OPTIMIZATION-BASED SOLUTIONS FOR PLANNING AND CONTROL

Optimization-based Solutions to Optimal Operation under Uncertainty and Disturbance Rejection

By Mahir Jalanko

Bachelor of Chemical Engineering and Management

A Thesis Submitted to the School of Graduate Studies in Partial Fulfillment of the Requirement for the Degree Doctor of Philosophy

Ph.D. Candidate (2021) McMaster University

Chemical Engineering Hamilton, Ontario

TITLE:	Optimization-based Solutions to Optimal Operations under Uncertainty and Disturbance Rejection
AUTHOR:	Mahir Jalanko
	Bachelor of Chemical Engineering and Management (McMaster University)
SUPERVISOR:	Professor Vladimir Mahalec
COSUPERVISOR:	Professor Prashant Mhaskar
NUMBER OF PAGES:	xi, 210

Lay Abstract

The operation of a chemical process involves many decisions which are normally distributed into levels referred to as process automation hierarchy. The process automation hierarchy levels are planning, scheduling, real-time optimization, and control. This thesis addresses two of the levels in the process automation hierarchy, which are planning and control. At the planning level, the objective is to ensure optimal utilization of raw materials and equipment to reduce production cost. At the control level, the objective is to meet and follow process setpoints determined by the real-time optimization level.

The main goals of the thesis are: (1) develop an efficient algorithm to solve a large-scale planning problem that incorporates uncertainties in components qualities and products demands to reduce the production cost and maximize profit for gasoline blending application. (2) Develop a novel hybrid-based model predictive control to improve the control strategy of an industrial distillation column that faces flooding issues.

Abstract

Industrial automation systems normally consist of four different hierarchy levels: planning, scheduling, real-time optimization, and control. At the planning level, the goal is to compute an optimal production plan that minimizes the production cost while meeting process constraints. The planning model is typically formulated as a mixed integer nonlinear programming (MINLP), which is hard to solve to global optimality due to nonconvexity and large dimensionality attributes. Uncertainty in component qualities in gasoline blending due to measurement errors and variation in upstream processes may lead to off-specification products which require re-blending. Uncertainty in product demands may lead to a suboptimal solution and fail in capturing some potential profit due to shortage in products supply. While incorporating process uncertainties is essential to reducing the production cost and increasing profitability, it comes with the disadvantage of increasing the complexity of the MINLP planning model. The key contribution in the planning level is to employ the inventory pinch decomposition method to consider uncertainty in components qualities and products demands to reduce the production cost and increase profitability of the gasoline blend application.

At the control level, the goal is to ensure desired operation conditions by meeting process setpoints, ensure process safety, and avoid process failures. Model predictive control (MPC) is an advanced control strategy that utilizes a dynamic model of the process to predict future process dynamic behavior over a time horizon. The effectiveness of the MPC relies heavily on the availability of a reasonably accurate process model. The key contributions in the control level are: (1) investigate the use of different system identification methods for the purpose of developing a dynamic model for high-purity distillation column, which is a highly nonlinear process. (2) Develop a novel hybrid based MPC to improve the control of the column and achieve flooding-free control.

Preface

Chapters 2-4 contain multi-authored work published in peer-reviewed scientific journals. Chapter 5 contains a manuscript submitted for publication in peer-reviewed scientific journal. My individual contributions to each of these chapters include:

- Implement the mathematical models in GAMS.
- Develop the steps of the solution algorithms.
- Implement the algorithms using GAMS, Python, MATLAB, and VBA.
- Run the case studies and gather numerical results.
- Analyze the numerical results.
- Write the initial draft and the final draft of each manuscript.

Contributions from Dr. Vladimir Mahalec in chapters 2-5 include:

- Provide research direction and insightful discussion about the applications of gasoline blend and distillation column, as well as discussion related to planning under uncertainty, different control strategies, potential solution strategies, and analysis of the numerical results.
- Proofread and edit each manuscript and approve their final draft.

Contributions from Dr. Prashant Mhaskar in chapters 4-5 include:

- Provide research direction and insightful discussion about different control strategies, potential solution strategies, and analysis of the numerical results.
- Proofread and edit each manuscript and approve their final draft.

Acknowledgements

I would like to start by thanking my supervisor, Dr. Vladimir Mahalec, for giving me the opportunity to work under his supervision, and for all his support, guidance, and constant encouragement during the last five years. Dr. Mahalec is not only my supervisor, but also my mentor and role model. He is very enthusiastic about his research and always looking for new opportunities in different research areas. Also, he is always supportive and caring for his students. I was extremely fortunate to have pursued my graduate studies under his supervision, and I cannot imagine having a better mentor than him.

Next, I would like to thank my co-supervisor, Dr. Prashant Mhaskar, for all his support and guidance during the last two and a half years of my studies. Dr. Mhaskar is brilliant, highly knowledgeable, and very enthusiastic about his research field. He really cares for his students and always makes himself available to discuss ideas or answer questions. I was extremely fortunate to work under Dr. Mhaskar's supervision and I learned a lot from him.

I would like to thank Dr. Ashtiani from the department of computing and software for accepting to be in my supervisory committee, and for all his advice, questions, and suggestions during our committee meetings. Also, I would like to thank Yoel Sanchez from NOVA Chemicals Inc. for providing industrial data and insights on their ethylene plant. Further, I would like to express my gratitude to Michelle Whalen, Kristina Trollip, Linda Ellis, and Lynn Falkiner for all the administrative help during my studies.

I would like to thank my fellow MACC friends for their support and valuable discussions during my studies. It was a pleasure to be part of such a brilliant group of people. Also, I would like to thank my friends outside school for always being supportive and encouraging during difficult moments.

Last but not least, I would like to thank my family, especially my parents, Nadia and Dhia. Thank you for always giving me love, support, and guidance throughout my life. Thank you to my siblings, Namir, Sarah, and Meriam, for always supporting me. Most importantly, I would like to express my love and gratitude to my fiancée, Aula, for her endless support and encouragement.

Table of Contents

List of Abbreviations

ANN	Artificial neural network
ARO	Aromatic content
BEN	Benzene content
BPTT	Back-propagation through time
CATP	Cumulative average total production
CTD	Cumulative total demand
GC	Gas chromatography
LP	Linear programming
LSTM	Long short-term memory
MILP	Mixed integer linear programming
MINLP	Mixed integer nonlinear programming
MON	Motor octane number
MPC	Model predictive control
NARX	Nonlinear autoregressive with exogenous inputs
NLP	Nonlinear programming
NMPC	Nonlinear model predictive control
OF-NMPC	Offset-free nonlinear model predictive control
OLF	Olefin content
PCA	Principal component analysis
PID	Proportional integral derivative
PRBS	Pseudo random binary sequence
RGS	Random gaussian signal
RNN	Recurrent neural network

RON	Research octane number
RTO	Real time optimization
RVP	Rapid vapor pressure
SGO	Specific gravity
SI	Sulfur content
SI	Subspace identification
WMSE	Weighted mean squared error

Declaration of Academic Achievement

I, Mahir Jalanko, declare this thesis to be my own work with the guidance of my supervisors: Dr. Vladimir Mahalec, and Dr. Prashant Mhaskar. I am the sole author of this document with exception to those chapters included as works published or submitted for publication in research journal in which case authorship, credit, and copyright are noted with respect to each such included items.

