{sc} lb/ft ³
0.00	59.18	61.75
0.05	58.12	60.37
0.10	57.07	59.11
0.15	56.07	57.96
0.20	55.12	56.9
0.25	54.24	55.93
0.30	53.41	55.04
0.35	52.65	54.21
0.40	51.93	53.45
0.45	51.26	52.73
0.50	50.64	52.07
0.55	50.07	51.46
0.60	49.53	50.89
0.65	49.03	50.35
0.70	48.57	49.85
0.75	48.13	49.39
0.80	47.73	48.95
0.85	47.36	48.55
0.90	47.01	48.17
0.95	46.69	47.81
1.00	46.40	47.48

Saturated mixture correlation:

$$\rho_{sat} = 59.215 - 23.01x + 13.289x^2 - 3.104x^3, R^2=1.0000 \quad (\text{B.6})$$

Subcooled mixture correlation:

$$\rho_{sc} = 61.696 - 27.477x + 19.492x^2 - 6.269x^3, R^2=1.0000 \quad (\text{B.7})$$

In the simulator, density of the mixture in the liquid phase is needed to calculate liquid holdup on each tray. For the experiment, subcooled mixture densities are needed to get flowrates of the feed and the reflux.

APPENDIX C

MABLAB CODES FOR THE TRAY-TO-TRAY DYNAMIC MODEL OF THE METHANOL-WATER DISTILLATION COLUMN

Algorithms and procedures for constructing the tray-to-tray dynamic model of the distillation column is described in Chapter 5. This section lists the Matlab codes.

C.1. Initialization

```
%function InitDistill.m
%   Initialization of the distillation simulator
%   Define and initialize global variables
% Note that in the simulator, the units are:
% flowrate in [lbmol/hr], Heat in [BTU/hr], Temperature in [F]
% Time in [hr], Volume in [ft^3]

global TimeStep;
global Stages FeedStage RefluxStage Column Reboiler Drum; %Column
Geometry
global Efficiency; %Process parameter
global Feed xz FeedT RefluxT Reflux Boilup TY; %Inputs

global trayT l v x y hl hv m xm; %internal states
global xR xD xB D B; %Distillate and Bottom flowrates and composition

%----- SIMULATION PARAMETERS
TimeStep=1.5/3600; %hr, If TimeStep is bigger,it will oscillate
SampleT=30/3600; %sampling period in hr

%----- COLUMN GEOMETRY
Stages=7; %Equilibrium Stage, reboiler is stage 1
FeedStage=4; RefluxStage=Stages;
    %These 2 are stages which have outside feed that need special
consideration

Column.weirH_S=0.25; %weir height in stripping stages, 1/4in obtained
from measurement;
Column.weirH_R=0.25; %weir height in rectifying stages,1/4in obtained
from measurement;
Column.Dia_S=2.65; %column diameter in stripping stages, 2.65in
obtained from measurement;
Column.Dia_R=2.65; %column diameter in rectifying stages,2.65in
obtained from measurement;

%Column diameter is deemed to be the same as that of the drum, which is
2.65in from measurement
Column.weirD_S=0.25;
```

```

%circular weir diameter in stripping stages,1/4in OD;
Column.weirD_R=0.25;
%circular weir diameter in rectifying stages,1/4in O.D.;
Column.weirL_S=pi*Column.weirD_S;
Column.weirL_R=pi*Column.weirD_R;
%NOTE: Feed stage is included in the Rectifying stages

Reboiler.volume=1.42e-3*35.3147; %ft^3, 1L=35.3147e-3;
%Reboiler volume, 1.42L is 1/10 of the real reboiler volume, which is
14.2L at overflow level

Drum.volume=0.535e-3*35.3147;
Drum.area=pi*2.65^2/4-pi*0.41^2/4; %[inch^2]
Drum.max_level=6.07;
Drum.level=Drum.max_level; %set drum to be full
%If we check the consistency,Drum.max_level*Drum.area=Drum.volume*12^3;

%-----
Efficiency=[1 0.2 0.4 0.7 0.7 0.8 0.8];

Reflux=100/454; TY=50;
Feed=280/454; xz=0.25; FeedT=35*1.8+32; RefluxT=50*1.8+32;

%----- INITIALIZATION
% l and x are the initialized independent variables in solving the Mass
Balance and Energy Balance PDEs. In our simulator, E.B. is algebraic
equations.
l=Reflux*ones(Stages,1); x=xz*ones(Stages,1);
l(1)=0; x(1)=0.1; %---Reboiler

B=0;xB=x(1); D=0;xD=x(end); xR=xD;

%----- INITIALIZE TRAY HOLD_UP
%----- Equilibrium Stages
[m,xm]=initialize_holdup(x,1);
%----- Drum, let it be 70% of overflow level
[amw,density]=get_properties('mw_and_dens',xR,'Subcool');
Drum.max_mass=Drum.volume*density/amw;
Drum.m=Drum.max_mass*0.7;
Drum.xm=Drum.m*xR;
volume=Drum.m*amw/density;
Drum.level=(volume)*12^3/Drum.area;%[inch]

%-----Assign memory space to v,y,hl and hv
v=zeros*ones(Stages,1); y=zeros(Stages,1);
hl=zeros(Stages,1); hv=zeros(Stages,1);

function [m,xm]=initialize_holdup(x,1)
%Initialize liquid holdup m,xm by correlation of weir dynamics (Francis
weir correlation)
global Stages FeedStage Column Reboiler;
m=zeros(Stages,1); xm=zeros(Stages,1);
condition='SatL';
%-----reboiler
xx=x(1);
[amw,density]=get_properties('mw_and_dens',xx,condition);

```

```

m(1)=Reboiler.volume*density/amw;
%-----stripping stages
for i=2:FeedStage-1
    xx=x(i);
    [amw,density]=get_properties('mw_and_dens',xx,condition);
    lv=l(i)*amw/density; %lv is Q_L in ft^3/hr
    hfow=(lv/(999*Column.weirL_S))^(2/3); %Francis weir correlation
    %hfow=get_weir('weir',lv,Column.weirD_S)/12;
    mv=(hfow+Column.weirH_S/12)*pi*(Column.Dia_S^2)/(4*144);
    %liquid holdup in ft^3/hr
    m(i)=mv*density/amw; %liquid holdup in lbmol/hr
end
%-----rectifying stages
for i=FeedStage:Stages
    xx=x(i);
    [amw,density]=get_properties('mw_and_dens',xx,condition);
    lv=l(i)*amw/density;
    hfow=(lv/(999*Column.weirL_R))^(2/3); %Francis weir correlation
    %hfow=get_weir('weir',lv,Column.weirD_R)/12;
    mv=(hfow+Column.weirH_R/12)*pi*(Column.Dia_R^2)/(4*144);
    m(i)=mv*density/amw;
end

xm=m.*x;

```

C.2 Main Calculation Procedures

```

function Distill()
% Last Updated: 2000-10-25, Modified from Distill-0727
% One step simulation of the Distillation process
global TimeStep;
global Stages FeedStage RefluxStage Column Reboiler Drum Efficiency;
%Column Geometry
global Feed xz FeedT RefluxT Reflux Boilup TY; %Process inputs
global l v x y hl hv m xm; %internal states
global D xD xR; %xD=xR in the simulation

get_vapor_phase; %1.-----GET vapor phase properties,y&v from x and
Energy Balance
get_holdup;      %2.-----GET m(holdup) FROM Mass Balance
update_x_and_l;  %3.-----UPDATE x&l:x=xm/m; l=Function(m)--weir
correlation
update_drum;     %4.-----UPDATE xD

%-----
%*****
%-----
function get_vapor_phase()
global Stages Efficiency FeedStage RefluxStage;
global x y hl hv l v;
global Feed xz FeedT RefluxT Reflux Boilup TY;
global xR;
y(1)=get_properties('vle',x(1));
for i=2:Stages,
    yideal=get_properties('vle',x(i));
    y(i)=(yideal-y(i-1))*Efficiency(i)+y(i-1);

```

```

end
[hl,hv]=get_properties('enthalpy',x,y,'Normal');

v(1)=reboiler('boilup',TY,x(1));
for i=2:Stages-1,
    v(i)=(hl(i+1)*l(i+1)+hv(i-1)*v(i-1)-hl(i)*l(i))/hv(i);
    if i==FeedStage,
        [Feed_enthalpy,tmp]=get_properties('enthalpy',xz,[],FeedT);
        v(i)=v(i)+Feed*Feed_enthalpy/hv(i);
    end
end
i=Stages;
[Reflux_enthalpy,tmp]=get_properties('enthalpy',xR,[],RefluxT);
v(i)=(hv(i-1)*v(i-1)-hl(i)*l(i)+Reflux*Reflux_enthalpy)/hv(i);

%-----
function get_holdup()
global TimeStep;
global Stages Reboiler Drum FeedStage RefluxStage;
global Reflux xR Feed xz;
global x y l v m xm;
dm_dt=zeros(Stages,1); dxm_dt=zeros(Stages,1);
%-----Reboiler,
[amw,density]=get_properties('mw_and_dens',x(1),'SatL');
m(1)=Reboiler.volume*density/amw; %assumption of perfect level control
l(1)=l(2)-v(1); %assume dm_dt=0, reasonable because Reboiler volume is
very large
if l(1)<0, l(1)=0; end
dxm_dt(1)=(l(2)*x(2)-v(1)*y(1)-l(1)*x(1));
%-----Equilibrium trays
for i=2:Stages-1,
    dm_dt(i)=(l(i+1)+v(i-1)-l(i)-v(i));
    dxm_dt(i)=(x(i+1)*l(i+1)+y(i-1)*v(i-1)-x(i)*l(i)-y(i)*v(i));
    if i==FeedStage,
        dm_dt(i)=dm_dt(i)+Feed; dxm_dt(i)=dxm_dt(i)+Feed*xz;
    end
end
end
%----- Top tray
i=Stages;
    dm_dt(i)=(v(i-1)-l(i)-v(i)+Reflux);
    dxm_dt(i)=(y(i-1)*v(i-1)-x(i)*l(i)-y(i)*v(i)+Reflux*xR);
%INTEGRATE
m=m+dm_dt*TimeStep; xm=xm+dxm_dt*TimeStep;

%-----
function update_x_and_l()
global Stages FeedStage Column;
global m xm x y l v;
%-----Equilibrium Stages
x=xm./m;
for i=1:Stages,x(i)=max(0,min(1,x(i)));end
%-----Stripping stages
for i=2:FeedStage-1,
    [amw,density]=get_properties('mw_and_dens',x(i),'SatL');
    how=(4*144/pi)*m(i)*amw/(density*(Column.Dia_S^2))-
Column.weirH_S/12.0;
    if how<0,l(i)=0;

