Dynamic Real-Time Optimization and Control of an Integrated Plant

A Thesis
Presented to
The Academic Faculty

by

Thidarat Tosukhowong

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Chemical and Biomolecular Engineering
Georgia Institute of Technology
December 2006

Dynamic Real-Time Optimization and Control of an Integrated Plant

Approved by:

Professor Jay H. Lee, Advisor
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Professor Matthew J. Realff
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Professor Martha Gallivan
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Professor F. Joseph Schork
School of Chemical and Biomolecular Engineering
Georgia Institute of Technology

Professor Shabbir Ahmed
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date Approved: 18 August 2006

To my parents, Soralux and Jurairat,

and my beloved brothers,

Thanawat, Thumrong, and Teera.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my advisors, Professor Jay H. Lee, for granting me a wonderful opportunity to conduct this interesting research and for his guidance and encouragement throughout the work. Throughout my study, he has been a role model of a good researcher possessing in-depth knowledge of the research field, a logical way of approaching problems, and superb technical writing and communication skills. This has been my motivation to always strive for perfection toward research and professional work. I also would like to acknowledge the helpful comments and advice I received from the professors who took the time to serve on my thesis committee: Professors Schork, Realff and Gallivan from the department, and Professor Ahmed from the School of Industrial and Systems Engineering.

I would like to acknowledge all of the past and present members of the Lee group (ISSICL). Jong Min Lee, Niket S. Kaisare, and Jaein Choi for showing me the ropes as senior students and for all the fun time we spent together in the group. Manish K. Gupta and Swathy Ramaswamy have been my great and dependable colleagues, who made my first arrival at Georgia Tech stable and enjoyable. And Anshul Dubey, Nikolaos Pratikakis, Wee Chin Wong, and Rakshita Agrawal for being nice and responsible junior students to work with. Over the past years, I have had great opportunities to get to know and to work with many visiting scholars. I also would like to thank them for their friendship, Dr. Jongku Lee, Dr. Kyung Joo Mo, Heejin Lim, and Prof. Gibaek Lee.

During my study, I also had wonderful opportunities to work for Weyerhaeuser as an engineering intern in the summers of years 2002, 2004, and 2005. The experience was of great benefit to my research and it motivated me to look at research problems from a more practical viewpoint. Besides that, I was more than fortunate to work with nice people. I would like to thank Dr. John Watkins and Gary Gover for giving me exciting opportunities to work on interesting projects in many different plant sites. I am greatful for the guidance

and practical discussions from Pami Singh and Dr. Philip P. Wisnewski. I really enjoyed my time working with them. I would also like to thank the other Process Information, Diagnostics, and Control group members who were of great help to me when I was in Federal Way, WA.

My colleagues in the department have always been good friends to turn to and a great diversion from work. They were always there when I needed some entertainment or some ethnic food. I thank them for our memories and friendship – Padmini Rangamani, Shabbir Husain, Kanrakot Beck, Preeti Chandra, Ashwini Sinha, Gracy Wingkono, Suchitra Konduri, and Yeni Hudiono. Thai students have always been there to share our fun time and to help each other. I especially thank them for always being there – Nantachai Kantanantha, Tiravat Assavapokee, Taweesup Apiwattanapong, Aphiwat Charnprasitphon, and Supaporn Erjongmanee.

I thank my uncle and aunts for their love and care. I also thank Nuttapong Srirattana and his family for their love, support, understanding, and encouragement throughout these years at Georgia Tech. And last but not least, I would like to thank my family for their support over the years. My brothers, Thanawat, Thumrong, and Teera, have always shown strong determination and endurance toward everything, which has inspired me since I was very young. My brother Teera has been very supportive to me in every way. He and his wife (Ms.Voraluk) have always taken great care of me any time I visited them in Washington. And to my parents, I would like to thank and express my deepest love to them from the bottom of my heart. They have always given me their endless love, but at the same time taught me to be strong and responsible. I would have not been the person I am today without their support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	**Description**
d	vector of disturbance variables
f_{eco}	plant-wide economic objective function
ϕ	stage-wise cost function
ϕ_t	terminal cost function
G_d	gain matrix of disturbance variables
G_u	gain matrix of manipulated variables
G_x	gain matrix of state variables
J	cost-to-go function
K	process gain
K_R	gain of the recycle process
p	prediction horizon
m	control horizon
n_u	number of manipulated variables
n_x	number of state variables
n_y	number of output variables
τ	time constant
u	vector of manipulated variable
x	vector of state variable
y	vector of output variable

Sub/superscript	**Description**
g	setpoints from the real-time optimizer
ls	setpoints from the least-square coordination layer
max	upper bound constraints
min	lower bound constraints
ss	steady-state condition

Abbreviation	**Description**
ADP	approximate dynamic programming
CV	controlled variable
DAE	differential algebraic equation
DP	dynamic programming
MPC	model predictive control
MV	manipulated variable
NMPC	nonlinear model predictive control
ODE	ordinary differential equation
POD	proper orthogonal decomposition
RTO	real-time optimization
SQP	sequential quadratic programming

SUMMARY

Applications of the existing steady-state plant-wide optimization and the single-scale dynamic optimization strategies to an integrated plant with a material recycle loop have been impeded by several factors. While the steady-state optimization strategy is very simple to perform, the very long transient dynamics of an integrated plant have limited the execution rate of the optimizer to be extremely low, yielding a suboptimal performance. On the other hand, the single-scale dynamic plant-wide optimizer that executes at the same rate as local controllers would require an exorbitant on-line computational load. In addition, it may be sensitive to high-frequency dynamics that are not relevant to the interaction dynamics of the plant, which are slow-scale in nature. This thesis presents a novel multi-scale plant-wide optimization strategy suitable for an integrated plant with recycle. The dynamic plant-wide optimizer in this framework executes at a slow rate in order to track the slow changes that are relevant to the plant-wide interactions and economics, while leaving the fast changes in unit operations to be handled by local controllers. Moreover, the computational requirement of solving the optimization problem will be much smaller than that of the single-scale dynamic optimizer running at a very high rate.

An important issue of the suggested method is in obtaining a suitable dynamic model for optimization. When dynamic first-principles models are available, model reduction techniques that reduce model order, while retaining slow-scale information in the frequency range of interest by the optimizer can be used. On the other hand, when fundamental constitutive equations are not available, system identification experiment needs to be performed to obtain information on the interaction dynamics of the system. The difficulties in this process are how to design input signals to excite this ill-conditioned system properly and how to handle the lack of slow-scale dynamic data when plant experiments cannot be conducted for a very long period of time compared to the plant's settling time. This work addresses the experimental design and suggests a new grey-box modeling method to incorporate steady-state information to improve model prediction quality.

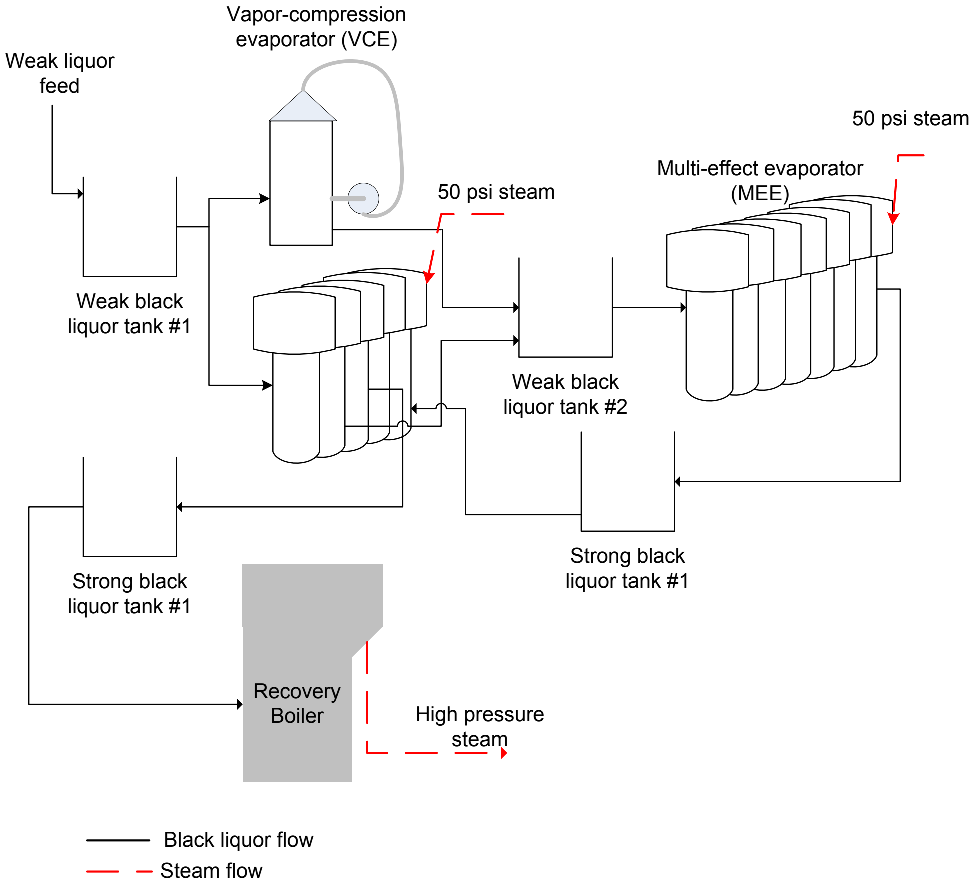
To extend this framework to a nonlinear integrated plant while ensuring a small on-line computational requirement and robustness against uncertainties, the Approximate Dynamic Programming (ADP) framework is adopted. This method offers advantages over conventional mathematical programming based approaches in that it can compute an optimal operating policy under uncertainties *off-line*. The on-line multi-stage optimization problem can be reduced to a single-stage problem, thus requiring much less real-time computational effort. In process system community, where the system has continuous state and action space, a simulation-based ADP method coupled with a function approximation scheme has been proposed. However, the existing ADP framework is inadequate to handle an integrated plant problem, which has a large action space and a high-dimensional system model. In this thesis, we use a case study to show the drawbacks of the existing mathematical programming framework and motivate the ADP approach. We combine a local gradient search technique and a nonlinear model reduction approach to overcome a very large off-line computational requirement of the existing ADP approach. The resulting framework shows superior performance in solving an optimal control problem of an integrated plant in both deterministic and stochastic cases and can be generalized to larger problems.

CHAPTER 1

INTRODUCTION

The objective of this thesis is to develop a novel dynamic plant-wide optimization and control framework for an integrated plant with a recycle loop. Such an integrated process system possesses very challenging process characteristics because this high-dimensional system is governed by both fast changes in individual process units, as well as slow changes from the recycle network. The presence of two time-scale dynamics leads to an ill-conditioned process model, which is difficult for process identification, control and optimization. The current state-of-the-art automation scheme in refineries typically has a fast-rate unit-based controller to optimize each unit operation separately, and a plant-wide real-time optimization (RTO) to find for all unit-based controllers in the plant the setpoints that maximize the economics of the plant's operations in the face of price variations of raw material and end products. In other process industries where the upstream unit operations can severely constrain the downstream process units, the plant-wide RTO system can be very beneficial in coordinating the entire plant and maximizing operating profit. Consider as an example, a pulp mill evaporation plant. There, weak black liquor consisting of chemicals and residual lignin from wood chip is evaporated by several evaporation stages as shown in Figure 1 in order to concentrate the black liquor until it can sustain combustion in the recovery boiler. The objective of this process is to supply the strong black liquor of desired concentration and production rate to the recovery boiler, while minimizing the electricity and the steam consumption. As a result, the plant-wide RTO is needed to coordinate the entire evaporation process in order to achieve the desired final liquor production in response to variable liquor supply, while minimizing the operating cost.

Currently, the RTO based on a steady-state plant model is the standard in the refineries. Due to the steady-state assumption, this RTO is only executed when the plant is near the steady-state condition. However, for an integrated plant with a large recycle loop and



Vapor-compression evaporator (VCE)
Weak liquor feed
50 psi steam
Multi-effect evaporator (MEE)
50 psi steam
Weak black liquor tank #1
Weak black liquor tank #2
Strong black liquor tank #1
Strong black liquor tank #1
Recovery Boiler
High pressure steam
Black liquor flow
Steam flow

Figure 1: Process schematic of the pulp mill evaporation plant

storage capacity, which are common in the chemical and the pulp and paper industries, the execution rate of the steady-state optimization can be severely limited because of the very long transient dynamics. The work in this thesis is based on the development of a dynamic plant-wide optimization scheme suitable for an integrated plant. The proposed optimizer is intended to *dynamically* track the dominant *slow* changes relevant to the plant-level interactions in the integrated plant. The main advantage of the new framework over the steady-state optimization is due to the fact that more dynamic degrees of freedom can be used to maximize the plant's economic performance. Outstanding issues related to dynamic optimization include obtaining a suitable dynamic model, handling of large on-line computational requirements, and ensuring robustness against uncertainties. Hence, this thesis brings forward a novel slow-scale dynamic optimization method designed for an integrated plant, presents a study on modeling of an integrated plant, and addresses practical issues associated with on-line computational time and handling of stochastic uncertainties.

1.1 Motivation

With an ever-increasing need for improving process economics, efficiency, and quality in a globalized market environment, plant-wide optimization schemes, such as the real-time optimization (RTO) technology have attracted the attention of process industries and has been adopted widely [2, 3]. The deployment of RTO in the refineries has been increasing and approximately 250 installations have been reported worldwide as of 1998 [4]. A typical RTO system is model-based and implemented on top of unit-based multivariate controllers, like the model predictive controllers (MPC). Although these two layers involve solving process-related optimization problems, they typically differ in the choice of the objective function and the scope of the problem. The objective function of the RTO is to optimize the plant operation based upon an economic performance measure under changing production schedule and other changes. This RTO layer takes into account the target from the production planning/scheduling layer and holistically calculates setpoints for all the multivariate controllers deployed throughout the plant. These local controllers function at high rates, typically in the range of minute(s), to steer their respective units to the given setpoints, while taking into account process constraints and local disturbances. Without the RTO, each unit-based controller may try to perform its best action in response to disturbances without realizing its effects on other unit operations and the economics of the entire plant operation.

Currently, the steady-state model-based RTO is the standard plant-wide optimization scheme in the refineries. One reason is that the steady-state plant model is easier to obtain than the dynamic model of the plant. Especially, for a large-scale integrated plant with recycle that has multiple time scales, identifying a complete dynamic model that is accurate across all frequencies is an excessively demanding task. In addition, for the steady-state model assumption to be valid, once the setpoints have been sent to the local controllers, the RTO has to wait until the entire plant settles down before the next execution commences

[2, 3]. For crude refinery plants where the process scheme is sequential, this steady-state requirement may not place a strong limitation on the execution rate of the RTO. However, in many integrated plant with material recycle loops, transportation delays, and large intermediate storage capacities, which are commonly found in the chemical processes and the pulp and paper industries, the transient dynamics of the plant often last several days. A good example illustrating the latter point is provided by the Tennessee Eastman process. This is one of the most well-known benchmark industrial chemical processes developed by Downs and Vogel [5], and consists of a reactor, a condenser, a stripper, a compressor, and a vapor/liquid separator with a recycle loop. While the residence time of each unit operation is less than half an hour and the recycle-to-feed ratio is about 2, the settling time of the plant has been shown in [6, 7] to last as long as 40 hours. In a reactor and a separator process with recycle studied by Wu and Yu [8], where the recycle-to-feed ratio is only 1.1 and the longest residence time of process is 2.5 hours, the transient dynamics of the plant also lasted 30 hours after a step change in the feed. Based on this, the application of a conventional steady-state RTO to an integrated plant can have several disadvantages. First, once a change occurs, it will take very long time for the plant to reach a new steady state, which limits the execution frequency of the RTO. In many situations, the plant seldom reaches the steady state due to additional changes or disturbances occurring in the meantime, so the usage of steady-state RTO may drop and the system may eventually be turned off [4]. Second, optimal operating conditions calculated from one predicting point at the steady state may not necessarily yield an optimal dynamic trajectory. In fact, it may even be infeasible at the local units due to several factors, such as the transient dynamics, model errors, or local disturbances. Third, the steady-state assumption automatically precludes the use of dynamic degrees of freedom present in the storage capacities of various units, leading to suboptimal solutions.

1.2 Issues in Applying Existing Dynamic Real-Time Optimization Methods

To remove the limitations in the steady-state RTO, some researchers have embraced the idea of performing the plant-wide dynamic optimization at the same frequency as the local unit-based controllers. One approach uses the model predictive control (MPC) formulation to perform both the control and the economic optimization functions by adding the economic objective term to the quadratic objective function used by MPC [9, 10, 11]. The resulting optimizer could respond to changes in the plant condition faster than does the steady-state RTO [10, 11]. However, this method is applicable to plants that are not very complex, such as an oil blending application, where a linear model is suitable to describe the plant and the settling time is relatively short. However, in many large-scale integrated plants with recycle loops, the systems are much more complex due to the presence of many highly nonlinear unit processes (e.g. distillation columns) and multiple time-scale problems. To ensure satisfactory optimizer performance in the face of long settling time, this *single-scale* fast-rate dynamic optimization and control scheme needs to have very long prediction horizon, thereby leading to a very large on-line optimization problem to be solved in a very short sample time. Furthermore, the problems of ill-conditioning and nonlinearity impose additional computational load in generating a multi-step output predictions.

Another single time-scale plant-wide optimization and control framework proposed in the literature is the single-point dynamic RTO scheme as used in [1]. In this approach, the RTO is separated from the MPC layer, but both are executed at the same rate. In order to reduce the on-line computational requirement of solving a plant-wide optimization problem at a fast rate, the RTO uses only one prediction point in the economic optimization. Nonetheless, because the optimal solution is determined with respect to a single time instance, it gives no guarantee that the entire dynamic trajectory is optimal. Moreover, one can expect significant model uncertainties and disturbances in the high-frequency range. This may lead to a robustness problem in the single-scale fast-rate RTO scheme.

Based on the aforementioned issues in existing plant-wide optimization techniques, this research proposes a multi-scale plant-wide optimization and control framework, where the plant-wide optimizer performs a dynamic optimization based upon multiple prediction points, but executes at a rate significantly lower than that of the local controllers. This slow execution rate allows the optimizer to respond to the slow changes that are relevant to the plant-wide interactions and economics, while leaving the fast changes in unit operations to be handled by local controllers. In addition, the use of multiple prediction points ensures that the performance along the dynamic trajectory is evaluated, unlike the single-point optimization approach, whose performance can be sensitive to the choice of the optimizing point. Nonetheless the applicability of this approach to large-scale integrated plants can still be limited by the followings:

1. **Difficulties in obtaining the dynamic model:** In practice, a complete first-principles dynamic model of the integrated plant is not always available and the dynamic model for optimization has to be obtained from identification experiments. When the plant has a recycle loop, the long-lasting slow-dynamics often complicates the system identification experiment. Literatures on identification often assume that model identification experiment can be performed over a long period of time, such as more than three days [12, 13]. However, in general situations the experiment may be limited to a much shorter period of time. In this case, the obtained model may be accurate in describing the initial response of the plant, but unable to capture the slow dynamics of the system accurately. Many practitioners try to overcome the inaccuracy in the model's long-term prediction by truncating the prediction horizon to a short period of time, thereby giving up the ability to make a long range prediction on the plant's output. As a result, there is a lot of room for improvement in the modeling of an integrated plant. On the other hand, in practice, steady-state description of the plant is usually available, such as from material balances, thermodynamic equations, flowsheet simulator, etc. Therefore, a dynamic model that is suitable for a long-range prediction should be developed by incorporating the initial dynamic information from the identified model and prior knowledge about the plant's steady state gains. Ultimately, the prediction from the resulting model should eventually converge to the steady-state that is known a priori.

2. **Handling of large on-line computational load and uncertainty:** Currently, the most commonly used approaches to solve dynamic optimization problems in MPC and RTO are based on mathematical programming frameworks. In these methods, dynamic models, state information and disturbances measured up to the current time are used to build a prediction of the future output of the plant, from which the on-line optimizer finds a sequence of manipulated variables that optimizes the objective function. Despite this popularity, there are two outstanding limitations of the mathematical programming based approaches. First, the on-line computational load of solving an on-line optimization problem can be very large when applied to a high-dimensional nonlinear integrated plant. This is because the optimization time depends on the time required to solve the system model to generate the output prediction and the length of the prediction horizon. In process industries, plant models often take the form of stiff and high dimensional ordinary differential equations (ODEs) or differential algebraic equations (DAEs). In addition, the long transient dynamics demand the use of long prediction and control horizon to ensure satisfactory performance. Given this, on-line optimization via mathematical programming can present a computational challenge. The second limitation of this framework is in the handling of uncertainty. Uncertainty is a crucial part of a plant-wide optimization problem of process industries. However, the current mathematical programming based approaches mainly solves at each sample time a *deterministic open-loop* optimal control problem, which precludes the considerations of stochastic uncertainty and future feedback. As a result, they are inherently suboptimal in situations where uncertainty and feedback are present.

   Alternatively, both issues can be addressed by the approximate dynamic programming framework (ADP). This approach offers a way to compute the approximate of an optimal feedback control policy for a multi-stage dynamic optimization off-line. The resulting 'cost-to-go' function of ADP can reduce an on-line multi-stage optimization problem into an equivalent single stage problem. As a result, the on-line optimization time dramatically decreases in this framework. In the process control community, applications of this approach are still in an early stage and are limited to problems with a small action space. The current state of the art of this approach presented by Lee and coworkers [14, 15] still has difficulties when directly applied to an integrated plant. In particular, it can suffer from an exponential growth in computation when applied to a problem with a large action space. In addition, when the process model is highly nonlinear and is of high order, the off-line computational load can be very large because of the large number of model simulations required to solve such a complex model to obtain the cost-to-go values. Given the fact that many variables of the chemical processes are often highly correlated, it is possible to derive a significantly lower-order plant model using an appropriate model reduction technique in order to reduce the simulation time. As a result, to overcome the drawbacks of existing mathematical programming framework, a systematic methodologies to bring together the existing ADP framework, the model reduction technique, and other tools are required in order to develop an adequate framework to solve the optimization problem of an integrated plant under uncertainties.

1.3 Thesis Objectives and Outline

In summary, the model-based RTO system attempts to find the setpoints for all the unit-based controllers in the plant that maximize the plant-wide economic objectives. Therefore, it can be applied not only to the refineries, but also to many other processes that have some of the following characteristics:

1. Processes that involve several final products, where a product allocation is an important decision to maximize the profit of the plant. This situations can be found in many chemical plants where it is important to maximize the yield of the valuable products in the face of variable raw material compositions and price fluctuations.

2. Processes whose upstream processes constrain the economic of the downstream units. For example, in semiconductor industries, the processing steps are very sequential and several process variables along the production line can be adjusted to change the quality of the final products. Therefore, the fab-wide RTO can be very useful in

finding dynamic setpoints for the upstream processes in order to achieve the target product quality and to minimize the processing cost.

3. An integrated plant with recycle, where the upstream processes can affect the downstream, and vice versa. As a result, the plant-wide RTO is very crucial in coordinating the upstream and the downstream processes in order to optimize the operation objective of the entire plant.

In this thesis, we proposes a novel multi-scale dynamic optimization and control scheme, specifically for solving an optimization of an integrated plant with recycle to overcome the limitations of the existing plant-wide optimization and control methods. For successful application of this framework, a systematic grey-box identification technique that provides reliable long-range prediction capability is developed. In addition, this work shows how to apply nonlinear optimal control to a large-scale integrated plant with stochastic uncertainty by combining the existing ADP, model reduction, and other methodologies. Note that this research does not limit the plant-wide optimizer to necessarily be an RTO that optimizes the economic objective of the plant. Instead, we are interested in a more general case where the plant-wide optimizer may have the control objective (e.g. to optimize the grade transition), or the economic objective (like in the RTO). Nonetheless, the literature review of the plant-wide optimization strategy will be based upon the RTO technologies due to the fact that there exists a wealth body of research in this area.

The remainder of this thesis is organized as follows. In Chapter 2, an overview of the existing model-based real-time optimization system is provided. In particular, we discuss the formulations of the steady-state RTO, and the single-scale dynamic RTO. To motivate the work, we address advantages and potential disadvantages of each approach. Although, these methods may have been applied to optimize the refineries, there is still much room for improvement when considering the applications to an integrated plant system.

In Chapter 3, the slow-scale dynamic optimization framework for an integrated plant is suggested. The proposed framework optimizes multiple prediction points, but executes at a rate significantly lower than that of the local controllers. Notable issues involve choosing an appropriate optimization frequency and designing the interface between the plant-wide optimizer and the unit-based control layer in a way that will guard against the transfer of infeasible setpoints to local controllers. A performance comparison between the suggested approach and previous approaches is also illustrated in two examples. The first example consists of two interacting linear processes studied by Lu [1]. In the second example is a system of a reactor, a storage tank, and a separator with recycle. We intentionally chose a very high recycle-to-feed ratio for this problem to accentuate the multiple time-scale and the ill-conditioning characteristics of the integrated plant. Nonetheless, in other processes where the recycle-to-feed ratios are smaller, the two time-scale problems can still be very pronounced, such as in the Tennessee Eastman challenge problem discussed previously.

In Chapter 4, we propose a dynamic model of an integrated plant that is suitable for the plant-wide optimization. We are interested in practical cases where identification experiment is limited to a much shorter period of time than the plant's largest time constant, but prior knowledge about the plant's steady state gains is available. Since the experiment is relatively short, the dominant pole (or the longest time constant) of the system will not be captured by the model, making it unreliable for the long range prediction. Therefore, our approach is to parameterize the identified model as a step response model truncated at the time when the prediction accuracy starts to degrade. Then the residual dynamics are approximated as a first-order system and augmented to the step response model, while ensuring that the settling gains of the model match up with the steady-state gains from the prior knowledge. The results show that a plant-wide optimizer using the augmented model has better performance than the one using an original identified model from the short experiment.

In Chapter 5, a representative case study of an integrated plant with a reactor, a distillation, and a material recycle loop is presented. This example shows the characteristics of most integrated plants that have large state and action spaces, as well as a multiple time scales behavior. We use this case study to present the inherent drawbacks of a conventional nonlinear model predictive control (NMPC) and point out the issues of the existing ADP method. From this example we can infer that the existing ADP framework requires an excessive off-line computational load due to the high dimensional action space and the computational time in approximating the cost-to-go function. In additional, because an interpolation scheme used to approximate the cost-to-go values is a function of the state, it is important to identify state variables that are relevant to the cost-to-go values and apply appropriate weighting factors to them in order to obtain a reliable cost-to-go approximation. Based on these considerations, the work presents a heuristic to restrict the search space for the optimal action within a region defined for each state by the good suboptimal control policy used in the simulation, and incorporates the nonlinear model reduction technique and data analysis to improve the efficiency and the accuracy of the cost-to-go learning step. These methodologies successfully apply a nonlinear optimal control to the integrated plant example and show a superior performance and a drastic reduction in on-line computational load compared to the conventional NMPC in both a deterministic and a stochastic cases.

Finally, Chapter 6 summarizes the contributions of this thesis and suggests possible directions for future work.

CHAPTER 2

PRELIMINARIES

In this chapter, we explain typical behaviors of an integrated plant with recycle, which show a two time-scale characteristics with fast initial changes followed by slow transient dynamics. Then, we present the architecture of the steady-state RTO in lieu of the steady-state model-based plant-wide optimization strategy, and identify its potential disadvantages. Finally, the alternative single-scale dynamic RTO formulations that have been proposed in the literature are presented.

2.1 Dynamic Behaviors of an Integrated Plant with Recycle System

Recycle streams are commonly found in the process plants nowadays. During the last two decades, several researchers have devoted their studies on the dynamic behaviors of the process with recycle in order to propose suitable regulatory control schemes for this system (see [16, 17, 18, 8, 19] for examples and references of research in this area). In this work, we are interested in applying a plant-wide optimization strategy to an integrated. Therefore it is important to understand the behaviors of this process and address key issues in dealing with this system.

One of the earlier works that provide insight into the behavior of the system with recycle is from Luyben [16], which considered the open-loop behavior of the system at different values of the recycle loop gains. The recycle system was represented by a process model shown in Figure 2. The forward path has a simple first-order dynamics with a steady-state gain K and a time constant τ. The recycle is of first order with a gain K_R and a time constant τ_R. The output and the input variables are denoted by y and u, respectively. This paper studied the eigenvalues, which are the inverse of the system's time constants, as the recycle gain varied. The system's transfer function is represented by Eq. 1 and the

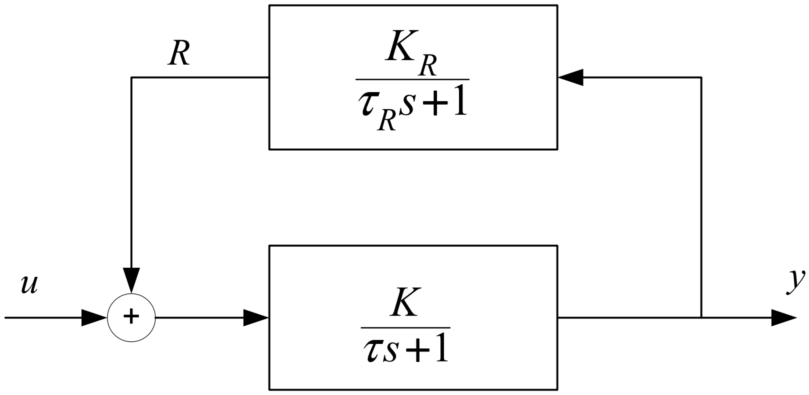


Figure 2: Simple open-loop process with recycle

characteristic equation of the open-loop system is shown by Eq. 2.

$$\frac{y}{u} = \frac{K(\tau_R s + 1)}{\tau\tau_R s^2 + (\tau + \tau_R)s + 1 - KK_R} \tag{1}$$

$$\tau\tau_R s^2 + (\tau + \tau_R)s + 1 - KK_R = 0 \tag{2}$$

Suppose $K = 1, \tau = 1$, and $\tau_R = 1$. When the recycle gains are varied, the eigenvalues are observed as follows:

- Case 1: $0 < KK_R \leq 1$. Many plants with material recycles fall into this category. As K_R approaches 1, one of the eigenvalues moves closer to the origin as shown in Figure 3, while the other eigenvalue is pushed away to a more negative territory. In other word, one of the time constants becomes larger as K_R increases from 0 to 1, and the other time constant exhibits a very fast response. This answers why an integrated plant with a material recycle displays a *two time-scale* behavior that shows a fast initial response to a step change and followed by a very slow transient behavior. Note that when KK_R is 1, the process is a pure integrator.

- Case 2: $KK_R > 1$ In this case one of the eigenvalues is positive as shown in Figure 3, making the system unstable. This unstable plant should be stabilized before the plant-wide optimizer is applied. If the gain K is not too large, installing a controller in the

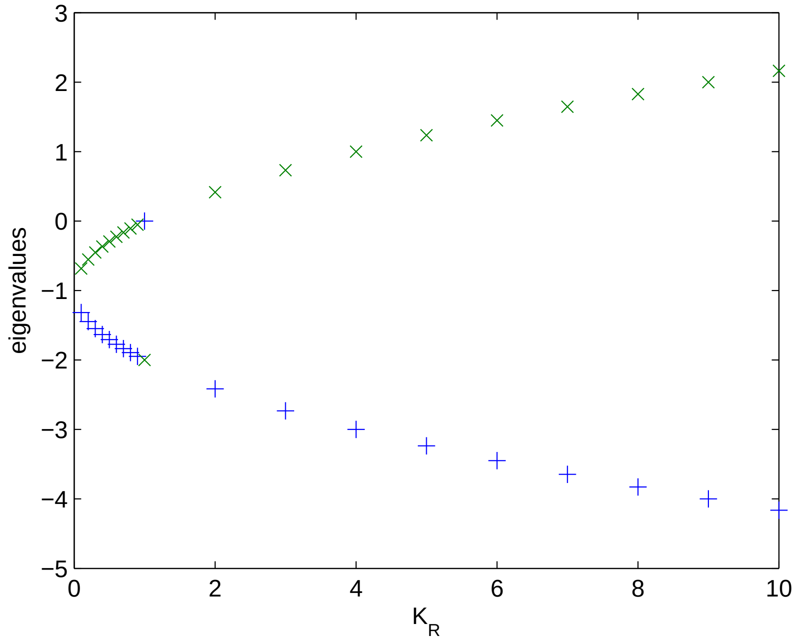


Figure 3: Eigenvalues of the open-loop system for the case $0 < KK_R \leq 10$

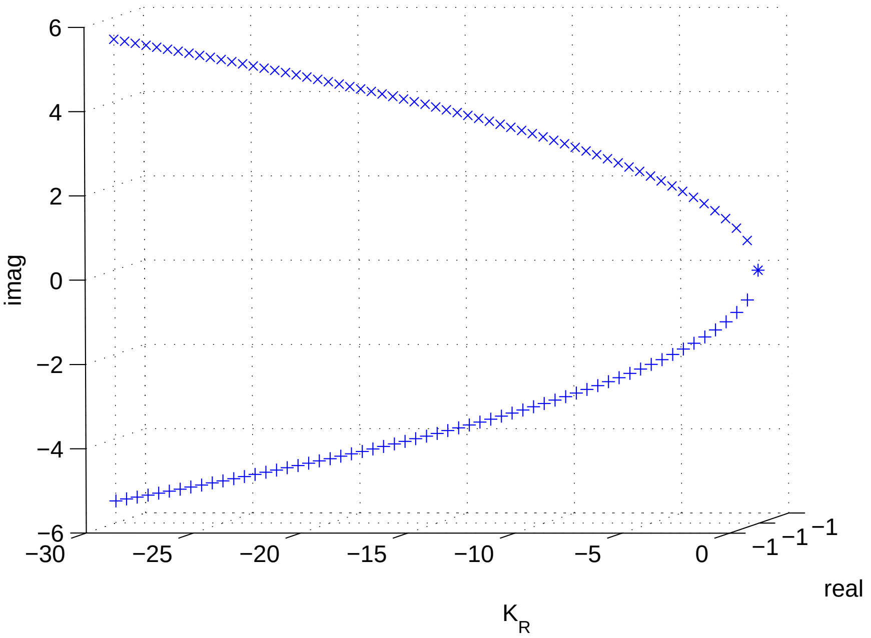


Figure 4: Eigenvalues of the open-loop system for the case $KK_R < 0$

forward path may make the system stable if it can make the resulting effective gain in the forward path be less than 1 [16]. However, if the system cannot be stabilized, a plant redesign should be considered.

- Case 3: $KK_R < 0$ In this case, the eigenvalues are complex conjugates as shown in Figure 4. The real part of the eigenvalue is -1 (for $\tau = 1, K = 1$) and the imaginary part moves further away from zero as K_R becomes more negative. This system would exhibit an underdamped behavior with higher oscillatory characteristic as K_R is more negative. Note that the systems with negative recycles are not common [16], but might be found in a unit operation that uses energy from the product stream to heat the feed (see examples in [16, 20]).

In this work we focus on a more general case of an integrated plant with a dominant material recycle loop that exhibits long transient dynamics as shown in case 1, which is commonly present in the process industries. This two-time scale behavior poses challenges to the plant-wide optimization as will be described in the next section.

2.2 Existing Plant-wide Optimization and Control Techniques

A typical automated decision hierarchy in the refineries consists of several cascade layers as shown in Figure 5.

The lowest control layer includes regulatory controllers, which are typically the single-input-single-output (SISO) controllers, such as level controllers, pressure controllers, etc. They are in charge of eliminating effects of disturbances on each output, and hence must be executed at a very fast rate. Multivariate unit-based controllers, such as the Model Predictive Controller (MPC), are deployed in major process units to supervise several regulatory controllers in order to drive the control variables (CVs) of respective units to the given setpoints under various process constraints, arising from equipment safety, output quality requirement, etc. Therefore, the execution rate of the MPC is typically in the time scale of minute(s). Typical optimization problem solved at each time step of the MPC is shown in Eq. 3 where a dynamic model is used to build a prediction of the future output, based

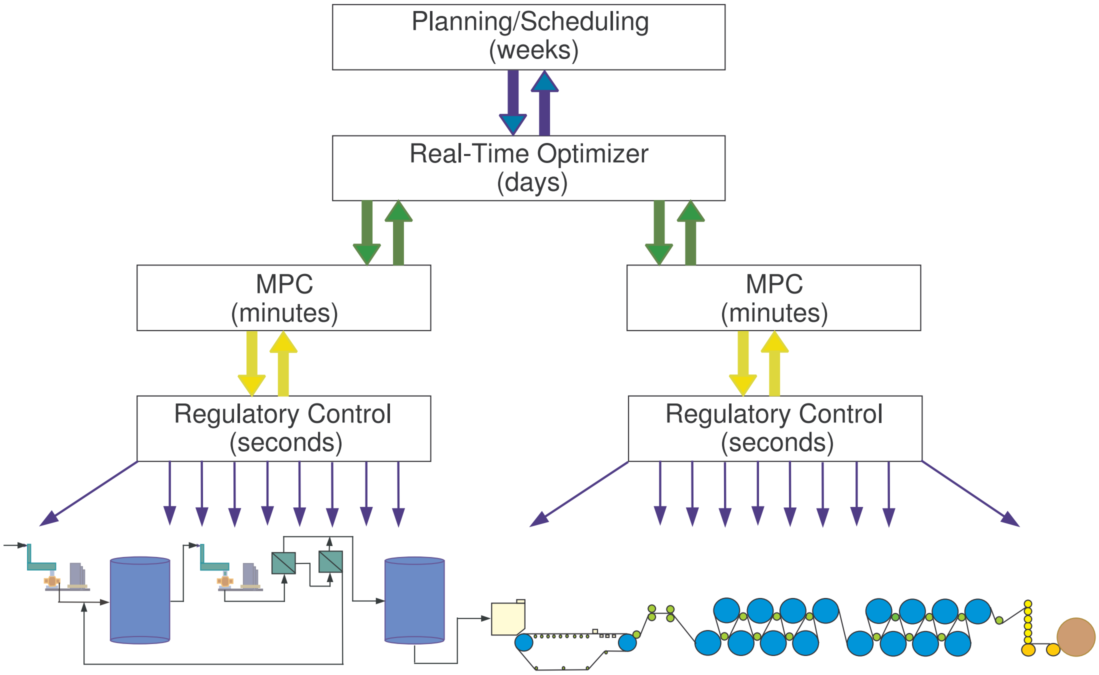


Figure 5: Typical plant automation hierarchy

upon which an open-loop optimal input profile is computed.

$$\min_{\Delta u(0),\ldots,\Delta u(m-1)} \sum_{i=1}^{p} \|Q\left(y_s - y(i)\right)\|_2^2 + \sum_{j=0}^{m-1} \|R\Delta u(j)\|_2^2 \tag{3}$$

subject to

$$y_{min} \leq y(i) \leq y_{max}, \quad i = 1, \ldots, p$$

$$\Delta u_{min} \leq \Delta u(j) \leq \Delta u_{max}, \quad j = 0, \ldots, m-1$$

where y_s is a vector of output setpoint specified for the MPC controller. It may be manually set or given by the higher automation layer. The output vector y is predicted out to several time instances i. The prediction and the control horizons are denoted by p and m, respectively. Weighting matrices Q and R are tuned to balance between the output tracking performance and the aggressiveness of the input moves. In addition, local constrains of the output and input variables are denoted by u_{min}, u_{max}, and y_{min}, y_{max}, respectively.