Chapter 1: Introduction

In this chapter, the overall context and objectives of the thesis are presented. First, it introduces the process automation hierarchy with its levels, placing more emphasis on the levels related to the thesis. Then, the objectives of the thesis are highlighted. Finally, the thesis outline is presented.

1.1. Process Automation Hierarchy

Process automation is used in chemical engineering processes to maximize the production and economic profit while maintaining a desired level of product quality and safety of the process. These goals apply to a variety of chemical engineering industries such as refineries, production of chemicals, metals, and pharmaceuticals. The principles of automatic control are generic in nature and can be applied to different industries, regardless of their method of production or the size of the plant. [1]

The process automation activities can be organized in the form of a hierarchy as presented in Figure 1 with the frequency of execution. At higher levels, decisions are made based on the objectives of the plant which are typically based on economic objectives. On the other hand, decisions at lower levels are made based on the requirements needed to achieve these economic objectives and constraints associated with the process. The frequency of execution at higher levels are much lower compared to the frequency of execution at lower levels.

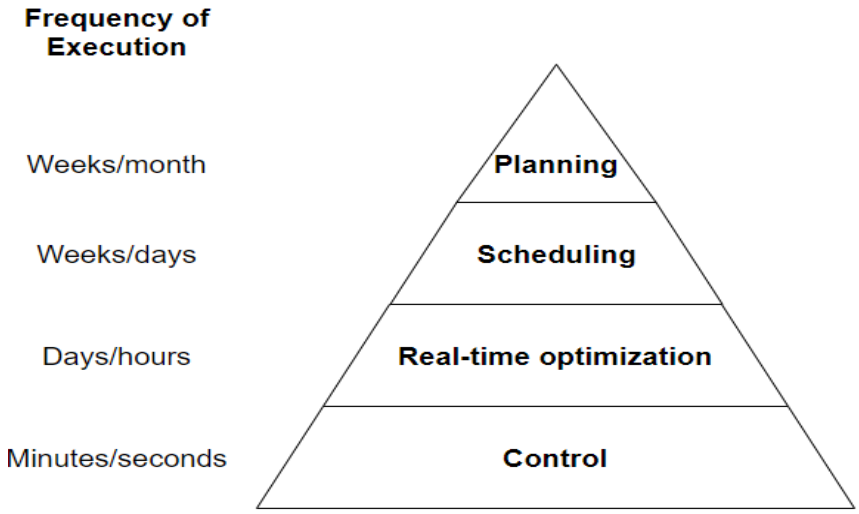

Figure 1. Process automation hierarchy

The highest level of the process automation hierarchy is the planning level, which focuses on economic forecasting and providing production goals. The goal at this level is the optimization of the production plan of the plant to maximize the profit by utilizing raw materials, storage capacity, and production equipment in the best way possible along a given time horizon, while considering current market conditions and forecasts. The time horizon in the planning level is typically in months or weeks. The second level of the process automation hierarchy is the scheduling level, which focuses on the time of actions and events necessary to execute the chosen plan. The goal at this level is to use the production plan obtained from the planning level and arrange the manufacturing sequence, while ensuring the feasibility of the chosen plan. The time horizon in the scheduling level is typically weeks or days. The third level of the process automation hierarchy is the real time optimization (RTO), which uses a steady-state model of the plant and the schedule obtained from the scheduling level to minimize the operating cost or maximize the operating profit. The goal of the RTO level is to provide the next level with the desired setpoints of the key process variables such as products purities. The time horizon in the RTO level is typically days to hours. The last level in the process automation hierarchy is the control level, which focuses on maintaining the setpoints of the key process variables obtained from the RTO level. The control level can be subdivided into two levels: advanced control and basic regulatory control. The advanced control typically uses model predictive control (MPC) to handle interactions between key process variables and to incorporate process constraints, while ensuring the process is operating optimally. The basic regulatory control typically uses standard feedback controllers such as P, PI or PID controllers to maintain the setpoints of the key process variables and to reject short-term disturbances in the process. The time horizon of the control level is typically minutes to seconds. [1] [2]

The current trend in the planning optimization is to increase the accuracy of the mathematical models employed to represent processing units and operational policies, as well as to account for process and demand uncertainties in the mathematical model to increase plant profitability. While increasing the complexity of the mathematical model and incorporating process uncertainty can increase plant profitability, such models might be difficult to optimize using commercial solvers. Therefore, another research trend is focused on the development of advanced algorithms to efficiently solve complex mathematical models to optimality. Production planning optimization models typically require the incorporation of discrete decisions into the

optimization model. Also, the production planning model typically requires both linear and nonlinear functions to accurately model different phenomenon, such as chemical separations, chemical reactions, and material flow through a production facility. Therefore, accurate production planning models require using mixed integer nonlinear programming (MINLP) models. MINLP models are considered a difficult class of optimization problems due to the following reasons: (1) potential need of a large number of partitions to represent the time domain, which can result in a model containing thousands of variables. (2) Possible nonconvexity, which can introduce multiple local and global optimal points. [3] [4]

Some of the current trends in the process control of chemical engineering application have focused on the use of MPC due to its capability to handle multivariable constrained control problems. The effectiveness of the MPC relies heavily on the availability of a reasonably accurate process model. Most industrial implementation of MPC technologies have focused on linear systems, and limited works have considered nonlinear systems. Two challenges arise from the implementation of nonlinear MPC (NMPC) to industrial application: (1) it is difficult, expensive, and time-consuming to obtain an exact nonlinear process model. (2) For realistic nonlinear processes, the numerically nonlinear constrained optimal control problem in NMPC can be hard to solve to optimality in a reasonable time. Therefore, while the topic of NMPC has been a very active area in academic research, it is still at the early stage in industrial practices. [5]

1.2. Objectives of the Thesis

The focus of the thesis is to improve the process automation of chemical engineering applications at the planning and control levels. The main contributions of the thesis on the process automation hierarchy are highlighted in Figure 2. At the planning level, we consider uncertainties in the production plan of gasoline blend application to improve the profitability of the plant and utilize the supply-demand pinch algorithm to solve the optimization problem efficiently. More specifically:

1. Consider uncertainty in components qualities and their impact on the product specification to avoid making off-specification products and to reduce production cost. Also, utilize the supply-demand pinch algorithm to efficiently solve the complex MINLP optimization problem.