```

```

    else
        l(i)=density*Column.weirL_S*999.0*how^1.5/amw;
    end
end
%-----Rectifying stages
for i=FeedStage:Stages,
    [amw,density]=get_properties('mw_and_dens',x(i),'SatL');
    how=(4*144/pi)*m(i)*amw/(density*(Column.Dia_R^2))-
Column.weirH_R/12.0;
    if how<0, l(i)=0;
    else
        l(i)=density*Column.weirL_R*999.0*how^1.5/amw;
    end
end

%-----
function update_drum()
global TimeStep;
global Drum Reflux xR xD D l v x y;
if Drum.max_level-Drum.level<0.001 & v(end)-Reflux>0, D=v(end)-Reflux;
else D=0;
end
dm_dt=v(end)-Reflux-D;
dxm_dt=v(end)*y(end)-Reflux*xR-D*xD;
Drum.m=Drum.m+dm_dt*TimeStep;
Drum.xm=Drum.xm+dxm_dt*TimeStep;
xR=max(0,min(1,Drum.xm/Drum.m));
xD=xR;
[amw,density]=get_properties('mw_and_dens',xR,'Subcool');
volume=Drum.m*amw/density;
Drum.level=(volume)*12^3/Drum.area;%[inch]
if Drum.level>Drum.max_level, Drum.level=Drum.max_level; end

```

C.3 Correlations of Thermodynamic Properties

```

function varargout=get_properties(name,varargin)
% Include all Thermo correlations,notice that all the unit are that of
the simulator
% Original data and plots from which correlations obtained are in
Ramchandran's PhD thesis
% "Neural Network Model-based Control of Distillation Columns"
%
% y=get_properties('vle',x);
% y in [mole fraction]
%
% [hl,hv]=get_properties('enthalpy',x,y,'Normal');
% OR [hl,hv]=get_properties('enthalpy',x,y,Subcooled_T);
% hl and hv in [BTU/lbmol]
%
% [amw,density]=get_properties('mw_and_dens',x,'Normal');
% OR [amw,density]=get_properties('mw_and_dens',x,Subcooled_T);
% amw in [lb/lbmol], density in [lb/ft^3]
%
% trayT=get_properties('Tfromx',x);
% trayT in [F]

```

```

%
% trayT=get_properties('Tfromy',y);
% trayT in [F]

if strcmp(name,'vle'),
    x=varargin{1}; y=get_vle(x);
    varargout{1}=y;
elseif strcmp(name,'enthalpy'),
    x=varargin{1}; y=varargin{2}; condition=varargin{3};
    [hl,hv]=get_enthalpy(x,y,condition);
    varargout{1}=hl; varargout{2}=hv;
elseif strcmp(name,'mw_and_dens'),
    x=varargin{1}; condition=varargin{2};
    [amw,density]=get_mw_and_dens(x,condition);
    varargout{1}=amw; varargout{2}=density;
elseif strcmp(name,'Tfromx'),
    x=varargin{1};
    trayT=get_Tfromx(x);
    varargout{1}=trayT;
elseif strcmp(name,'Tfromy'),
    y=varargin{1};
    trayT=get_Tfromy(y);
    varargout{1}=trayT;
else disp('no correlation for this property');
end

%-----
function yy=get_vle(xx)
if xx>1 | xx<0, error('in VLE correlation, x mole fraction beyond
0~1'); end
yy=2.07e-2+5.6509*xx-20.2753*xx.^2+37.8756*xx.^3-
33.4747*xx.^4+11.2092*xx.^5;
yy=min(ones(size(yy)),yy);

%-----
function [hll,hvv]=get_enthalpy(xx,yy,condition)
if xx>1 | xx<0, error('in ENTHALPY correlation, x mole fraction beyond
0~1'); end
if yy>1 | yy<0, error('in ENTHALPY correlation, y mole fraction beyond
0~1'); end
if isstr(condition),
    if condition=='Normal',
        hll=3218.5-2918.9*xx+3631.7*xx.^2-1692.5*xx.^3;
        hvv=20669.1-3338.3*yy;
    else disp('invalid condition');
    end
else
    T=condition; %subcooled liquid
    yy=get_vle(xx); Tsat=get_Tfromx(xx);
    hl_sat=3218.5-2918.9*xx+3631.7*xx.^2-1692.5*xx.^3;
    hll=hl_sat-(19.49*xx+17.98*(1-xx)).*(Tsat-T);
    hvv=0*ones(size(xx));
end

%-----
function [amw,density]=get_mw_and_dens(xx,condition)

```



```

if xx>1 | xx<0, error('in DENSITY correlation, x mole fraction beyond
0~1'); end
amw=18.015+14.027*xx; %Equivalent to: 18*(1-x)+32*x;

switch condition,

case 'SatL'    %saturated liquid density
    density=59.215-23.01*xx+13.298*xx.^2-3.104*xx.^3;
case 'SatV'    %saturated vapor density????? needs to be checked
    %on the other hand, saturated vapor density seems never used in the
simulator
    density=3.68e-2+3.02e-2*xx+5.57e-3*xx.^2-2.02e-4*xx.^3;
case 'Subcool' %liquid density at 120F(50C)
    density=61.696-27.477*xx+19.492*xx.^2-6.269*xx.^3;
otherwise
    disp('invalid condition');
end

%-----
function TT=get_Tfromx(xx)
if xx>1 | xx<0, error('in Tx correlation, x mole fraction beyond 0~1');
end
TT=210.76-243.45*xx+515.74*xx.^2-547.70*xx.^3+213.33*xx.^4;

%-----
function TT=get_Tfromy(yy)
if yy>1 | yy<0, error('in Ty correlation, y mole fraction beyond 0~1');
end
TT=14.179*yy.^3-39.778*yy.^2-38.363*yy+211.54;

%-----END
get_properties()

function output=reboiler(name,TY,varargin)
% ReboilerDuty=reboiler('duty',TY);
% boilup=reboiler('boilup',TY,x); %[lbmol/hr]
%
Epower=13850; % Btu/hr, 13850Btu/hr corresponds to 205V*19.8A=4.059kJ/s
Epower=Epower/1.414; %9795; effective power
realpercentage=1.5525*TY-2.6871; %R^2=0.9825; percentage=[12.5,70];
realpercentage=min(realpercentage,100);
ReboilerDuty=Epower*(realpercentage/100);
if strcmp(name,'duty'),
    output=ReboilerDuty;
elseif strcmp(name,'boilup'),
    x=varargin{1};
    y=get_properties('vle',x);
    [hl,hv]=get_properties('enthalpy',x,y,'Normal');
    latentheat=hv-hl; %Btu/lbmol
    output=ReboilerDuty/latentheat;
else disp('???err-->function reboiler()');
end

```

APPENDIX D

GNNMPC MATLAB CODES

GNNMPC are solved as a nonlinear constrained optimization problem. The Matlab optimization toolbox (version 1.5.2) was used. In the Matlab code, the function 'constr()' is called to generate optimized decision variables. For GNNMPC, the format of the function call is as ' $x = \text{constr}(\text{'obj_fun'}, x_0, \text{'', ''}, \text{'', ''}, p1, p2, \dots)$ '. Where 'obj_fun' is another function to calculate the objective function and the constraint matrix with problem-dependent parameters $p1, p2$, etc. Optimized control actions at each control step are used as the initial values (starting point x_0 in the parameter list of function 'constr()') for the search of the next optimized control actions. The function 'constr()' uses Sequential Quadratic Programming technique. Default values of the stopping criteria in the Matlab optimization algorithm is used, as listed in Table D.1. When any of the stopping criteria is met, the optimization algorithm stops search and returns the current values as the optimized values.

Table D.1 Default stopping criteria of the Matlab optimization algorithm

Termination tolerance of decision variables	10^{-4}
Termination tolerance of objective function	10^{-4}
Termination criterion on constraint violation	10^{-6}
Maximum number of function evaluations	100*number of decision variables

D.1 GNNMPC for Distillation Simulator

D.1.1 Main Subroutine

```
clear all; close all; echo off;
load initNMPC;

%===== SETPOINT TRACKING =====
runno=1; figno=runno;
runsp.series=[0.8414;0.0337];
runsp.time=[30];
runsp.pnts=min(size(runsp.series,2),length(runsp.time));

MPC.MODE='AUTO';
simu_length=90;
echo off;
%----- mainbody;
starttime=cputime;
record=[]; rec_funf=[]; SPchg=[0;SP];
options=foptions;

for ii=1:simu_length, % counting in no. of SAMPLE TIME

    if runsp.index<runsp.pnts,
        if ii==runsp.time(runsp.index+1),
            runsp.index=runsp.index+1;
            SP=runsp.series(:,runsp.index);
            SPchg=[SPchg [ii;SP]];
        end
    end

    if runmv.index<runmv.pnts,
        if ii==runmv.time(runmv.index+1),
            runmv.index=runmv.index+1;
            [Reflux,TY]=chgMV(runmv.series(runmv.index,:));
            SPchg=[SPchg [ii;SP]];
        end
    end

    if runld.index<runld.pnts,
        if ii==runld.time(runld.index+1),
            runld.index=runld.index+1;
            Feed=chgLD(runld.series(runld.index,:));
        end
    end

    [funf,func,deviation]=obj_fun(decisionVn,net_distill,history,SP);
    rec_funf=[rec_funf;[funf deviation]];

    for simuno=1:floor(SampleT/TimeStep), Distill_plant; end
    online.MVs=[Reflux*454 TY]; %change from lbmol/hr to gmol/hr
    online.LDs=[Feed*454];
    online.CVs=[y(end) x(1)];
    record=[record;[online.MVs online.CVs online.LDs]];
    history=[history(2:end,:);record(end,:)];

    pmm=pmmadjustment(net_distill,history);
```

```

%-----CONTROL
if strcmp(MPC.MODE, 'AUTO'),
    optn=constr('obj_fun', decisionVn, options, [], [], [], ...
        net_distill, history, SP);
    decisionVn=optn;
    mvs=unnormmmmx(optn(1:2,1), range(1:2,:));
    [Reflux, TY]=chgMV(mvs');
end

end

%
SPchg=[SPchg [simu_length;SPchg(2:3,end)]];

%-----END mainbody -----
myplotperf;