Above the unit-based control level, there typically exists an optimization layer that considers the entire plant operation. The plant-wide optimizer whose objective function is to optimize the economics of the plant explicitly is known as the *real-time optimizer (RTO).* It utilizes a plant model to determine the manipulated variable actions that steer the entire plant to the economic optimum. The solution from the RTO is then sent to the control layer to implement. A general optimization problem in the RTO can be written as a following nonlinear program

$$\max_{y_g, u_g, x} f_{eco} \tag{4}$$

subject to

$$\begin{aligned} h(y_g, u_g, x) &= 0 \\ g(y_g, u_g, x) &< 0 \\ y_{g,min} \leq y_g &\leq y_{g,max} \\ u_{g,min} \leq u_g &\leq u_{g,max} \end{aligned}$$

where f_{eco} is the economic objective function, y_g is a vector of calculated global setpoints for the controlled variables, u_g is a vector of calculated global setpoints for the manipulated variables, x is a vector of state variables, h is a set of equality constraints, and g is an additional set of inequality constraints. Lower bounds and upper bounds of controlled variables are denoted by $y_{g,min}$ and $y_{g,max}$, respectively, whereas those of the manipulated variables are $u_{g,min}$ and $u_{g,max}$, respectively. In RTO, the above problem is repeatedly solved at an appropriate frequency, with the model state and parameters being updated before each optimization.

Currently, several real-time optimization schemes have been proposed, which can be divided into two major categories, namely the steady-state RTO, and the single-scale fast-rate dynamic RTO.

2.2.1 Steady-State Real-Time Optimization

In current industrial practice, RTO is mostly based on a steady-state model, using only steady-state gains to calculate output y_g in solving the optimization problem (4). The frequency of optimization depends on the settling time of the entire process, since this approach requires the process to be near a steady state before performing data reconciliation, parameter estimation, and optimization [3, 21]. The main advantage of this approach is that it requires less modeling and on-line computational efforts. The former is because steady-state description of the plant is in general much easier to develop than dynamic models; whereas the latter is because the on-line optimization is only done at only a single predicted future point (i.e. the steady-state), and the computational time is allowed to be long due to the fact that the window of the settling time of the plant is much longer. However, the restricted frequency of optimization in this approach results in the shortcomings, which include the limited utilization of the dynamic degrees of freedom in the plant as well as the discrepancy of the steady-state prediction and the dynamic behavior of the plant.

2.2.2 Single-scale Dynamic Real-Time Optimization

To overcome the shortcomings of the steady-state approach, many researchers have adopted the idea of performing the plant-wide optimization at the same frequency as the local unit-based controllers. The plant measurement and the state update for the plant-wide optimizer are all performed at the fast time scale of minutes. In this section, we highlight two formulations based on this framework that have been proposed in the literature.

2.2.2.1 Receding-Horizon Optimization and Control Method

In this approach the RTO and the model predictive controller (MPC) are combined into a single entity [9, 10, 11]. The architecture of this approach can be represented as in Figure 6.

The formulation of this approach from [10] is similar to the MPC formulation except that the economic objective function is added to the typical quadratic control objective term used by the MPC as shown in Eq. 5.

$$\begin{aligned}\min_{y_s,u_s,\Delta u(0),\ldots,\Delta u(m-1)} W_1 f_{eco} &+ \sum_{i=1}^{p} \| W_2 \left(y_s - y(i) \right) \|_2^2 \\ &+ \sum_{j=0}^{m-1} \| W_3 \Delta u(j) \|_2^2 + \| W_4 (u(0) + \sum_{j=0}^{m-1} \Delta u(j) - u_s) \|_2^2 \end{aligned} \tag{5}$$

subject to

$$u_s = u(0) + \sum_{j=0}^{m-1} \Delta u(j)$$

$$h(u_s, y_s, x) = 0$$

$$y_{min} \le y(i) \le y_{max}, \qquad i = 1, \ldots, p$$

$$u_{min} \le u(j) \le u_{max}, \qquad j = 0, \ldots, m-1$$

$$\Delta u_{min} \le \Delta u(j) \le \Delta u_{max}, \qquad j = 0, \ldots, m-1$$

where y_s and u_s are the calculated output and input vectors at the steady-state condition. The prediction and the control horizons are denoted by p and m, respectively. In this scheme, the weighting factors, $W_1 - W_4$, must be carefully tuned to strike the balance between the importance of the control and the economic objective. The optimizer formulated by this method has been applied to optimize the fluid catalytic cracking (FCC) converter used in the LPG production and shown to respond to changes in the economic objective much faster than does the existing steady-state RTO. However, because the economic and the control objectives are formulated together, its economic performance can be very poor when implemented on the process with large disturbances and model errors [10, 11].

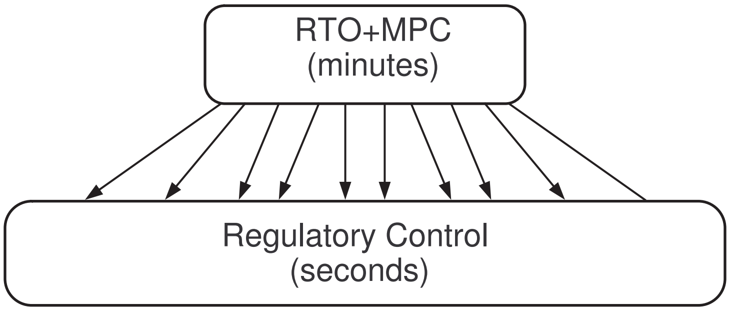


Figure 6: Schematics of the full-fledged dynamic RTO scheme

Many integrated plants are much more complex due to the presence of many highly nonlinear unit processes (such as distillation columns) and a recycle loop. As a result, the computational and modeling demand of this approach will become unmanageable when applied to such problems. This is because the computational time of dynamic optimization depends strongly on the size of the system (number of state and input variables), the prediction horizon, and the system's complexity. With the recycle, the settling time of the plant will be extremely long that the fast-rate optimizer in this scheme will need to carry a very large prediction horizon. This will result in a potentially exorbitant on-line computational requirement to calculate the optimal solution at a very fast sample time. As a result, it is unlikely that such approach can efficiently solve an optimization problem of an integrated plant despite all the advances in computational hardware and software nowadays.

2.2.2.2 Single-point Fast-rate Dynamic RTO Scheme with a Coordination Layer

Another plant-wide optimization framework that allows the optimization to be performed while the plant is in transient state is the single-point dynamic RTO scheme recently proposed by Lu [1]. This framework employed the hierarchical structure, where the RTO is separated from the MPCs, but both are executed at the same rate as shown in Figure 7. At each execution, the plant's output is predicted out to a single future point, referred to as the 'optimization point'. This way, the optimizer works like a gain-only predictive optimizer based on the gain matrices evaluated at the optimization point. As a result, this method will not require as much computational load as the previous approach that considers multiple

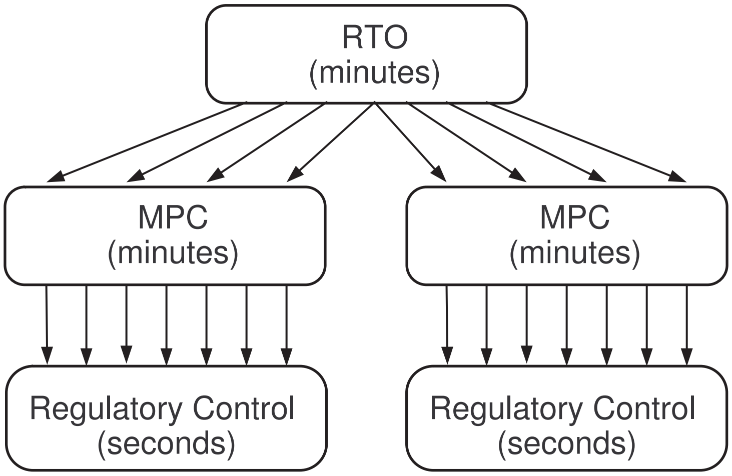


Figure 7: Schematics of the single-point dynamic RTO strategy

optimization points, while executing at a very fast rate. In addition, this scheme considers the use of a least-square coordination collar to ensure that the setpoint calculated by the RTO is feasible at the local MPCs. This coordination layer will be explained in Chapter 3.

Because the execution rate is in the range of the fast-scale dynamics, a potential disadvantage of this approach is that the prediction can be sensitive to high-frequency model errors or high frequency dynamics in the unit-level that may not be related to the plant-wide interactions, which are much slower changes in nature [22, 23]. In addition, the optimal solution that was determined with respect to a single optimization point may not guarantee the optimality of the entire dynamic trajectory.

2.3 Conclusions

In this chapter, we introduced the characteristics of an integrated plant with a recycle. The plant with a material recycle interested in this work shows a two-time scale behavior with fast dynamics and slow dynamics. Then, the formulations of the existing frameworks for plant-wide optimization schemes, including the steady-state RTO, and the single-scale fast-rate dynamic RTO have been presented. We pointed out potential drawbacks of each approach when applied to an integrated plant with a material recycle. This motivates the need for an alternative plant-wide optimization approach that can perform better than the existing frameworks. We will discuss the proposed framework and show its efficacy in applying to an integrated plant in Chapter 3.

CHAPTER 3

MULTI-SCALE REAL-TIME OPTIMIZATION AND CONTROL FRAMEWORK

The objective of this chapter is to propose a novel multi-scale dynamic optimization and control strategy suitable for an integrated plant with a recycle. We reason why the framework is appealing for this problem and present its architecture. We use two examples to illustrate the efficacy of the proposed approach.

3.1 Overall Multi-scale Optimization and Control Structure

An integrated plant is characterized by a two-time scale dynamics, with fast modes from individual units' dynamics and slow modes from the interactions introduced by the recycle. The steady-state RTO strategy introduced in Chapter 2 will, therefore, suffers from a limited execution rate, while the single-scale optimization strategies performed at a fast rate can be very sensitive to high-frequency changes that are not pertaining to the plant-wide interactions.

Based on the aforementioned problems, this research proposes a multi-scale dynamic optimization strategy, where the plant-wide optimizer is executed at the slow dynamic time scale of the plant. In this approach, the optimizer is performed at a rate significantly lower than the local unit-based MPC controllers in order to dynamically track changes at the plant-wide level, but it does not have to wait for the plant to reach the steady state. The lower execution rate would be more manageable from the modeling viewpoint because a slower rate dynamic model, which describes only dominant slow modes, should be easier to identify than the fast-rate dynamic model. The overall architecture of the proposed approach can be represented by Figure 8.

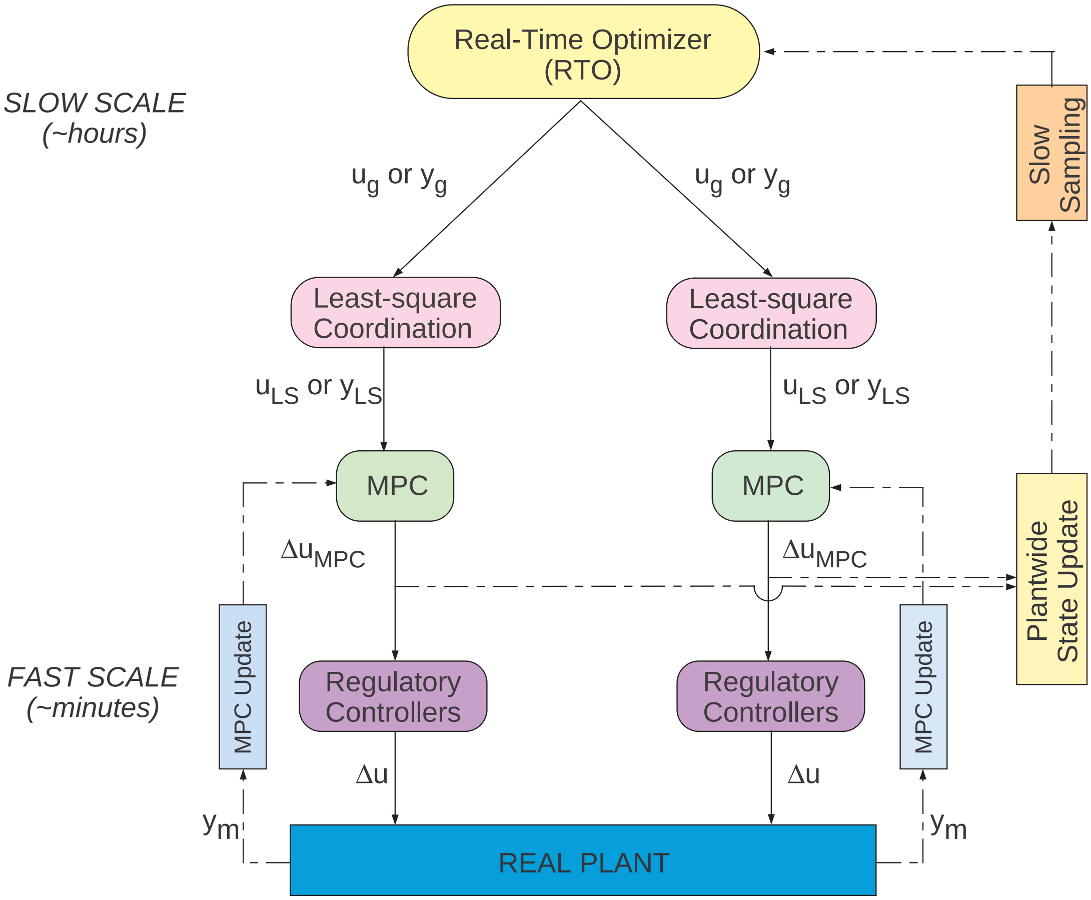


Figure 8: Architecture of the multi-scale plant-wide dynamic RTO.

In this scheme, the dynamic optimization problem (6) is solve at every T_{opt} time interval, which is slower than the sample time of the MPCs.

$$\begin{gathered}\min_{U_g(k)} f(U_g(k)) \\ Y_{g,min} \leq Y_g(k+1|k) \leq Y_{g,max} \\ U_{g,min} \leq U_g(k) \leq U_{g,max} \\ \Delta U_{g,min} \leq \Delta U_g(k) \leq \Delta U_{g,max}\end{gathered} \tag{6}$$

where

$$U_g(k) = \begin{bmatrix} u_g(k) \\ u_g(k+1) \\ \vdots \\ u_g(k+M-1) \end{bmatrix}, \qquad Y_g(k+1|k) = \begin{bmatrix} y_g(k+1|k) \\ y_g(k+2|k) \\ \vdots \\ y_g(k+P|k) \end{bmatrix}. \tag{7}$$

The current sample time is denoted by k. In this framework, the output vectors y_g are optimized along the prediction horizon of P, while constrained within lower and upper bounds of $Y_{g,min}$ and $Y_{g,max}$, respectively. The vector of manipulated variables is denoted by u_g. The input profile in this problem has the control horizon of M and is subject to the magnitude constraints $U_{g,min}$ and $U_{g,max}$, as well as the rate constraints $\Delta U_{g,min}$ and $\Delta U_{g,max}$. Once the optimal setpoint profile is found, the first move ($u_g(k)$ and/or $y_g(k+1|k)$) is sent to the lower-level controllers to implement. In many chemical plants nowadays, there exist MPCs in major unit operations to track the setpoints from the plant-wide optimizer as illustrated in Figure 8. At each MPC sample time, the MV profile is calculated and the first input move is implemented on the plant. Then plant output is measured and used to update the MPC state vector for the next execution. With some appropriate filtering, this information is also used to update the plant-wide state vector at each MPC sample time. This process is repeated until the next execution time of the RTO is reached, then the plant-wide state vector is sampled and used in the optimization.

Key questions related to the architecture design of this framework are how to choose the optimization frequency for the plant-wide optimizer, how to obtain the model for the optimizer, and how to coordinate the slower-rate plant-wide optimizer to the lower-level fast-scale control layer. We address these questions and propose a design guideline in the following section.

3.2 Design Issues of the Proposed Framework

3.2.1 Determination of the Optimization Frequency

Choosing an appropriate dynamic optimization frequency is an important step. It should be decided based on various factors, including

- Bandwidth of interaction dynamics. One may obtain an estimate of the fast interaction dynamics by examining the time constant of each individual unit operation. The plant-wide optimization frequency should not be performed at a rate faster than the time constant of the units; otherwise the optimizer may observe a lot of local fast-rate dynamics, which should be handled by local controllers instead of the plant-wide optimizer. For the case where the first-principles dynamic model is available, one may examine the eigenvalues of the system around its steady state. Those eigenvalues lying far away from zero correspond to fast modes of the process, and should be of interest at the unit-based control level. In contrast, those eigenvalues that are close to zero correspond to slow modes, which should be considered by the plant-wide optimizer. Therefore, one may choose the optimization frequency that marks a time-scale separation as indicated by a large magnitude of difference between two adjacent eigenvalues.

- Computational feasibility. Once the fast frequency region of the process is determined, we can choose the optimization frequency anywhere slower than that range. However, the optimization sample time has an implication on the computational load. Therefore, apart from the consideration of the inherent dynamic characteristics of the system, in practice one also has to consider the computational capability of their system. When the transient dynamics of the plant is very long, the prediction horizon of the optimizer should span as far into the future as possible to ensure a satisfactory performance. However, if this requires a significantly large on-line computational load, increasing the sampling interval by few hours may substantially reduce the on-line computational requirement without causing a significant impact in the performance of the optimizer.

3.2.2 Obtaining Model for Slow-scale Dynamic Optimization

Once the frequency of the plant-wide optimizer is decided, one must develop a plant-wide model accurate within the chosen frequency range. This model may be derived from first-principles fundamental equations or from system identification. In the former case, one usually gets a very large set of differential algebraic equations (DAEs), which tend to be very stiff. In the simplest scenarios, users may use linear/linearized models of all the process units and connect them through some 'bridge' dynamics to formulate the plant-wide model. The linear plant-wide model so obtained can be reduced through the procedure of a frequency-weighted model reduction technique (FWMR) [24, 25, 26], which attempts to derive a lower dimensional approximate of the original model that minimizes the truncation error within the chosen frequency range, i.e.

$$\|W_o(G - G_r)W_i\|_\infty \tag{8}$$

where G and G_r are the original and the reduced-order model, respectively. The output and input weighting matrices, W_o and W_i, are designated to make the approximation more accurate at the low frequency.

In the case of a highly nonlinear system that cannot be well approximated by a linearized model, nonlinear model reduction techniques such as the proper orthogonal decomposition method [27, 28, 29, 30] coupled with residualization technique should be used. This will result in a lower-order nonlinear system of equations, which allows the nonlinear model integration to be performed faster. We will illustrate the use of this method in Chapter 5.

When the first-principles dynamic model is unavailable, system identification needs to be performed on the plant. Subspace identification algorithms [31, 32] are commonly used methods to derive a multi-input multi-output subspace model that can be used for optimization directly. The most challenging steps in identifying an integrated plant model, however, lie in the design of experiment and making sure that the long-term slow-scale prediction of the model is reasonably accurate. These issues will be discussed in more details in Chapter 4.

3.2.3 Coordinating with the Lower-level Fast-rate Control Layer

In the situations where there are multiple MPC controllers in the plant, it is possible that the setpoints calculated from the RTO are infeasible for the local MPCs. This is because each MPC is built from a local process model and it makes an open-loop prediction of the plant's future behavior using only the available information of the state and disturbance estimate up to the current time. When the new setpoint from the RTO is issued, each MPC has no information about the future changes in other units, and therefore may find the RTO setpoint infeasible.

In light of this, Lu [1] suggested building a coordination collar on top of each MPC controller to find for each MPC a locally feasible setpoint that are closest to the global solution in a least-square sense:

$$\begin{aligned} &\min_{u_{ls}(k)} [u_{ls}(k) - u_g]^T [u_{ls}(k) - u_g] \qquad (9) \\ &y_{min} \leq y_{ls}(k) \leq y_{max} \\ &u_{min} \leq u_{ls}(k) \leq u_{max} \end{aligned}$$

where

$$y_{ls} = G_x x(k) + G_u u_{ls}(k) + G_d d(k) \qquad (10)$$

The subscript ls denote the setpoint computed from the least-square coordination collar. Output constraints, y_{min} and y_{max}, as well as input constraints, u_{min} and u_{max}, are the constraint sets of the local MPC at the end of its prediction horizon. G_x, G_u, and G_d in Eq. 10 are the dynamic gain matrices at the optimization point, which relate the state, the input, and the disturbance to the output, respectively. Note that it is also possible to formulate Eq. 9 to find the feasible CV values closest to the RTO solution. The choice of minimized variables should be based on the given plant-wide objective.

In our proposed multi-scale dynamic RTO scheme, the coordination collar is built for each MPC to ensure the feasibility of the setpoints from the plant-wide optimizer. the execution rate of this coordination layer should be chosen according to the dynamic interactions among the unit operations. That is, when the process interactions are rather fast, the coordination layer may need to be executed at higher rate than the RTO. Nevertheless, in most cases the process interactions tends to be slow and the coordination collar can be executed at the same rate as the RTO.

3.3 Application to an Integrated Plant Example

In this section we provide two examples to compare performance of the proposed multi-scale dynamic plant-wide optimization and control approach to the previously existing methods. The first example considers a two interacting units example studied in [1]. Then, the second example involves a reactor, a storage tank, and a flash tank connected via a large recycle stream studied in [33, 34]. This example is very stiff, and therefore adds further complexity to the control and optimization calculations.

3.3.1 Example 1

3.3.1.1 Process Description

The transfer function model of this problem is given in Figure 9. Each unit comprises two controlled variables and two manipulated variables. The controlled variables are denoted as CVij, where i represents unit operation and j indicates whether it is the first or the second variable in that unit. Similar indices are also used for the MVs. At the initial state, all variables are scaled to 0. The two MVs in the first unit are the product draws, which are processed and fed to the second unit. The first MV in the second unit is a recycle stream that goes back to the first unit. The first CVs of both units are quality variables that must be controlled to the setpoints. The seconds CVs of the units are economic variables to be optimized. The simulation begins at time 0 to increase the setpoint of CV11 to 1 and maximize the summation of CV12 and CV22, while maintaining CV21 at 0

3.3.1.2 Unit-based Control Layer

At the unit level, an MPC controller was built for each unit. Their parameters are given in Table 1, where p is prediction horizon, m is control horizon, Q and R are weighting matrices

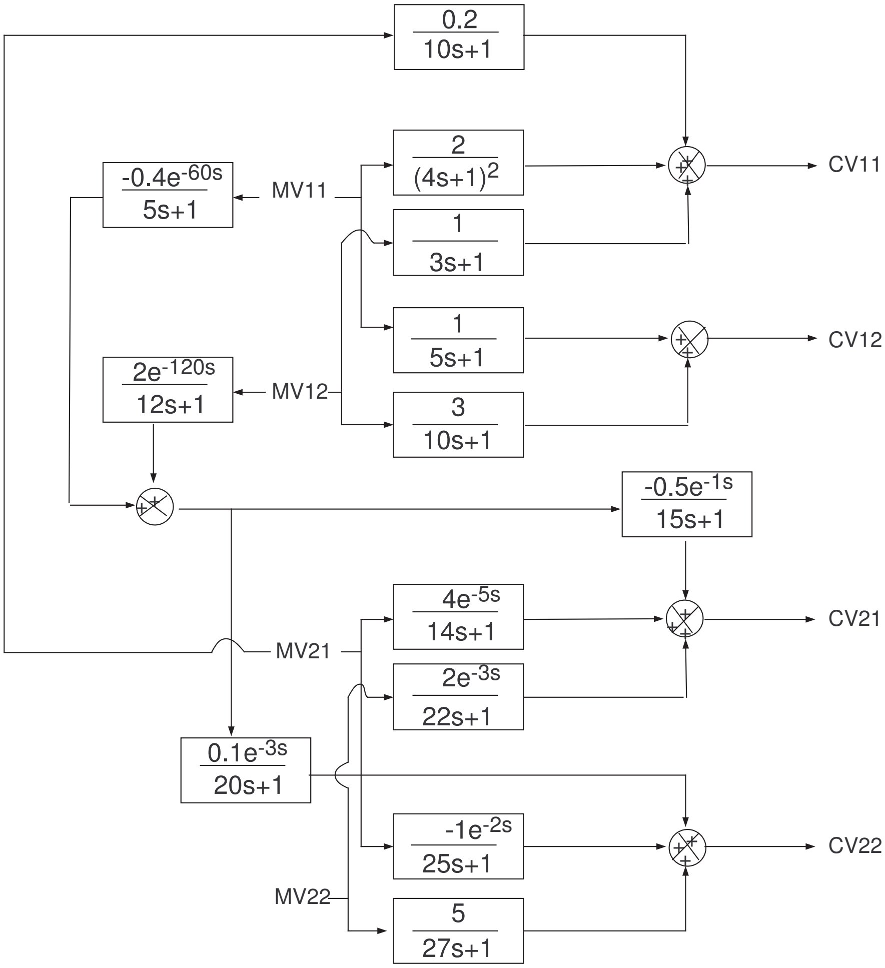


Figure 9: Schematic of the two-interacting unit processes

Table 1: Parameters used in local MPCs

Unit	p	m	$\Delta u_{ij,max}$	$[u_{ij,min}, u_{ij,max}]$	$[y_{ij,min}, y_{ij,max}]$	Q	R
1	20	10	0.3	[-10, 10]	[-10, 10]	$\begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 30 & 0 \\ 0 & 30 \end{bmatrix}$
2	100	20	0.3	[-10, 10]	[-10, 10]	$\begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 30 & 0 \\ 0 & 30 \end{bmatrix}$

for CV and MV, respectively. Each controller is built based on the local model, so it has no knowledge of the interaction dynamics from the bridge model. The local MPCs compute control actions every one minute.

3.3.1.3 Plant-wide Optimization by the Steady-state RTO

The economic optimization problem for the steady-state RTO was formulated as a linear program (LP) of the following:

$$\max_{\Delta MV_{ij}} [CV12 + CV22] \tag{11}$$

subject to

$$-10 \leq CVij \leq 10$$
$$-10 \leq MVij \leq 10$$

where CVij were computed from the MV move times the steady-state gains of the process. To compare the economic performance of the different RTO schemes, the performance measure defined by Eq. 12 was used. The results are summarized in Table 2

$$E = \sum_{t=1}^{200} CV12(t) + CV22(t) \tag{12}$$

At time 0, the steady-state RTO calculated the input setpoints for MV11, MV12, MV21, and MV22 to be -1.4037, 3.8012, 0.0306, and 1.9797, respectively. However, at that point in time, each MPC did not know the future disturbance from the other unit. The second MPC only had the information that the current disturbance was zero. From the local

Table 2: Economic performances of different RTO strategies applied to an interacting plant example [1]

Strategy	E
One execution of the steady-state RTO without a coordination layer	Infeasible
One execution of the steady-state RTO with a coordination layer	3281
Steady-state RTO executed every 30 min interval with a coordination layer	3199
Single-point dynamic RTO	3309
Proposed multi-scale dynamic RTO	3570

model and the disturbance information at that time, the setpoint for CV21 was infeasible to the second MPC. As a result, we modified the steady-state RTO scheme by applying the coordination collar to each MPC and executing them at the same rate. The steady-state RTO scheme executed only once at the initial steady state yielded the performance of 3281. The trajectories of the CVs and MVs are shown in Figures 10 and 11.

Next, as the execution rate was increased to once every 30 minute without waiting for the system to reach steady-state, the worse performance of 3199 resulted. This was because the optimizer was re-executed when the plant was still in the transient state. Therefore, the assumption that the initial state was at the steady-state condition was severely violated, causing incorrect prediction. Therefore, it did not help running the RTO faster when the prediction was poor.

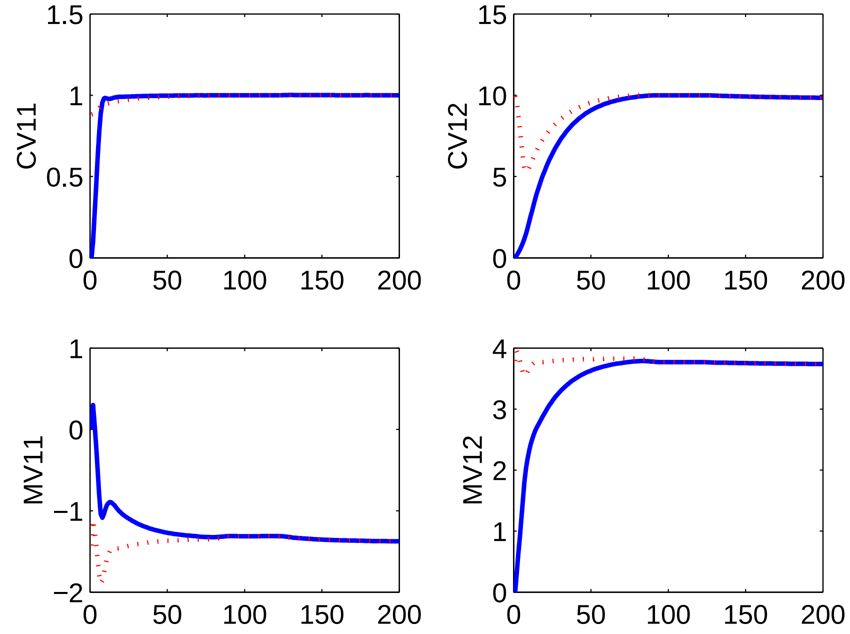


Figure 10: CVs (top) and MVs (bottom) of unit 1 using the steady-state RTO scheme (solid lines: measured outputs, dotted lines: setpoint trajectories).

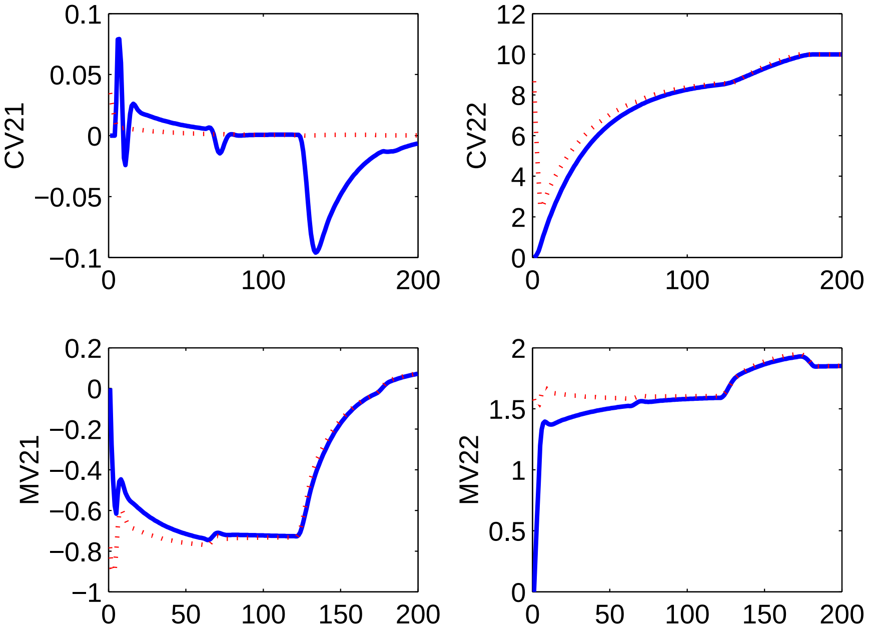


Figure 11: CVs (top) and MVs (bottom) of unit 2 using the steady-state RTO scheme. (solid lines: measured outputs, dotted lines: setpoint trajectories).

3.3.1.4 Plant-wide Optimization by the Single-point Dynamic RTO

For the single-point dynamic RTO, the optimization point, from which the dynamic gains were obtained, was at the end of the MPCs' prediction horizons. The calculated MV setpoints from the RTO were sent to the coordination layer to check for feasibility before sending them to the MPCs. In this case, the optimizer and the coordination collar run at same rate as the MPCs.

The simulation plots are shown in Figures 12 and 13 where solid lines represent MPC prediction and dotted lines represent setpoint trajectory sent to the MPCs. The result from Table 2 shows a slight improvement in performance from the steady-state RTO with a coordination collar case. The time required for CV22 to reach the maximum value still takes as long as 200 minutes. There is a big room for improvement

3.3.1.5 Plant-wide Optimization by the Proposed Multi-scale Dynamic RTO

In this scheme, one first needs to determine the optimization frequency. To prevent the plant-wide optimizer from capturing a lot of the fast dynamics from the local units, the optimization frequency has to be slower than the largest time constant of the process units, which is 27 minutes. Since this system model is rather simple and would not require a high computational load, we chose the time period of 30 minutes as the optimization interval. The low-order model for optimization was derived from the frequency-weighted model reduction technique, where the diagonal elements of the weighting matrices W_o and W_i in Eq. 8 are the low-pass second order Bessel filter of the following form:

$$\frac{\omega^2}{(s/\omega_B)^2 + 2\zeta\omega\,(s/\omega_B) + \omega^2} \tag{13}$$

where $\omega = 1.27, \zeta = 0.87$, and $\omega_B = 0.21$. After this procedure, the plant model was reduced down to 5 states. We used the prediction and the control horizons of 4 and 1, respectively. The economic optimization problem was formulated as a linear program (LP) of the followings:

$$\max_{\Delta MV_{ij}(k)} \sum_{\ell=1}^{4} [CV12(k+\ell) + CV22(k+\ell)] \tag{14}$$

subject to

$$\begin{aligned} &-10 \leq CVij(k+\ell|k) \leq 10, \qquad \ell = 1, \cdots, 4 \\ &-10 \leq MVij(k|k) \leq 10 \end{aligned}$$

The calculated MV setpoints from the optimizer were sent to the coordination layer for feasibility check. Then, locally feasible output setpoint trajectories were generated from Eq. 10 and sent to the MPC. The performance of the suggested approach compared to the previous methods is shown in Table 2. The trajectories of the MVs and CVs from this strategy are depicted in Figures 14 and 15, which showed better transition of CV12 and CV22 to the target. This is because the optimizer calculated the MV setpoints based upon multiple output prediction points along the dynamic trajectory. Therefore, it provided much better transition trajectories than the previous approaches that considered only a single optimization point.

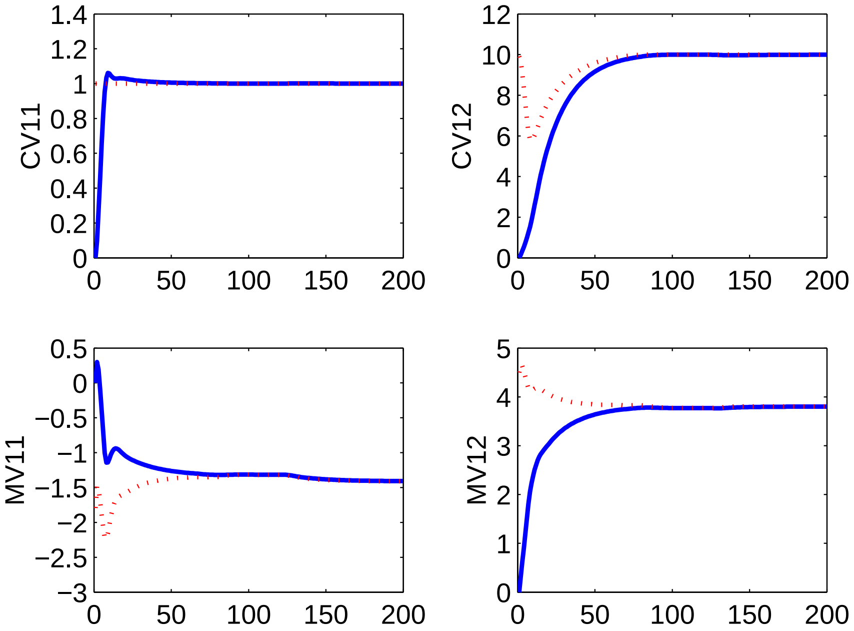


Figure 12: CVs (top) and MVs (bottom) of unit 1 using the single-point dynamic RTO scheme. (solid lines: measured outputs, dotted lines: setpoint trajectories).

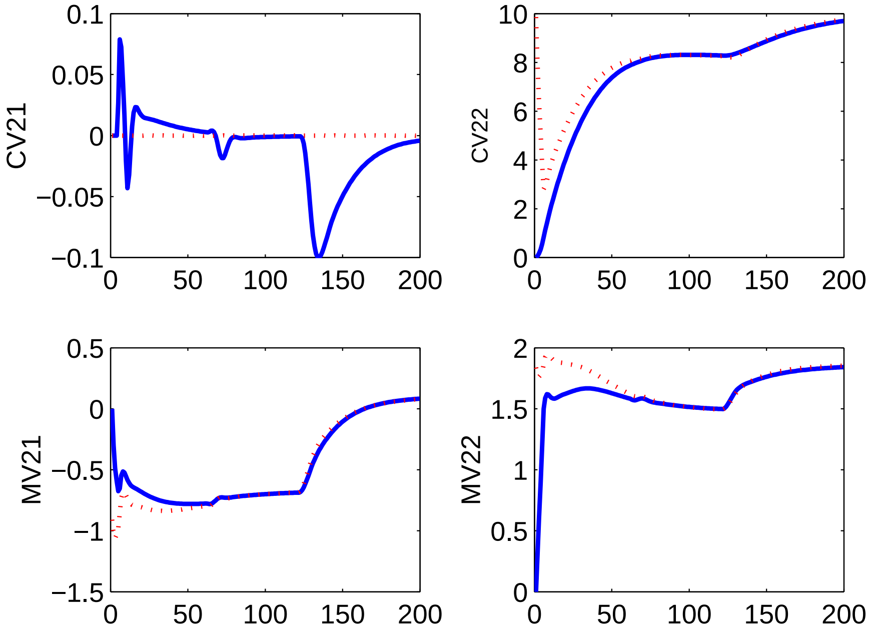


Figure 13: CVs (top) and MVs (bottom) of unit 2 using the single-point dynamic RTO scheme. (solid lines: measured outputs, dotted lines: setpoint trajectories).

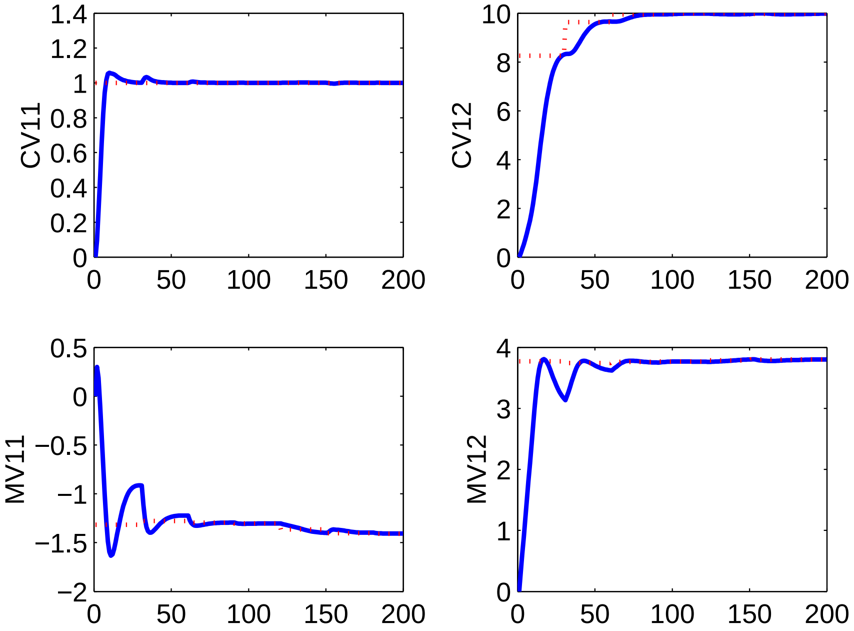


Figure 14: Output (top) and input (bottom) variables of unit 1 in example 1 using the multi-scale dynamic RTO. (solid lines: measured outputs, dotted lines: setpoint trajectories).