2. Consider uncertainty in the products demands to increase the profitability of the plant by capturing uncertain demands using rolling horizon optimization framework. Also, adapt the supply-demand pinch algorithm within the rolling horizon optimization approach to efficiently solve the complex MINLP optimization problem.

At the control level, we focus on improving the control strategy of an industrial distillation column which faces flooding issues. The goal is to replace the current control strategy which uses a basic regulatory control with an advanced control strategy to improve process control of the key variables and avoid tower flooding. The contributions at the control level are:

(1) Compare different system identification methods based on linear subspace identification and nonlinear artificial neural network for the purpose of developing a dynamic model for advanced control.

(2) Develop a novel hybrid model based NMPC for the purpose of achieving free flooding control for the industrial distillation column.

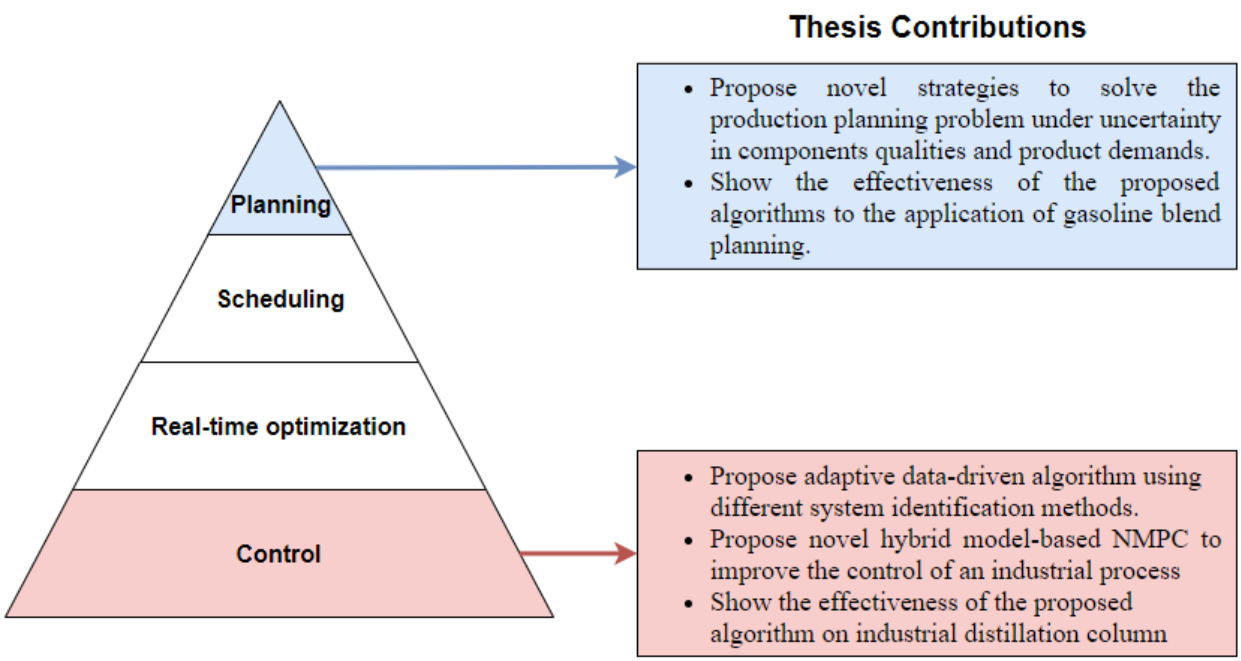

Figure 2. Thesis contribution on the process automation hierarchy

1.3. Thesis Outline

Chapter 1: Introduction. This chapter provides the overall context and the outline of the thesis, as well as the objectives of the research.

Chapter 2: "Supply-Demand Pinch based Methodology for Multi-Period Planning under Uncertainty in Components Qualities with Application to Gasoline Blend Planning". This chapter considers the problem of production planning under uncertainty in components qualities for gasoline blending application. First, it presents details about the supply-demand pinch concept for production planning used to efficiently optimize the production planning model. Then, it describes the mathematical model for the gasoline blend planning problem under uncertainty in components qualities. Finally, it shows the capability of the supply-demand pinch algorithm to efficiently solve the production planning problem at hand. This work has been published in the *Computer & Chemical Engineering Journal*.

Chapter 3: "Gasoline Blend Planning under Demand Uncertainty: Aggregate Supply-Demand Pinch Algorithm with Rolling Horizon". This chapter considers the problem of production planning under uncertainty in products demands. First, it presents details about the gasoline blend planning under uncertainty in product demands, and the concept of rolling horizon utilized to consider time-varying uncertainty in products demands. Then, it describes the employment of the supply-demand pinch algorithm within the rolling horizon formulation for the purpose of efficiently solving the production plan under uncertainty in products demands. Finally, it shows the capability of the supply-demand pinch algorithm to efficiently solve the production planning under uncertainty in products demands. This work has been published in the *Industrial & Engineering Chemistry Research Journal*.

Chapter 4: "Adaptive System Identification of Industrial Ethylene Splitter: A comparison of Subspace Identification and Artificial Neural Networks". This chapter considers different data-driven modeling approaches for control purposes of an industrial ethylene splitter. First, it describes the details of the industrial ethylene splitter considered in this work. Then, it presents the simulation model developed in Aspen Dynamics that replicates industrial operation to be used as a test bed for control purposes. Also, the details of the three different system identification methods used and compared are presented. Finally, two adaptive strategies have been proposed to allow more recent data to be incorporated into the training approaches. The two adaptive strategies show great improvement in the model prediction capabilities for all three system identification methods. This work has been published in the *Computer & Chemical Engineering Journal*.

Chapter 5: "Flooding and Offset-Free Nonlinear Model Predictive Control of a High-Purity Industrial Ethylene Splitter Using a Hybrid Model". This chapter considers the control problem of the industrial ethylene splitter which faces flooding issues. First, it describes the current control strategy of the industrial ethylene splitter. Then, it introduces a novel hybrid model based NMPC that utilizes a data-driven model for predicting dynamics, and a first principles steady state model to capture tower flooding. Finally, it shows the effectiveness of the hybrid model based NMPC in controlling the key process variables and avoiding tower flooding when compared to the current control strategy. This work has been submitted for publication in the *Computer & Chemical Engineering Journal*.

Chapter 6: Concluding Remarks. This chapter describes the major contributions, key findings, and future work direction of the thesis.