```

D.1.2 GNNMPC Initialization

```

%-----INITIALIZATION -----
clear all; close all; clc;
load ss4control; %Nominal Point Steady State
global range;
MVrange=[90 150; 45 60]; CVrange=[0.6 0.95; 0 0.2]; LDrange=[250 310];
range=[MVrange;CVrange;LDrange];
%----- CONSTRUCT MPC PARAMETERS
global MPC;
MPC.numMVs=2; MPC.numLDs=1; MPC.numCVs=2; %structure of system
%---model information
MPC.modelFile='net_distill_short_0803';
MPC.minmax=range;
MPC.pastWin=5;
MPC.predPnts=[1 2 3 5 10];
MPC.futureWin=MPC.predPnts(end);
%---control parameters
MPC.MODE='MANUAL';
MPC.deltamvW=0.1;
MPC.deltacvW=ones(size(MPC.predPnts));
MPC.mvChgPnts=[1 4];
%-----
SteadyState=[online.MVs online.CVs online.LDs];
history=ones(MPC.pastWin+MPC.futureWin+2,1)*SteadyState;
decisionV=nncopy([Reflux*454;TY],1,size(MPC.mvChgPnts,2));
decisionVn=normmmmx(decisionV,range(1:2,:));
SP=(online.CVs)'; SPn=normmmmx(SP,range(3:4,:));

global pmmfilter pmm;
pmm=zeros(2,length(MPC.predPnts)); pmmfilter=pmm;

%=====
runsp.series=[]; runsp.time=[]; runsp.index=0;
    runsp.pnts=min(size(runsp.series,2),length(runsp.time));
runmv.series=[]; runmv.time=[]; runmv.index=0;
    runmv.pnts=min(size(runmv.series,2),length(runmv.time));
runld.series=[]; runld.time=[]; runld.index=0;
    runld.pnts=min(size(runld.series,2),length(runld.time));

```



```

PastWin=MPC.pastWin;
pred_pnts=MPC.predPnts;
range=MPC.minmax;
%---Normalization
historyn=normmmmx(history',range)';
pred_values=[];
% Calculate prediction value of NOW from pred_pnts(i) time step ago
for lll=1:length(pred_pnts),
    index=size(historyn,1)-pred_pnts(lll); %pointing to the current
time: time instant NOW
    PastMVsCVs=[];
    for i=PastWin:-1:0,
        PastMVsCVs=[PastMVsCVs;historyn(index-i,1:4)'];
    end
    CurrentLDs=historyn(index,5)';

    FutureMVs=[];
    if pred_pnts(lll)>1,
        tmp=historyn(index+1:index+pred_pnts(lll)-1,1:2)';
        for kkk=1:size(tmp,2),
            FutureMVs=[FutureMVs;tmp(:,kkk)];
        end
    end

    NNinputs=[PastMVsCVs;CurrentLDs;FutureMVs];
    output=sim(NNs{pred_pnts(lll)},NNinputs);
    pred_values=[pred_values output];
end

onlineCVn=historyn(end,3:4) '*ones(size(pred_pnts));
pmm=onlineCVn-pred_values;

```

D.1.5. Objective Function

```

function [funf, fung, deviation]=obj_fun(decisionVn,NNs,history,SP)
%Constrained objective function
% ~~~~~~
global MPC pmm;
range=MPC.minmax;
[SP,SPn]=chgSP(SP);
mvs_now=history(end,1:2)';
mvs_now_norm=normmmmx(mvs_now,range(1:2,:));

% ~~~~ [ constraints ] ~~~~
if size(decisionVn,2)<2,
    absdeltaMV=abs(decisionVn-mvs_now_norm);
else
    oldV=[mvs_now_norm decisionVn(:,1:end-1)];
    absdeltaMV=abs(decisionVn-oldV);
end

tmp1=[decisionVn-1 0-decisionVn];
fung=tmp1(1:end);

%~~~~~ [ objective function ] ~~~~~~
predn=NNDistill(NNs,history,decisionVn);

```

```

adjust=predn+pmm;

SPn=SPn*100; adjust=adjust*100;
absdeltaMV=absdeltaMV*100;
absMV=abs(decisionVn)*100;

[R1,Q1]=size(adjust);
tmpdevCVs=sum(sum((SPn*ones(1,Q1)-adjust).^2).*MPC.deltacvW)/(R1*Q1);

[R2,Q2]=size(decisionVn);
tmpdeltaMVs=sum(sum(absdeltaMV.^2))/(R2*Q2);

deviation=[tmpdevCVs tmpdeltaMVs];
funf=10*(tmpdevCVs + MPC.deltamvW*tmpdeltaMVs);

```

D.1.6. Auxiliary Subroutines

```

function [SP,SPn]=chgSP(newSP)
% [SP,SPn]=chgSP(newSP)
% newSP(2*1) is setpoints in C
% SP is in C, SPn is normalized SP using normmmmx()
% global variable 'range' is needed

global range;
if any(size(newSP)-[2 1]), error('SP should be 2*1 vector'); end
if or(~isempty(find(newSP-range(3:4,1)<0)),~isempty(find(newSP-
range(3:4,2)>0))),
    error('one SP is out of range');
end
if ~(nargout==2),warning('chgSP output is not equal to 2'); end
SP=newSP;
SPn=normmmmx(SP,range(3:4,:));

function [Reflux,TY]=chgMV(MVinput)
% [Reflux,TY]=chgMV([MVINPUT]);
% MVINPUT(1,1)=reflux in gmol/hr, 150~300
% MVINPUT(1,2)=TY % 45~60
range=[90 150;45 60];

if any(size(MVinput)-[1 2]), error('input size should be 1*2 vector');
end
if or(~isempty(find(MVinput-range(:,1)'<-0.001)),~isempty(find(MVinput-
range(:,2) '>0.001))),
    disp('MVinput range is'); disp(num2str(range));
    error('one MVinput is out of range');
end
%if (MVinput(2)/MVinput(1))<1, error('Boilup less than Reflux,this wont
work'); end
Reflux=MVinput(1)/454; TY=MVinput(2);

function [Feed,xz,FeedT,RefluxT]=chgLD(LDinput)
% [Feed,xz,FeedT,RefluxT]=chgLD([LDINPUT])
% LDINPUT is a matrix of size [1,1] to [1,4]
% LDINPUT(1): feed in gmol/hr, 100~500 gmol/hr
% LDINPUT(2): xz in mole fraction, 0.1~0.3
% LDINPUT(3): FeedT in C, 20~40 C

```

```

% LDINPUT(3): RefluxT in C, 30~50 C

range=[250 310;0.1 0.3;20 40;30 50];
[R,Q]=size(LDinput);
if (R-1), error('input should be a one row vector'); end
if ~isempty(find(LDinput-range(1:Q,1)'<0)) | ~isempty(find(LDinput-
range(1:Q,2) '>0)),
    disp('LDinput range is');    disp(num2str(range));
    error('one LDinput is out of range');
end
if ~(nargout==Q),warning('#LDoutput is not equal to #LDinput'); end
switch length(LDinput),
case 1, Feed=LDinput(1)/454;
case 2, Feed=LDinput(1)/454; xz=LDinput(2);
case 3, Feed=LDinput(1)/454; xz=LDinput(2); FeedT=LDinput(3)*1.8+32;
case 4, Feed=LDinput(1)/454; xz=LDinput(2);
        FeedT=LDinput(3)*1.8+32; RefluxT=LDinput(4)*1.8+32;
otherwise, error('too many LD variables');
end

function pn=normmmmx(p,range)
% this function is modified from nnet\premmmx.m
% p: R*Q matrix
% range: R*2 matrix;

if nargin > 2
    error('Wrong number of arguments. ');
end

minp=range(:,1); maxp=range(:,2);
[R,Q]=size(p);
oneQ = ones(1,Q);

equal = minp==maxp;
nequal = ~equal;
if sum(equal) ~= 0
    warning('Some maximums and minimums are equal. Those inputs won''t be
transformed. ');
    minp0 = minp.*nequal - 1*equal;
    maxp0 = maxp.*nequal + 1*equal;
else
    minp0 = minp;
    maxp0 = maxp;
end

pn = (p-minp0*oneQ) ./ ((maxp0-minp0)*oneQ);

function p=unnormmmmx(pn,range)
% this function is modified from nnet\premmmx.m
% p: R*Q matrix
% range: R*2 matrix;

if nargin > 2
    error('Wrong number of arguments. ');
end

minp=range(:,1); maxp=range(:,2);

```



```

[R,Q]=size(pn);
oneQ = ones(1,Q);

equal = minp==maxp;
nequal = ~equal;
if sum(equal) ~= 0
    warning('Some maximums and minimums are equal. Those inputs won't be
transformed. ');
    minp0 = minp.*nequal - 1*equal;
    maxp0 = maxp.*nequal + 1*equal;
else
    minp0 = minp;
    maxp0 = maxp;
end

p = pn.*((maxp0-minp0)*oneQ)+minp0*oneQ;

```

D.2 GNNMPC for Experimental Distillation

Experimental runs involves communication between the upper-level control application, the Matlab code, and the lower-level Data Acquisition and Control (DAC) system, the Camile TG 4.05 used for the distillation column. Communication was realized by intermediate text files. Camile writes to the text files in certain format, and Matlab reads the data in the text files according to certain format, and vice versa. Online commands and manipulations were carried out in the Camile interface and were read by the Matlab code (the GNNMPC) when needed. Only Matlab codes are listed here.