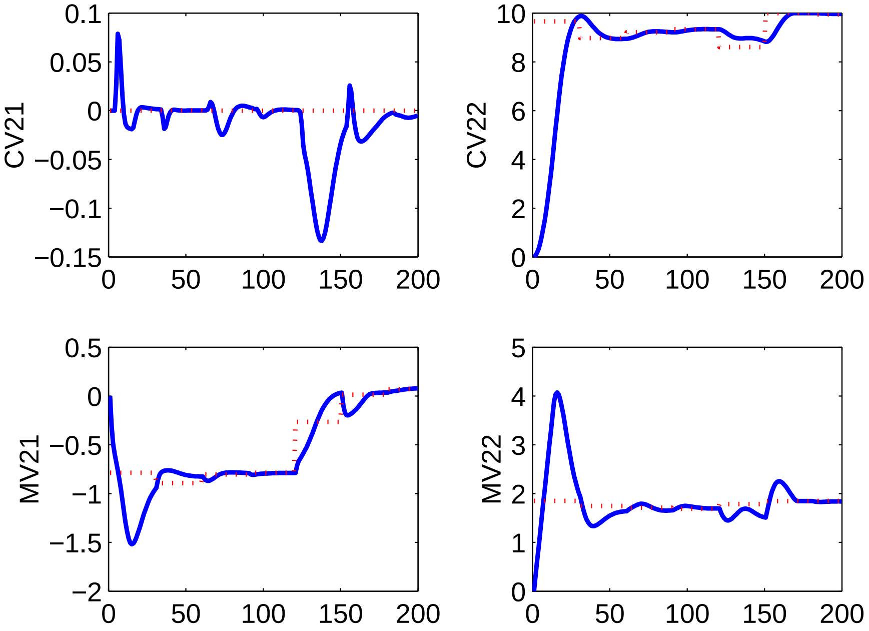


Figure 15: Output (top) and input (bottom) variables of unit 2 in example 1 using the multi-scale dynamic RTO. (solid lines: measured outputs, dotted lines: setpoint trajectories).

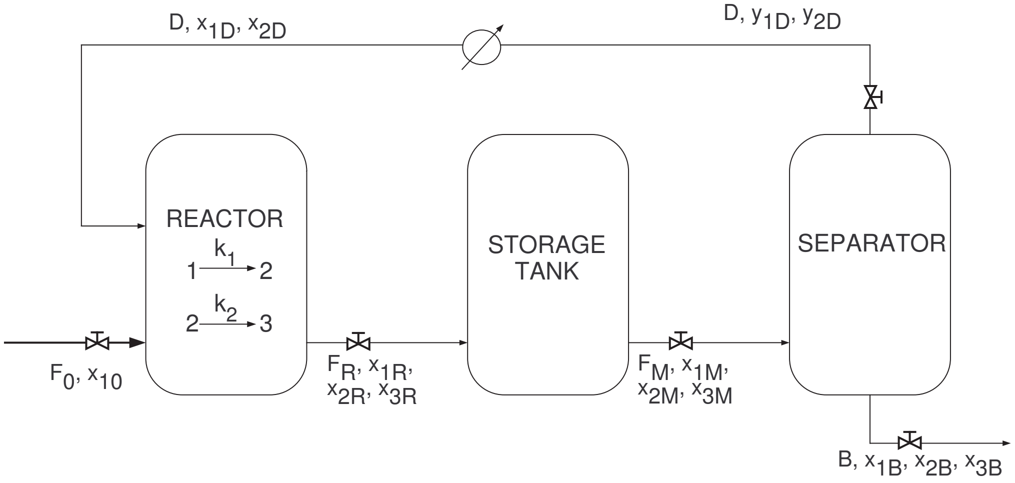


Figure 16: Schematic of the reaction-storage-separation network in example 2.

3.3.2 Example 2

Next, we consider a process with a CSTR, a storage tank, and a flash tank with a material recycle stream shown in Figure 16. A fresh feed stream F_0 consisting of pure component 1 is fed to the reactor, where two irreversible reactions $1 \overset{k_1}{\rightarrow} 2 \overset{k_2}{\rightarrow} 3$ take place to produce a desired product 2 and an undesired product 3. The reactor effluent stream F_R consisting of components 1, 2, and 3 enters the storage tank, from which the downstream flow F_M leads to the flash tank. We assumed that the volatility of component 1 is much higher than that of component 2, and component 3 is nonvolatile. Hence, most of reactant 1 goes up the overhead, where it is completely condensed into liquid and sent back to the reactor. Since it is important to keep the selectivity of component 2 high, a single-pass conversion has to be low. High yields can still be achieved by maintaining a high ratio of the recycle flow to the fresh feed flow (D/F_0). This is a challenging control problem as the system can exhibit a severe *snowball effect* [35] if not properly controlled. That is, only a small change in the feed stream can cause a large variation in the process, especially when the recycle-to-feed ratio is very high.

For simplicity we assume a constant liquid density in every vessel and an isothermal operation for the entire process. Under this circumstance, the material balance consists of

12 equations as follows:

$$
\begin{aligned}
\dot{H_R} &= \frac{1}{\rho A_R}(F_0 + D - F_R) \\
\dot{x_{1R}} &= \frac{F_0(x_{10} - x_{1R}) + D(x_{1D} - x_{1R})}{\rho A_R H_R} - k_1 x_{1R} \\
\dot{x_{2R}} &= \frac{-F_0 x_{2R} + D(x_{2D} - x_{2R})}{\rho A_R H_R} + k_1 x_{1R} - k_2 x_{2R} \\
\dot{x_{3R}} &= \frac{-(F_0 + D)x_{3R}}{\rho A_R H_R} + k_2 x_{2R} \\
\dot{H_M} &= \frac{1}{\rho A_M}(F_R - F_M) \\
\dot{x_{1M}} &= \frac{F_R}{\rho A_M H_M}(x_{1R} - x_{1M}) \\
\dot{x_{2M}} &= \frac{F_R}{\rho A_M H_M}(x_{2R} - x_{2M}) \\
\dot{x_{3M}} &= \frac{F_R}{\rho A_M H_M}(x_{3R} - x_{3M}) \\
\dot{H_B} &= \frac{1}{\rho A_B}(F_M - B - D) \\
\dot{x_{1B}} &= \frac{1}{\rho A_B H_B}[F_M(x_{1M} - x_{1B}) - D(x_{1D} - x_{1B})] \\
\dot{x_{2B}} &= \frac{1}{\rho A_B H_B}[F_M(x_{2M} - x_{2B}) - D(x_{2D} - x_{2B})] \\
\dot{x_{3B}} &= \frac{1}{\rho A_B H_B}[F_M(x_{3M} - x_{3B}) + D x_{3B}] \qquad (15)
\end{aligned}
$$

where H_R, H_M, and H_B denote the liquid levels in the reactor, the storage tank, and the flash tank, respectively. Here, x_{ij} denotes the molar liquid fraction of component i ($i = 1, 2, 3$) in the stream j ($j = 0, R, M, B, D$ for feed, reactor effluent, storage tank effluent, product draw, and recycle stream, respectively). The nominal values of the process and operating parameters are given in Table 3.

The test scenario is to increase the production throughput by 20%, while keeping the product compositions and operating conditions within the constraints given in Table 4. In this problem, there are 5 MVs, including F_0, F_R, F_M, B, and D. However, as the liquid level in each tank behaves as an integrator, some of these streams must be used to stabilize the levels. According to Richardson's rule [35], the largest stream should be selected to control the liquid level in the vessel. However, if we select F_R, F_M, and D to control the levels of the reactor, the storage tank, and the flash tank, respectively, the three levels are

Table 3: Nominal values for the process and operating parameters for example 2

Parameters	Value		
Liquid density	$\rho = 1$		
Volatility	$\alpha_1 = 90$	$\alpha_B = 1$	
Rate constant	$k_1 = 0.0167$	$k_2 = 0.0167$	
Vessel area	$A_R = 5$	$A_M = 10$	$A_B = 5$
Vessel holdup	$H_R = 20$	$H_M = 20$	$H_B = 20$
Flowrate, hr^{-1}	$F_0 = 1.667$	$F_R = 31.33$	$F_M = 31.33$
	$B = 1.667$	$D = 29.67$	
Mole fraction	$x_{10} = 1.00$	$x_{20} = 0$	$x_{30} = 0$
	$x_{1R} = 0.8861$	$x_{2R} = 0.1082$	$x_{3R} = 0.0058$
	$x_{1M} = 0.8861$	$x_{2M} = 0.1082$	$x_{3M} = 0.0058$
	$x_{1B} = 0.1139$	$x_{2B} = 0.7779$	$x_{3B} = 0.1082$
	$x_{1D} = 0.9295$	$x_{2D} = 0.0705$	
P-Controller gains	$K_{C,R} = -10$	$K_{C,M} = -10$	$K_{C,B} = -5$

not independently controllable as the MVs are all internal flow variables. Instead, we used F_R, F_M, and B to stabilize the levels through P-only controllers. Although these flows are no longer available as manipulated variables for the plant-wide optimizer, the degree of freedom remains the same as the level setpoint of each vessel can be used as a MV. The following subsections provide more details on the MPC and the plant-wide optimization formulations, which were obtained from the linearized model of the nonlinear plant.

3.3.2.1 Unit-based Control Layer

The plant was divided into two process units: Unit 1 consists of the reactor and the intermediate tank, whereas Unit 2 includes the flash tank. An MPC controller was built for each unit in order to steer the CVs to the setpoints, which are specified by the plant-wide optimizer, while respecting the constraints. A list of output and manipulated variables of each unit as well as their constraints are given in Table 4.

The sample times of both MPCs are 6 minutes. The linearized plant model model was used to formulate the prediction model of the MPCs. As the transient dynamics last as

Table 4: Output and manipulated variables of unit 1 and unit 2

Unit 1			Unit 2		
output	variables	operating range	output	variables	operating range
1	x_{1R}	[0,1]	1	x_{1B}	[0,0.15]
2	x_{2R}	[0,0.15]	2	x_{2B}	[0.75,1]
3	x_{3R}	[0,0.02]	3	x_{3B}	[0,0.15]
4	x_{1M}	[0,1]	4	B	[0.67,3]
5	x_{2M}	[0,0.15]			
6	x_{3M}	[0,0.02]			
7	F_R	[8,47]			
8	F_M	[8,47]			
MV	variables	operating range	MV	variables	operating range
1	H_R	[10, 30]	1	H_B	[10, 30]
2	H_M	[10, 30]	2	D	[8, 45]

long as 12 hours, we used the model predictive control formulation for integrating dynamics [36, 37], which allows the step-response models to be truncated well before the responses settle with little sacrifice in accuracy. The parameters for both MPCs are given in Table 5, where t_{trnc} is the truncation time of the step-response model, p is the prediction horizon, m is the control horizon, Q and R are output and input weighting matrices, respectively. Note that in the MPC optimization problem, the output constraints were implemented as soft constraints to avoid computational infeasibility. Nevertheless, a large penalty term (equal to $10^6 \sum \epsilon_i^2$, where ϵ_i is a slack variable representing the magnitude of violation of the i^{th} output) was added to the objective function to avoid output constraint violations.

3.3.2.2 Plant-wide Optimization by a Steady-state Optimizer

First, the steady-state optimization strategy was applied to this system. The optimizer sample time was chosen to be 10 hours, as the plant dynamics would be very close to the

Table 5: Parameters for local MPCs

Parameters	MPC1	MPC2
Δt	6 min	6 min
t_{trnc}	8 hr	8 hr
p	40	40
m	10	10
Δu_{max}	[0.3; 0.3]	[0.04; 0.5]
Q	$\begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&3&0&0&0&0 \\ 0&0&0&0&7&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \end{bmatrix}$	$\begin{bmatrix} 0&0&0&0 \\ 0&1&0&0 \\ 0&0&0&0 \\ 0&0&0&3 \end{bmatrix}$
R	$\begin{bmatrix} 1&0 \\ 0&1 \end{bmatrix}$	$\begin{bmatrix} 3&0 \\ 0&1 \end{bmatrix}$

steady-state conditions after 10 hours. The plant-wide optimization problem is as follows:

$$\min_{\Delta u_g, y_g} (y_{sp} - y_g)^T Q_g^T Q_g (y_{sp} - y_g) + \Delta u_g^T R_g^T R_g \Delta u_g + 10^6 \epsilon^T \epsilon \tag{16}$$

subject to

$$y_{min} \leq y_g \leq y_{max}$$

$$u_{min} \leq u_g \leq u_{min}$$

where y_{sp} is the output target vector, i.e. to increase the production rate B by 20%. Q_g is a diagonal output weighting matrix, whose only nonzero element is for the production rate and is equal to 10. R_g is a diagonal input weighting matrix equal to $I_{5\times5}$. The output vector y_g is calculated from the input u_g times the gain matrix at the steady-state. The simulation scenarios include cases when: (A) there is no disturbance, and (B) there is a feed bias disturbance of +0.2 at time 700 minutes. Performances of the different plant-wide optimization schemes were measured by an integral squared error between the production target and the actual production rate from a nonlinear plant over 30-hour period.

The changes in major MVs and CVs when there was no disturbance are shown in Figures 17 and 18, respectively. The dash-dot lines represent the global setpoints from the plant-wide optimizer. At time 0, the optimizer decided to increase the setpoints of the feed by 20% to the value of 2 and increase the recycle to 45, which translated to a decrease in the concentration x_{2M} to 0.078 and an increase the concentration of the product x_{2B} in the long run. However, this optimal solution was only determined at the steady-state point. The optimizer had no knowledge of the transient behavior of the plant, and therefore did not realize that these changes were very aggressive for the plant. As the MPC 1 tracked the setpoints of x_{2M}, the product concentration x_{2B} quickly violated its lower bound constraints within 1 hour. Furthermore, because the optimization frequency of the steady-state optimizer was very low, the setpoints were not corrected until the system reached the next steady state. As a result, the constraint violation lasted for around 5 hours. For the other case where a bias disturbance was introduced to the feed at the time 700 minutes, which was in between the sample time of the optimizer, the steady-state optimizer performance severely deteriorated as shown in Table 6. This was because it had

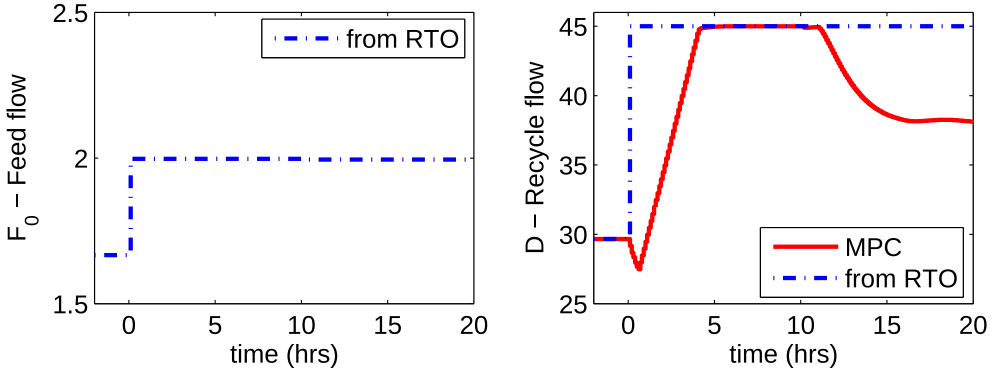


Figure 17: Selected manipulated variables from the steady-state optimization scheme when there was no disturbance.

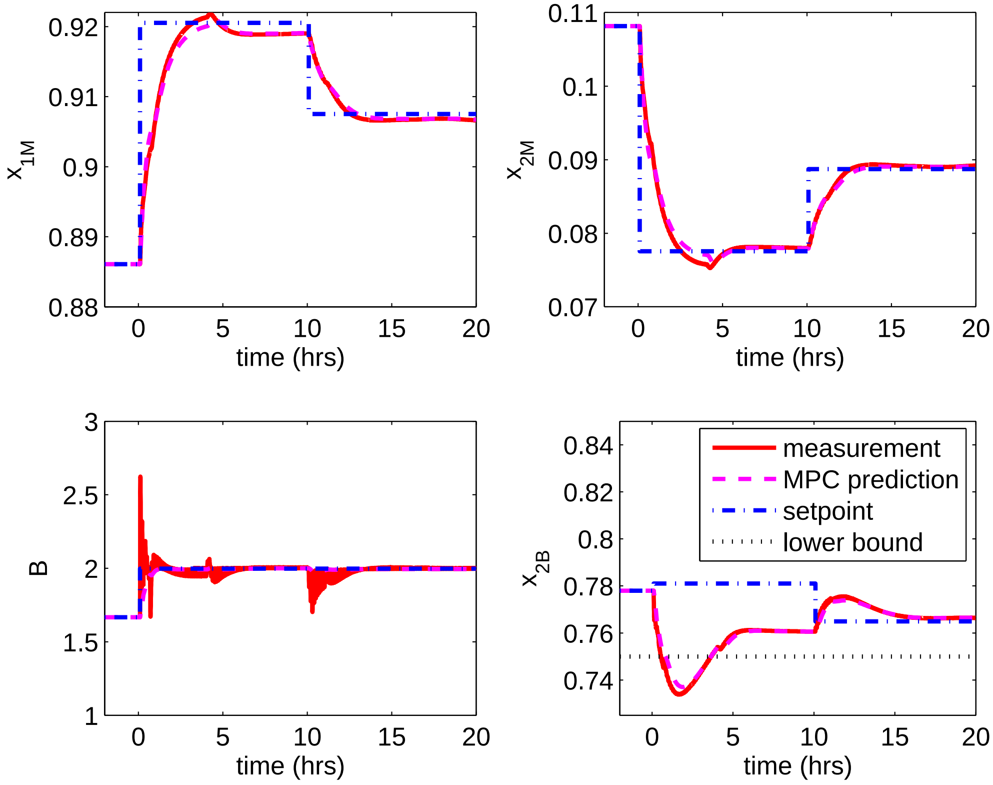


Figure 18: Controlled variables of unit 1 (*top*) and unit 2 (*bottom*) from the steady-state optimization scheme when there was no disturbance.

Table 6: Economic loss defined by the integral square of error of the production rate over 30-hour period

Simulation scenarios	Steady-state optimizer	Single-point dynamic optimizer	Multi-scale dynamic optimizer
(a) No disturbance	5.02	9.95	3.98
(b) Feed disturbance of 0.2 at t = 700 min.	12.17	10.54	4.94

to wait for the plant to reach the steady state before the optimizer could recalculate the setpoints.

3.3.2.3 Plant-wide Optimization by the Single-point Dynamic Optimizer

The optimization point of this scheme was chosen to be at the end of the MPC prediction horizon, which was 240 minutes into the future. The least-square coordination collars were applied to check the global setpoint feasibility before passing them to the MPCs. In this scheme, the optimizer, the coordination layer, and the MPCs were all running at every 6 minutes. The grade transition performance for the case where there was no disturbance is shown in Figures 19 and 20.

At time 0, the optimizer calculated the MV setpoints for the feed flow and the recycle setpoints to be 1.96 and 43.5, respectively. This corresponded to the setpoints of 0.063 for x_{2M} and 0.81 for x_{2B}. However, during the first six minutes, the output concentration x_{2B} decreased as x_{2M} decreased. After 6 minutes the optimizer took the measurement and responded to the initial dynamic changes by reducing the recycle setpoint sharply to 37.0 and increasing the feed to 2.16. In the subsequent sample times, the feed and the recycle were adjusted multiple times in response to the fast-rate measurements obtained from the plant. As a result, the optimizer could prevent the concentration x_{2B} from violating its lower bound constraint. However, because the MV setpoints changed very often, the production stream, which was paired with the level controller for the flash tank, was moved very widely as shown in Figure 20 as a dark area. For the case where a bias disturbance entered the feed at the time 700 minutes, the performance of the single-point dynamic optimizer was superior

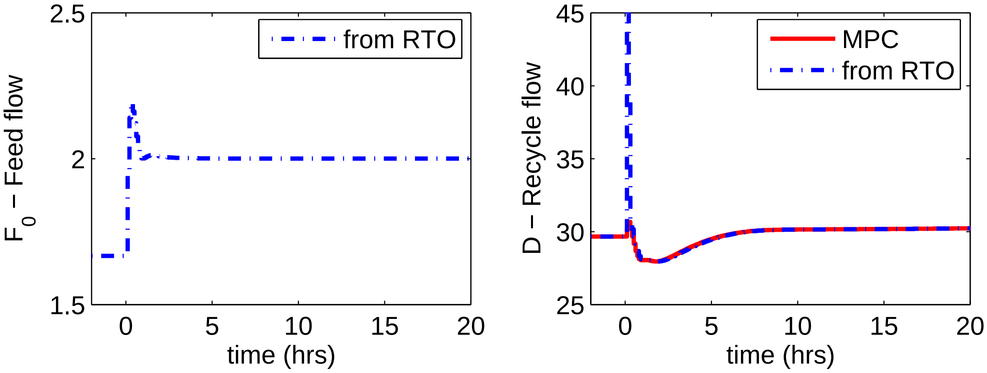


Figure 19: Selected manipulated variables from the single-point dynamic optimization scheme when there was no disturbance.

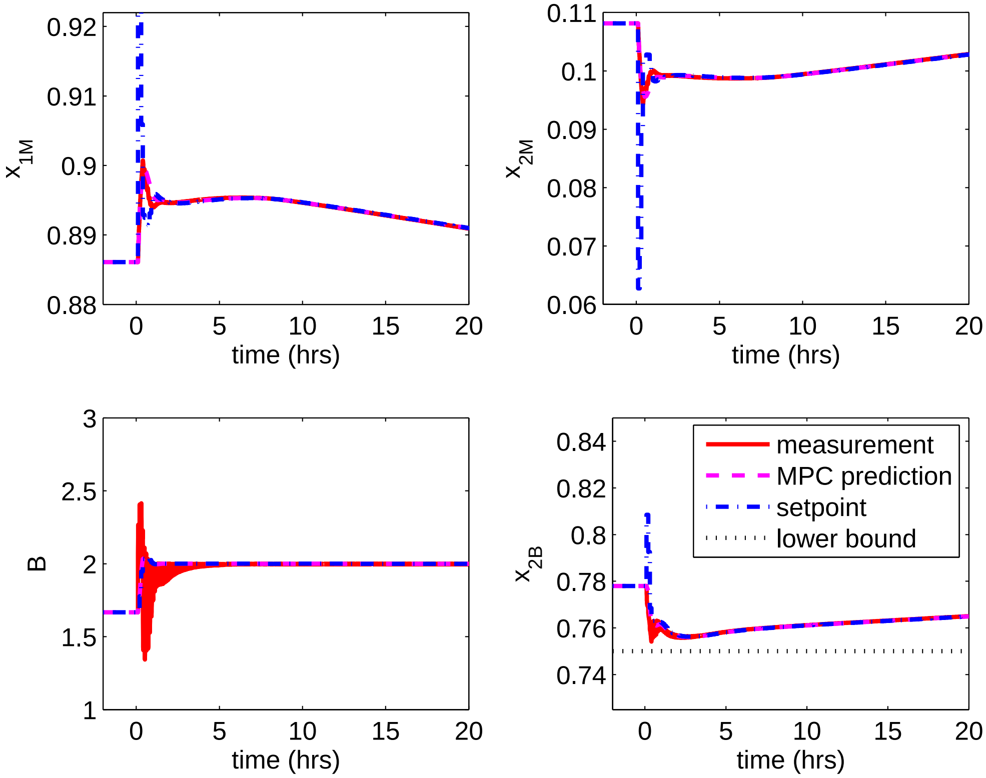


Figure 20: Controlled variables of unit 1 (*top*) and unit 2 (*bottom*) from the single-point dynamic optimization scheme when there was no disturbance.

to the steady-state optimizer. This was because it utilized the feedback measurement to recompute the setpoint without waiting for the steady state.

3.3.2.4 Plant-wide Optimization by the Multi-scale Dynamic Optimizer

To determine the optimization frequency for the slow-scale dynamic optimizer, the eigenvalues of the system around its steady state were examined in Table 7.

Table 7: Eigenvalues of the integrated plant with a reactor, a storage, and a flash separator

Magnitude of the eigenvalue	Corresponding time constant (min)
0.0097	103.29
0.0167	60
0.0167	60
0.1567	6.38
0.1567	6.38
0.3133	3.19
0.3279	3.05
0.3458	2.59
0.3458	2.59
1	1
1	1
2	0.5

There is a large time scale separation between the frequencies of 0.0167 and 0.1567 min^{-1}, which corresponds to the time constants of 6.38-60 minutes. As a result, we selected the optimization period of 60 minutes and obtained a corresponding slow-scale model by the frequency-weighted model reduction technique. The optimization is a quadratic program (QP) shown below.

$$\min_{\mathcal{U}_g} \left(\mathcal{Y}_{sp} - \mathcal{Y}_g(k+1|k)\right)^T \mathcal{Q}^T \mathcal{Q} \left(\mathcal{Y}_{sp} - \mathcal{Y}_g(k+1|k)\right) + \mathcal{U}_g^T \mathcal{R}^T \mathcal{R} \mathcal{U}_g + 10^6 \epsilon^T \epsilon \tag{17}$$

subject to

$$\mathcal{Y}_{\min} \leq \mathcal{Y}_g(k+1|k) \leq \mathcal{Y}_{\max}$$
$$\mathcal{U}_{\min} \leq \mathcal{U}_g \leq \mathcal{U}_{\max}$$

$$\text{where} \qquad \mathcal{Y}_g(k+1|k) = [y_g(k+1|k)^T, \ldots, y_g(k+P|k)^T]^T$$
$$\mathcal{U}_g(k) = [u_g(k)^T, \ldots, u_g(k+M-1)^T]^T$$

The prediction (P) and the control horizons (M) were chosen as 8 and 2, respectively. The weighting matrices were constructed as follows:

$$\mathcal{Q} = diag([\underbrace{Q_g, \quad Q_g, \quad \ldots, \quad Q_g}_{P}])$$
$$\mathcal{R} = diag([\underbrace{R_g, \quad R_g, \quad \ldots, \quad R_g}_{M}])$$

where $diag(\cdot)$ is an operator that diagonalizes all the matrices inside. The first MV setpoints, u_g, were sent to the coordination layer to compute the closest feasible MV setpoints. Then, the output setpoints (y_{ls}) for each MPC were computed from Eq. (10) and sent to the MPCs.

The linearized plant model was used to formulate the prediction model for the plant-wide optimizer. The implementation result on the nonlinear plant when there was no disturbance is shown in Figures 21 and 22. At time 0, the setpoints for the feed and the recycle were 2 and 37, respectively, which were not as aggressive as in the previous RTO cases. As a result, the setpoint for x_{2M} was much less aggressive and did not cause the concentration of x_{2B} to violate the lower bound constraint. In addition, the fluctuation in the product flow rate was much less than in the case of the single-point dynamic optimizer because the MV setpoints were changed smoothly. This was because the optimizer waited 60 minutes for the short-term dynamics to be handled by the local controllers before it recalculated the setpoints for the plant. In addition, with higher execution rate than the steady-state optimizer, this scheme could regulate the output concentration above its lower bound constraint at all time. The performance of this method is summarized in Table 6. In the case where a feed bias disturbance entered the system, this scheme still showed satisfactory performance compared to other methods.

3.4 Conclusions

The novel plant-wide dynamic optimization based on a slow-scale model is a computational middle ground between the existing steady-state optimization and the single-scale fast-rate dynamic optimization schemes proposed earlier. This scheme provides a hierarchical decomposition in the automation layer, where the plant-wide optimizer tracks slow changes relevant to the plant-wide interactions and economics, and the local control layer functions at the fast time scale to control the individual units. In choosing the optimization frequency, one should consider the bandwidth of the system dynamics as well as the computational capability of their hardware/software system. The examples in this chapter showed that suggested method is a promising alternative compared to the current steady-state or the single-scale dynamic optimization schemes. The former is limited in terms of the execution frequency, whereas the latter may be very sensitive to fast dynamics that are irrelevant to the plant-wide objective.

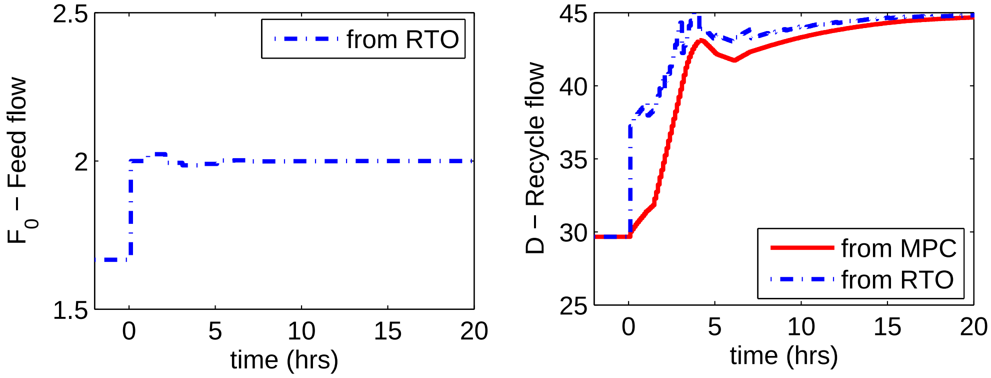


Figure 21: Selected manipulated variables from the multi-scale dynamic optimization scheme when there was no disturbance.

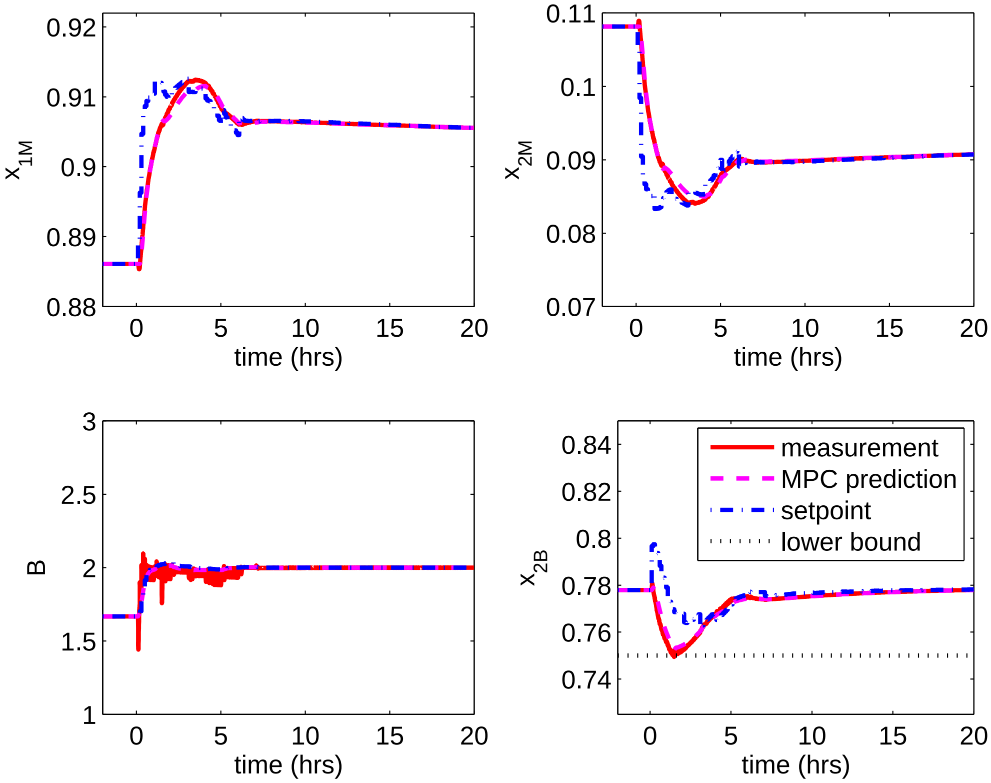


Figure 22: Controlled variables of unit 1 (*top*) and unit 2 (*bottom*) from the multi-scale dynamic optimization scheme when there was no disturbance.

CHAPTER 4

GREY-BOX MODEL IDENTIFICATION OF AN INTEGRATED PLANT

The objective of this chapter is to provide a set of guideline for systematically identifying a dynamic model of an integrated plant that is suitable for the plant-wide optimization. We are interested in cases where the identification experiment is limited to a much shorter period of time than the plant's settling time. The method is intended to take advantage of prior knowledge about the plant's steady-state gains. We use an example of an integrated plant composed of a reactor and a distillation column to identify the potential issues and to illustrate the effectiveness of the proposed grey-box identification approach.

4.1 Introduction

In a plant-wide dynamic optimization application, a model is required to predict future behavior of the system. Typically, models built from fundamental principles are globally valid, and hence well-suited to the predictive optimization and control task, which can involve a large range of operating variables. However, in practice it is often very difficult to obtain an accurate first-principles model for an entire plant. Even when it is possible, rigorous first-principles models for complex plants can be of very high order, which makes them unsuitable for the on-line computation.

When a suitable fundamental model is unavailable, system identification may be the only viable approach to obtain a predictive dynamic model. The system identification process is depicted in Figure 23. First, an identification experiment is conducted to obtain plant data rich in the information relevant for the intended dynamic optimization. After outlier removal and appropriate pre-filtering, data is used to select a model structure and estimate its parameters. The obtained model is validated. Based on the outcome, the whole or a part of the process may have to be repeated until an adequate model is found.

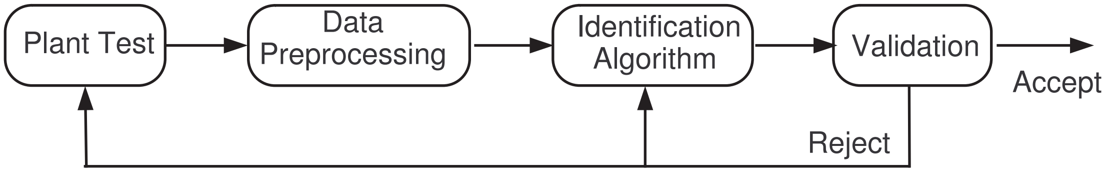


Figure 23: System identification process

Currently, there are many well-established identification techniques for linear systems. For a large dimensional multivariable system, nonparametric model identification methods such as finite impulse response (FIR) model identification require a very large number of parameters to be estimated. As shown in [38], the variance of an identified model is proportional to the number of estimated parameters. Large parametric uncertainty could result in a nonparsimonious model, which can lead to a robustness problem. Recently, subspace identification method, which identifies a linear state-space model from input-output plant test data, has attracted much attention in identifying a MIMO system. Most subspace identification algorithms attempt to construct the state variables via certain projection of data matrices. Once the state data is constructed, it is straight forward to fit a state space model. This method involves numerically simple linear algebra, unlike most other parametric identification approaches that require solving iterative optimization problems to find parameters that best fit the given data. In addition, according to the general algorithm of this approach shown in [31, 32], one can automatically obtain a parsimonious model from this algorithm.

Despite advances in subspace identification algorithms, designing a good identification experiment for an integrated plant is not simple. Two requirements need to be fulfilled in order to have a model with good long-term prediction accuracy. First, input signals must excite the dynamic system sufficiently in order to obtain information rich in the relevant dynamic behavior of the system. Secondly, the duration of the plant experiment has to be adequate so that the slow-scale dynamic behavior of the system be captured in the data. However, the characteristics of an integrated plant make those conditions difficult to be fulfilled in practice.

4.1.1 Difficulties Associated with Identifying a System with Very Long Time Constant

One of the biggest concerns in multivariable system identification of an industrial process is the cost of conducting a plant test. This is one of the motivations for the departure from the traditional step test experiments, where a step signal is injected to each MV separately and the output responses are used to identify the transfer function between each input and output pair. Zhu [39] had studied an identification problem of a crude distillation process consisting of three distillation columns in series with 19 MVs, 36 CVs and the settling time of 2.5 hours. The study reported that when the step test method was used to identify the system model, the entire experiment required as long as 2 weeks to complete. However, the model derived from this approach still showed poor prediction quality [40]. This is often the case because the multivariable relationship cannot be capture when only a single input is perturbed at a time. In an effort to improve the identification process of multivariable systems, pseudo random binary signals (PRBS) are normally used as test signals and MVs are perturbed simultaneously. An existing guideline for identifying models for MPC application in [40] suggested that the optimal average switching time of the PRBS signal should be about 1/3 of the estimated process settling time. For the same crude distillation process, the PRBS test lasted for about 2 days.

In the case of identifying a model of an integrated plant with recycle, we cannot use the existing guideline of the MPC model identification. This is because the plant settling time can be several days, and it is unlikely that one could hold the plant in the test mode for such a long period of time. We may not even expect to hold the plant test as long as the plant's largest time constant. This will result in an identified model that shows poor prediction quality in the low frequency range. When it is used in the plant-wide predictive control, the controller performance can be severely compromised. One may argue that the model can still be used in the plant-wide control by reducing the controller prediction horizon so that the inferior long-term prediction information is not considered. Nonetheless a systematic approach is needed to improve the long-term prediction quality of the model, instead of trading off the long-term prediction capability, because the long term model prediction information can be very useful for inventory planning and other higher level decision making. As a result, in Section 4.3 we present a method to improve the long-range model prediction quality by incorporating the available steady-state model information of the system. Oftentimes in practice, a steady-state description of the plant is available or can be obtained, such as from material balances, thermodynamic equations, flowsheet simulator, etc. In such cases, it is sensible to try to incorporate such prior knowledge to improve the model quality.

4.1.2 Directionality of an Integrated Plant

In this section we want to show that a plant with large recycle stream can exhibit ill-conditioned behavior. One way to examine the ill-conditioning of a system is to perform a singular value decomposition on the steady-state gain matrix of the system. For example, to identify a model of the reactor-storage-flash separator system shown in Chapter 3, it is natural to make equal step changes of one unit in the feed and the recycle flow rates, which would result in an equal amount of change in the internal flow rate (F_R and F_M) of the system. The steady-state gains of the concentrations observed from this experiment are as follows:

$$G_{ss} = \begin{bmatrix} 0.0518 & -0.0009 \\ -0.0491 & 0.0010 \\ 0.0518 & -0.0009 \\ -0.0491 & 0.0010 \\ 0.2970 & -0.0035 \\ -0.2259 & 0.0019 \end{bmatrix} \tag{18}$$

where G_{ss} denotes the steady-state gain matrix. The first column in the gain matrix in Eq. 18 represents the gains from a step change in the feed, and the second column is the gains from the recycle. The outputs are $x_{1R}, x_{2R}, x_{1M}, x_{2M}, x_{1B}$ and x_{2B}. The singular value decomposition of these steady-state gains is:

$$G_{ss} = U\Sigma V^T, \tag{19}$$

$$\text{where} \qquad U = [u_1, u_2] = \begin{bmatrix} 0.13409 & -0.34060 \\ -0.12712 & 0.45626 \\ 0.13409 & -0.34061 \\ -0.12712 & 0.45626 \\ 0.76828 & -0.16363 \\ -0.58436 & -0.56996 \end{bmatrix},$$

$$\Sigma = diag(\sigma_1, \sigma_2) = \begin{bmatrix} 0.3866 & 0 \\ 0 & 0.0011 \end{bmatrix},$$

$$V = [v_1, v_2] = \begin{bmatrix} 0.9999 & 0.0111 \\ -0.0111 & 0.9999 \end{bmatrix}.$$

where U is an orthonormal output rotational matrix, Σ is a matrix of singular values, and V is an orthonormal input rotational matrix. This decomposition shows that changes along the first input direction, which is an external flow change, has a much greater impact on the system's output at the steady state than changes along the second input direction, which is an internal flow change. Note that the ratio of the singular values σ_1/σ_2, or the *condition number*, of the system is $0.3866/0.0011 = 351$. This presents a problem during the identification experiment because output changes along the direction v_2 would be very small, and hence its effect can be completely masked by the bigger effect and system noise in real situations. This would be critical in on-line optimization because poor model accuracy along the low gain direction can result in an adverse control decision. Note that this singular value analysis of the system's steady-state gain matrix is by no mean a complete description of the directionality problem in the integrated plant. The singular values and the rotational matrices vary along the frequencies. But in general the condition number is high over a medium and low frequency range as shown in Figure 24 for this example and this is consistent with what was reported in [41] for other systems.