1.4. References

[1] Edgar, T. F., & Hahn, J. (2009). Process Automation. In *Springer Handbook of Automation* (pp. 529-543). Springer, Berlin, Heidelberg.

[2] Huang, R. (2010). Nonlinear model predictive control and dynamic real time optimization for large-scale processes.

[3] Castillo Castillo, P. A. (2020). *PLANNING AND SCHEDULING OF CONTINUOUS PROCESSES VIA INVENTORY PINCH DECOMPOSITION AND GLOBAL OPTIMIZATION ALGORITHMS* (Doctoral dissertation).

[4] Kronqvist, J., Bernal, D. E., Lundell, A., & Grossmann, I. E. (2019). A review and comparison of solvers for convex MINLP. *Optimization and Engineering*, *20*(2), 397-455.

[5] Xi, Y. G., Li, D. W., & Lin, S. (2013). Model predictive control-status and challenges. *Acta automatica sinica*, *39*(3), 222-236.

Chapter 2: Supply-demand Pinch based Methodology for Multi-period Planning under Uncertainty in Components Qualities with Application to Gasoline Blend Planning

The contents of this chapter have been published in the *Computer & Chemical Engineering Journal*. Reprinted with permission from

Jalanko, M., & Mahalec, V. (2018). Supply-demand pinch based methodology for multi-period planning under uncertainty in components qualities with application to gasoline blend planning. *Computers & Chemical Engineering*, *119*, 425-438.

Abstract

Uncertainty in component quality in gasoline blending due to measurement errors and variation in operation leads to planned blends which may not meet quality specifications and re-blending is required. Formulating gasoline blending as chance constrained programming enables a decision maker to decide what percentage of blends will be guaranteed to meet the specifications and balance the increased cost of blends vs. the cost of having to re-blend the off-spec blends. Chance constrained formulation makes the gasoline blend problem nonlinear and nonconvex. In this work, we employ a supply-demand pinch-based algorithm to optimize gasoline blend planning with uncertainty in components qualities and examine its performance vs. full-space model. The supply-demand pinch algorithm decomposes the problem into two sub-problems, top-level (NLP) computes optimal blend recipes and the bottom-level (MILP) computes an optimal production plan using the recipes computed at the top-level. Computational efficiency of the algorithm is verified by case studies.

Keywords:

Production planning under uncertainty in raw material qualities; Gasoline blending planning with uncertainty in component qualities; Supply-demand pinch; Inventory pinch; Joint Chance constrained programming.

2.1. Introduction

Refining plants are a key element in the supply chain of the petroleum industry. In the product blending section of the refinery, feedstocks components coming from various upstream processes are mixed together to produce different products that meet varying specification. Gasoline products contribute 60%-70% of the total revenue of refineries, therefore decreasing the gasoline blending costs can greatly impact the profitability of the refinery (Biegler, 2010; Wei Wang, Zefei Li, Qiang Zhang, & Yankai Li, 2007). The gasoline blend planning problem requires computation of optimal blend recipes and production plan for the planning horizon that minimize operation cost while meeting products demands, products qualities specifications, and constrained by components supply rates and capacity limitations on the tanks and the blenders. The optimal blend recipes associated with the minimal blending cost are highly dependent on the component's qualities. Component's qualities are not known accurately at the time the plan is made due to two factors: (i) changes throughout the planning horizon due to operating conditions changes in the upstream processes, and (ii) measuring instruments have inherent measurement errors. Ignoring uncertainties and modelling components qualities as nominal parameters might lead to produce large volume of products that are off specs. Off specifications products requires re-blending which increases operating costs and might affect the schedule and delivery of the orders (Wei Wang et al., 2007). In practice, addressing the uncertainty in components qualities can be highly beneficial for existing oil refineries that operates without an on-line blending analyzer tracking components qualities when blending gasoline products. Since components qualities are not being measured, variations in quality values will result in producing high levels of off-spec products which adds additional blending cost and delay in their production plan. Also, even oil refineries that use on-line analyzer to measure octane number (RON and MON) need to consider uncertainty in the component's qualities due to the inherent error in on-line analyzers. The ZX101™ is an octane analyzer from Zeltex, Inc. that measures octane number via near-infrared transmission spectroscopy. The analyzer accuracy has ± 0.7 reproducibility and ± 0.2 repeatability with 95% confidence for RON, and it has ± 0.9 reproducibility and ± 0.3 repeatability with 95% confidence for MON (Merberg, n.d.). ASTM D2699-18 standard test method for measuring RON value has repeatability and reproducibility to exceed 0.2 and 0.7 respectively for one case in twenty when tests are taken under normal and correct operation. ASTM D2700-18 standard for MON has repeatability and reproducibility of exceeding 0.2 and

0.9 respectively for one case in twenty when tests are taken under normal and correct operation (ASTM international, 2018). Therefore, taking into consideration uncertainty in components qualities is required to reduce the number of off-spec blends even when on-line analyzers are used. One approach to avoid having off specs products is to add an offset that acts as safety margin on the qualities with uncertainties. This approach faces challenging in determining the value of the offset for each quality; large offset values guarantee production of on-specs products with high probability, but it leads to suboptimal solutions due to large quality giveaway. Therefore, it is essential to determine a multi-period production plan model that considers uncertainties in components qualities to compute optimal blend recipes associated with minimal blending cost which balances between infeasibility and sub optimality. Chance constrained formulation is one way to minimize blend cost while controlling the percentage of the off-specs blend. Blended products are considered on-specs only if all their qualities are within the required range, otherwise the product is considered off-specs. Therefore, to ensure that all uncertain feedstocks qualities are within the required range, joint probability constraint formulation is required which increases the complexity of the model.

One approach to solve the gasoline blending problem is to simultaneously compute the blend recipes and the blended volume of the products at each period. Such approach leads to large MINLP problems which lead to difficulties in computing the optimal solution, especially when dealing with nonlinear blending properties. Therefore, such approach has been limited to use approximate linear formulation. Li and coworkers developed a slot-based MILP formulation to solve the gasoline blend problem as an MILP instead of MINLP using blending indices and linear blending correlation to address nonlinear blending properties (Li, Karimi, & Srinivasan, 2009). Li and Karimi then improved their formulation efficiency and decreased execution time by using unit-slots instead of process-slots (Li & Karimi, 2011). Cerda and coworkers developed an MINLP continuous time formulation to solve the gasoline blend scheduling problem (Cerdá, Pautasso, & Cafaro, 2016). Their formulation is based on floating slots where the slots are dynamically allocated to the periods while solving the problem and the MINLP model was approximated as an MILP model derived from assuming ideal mixing. Their problem was solving using two strategies; the first uses MILP-MINLP solution strategy which gives the optimal solution, the second finds a near optimal-solution which uses a MILP-NLP solution strategy where the integer variables of the MINLP model are fixed.