D.2.1 Main Subroutine

```

%*****
CamileOutputFile='CamileOutput'; MatlabOutputFile='MatlabOutput';
PredictionFile='Prediction';
READ_FREQ=10;

global SP StartNMPC StopMatlab Measurement MPC;
%----the above are variables coming from Camile
global history NNs obj_func_name;
obj_func_name='obj_fun';
modelfile='NNs'; %'net_distill_short_0803';
load(modelfile); %the output is the global structure variable:NNs
history=[]; Tsample=60; %60 seconds

%Line 1 from the Camile's output file is initialized as following:
SP=zeros(2,1); StartNMPC=0; StopMatlab=0;
%Line 2 from the Camile's output file is initialized as following:
MPC.MovePnts=zeros(1,1); MPC.MoveW=MPC.MovePnts;
MPC.PredPnts.top=1; MPC.PredW.top=MPC.PredPnts.top;
MPC.PredPnts.bot=1; MPC.PredW.bot=MPC.PredPnts.bot;
MPC.Wmv=ones(2,1); MPC.Wcv=ones(2,1);
MPC.PredPnts.P=1;
MPC.f_mv=0.01;

HISTORY_LENGTH=NNs.PastH + NNs.PredH + ...

```

```

    max([NNS.top.Tmodel,NNS.bot.Tmodel,NNS.P.Tmodel])*2; %In minute
%-----Get SSHistory and initialize tuning parameters;
fprintf(1,'CONSTRUCTING INITIAL HISTORICAL DATA,PLEASE WAIT...\n');
steadystate=[];
startss=cputime;
while (size(steadystate,1)<20),
    if (floor(cputime-startss)>=READ_FREQ),
        startss=cputime;
        if exist(CamileOutputFile,'file'),
            ReadfromCamile(CamileOutputFile); %get global variables
        end
        steadystate=[steadystate;Measurement];
    end
end
meansteadystate=mean(steadystate);
fprintf(1,'CONSTRUCTION ENDS.\n');
display(meansteadystate);
history=nncopy(meansteadystate,HISTORY_LENGTH,1);
decisionV=nncopy(Measurement(1:2)',1,length(MPC.MovePnts));
%-----

rec=[]; rec_funf=[];
fprintf(1,mfilename);
fprintf(1,'  Reflux-----TY----ytop-----xbot----Pressure-----Feed
-----Time-----\n');
firstPMM=1; firstControl=1; %for filter purpose
deviations=[0 0 0];
startRead=cputime;startTsample=cputime;startTmodel=cputime;
startTcontrol=cputime;

while ~floor(StopMatlab),
%*****
%-----*****          ONLINE MEASUREMENT          *****-----
if (floor(cputime-startRead)>=READ_FREQ),
    startRead=cputime;
    if exist(CamileOutputFile,'file'),
        ReadfromCamile(CamileOutputFile); %get global variables
    end
end

%-----***** SAMPLE AND RECORD ONLINE MEASUREMENT *****-----
if (floor(cputime-startTsample)>=Tsample), %sample every 1 minute
    startTsample=cputime;
    history=[history(2:end,:);Measurement];
end

%-----***** CALCULATE and DISPLAY PMM EVERY Tmodel *****-----
if (floor(cputime-startTmodel)>=
min([NNS.top.Tmodel,NNS.bot.Tmodel,NNS.Tcontrol])*Tsample),
    startTmodel=cputime;
    aaa.top=NNS.top.PredPnts; aaa.bot=NNS.bot.PredPnts;
aaa.P=NNS.P.PredPnts;
    [pmm.top,pmm.bot,pmm.P]=groupPmm(history,aaa);
    timenow=clock;
    display_format='  %-8.1f %-8.1f %-8.3f %-8.3f %7.2f %10.0f  ---
-----@%2d:%2d\n';

```

```

fprintf(1,display_format,[Measurement timenow(4:5)]);
fprintf(1,'pmmtop      ');
fprintf(1,'%8.3f',pmm.top);fprintf(1,'\n');
fprintf(1,'pmmbot      ');
fprintf(1,'%8.3f',pmm.bot);fprintf(1,'\n');
if MPC.MaxPressure>0,
    fprintf(1,'pmm_Pressure  '); fprintf(1,'%8.3f',pmm.P);
    fprintf(1,'\n');
end
rec=[rec;[timenow(2:5) Measurement]];
end

%-----***** CALCULATE OptMV EVERY Tcontrol *****-----
if (floor(cputime-startTcontrol)>=NNS.Tcontrol*Tsample),
    startTcontrol=cputime;
    if firstControl,
        firstControl=0;
        MV.current=Measurement(1:2)';
        MV.old=MV.current; MV.f=MV.current;
        MV.cusum.k=zeros(size(MV.current)); MV.cusum.rou=MV.cusum.k;
        MV.cusum.sum=MV.cusum.k;
    end
    if StartNMPC,
        %check if MPC.PredPnts are all subsets of NNS.PredPnts
        MPC=checkMPC(MPC); %check 1.if MPC.PredPnts are trained; 2.if
size of PredPnts and PredW match
        [pmm.top,pmm.bot,pmm.P]=groupPmm(history,MPC.PredPnts);
        [decisionV,options]=constr(obj_func_name,decisionV,...
            [], [], [], [], SP,pmm);
        [funf,func,deviations]=feval(obj_func_name,decisionV,SP,pmm);
        rec_funf=[rec_funf; deviations];
        MV.current=decisionV(:,1);
        if MPC.mvfilter,
            [MV.f,MV.cusum]=MyFilter(MPC.filterManner,MV.current,...
                MV.old,MV.f,MV.cusum);
        else MV.f=MV.current;
        end
        WritetoCamile(MV.f,MatlabOutputFile);
    else decisionV=nncopy((Measurement(1:2))',1,length(MPC.MovePnts));
    end

Disp_Output_ModelInfo(decisionV,deviations,StartNMPC,PredictionFile);
end

%-----***** UPDATE time counter *****-----
%in case that the optimizer takes a period of time greater than
READ_FREQ and Tsample
if (cputime-startRead)>=READ_FREQ,
    startRead=cputime-mod((cputime-startRead),READ_FREQ);
end
if (cputime-startTsample)>=Tsample,
    startTsample=cputime-mod((cputime-startTsample),Tsample);
end
%*****
end

```



```

        historyn(end, :)], NNS.P, PredPnts.P);
    varargout{1}=normP*(range(5,2)-range(5,1));
end

%*****
function normpmm=subpmm(Tcontrol,ND, struc, PredPnts)

%ND represents 'NormData', each ROW is one variable
%Notice that PredPnts comes from MPC(Camile), which is a subset of that
in NNS.

%Tcontrol=NNS.Tcontrol;
ND=[historyn(1:2, :);historyn(4, :);historyn(end, :)];
%struc=NNS.bot; PredPnts=aaa.bot;

Tmodel=struc.Tmodel; PW=struc.PastWin;
UseAvgMV=struc.UseAvgMV;net=struc.net;
FW=PredPnts(end);

pred_n=[];
for kk=1:length(PredPnts),
    index=size(ND,2)-PredPnts(kk)*Tmodel;
        %pointing to the current time: time instant NOW
    if UseAvgMV,
%=====
        PastMVsCVs=[];
        if PW>0,
            for i=PW:-1:1,
                %average forward, MVCVs from index-PW*Tmodel to index-1*Tmodel
                avgPastMV=mean(ND(1:2, index-i*Tmodel+1:index-i*Tmodel+Tmodel-1), 2);
                PastMVsCVs=[PastMVsCVs;avgPastMV;ND(3, index-i*Tmodel)];
            end
        end
        CurrentLDs=mean(ND(end, index-Tmodel+1:index), 2); %backward averaging!
        CurrentCVs=ND(3, index);
        u0=mean(ND(1:2, index+1:index+Tmodel-1), 2);
        FutureMVs=[]; FutureMVs=u0;
        num_c=floor(PredPnts(kk)*Tmodel/Tcontrol);
        if num_c>1,
            for j=1:num_c-1,
                avgFutureMV=mean(ND(1:2, index+j*Tcontrol+1:index+(j+1)*Tcontrol-1), 2);
                FutureMVs=[FutureMVs;avgFutureMV];
            end
        end
    else %PW<0
%=====
        PastMVsCVs=[];
        if PW>0,
            for i=PW:-1:1,
                PastMVsCVs=[PastMVsCVs;ND(1:3, index-i*Tmodel+1)];
            end
        end
        CurrentLDs=ND(end, index);
        CurrentCVs=ND(3, index);

        u0=ND(1:2, index+2);
        FutureMVs=[]; FutureMVs=u0;
    end
end

```



```

sum_top=MPC.Wcv(1)*sum(MPC.PredW.top.*adp_wtop.*dev_top_pcg.^2)/length(
Fn.top);

dev_bot_pcg=(SPn(2)-Fn.bot)*100;
sum_bot=MPC.Wcv(2)*sum(MPC.PredW.bot.*adp_wbot.*dev_bot_pcg.^2)/length(
Fn.bot);

%----- MV moves
mvs_now=history(end,1:2)';
[R2,Q2]=size(decisionV);
if size(decisionV,2)<2,
    deltaMV=abs(decisionV-mvs_now);
else
    oldMV=[mvs_now decisionV(:,1:end-1)];
    deltaMV=abs(decisionV-oldMV);
end
deltaMV_perc=deltaMV./((range(1:2,2)-range(1:2,1))*ones(1,Q2))*100;
tmp=zeros(1,R2);
for numMV=1:R2,
    tmp(numMV)=MPC.Wmv(numMV)*sum(MPC.MoveW.*deltaMV_perc(numMV,:).^2);
end
moveMVs=sum(tmp)/(R2*Q2);
deviation=[sum_top sum_bot moveMVs];
funf=(sum_top + sum_bot + MPC.f_mv*moveMVs);

```

D.2.5 Interfaces with Camile

```

function ReadfromCamile(FileName)
% ReadfromCamile(FileName)
% Read global variables from FileName and delete Filename

global SP StartNMPC StopMatlab Measurement MPC;
fid=fopen(FileName,'rt');
while fid<0, fid=fopen(FileName,'rt'); end;
line1=fgetl(fid); line2=fgetl(fid);line3=fgetl(fid);line4=fgetl(fid);
%line1:[SP_top,SP_bot,StartNMPC, StopMatlab]
%line2:Measurement=[Reflux TY ytop xbot Pressure Feed]
%line3:tuning para.: [MovePnts MoveW PredPnts PredW Wmv Wcv
MaxPressure]
%line4:[SuppressionManner TrajSpeed pmmfilter mvfilter FilterManner
Lamda CusumWin CusumThreshold]

if ~ischar(line1) | ~ischar(line2) | ~ischar(line3) | ~ischar(line4),
    disp('??? Readings from CamileOutput not correct');
    fprintf(1,'line1=[%s],line2=[%s],line3=[%s],line4=[%s]\n',
line1,line2,line3,line4);
    return;
end
%-----LINE 1
x=sscanf(line1,'%f,'); %x is a vector
SP=x(1:2); StartNMPC=x(3); StopMatlab=x(4);
%-----LINE 2
x=sscanf(line2,'%f,');
Measurement=x';
%-----LINE 3
vararray=getVariables(line3);