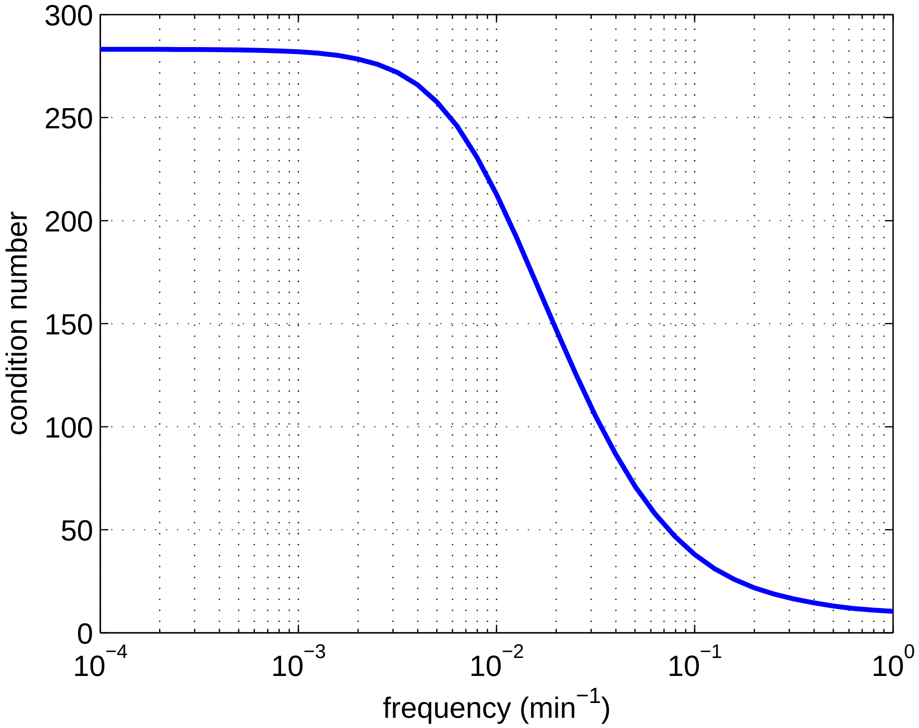


Figure 24: Condition number $\|\sigma_1(j\omega)/\sigma_2(j\omega)\|$ of the reactor-flash separator example.

This directionality problem in an integrated plant resembles that of the high-purity distillation system posed by Skogestad and Morari [42] and studied in the identification context during the past several years by several researchers [43, 44, 45, 41, 46]. Because the input design is very important in getting informative data of the system's behavior, the literature review on the input magnitude design for an ill-conditioned system is presented in Section 4.2. These approaches primarily use the approximation of the singular values and the directionality of the steady-state gain matrix to design the magnitude of input perturbation signal. Even though, the directionality and the condition number do not stay same over the whole frequency spectrum, we base the input design on the steady-state information because in our application we use the model to make a long range prediction of the system's slow-scale behavior. Over the frequency spectrum of our interest, we can expect the condition number to be high and the gain directionality to remain similar to that of the steady-state. Therefore, some of the techniques will be incorporated into the identification of an integrated plant model and we demonstrate the impact of input design in Section 4.4.

4.2 Guideline on Input Design for Ill-conditioned System Identification

The work presented in [43] and [41] have demonstrated that when using a traditional PRBS input design that perturbs all inputs by equal magnitude can result in nearly collinear output perturbations. This is because changes along the high gain direction will show up in a much more pronounced way than the low gain direction.

Consider the system

$$y(s) = G(s)\Delta MV(s) \tag{20}$$

where y and ΔMV is magnitudes of change of the output and the manipulated variable, respectively. Let the singular value decomposition of the model be represented by $G(s) = U(s)\Sigma(s)V(s)^T$. To avoid the collinearity problem, all output directions should contribute equally to the data. That is, by Parseval's theorem, it is suggested that

$$\int_0^\infty \|u_i(j\omega)\|^2|\sigma_i(j\omega)|^2|v_i(j\omega)^T\Delta MV(j\omega)|^2 d\omega = \text{constant},\ \ \forall i = 1,\ldots,\text{rank}[G(s)] \tag{21}$$

Because u_i are orthonormal, assuming

$$|\sigma_i(j\omega)||v_i(j\omega)^T\Delta MV(j\omega)| = \text{constant},\ \ \forall i = 1,\ldots,\text{rank}[G(s)], \forall\omega \tag{22}$$

would satisfy the condition given in Eq. 21. In the guidelines provided in [43] and [41], the authors suggested designing the magnitudes of the rotated inputs $|v_k^T\Delta MV|$ based on the ratio of the steady-state singular values. That is,

$$\frac{|v_k^T\Delta MV|}{|v_1^T\Delta MV|} = \frac{\sigma_1}{\sigma_k}, \qquad \forall k, k \neq 1 \tag{23}$$

Bruwer and MacGregor [46] extended this concept to consider process constraints when designing the magnitude of the input for the high-purity distillation column example. They also suggested reducing the input magnitude ratios to 10 percent of the singular value ratios as in Eq. 24 to recognize the fact that the initial estimate of input rotational vectors could be inaccurate.

$$\frac{|v_k^T \Delta MV|}{|v_1^T \Delta MV|} = 0.1\frac{\sigma_1}{\sigma_k}, \qquad \text{for } k = 2, \ldots, rank[G(s)] \tag{24}$$

If that is the case and the condition number is very high, the design guideline from Eq. 23 can result in a large perturbation along the high-gain direction. However, there is no rigorous reason given on why 10% is the chosen value and whether this is generalizable to other systems.

In our work, we recognize the importance of input magnitude design as discussed previously. One could argue that since these input magnitude designs were based on the steady-state directionality, it may not be optimal across all the frequencies because the directionality and the condition number do vary as mentioned before. Nonetheless, one generally does not have any information on the directionality of the system at other frequency before the dynamic model is obtained. Therefore, the steady state gains, which can be used to identify the steady-state gain directionality, are often the only information available for designing the input signal. In addition, because our plant-wide dynamic model is intended to be used over the low-frequency range, designing the inputs to excite the steady-state's low-gain directions will generally give better results than completely ignoring the gain directionality as will be shown in Section 4.4.

4.3 Proposed Grey-box Modeling Method

Because a model obtained from a short experiment can suffer from poor quality of long-term prediction, in this section we suggest incorporating into system identification the information of the steady-state gains and the settling time of the system. In our approach, one first performs an identification experiment on the integrated plant up to an allowable period of time, which might be up to 2-3 days, with appropriately designed input signals as discussed in the previous section. Then, the data is used to identify a dynamic model. Because the high-frequency model prediction quality should not be affected by truncating the experiment, we can use this model for short-range predictions by parameterizing it as a step response model truncated at the time when the prediction accuracy is expected to start to degrade. Then the residual dynamics are approximated as first-order dynamics and augmented to the truncated step response, while ensuring that the settling gains of the augmented model match the steady-state gains from the prior knowledge. Note that the idea of approximating the slow-scale dynamics as a first-order system here resembles the work presented by Hovd and coworkers [47]. However, in their work the entire step response coefficients for an integrating system were available, and they attempted to derive a compact representation of this system for the MPC calculation by truncating the step response model and approximating the residual dynamics as a first-order system. They used the slope of the step response before the truncation point to find the poles of the first-order system. On the contrary, the purpose of approximating the residual dynamics as a first-order system here is to extend the prediction capability of an identified model. In addition, the first-order model parameters are derived so as to ensure that the long-term prediction from the model converges to that predicted by the known steady state gain of the plant.

4.3.1 Model Prediction Formulation

Let the identified model be represented by

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned} \tag{25}$$

Suppose the obtained model is reliable up to the time step n. In other words, the model can be used to predict a step response up to this time. The step response coefficient matrix can be built from the model in Eq. 25, and the output prediction up to n future samples can be derived as follows:

$$\begin{bmatrix} y(k+1) \\ y(k+2) \\ \vdots \\ y(k+n-1) \\ y(k+n) \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{n-1} \\ S_n \end{bmatrix} \Delta u(k) \tag{26}$$

where

$$S_i = \begin{bmatrix} S_{1,1,i} & S_{1,2,i} & \dots & S_{1,n_u,i} \\ S_{2,1,i} & S_{2,2,i} & \dots & S_{2,n_u,i} \\ \vdots & \vdots & \ddots & \vdots \\ S_{n_y,1,i} & S_{n_y,2,i} & \dots & S_{n_y,n_u,i} \end{bmatrix}, \qquad i = 1, \dots, n \tag{27}$$

The coefficient $S_{k,l,i}$ is the i^{th} step response coefficient of the l^{th} manipulated variable on the k^{th} output. n_y and n_u are numbers of output and manipulated variables, respectively.

To extend the prediction up to a longer horizon, some physical insight about the system's residual dynamics is required. It is known that a system with recycle has poles that are close to the origin, which causes the system to slowly approach the steady state. In this work we assume that the residual dynamics can be approximately modeled as a first-order system. Hence, there are only two parameters, the steady state gains and the dominant time constant, required to specify the model. Let the steady state gains of the system be denoted by G_{ss}. Then, the gain of the residual dynamics (K) is the equal to $K = G_{ss} - S_n$. The dominant time constant of the system can also be approximated from the settling time as $\tau = (1/4)(T_{ss} - n\Delta t)$, where T_{ss} is the settling time of the plant and Δt is the sample time of the step-response model. The residual dynamics of each input and output pair is to be modeled as:

$$y_k^r(s) = \frac{K_{kl}}{\tau s + 1} \Delta u_l(s) \tag{28}$$

where $k = 1, \dots, n_y$ and $l = 1, \dots, n_u$. This can be written in the discrete-time domain as

$$y_k^r(z) = \frac{b_{kl}}{z - a_{kl}} \Delta u_l(z) \tag{29}$$

In the above equation, the residual dynamics are assumed to be decoupled. a_{kl} and b_{kl} are the discrete-time model parameters between each output k and input l. The residual dynamics are to be augmented onto the initial step response model obtained from identification as shown in Figure 25.

Let A_T, B_T, C_T be a state space model of the residual dynamics that describes the effect of changes in the manipulated variables on the outputs after the truncation point at the

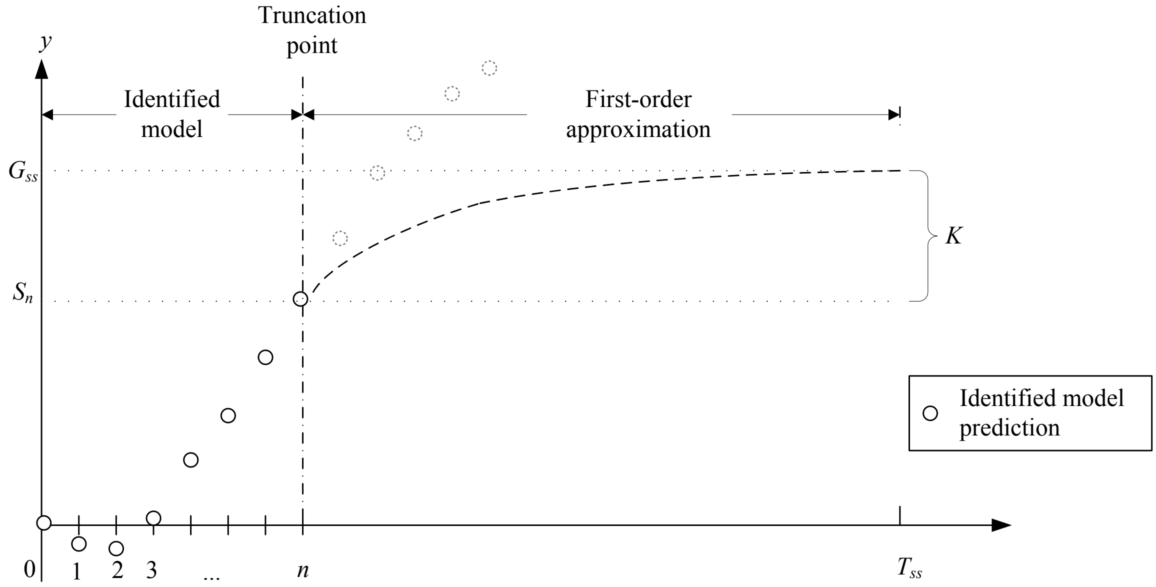


Figure 25: The suggested grey-box identification approach

n^{th} sample into the future. The augmented prediction model up to the p^{th} $(p > n)$ sample time takes the following form:

$$\begin{bmatrix} y(k+1) \\ y(k+2) \\ \vdots \\ y(k+n-1) \\ y(k+n) \\ y(k+n+1) \\ \vdots \\ y(k+p) \end{bmatrix} = \mathcal{M}^p \underbrace{\begin{bmatrix} \tilde{y}(k|k) \\ \tilde{y}(k+1|k) \\ \tilde{y}(k+2|k) \\ \vdots \\ \tilde{y}(k+n-1|k) \\ x_T(k|k) \end{bmatrix}}_{\tilde{Y}(k|k)} + \mathcal{S} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+m) \end{bmatrix} \tag{30}$$

where $\tilde{y}(i|k)$ is the expected process output at time i based on the input and the measurement information up to time k from the memory, x_T represents the state of the residual dynamics through which the change in the input enters,

$$\mathcal{M}^p = \begin{bmatrix} 0 & I & 0 & \cdots & 0 & 0 \\ 0 & 0 & I & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & I & 0 \\ 0 & 0 & 0 & \ddots & I & C_T A_T \\ 0 & 0 & 0 & \ddots & I & C_T \sum_{i=1}^{2} (A_T)^i \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & I & C_T \sum_{i=1}^{p-n} (A_T)^i \end{bmatrix} \tag{31}$$

$$\mathcal{S} = \begin{bmatrix} S_1 & 0 & \cdots & 0 \\ S_2 & S_1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ S_m & S_{m-1} & \ddots & S_1 \\ S_{m+1} & S_m & \ddots & S_2 \\ \vdots & \ddots & \ddots & \vdots \\ S_n & S_{n-1} & \ddots & S_{n-m+1} \\ S_n + C_T B_T & S_n & \ddots & S_{n-m+2} \\ S_n + C_T \sum_{i=0}^{1} (A_T)^i B_T & S_n + C_T B_T & \ddots & S_{n-m+3} \\ \vdots & \ddots & \ddots & \ddots \\ S_n + C_T \sum_{i=0}^{p-n-1} (A_T)^i B_T & S_n + C_T \sum_{i=0}^{p-n-2} (A_T)^i B_T & \ddots & \ddots \end{bmatrix}. \tag{32}$$

With the first-order approximation, the state space model of the residual dynamics can be constructed as follows.

$$A_T = \begin{bmatrix} a_{11} & & & \\ & a_{21} & & \\ & & \ddots & \\ & & & a_{n_y n_u} \end{bmatrix} \tag{33}$$

$$B_T = \begin{bmatrix} \begin{bmatrix} b_{11} \\ \vdots \\ b_{n_y 1} \end{bmatrix} & & \\ & \ddots & \\ & & \begin{bmatrix} b_{1n_u} \\ \vdots \\ b_{n_y n_u} \end{bmatrix} \end{bmatrix} \tag{34}$$

$$C_T = \overbrace{\begin{bmatrix} I_{n_y} & I_{n_y} & \ldots & I_{n_y} \end{bmatrix}}^{n_u} \tag{35}$$

Once a sequence of optimal MV moves is calculated, the first move is implemented to the plant. Then memory vector can be updated as follows:

$$\tilde{Y}(k|k) = \begin{bmatrix} 0 & I & \cdots & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & 0 & I & 0 \\ 0 & 0 & \ddots & 0 & I & C_T A_T \\ 0 & 0 & \ddots & 0 & 0 & A_T \end{bmatrix} \tilde{Y}(k-1|k-1) + \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_n \\ C_T B_T \\ B_T \end{bmatrix} \Delta u(k-1)$$
$$+ K(y_m(k) - \tilde{y}(k|k-1)) \tag{36}$$

where $\tilde{Y}(k|k) = [\tilde{y}(k|k)^T, \tilde{y}(k+1|k)^T, \ldots, \tilde{y}(k+n-1|k)^T, \tilde{y}(k+n|k)^T, x_T(k|k)^T]^T$. The measurement vector at time k is denoted by $y_m(k)$. The filter gain matrix K can be parameterized as follows:

$$K = \begin{bmatrix} \mathbf{I} \\ 0 \end{bmatrix} \begin{bmatrix} f_1 & & \\ & \ddots & \\ & & f_n \end{bmatrix} \tag{37}$$

$$\mathbf{I} = \overbrace{\begin{bmatrix} I_{n_y} & I_{n_y} & \ldots & I_{n_y} \end{bmatrix}}^{n} \tag{38}$$

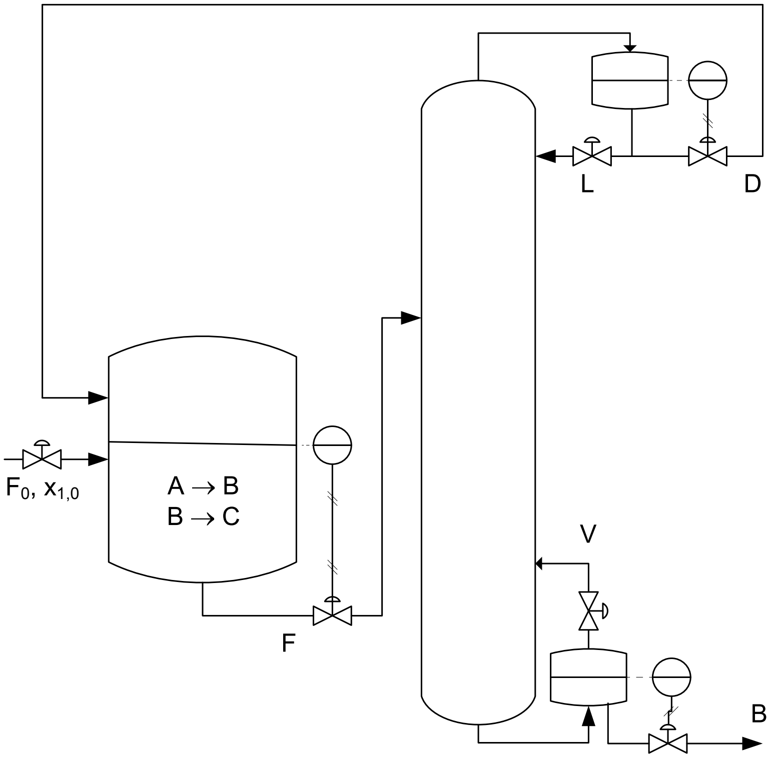


Figure 26: Reactor-distillation example for identification

4.4 Illustrative Example

In this section we use an example of an integrated plant with a reactor and a distillation column studied by Kumar and Daoutidis [48] to show the efficacy of the proposed modeling method. In this example, the feed stream consists of reactant A, which reacts to yield a desired product B in the reactor. Impurity product C is also generated by a side reaction. The reactor effluent is fed to the 4^{th} tray of a 15-tray distillation column, which separates the desired product B as the bottom product and recycles back the distillate containing primarily the reactant A. We assume the levels of the reactor, the condenser, and the reboiler are perfectly controlled by the regulatory control scheme shown in Figure 26.

There are 3 manipulated variables for plant-wide control including the feed (F_0), the reflux flow rate (L), and the vapor flow rate (V). The concentrations in the reactor, the condenser, and the product stream are denoted by x_{ij}, where i indicates chemical species (1: reactant A, 2: desired product B, 3: undesired product C) and j represents locations (R:

Table 8: Nominal values for the process variables of the reaction-distillation integrated plant example

feed flow rate (F_0)	100 hr^{-1}	reactor outlet flow rate (F)	1880 hr^{-1}
recycle flow rate (D)	1780 hr^{-1}	reflux flow rate (L)	290 hr^{-1}
vapor boilup rate (V)	2070 hr^{-1}	product flow rate (B)	100 hr^{-1}
kinetic rate 1 (k_1)	1	kinetic rate 2 (k_2)	1
volatility constant 1 (α_A)	4	volatility constant 2 (α_B)	2
reactor holdup (M_R)	110	condenser holdup (M_D)	173
reboiler holdup (M_B)	181	tray liquid holdup (M_T)	175
x_{1R}	0.8996	x_{2R}	0.0939
x_{1D}	0.9496	x_{2D}	0.0494
x_{1B}	0.0104	x_{2B}	0.8863

Table 9: Hard limits on the flow rates of the reactor-distillation plant

	F_0	L	V	F	D	B
nominal value	100	290	2070	1880	1780	100
lower limit	20	20	500	500	500	10
upper limit	300	1340	3570	3570	3570	300

reactor, D: condenser, B: reboiler, 0: feed). There are 39 nonlinear differential equations that govern the dynamics of this system as will be shown in Section 5.2. The nominal operating point of the system is summarized in Table 8. In addition, we specify the hard constraints on the flow rates as shown in Table 9.

For the purpose of demonstrating the linear identification method, we linearize the model around this operating point and use it as a virtual plant for the identification. The state-space model representing this plant can be found in Appendix A.

4.4.1 Problem Statement

In this example, measured output variables are $y = [x_{1R},\ x_{2R},\ x_{1D},\ x_{2D},\ x_{1B},\ x_{2B},\ B]^T$. With the perfect level control assumption, the production rate is equal to the feed. The identification experiment is, therefore, intended to produce a model between the concentrations and the MVs. In this case, the system exhibits a strong two-time scale behavior. The residence times of the reactor and the distillation column calculated from the capacities divided by the throughputs are approximately 4 and 50 minutes, respectively. However, the plant's settling time is on the order of several days. In addition, we observe strong gain directionality in the input and the output changes. Figure 27 shows the condition number of this system as a function of frequency.

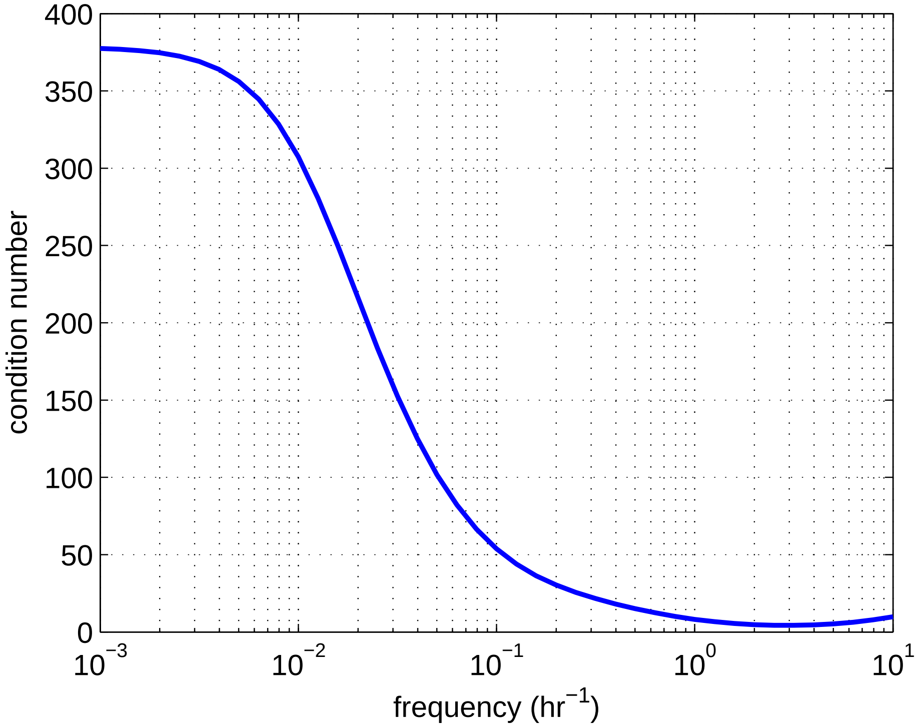


Figure 27: Condition number of the linearized reaction-distillation integrated plant

We define the output and manipulated input vectors as

$$y = [\ x_{1R} \quad x_{2R} \quad x_{1D} \quad x_{2D} \quad x_{1B} \quad x_{2B}\ ]^T, \tag{39}$$

$$u = [\ F_0 \quad L \quad V\ ]^T \tag{40}$$

Results of the singular value decomposition of the steady state gains are shown in Eqs. 41-43.

$$U = \begin{bmatrix} -0.43456 & -0.40668 & -0.25079 \\ 0.40501 & 0.31760 & 0.22842 \\ -0.48070 & 0.26074 & 0.48836 \\ 0.47485 & -0.25729 & -0.41437 \\ -0.41981 & 0.33657 & -0.62845 \\ -0.11967 & -0.69734 & 0.28246 \end{bmatrix} \tag{41}$$

$$\Sigma = 10^{-2} \cdot \begin{bmatrix} 1.0980 & 0 & 0 \\ 0 & 0.0046 & 0 \\ 0 & 0 & 0.0029 \end{bmatrix} \tag{42}$$

$$V = \begin{bmatrix} -1.00000 & -0.00050 & -0.00267 \\ -0.00272 & 0.18968 & 0.98184 \\ -0.00001 & -0.98185 & 0.18968 \end{bmatrix} \tag{43}$$

The input rotational matrix suggests that the strongest input direction is governed by the change in the feed, which amounts to a change in the external flow to the system. The internal flow changes (L and V) correspond to the low-gain input directions. Large differences in the singular values mean it requires relatively large input moves in order to make the system change along the second and the third output directions.

In the next section, we proceed to identify the model of this system and use it for plant-wide control. In the first case, we examine the conventional approach that applies perturbations of magnitudes to all the input variables. The obtained model will be used on its own, as well as with the augmentation of the first-order residual dynamics for the plant-wide control as described earlier. In this work, we will use a MPC running at a slow rate as a plant-wide controller. Then, in the second case the input is designed to perturb the low gain directions more than the high gain direction and the control performances will be compared. The test parameters and conditions to be kept in common are as follows:

- Test duration: In each case, we limit the identification experiment to last up to 40 hours. This amount of time is much shorter than the plant's settling time. However, longer experiments would incur much more cost in practice. During the experiment, the interval between the input signal switching is 12 minutes.

- Measurement noise: Measurement noise is present in all the output channels with the magnitudes ±10% of the signals.
- Unmeasured feed disturbances: Feed impurities include components B and C, which are not measured. Their changes are modeled as two random independent integrated white noise processes:

$$\eta_1(k) = \eta_1(k-1) + e_1(k) \tag{44}$$

$$\eta_2(k) = \eta_2(k-1) + e_2(k) \tag{45}$$

  where $e_1(k)$ and $e_2(k)$ are white noises of $\mathcal{N}(0, 5 \times 10^{-5})$.

- MPC parameters: The sample time for the MPC is 1 hour. The prediction and control horizons are 24 and 8, respectively. Note that the controller sample time was chosen to be larger than the residence times of both the reactor and the distillation column. This is so that the plant-wide controller tracks changes that are in the slower time scale corresponding to the plant-wide interaction introduced by the recycle stream. The optimization problem is written as follows:

$$\min_{U(k)} (Y^{sp} - Y(k+1|k))^T \mathcal{Q}^T \mathcal{Q} (Y^{sp} - Y(k+1|k)) + \Delta U(k)^T \mathcal{R}^T \mathcal{R} \Delta U(k) \tag{46}$$

$$\begin{aligned} & Y_{min} \leq Y(k) \leq Y_{max} \\ & U_{min} \leq U(k) \leq U_{max} \end{aligned} \tag{47}$$

where

$$\begin{aligned}
Y(k+1|k) &= \left[\begin{array}{cccc} y(k+1|k)^T, & y(k+2|k)^T, & \ldots, & y(k+24|k)^T \end{array}\right]^T, \\
U(k) &= \left[\begin{array}{cccc} u(k)^T, & u(k+1)^T, & \ldots, & u(k+7)^T \end{array}\right]^T, \\
y &= \left[\begin{array}{ccccccc} \frac{x'_{1R}}{\bar{x}}, & \frac{x'_{2R}}{\bar{x}}, & \frac{x'_{1D}}{\bar{x}}, & \frac{x'_{2D}}{\bar{x}}, & \frac{x'_{1B}}{\bar{x}}, & \frac{x'_{2B}}{\bar{x}}, & \frac{B'}{\bar{B}} \end{array}\right]^T, \\
u &= \left[\begin{array}{ccc} \frac{F'_0}{\bar{F}_0}, & \frac{L'}{\bar{L}}, & \frac{V'}{\bar{V}} \end{array}\right]^T
\end{aligned}$$

The $'$ denotes the deviation variable. The scaling factors $\bar{x}, \bar{B}, \bar{F}_0, \bar{L}$ and $\bar{V}$ are 0.1, 10, 10, 30, and 100, respectively. The plant-wide controller tries to manipulate x_{1D}, x_{2B} and B to the setpoints. The weighting matrices are chosen as follows:

$$\mathcal{Q} = diag([\underbrace{\begin{array}{cccc} Q, & Q, & \ldots, & Q \end{array}}_{24}]) \tag{48}$$

$$\mathcal{R} = diag([\underbrace{\begin{array}{cccc} R, & R, & \ldots & R \end{array}}_{8}]) \tag{49}$$

where

$$Q = diag([0,0,50,0,0,50,50]^T) \tag{50}$$

$$R = I_{n_u \times n_u} \tag{51}$$

where $diag(\cdot)$ is an operator that diagonalizes the elements inside.

4.4.2 Case 1: Model Identification with Equally Perturbed Input Signals

To design the input magnitude, we first consider the physical constraints of the system. Since the lower bound on the production rate is the tightest among the hard constraints, it is used to calculate the input magnitude of change. The steady-state mass balance suggests that $B = F + L - V$. To keep the production rate above 10, the magnitude of the production rate perturbation from the nominal value must be less than or equal to 90 . As a result, the input perturbation signals in this scheme will be kept as ±30 to comply with this constraint. The input perturbation signals are PRBSs with the switching time of 12 minutes. We performed 50 disturbance realization scenarios and identified the model using the N4SID subspace identification algorithm in MATLAB^{TM}.

The N4SID algorithm in Matlab always suggested models of order 3 for all the 50 realizations. However, during the model validation, such model order gave a very poor prediction result. The low model order resulted from the fact that the output data might be collinear because the low-gain output directions were not strongly perturbed. Therefore, we purposely increased the identified model order to 5. The model prediction results from one of the realizations are shown in Figures 32-34, which clearly show that the predictions of the output responses to the feed change were much more accurate than those of the other input changes. In all the other realizations, the same model characteristics were observed. This can be attributed to the fact the plant is ill-conditioned and the strong input direction was dominated by the feed in the medium and low frequency ranges. Therefore, when all the inputs were equally perturbed during the identification experiment, the feed change would contribute more to the output data. As a result, the prediction model is not very accurate in the directions of the reflux and the reboiler changes.

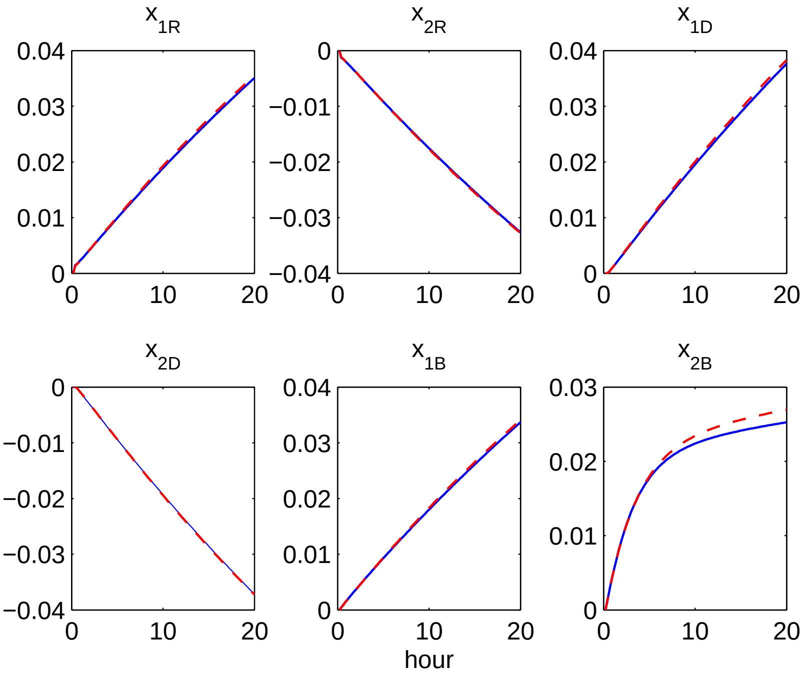


Figure 28: Model prediction (dashed) vs. actual plant response (solid) for a 30-unit step change in the feed

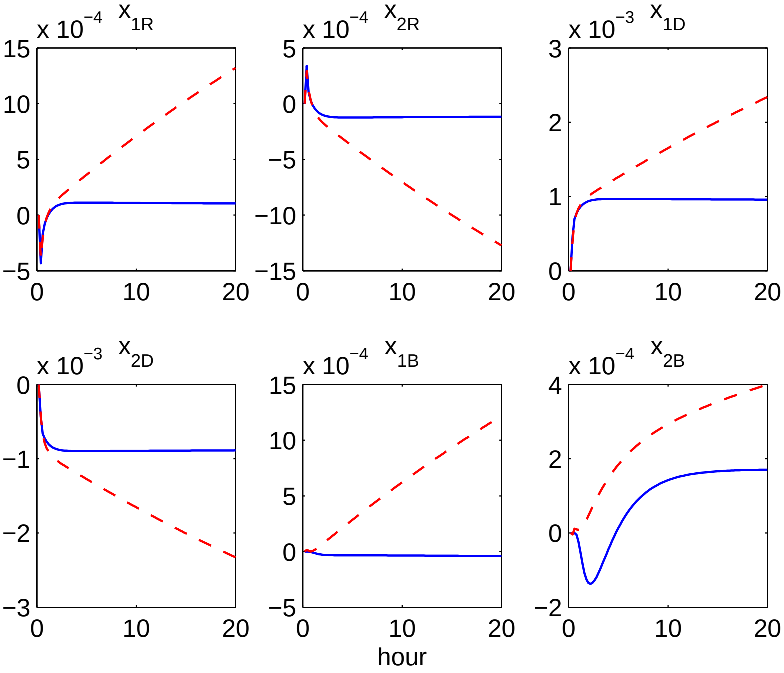


Figure 29: Model prediction (dashed) vs. actual plant response (solid) for a 30-unit step change in the reflux

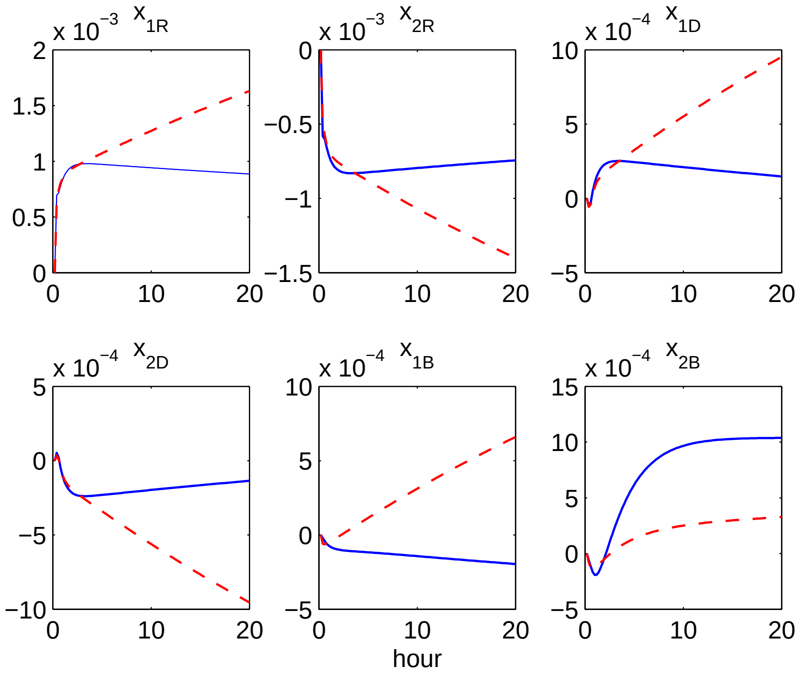


Figure 30: Model prediction (dashed) vs. actual plant response (solid) for a 30-unit step change in the vapor boilup

4.4.2.1 Plant-wide Control Results

In this section we tested several setpoint changes by the MPC using the identified model. The performance is measured by Eq. 52.

$$\text{performance} = \sum_{i=1}^{40} \left[(Y^{sp} - y(i))^T Q^T Q (Y^{sp} - y(i)) + \Delta u(i)^T R^T R \Delta u(i) \right] \tag{52}$$

For the case of the grey-box model, the model obtained from N4SID was used for prediction during the initial period of 5 hours. After that the first-order approximation is used to extend the prediction up to the settling gain. From the results in Table 10 the mean performance of the grey-box model showed significant improvement over the use of the identified model. In addition, the histograms in Figure 31 showed the number of realizations vs. the range of the MPC performance from all tests. Under all the setpoint change scenarios, the worst performance of the MPC resulted from the use of the original identified model. These worst case scenarios differed significantly from the mean performance.

Table 10: Grade transition performances using the model identified from equally perturbed input signals

	Setpoints for $[x_{1D}, x_{2B}, B]$			
	[0.99, 0.86, 120]	[0.98, 0.94, 110]	[0.96, 0.93, 90]	[0.98, 0.94, 100]
Original Model				
mean	717.74	386.05	344.94	402.62
standard deviation	111.52	152.94	105.54	125.66
Grey-box Model				
mean	703.77	377.09	320.50	382.08
standard deviation	52.62	76.73	33.09	59.48

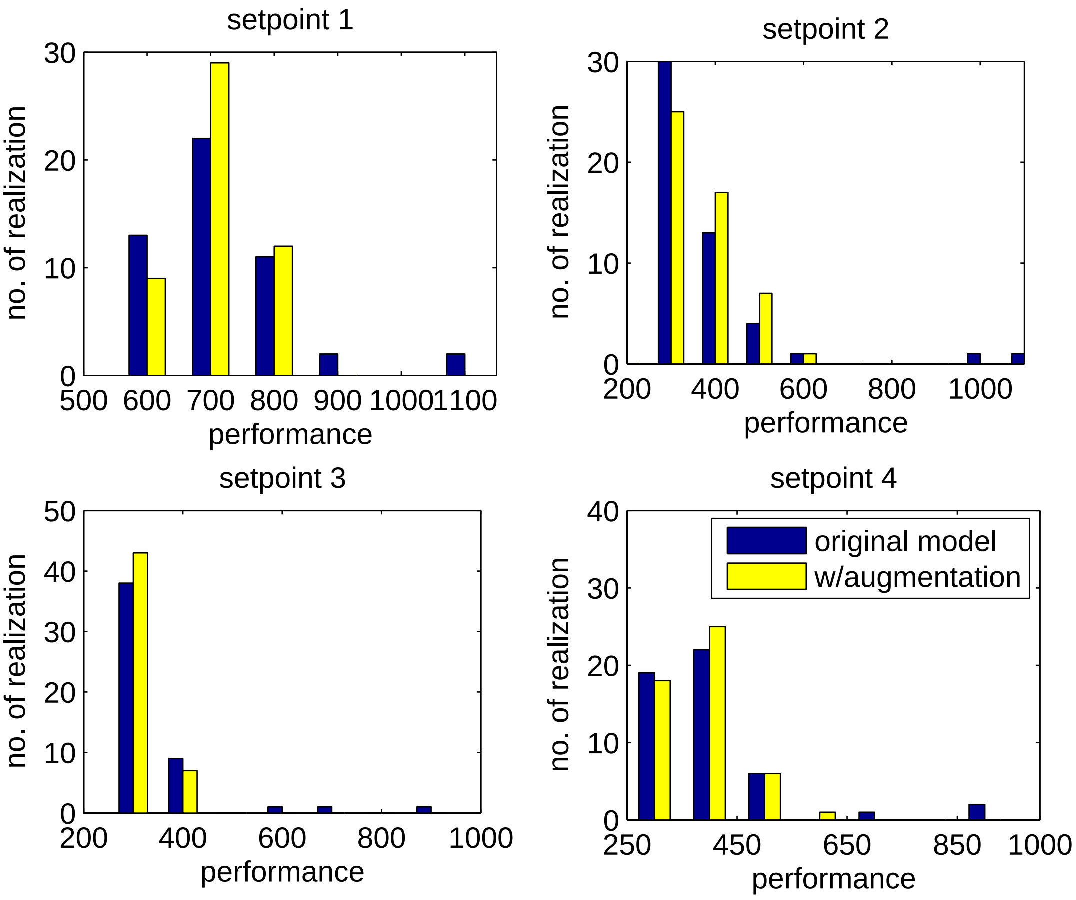


Figure 31: Performance comparison between the use of the original model and the grey-box model

4.4.3 Case 2: Model Identification with Ill-conditioned Adjusted Input Signals

One way to design the input signal to excite the low-gain direction more than the high-gain direction is to design the input magnitude according to the steady-state singular values shown in Eq. 42. Because the input variables in the input rotational matrix are almost decoupled, for simplicity we regard the 1st, 2nd, and 3rd strongest input directions as the feed, the vapor boilup, and the reflux flow rate directions, respectively. Applying the formula in Eq. 24, the desired ratios for $|\Delta L| : |\Delta F_0|$ and $|\Delta V| : |\Delta F_0|$ are 38:1 and 24:1, respectively. To keep $|\Delta B| \leq 90$, $|\Delta F_0|$ needs to be less than 1.43, which is a very small perturbation and will result in an extremely poor signal to noise ratio. In this work, we set the magnitude of $|\Delta F_0|$ to be 10 so that the signal-to-noise ratio be improved. Then, the magnitudes of $|\Delta L|$ and $|\Delta V|$ are set to 50 and 30, respectively. This input magnitude in the low gain direction may not be the as strong as suggested by the guideline, but will give an improved performance compared to the previous case while respecting the process constraints.