Another approach to solve the gasoline blending problem is to decompose it into planning and scheduling. Glismann and Gruhn presented an integrated optimization model of planning and scheduling (Glismann & Gruhn, 2001). At the long-range planning level, a nonlinear optimization problem has been solved to maximize profit and the results of this level are the optimum production volumes and blend recipes. At the short-term scheduling level, a mixed integer linear optimization problem has been solved and the objective is to meet the goals set by the long-term planning level. Jia and Ierapetritou presented an MILP model for gasoline blending and distribution scheduling with the assumption of having fixed preferred blend recipe (Jia & Ierapetritou, 2003). This allowed them to avoid the complexity of the MINLP model wile opening the possibility that their solution is not optimal. The drawback of such approach is that the blend recipe chosen is not necessarily an optimum blend recipe. Méndez and his coworkers presented an integrated MILP model to simultaneously optimize gasoline blending and scheduling problem based on discrete or continuous time domain (Méndez, Grossmann, Harjunkoski, & Kaboré, 2006). The method dealt with the nonlinearity in some products blending properties and variables recipes via LP or MILP iterative procedure. Also, to ensure feasibility in some circumstances and minimize the deviation from preferred recipe a penalty coefficient for preferred recipe deviation were introduced.

Castillo and Mahalec introduced the concept of supply-demand pinch points in production planning by decomposing the MINLP planning problem into two sub problems (NLP and MILP) which are solved in a sequence (Castillo & Mahalec, 2014a). At the top-level, the number of periods is delimited by pinch points and the optimal production modes (blend recipes) are computed. It should be noted that the number of periods in this top-level formulation is typically much lower than the number of periods in a typical multi-period planning model. The lower level uses much finer time grid to define the periods; it computes products volumes blended during each of the second level periods using the blend recipes computed at the top level. They extended their formulation to include a third-level to compute detailed schedule (Castillo & Mahalec, 2014b).

Two methods have been commonly used to model uncertainty in optimization planning problems: robust optimization and stochastic optimization. Robust optimization is mainly used when the uncertain parameters are bounded, symmetric, and no information known about the probability distribution of the uncertain parameters exist. Robust optimization is more

conservative due to the lack of knowledge about the distribution of the uncertain parameter. Stochastic optimization is mainly used when uncertain parameters follow a known probability distribution to obtain a less conservative solution by benefiting from such information (Yang & Barton, 2016).

Monder incorporated uncertainty in gasoline blend optimization using probabilistic programming to deal with uncertain parameters existing in multiple constraints assuming single probability constraint (Monder, 2001). Monder designed a real-time optimization layer to deal with the disturbance (uncertainty) in the feedstock qualities for linear and nonlinear blending rules. Zhang and coworkers improved the previous results by dealing with uncertain parameters in multiple constraints as a joint probability constraint (Zhang, Monder, & Forbes, 2002). Wang and coworkers introduced a new on-line gasoline blending system which solved an off-line optimization problem first to give a starting blend recipe and then used an online optimization model which incorporates uncertainty in feedstocks qualities to obtain real-time blending recipes (Wang et al., 2007). In their work, they utilized a hybrid intelligent algorithm based on Neural Network and Genetic Algorithm to overcome the difficulty of solving the chance constraint problem. Zhan and Wang solved the gasoline blending problem assuming nonlinear blending rules using Particle Swarm Optimization algorithm (Zhao & Wang, 2009). Yang and Barton proposed a near-global optimization approach which utilizes chance-constrained formulation to so solve a simple gasoline blend planning problem under uncertainty in components qualities for a single period (Yang & Barton, 2016).

This work deals with large scale multi-period gasoline planning problem (off-line blend recipe optimization over extended time horizon). The uncertainty in components qualities are dealt with using chance constraint programming similarly to the formulation used by (Yang & Barton, 2016), but our model considers large scale with multi-periods problem and shows how fixing production plan based on aggregate demands for each pinch can improves the computations time greatly. The large scale problem with multi-periods is solved using two-level algorithm that utilizes demand pinch concept introduced by (Castillo & Mahalec, 2014a) to solve the blend planning problem when uncertainty in multiple components qualities exist and compare it to solving the problem as a single level full space model. The model introduced by (Castillo & Mahalec, 2014a) requires contents components qualities along the planning horizon which is not realistic since variation in the conditions of upstream processes lead to small variations in the

qualities of the feedstocks components. Our model can handle the small variation in component qualities the planning horizon.

At the top-level of the two-level algorithm, the supply-demand pinch points are used to delineate the periods where optimal operating states are likely to remain constant. At this level both storage tanks and parallel process units are aggregated as inventory pools and single processing unit respectively. Nonlinear programming problem (NLP) at this level includes chance constrained programming formulation to account for uncertainties in the component qualities. Its solution determines optimal blend recipes for each grade of the gasoline products while minimizing the total cost of blending. At the bottom-level, an integer linear programming problem (MILP) is solved to determine a detailed production plan (how much to produce in each unit in each period) based on the blending recipes calculated in the top level. At this level the periods are of fixed lengths which are defined by the planner. If the blend recipes calculated at the first level result in inventory infeasibilities at the bottom level, then the top-level period corresponding to the point of infeasibility is subdivided and the blend recipe at top-level is re-optimized.

The remainder of this paper is organized as follows. The gasoline blend planning problem addressed in this work is presented in part 2, including detailed description and assumptions. The solution approach for the two-level algorithm, the supply-demand pinch point concept, and the chance constrained concepts used in this work are described in detail in part 3. The mathematical formulations for the chance constrained, the two-level gasoline blend planning model, and the single-level full space model that deal with component qualities uncertainty are introduced in part 4. Different case studies used to test our model are introduced in part 5. Part 6 contains the results of the case studies and the discussion. Part 7 contains the summary, conclusions, and suggestions for future work.

2.2. Problem Statement

Figure 1 shows a sample gasoline blending system. The system studied in this paper has seven feedstocks' components produced by the upstream processes; they are stored in individual component tanks. These components are sent to blenders to produce three different products, subject to meeting their quality specifications. Blended products are sent to six storage tanks before being shipped to satisfy the required demands. Three of the products storage tanks are

dedicated to the three products, while the other three tanks can store any product. The qualities monitored are aromatic content (ARO), benzene content (BEN), olefin content (OLF), research octane number (RON), motor octane number (MON), rapid vapor pressure (RVP), sulfur content (SI) which assumed to blend linearly on a volume basis, while Specific gravity (SGI) is assumed to blend linearly on a weight basis. The feedstocks components are assumed to have uncertainty in their RON, MON, and BEN qualities, while the other qualities are assumed to be accurately known.