```



```

global history NNs;
aaa.top=NNs.top.PredPnts; aaa.bot=NNs.bot.PredPnts;
aaa.P=NNs.P.PredPnts;
[pred.top,pred.bot,pred.P]=groupModel(history,decisionV,MPC.MovePnts,aa
a);
[pmm.top,pmm.bot,pmm.P]=groupPmm(history,aaa);
F.top=pred.top+pmm.top; F.bot=pred.bot+pmm.bot; F.P=pred.P+pmm.P;
timenow=clock;
fprintf(1,'                                @%2d:%2d\n',timenow(4:5));
fprintf(1,'top_pred      '); fprintf(1,'%8.3f',pred.top);fprintf(1,'\n');
fprintf(1,'top_adjust   '); fprintf(1,'%8.3f',F.top);fprintf(1,'\n\n');
fprintf(1,'bot_pred      '); fprintf(1,'%8.3f',pred.bot);fprintf(1,'\n');
fprintf(1,'bot_adjust   '); fprintf(1,'%8.3f',F.bot);fprintf(1,'\n\n');
if StartNMPC,
fprintf(1,'OPT_MV  ');fprintf(1,'%6.1f',decisionV(1,:));fprintf(1,'\n');
fprintf(1,'          ');fprintf(1,'%6.1f',decisionV(2,:));fprintf(1,'\n');
fprintf(1,'deviations'); fprintf(1,'%8.3f',deviations);fprintf(1,'\n');
end
fprintf(1,'pred_Pressure  ');fprintf(1,'%8.3f',pred.P);fprintf(1,'\n');
fprintf(1,'adjust_Pressure'); fprintf(1,'%8.3f',F.P); fprintf(1,'\n');
fprintf(1,'-----\n\n');

fid=fopen(filename,'wt');
if fid==-1,
    warning(strcat('can not open file',filename)); return; end
% -----Line 1 is read in Camile as a String line
fprintf(fid,'%@ %2d:%2d SP=[%5.2f;%5.2f]%%    CV=[%5.2f;%5.2f]%%\n',...
    timenow(3:4),SP(1)*100,SP(2)*100, ...
    Measurement(3)*100,Measurement(4)*100);
%-----Line 2&3 : Prediction
fprintf(fid,'%10.4f ',pred.top(1:end-1));
fprintf(fid,'%10.4f\n',pred.top(end));
fprintf(fid,'%10.4f ',pred.bot(1:end-1));
fprintf(fid,'%10.4f\n',pred.bot(end));
%-----Line 4&5 : pmm
fprintf(fid,'%10.4f ',pmm.top(1:end-1));
fprintf(fid,'%10.4f\n',pmm.top(end));
fprintf(fid,'%10.4f ',pmm.bot(1:end-1));
fprintf(fid,'%10.4f\n',pmm.bot(end));
%-----Line 6&7 : OptMvs
if ~StartNMPC, decisionV=zeros(size(decisionV));end
for kk=1:2,
    for i=1:size(decisionV,2)-1, fprintf(fid,'%10.3f ',decisionV(kk,i));
end
    fprintf(fid,'%10.3f\n',decisionV(kk,end));
end
%-----Line 8: funf
if StartNMPC,fprintf(fid,'%10.3f\n',sum([1 1 0.01].*deviations));
else fprintf(fid,'%10.0f\n',0);
end
%-----Line 9: predictionP
if length(F.P)>1,
    fprintf(fid,'%10.4f ',F.P(1:end-1));fprintf(fid,'%10.4f\n',F.P(end));
else fprintf(fid,'%10.4f\n',F.P);
end
%-----Line 10: pmmP
if length(pmm.P)>1,

```

```

fprintf(fid, '%10.4f, ', pmm.P(1:end-1));
fprintf(fid, '%10.4f\n', pmm.P(end));
else fprintf(fid, '%10.4f\n', pmm.P);
end
%-----
fclose(fid);

%+++++

function MPC=check(MPC)
global NNs;
%==== Check if MPC.PredPnts are subsets of NNs.PredPnts, correct if not
wholeset={};
wholeset{1}=NNs.top.PredPnts;wholeset{2}=NNs.bot.PredPnts;wholeset{3}=N
Ns.P.PredPnts;
subset={};
subset{1}=MPC.PredPnts.top;subset{2}=MPC.PredPnts.bot;subset{3}=MPC.Pre
dPnts.P;

for seti=1:3,
    whole=wholeset{seti}; sub=subset{seti};
    rec=[]; warn=0;
    for leni=1:length(sub),
        if isempty(find(whole==sub(leni))),
            warn=1;
            switch seti,
                case 1,
                    fprintf(1, strcat(' ???warning...!-
                                PredPnts_top_#', num2str(sub(leni))));
                    fprintf(1, blanks(2)); fprintf(1, 'was not trained\n');
                case 2,
                    fprintf(1, strcat(' ???warning...!---
                                PredPnts_bot_#', num2str(sub(leni))));
                    fprintf(1, blanks(2)); fprintf(1, 'was not trained\n');
                case 3,
                    fprintf(1, strcat(' ???warning...!---
                                PredPnts_P_#', num2str(sub(leni))));
                    fprintf(1, blanks(2)); fprintf(1, 'was not trained\n');
            end
        else rec=[rec leni];
        end
    end
end
if warn,
    switch seti,
        case 1,
            MPC.PredPnts.top=[];
            for setii=1:length(rec),
                MPC.PredPnts.top=[MPC.PredPnts.top, sub(rec(setii))];
            end
        case 2,
            MPC.PredPnts.bot=[];
            for setii=1:length(rec),
                MPC.PredPnts.bot=[MPC.PredPnts.bot, sub(rec(setii))];
            end
        case 3,
            MPC.PredPnts.P=[];

```

```

        for setii=1:length(rec),
            MPC.PredPnts.P=[MPC.PredPnts.P,sub(rec(setii))];
        end
    end
end
end

%Check if PredW is in the same dimension as PredPnts,if not, correct it
if length(MPC.PredW.top)>length(MPC.PredPnts.top),
    MPC.PredW.top=MPC.PredW.top(1:length(MPC.PredPnts.top));
else MPC.PredW.top=[MPC.PredW.top ones(1,(length(MPC.PredPnts.top)-
length(MPC.PredW.top)))]];
end

if length(MPC.PredW.bot)>length(MPC.PredPnts.bot),
    MPC.PredW.bot=MPC.PredW.bot(1:length(MPC.PredPnts.bot));
else MPC.PredW.bot=[MPC.PredW.bot ones(1,(length(MPC.PredPnts.bot)-
length(MPC.PredW.bot)))]];
end

%Check future MovePnts
if MPC.MovePnts(1)~=0,
    fprintf(1,' ???Warning: futureMV does not have control
            action for NOW,corrected!\n\n');
    MPC.MovePnts(1)=0;
end

```

D.2.6 Auxiliary Subroutines

```

function [fy,para]=MyFilter(Manner,y,y_old,fy_old,para)
% SYNTAX:
% [fy,para]=MyFilter('NONE',y,[],[],para);
% [fy,para]=MyFilter('first order',y,[],fy_old,para);
% [fy,cusum]=MyFilter('cusum',y,y_old,fy_old,cusum);
% global variable MPC is needed
%
global MPC;
switch lower(Manner),
case 'none', fy=y;
case 'first order',
    lamda=MPC.Lamda;
    fy=first_order(y,fy_old,lamda);
case 'cusum',
    win=MPC.cusum_win; Threshold=MPC.cusum_n;
    [fy,para]=cusumfilter(y,y_old,fy_old,para,win,Threshold);
otherwise, disp('this filter method not available');
end

%*****
function fy=first_order(y,fy_old,lamda)
fy=lamda*y+(1-lamda)*fy_old;

%*****
function [xspc,cusum]=cusumfilter(x,xold,xspc,cusum,win,Threshold)
cusum.k=cusum.k+1;

```

```

cusum.rou=1/(win-1)*(x-xold).^2+(win-2)/(win-1)*cusum.rou;
Sigma=sqrt(cusum.rou/2);
cusum.sum=cusum.sum+(x-xspc);
for i=1:length(x),
    if abs(cusum.sum(i))>Threshold*Sigma(i),
        xspc(i)=xspc(i)+cusum.sum(i)/cusum.k(i);
        cusum.k(i)=0;    cusum.sum(i)=0;
    end
end
end

%0+++++

function varargout=UnitConvert(value,fromto,addinfo)
%flowrateinlbmol=UnitConvert(flowrateinlb,'lb2lbmol',x);
%flowrateinlb=UnitConvert(flowrateinlbmol,'lbmol2lb',x);
%TinF=UnitConvert(TinC,'C2F');
%TinC=UnitConvert(TinF,'F2C');

% 'value' is the parameter to be converted
% 'fromto' is a string asking for the convert units
% addinfo is addition information needed for certain conversion,
% e.g. if convert the flowrate of a mixture, then mass fraction is
needed

nout=nargout;

if nargin>2, x=addinfo; end

if strcmp(fromto,'lb2lbmol'),
    amw=18.015*(1-x)+32.043*x; value=value./amw;
elseif strcmp(fromto,'lbmol2lb'),
    value=value.*x*32.043+value.*(1-x)*18.015;
elseif strcmp(fromto,'C2F'),
    value=value*1.8+32;
elseif strcmp(fromto,'F2C'),
    value=(value-32)/1.8;
else disp('unit conversion failed! I can not find a match!');
end

if nout==1, varargout{1}=value;
else for i=1:nout, varargout{i}=value(i);end
end

```

APPENDIX E
TUNING OF THE CONTROLLER – CV DAMPING VERSUS MOVE
SUPPRESSION

This section describes the preliminary test of comparison of two tuning approaches of MPC, the CV damping tuning approach and the move suppression tuning approach. This preliminary test shows certain advantages of using CV damping over move suppression.