We performed simulations under 50 disturbance realization scenarios and identified the models using the N4SID subspace identification algorithm in Matlab. This time the algorithm suggested the model of 6^{th} order. The model prediction results from one of the realizations are shown in Figures. 32-34, which show that the prediction of the output responses to the low-gain input directions are more accurate than those of the previous case.

4.4.3.1 Plant-wide Control Results

The performance of the plant-wide controller using the model obtained from the input signals that perturbed the weak directions more strongly is shown in Table 11 and Figure 35. In this case, the controller showed improvement in performance from the previous case, in both the mean and the worst case performance. This can be attributed to better accuracy in the prediction model. In addition, the performance of the grey-box model is slightly better than the case of using the original model.

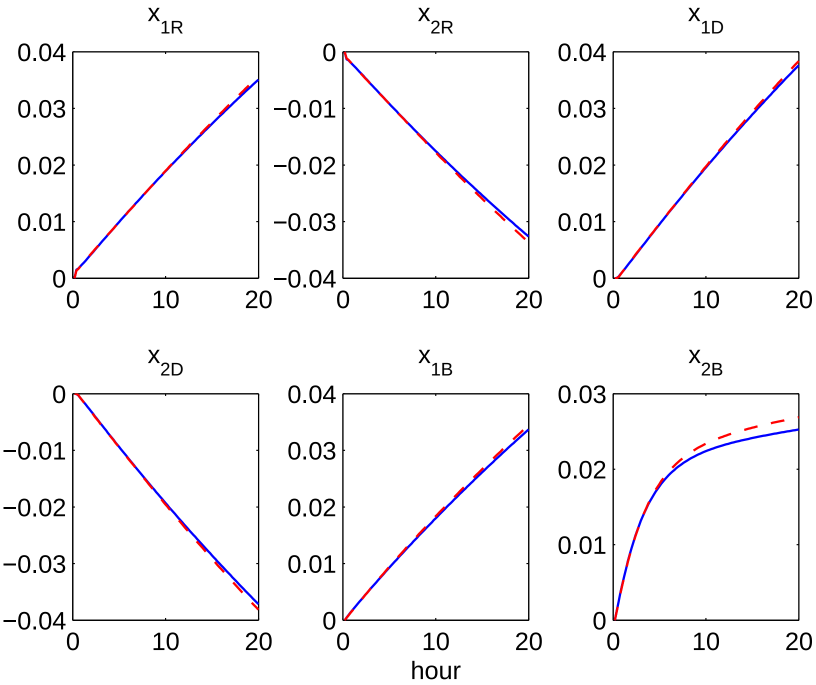


Figure 32: Model prediction (dashed) vs. actual plant response (solid) for a 30-unit step change in the feed

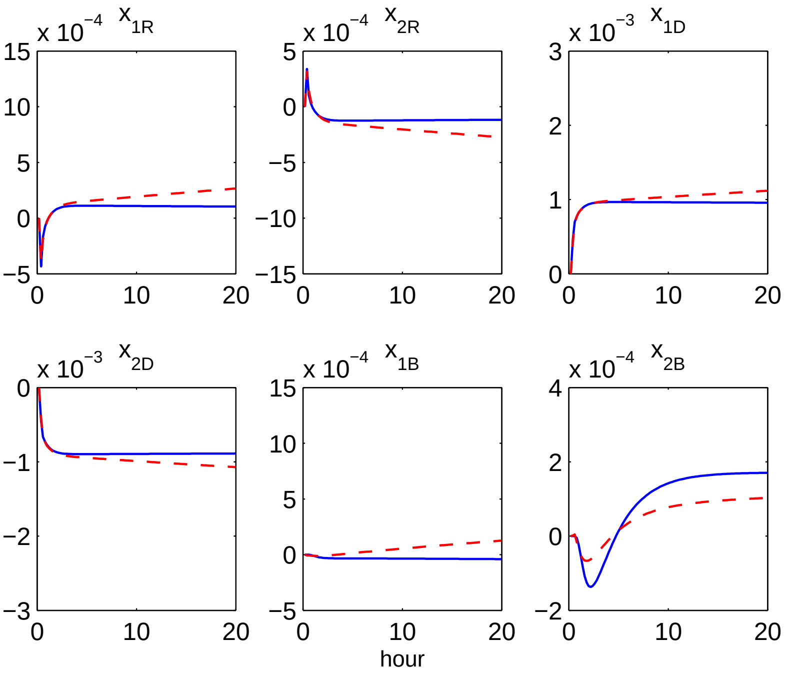


Figure 33: Model prediction (dashed) vs. actual plant response (solid) for a 30-unit step change in the reflux

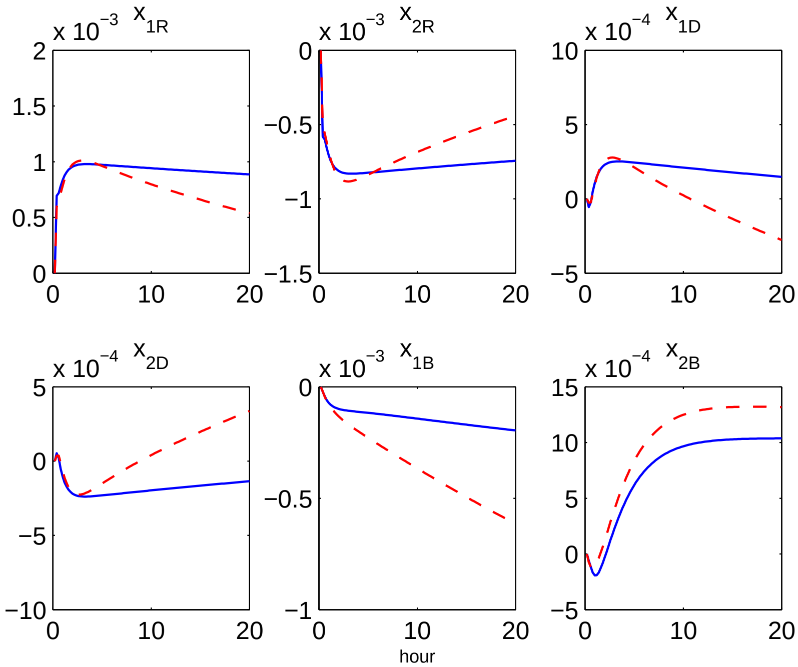


Figure 34: Model prediction (dashed) vs. actual plant response (solid) for a 30-unit step change in the vapor boilup

Table 11: Grade transition performance using the model identified from ill-conditioned adjusted input signals

	Setpoints for $[x_{1D},\, x_{2B},\, B]$			
	[0.99, 0.86, 120]	[0.98, 0.94, 110]	[0.96, 0.93, 90]	[0.98, 0.94, 100]
Original Model				
mean	680.65	347.78	294.97	383.38
standard deviation	25.78	23.36	4.86	28.70
Grey-box Model				
mean	665.81	338.54	294.00	367.59
standard deviation	18.47	15.67	2.55	17.57

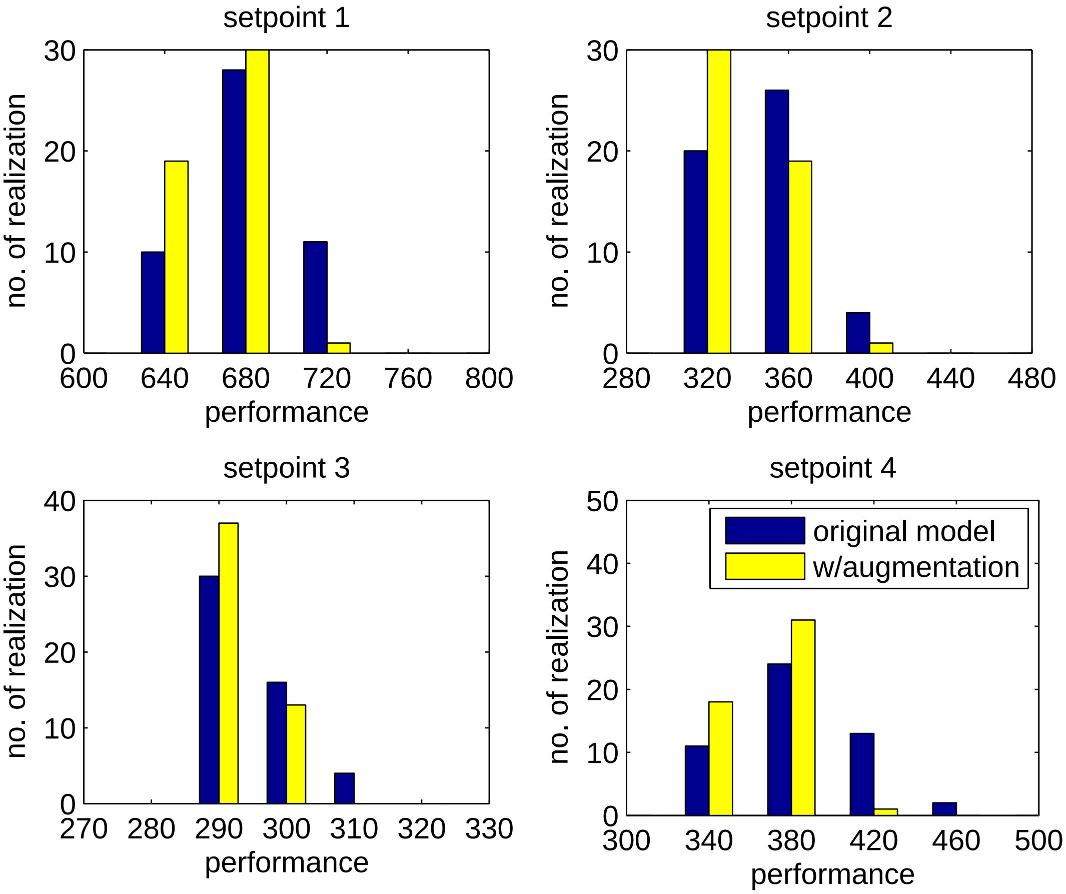


Figure 35: Performance comparison between the use of the original model and the grey-box model

4.5 Conclusions

The challenging aspects of obtaining a dynamic model of an integrated plant with a material recycle, which is very ill-conditioned, are in the experimental design and the lack of slow-scale dynamic information due to limited identification experiment time. The guideline to obtain an appropriate model for plant-wide optimization is established in this chapter. This includes a proper design of input signals to excite the low-gain directions more than the high-gain ones. In addition, the step response coefficients from the obtained model are only used for the short-term prediction. The prediction from that point onward is approximated as the first order systems that will extend the prediction to the steady state values as predicted by the gains available from the prior knowledge. The resulting method was shown to improve the performance of the plant-wide control in an integrated plant example composed of a reactor and a distillation column. The ease of design and implementation makes it potentially appealing to extend to other industrial examples.

CHAPTER 5

CASE STUDY: A NONLINEAR OPTIMAL CONTROL OF AN INTEGRATED PLANT OF REACTOR & DISTILLATION COLUMN WITH RECYCLE

So far we have discussed the proposed plant-wide optimization scheme and demonstrated its efficacy in linear system examples. Given the fact that the actual chemical process is nonlinear, the linearized model used in the optimization may only be accurate near the operating point where the model was obtained. In addition, increasing market competition will continue to force process industries to deal more and more with complex integrated processes with highly nonlinear unit operations, recycle streams, and several operating modes. In this situation, it is more appropriate to perform a multi-stage nonlinear optimization of an integrated plant. In this chapter, we present a case study of a nonlinear optimal control of an integrated plant with a reactor, a distillation column, and a material recycle loop. This example is chosen because it is a representative example of a common integrated plant that shows a multiple time scale characteristics and has high dimensional state and action spaces.

Currently, there are two conceptual frameworks for solving a nonlinear multi-stage optimal control problem: a mathematical programming framework and an approximate dynamic programming (ADP) framework. The first part of this case study is to justify why a nonlinear model predictive control (NMPC), which is the most popular mathematical programming based approach in solving a nonlinear optimal control problem in the process industries nowadays, may not necessarily be efficient in terms of the on-line computational requirement and the robustness against stochastic uncertainty. We also apply a nonlinear model reduction technique via Proper Orthogonal Decomposition (POD) as a tool to reduce the on-line computational time in solving the nonlinear model, which can moderately reduce the optimization time of NMPC. Then, the second part of this study outlines the current state of the art of the ADP method for an optimal control problem and addresses the difficulties in applying it in the way presented in the literature to this nonlinear integrated plant. Finally, we propose several methodologies to combine with the existing ADP framework to apply a nonlinear optimal control to an integrated plant and show its superior performance compared to NMPC in both deterministic and stochastic situations.

5.1 Problem Description

In this study we use a reactor-distillation-recycle system shown in Figure 36, which was first studied by Kumar and Daoutidis [48], to represent an integrated plant with a material recycle.

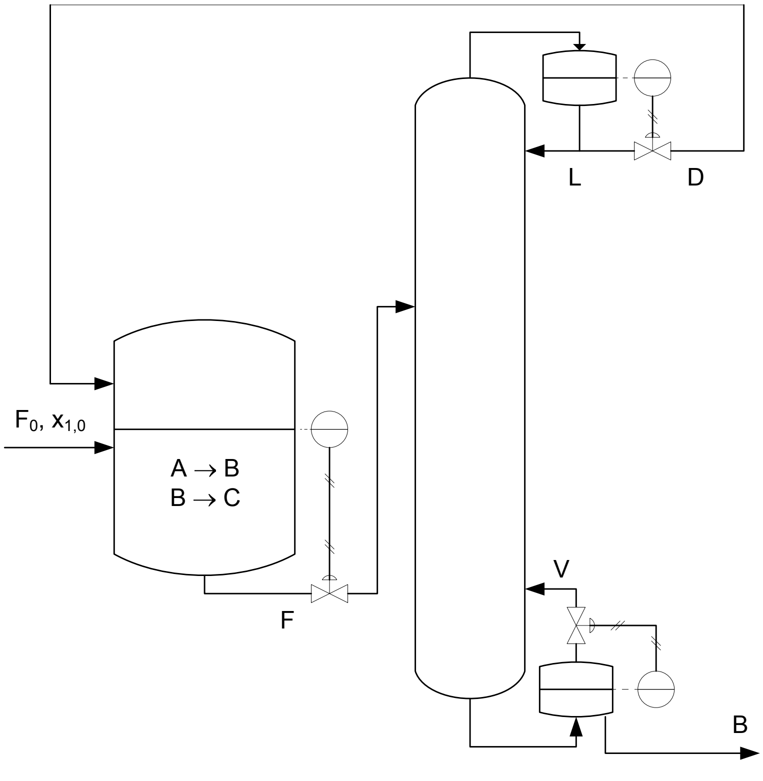


Figure 36: Process schematic of the reactor-distillation integrated plant

Note that the regulatory control scheme used in this case is slightly different from the one used in Chapter 4. The holdups in the reactor, the condenser, and the reboiler are stabilized by proportional controllers that manipulate the reactor effluence, the recycle flow rate, and the vapor boilup flow rate, respectively.

This system is described by 39 nonlinear ODEs as shown below.

$$\begin{aligned}
\dot{M}_R &= F_0 + D - F \\
\dot{x}_{1R} &= \frac{1}{M_R}\left[F_0(x_{1,0} - x_{1R}) + D(x_{1D} - x_{1R})\right] - k_1 x_{1R} \\
\dot{x}_{2R} &= \frac{1}{M_R}\left[F_0(x_{2,0} - x_{2R}) + D(x_{2D} - x_{2R})\right] + k_1 x_{1R} - k_2 x_{2R} \\
\dot{M}_D &= V - L - D \\
\dot{x}_{1D} &= \frac{V}{M_D}(y_{1,1} - x_{1D}) \\
\dot{x}_{2D} &= \frac{V}{M_D}(y_{2,1} - x_{2D})
\end{aligned}$$

For trays 1 to 3 ($1 \leq i < 4$):

$$\begin{aligned}
\dot{x}_{1,i} &= \frac{1}{M_T}\left[V(y_{1,i+1} - y_{1,i}) + L(x_{1,i-1} - x_{1,i})\right] \\
\dot{x}_{2,i} &= \frac{1}{M_T}\left[V(y_{2,i+1} - y_{2,i}) + L(x_{2,i-1} - x_{2,i})\right]
\end{aligned}$$

For the feed tray:

$$\begin{aligned}
\dot{x}_{1,4} &= \frac{1}{M_T}\left[V(y_{1,5} - y_{1,4}) + F(x_{1R} - x_{1,4}) + L(x_{1,3} - x_{1,4})\right] \\
\dot{x}_{2,4} &= \frac{1}{M_T}\left[V(y_{2,5} - y_{2,4}) + F(x_{2R} - x_{2,4}) + L(x_{2,3} - x_{2,4})\right]
\end{aligned}$$

For trays 5 to 15 ($5 \leq i < 15$):

$$\begin{aligned}
\dot{x}_{1,i} &= \frac{1}{M_T}\left[V(y_{1,i+1} - y_{1,i}) + (L+F)(x_{1,i-1} - x_{1,i})\right] \\
\dot{x}_{2,i} &= \frac{1}{M_T}\left[V(y_{2,i+1} - y_{2,i}) + (L+F)(x_{2,i-1} - x_{2,i})\right] \\
\dot{M}_B &= L + F - V - B \\
\dot{x}_{1B} &= \frac{1}{M_B}\left[(L+F)(x_{1,15} - x_{1B}) - V(y_{1B} - x_{1B})\right] \\
\dot{x}_{2B} &= \frac{1}{M_B}\left[(L+F)(x_{2,15} - x_{2B}) - V(y_{2B} - x_{2B})\right]
\end{aligned}$$

where the concentrations are denoted by x_{ij}, where i indicates chemical species (1: reactant A, 2: desired product B, 3: undesired product C) and j represents processes (R: reactor, D: condenser, B: reboiler, 0: feed, 1-15: tray location). The vapor compositions in the distillation column of components A ($y_{1,i}$) and B ($y_{2,i}$) are computed as follows:

$$y_{1,i} = \frac{\alpha_A x_{1,i}}{1 + (\alpha_A - 1)x_{1,i} + (\alpha_B - 1)x_{2,i}} \tag{53}$$

$$y_{2,i} = \frac{\alpha_B x_{2,i}}{1 + (\alpha_A - 1)x_{1,i} + (\alpha_B - 1)x_{2,i}} \tag{54}$$

where α_i is the volatility constant of component i. We define 4 operating modes for this system as shown in Table 12.

Table 12: Operating modes of the reactor-distillation integrated plant

Specifications	Mode 1	Mode 2	Mode 3	Mode 4
Product composition x_{2B}	0.886	0.85	0.905	0.82
Production rate B	100	115	85	125

Nominal values of process variables in the operating mode 1 are shown in Table 8. The feed compositions of component A ($x_{1,0}$) and component B ($x_{2,0}$) are 1.0 and 0.0, respectively. There are 6 manipulated variables for plant-wide optimization including the feed (F_0), the holdup setpoints (i.e. $M_R^{sp}, M_D^{sp}, M_B^{sp}$), the reflux flow rate (L), and the production rate (B). This system is very ill-conditioned because of the large recycle stream that causes a two-time scale behavior. The residence times of the reactor and on each tray of the distillation column are within a few minutes. However, the transient dynamics of the entire plant last 2-3 days after a step change. The objective of the optimal control is to maneuver the system from the nominal operating mode 1 to other modes, while constraining the manipulated and the output variables within their operating limits as shown in Tables 13 and 14, respectively.

Table 13: Operating range of the inputs in the reactor-distillation-recycle system

	F_0	M_R^{sp}	M_D^{sp}	M_B^{sp}	R	B
nominal value	100	110	173	181	290	100
lower limit	60	50	113	121	60	60
upper limit	140	170	233	241	350	140

Table 14: Output constraints in the reactor-distillation-recycle system

	x_{3B}	F	D	V
nominal value	0.1033	1880	1780	2070
lower limit	0	1000	1000	1200
upper limit	0.25	2400	2400	2600

5.2 Optimal Control via Nonlinear Model Predictive Controller (NMPC)

Recently, nonlinear dynamic optimization has gained a wide interest in the process system community. In particular, the nonlinear model predictive control (NMPC) is currently the accepted framework for advanced process control of nonlinear systems. In this framework, an optimal control problem over a moving time window is cast as a nonlinear program (NLP) based on a dynamic nonlinear model and the information of the state and the disturbance up to the current time. Then an on-line optimizer is used to find the optimal input profile. Currently there are two ways of solving this optimal control problem on-line: a sequential and a simultaneous approaches. The sequential approach, often called a *control vector parameterization (CVP)* method, discretizes only the control trajectory [49] and obtains the state profile by integrating the ODEs/DAEs model using the current state information as the initial condition. In other words, the state prediction is obtained by solving an initial value problem (IVP), which can be done by efficient IVP solvers such as DASSL [50], and DASSPK [51, 52]. Then a general-purpose NLP solver, such as a successive quadratic programming (SQP) method can be employed to find the optimal values of the discretized control sequence. An overview of SQP can be found in general optimization textbooks such as [53, 54, 55]. It has been reported that the computational effort of this sequential optimization technique is $\mathcal{O}\{(n_x + n_u + n_y)(p+1)\}^3$ where n_x, n_u, and n_y are numbers of state, manipulated, and output variables, respectively, and p is the prediction horizon. A more tailored decomposition technique has been proposed to solve problems that have a smaller control horizon than a prediction horizon ($m < p$), leading to the computational effort of $\mathcal{O}\{(n_u m)^3 + [(n_x + n_y)(p+1)]^3\}$, which may still be very large for a large-scale problem [56].

On the other hand, the simultaneous approach discretizes both the control and the state variables using polynomials, such as Lagrange interpolation polynomials. The polynomial coefficients then become the decision variables in a much larger NLP problem [57]. For example, if the orthogonal collocation on finite element method is used for discretization, number of variables in the optimization problem will scale with the number of elements and the number of collocation points within each element. Therefore, a special solution strategy is required to solve the resulting large-scale NLP. In this approach, the model is only solved once at the optimal point, and therefore it avoids the computation involved in solving for the intermediate solutions. Several tailored algorithms try to exploit the almost block diagonal structure of the Kuhn-Tucker matrix and this can lead to much faster computational time than in using the sequential optimization approach. An excellent review of the simultaneous strategies can be found in [58].

To this end, it seems that the recent and future development of sophisticate NLP algorithms may allow larger and larger nonlinear predictive optimization problems to be solved in real time. However, the problem solved at each sample time is a *deterministic* open-loop optimization problem, which does not consider the future uncertainties or feedbacks. Although the receding horizon implementation and the state update scheme attempt to address the uncertainties by incorporating the output mismatch into the state estimate and resolving the optimization problem at every sample time, the solution of this approach can still be highly suboptimal.

In the following subsection, the base case control study with a NMPC controller that uses a full-scale nonlinear model and a conventional simultaneous optimization approach is presented to point out its inherent problem with the computational load. Then, the nonlinear model reduction technique is applied to improve the computational time. Although, the model reduction technique may not drastically reduce the on-line optimization time when using the mathematical programming framework, it is a very important tool to be used with the ADP method in the later section to improve the off-line computational load of the ADP.

5.2.1 Grade Transition by a full-order NMPC

In this section, the centralized NMPC uses the full-order nonlinear model to generate the output prediction, based upon which the optimal input profile for the grade transition is computed. We chose the sequential quadratic programming (SQP) method to solve the optimization for the NMPC in MATLAB^{TM} 7.0 environment. The hessian and gradient information was numerically computed by the algorithm. The optimization was carried out in a Xeon dual processor 2.66 GHz/2.66 GHz and 2.00 GB of RAM. The optimization frequency was chosen to be at every 4 hours, which was larger than the residence time of the distillation column in order to optimize the slow-scale dynamics of the plant. The output of the plant has white measurement noise with maximum magnitude of ±5% of the nominal values of the output. An extended Kalman filter (EKF) was used to update the state from the measurement information, and it had a sample time of 0.02 hr. The objective function was formulated as follows:

$$\begin{aligned} \min_{\Delta\mathbf{u}(k),\ldots,\Delta\mathbf{u}(k+m-1)} \quad & \sum_{i=1}^{p} Q_y\left(\frac{x_{2B}^{sp} - x_{2B}(k+i)}{x_{2B,ss}}\right)^2 \\ & + \sum_{j=0}^{m-1}\left[Q_u\left(\frac{B^{sp} - B(k+j)}{B_{ss}}\right)^2 + \Delta\mathbf{u}(k+j)^T R \Delta\mathbf{u}(k+j)\right] \end{aligned} \tag{55}$$

where $\Delta\mathbf{u} = [\Delta F_0,\ \Delta M_R^{sp},\ \Delta M_D^{sp},\ \Delta M_B^{sp},\ \Delta L,\ \Delta B]^T/100$. The prediction and the control horizons were chosen as $p = 12$ and $m = 3$, respectively. B_{ss} and $x_{2B,ss}$ are the steady-state values of B and x_{2B}, respectively. The weighting factors Q_u, Q_y, and R in Eq. 55 are 10000, 6000, and $20I_{6\times 6}$, respectively.

The result of grade transition from mode 1 to mode 2 via the full-order NMPC is shown in Figures 37 and 38. In this case, the NMPC was able to steer the system to the second operating mode successfully and smoothly. The performance of the controller as measured by Eq. 56 over the 60-hour period is equal to 90.70.

$$\text{performance} = \sum_{k=1}^{15}\left[Q_u\left(\frac{B^{sp} - B(k)}{B_{ss}}\right)^2 + Q_y\left(\frac{x_{2B}^{sp} - x_{2B}(k)}{x_{2B,ss}}\right)^2 + \Delta\mathbf{u}(k-1)^T R \Delta\mathbf{u}(k-1)\right] \tag{56}$$

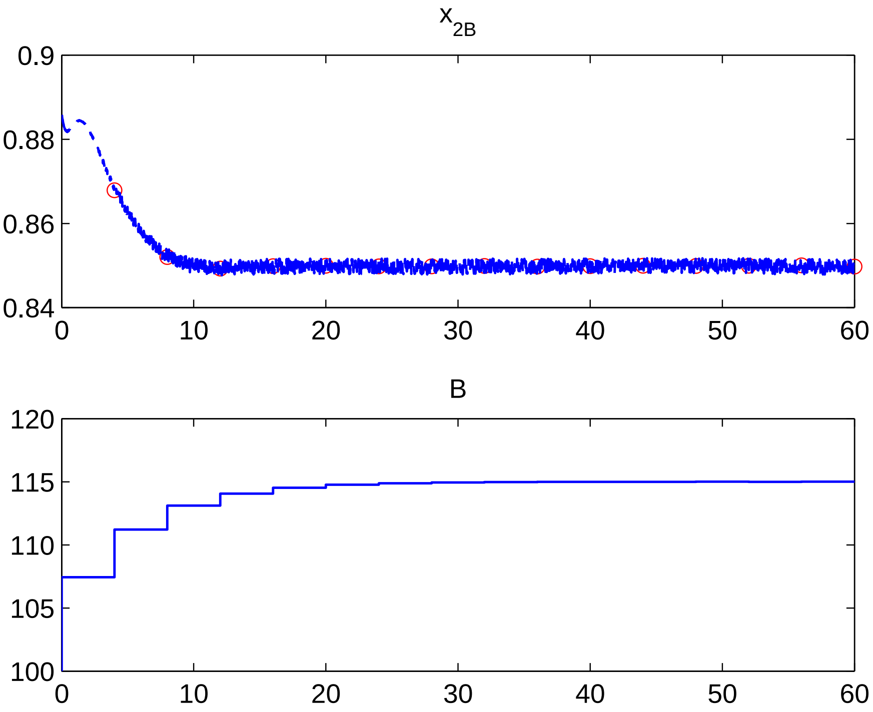


Figure 37: Product variables during the grade transition performed by a NMPC with a full-order nonlinear model (solid: output, o: prediction)

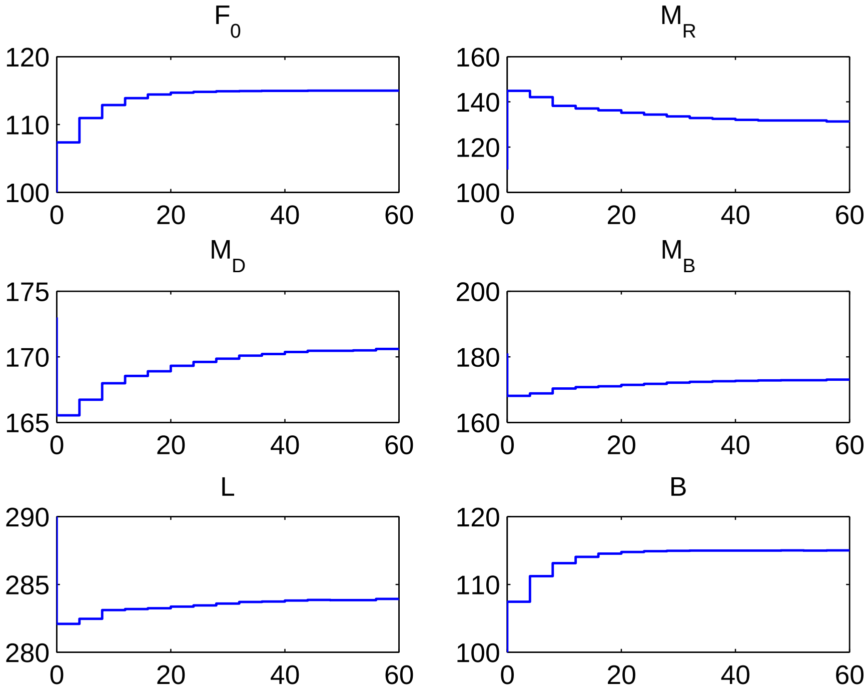


Figure 38: MV profiles during the grade transition performed by a NMPC with a full-order nonlinear model

However, it required several minutes to solve the optimization of this nonlinear system as shown in Table 16. For a problem with a much large size, this method can pose a challenge in on-line computation.

Table 15: Computational time (in minutes) of NMPC schemes using a full-order nonlinear model running at every 4 hours

run number	1	2	3	4	5	6	7	8	9	10	11
	8.6883	3.3734	3.6133	3.4581	3.8854	2.3268	2.2977	2.2482	2.1891	1.9180	2.9276

5.2.2 Grade Transition by a reduced-order NMPC

5.2.2.1 Background

Given the fact that many variables of a process system are highly correlated, it is possible and necessary to derive a significantly lower-order plant-wide model by using an appropriate method available in the literature. Most importantly, the reduced-order model should provide an accurate estimate of the slow-scale dynamics of the real plant. For a linear system, there are several well-proven model reduction approaches. Among these techniques, the optimal Hankel norm approximation [59, 60] and the balanced truncation [61] methods are the most frequently used. The former method looks at the singular value of the Hankel matrix (a product of the controllability and observability matrices) and attempts to truncate the part that corresponds to small singular values. However, this method suffers from the difficulty in recovering the system matrix from the truncated Hankel matrix. On the other hand, the balanced truncation finds the coordinate changes that transform the observability and controllability gramians into the same diagonal and ordered matrices with the first element corresponding to the most observable and controllable state and so on. Then, less observable and controllable states are removed and the system matrices can be recovered from the reduced-order gramians. A general procedure of this linear balanced truncation approach can be found in Appendix B. As our proposed plant-wide optimization scheme is interested in the slow dynamics of the plant, the model reduction method to be used must ensure that the slow dynamics in the bandwidth of interest by the optimizer are accurately described. Enns [24] proposed the method known as the frequency-weighted balanced truncation, which minimizes the frequency weighted error between the original model and the reduced-order model:

$$\|W_o(G - G_r)W_i\|_\infty \tag{57}$$

where G is the original model, G_r is the reduced model, W_o and W_i are the output and input weighting transfer function matrices, respectively. The weighting matrices are normally diagonal, wherein each diagonal element is a filter of selected bandwidth. By choosing the low-pass filter with a "cutoff" frequency higher than the optimizers' frequency, we can emphasize the state information of the desired frequency band, and the reduction in the model order will not cause large errors in the slow frequency range. Therefore, this method is suitable for constructing the low-frequency, reduced-order approximation for the linear system.

Since real chemical processes are mostly nonlinear, several approaches for nonlinear model reduction have been proposed. Unlike in the linear case, there exists no complete theory for nonlinear model reduction [62]. The earliest approach is to lump the original model parameters into a low-dimensional one, such as the approaches proposed to simplify the kinetic models [63, 64]. In particular, the work by Li et al [64] applied a nonlinear perturbation theory to separate the fast reaction variables from the slow variables and transformed the system into a canonical form by nonlinear transformation. The fast variables are lumped and its analytical solution can be obtained by singular perturbation method. A special case where important products or initial reactant should not be lumped was also discussed in [65]. There are also other simplification techniques that substituting complex thermodynamic equations by simpler models. For steady-state flowsheet simulation, Ganesh and Biegler [66] proposed the use of ideal mixture properties to perform flash calculation instead of a rigorous hydrocarbon thermodynamic model represented by Soave-Redlich-Kwong (SRK) equations. Their method used a rigorous model to compute the initial parameters for the reduced problem, which was solved until convergence. This method could save 20-40% of the computational time, while ensuring the convergence to the true values. Hendriks and van Bergen [67] studied the model reduction method applied to the phase equilibria calculations in the oil reservoir simulation. Their approach was to approximate the interaction coefficients on a basis of eigenvectors and truncate the terms associated with small eigenvalues. The resulting reduced-order model drastically reduced the computational requirement, but the method could not be generalized to arbitrary mixture. Perregaard [68] used the simplified model to provide a cheap computation of the Jacobian matrix, which subsequently reducing the model simulation time for any Newton-like integration method.

Model reduction method based on singular perturbation approach has been used to reduce the nonlinear model of chemical reaction systems. The underlying assumption was that the system consists of both the fast and the slow reactions. The method involved deriving an appropriate change of coordinate to transform the original model into a two-time scale standard form of singular perturbation, and the fast reaction is assumed to be at the quasi-steady state. The work by Breusegem and Bastin [69] considered isothermal reaction systems and used a linear change of coordinate. Kumar et al. [70] derived necessary and sufficient conditions for the existence of a nonlinear coordinate change for the general cases where the system may not have an explicit time-scale separation. However, the analytical procedure to derive the ordinate changes becomes increasingly tedious with the problem complexity. Vora et al. [71] also applied the singular perturbation approach to non-isothermal reaction systems to identify the low-dimensional equilibrium manifold where the slow dynamics of the system evolved.

While these methods may work well to suit their specific tasks, when the problem gets larger, it becomes very complicated or impossible to generalize these methods to other system model. Therefore, in this chapter we consider the proper orthogonal decomposition (POD) approach, which is a semi-empirical technique. It employs a linear projection approach to transform the system onto the new coordinate, where a reduced-order model can be derived. This method does not require analytical derivation procedures, and therefore can be applied to high dimensional nonlinear systems. Another projection method that has been proposed in the literature is the method based on empirical gramians, pioneered by Lall [72, 73] for the continuous-time formulation. The method was further elaborated and posed for the discrete-time system by Hahn et al. [74]. The main difference between this method and the POD is in the obtaining of a projection matrix. We include the procedures of this approach in Appendix B. However, an extensive study by van den Berg [75] that applied the reduced-ordered models via POD and via the empirical gramians to optimization problem has revealed that POD model was computationally more efficient in many cases. In addition, from a practical viewpoint the procedures of POD approach is much simpler.

5.2.2.2 Proper Orthogonal Decomposition Approach

This method is also known as the empirical eigenfunctions or the Karhunen-Loève decomposition approach. Earlier work has applied this approach to study the coherent structures in turbulent flows (see [27] for references therein). During the past recent years, it has been applied in the dynamical system context, especially to derive a low-order dynamical system for controller design. The applications include the distributed parameter system where the system is described by partial differential equations (PDEs) [76, 77], as well as the systems of ODEs [72, 74]. Sun and Hahn [30] also applied this technique to reduce the differential states of the index-1 DAE system and used a black-box identification technique to reduce the algebraic constraints, leading to the reduction of the overall DAE system.

For a nonlinear ODE system of the form:

$$\dot{x} = f(x, u), \qquad x \in \mathbb{R}^n \tag{58}$$

the model reduction technique based on POD has three important steps:

1. Collecting representative simulation data of the dynamical system. Let the data ensemble be denoted by X:

   $$X = \begin{bmatrix} x(t_1) - x_{ss}, & x(t_2) - x_{ss}, & \ldots, & x(t_N) - x_{ss} \end{bmatrix} \in \mathbb{R}^{n \times N} \tag{59}$$

   where n is the dimension of the state, N denotes number of the data set, x_{ss} is a vector of the steady-state values of the system, and t_i denotes the snapshot when data was captured. This step is very crucial for the success of this method because the representative data is required to form the approximate of the lower-dimensional subspace.

2. Identify an empirical eigenfunction basis of the data from the eigenvalue decomposition of the covariance matrix.

   $$XX^T = \Lambda V \tag{60}$$

   where Λ is a diagonal matrix of eigenvalues and V is a matrix of corresponding eigenvectors. We can arrange the element in the eigenvalue matrix in a descending order, i.e.

   $$\Lambda = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix}, \tag{61}$$

   $$V = \begin{bmatrix} v_1, & v_2, & \ldots, & v_n \end{bmatrix} \tag{62}$$

   where $\lambda_1 > \lambda_2 > \ldots > \lambda_n$ are the eigenvalues and v_i are the orthogonal eigenvectors of the system. High correlations in the state variables will result in several eigenvalues with small magnitudes.

3. Perform the Galerkin projection. The idea of this step is to replace the original model by the dynamical system that evolves on its eigenvector subspace. Let $P = [\ v_1, \ v_2, \ \ldots, \ v_r\ ]^T$ and $Q = [\ v_{r+1}, \ v_{r+2}, \ \ldots, \ v_n\ ]^T$, where r is the order of the reduced-order model. Typically, r is chosen so that $\lambda_{r+1}, \ldots, \lambda_n$ are very small. Once the order is chosen, the original nonlinear system can be transformed to:

   $$\dot{\bar{x}}_1(t) = Pf\left([P^T, Q^T]\bar{x} + x_{ss}, u(t)\right) \tag{63}$$

   $$\dot{\bar{x}}_2(t) = Qf\left([P^T, Q^T]\bar{x} + x_{ss}, u(t)\right) \tag{64}$$

   where

   $$\bar{x} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix} = \begin{bmatrix} P \\ Q \end{bmatrix} (x - x_{ss}) \tag{65}$$

In this step, the nonlinear dynamic states $\bar{x}_1$, which are the projection onto $[\ v_1, \ \ldots, \ v_r\ ]$ eigenvector directions are the states that encounter much larger magnitudes of change than those projected onto the $[\ v_{r+1}, \ \ldots, \ v_n\ ]$ eigenvector directions. Based on this consideration, the reduced-order dynamic model can be derived by two approaches.