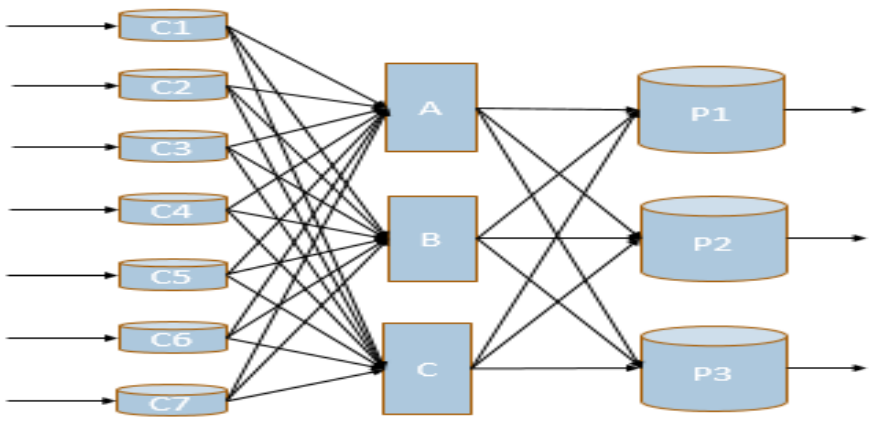

Figure 1. Sample of Gasoline Blending System

The blend planning problem addressed in this paper is described as follows:

Given:

(1) A predefined short-term planning horizon (0, H) that is divided into fixed durations time periods 1, 2… N.

(2) A set of blend components, initial inventories, cost, supply flow rates, and properties including the uncertainty along the planning horizon.

(3) A set of products, initial inventories, and their maximum and minimum quality specifications.

(4) A set of demands for each product along the planning horizon.

(5) A set of components and product storage tanks with their minimum and maximum hold ups.

(6) A set of blenders' maximum and minimum blending capacity.

Assumptions are:

(1) Refinery production plan model has determined components volumetric flow rate that are capable of meeting products demands which meet quality specifications. In addition, the volumetric flow rate of each component is piecewise constant.

(2) Component's qualities uncertainty are modelled as normal distribution along the planning horizon.

(3) Perfect mixing occurs in the blenders.

(4) Each order involved only one product and all orders are completed during the blending period.

(5) Products demands are hard constraints; should be met.

(6) Component and product tanks may receive and feed simultaneously.

The objective of the algorithm is:

Obtain optimal blend recipes for all products along the planning horizon when some of the feedstock's qualities have uncertainty. Optimal blend recipes correspond to the recipes that minimize the blending cost while meeting the probability of meeting the products specs required by the decision maker.

We need to compute:

(1) Volumes of each grade of gasoline produced at each period.

(2) Volumes of each component used to blend each gasoline grade product and which blender will carry out the blending process in each period.

(3) Inventory profiles for the components and gasoline grade products.

Subject to constraints on:

(1) Minimum and maximum volume of components and of products in the tanks.

(2) Minimum and maximum products qualities specifications.

(3) Minimum and maximum blending capacity (including the idle time required to switch a blender from one service to another)

(4) Maximum delivery rate from blenders to product tanks.

2.3. Solution Approach

A full space MINLP model can be used to solve the multi-period gasoline blend planning problem with uncertainty in components qualities for fixed discretized periods along the planning horizon. Such model faces computations difficulties obtaining the optimal solution or closing optimality gap, especially for large scale cases, due to the nonlinearities in the blend recipe calculation stemming from the chance constraint formulation.

2.3.1. Supply-demand Pinch Concept

Utilizing supply-demand pinch concept, aggregation and disaggregation techniques allow us to compute an optimal blend recipe in shorter executions times by decomposing the problem into two smaller sub-problems. The supply-demand pinch points can be identified by constructing a cumulative total demand (CTD) curve which is obtained by adding the cumulative demands of all products along the planning horizon. Then a cumulative total average production (CATP) curve is constructed as a straight line with a starting point representing initial inventory and extending to the point where it is tangent to the CTD curve. The point where CATP and CTD curves intersect is called the supply-demand pinch point. At the supply-demand pinch points, the slope of the CATP curve changes as shown in Figure 2. Hypothesis is that the optimal blend recipe for a time period between the supply-demand pinch points is constant; this indeed is the case for linear systems (Castillo & Mahalec, 2014a), while for nonlinear models this leads to solutions which are very close to the global optimum (Castillo & Mahalec, 2014a). Aggregate blend between the two pinch points is the lowest possible cost blend since it assumes that the blend components will be available just in time when they are required to produce the required amount of products to meet the demands at a given point in time between the pinch points. If the recipe computed from aggregate solution can be used all along the horizon between the two pinch points, we are guaranteed to have the lowest cost blend. If it is not possible at any point in time between the pinch points to produce the required amount of product based on the aggregate recipe, it means that at that time there is not enough of one or more blend components. In order to resolve this infeasibility, the aggregate time period is subdivided at the point of infeasibility and the problem is resolved. More details about the concept of supply-demand pinch concept have been discussed by (Castillo, Kelly. & Mahalec, 2013). The supply-demand pinch concept reduces the complexity of the chance constraint formulation by fixing the amount of each

product blended which greatly improves computation times since amount of products blended directly impact the complexity of the constraints involving the chance constraint formulation.

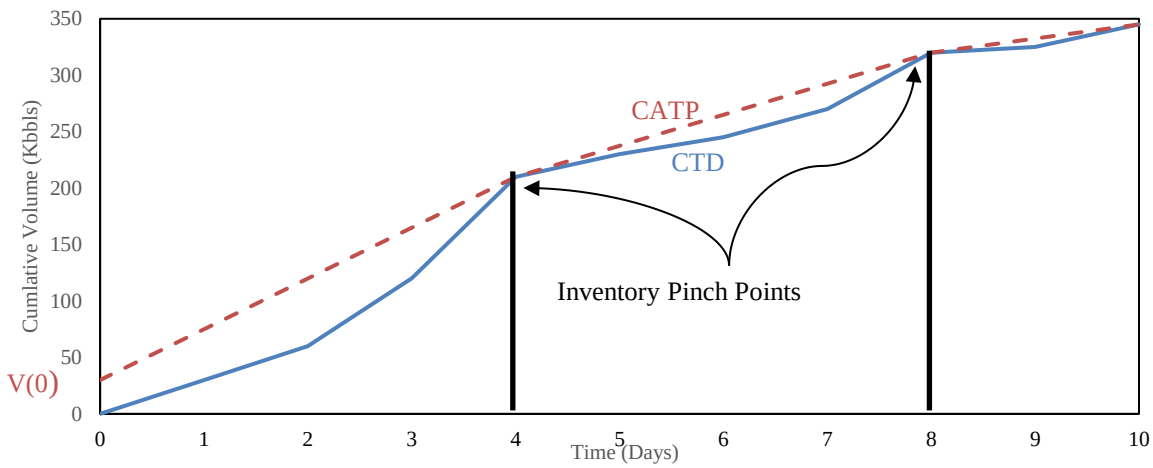

Figure 2. Example of supply-demand pinch points obtained from constructing CTD and CATP curves