E.1 CV Damping and Move Suppression for Tuning MPC

The CV damping and the move suppression are two common approaches for tuning MPC and are both practiced in industry. As summarized in the review paper of Qin and Badgwell (1997), some commercial linear MPC products such as DMC (Cutler & Ramaker, 1979) apply move suppression to prevent aggressive MV moves, while other commercial linear MPC products such as IdCom (Richalet, Rault, Testud, and Papon, 1979) use CV damping to avoid aggressive MV moves.

The difference between these two approaches is embodied in the general objective function as described in Equation (3.8) (rewritten as Equation (E.1) as follows). For the move suppression approach, $y_{ref}(i) = y_{sp}$, while the Move suppression factor, W_{MV} , acts as the tuning parameter. For the CV damping approach, W_{MV} is bypassed by setting it to zero while the controller is tuned by adjusting the dynamics of a predefined reference trajectory, represented by $y_{ref}(i)$, which is usually of first-order linear dynamics from the

present CV value, $y(k)$, to the setpoint, y_{sp} . Time constant of the first order dynamics is used as the tuning parameter.

$$J = W_{CV} \sum_i (\tilde{y}(k+i|k) - y_{ref}(i))^2 + W_{MV} \sum_j (\Delta u(k+j|k))^2 \quad (\text{E.1})$$

No literature was found to compare these two approaches. However, industrial practice on linear MPC seems to show no preference or obvious advantages of one approach over the other. However, preliminary tests of comparing these two approaches for the nonlinear distillation process in this work showed potential advantages of using CV damping over using move suppression, as described in the following section.

E.2 GNNMPC for the Distillation Process Using CV Damping

E.2.1 The CV Damping Approach

Simulation results described in Section 5.6.1 show that when using the move suppression approach to tune GNNMPC, the controller's performance is sensitive to the suppression factor, W_{MV} . W_{MV} has to be adjusted (manually in this work) in different operating regions to achieve desired control performance. When the setpoint of the top composition, y , and the bottom composition, x , is changed from $\{y,x\}=\{0.85, 0.03\}$ to $\{y,x\}=\{0.65, 0.005\}$, a W_{MV} value of 0.1 (Case 1 in Table 5.9) leads to instability while a value of 0.5 (Case 2 in Table 5.9) leads to good performance. However, when the setpoint is changed from $\{y,x\}=\{0.85, 0.03\}$ to $\{y,x\}=\{0.9, 0.08\}$, a W_{MV} value of 0.1 (Case 3 in Table 5.9) leads to good performance, implying that a value of 0.5 would lead to sluggish control.

In the following tests, CV damping is used for both cases of setpoint changes. The objective function is:

$$\underset{R_f, H_f}{Min} = \sum_{i \in y_p} (\tilde{y}(k+i) - y_{ref}(k+i))^2 + \sum_{i \in x_p} (\tilde{x}(k+i) - x_{ref}(k+i))^2$$

subject to:

$$[\tilde{y}(k+i), \tilde{x}(k+i)] = GNN(y(k-5), \dots, y(k), x(k-5), \dots, x(k), \dots, F(k), R(k-5), \dots, R(k+i-1), H(k-5), \dots, H(k+i-1)) + [dy(k+i), dx(k+i)]$$

$$[dy(k+i), dx(k+i)] = [y(k), x(k)] - GNN(y(k-5-i), \dots, y(k-i), x(k-5-i), \dots, x(k-i), \dots, F(k-i), R(k-5-i), R(k-1), H(k-5-i), H(k-1))$$

$$\Delta R(k+j) = R(k+j) - R(k+j-1)$$

$$\Delta H(k+j) = H(k+j) - H(k+j-1)$$

$$0 \leq \tilde{y}(k+i) \leq 1$$

$$0 \leq \tilde{x}(k+i) \leq 1$$

$$0 \leq R(k+j) \leq 1$$

$$0 \leq H(k+j) \leq 1 \tag{E.2}$$

As compared to Equation Set (5.10), where move suppression is used, the only difference in CV damping lies in the objective function. The reference trajectories for the top composition, y , as represented by $y_{ref}(k+i)$, and the bottom composition, x , as represented by $x_{ref}(k+i)$, are the trajectory of first order linear dynamics starting from the present value to the setpoint. The reference trajectories for the top composition and the bottom composition are,

$$\tau_y \frac{dy_{ref}}{dt} + y_{ref} = y_{sp} - y_0, \quad y_{ref,0} = y_0 \tag{E.3}$$

$$\tau_x \frac{dx_{ref}}{dt} + x_{ref} = x_{sp} - x_0, \quad x_{ref,0} = x_0 \quad (E.4)$$

Where τ_y and τ_x are the predetermined tuning parameters, y_0 and x_0 are the present value of the top composition and the bottom composition, respectively.

The solutions for equations (E.3) and (E.4) are:

$$y_{ref} = (y_{sp} - y_0)(1 - e^{-\frac{t}{\tau_y}}) + y_0 \quad (E.5)$$

$$x_{ref} = (x_{sp} - x_0)(1 - e^{-\frac{t}{\tau_x}}) + x_0 \quad (E.6)$$

The values at i -th discrete time interval in the future with a sampling time interval of T are:

$$y_{ref}(k+i) = (y_{sp} - y_0)(1 - (e^{-\frac{T}{\tau_y}})^i) + y_0 \quad (E.7)$$

$$x_{ref}(k+i) = (x_{sp} - x_0)(1 - (e^{-\frac{T}{\tau_x}})^i) + x_0 \quad (E.8)$$

Let $\lambda_y = e^{-\frac{T}{\tau_y}}$, and $\lambda_x = e^{-\frac{T}{\tau_x}}$. λ_y and λ_x are used in the simulation as the tuning parameter, which is between 0 and 1. The smaller the value of λ , the smaller the value of τ , the faster the reference trajectory to reach the setpoint, thus the more aggressive the controller.

E.2.2 Setpoint Tracking

The GNNMPC's control performance using CV damping approach for setpoint tracking at different operating points are shown in Figures E.1 and E.2. Figure E.1 shows the results when the setpoint changes from $\{y,x\}=\{0.85, 0.03\}$ to $\{y,x\}=\{0.65, 0.005\}$.

Figure E.2 shows the results when the setpoint changes from $\{y,x\}=\{0.85, 0.03\}$ to $\{y,x\}=\{0.9, 0.08\}$. For both cases, $\lambda_y = 0.7$ and $\lambda_x = 0.85$.

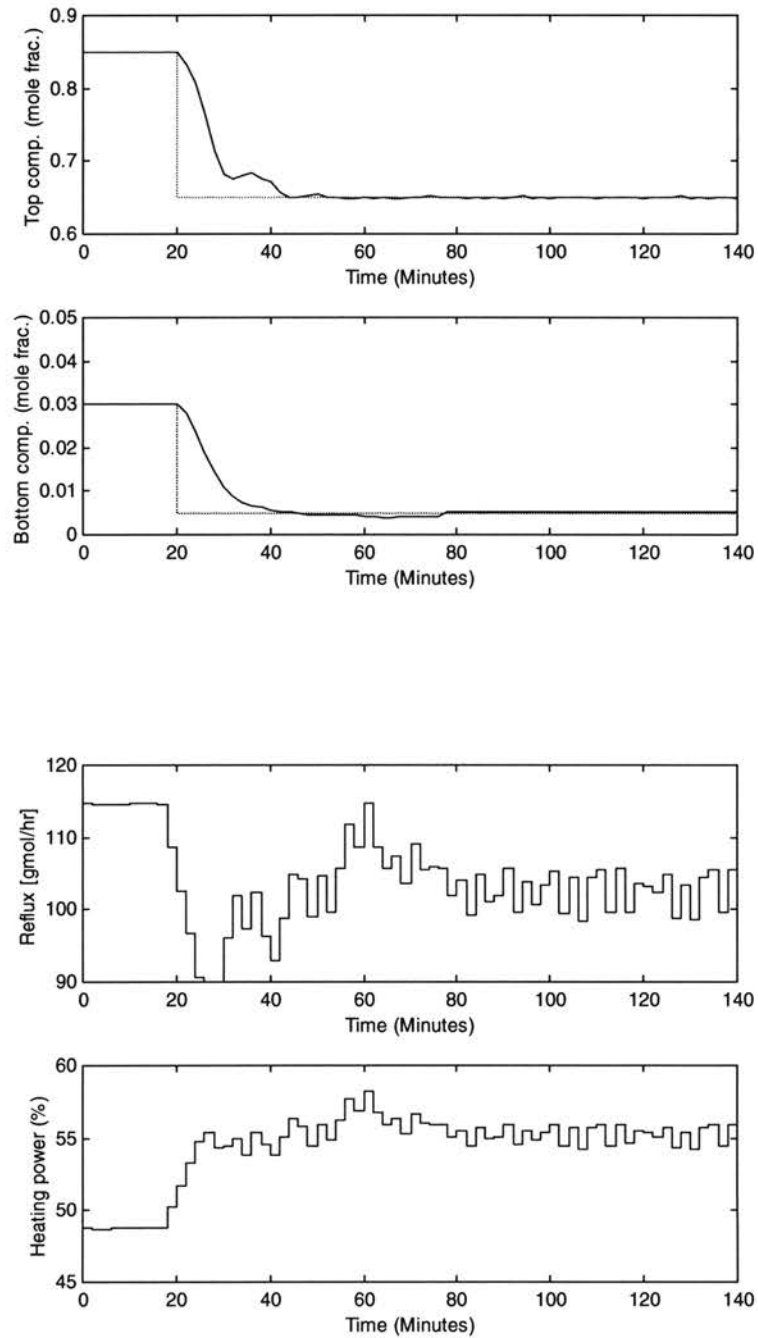


Figure E.1 GNNMPC for setpoint tracking ($\{y,x\}$ changes from $\{0.85, 0.03\}$ to $\{0.65, 0.003\}$)