Residualization: Intuitively, the eigenvectors with small eigenvalues correspond to the directions of state changes that are not prominent. These are mostly the high-frequency dynamic changes that occur and disappear quickly. In residualization method, the projection of these modes is assumed to be at the quasi-steady state and the approximate system is in a DAE form.

$$\begin{aligned} \dot{\bar{x}}_1(t) &= Pf\left([P^T, Q^T]\bar{x} + x_{ss}, u\right) \\ 0 &= Qf\left([P^T, Q^T]\bar{x} + x_{ss}, u\right) \end{aligned} \tag{66}$$

So in this case, the original state vector is approximated as:

$$x = [P^T, Q^T]\bar{x} + x_{ss} \tag{67}$$

To integrate the system in Eq. 66, the initial values from the original subspace $x(0)$ can be transformed as follows:

$$\bar{x}(0) = \begin{bmatrix} P \\ Q \end{bmatrix} (x(0) - x_{ss}) \tag{68}$$

Truncation: On the other hand, model reduction via truncation assumes that the states with small eigenvalues do not change at all. Therefore, only the states $\bar{x}_1$ is solved in the reduced-order ODE system of the form:

$$\dot{\bar{x}}_1(t) = Pf\left(P^T\bar{x}_1 + x_{ss}, u\right) \tag{69}$$

$$x = P^T\bar{x}_1 + x_{ss} \tag{70}$$

With a reduced-order system from either (66) or (69), the on-line computational time in solving the model equation can be reduced.

5.2.2.3 *Simulation Results*

To identify the underlying empirical eigenvector basis, the inputs to the plant were excited in the range of ±10% around their nominal operating points. Eigenvalue decomposition of the simulation data was performed and the eigenvalue plot is shown in Figure 39. This suggested that only the first 7 eigenvectors should be considered as the basis in the projection matrix P of Eq. 63, since the eigenvalues become very close to zero after 7 modes. Hence, the reduced-order model via residualization method contains 7 ODEs and 32 algebraic equations, while the truncation method considers only the 7 ODEs.

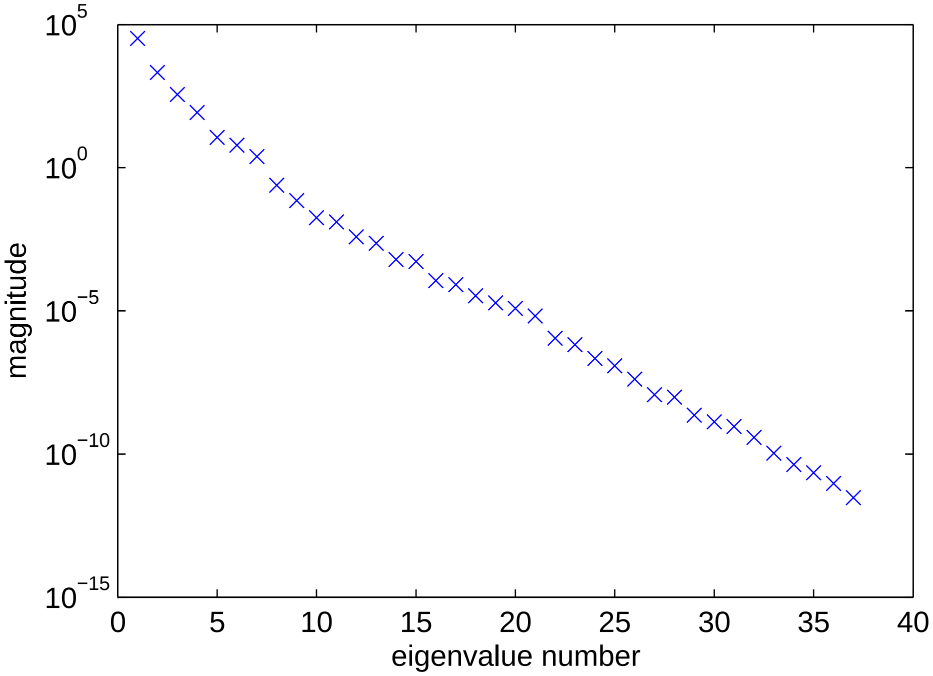


Figure 39: Eigenvalue plots of the data collected from dynamic simulations of the reactor-distillation-recycle system

Grade transition results from the reduced-order NMPC based on residualization are shown in Figures 40 and 41. This controller can also steer the system to the mode 2 successfully. The performance over the 60-hour period is 91.73. On the other hand, the NMPC based on the truncation method shows worst performance of 96.02. The output and input trajectories from this approach are shown in Figures 42 and 43, where there is an offset between the model prediction and the actual output concentration. This is because

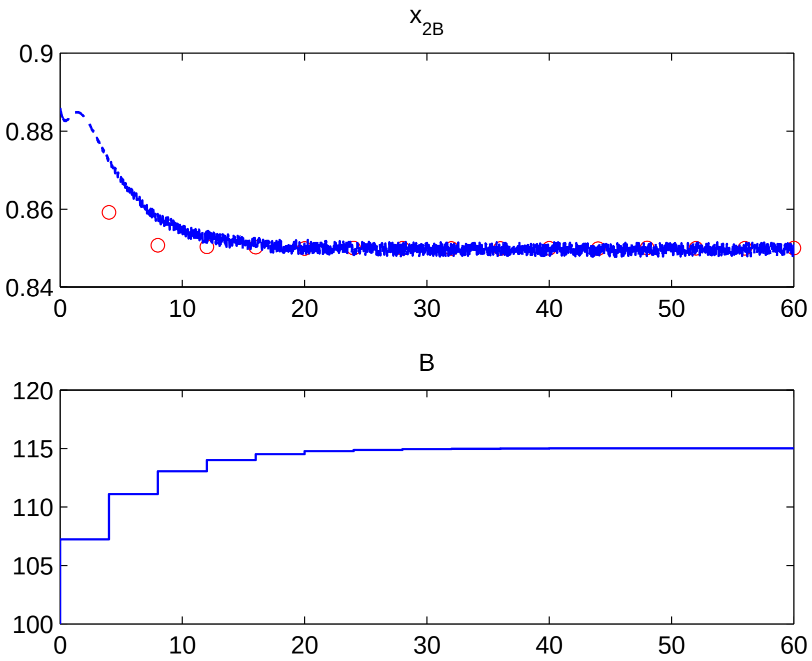


Figure 40: Product variables during the grade transition performed by a NMPC derived from a reduced-order model with residualization (solid: output, o: prediction)

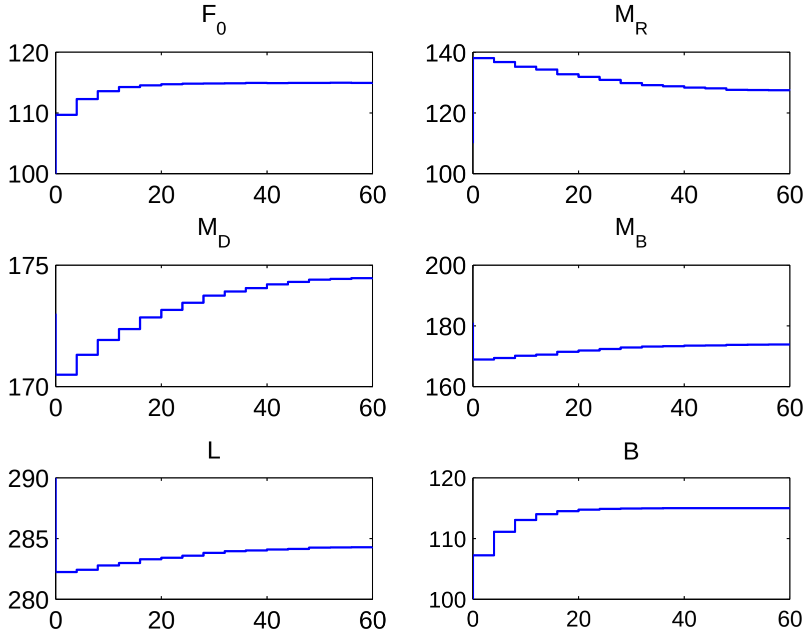


Figure 41: MV profiles during the grade transition performed by a NMPC derived from a reduced-order model with residualization

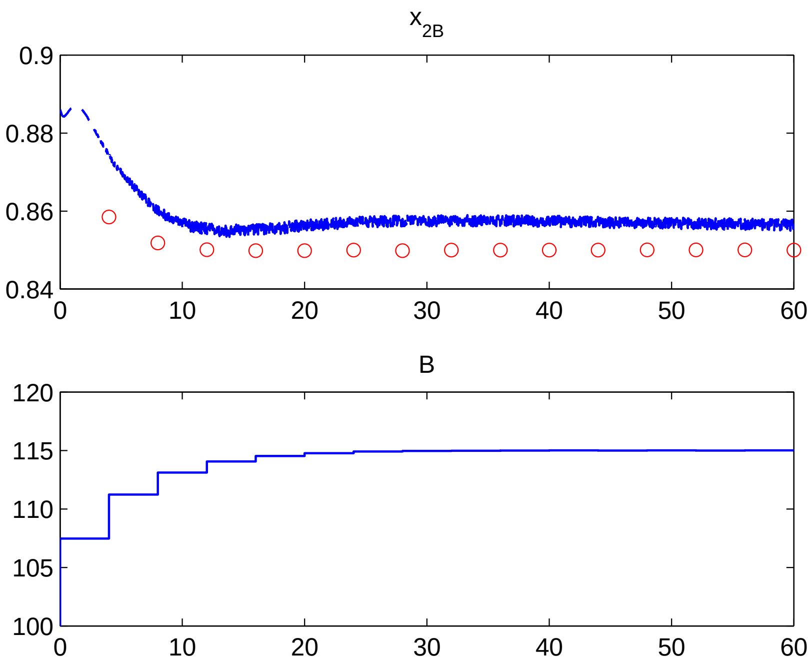


Figure 42: Product variables during the grade transition performed by a NMPC derived from a reduced-order model with truncation (solid: output, o: prediction)

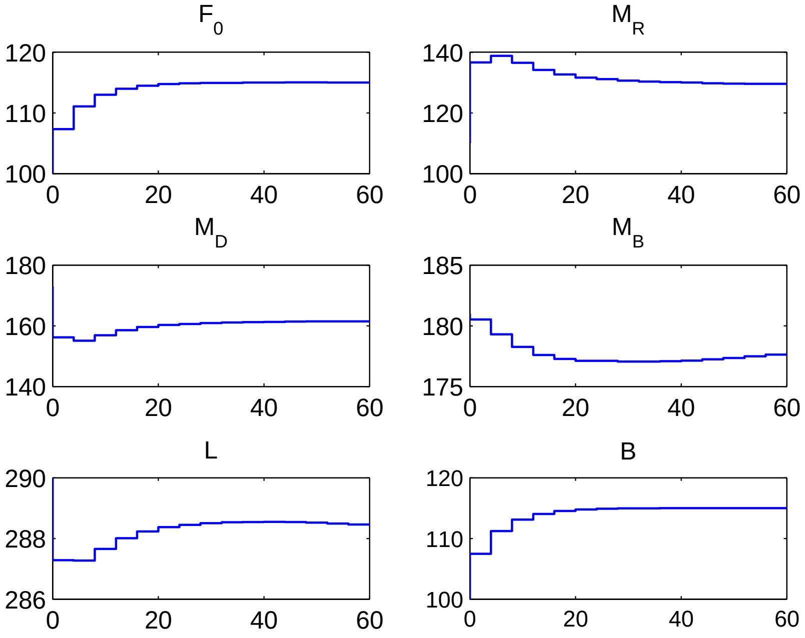


Figure 43: MV profiles during the grade transition performed by a NMPC derived from a reduced-order model with truncation

the changes of the fast modes occur and reach the new steady states so quickly. Without taking into account these changes, the truncation approach results in the approximation error. On the other hand, the residualized model takes into account the influence of the new steady states of the fast modes. Therefore, the projection of the transformed states back to the original state space in Eq. 67 is more accurate than the one in Eq. 70 of the truncation method. In terms of computational time comparison, the reduced-order NMPCs require almost half the time of the full-order NMPC as shown in Table 16. This method, if coupled with even more powerful optimization algorithms, will be much more efficient for solving an on-line optimization problem.

Table 16: Computational time (in minutes) of NMPC schemes running at every 4 hours

run number	NMPC with full order	NMPC with POD/ residualization	NMPC with POD/ truncation
1	8.6883	5.8588	5.9911
2	3.3734	1.9979	2.0766
3	3.6133	1.6026	1.7638
4	3.4581	1.5703	1.8888
5	3.8854	1.8060	1.5893
6	2.3268	1.3018	1.8138
7	2.2977	1.5320	1.5940
8	2.2482	1.5677	1.5466
9	2.1891	1.5573	1.5086
10	1.9180	1.3203	0.7323
11	2.9276	0.8359	0.7198

5.3 Optimal Control via Approximate Dynamic Programming (ADP)

In this section, an Approximate Dynamic Programming (ADP) method is applied to the case study. ADP has been shown to be useful in applying optimal control to process control applications. One of the outstanding advantages of this approach is the significantly lowered real-time computational requirement because one solves an equivalent single-stage optimization of the original multi-stage optimization problem. In addition, stochastic uncertainty and feedback information can be taken into account in a general and computationally amenable manner, unlike in the conventional mathematical programming framework. Nonetheless, the existing framework can require excessive off-line computational time to obtain the cost-to-go table when applied to problems with large state and action spaces typical of an integrated plant optimization. This section provides useful insights into the issues and shows how to apply systematic approaches to reduce the off-line computational load so that the ADP framework can be applied to optimization of high-dimensional integrated plants efficiently.

5.3.1 Introduction

A dynamic programming (DP) allows us to find optimal control strategies under stochastic uncertainties in complex multi-stage decision making problems. The objective of this framework is to compute the optimal 'cost-to-go' function, which parameterizes the control solution with respect to the state. With the known 'cost-to-go ' function, the on-line multi-stage optimization can be reduced to an equivalent single-stage optimization problem, and hence the real-time computation can be drastically simplified. Nevertheless, the original algorithms based on solving the Bellman's equation suffer from an exponential growth in computation with the size of the problem. This problem is known as the *curse of dimensionality*, which makes this framework impractical for most of the real world problems.

During the past two decades, a wealth of research in various disciplines such as machine learning, operations research, and artificial intelligence, has focused on finding approximate solution to the Bellman's equation so that the framework can be applied to practical problems. This framework is known as Approximate Dynamic Programming (ADP). Many real world problems, such as power system control [78], helicopter flight control [79], large-scale job shop scheduling [80], financial problems [81], etc. have been solved within this framework. There are many different ADP strategies developed and adopted by different disciplines, and they are all based on the use of value function approximation, with the

Taylor series, the lookup tables, the multi-layer perceptrons, etc. There is no consensus on which method is the best, because some are better in some applications, while others are better suited in other application domains [82].

In process control community, applications of ADP are still in an early stage. This is perhaps because the nature of process control problems differs from those in the robot learning and operations research problems. Outstanding features of process control problems are continuous state and action spaces, small operating space relative to the size of the entire state space, and very costly on-line experiment [14]. Selection of value function approximator is critical to ensuring the accuracy of the value function and the robustness of the ADP-based controller. Recently, Lee and coworkers have developed an ADP algorithm for process control problems based on the use of lookup table and k-nearest neighbor function approximator. This method overcomes the curse of dimensionity in the state space and has been applied to several complex nonlinear processes, such as a 2x2 bioreactor control [83], and a polymerization reactor control problem [15] with 8 state and 2 manipulated variables. Since the action space was small in these problems, the Bellman's equation could be solved by evaluating every discretized action without encountering excessive computational issue.

5.3.2 Mathematical Preliminaries

5.3.2.1 Dynamic Programming Framework

A multi-stage optimization problem to find an action sequence that minimizes a certain objective function over a future time horizon may be represented as follows:

$$\min_{u(0),\dots,u(p-1)} \sum_{i=0}^{p-1} \phi(x(i), u(i)) + \phi_t(x(p)) \tag{71}$$

subject to

$$\text{Path constraint:} \qquad g(x(i), u(i)) \geq 0, \qquad i = 0, 1, \dots, p \tag{72}$$

$$\text{Model constraint:} \quad x(i+1) = F_h(x(i), u(i)) \equiv \int_{i\Delta t}^{(i+1)\Delta t} f(x(\tau), u(\tau)) d\tau \tag{73}$$

where $x(i)$ represents a state vector at the i^{th} sample time, u is a vector of manipulated variables, and Δt is the sampling interval. The stage-wise cost function and the terminal cost function are denoted by ϕ, and ϕ_t, respectively. The path and model constraints may be nonlinear functions. A continuous process model $\dot{x} = f(x, u)$ is assumed to be available to be integrated over the sample time Δt with a piece-wise constant input $u(\tau) = u(i)$ for $i \cdot \Delta t \leq \tau < (i+1) \cdot \Delta t$.

Dynamic Programming (DP) framework is based upon the *Bellman's optimality principle*, which states that if a control policy μ^* is optimal, then no matter how the intermediate state $x(k)$ is reached the rest of the trajectory under the policy μ^* will be optimal for the remaining subproblem:

$$\min_{u(k),\ldots,u(p-1)} \sum_{i=k}^{p-1} \phi(x(i), u(i)) + \phi_t(x(p)) \tag{74}$$

The term "policy" (μ) here represents a rule that maps the state to the control action, i.e. $u(i) = \mu(x(i))$. The theoretical proof of the Bellman's optimality principle can be found in various DP textbooks such as [84, 85]. This principle is used to derive the optimal control policy. In this approach, we define the 'cost-to-go' of a starting state x under the control policy μ, denoted by $J^\mu(x)$, as the sum of a stage-wise cost incurred from the state x until the end of horizon, i.e.

$$J^\mu(x) = \sum_{i=0}^{p-1} \phi\left(x(i), \mu(x(i))\right) + \phi_t(x(p)), \qquad x(0) = x \tag{75}$$

$J^\mu(x)$ is to be defined over S, which is a set of all possible state points. The goal of DP is to derive the optimal cost-to-go function, J^*, which is the minimum cost-to-go function under the optimal policy. That is, $J^* = \inf_\mu J^\mu$.

From Bellman's optimality principle, it follows that the optimal cost-to-go function satisfies the following *Bellman's optimality equation* [86].

$$J^*(x(i)) = \min_{u \in U} \left[\phi(x(i), u(i)) + J^*(x(i+1))\right], \qquad \forall x \in S \tag{76}$$

where U is a set of all possible actions. The objective of the off-line cost-to-go learning is to arrive at the optimal cost-to-go function. Once J^* value for every state point is known, the solution to an on-line optimization of Eq. 71 can be obtained by solving an equivalent single-stage optimization problem shown in Eq. 77. As a result, the on-line computational load can be drastically reduced.

$$\mu^*(x(i)) = \arg \min_{u(i) \in U} \left[\phi(x(i), u(i)) + J^*(F_h(x(i), u(i))) \right] \tag{77}$$

5.3.2.2 Stochastic System

So far, we explain the deterministic formulation of DP. For stochastic system, the optimization problem can be represented as follows:

$$\min_{u(i),\ldots,u(p-1)} \mathbb{E} \left[\sum_{i=0}^{p-1} \phi(x(i), u(i)) + \phi_t(x(p)) \right] \tag{78}$$

$$u(i) = \mu(\mathcal{I}(i)) \tag{79}$$

where $\mathbb{E}$ is an expectation operator. Note that the action is no longer evaluated in a deterministic manner. Instead, it is evaluated from the available stochastic information denoted by $\mathcal{I}$. This information vector typically includes the conditional probability distribution of the state vector. Then the Bellman equation is defined as the following.

$$J^*(\mathcal{I}(i)) = \min_{u(i)} \mathbb{E} \left[\phi(x(i), u(i)) + \alpha J^*(\mathcal{I}(i+1), u(i)) | \mathcal{I}(i) \right] \tag{80}$$

where $\alpha \in [0, 1)$ is a discount factor that signifies the importance of the immediate cost compared to the future cost. It is assumed that the stochastic equation governing the one-time step transition of $\mathcal{I}$ is available.

5.3.2.3 Conventional DP Algorithms

Because the solution to the Bellman equation can seldom be obtained in a closed-form, value iteration or policy iteration algorithms are commonly employed to obtain optimal cost-to-go function numerically. For a discrete finite state space problem, these algorithms can be described as follows.

Value Iteration:

Value iteration starts with an initial estimate of the cost-to-go for each state, and then iterates the Bellman equation for every state point until convergence. The procedure of this approach is as follows:

1. Initialize the cost-to-go function $J^0(x)$ for all state points $x \in S$

2. For each state $x \in S$, evaluate

$$J^{i+1}(x) = \min_{u \in U} \mathbb{E}\left[\phi(x,u) + \alpha J^{i}(x')\right] \tag{81}$$

   where the superscript i here denotes the iteration index, and x' represents the successive state.

3. Repeat the iteration step 2 until convergence

In this framework, every state point has to be updated once in each iteration.

Policy Iteration:

Policy iteration iterates between two steps: a policy evaluation and a policy improvement. The policy evaluation step starts with a certain policy and computes the cost-to-go values under that policy. Then the improvement over the previous policy is done in the policy improvement step. These two steps are repeated until the policy no longer changes. These procedures are summarized as follows:

Policy Evaluation:

1. Given a policy μ, initialize $J^{\mu}(x), \forall x \in S$.

2. For each state x, update the cost-to-go value according to

$$J^{\mu,j+1}(x) = \mathbb{E}\left[\phi(x,\mu(x)) + \alpha J^{\mu,j}(x')\right] \tag{82}$$

   where the superscript j is the iteration index of the policy evaluation step

3. Repeat the iteration step 2 until J^{μ} converges.

Policy Improvement:

Given a policy μ^{i} and the corresponding cost-to-go function $J^{\mu^{i}}$, the next improved policy is given by

$$\mu^{i+1}(x) = \arg\min_{u} \mathbb{E}\left[\phi(x,u) + \alpha J^{\mu^{i}}(x')\right] \tag{83}$$

Value iteration converges monotonically to the optimal cost-to-go function [87]. Whereas the policy iteration typically requires lesser iterations than value iteration, each iteration may take very long time as finding the cost-to-go function itself involves iterations [88].

Both value iteration and policy iterations perform full update of every state point in the stored state set in a sequential manner. That is, the cost-to-go update of a state is performed once in each iteration and not repeated until every state is updated. This can lead to a prohibitively large off-line computational load. Although there are other algorithms to improve the cost-to-go learning rate without updating every state in such a strict order in every iteration, we argue that they are unlikely to improve the off-line computational time when applied to the process control problem of an integrated plant. The well-known methods include the *Asynchoronous* value iteration algorithm as in [89, 90, 91], and a family of so-called *General Prioritized Solvers (GPS)* [92]. The former considers the situation whereby several processors participate simultaneously in the computation while maintaining coordination by information exchange to the neighboring nodes via communication links. In this method some states may be updated several times, while others are only updated once. Lee et al. [93] argued that many problems require large numbers of update to attain the optimal cost-to-go function. The GPS methods include three principal enhancements to accelerate the value and policy iterations: partitioning, prioritization, and variable reordering. Partitioning method groups states that are interdependent together into sets, and updates the cost-to-go of the whole set before moving on to another set. Prioritization and reordering methods attempt to order the state updates in a more effective way, such that after the value function of a state point x is updated, every states dependent on x will be updated. The case studies in [94, 95, 92] showed that these techniques are useful in the problems where states are not highly dependent, such as in an acyclic markov chain. That is because states that are not dependent on many other states are not required to update in every iteration. On the other hand, in the problems where all states are highly interdependent, these enhancement techniques may require more computational overhead to update the priority queue when most states have to be updated in each iteration anyway.

5.3.2.4 Approximate Dynamic Programming

DP is often considered impractical for solving medium- or large-scale problem because its off-line computational load is proportional to the cardinalities of the state space, the action space, and the uncertainty space, which can grow exponentially with their dimensions. This is the ‘curse-of-dimensionality’ problem. As a result, an approximate dynamic programming (ADP) framework has been widely studied in the literature in order to approximate the optimal cost-to-go function of the state. A comprehensive review of ADP research and applications can be found in [96] and [93]. Among these works, only few can be applied to process control problems in which the state space and the action space are continuous and can have a very large number of discretized values. In addition, the training data is limited due to the fact that the processes are normally controlled in small operating regions compared to the size of entire state space. Earlier approach applied to continuous state space problems relied upon the use of global approximators, such as artificial neural networks in the Neuro-dynamic programming (NDP) framework [88], to fit simulation data to the cost-to-go function. When this technique is applied to process control problem with sparse data, it can often cause divergence in the iterative off-line learning due to overfitting as demonstrated in [15]. To overcome such problems, Lee and coworkers [15] combined the lookup-table approach with the use of a local approximator with nonexpansion property, particularly a k-nearest neighbor approximator, as well as a quadratic penalty term to guard against cost-to-go extrapolation in the region not covered by the training data. Their suggested ADP framework with off-line value iteration scheme is depicted in Figure 44. It is important to note that at the stage of their framework development, the Bellman equation was solved by enumerating all possible discretized actions in the entire action space because the problems of interest had small action space of 1 or 2 manipulated variables.

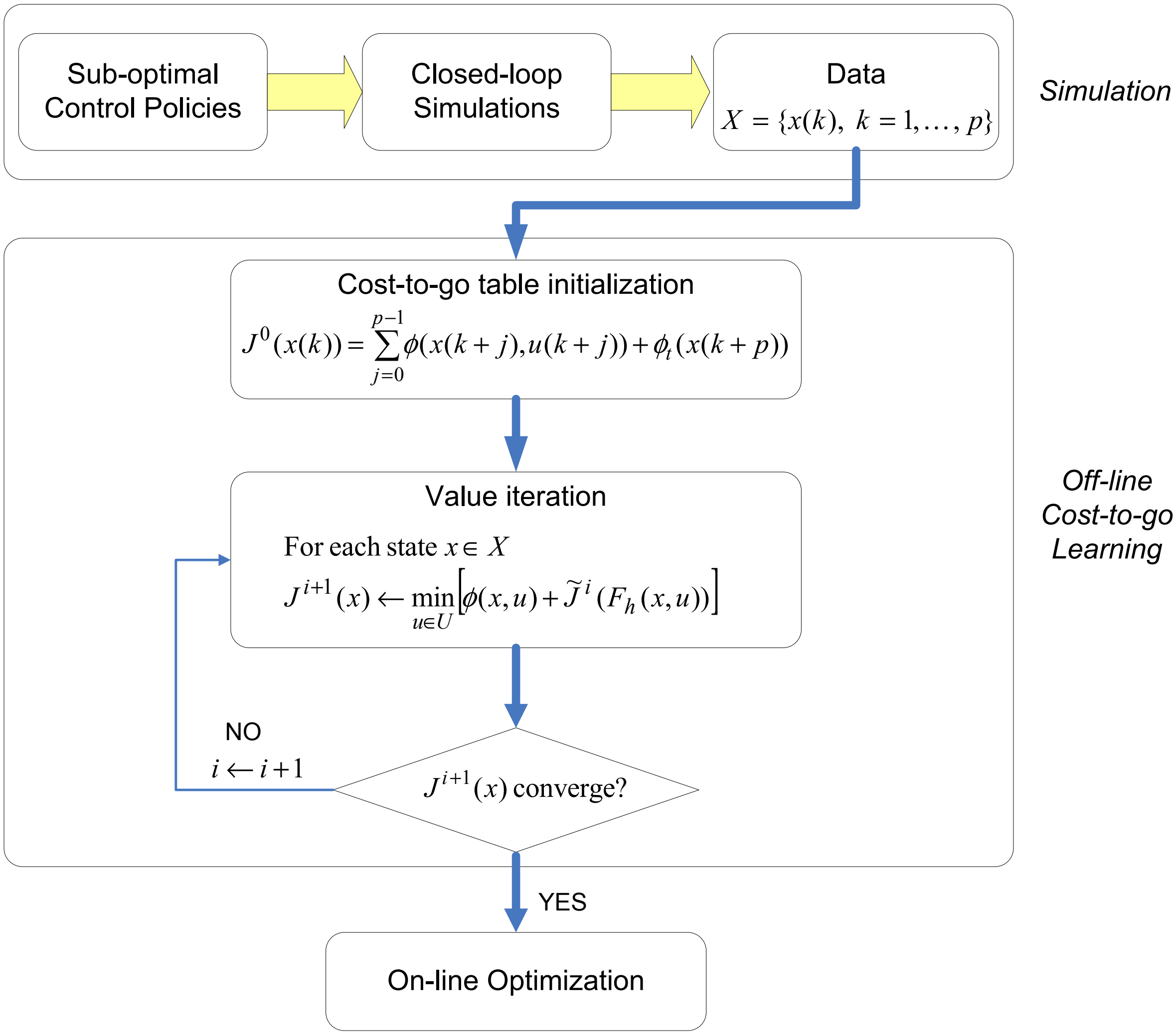

Sub-optimal Control Policies
Closed-loop Simulations
Data
$X = \{x(k),\ k = 1, \ldots, p\}$
Simulation
Cost-to-go table initialization
$J^0(x(k)) = \sum_{j=0}^{p-1} \phi(x(k+j), u(k+j)) + \phi_t(x(k+p))$
Value iteration
For each state $x \in X$
$J^{i+1}(x) \leftarrow \min_{u \in U} \left[\phi(x,u) + \tilde{J}^i(F_h(x,u)) \right]$
Off-line Cost-to-go Learning
NO
$i \leftarrow i+1$
$J^{i+1}(x)$ converge?
YES
On-line Optimization

Figure 44: Algorithmic framework of Approximate Dynamic Programming

Because the optimization problem of an integrated plant encompasses continuous variables and sparse operating data similar to the optimal control problems solved by Lee and coworkers, we adopt their approach as a precursor to our work. The key idea of this framework is not to solve for the optimal cost-to-go function for the entire state space, but to use the closed-loop simulations with suboptimal control policies to generate state trajectories that cover the state space, denoted by X, relevant to the optimal control problem. The premise is that the optimally controlled trajectories are confined to very small region of the entire state space. Therefore, the search for optimal policy could be limited to this relevant state space. However, this relevant space is not known a priori, so it is approximated by simulating a number of suboptimal policies. Once the simulation data is available, the cost-to-go lookup table can be initialized by summing the cost starting from each state point until the end of horizon. Then, this lookup table is iteratively improved in the off-line value iteration step.

5.3.2.5 Cost-to-go function approximation

In the algorithm proposed by Lee et al. [15], the cost-to-go update of each state point is done by evaluating every discretized action and computing the corresponding immediate cost and the cost-to-go of the successive state, as shown in Eq. 84 and Figure 44.

$$J^{i+1}(x) \quad \leftarrow \quad \min_{u \in U} \phi(x,u) + \tilde{J}^i(F_h(x,u)) \tag{84}$$

Since the subsequent state $F_h(x,u)$ may not be pre-stored in the lookup table, the cost-to-go approximation scheme has to be used. As mentioned previously, the operating space of process control problem can be much smaller than the size of entire state space, and hence this approximation step has to be done cautiously. The approximation function should interpolate the cost-to-go value within the region covered by the simulation data and avoid extrapolation into an unexplored region to avoid the risk of introducing an unsafe operation. This is the main reason why the popular Neuro-dynamic programming (NDP) framework, which utilizes Artificial Neural Networks (ANN), as function approximators may not be applied to process control problems, which often have sparse training data sets. Therefore, the use of k-nearest neighbor function approximator for cost-to-go approximation and the parzen density estimation to guard against extrapolation as described in [15] are more suitable for process control problems. There are two steps in computing the cost-to-go estimate:

1. Local cost-to-go approximation: The k-nearest neighbor approach approximates the cost-to-go of the query point $x_0 = F_h(x, u)$ as a weighted average of the cost-to-go values of its neighbors. The first step in the procedure is to compute the Euclidean distance between the query point x_0 and every state point x_i in X, where X is a set of the stored state points as follow:

   $$d_i = \sqrt{(x_0 - x_i)^T W (x_0 - x_i)} \tag{85}$$

   where W is a user-defined diagonal matrix assigning weights to different state variables. Then the distance is sorted and only the k closest points are considered for the cost-to-go approximation:

   $$\tilde{J}(x_0) = \sum_{x_i \in N_k(x_0)} w_i J(x_i) \tag{86}$$

   $$w_i = \frac{1/d_i}{\sum_{i=1}^{k} (1/d_i)} \tag{87}$$

   where $N_k(x_0)$ is a set of the k-nearest neighbors of x_0 and w_i is a distance weighting parameter.

2. Local data density estimation: To discourage excessive extrapolation into the region faraway from the training data set, Parzen data density estimator is employed. The Parzen density estimate of a query point x_0 in the training data set X is defined as

   $$f_X(x_0) = \frac{1}{N\sigma^n} \sum_{i=1}^{N} K\left(\frac{x_0 - x_i}{\sigma}\right) \tag{88}$$

where n is the state dimension, x_i represents the stored state point, N is the number of stored state points in X, σ is a parameter defining confidence bandwidth, and K is a multivariate Gaussian Kernel function defined as:

$$K\left(\frac{x_0 - x_i}{\sigma}\right) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{\|x_0 - x_i\|_2^2}{2\sigma^2}\right) \tag{89}$$

Hence, the parzen density function is based on the Euclidean distance between the query data point x_0 and every point in the stored data set. Kernel function value is high when the data point x_i is close to the query point, and low when they are far apart. Finally, if the Parzen density estimate of the query point is too small, a quadratic penalty term is added to the cost-to-go estimate as follows:

$$J_{bias}(x_0) = \begin{cases} J_{max}\left[\frac{(f_X(x_0))^{-1}-\rho}{\rho}\right]^2 & \text{if } (f_X(x_0))^{-1} > \rho \\ 0 & \text{otherwise} \end{cases} \quad (90)$$

$$\tilde{J}(x_0) \leftarrow \tilde{J}(x_0) + J_{bias}(x_0) \quad (91)$$

where parameters J_{max} is the upper bound on the cost-to-go value, and ρ is the threshold value. A guideline on the design of these parameters can be found in [15].

5.3.3 Computational Issues of the Existing ADP Method

According to the procedure of the existing ADP approach described in the previous section, total off-line value iteration time of the ADP approach will be proportional to (i) the number of discretized actions in the action space, (ii) the time required to approximate the cost-to-go value of the successive state resulting from each action in Eq. 84, (iii) the number of stored state points N, and (iv) the number of iterations needed for the cost-to-go function to converge. This paper is concerned with reducing the computational time arising from the first three factors. The number of iterations required typically depends upon the effectiveness of the initial control policy used in the simulations. The closer they are to the optimal policy, the less number of iterations is required.

5.3.3.1 Curse of dimensionality in the action space

In this case study, the action space has 6 dimensions, including the feed (F_0), holdup setpoints for the reactor (M_R^{sp}), the condenser (M_D^{sp}), and the reboiler (M_B^{sp}), the reflux flow rate (L), and the production rate (B). The operating range of these input variables is shown in Table 13. Clearly, there needs to be many discretized values for each input variable in order to ensure the solution quality of the ADP-based controller. Unfortunately, this will lead to the curse of dimensionality in the action space. Even if we use coarse discretization grid size, there will still be too many action combinations to evaluate in order to update the cost-to-go of each state point. For example, if the grid size is 5 for each input, there will be as many as $16 \times 24 \times 24 \times 24 \times 58 \times 16 \approx 205 \times 10^6$ action combinations!

5.3.3.2 *Computational load to approximate $\tilde{J}(F_h)$ per action*

As explained in the previous section, approximating $\tilde{J}(F_h(x, u))$ for each action u involves the model integration and the computation of Eqs. 85-91. We perform the value iteration on a Xeon^{TM} dual processor with 2.66 GHz/2.66 GHz and 2.00 GB of RAM and using Matlab 7 software. The average time to compute the approximate of the cost-to-go in Eq. 84 per state per action was 0.664 seconds. Suppose that there are 1000 stored data points. If we assume optimistically that the optimal solution to Eq. 84 can be found within some heuristic trials of 1000 actions, then the time required to perform 1 value iteration would be ≈ 0.664 s. $\times 1000 \times 1000 = 7.7$ days! If the control policies used in close-loop simulation were far from optimal, number of iterations for the cost-to-go to converge could be large. This means that it would take several months to have the converged cost-to-go function for on-line optimization!

5.3.4 Proposed Methodologies to Apply ADP to Optimal Control of an Integrated Plant

In this section, we propose a method to overcome the curse-of-dimensionality in the action space. In addition, for problems with high-dimensional state space, the feature weighting matrix in the k-nearest neighbor cost-to-go approximating step has to be carefully chosen so that the approximation is accurate. Furthermore, removing state dimensions irrelevant to the cost-to-go values from the computation of Eq. 85 will save additional computational overhead. Finally, because the cost-to-go update requires a simulation to generate the state transition, the reduction in model integration time plays a crucial role in reducing the overall off-line computational time.

5.3.4.1 *Gradient Search Approach for Finding Optimal Action*

As mentioned in the previous section, applying a poorly directed search strategy for optimal action to high dimensional problems can lead to extremely large computational time. Our goal is to systematically direct the search in a way that allows quick convergence to locally optimal actions. This is because in large real-time control problems, convergence to globally optimal policies is not always possible. The approach suggested here is to apply the gradient-based search approach for finding an optimal action. Secondly, we further reduce the computational load by restricting the search regions for the optimal action making it possible to apply to high-dimensional problems.

Because the dynamical system and the cost-to-go function studied in this work are nonlinear, an efficient nonlinear optimization algorithm has to be used. This can be a general-purpose NLP solver, such as the Sequential Quadratic Programming (SQP) solver. In addition, a more sophisticate simultaneous optimization solver can be applied, although it is not clear whether it will give an advantage over the sequential optimization approach. This is because in this case only a one-time step integration of the model equation is required, which is very simple to perform in a sequential optimization setting. On the other hand, using a simultaneous approach will render a larger optimization problem due to discretization.

In this case the cost-to-go update for each state point $x \in X$ is the solution of a constrained optimization of the form:

$$\begin{aligned} J^{i+1}(x) \quad &= \min \phi(x,u) + \tilde{J}^i(F_h(x,u)) \\ \text{s.t.} \qquad &x_{LB} \leq F_h(x,u) \leq x_{UB} \\ &u_{LB} \leq u \leq u_{UB} \end{aligned} \tag{92}$$

where x_{LB} and x_{UB} are the lower and the upper bounds on the state variables, respectively. Oftentimes, a nonlinear constrained problem can be solved in fewer iterations than an unconstrained problem using SQP. This is because the optimizer can use the information of the feasible area to make decisions regarding directions of search and step length. As a result, it is important to specify appropriate lower and upper limits, u_{LB} and u_{UB}, and good initial guess that allow the optimizer to quickly converge to a locally optimal solution.

Local Action Set Heuristics

Our proposed heuristic is to construct for each state point $x \in X$ a local action set denoted by A^x. This local action set is initialized by storing good actions that have been applied to the state point x and the nearest neighbors of x during the off-line simulation. Then, the cost-to-go update of each state point x is done by solving the optimization problem (Eq. 92) with the input feasible region be the polyhedron spanned by the local action set A^x. That is, u_{LB} and u_{UB} are constructed from the minimum and maximum magnitudes in each input dimension of A^x. By including the information of past local actions, we can assure that the best of the search covers the regions of the action space where good starting control policies used during the close-loop simulations have been found. The idea of the local action set can be visualized as shown in Figure 45.