2.3.2. Chance Constrained Formulation

In this paper, chance constrained formulation is used to satisfy the product qualities constraints at a predefined confidence interval chosen by the operator using known probability distribution for the uncertain component's qualities. The joint chance constrained formulation is required to deal with uncertainty in components qualities, since products must meet all their quality specification to be considered on-specs products. A joint probability constraint requires all products qualities to be satisfied with a certain confidence level as shown in problem JCC.

$$\min_{x} f(x) \quad s.t. \Pr\{g_i(x,\varepsilon) \geq b_i \,, \forall i = 1, \dots, m\} \geq 1 - \alpha \qquad \text{(JCC)}$$

Where $f(x)$ represent the objective function which does not have uncertain parameters, $Pr\{.\}$ is the probability of all constraints being satisfied, $g_i(x,\varepsilon)$ represents the product quality value as function of the volume of components blended (x) and the uncertain components qualities parameters (ε), b_i represents the minimum products quality required, m is the set of uncertain qualities, and α represent the tolerance for off-specs blend. Since the above joint chance constrained formulation is generally intractable, Bonferroni inequality is used to obtain a conservative approximation with individual chance constrained formulation (ICC).

$$\min_{x,\alpha_i} f(x)$$

$$s.t. \Pr\{g_i(x,\varepsilon) \geq b_i \} \geq 1 - \alpha_i \quad \forall i \in m$$

$\sum_{i \in m} \alpha_i = \alpha$ (ICC)

The above ICC formulation will try to minimize the cost of blend, while forcing the sum of violation of all uncertain qualities constraints to equal the tolerance for off-specs blend. Under the assumption of normal distribution for the uncertain parameters, $\varepsilon \sim N(\hat{\varepsilon}, \sum)$, The ICC problem can be converted to a deterministic problem (ICCN)

$$\min_{x,\alpha_i} f(x)$$

$$s.t. \hat{\varepsilon}^T x - \Phi^{-1}(1-\alpha_i)\sqrt{x^T \sum x} \geq b_i \quad \forall i \in m$$

$\sum_{i \in m} \alpha_i = \alpha$ (ICCN)

Where $\hat{\varepsilon}$ is the mean value of the uncertain parameter, Φ^{-1} is the inverse cumulative distribution function of the standard normal distribution, and $\sum$ is a matrix of the components qualities variances. The above (ICCN) problem is a conservative approximation for the JCC and it provides an upper bound to the (JCC), therefore problem requires computing the blend recipes and the distribution of the constraints violation specified by the user over all the qualities constraints. The above formulation requires a function that describes the inverse cumulative distribution $\Phi^{-1}(1-\alpha_i)$ for small values of violation tolerance $(\alpha_i = 0.05)$ which its plot shown below.

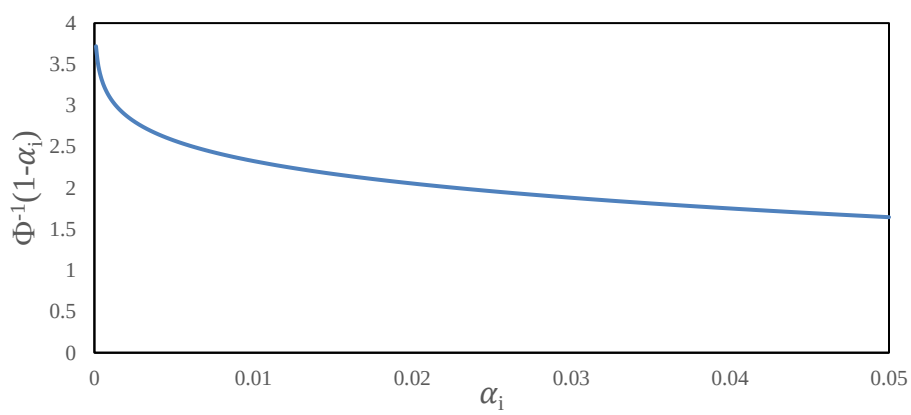

Figure 3. Inverse cumulative distribution function

The above function is highly nonlinear, and we approximated into our model using outer approximation cutting planes with spacing of 0.001 for the range of α_i from 0.05 to 0.01 and spacing of 0.0001 for the range of 0.01 to 0.0001. To avoid gaps between the cutting planes and

the actual cumulative distribution function, smaller spacing between the cutting planes is required when the function values are changing rapidly with α_i.

In this work, the gasoline blend planning problem has been solved using the hierarchical framework similar to the one used by (Castillo & Mahalec, 2014a) as shown in Figure 4. The top-level computes the optimal blend recipes under uncertainty in components quality after delineating the planning time horizon using the supply-demand pinch point concept. The bottom-level uses the blend recipe computed at the top-level to compute the volume required to be blended at each period in the bottom-level (where the time periods at this level are smaller time duration and therefore are more than time periods at-top level). If the second level is infeasible when using blend recipes from the top level, then the top-level time horizon is subdivided with a new period boundary at the point of infeasibility.