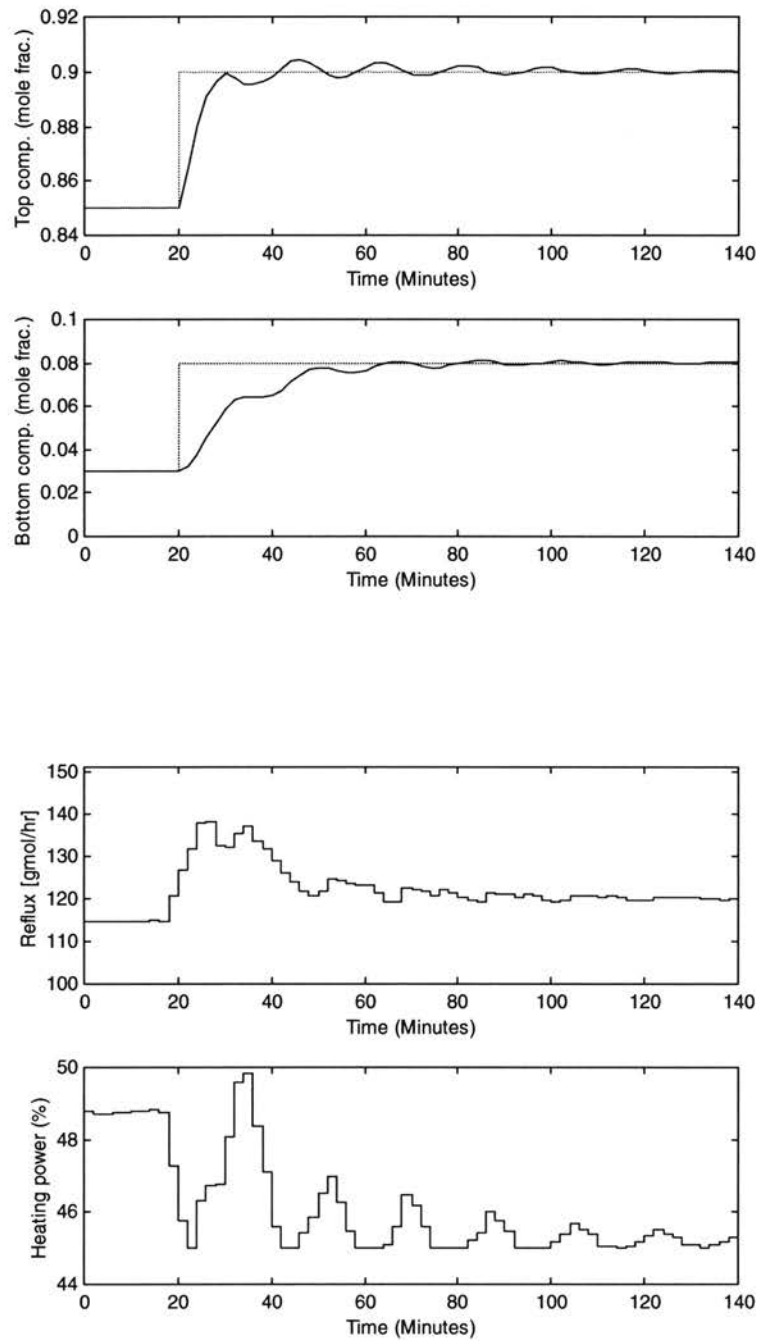


Figure E.2 GNNMPC for setpoint tracking
 ($\{y,x\}$ changes from $\{0.85, 0.03\}$ to $\{0.9, 0.08\}$)

In both cases, the controller is able to track the setpoint decently. Notice the MVs walk around due to the lack of constraint (penalty) on their moves. In addition, comparing with the cases studied in Table 5.9, the CV damping tuning approach is not sensitive to the operating point (which implies nonlinearity of the process), as the same values of the tuning parameters lead to good control performance in both cases. While as shown in Figures 5.16 to 5.19, when using move suppression method, the tuning parameters must be adjusted to achieve good control performance in these same two regions.

E.2.3 Disturbance Rejection

The following simulation results compare the disturbance rejection performance of GNNMPC for feedforward disturbance (feed flowrate in the case) and unmeasured disturbance (feed composition in the case) using the CV damping tuning approach as described in Section E.1.1. This investigation was motivated by the observation from the simulation results as described in Section 5.6.2 that the controller did no better in rejecting the feedforward disturbance than in rejecting the unmeasured disturbance when move suppression tuning approach was used, which is contradictory to the experience with a linear controller that a feed forward model leads to a much better disturbance rejection performance when the model is “good”.

Cases 5 to 7 as described in Table 5.10 are reinvestigated using CV damping approach. The simulation results are shown in Figure E.3 to E.7. In all cases, $\lambda_y = 0.5$ and $\lambda_x = 0.7$.

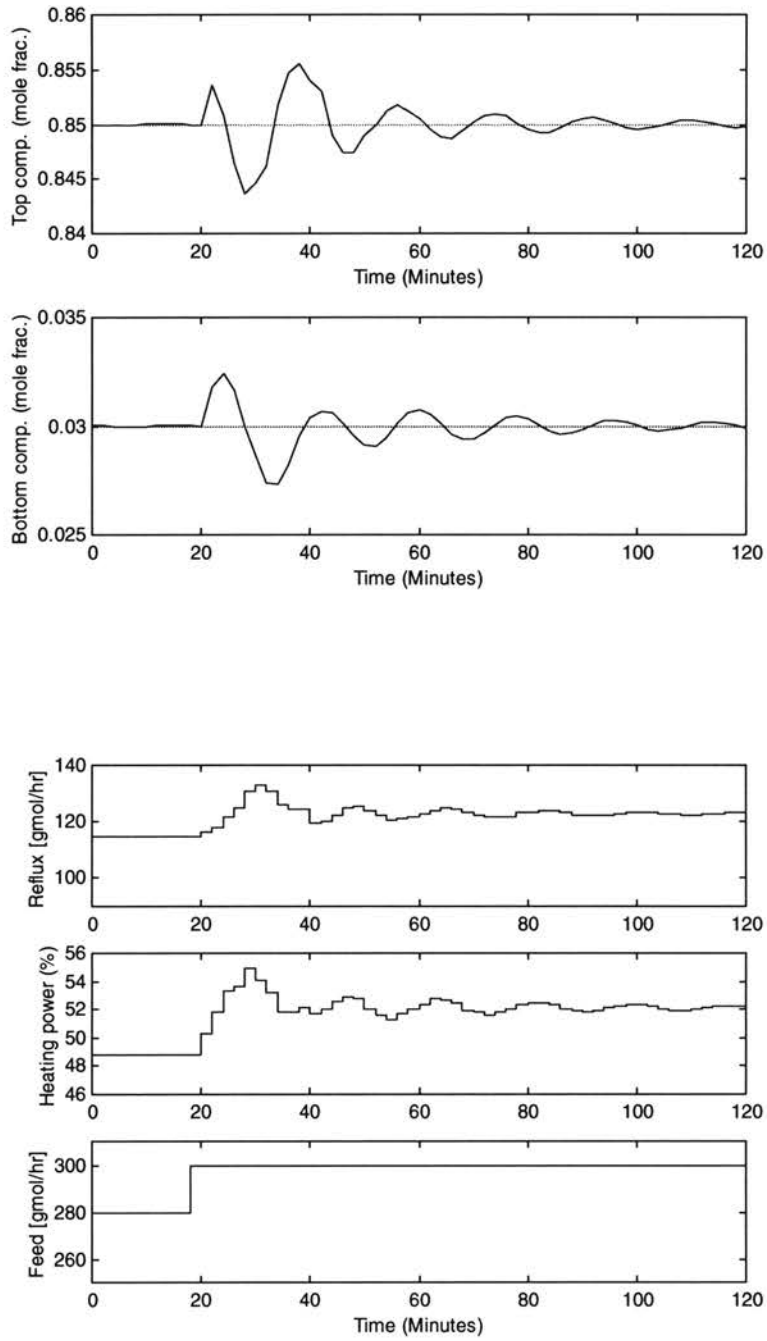


Figure E.3 GNNMPC for feedforward disturbance rejection
(feed flowrate changes from 280 mol/hr to 300 mol/hr)

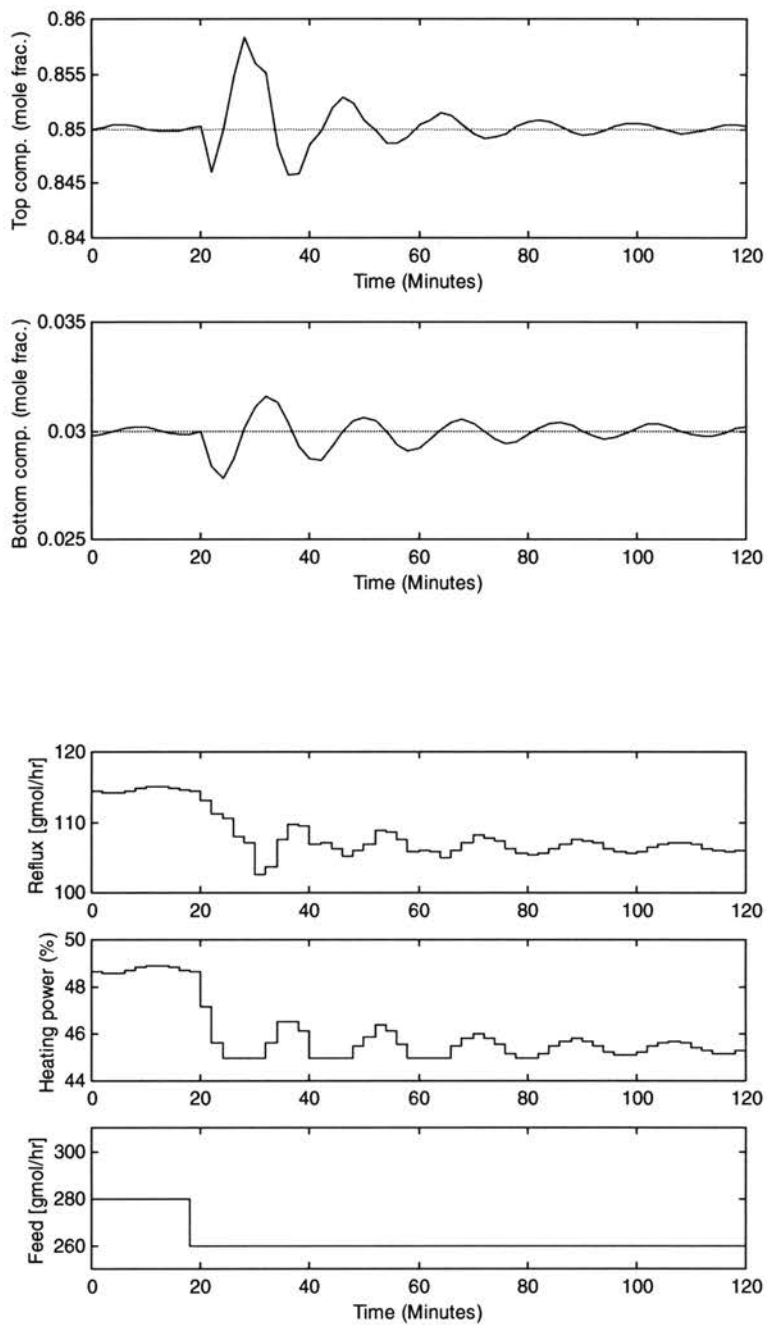


Figure E.4 GNNMPC for feedforward disturbance rejection
 (feed flowrate changes from 280 mol/hr to 260 mol/hr)