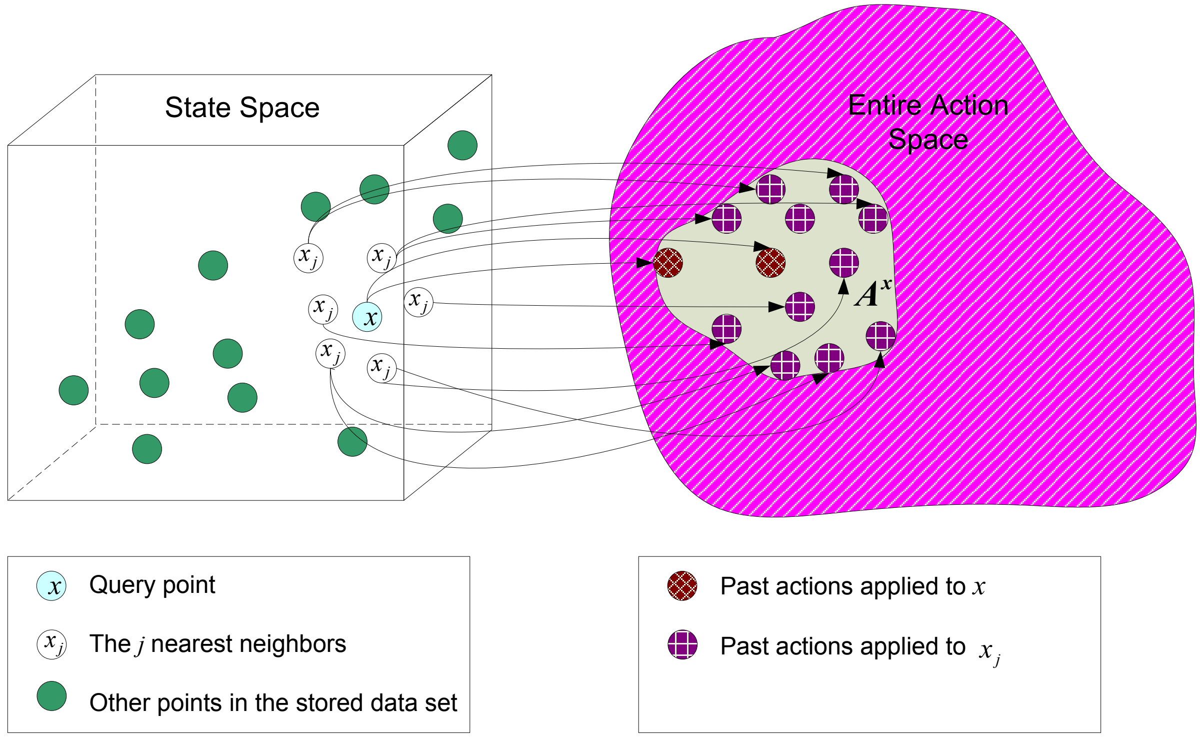

State Space
Entire Action Space
x_j
x
A^x
Query point
The j nearest neighbors
Other points in the stored data set
Past actions applied to x
Past actions applied to x_j

Figure 45: Local action set heuristic

Initial Guess

The lookup table of the local action set can be arranged in a way that the local actions are ordered from the best to the worst, measured in terms of the corresponding cost-to-go values of the resulting next state. During the cost-to-go update of each state x, the initial guess for the optimization Eq. 92 is the first action from the local action lookup table. Then after the optimal solution to Eq. 92 is found, the optimal action is stored as the first action on the local action set and can be used as initial guess for the optimization in the next iteration. The modified ADP algorithm that incorporates this local gradient search approach is summarized in Table 17.

5.3.4.2 Data Analysis and Dimensionality Reduction in Computing k-nearest Neighbor

As described in the previous section, this ADP algorithm requires the computation of distance between the query point and every stored state point. Oftentimes for processes with large state dimensions, not every state variable is relevant for determining the cost-to-go value of the state. Therefore, those state dimensions that do not affect the cost-to-go value can be taken out from the Euclidean distance computation (d_i) to save the computational overhead and to improve the prediction accuracy of the k-nearest neighbor cost-to-go approximation step. Nonetheless, the naive approach that keeps only the variables that show up explicitly in the stage-wise cost function may leave out other state variables that are strongly correlated with the cost-to-go. Therefore, we suggest the use of a systematic *variable selection* or *feature selection* technique to select a subset of state variables from the original state vector and use them in the k-nearest neighbor cost-to-go value approximation. There are several techniques in the literatures that attempt to capture the correlations between the predictors and the regressors; they include linear projection techniques like the Partial Least Square (PLS) [97] method, or nonlinear methods like Kernel-based regression. Choice of methods is problem-dependent and users should cross-validate the chosen method to make sure the reduced-dimensional state vector still captures the relationship between the state and the cost-to-go value adequately. In addition, correlation coefficients between the state variables and the cost-to-go value can be used as the feature weights in the Euclidean distance computation (Eq. 85).

For brevity, we only explain the use of PLS technique for dimensionality reduction. The objective of PLS in choosing the input subspace is to maximize the covariance between the

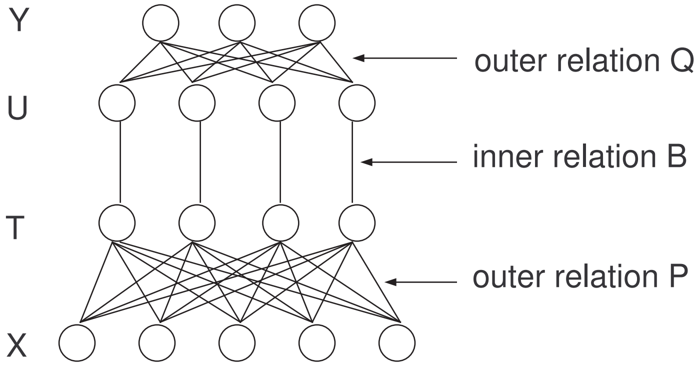


Figure 46: The structure of Partial Least Square (PLS) method

input data matrix $\mathbf{X}$ and the correlation to the output matrix $\mathbf{Y}$. The computation involves the determination of the orthogonal score matrices $\mathbf{T}$ and $\mathbf{U}$, and the loading matrices $\mathbf{P}$ and $\mathbf{Q}$. Because this method is scaled dependent, both the input and the output data matrices should be mean centered and scaled by their standard deviations. The PLS model can be visualized as shown in Figure 46. Currently, the most commonly used PLS algorithm is called NIPALS for nonlinear iterative partial least square technique, which can be found in [97].

This algorithm gives regression model as shown in Eq. 93, where $\hat{\mathbf{Y}}$ is the predicted output matrix and $\mathbf{B}_{pls}$ is a matrix of regression coefficients that can be computed from Eq. 94.

$$\hat{\mathbf{Y}} = \mathbf{X}\mathbf{B}_{pls} \tag{93}$$

$$\mathbf{B}_{pls} = \mathbf{W}(\mathbf{P}^T\mathbf{W})^{-1}\mathbf{B}\mathbf{Q}^T \tag{94}$$

where

$$\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k]$$

$$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_k]$$

$$\mathbf{B} = diag([\mathrm{b}_1, \mathrm{b}_2, \ldots, \mathrm{b}_k])$$

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k]$$

where k the number of scores and loadings (commonly called latent variables in PLS) required to adequately explain the covariance in the data. From PLS model, state variables whose correlation coefficients are very small can be removed from the k-nearest neighbor computation. Let the superscript r denote the state vector after removing the variables with small coefficients, and B^r_{pls} contains only those large coefficients. The distance computation in Eq. 85 can be replaced by

$$d_i = \sqrt{(x^r - x^r_i)^T |B^r_{pls}| (x^r - x^r_i)} \tag{95}$$

5.3.4.3 Reducing Model Simulation Time via Nonlinear Model Reduction

Integrating high-order nonlinear dynamic plant model to find the successive state can be computationally demanding especially when the system is "stiff", which is often the case when the plant has a large recycle loop like in this example. Given the fact that many variables of a plant-wide system are often highly correlated, it is possible to derive a significantly lower-order plant model using an appropriate method that preserves the accuracy of the resulting model. As a result, we consider the Proper Orthogonal Decomposition (POD) method coupled with the residualization technique presented in the previous section as a step to reduce the model integration time.

With all the improvement techniques proposed in Sections 5.3.4.1-5.3.4.3, the modified ADP procedures can be summarized in Table 17.

Table 17: Modified ADP algorithm with local gradient-based search for optimal action

```
1.  Closed-loop simulation with known control policies $\mu_i, i = 1, ..., n_\mu$.
    Store visited states $\{x(1), \ldots, x(N)\} \equiv X$ and the actions applied to the state point
    $x(j)$: $u^{x_j} = \mu_i(x(j))$, $j = 1, \ldots, N$.
2.  Initialization of the cost-to-go table:
        $J^0(x(k)) = \sum_{j=0}^{p-1} \phi(x(k+j), u(k+j)) + \phi_t(x(k+p)),\ k = 1, \ldots, N$
3.  Initialization of the local action sets.
    For each $x$, construct a local action set $A^x$ from
        $[u^x, u^{x_1}, \ldots, u^{x_m}] \in A^x$
    where $x_j, j = 1, \ldots, m$ is the first $m$ nearest neighbor points of $x$.
4.  Data Analysis for dimensionality reduction:
    – Apply a feature selection method (such as the PLS method) to find appropriate
      feature weighting matrix and to reduce the dimension in the $k$-nearest
      neighbor procedure.
    – Obtain a reduced-order model (such as via POD/residualization technique).
5.  Value Iteration
     REPEAT
        For each $x \in X$, solve
           $J^{i+1}(x) = \min \phi(x, u) + \tilde{J}^i(F_h(x, u))$
              s.t. $x_{LB} \le F_h(x, u) \le x_{UB}$
                   $u_{LB} \le u \le u_{UB}, \quad u_{LB} = \inf(A^x), u_{UB} = \sup(A^x)$
           where $\tilde{J}^i$ is the $k$-nearest neighbor cost-to-go approximation.
        LOOP
        $i \leftarrow i + 1$
    UNTIL convergence
```

5.3.5 Results of ADP Applied to a Deterministic Nonlinear Optimal Control

In this section, we implemented the proposed ADP method to the grade transition problem of an integrated plant shown in Figure 36. To generate the sample trajectories of the grade transitions from mode 1 to 2, 3, and 4, we used a reduced-order NMPC controller designed with prediction and control horizons of 5 and 2, respectively. We varied the weighting parameters of the controller so as to cover wider regions of the state space. The sample time of the controller was chosen as 4 hours. Here we assumed that the full state variables were measured. Total number of sample data collected was 990 points. The state for ADP was defined as:

$$X = \left[x_1^{plant}, \ldots, x_{39}^{plant}, x_{3B}, F, D, V, (x_{2B}^{sp} - x_{2B}), (B^{sp} - B), x_{2B}^{sp}, B^{sp}\right]^T \tag{96}$$

We include all the output variables of the process. x_{3B} was a concentration of product impurities, which has to be within a product specification limit. The internal flow rates F, D, and V were also included, since there are constraints on their upper and lower limits, as shown in Table 14. The setpoints as well as errors from setpoints were part of the state as they are required to compute the stage-wise cost, which is defined in Eq. 97, where $Q_u = 10000, Q_y = 6000$, and $R = 20I_{6\times6}$.

$$\phi(x(k), u(k)) = Q_y \left(\frac{x_{2B}^{sp} - x_{2B}(k+1)}{x_{2B,ss}}\right)^2 + Q_u \left(\frac{B^{sp} - B(k)}{B_{ss}}\right)^2 + \mathbf{\Delta u}(k)^T R \mathbf{\Delta u}(k) \tag{97}$$

In this work, we design the confidence bandwidth parameter σ as 3% of $6\sqrt{n}$, where n is the state dimension. The initial cost-to-go values of the stored state points range from 0 to 479. A quadratic penalty term is added to discourage and extrapolation to the region not covered by the simulation data. The maximum cost-to-go value J_{max} is set to 1000. The number of nearest neighbors used for the cost-to-go estimation was 4. The value iteration was performed using MATLAB 7 software in a Xeon^{TM} dual processor with 2.66 GHz/2.66 GHz and 2.00 GB of RAM.

5.3.5.1 Implementation of the Improved Off-line Learning Approach

First, the curse of dimensionality in the action space was overcome by using the SQP technique to find the optimal action within the region spanned by past actions stored for a given state and its neighbors. For each sample point, the number of nearest neighbors whose actions were kept in the local action set were 10. Since during our off-line simulation we had applied good MPC controllers, actions applied to the nearest neighbors provided good starting points from which the SQP search could be performed. In most cases, the search converged quickly to the optimal solution. Had the initial controllers been very poor, one should consider exploring many more nearest neighbors' actions to expand the search region.

To reduce the state dimension in the k-nearest neighbor computation, the PLS technique was applied to determine the correlations between the state and the cost-to-go value. The PLS model coefficients are plotted in Figure 47 and the descriptions of the states with large coefficients are given in Table 18. In addition to the state variables 39, 45-47, which explicitly show up in the stage-wise cost function, the PLS model also shows that reactor variables (states 1-3, and 41) and some distillation variables (states 33, 35, 36, 38, 40) have a lot of variations that affect the cost-to-go values. Therefore, to reduce the computational overhead and improve the accuracy of $\tilde{J}(F_h)$ computation, only these state variables were used in the computation and their corresponding feature weights were set as the magnitudes of the PLS regression coefficients shown in Table 18.

For the nonlinear model integration to find the successive state, we applied the POD method to reduce the model integration time. As shown in Section 5.2, the projection matrices P and Q in this case have 7 and 32 row ranks, respectively. In addition, the residualization technique was better than the truncation in terms of preserving the steady-state characteristics of the system. Therefore, in this study the residualization method was used to derive the reduced order model of the form in Eqs. 98-99 for the integration, where $x = [P^T, Q^T]\bar{x} + x_{ss}$.

$$\dot{\bar{x}}_1(t) = Pf\left([P^T, Q^T]\bar{x} + x_{ss}, u\right) \tag{98}$$

$$0 = Qf\left([P^T, Q^T]\bar{x} + x_{ss}, u\right) \tag{99}$$

The value iteration step was performed with the convergence criteria as shown in Eq. 100

$$\left|J^{i+1}(x) - J^{i}(x)\right|_{\infty} < 1 \tag{100}$$

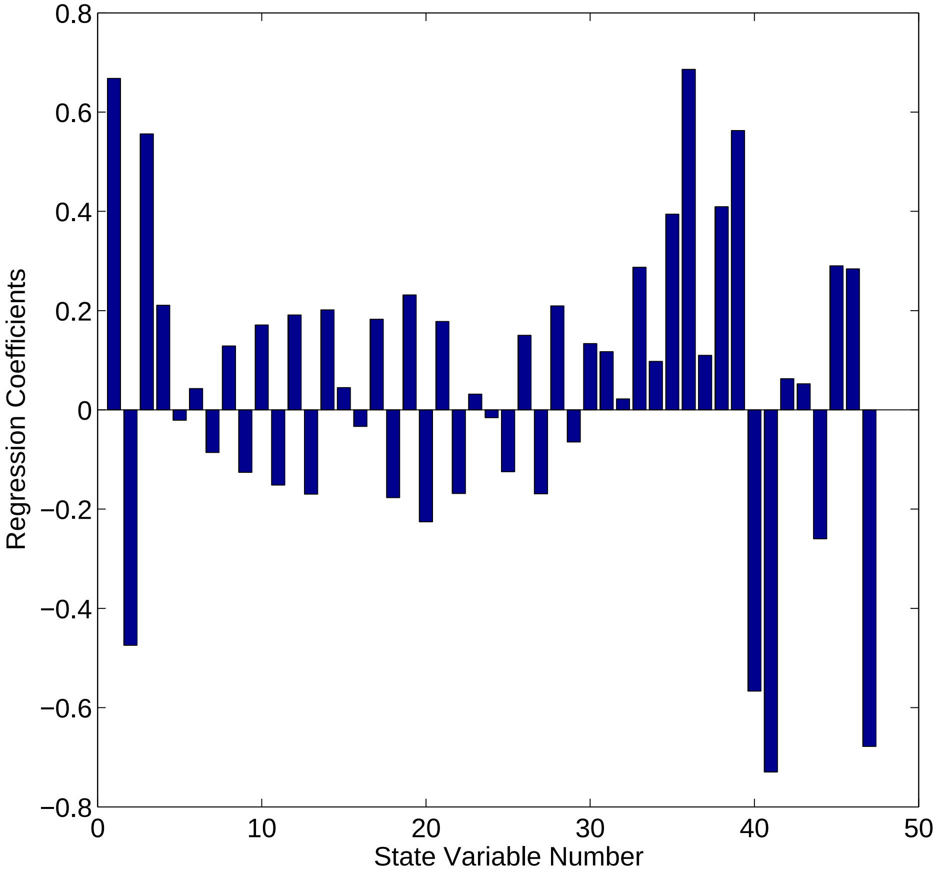


Figure 47: Coefficient of the PLS Model

Table 18: Descriptions of the state variables with large PLS regression coefficients

State Number	Description	Coefficient
1	Reactor holdup (M_R)	0.668
2	Composition of A in the reactor (x_{1R})	-0.474
3	Composition of B in the reactor (x_{2R})	0.556
33	Composition of A on tray 14 ($x_{1,14}$)	0.287
35	Composition of A on tray 15 ($x_{1,15}$)	0.394
36	Composition of B on tray 15 ($x_{2,15}$)	0.686
38	Composition of A in the product (x_{1B})	0.409
39	Composition of B in the product (x_{2B})	0.563
40	Impurity in the product (x_{3B})	-0.567
41	Reactor effluent flow rate (F)	-0.730
45	Error from x_{2B} setpoint ($B^{sp} - B$)	0.290
46	Error from B setpoint (x_{2B}^{sp})	0.284
47	Production rate setpoint (B)	-0.678

The value iteration converged after 18 iterations as shown in Figure 48.

We also compare the off-line computational time between before and after the improvement methods were applied and results are summarized in Table 19. Clearly, without the model reduction technique, solving the high-dimensional model equation would be require very high off-line computational load.

5.3.5.2 *On-line Performance Compararison*

Once the cost-to-go table converged, it was used in the on-line optimization. The result of grade transition from mode 1 to 2 is shown in Figures 49 and 50, which show that the system can achieve the grade transition very quickly. The resulting performance as defined by Eq. 56 with Q_u, Q_y and R equal to 10000, 6000, and $20I_{6\times6}$, respectively, is 13.27. This shows a lot of improvement from the use of the NMPC shown in Section 5.2 as compared in Table 20.

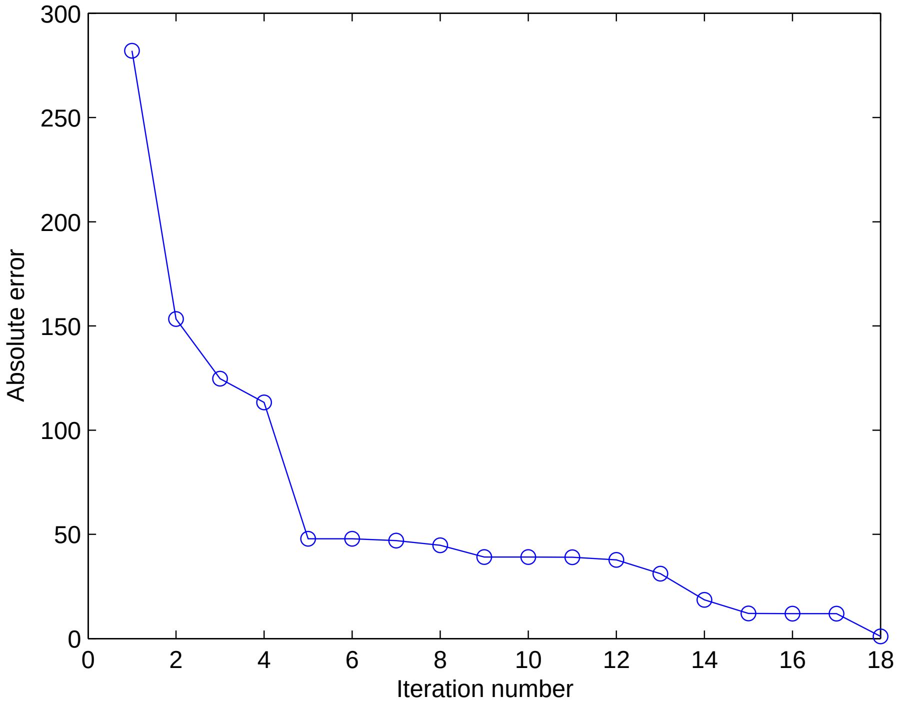


Figure 48: Absolute errors of value iteration: deterministic case

Table 19: Value iteration time of different improvement methods

Methods	Average computational time per iteration (minutes)
• SQP search over local action sets	595.13
• SQP search over local action sets • reduced order model for state transition	112.30
• SQP search over local action sets • reduced order model for state transition • reduced dimension in k-nearest neighbor computation	92.31

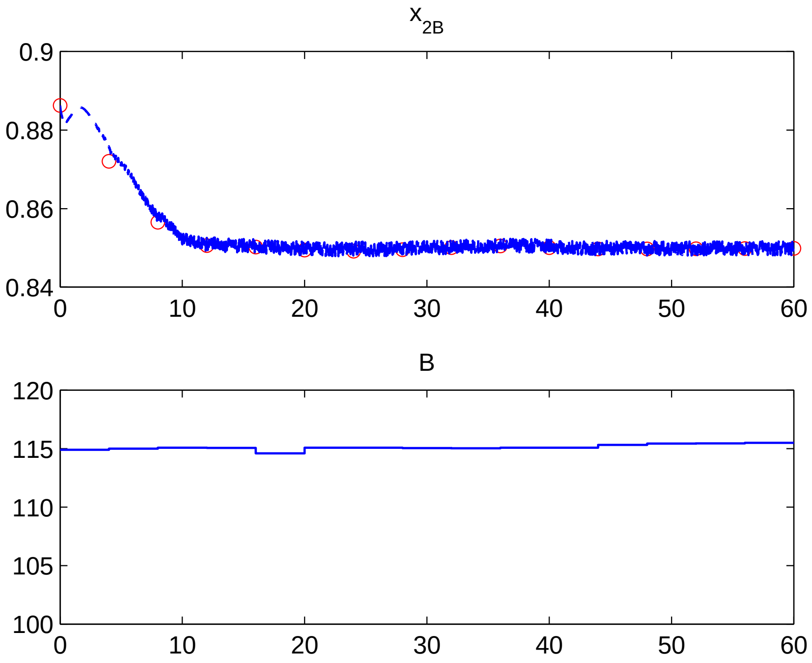


Figure 49: Product variables during the grade transition performed by ADP controller (solid: output, o: prediction)

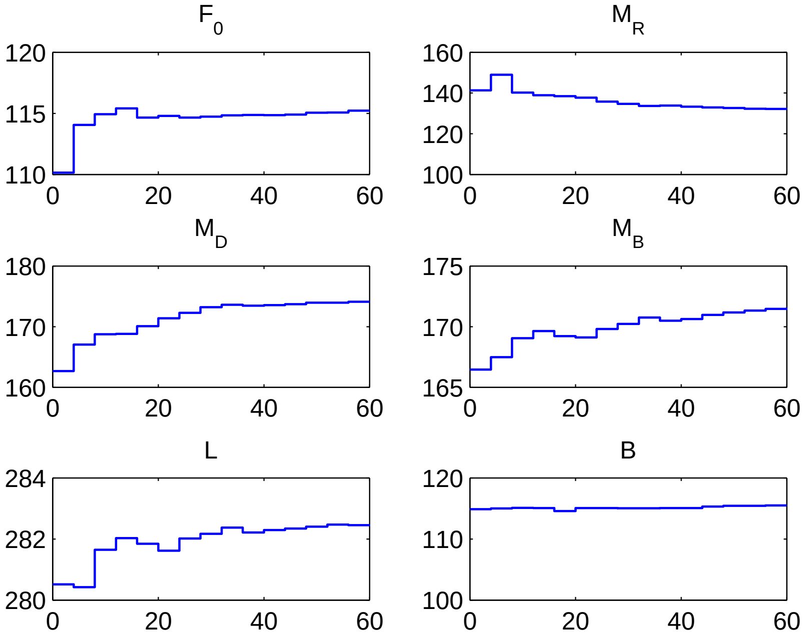


Figure 50: MV profiles during the grade transition performed by ADP controller

Table 20: On-line performance comparison between of NMPC and ADP schemes running at every 4 hours

Method	Performance
NMPC with a full-order model	90.70
NMPC with a reduce-order model via residualization	91.73
NMPC with a reduced-order model via truncation	96.02
ADP-based controller	13.27

In addition, the most striking feature of this method is in its drastic reduction in the on-line computation as shown in Table 21. This is due to the fact that the ADP controller only needs to solve a single-stage optimization problem at each time, which is a much smaller problem than those solved in NMPCs. Note that the computational load of the mathematical programming-based approach, whether it is solved via a sequential or a simultaneous method, is a function of the prediction and the control horizon. While longer prediction horizons can potentially offer better performance, it increases the on-line computational load of this approach. However, the on-line computational load of ADP is completely independent of the size of prediction horizon as the multi-stage optimization problem is solved off-line and the on-line optimization problem involves only a single stage optimization.

Table 21: Computational time (in minutes) of NMPC and ADP schemes running at every 4 hours

run number	NMPC with full order	ADP
1	8.6883	0.4207
2	3.3734	0.1905
3	3.6133	0.1892
4	3.4581	0.0906
5	3.8854	0.1507
6	2.3268	0.1261
7	2.2977	0.1335
8	2.2482	0.0990
9	2.1891	0.0663
10	1.9180	0.0765
11	2.9276	0.0547

5.3.6 Results of ADP Application to a Stochastic Nonlinear Optimal Control

In this section, we consider the case where the integrated white noises are introduced to the feed composition and the kinetic rate constant k_1. The feed now consist of components A ($x_{1,0}$) and C ($x_{3,0}$). The changes in $x_{3,0}$ and k_1 have the sample time of 0.02 modeled as

$$x_{3,0}(i+1) = x_{3,0}(i) + e_1(i) \tag{101}$$

$$k_1(i+1) = k_1(k) + e_2(i) \tag{102}$$

where $e_1(k) \sim \mathcal{N}(0, 0.0005^2)$ and $e_2(k) \sim \mathcal{N}(0, 0.0005^2)$.

To maintain the data set at reasonable size, we chose 4 representative realizations of the stochastic disturbances for the simulation:

1. k_1 and $x_{3,0}$ are trending up
2. k_1 is trending down and $x_{3,0}$ is trending up
3. k_1 is trending up and $x_{3,0}$ is up for the first 30 hours and down from there.
4. k_1 is trending down and $x_{3,0}$ is up for the first 30 hours and down from there.

The realizations of these disturbance scenarios are shown in Figure 51.

We designed seven reduced-order NMPC controllers with the sample time of 4 hours for use as the initial controllers for simulations. The prediction and the control horizons of all controllers were chosen as of 5 and 2, respectively. The weighting matrices of these controllers are shown in Table 22.

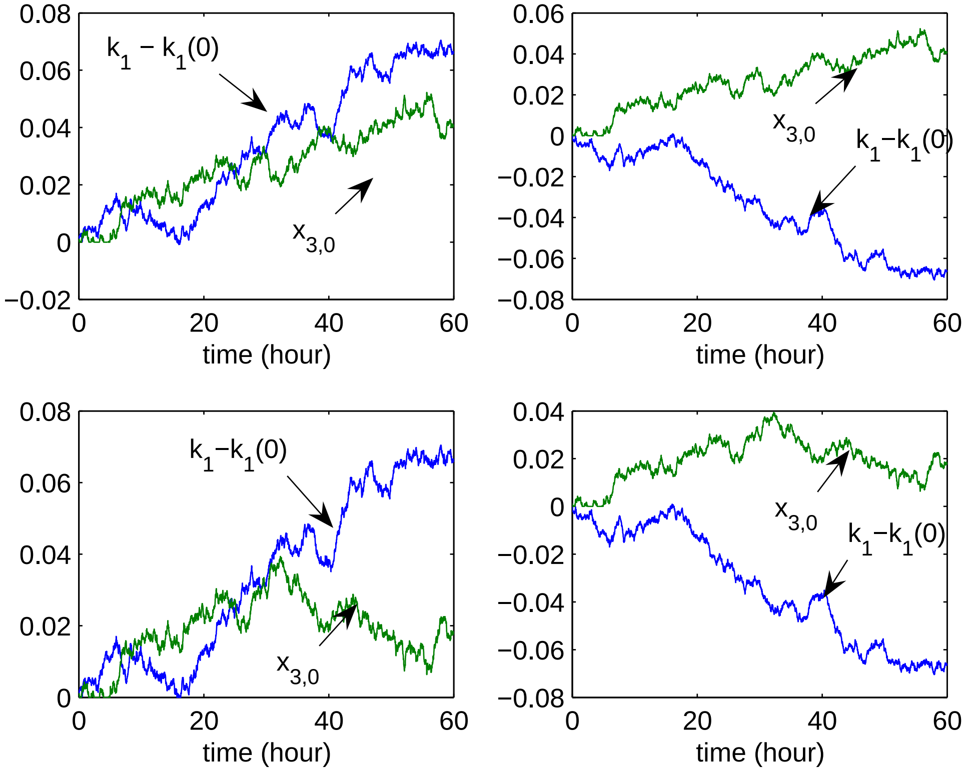


Figure 51: Representative disturbance scenarios for data generation

Table 22: Weighting matrices of the initial controllers

	Q_u	Q_y	R
Controller 1	10000	6000	$20I_{6\times6}$
Controller 2	10000	96000	$20I_{6\times6}$
Controller 3	1250	6000	$20I_{6\times6}$
Controller 4	10000	6000	$160I_{6\times6}$
Controller 5	80000	6000	$20I_{6\times6}$
Controller 6	10000	750	$20I_{6\times6}$
Controller 7	10000	3000	$160I_{6\times6}$

We consider the grade transition scenario from mode 1 to mode 2. We store 600 representative data points from the simulations for the cost-to-go training. The discounted factor used to compute the cost-to-go is 0.9.

The cost-to-go update in the off-line value iteration was performed by taking the expectation of the cost over 30 stochastic disturbance realizations. The initial cost-to-go values of the stored state points range between 0.398 to 167. The convergence criteria of the iteration was chosen as $\left|J^{i+1}(x) - J^{i}(x)\right|_{\infty} < 1$, which is small enough relative to the range that the cost-to-go values varied.

The cost-to-go learning converged after 12 iterations as shown in Figure 52.

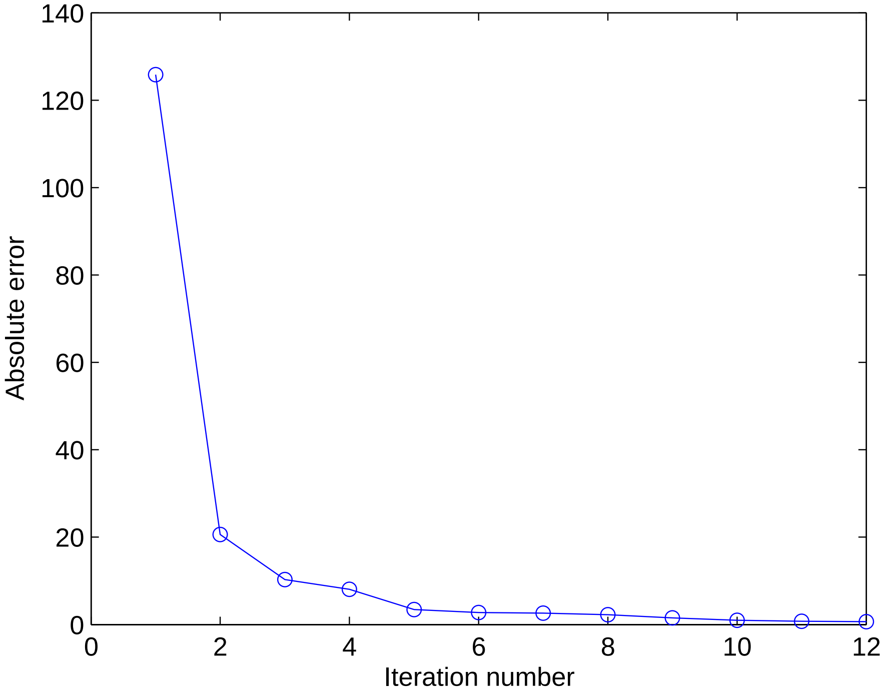


Figure 52: Absolute errors of value iteration: deterministic case

The on-line performance was compared by generating 12 new realizations of the stochastic disturbances in k_1 and $x_{3,0}$. We compared the performance of the ADP optimizer with the seven reduced-order NMPCs used as the initial controllers for simulations. The performance as measured by Eq. 56 over 60-hour horizon are shown in Table 23. The ADP based controller shows superior performance compared to all the other reduced order NMPC controllers under uncertainty.

Table 23: On-line performance comparison with 12 stochastic disturbance scenarios

Performance	mean cost	standard deviation
ADP controller	37.83	5.30
NMPC 1	130.28	22.91
NMPC 2	122.33	13.62
NMPC 3	136.56	18.26
NMPC 4	152.98	27.84
NMPC 5	125.52	17.55
NMPC 6	147.43	29.76
NMPC 7	131.21	20.70

In this experiment, the disturbance scenario 6 was the one that the ADP controller showed the median performance of 34.27. The disturbances are plotted in Figure 53.

The performance of the ADP controller are shown in Figures 54 and 55. From the time 10 hour, the production concentration was closely controlled around 0.85-0.86, while the production rate reached reached the new target in 12 hours.

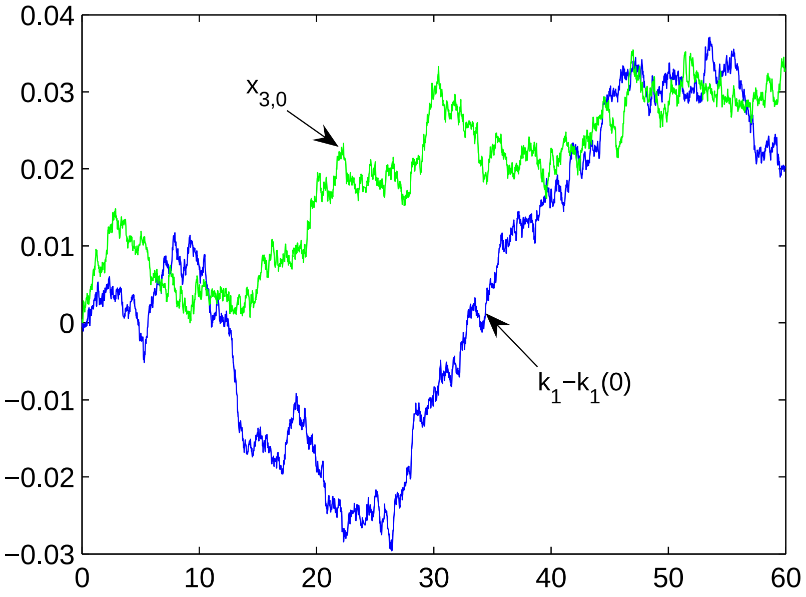


Figure 53: Disturbance scenario 6

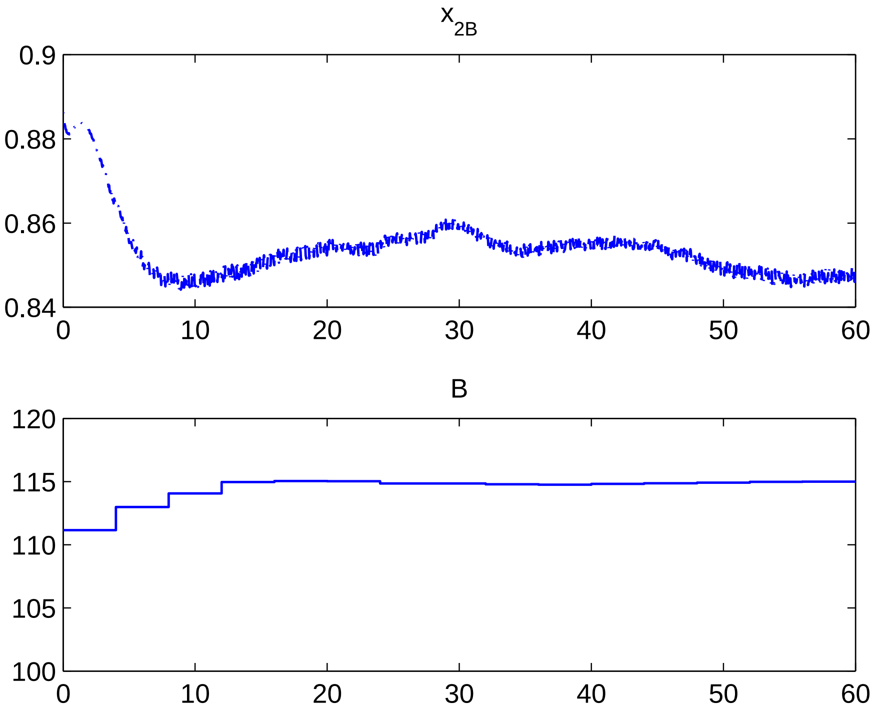


Figure 54: On-line performance of the ADP controller during the disturbance scenario 6: Output variables

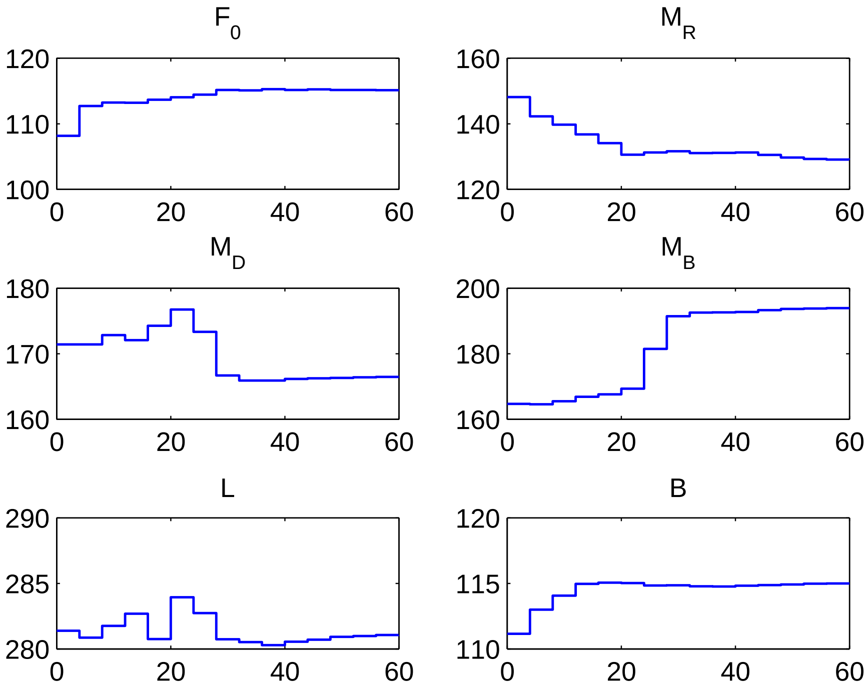


Figure 55: On-line performance of the ADP controller during the disturbance scenario 6: Manipulated variables

The performance of NMPC number 1, which was the best performance among all the NMPC controllers in this case are shown in Figures 56 and 57. From the time 10 hour, the product concentration varied between 0.84-0.86. In this case, the production rate reached the target after 20 hours.

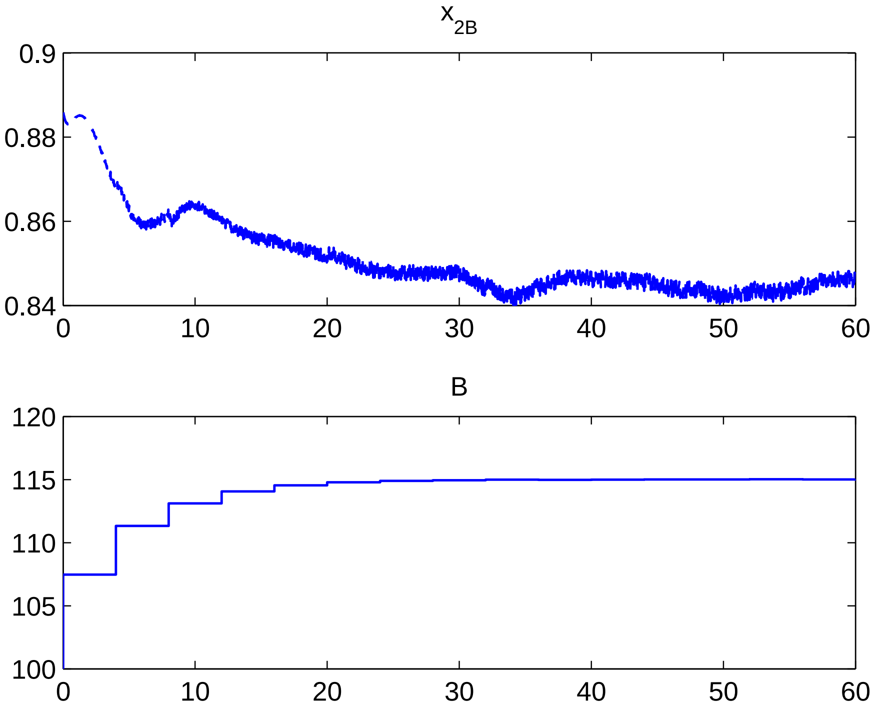


Figure 56: On-line performance of the NMPC controller 1 during the disturbance scenario 6: Output variables

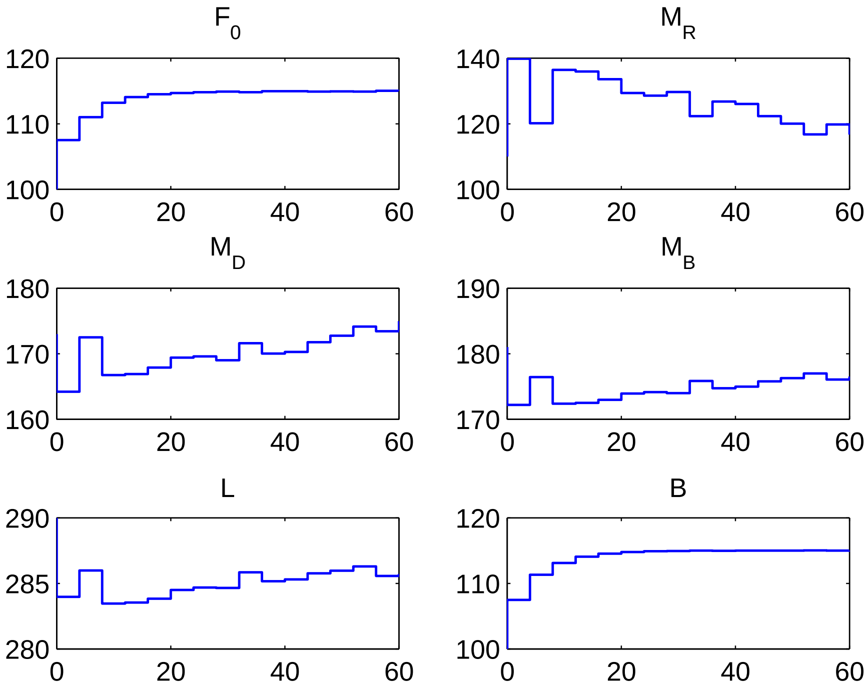


Figure 57: On-line performance of the NMPC controller 1 during the disturbance scenario 6: Manipulated variables

5.4 Conclusions

The approximate Dynamic Programming approach proves to be a more efficient method for on-line optimization of an integrated plant than the conventional NMPC. However, the off-line learning procedures suggested in [15] can take significantly long time for the cost-to-go value function to converge. Therefore, this case study presented systematic ways to reduce the search space for optimal action and the time taken in the cost-to-go update of each state point. We incorporated local gradient-based search technique to accelerate the convergence to local optimal action and overcome the curse-of-dimensionality in the action space. The dimensionality reduction technique via Partial Least Square (PLS) technique has been used to reduce the state dimension and to improve the prediction accuracy in the k-nearest neighbor cost-to-go approximation. Finally, the nonlinear model reduction technique via Proper Orthogonal Decomposition (POD) and residualization was applied to reduce the model integration time, thus reducing the computational time of the cost-to-go approximation. With the improvement techniques, the method can be applied to solve the integrated plant problem efficiently and results in superior performance to conventional NMPC framework in both deterministic and stochastic cases.

CHAPTER 6

CONTRIBUTIONS AND FUTURE WORK

6.1 Contributions

This thesis was motivated by the need for efficient dynamic plant-wide optimization framework for an integrated plant with material recycle loop. This system presents a challenge in modeling, control, and optimization. This is because the system exhibits a multiple time-scale phenomena, where the system shows very fast response at the initial time and followed by very slow dynamics over a long period of time. The current steady-state optimization strategy does not work very well in this system because of the limited execution rate. Furthermore, the fast-rate dynamic optimization can result in an extremely high computational requirement and sensitivity to high frequency dynamics that are irrelevant to the plant-wide interactions. The major contributions of this thesis are:

- Proposing a novel multi-scale dynamic plant-wide optimization and control architecture suitable for an integrated plant.
- Addressing problems in identifying a dynamic model of an integrated plant and proposing a grey-box modeling technique to improve the long-range prediction accuracy of the model.
- Understanding and overcoming the off-line computational difficulties in applying the existing approximate dynamic programming (ADP) framework to solving a nonlinear optimization of an integrated plant.

In Chapter 3, we proposed the multi-scale dynamic optimization framework for an integrated plant. The optimization frequency should be chosen so as to avoid the fast frequency range of the unit operations and to take into account the computational feasibility of the framework. In the case where there are multiple MPCs at the unit-base control level, a coordination scheme should be used to avoid passing infeasible setpoints to the MPCs. The examples used in this chapter showed that suggested method is superior to the current steady-state and the single-scale dynamic optimization schemes. The former is limited in terms of the execution frequency, whereas the latter may be very sensitive to fast dynamics that are irrelevant to the plant-wide objective.

When the plant-wide interaction model based on first-principles is unavailable, a system identification experiment has to be conducted. In Chapter 4, we addressed the difficulties associated with the ill-condition and the long settling time characteristics of the integrated plant. Input has to excite the low-gain directions more than the high-gain direction in order to obtain a model that adequately explains the low-gain direction, which is very difficult to control well. We also propose a grey-box modeling method that incorporates the steady-state information of the plant to improve the model prediction quality of the identified model. The main premise is that we assume the slow-scale dynamics can be modeled as a first-order system. Then only the steady-state gain and the settling time of the plant need to be supplied to construct the approximation. The proposed method was shown to improve the performance of the plant-wide control in an integrated plant example composed of a reactor and a distillation column. The ease of design and implementation makes it potentially appealing to extend to other industrial examples.

In Chapter 5, a case study of a nonlinear integrated plant with a reactor, a distillation column and a recycle loop is presented to show that the nonlinear model predictive control can be very inefficient in solving an optimal control problem of this system. This study considered the use of the ADP framework for plant-wide dynamic optimization. This method offers a way to compute the optimal policy with or without stochastic uncertainty in an off-line manner. Then, the on-line optimization can be reduced to a single-stage optimization, and thus decreasing the on-line computational load dramatically. The existing simulation-based ADP framework for process control problems is based on the premise the part of the state space that is relevant to optimal control or near optimal is much smaller than the entire state space. The simulations with good suboptimal control policies are used to generate state points. Then, cost-to-go values are improved by the value iteration scheme. We addressed the difficulties in applying the existing ADP method to an optimal control problem of an integrated plant; They include the curse-of-dimensionality in the action space, the long model simulation time in a high dimensional system, and the determination of suitable feature weighting factors in computing the cost-to-go approximation. Our contribution is to combine several methodologies with ADP to overcome these difficulties and show how to apply optimal control to an integrated plant. This resulting approach showed superior performance in solving the optimal control problem of the integrated plant case study, with or without uncertainty, compared to the conventional NMPC.

6.2 Future Work

The conceptual framework of a multi-scale optimization and control strategy can be applied to other challenging integrated plants with recycles. Some complex nonlinear process systems, such as the recovery plant in the pulp and paper mill, has a very long transient dynamics that it is seldom at the steady-state. One of the main advantages of our approach is that it does not require the system to be at the steady state. Nonetheless, many nonlinear complex process systems are still very difficult to optimize in real time. The ADP approach may be the solution to these problems. Possible directions of future research should address the following topics:

- Systematic exploration: Systematic way of exploring state space is an open question for the ADP research. This may give a significant improvement in performance, especially when the initial control policy is far from optimal. However, on-line learning is often expensive in process industries and a cautious scheme to expand the learning is necessary.

- Systematic on-line update of the cost-to-go function: Like any other model-based control schemes, when the actual process changes, the process model used in the optimization has to be re-evaluated. However, in the case of the ADP framework, the cost-to-go function also needs to be retrained. The future research in ADP should, therefore, consider on-line monitoring and updating scheme to detect the drift of the process and provide an on-line update for the cost-to-go value.

- Parallelization of the off-line iteration: The sequential update scheme in the off-line value iterations can require significantly long time to complete as one considers an application to larger and larger problems. In this case, parallelizing the value iteration scheme can reduce the total time required to obtain the optimal cost-to-go. Partitioning scheme has been proposed to handle some discrete-time markov decision processes where states are not highly dependent as mentioned in Section 5.3. However, for problems with continuous state-space, it remains an issue how to systematically partition the state space and how to handle the trade-off between the convergence speed gain and the potential information loss when states in different partitions are not updated together.

APPENDIX A

LINEARIZED MODEL OF THE ILLUSTRATIVE REACTOR-DISTILLATION INTEGRATED PLANT

The state of the reactor-distillation integrated plant is defined as

$$X = [\ x_{1R},\ \ x_{2R},\ \ x_{1D},\ \ x_{2D},\ \ x_{1,1},\ \ x_{1,2},\ \ \ldots,\ \ x_{1,N},\ \ x_{2,N},\ \ x_{1B},\ \ x_{2B}\ ]^T \tag{103}$$

where concentrations in the reactor, the condenser, and the product stream are denoted by x_{ij}, where i indicates chemical species (1: reactant A, 2: desired product B, 3: undesired product C) and j represents processes (R: reactor, D: condenser, B: reboiler). The distillation column compositions are given by $x_{i,k}$ where k is the tray location.

The nonlinear ordinary differential equation of this system is given in Section 5.1. With a perfect level control assumption, the linearized model of this system in the state space form is given as follows:

$$\dot{x} = Ax + [B^u, B^d] \begin{bmatrix} u \\ d \end{bmatrix} \tag{104}$$

$$y = Cx + Du \tag{105}$$

where $u = [F_0, L, V]^T, d = [x_{2,0}, x_{3,0}]^T$. Feed stream may consist of impurity components 2 or 3 as denoted by $x_{2,0}$ and $x_{3,0}$, respectively. The non-zero elements in the matrix A before any scaling is applied are shown in Table 24. Matrix B is shown in Eqs. 106.

Table 24: Non-zero elements in the state space matrix A of the reactor-distillation integrated plant

(1,1)	-18.091	(1,3)	16.182						
(2,1)	1	(2,2)	-18.091	(2,4)	16.182				
(3,3)	-11.965	(3,5)	3.6241	(3,6)	-2.9894				
(4,4)	-11.965	(4,5)	-0.46651	(4,6)	6.1407				
(5,3)	2.6364	(5,5)	-8.3361	(5,6)	4.7016	(5,7)	5.8389	(5,8)	-4.7016
(6,4)	2.6364	(6,5)	0.7337	(6,6)	-12.294	(6,7)	-0.83207	(6,8)	9.6946
(7,5)	2.6364	(7,7)	-8.4753	(7,8)	4.7016	(7,9)	5.8773	(7,10)	-4.7025
(8,6)	2.6364	(8,7)	0.83207	(8,8)	-12.331	(8,9)	-0.85709	(8,10)	9.7066
(9,7)	2.6364	(9,9)	-8.5136	(9,10)	4.7025	(9,11)	5.8884	(9,12)	-4.7031
(10,8)	2.6364	(10,9)	0.85709	(10,10)	-12.343	(10,11)	-0.86335	(10,12)	9.7111
(11,1)	17.091	(11,9)	2.6364	(11,11)	-25.616	(11,12)	4.7031	(11,13)	6.1631
(11,14)	-4.6951								
(12,2)	17.091	(12,10)	2.6364	(12,11)	0.86335	(12,12)	-29.438	(12,13)	-1.0745
(12,14)	9.766								
(13,11)	19.727	(13,13)	-25.89	(13,14)	4.6951	(13,15)	6.6821	(13,16)	-4.6759
(14,12)	19.727	(14,13)	1.0745	(14,14)	-29.493	(14,15)	-1.4763	(14,16)	9.8628
(15,13)	19.727	(15,15)	-26.409	(15,16)	4.6759	(15,17)	7.647	(15,18)	-4.6279
(16,14)	19.727	(16,15)	1.4763	(16,16)	-29.59	(16,17)	-2.2326	(16,18)	10.021
(17,15)	19.727	(17,17)	-27.374	(17,18)	4.6279	(17,19)	9.3807	(17,20)	-4.508
(18,16)	19.727	(18,17)	2.2326	(18,18)	-29.748	(18,19)	-3.6167	(18,20)	10.247
(19,17)	19.727	(19,19)	-29.108	(19,20)	4.508	(19,21)	12.288	(19,22)	-4.232
(20,18)	19.727	(20,19)	3.6167	(20,20)	-29.974	(20,21)	-5.9933	(20,22)	10.494
(21,19)	19.727	(21,21)	-32.016	(21,22)	4.232	(21,23)	16.596	(21,24)	-3.7013
(22,20)	19.727	(22,21)	5.9933	(22,22)	-30.222	(22,23)	-9.6039	(22,24)	10.649
(23,21)	19.727	(23,23)	-36.324	(23,24)	3.7013	(23,25)	21.907	(23,26)	-2.9121
(24,22)	19.727	(24,23)	9.6039	(24,24)	-30.376	(24,25)	-14.148	(24,26)	10.605
(25,23)	19.727	(25,25)	-41.634	(25,26)	2.9121	(25,27)	27.172	(25,28)	-2.031
(26,24)	19.727	(26,25)	14.148	(26,26)	-30.333	(26,27)	-18.707	(26,28)	10.397
(27,25)	19.727	(27,27)	-46.899	(27,28)	2.031	(27,29)	31.447	(27,30)	-1.2724
(28,26)	19.727	(28,27)	18.707	(28,28)	-30.124	(28,29)	-22.389	(28,20)	10.169
(29,27)	19.727	(29,29)	-51.175	(29,30)	1.2724	(29,31)	34.5	(29,32)	-0.73635
(30,28)	19.727	(30,29)	22.389	(30,30)	-29.896	(30,31)	-24.897	(30,32)	10.056
(31,29)	19.727	(31,31)	-54.227	(31,32)	0.73635	(31,33)	36.687	(31,34)	-0.40378
(32,30)	19.727	(32,31)	24.897	(32,32)	-29.783	(32,33)	-26.409	(32,34)	10.146
(33,31)	19.727	(33,33)	-56.414	(33,34)	0.40378	(33,35)	38.618	(33,36)	-0.21278
(34,32)	19.727	(34,33)	26.409	(34,34)	-29.874	(34,35)	-27.217	(34,36)	10.556
(35,33)	11.989	(35,35)	-24.022	(35,36)	0.12931				
(36,34)	11.989	(36,35)	16.541	(36,36)	-6.9674				

$$B = 10^3 \begin{bmatrix} 0.91234 & -0.45416 & 0.45416 \\ -0.85375 & 0.40469 & -0.40469 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0.42988 & -0.060225 \\ 0 & -0.40434 & 0.056646 \\ 0 & 0.1142 & -0.015998 \\ 0 & -0.10093 & 0.01414 \\ 0 & 0.031583 & -0.0044246 \\ 0 & -0.02468 & 0.0034576 \\ 0.13097 & -0.1215 & 0.010695 \\ -0.13111 & 0.12525 & -0.011213 \\ 0.21134 & 0 & -0.01021 \\ -0.21068 & 0 & 0.010178 \\ 0.37593 & 0 & -0.018161 \\ -0.37473 & 0 & 0.018103 \\ 0.6289 & 0 & -0.030382 \\ -0.6268 & 0 & 0.03028 \\ 0.95141 & 0 & -0.045962 \\ -0.94792 & 0 & 0.045793 \\ 1.2407 & 0 & -0.059936 \\ -1.2351 & 0 & 0.059669 \\ 1.3385 & 0 & -0.064661 \\ -1.3297 & 0 & 0.064239 \\ 1.1786 & 0 & -0.056939 \\ -1.1638 & 0 & 0.056224 \\ 0.86625 & 0 & -0.041848 \\ -0.83862 & 0 & 0.040513 \\ 0.55581 & 0 & -0.026851 \\ -0.50093 & 0 & 0.0242 \\ 0.32624 & 0 & -0.01576 \\ -0.21525 & 0 & 0.010399 \\ 0.18155 & 0 & -0.0087707 \\ 0.040474 & 0 & -0.0019553 \\ 0.059491 & 0 & -0.0028739 \\ 0.20101 & 0 & -0.0097108 \end{bmatrix} \tag{106}$$

APPENDIX B

MODEL REDUCTION BASED ON THE BALANCED METHOD

Consider a stable linear time-invariant system of the form:

$$\begin{aligned} \dot{x} &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{107}$$

where x, u, and y represent a state vector, an input vector, and an output vector, respectively. The controllability and the observability grammians of this system are defined as shown in **Definitions** 1 and 2.

Definition 1: Controllability gramian

$$P = \int_0^\infty e^{At} B B^T e^{A^T t} dt \tag{108}$$

Definition 2: Observability gramian

$$Q = \int_0^\infty e^{A^T t} C^T C e^{A^T t} dt \tag{109}$$

The controllability gramian has full rank for a stable and controllable system. For stable and observable systems the observability gramian will have full rank. In addition, these gramians are the unique positive definite solutions of the Lyapunov equations as follows:

$$AP + PA^T + BB^T = 0 \tag{110}$$

$$A^T Q + QA + C^T C = 0 \tag{111}$$

These controllability and observability gramians can be coordinate transformed and denoted by $\bar{P}$ and $\bar{Q}$, respectively. The system is in a *balanced* form if and only if

$$\bar{P} = \bar{Q} = \Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_n \end{bmatrix} \tag{112}$$

where

$$\bar{P} = TPT^T \tag{113}$$

$$\bar{Q} = (T^{-1})^T Q T^{-1} \tag{114}$$

where T is a coordinate transformation matrix. $\sigma_1 \leq \sigma_2 \leq \dots \sigma_n$ are referred as the Hankel singular values of the system. The main idea here is that the singular values of the controllability gramian correspond to the amount of input energy required to change the corresponding states. On the other hand, singular values of the observability gramian are the energy generated by the corresponding states. With this coordinate transformation matrix, the system can be transformed into the the balanced form given in Eq. 115.

$$\bar{x} = TAT^{-1}\bar{x} + TBu \tag{115}$$

$$y = CT^{-1}\bar{x} \tag{116}$$

In this balanced form, the elements in the state vector $\bar{x}$ are ranked form the most observable and controllable states to the least ones. As a result, one can obtain a reduced order model by eliminating the states that contribute very little to the input-output behavior of the system. To obtain the coordinate transformation T, the following steps are involved.

$$R = Q^T Q \tag{117}$$

$$RPR^T = U\Sigma^2 U^T \tag{118}$$

$$T = \Sigma^{-1/2} U^T R^T \tag{119}$$

For nonlinear system, the gramians as defined in Eqs. 108 and 109 cannot be computed. The original work in deriving the empirical gramians for nonlinear system by Lall [72, 73] only applied to a nonlinear control-affine system. Hahn et al. [74] generalized the method to general nonlinear systems, which can be represented as follows:

$$\dot{x} = f(x(t), u(t)) \tag{120}$$

$$y(t) = h(x(t)) \tag{121}$$

where f and h are continuous nonlinear functions. The idea is to compute the so-called controllability and observability covariance matrices by exciting the system and collecting the data on the input-to-state and state-to-output behavior of the system. Lall defined the following sets for the empirical controllability gramian:

$$\mathcal{T}^p = \{T_1, \ldots, T_r | T_i \in \mathbb{R}^{n\times n}, T_i T_i^T = I, i = 1, \ldots, r\} \tag{122}$$

$$\mathcal{M} = \{c_1, \ldots, c_s | c_i \in \mathbb{R}^+, i = 1, \ldots, s\} \tag{123}$$

$$\mathcal{E}^p = \{e_1, \ldots, e_p | \text{standard unit vectors in } \mathbb{R}^p\} \tag{124}$$

where r is the number of matrices for perturbation directions, s is the number of excitation magnitudes in each direction, and p is the number of system inputs. The definitions for the controllability covariance matrix is given as follows:

$$P = \sum_{l=1}^{r} \sum_{m=1}^{s} \sum_{i=1}^{p} \frac{1}{rsc_m^2} \int_0^\infty \Phi^{ilm}(t) dt \tag{125}$$

where the covariance of the state $\Phi^{ilm}(t) \in \mathbb{R}^{n\times n}$ is computed from:

$$\Phi^{ilm}(t) := (x^{ilm}(t) - x_{ss})(x^{ilm}(t) - x_{ss})^T \tag{126}$$

where $x^{ilm}(t)$ is the state corresponding to the input $u(t) = c_m T_l e_i v(t) + u_{ss}(0)$. The input direction is determined by $T_l e_i$ and the type of input signal is denoted by $v(t)$. The subscript ss denotes the vector at the steady-state.

The observability covariance matrix is defined as follows:

$$Q = \sum_{l=1}^{r} \sum_{m=1}^{s} \frac{1}{rsc_m^2} \int_0^\infty T_l \Psi^{lm}(t) T_l^T dt \tag{127}$$

where the covariance of the output $\Psi^{lm}(t) \in \mathbb{R}^{n\times n}$ is given by

$$\Psi_{ij}^{lm}(t) := (y^{ilm}(t) - y_{ss}^{ilm})^T (y^{ilm}(t) - y_{ss}^{ilm}) \tag{128}$$

where $y^{ilm}(t)$ is the output corresponding to the initial condition $x(0) = c_m T_l e_i + x_{ss}$. Once these gramians are constructed, method to derive the coordinate transformation is same as in the linear case.

REFERENCES

[1] J. Z. Lu. Challenging control problems and emerging technologies in enterprise optimization. In *Proceedings of the 6th IFAC Symposium on Dynamics and Control of Process Systems (DYCOPS-6)*, pages 23–34. IFAC, 2001.

[2] C. R. Cutler and R. T. Perry. Real time optimization with multivariable control is required to maximize profits. *Comput. Chem. Eng.*, 7(5), 1983.

[3] T. E. Marlin and A. N. Hrymak. Real-time operations optimization of continuous processes. In *Proceedings of Chemical Process Control V Conference*, pages 156–164, Tahoe City, NV, 1996.

[4] D. C. White. Online optimization: What have we learned? *Hydrocarbon Processing*, 77(6):55–59, June 1998.

[5] J. J. Downs and E. F. Vogel. A plant-wide industrial process control problem. *Comput. Chem. Eng.*, 17(3):245–255, 1993.

[6] T. J. McAvoy and N. Ye. Base control for the tennessee eastman problem. *Comput. Chem. Eng.*, 18(5):383–413, 1994.

[7] N. L. Ricker. Decentralized control of the tennessee eastman challenge process. *J. Process Contr.*, 6(4):205–221, 1996.

[8] K. L. Wu and C. C. Yu. Reactor/separator processes with recycle –1. candidate control structure for operability. *Comput. Chem. Eng.*, 11:1291–1316, 1996.

[9] C. C. Pederson P. J. Vermeer, W. M. Canney, and J. S. Ayala. Blend-control system all but eliminates reblends for canadian refiner. *The Oil and Gas Journal*, 95(30):74–79, 1997.

[10] M. T. de Gouvêa and D. Odloak. One-layer real time optimization of LPG production in the FCC unit: procedure, advantages and disadvantages. *Comput. Chem. Eng.*, 22:S191–S198, 1998.

[11] A. C. Zanin, M. T. de Gouvêa, and D. Odloak. Industrial implementation of a real-time optimization strategy for maximizing production of LPG in a FCC unit. *Comput. Chem. Eng.*, 24:525–531, 2000.

[12] R. Amirthalingam and J. H. Lee. Subspace identification based inferential control applied to a continuous pulp digester. *J. Process Contr.*, 9:397–406, 1999.

[13] B. C. Juricek, D. E. Seborg, and W. E. Larimore. Identification of the Tennessee Eastman challenge process with subspace method. *Control Eng. Pract.*, 9:1337–1351, 2001.

[14] J. M. Lee. *A study on architecture, algorithms, and applications of approximate dynamic programming based approach to optimal control.* PhD thesis, Georgia Institute of Technology, 2004.

[15] J. M. Lee, N. S. Kaisare, and J. H. Lee. Choice of approximator and design of penalty function for an approximate dynamic programming based control approach. *J. Process Contr.*, 16:135–156, 2006.

[16] W. L. Luyben. Dynamics and control of recycle system. 1. simple open-loop and closed-loop systems. *Ind. Eng. Chem. Res.*, 32:466–475, 1993.

[17] W. L. Luyben. Plantwide regulatory control design procedure using a tiered framework. *Ind. Eng. Chem. Res.*, 32:2693–2705, 1993.

[18] P. Mizsey and I. Kalmar. Effects of recycle on control of chemical processes. *Comput. Chem. Eng.*, 20:S883–S888, 1996.

[19] O. Taiwo and V. Krebs. Robust control system design for plants with recycle. *Chem. Eng. J.*, 61:1–6, 1669.

[20] J. Morud and S. Skogestad. Dynamic behavior of integrated plant. *J. Process Contr.*, 6(2):145–156, 1996.

[21] Y. Zhang and J. F. Forbes. Extended design cost: a performance criterion for real-time optimization systems. *Comput. Chem. Eng.*, 24:1829–1841, 2000.

[22] J. H. Lee, J. M. Lee, T. Tosukhowong, and J. Lu. On interfacing model predictive controllers with a real-time optimizer. In *Proc. of 2003 Proc. System Eng. Conf.*, pages 910–915, Kunming, P.R. China, 2004.

[23] T. Tosukhowong, J. M. Lee, J. H. Lee, and J. Lu. An introduction to a dynamic plantwide optimization strategy for an integrated plant. *Comput. Chem. Eng.*, 29:199–208, 2004.

[24] D. F. Enns. Model reduction with balance realization: An error bound and frequency weighted generalization. In *Proc. 23th IEEE Conf. Decision and Control*, pages 127–132, Las Vegas, NV, 1984.

[25] G. Wang, V. Sreeram, and W. Q. Liu. A new frequency-weighted balanced truncation method and an error bound. *IEEE T. Automat. Contr.*, 44(9):1734–1737, 1999.

[26] A. Varga and B. D. O. Anderson. Accuracy enhancing methods for the frequency-weighted balancing related model reduction. In *Proc. 40th IEEE Conf. Decision and Control*, page 2001, Orlando, FL, 2001.

[27] P. J. Holmes, L. Lumley, and G. Berkooz. *Turbulence, coherent structures, synamical systems and symmetry.* Cambridge University Press, New York, 1996.

[28] E. Bendersky and P. D. Christofides. Optimization of transport-reaction processes using nonlinear model reduction. *Chemical Engineering Science*, 55:4349–4366, 2000.

[29] J. Hahn and T. F. Edgar. An improved method for nonlinear model reduction using balancing of empirical gramians. *Comput. Chem. Eng.*, 26:1380, 2002.

[30] C. Sun and J. Hahn. Reduction of stable differntial-algebraic equation systems via projections and system identification. *J. Process Contr.*, 15:639–650, 2005.

[31] L. Ljung. *System identification: Theory for user.* Prentice Hall PTR, Upper Saddle River, NJ, 2nd edition, 1999.

[32] P. van Overschee and B. L. De Moor. *Subspace Identification for Linear Systems: Theory - Implementation - Applications.* Kluwer Academic Publishers, 1996.

[33] T. Tosukhowong and J. H. Lee. Plantwide dynamic optimization and control of a reactor-storage-separator network with a recycle stream. In *AIChE Annual Meeting 2003*, San Francisco, CA, 2003. paper no. 437f.

[34] T. Tosukhowong and J. H. Lee. Real-time economic optimization for an integrated plant via a dynamic optimization scheme. In *Proc. of 2004 American Control Conference*, pages 233–238, Boston MA, 2004.

[35] W. L. Luyben, B. D. Tyréus, and L. M. Luyben. *Plantwide Process Control.* McGraw-Hill, New York, 1999.

[36] J. H. Lee, M. Morari, and C. E. Garcia. State-space interpretation of model predictive control. *Automatica*, 30(4):707–717, 1994.

[37] P. Lundström, J. H. Lee, M. Morari, and S. Skogestad. Limitations of dynamic matrix control. *Comput. Chem. Eng.*, 19(4):409–421, 1995.

[38] W. E. Larimore. Statistical optimality and canonical variate analysis system identification. *Signal Processing*, 52(2):131–144, 1996.

[39] Y. C. Zhu, M. van Wijck, E. Janssen, A. J. M. Graaf, C. H. van Aalst, and L. Kieviet. Crude unit identification for MPC using ASYM method. In *Proc. of American Control Conference*, Albuquerque, New Mexico, 1997.

[40] Y. Zhu. Multivariable process identification for MPC: the asymptotic method and its applications. *J. Process Contr.*, 8(2):101–115, 1998.

[41] P. Misra and M. Nikolaou. Input design for model order determination in subspace identification. *AIChE J.*, 49(8):2124–2132, 2003.

[42] S. Skogestad and M. Morari. Understanding the dynamic behavior of distillation column. *Ind. Eng. Chem. Res.*, 27:1848–1862, 1988.

[43] C. W. Koung and J. F. MacGregor. Identification for robust multivariable control: the design of experiments. *Automatica*, 30(10):1541–1553, 1994.

[44] W. Li and J. H. Lee. Control relevant identification of ill-conditioned systems: estimation of gain directionality. *Comput. Chem. Eng.*, 20(8):1023–1042, 1996.

[45] J. Lee, W. Cho, and T. F. Edgar. Iterative identification methods for ill-conditioned processes. *Ind. Eng. Chem. Res.*, 37:1018–1023, 1998.

[46] M. J. Bruwer and J. F. MacGregor. Robust multi-variable identification: Optimal experimental design with constraints. *J. Process Contr.*, 16:581–600, 2006.

[47] M. Hovd, J. H. Lee, and M. Morari. Truncated step response models for model predictive control. *J. Process Contr.*, 3(2):67–73, 1993.

[48] A. Kumar and P. Daoutidis. Nonlinear dynamics and control of process system with recycle. *J. Process Contr.*, 12:475–484, 2002.

[49] C. J. Goh and K. L. Teo. Control parameterization: a unified approach to optimal control problems with general constraints. *Automatica*, 24:3–18, 1988.

[50] L. R. Petzold. *A description of* DASSL*: A differential/algebraic system solver*, pages 65–68. In R.S. Stepleman et al., editors, Scientific Computing. North-Holland Publishing, Amsterdam, 1983.

[51] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold. Using krylov methods in the solution of large-scale differential-algebraic systems. *SIAM J. Sci. Comp.*, 15:1467–1488, 1994.

[52] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold. Consistent initial condition calculation for differential-algebraic systems. *SIAM J. Sci. Comp.*, 19:1495–1512, 1998.

[53] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 1987.

[54] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* London, Academic Press, 1981.

[55] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer-Verlag, New York, NY, 1999.

[56] L. T. Biegler. *Efficient solution of dynamic optimization and* NMPC *problems.* In F. Allgöwer and A. Zheng, editors, Nonlinear Model Predictive Control. Birkhäuser Verlag, Basel, Switzeland, 2000.

[57] J. E. Cuthrell and L. T. Biegler. On the optimization of differential-algebraic process systems. *AIChE J.*, 33:1257–1270, 1987.

[58] L. T. Biegler, A. M. Cervantes, and A. Wächter. Advances in simultaneous strategies for dynamic process optimization. *Chem. Eng. Sci.*, 57:575–593, 2002.

[59] K. Glover. All optimal hankel-norm approximation of linear multivariable systems and their $\mathcal{L}_\infty$ error bounds. *Int. J. Control*, 39:1115–1193, 1984.

[60] Y. S. Hung and K. Glover. Optimal Hankel-norm approximation of stable system with first-order stable weighting functions. *Systems & Control Lett.*, 7:165–172, 1986.

[61] B. C. Moore. Principle component analysis in linear systems: controllability, observability, and model reduction. *IEEE T. Automat. Contr.*, 26(1):17–32, 1981.

[62] W. Marquardt. Nonlinear model reduction for optimization based control of transient chemical processes. In *Proc. CPC VI*, pages 30–60, Tucson, AZ, 2001.

[63] J. E. Bailey. Lumping analysis of reactions in continuous mixtures. *Chem. Eng. J.*, 3:52–61, 1972.

[64] G. Li, A. S. Tomlin, H. Rabitz, and J. Tóth. A general analysis of approximate nonlinear lumping in chemical kinetics. I. Unconstrained lumping. *J. Chem. Phys.*, 101(2):1172–1187, 1994.

[65] G. Li, A. S. Tomlin, H. Rabitz, and J. Tóth. A general analysis of approximate nonlinear lumping in chemical kinetics. II. Constrained lumping. *J. Chem. Phys.*, 101(2):1188–1201, 1994.

[66] N. Ganesh and L. T. Biegler. A robust technique for process flowsheet optimization using simplified model approximations. *Comput. Chem. Eng.*, 11(6):553–565, 1987.

[67] E. M. Hendriks and A. R. D. van Bergen. Application of a reduction method to phase equilibria calculations. *Fluid Phase Equilibria*, 74:17–34, 1992.

[68] J. Perregaard. Model simplification and reduction for simulation and optimization of chemical processes. *Comput. Chem. Eng.*, 17(5-6):465–483, 1993.

[69] V. Van Breusegem and G. Bastin. Reduced-order dynamical modelling of reaction systems: a singular perturbation approach. In *IEEE Proc. 30th Conf. Decision and Control*, pages 1049–1054, 1991.

[70] A. Kumar, P. D. Christofides, and P. Daoutidis. Singular perturbation modeling of nonlinear processes with non-explicit time- scale separation. *Chem. Eng. Sci.*, 53:1491–1504.

[71] N. Vora and P. Daoutidis. Nonlinear model reduction of chemical reaction systems. *AIChE J.*, 47(10):2320–2332, 2001.

[72] S. Lall, J. E. Marsden, and S. Glavaski. Empirical model reduction of controlled nonlinear systems. In *Proc. of the 14th IFAC World Congress*, volume F, pages 473–478, 1999.

[73] S. Lall, J. E. Marsden, and S. Glavaski. A subspace approach to balanced trunation for model reduction of nonlinear control systems. *Int. J. Robust Nonlinear Control*, 2:519–535, 2002.

[74] J. Hahn, T. F. Edgar, and W. Marquardt. Controllability and observability covariance matrices for the analysis and order reduction of stable nonlinear systems. *J. Process Contr.*, 13:115–127, 2003.

[75] J. van den Berg. *Model reduction for dynamic real-time optimization of chemical processes.* PhD thesis, Delft University of Technology, The Netherlands, 2005.

[76] J. Baker and P. D. Christofides. Finite-dimensional approximation and control of nonlinear parabolic PDE systems. *Int. J. Contr.*, 73(5):439–456, 2000.

[77] S. Y. Shvartsman, C. Theodoropoulos, R. Rico-Martnez, I. G. Kevrekidis, E. S. Titi, and T. J. Mountziaris. Order reduction for nonlinear dynamic models of distributed reacting systems. *J. Process Contr.*, 10:177–184, 2000.

[78] G. K. Venayagamoorthy and R. G. Harley. A continually online trained neuro-controller for excitation and turbine control of a turbogenerator. *IEEE T. ENERGY. CONVER.*, 16(3):261–269, 2001.

[79] R. Enns. *Neural Dynamic Programming applied to rotocraft flight control and reconfiguration.* PhD thesis, Arizona State University, Tempe, AZ, 2001.

[80] W. Zhang. *Reinforcement Learning for job-shop scheduling.* PhD thesis, Oregon State University, 1996.

[81] R. Neuneier. Enhancing Q-learning for optimal asset allocation. In *Proc. 1997 Conf. Advances in Neural Information Processing Systems 10*, pages 936–942, Denver, Co, 1998.

[82] P. Werbos. *ADP:Goals, Opportunities and Principls.* In J. Si, A. G. Barto, W. B. Powell, D. Wunsch, editors, Handbook of Learning and Approximate Dynamic Programming. IEEE Press, Piscataway, NJ, 2004.

[83] N. S. Kaisare, J. M. Lee, and J. H. Lee. Simulation based strategy for nonlinear optimal control: Application to a microbial cell reactor. *Internat. J. Robust Nonlinear Control*, 13(3-4):347–363, 2002.

[84] D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 2001.

[85] M. L. Putterman. *Markov Decision Processes: Discrete stochastic dynamic programming.* John Wiley & Sons, 1994.

[86] R. Bellman. *Dynamic Programming.* Dover Publications, 2003.

[87] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, 1998.

[88] D. P. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, 1996.

[89] D. P. Bertsekas. Distributed dynamic programming. *IEEE T. Automat. Contr.*, 27:610–616, 1982.

[90] D. P. Bertsekas. Distributed asynchronous computation of fixed points. *Mathematics Programming*, 27:107–120, 1983.

[91] V. Gullapalli and A. G. Barto. Convergence of indirect adaptive asynchronous value iteration algorithms. In *Advances in Neural Information Processing Systems 6*, pages 695–702, San Mateo, CA, 1994.

[92] D. Wingate and K. D. Seppi. Prioritized method for accelerating MDP solvers. *J. Mach. Learn. Res.*, 6:851–881, 2005.

[93] J. M. Lee and J. H. Lee. Approximate dynamic programming strategies and their applicability for process control: A review and future directions. *Int. J. Control Autom.*, 2(3):263–278, 2004.

[94] A. W. Moore and C. G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Mach. Learn.*, 13:103–130, 1993.

[95] R. Munos and A. W. Moore. Variable resolution discretization in optimal control. *Mach. Learn.*, 49:291–323, 2002.

[96] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch II, editors. *Handbook of Learning and Approximate Dynamic Programming.* IEEE Press, Piscataway, NJ, 2004.

[97] S. Wold, A. Ruhe, H. Wold, and W. J. Dunn III. The colinearity problem in linear regression. the partial least square approach to generalized inverses. *SIAM J. Sci. Stat. Compute.*, 5:735–743, 1994.

[98] T. Jockenhvel, L. T. Biegler, and A. Wächter. Dynamic optimization of the Tennessee Eastman process using the OptControlCentre. *Comput. Chem. Eng.*, 27(11):1513–1531, 2003.

VITA

Thidarat Tosukhowong was born in Bangkok, Thailand in 1979. She attended Chulalongkorn University in Bangkok, Thailand in 1996. During Fall 1998 to Spring 1999, she was a visiting student at University of Washington in Seattle, WA. Then, she worked for Dow Chemical Company as an engineering intern in Midland, MI. In October 2000, she completed here B.S. in Chemical Engineering from Chulalongkorn University with Highest Honor. From November 2000 to July 2001 she worked for Bayer AG in Leverkusen, Germany as an engineering intern. She joined Georgia Institute of Technology in August 2001. During her Ph.D. education, she had opportunities to work for Weyerhaeuser Company as an intern in the Process Information, Diagnostics & Control team in Federal Way, WA during summers of 2002, 2004, and 2005. Her dissertation title was "Dynamic Real-time Optimization and Control of an Integrated Plant." She defended her thesis on August 18, 2006 and obtained her Ph.D. in Chemical and Biomolecular Engineering and a Certificate in Management of Technology on December 16, 2006.