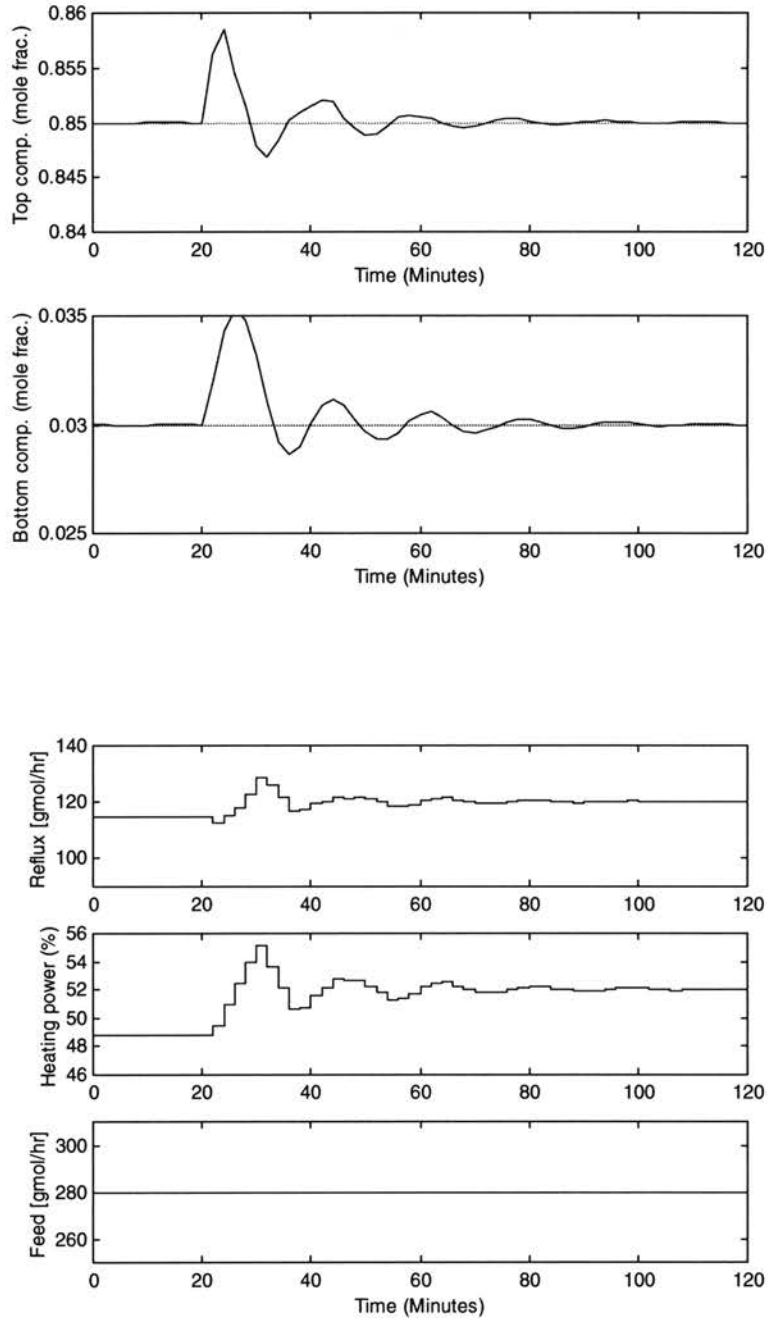


Figure E.5 GNNMPC for unmeasured disturbance rejection (feed composition changes from 0.25 mole fraction to 0.28 mole fraction)

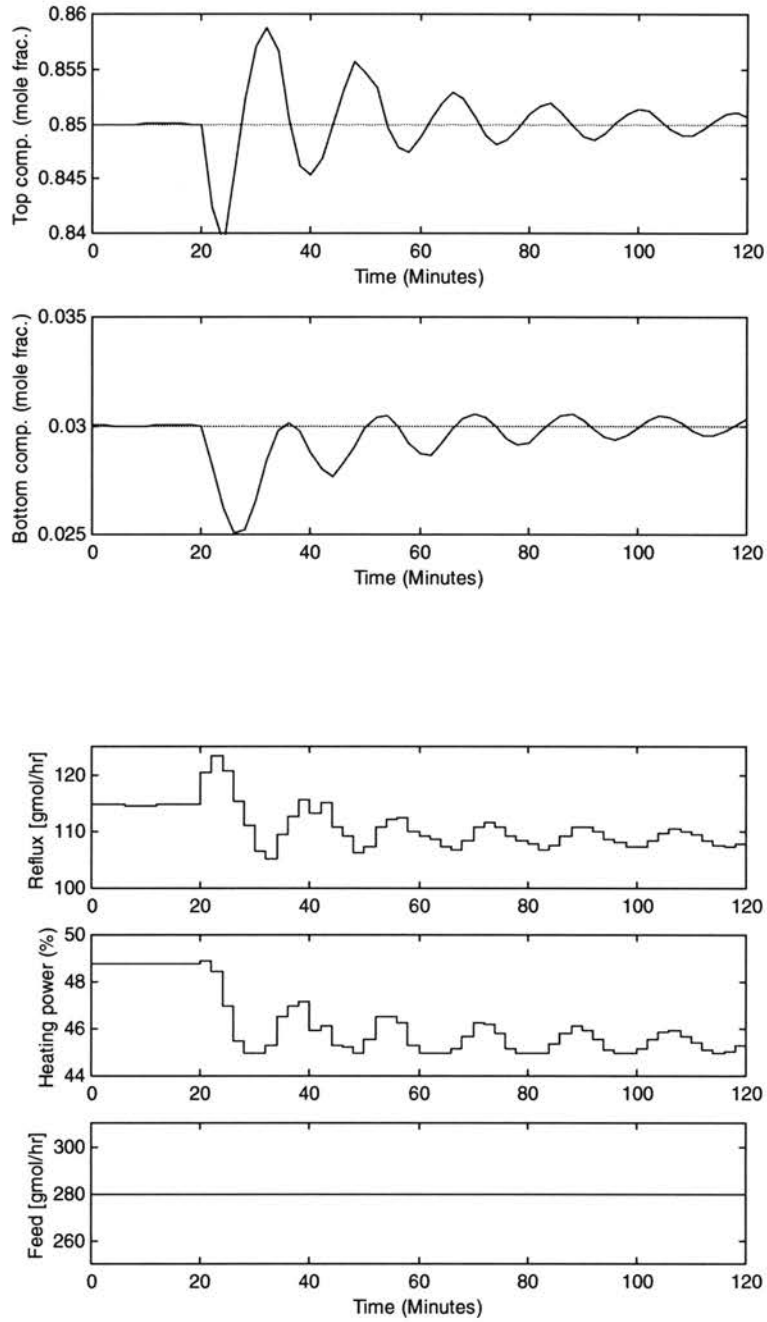


Figure E.6 GNNMPC for unmeasured disturbance rejection (feed composition changes from 0.25 mole fraction to 0.22 mole fraction)

For the feed flowrate disturbance of ± 20 mol/hr, the maximum amount of overshoot for the top composition is about 0.008 mole fraction and that for the bottom composition is about 0.002 mole fraction. While for the feed composition disturbance of ± 0.03 mole fraction, the maximum amount of overshoot for the top composition is about 0.012 mole fraction and that for the bottom composition is about 0.005 mole fraction. This shows better regulatory effect for feedforward disturbance than that for unmeasured disturbance.

E.3 Summary

This Appendix describes preliminary tests on using the CV damping approach for tuning GNNMPC, which shows that using CV damping approach has the potential of easier tuning and improved performance for disturbance rejection. Further investigations on comparing CV damping with move suppression are recommended for future work.

RECENT PAPERS

Ou, J., Rhinehart, R.R. (2001). "Grouped neural network model-predictive control", submitted to Control Engineering Practice on April 23, 2001

Ou, J., Rhinehart, R.R. (2001). "Grouped-neural network model for model predictive control", ISA Transactions, in press

Zhao, F.T., **Ou, J.**, and Du, W. (2000). "Simulation modeling of nuclear steam generator water level process - a case study", ISA Transactions, 39(2): 143-152

Zhao, F.T., **Ou, J.**, and Du, W. (2000). "Pattern-based fuzzy predictive control for a chemical process with dead time", Engineering Applications of Artificial Intelligence, 13(1): 37-45

Ou, J., Narayanaswamy, G., Rhinehart, R. R. (1998), "External reset feedback for generic model control", ISA transactions 37(3): 189-199

VITA

Jing Ou

Candidata for the Degree of

Doctor of Philosophy

Thesis: GROUPED NEURAL NETWORK MODEL-PREDICTIVE CONTROL AND
ITS EXPERIMENTAL DISTILLATION APPLICATION

Major Field: Chemical Engineering

Biographical:

Education: Received Bachelor of Engineering degree in Chemical Engineering and Doctoral degree in Control Science and Engineering from Zhejiang University, China in June 1992 and March 1997, respectively. Completed the requirements for the Doctor of Philosophy degree with a major in Chemical Engineering at Oklahoma State University in May 2001.

Experience: Employed as a graduate assistant in Zhejiang University from 1992 to 1997. Employed as a graduate assistant in Oklahoma State University from 1997 to present.