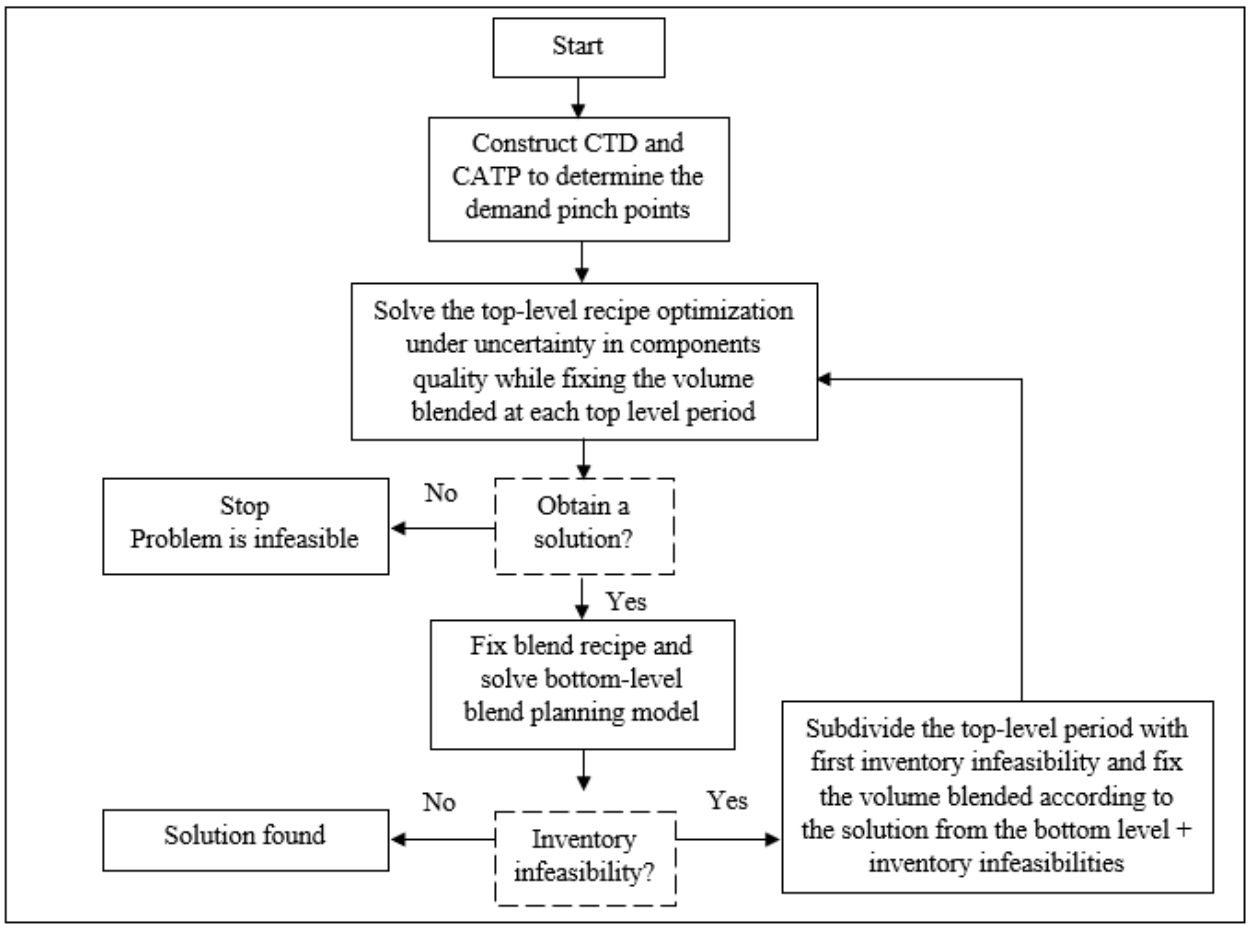

Figure 4. Inventory pinch algorithm under uncertainty in components quality

2.4. Mathematical Models

2.4.1. Mathematical model for Supply-demand Pinch Point Model

In this section, the supply-demand pinch (two-level) model based on (Castillo & Mahalec, 2014a) model is introduced with modification to incorporate uncertainty in components qualities The objective of the top level is to find an optimal blend recipe while meeting products demands and quality specifications. At this level, product demand, blend components supply, and blender capacity are aggregated values for each of the bottom-level periods. In addition, product storage tanks are aggregated into product pools. The solution obtained from this level will minimize the blend cost and is a lower bound for the optimal solution. The solution for the problem will be infeasible if the amounts of the blend components are not sufficient to blend the products to meet the required demand or quality specifications. The objective function of the top-level model, Eq. (3), minimizes the cost of all the blend components *i* used to blend product *p* during period *k*.

min. $BlendCost_{L1} = \sum_{k\epsilon KC}(\sum_{i,p} Cost(i)V_{comp}(i,p,k))$ (3)

Subject to:

The inventory balance equations for blend component *i* and products *p* tanks, equations (4) and (5).

$$\sum_{a\epsilon KA} F_{bc}(a,i)t_{bc}(a,k) + V_{bc}(i,k-1) - V_{bc}(i,k) - \sum_{P} V_{comp}(i,p,k) = 0 \ \ \forall i,k\epsilon KC \quad (4)$$

$$V_{blend\_agg}(p,k) + V_{pool}(p,k-1) - V_{pool}(p,k) - Demand_{agg}(p,k) = 0 \ \ \forall p,k\epsilon KC \quad (5)$$

The set a stands for a specific supply profile and $F_{bc}(a,i)$ is the flow rate according to the supply profile a for each component. The amount of component flow during period *k* is equal the flow rate multiplied by the duration of the time interval when the supply a occurs. The inventory constraints for components and products, Eq. (6) and (7).

$$V_{bc}^{min}(i) \le V_{bc}(i,k) \le V_{bc}^{max}(i) \qquad \forall i,k\epsilon KC \quad (6a)\ (6b)$$

$$V_{pr}^{min}(p) \le V_{pool}(p,k) \le V_{pr}^{max}(p) \qquad \forall p,k\epsilon KC \quad (7a)\ (7b)$$

The initial inventories in components tanks and products pools, Eq. (8) and (9)

$$V_{bc}(i,k) = V_{bc}^{initial}(i) \qquad \forall i,k = 0 \quad (8)$$

$$V_{pool}(p,k) = V_{pool}^{initial}(p) \qquad \forall p,k = 0 \quad (9)$$

The blend recipe, Eq. (10) defines the blending recipe for each product (the fraction of the component in each product). The fraction of each component *i* in a product *p* should sum to 1 for all products and periods which is forced by Eq. (11). the minimum and maximum blend recipe for each product is forced by Eq. (12).

$$V_{comp}(i,p,k) = r(i,p,k)V_{blend\_agg}(p,k) \quad \forall p,k\epsilon KC \tag{10}$$

$$\sum_i r(i,p,k) = 1 \qquad \forall p,k\epsilon KC \tag{11}$$

$$r^{min}(i,p) \le r(i,p,k) \le r^{max}(i,p) \qquad \forall i,p,k\epsilon KC \tag{12a) (12b}$$

The volumetric and weight basis linear quality constraints for blending with no uncertainty, are described by Eq. (13) and (14) respectively. $Q_{bc}(i,s)$ represents the value of quality *s* of component *i*, where $Q_{pr}^{min}(p,s)$ and $Q_{pr}^{max}(p,s)$ are the minimum and maximum quality *s* specification for product *p* respectively.

$$V_{blend\_agg}(p,k)\ Q_{pr}^{min}(p,s) \le \sum_i(V_{comp}(i,p,k)\ Q_{bc}(i,s)) \le V_{blend\_agg}(p,k)\ Q_{pr}^{max}(p,s)$$
$$\forall p,s = linear\ blended\ qualities\ with\ certain\ values\ (volumetric\ basis)\ ,k\epsilon KC$$

(13a) (13b)

$$Q_{pr}^{min}(p,s)\sum_i(V_{comp}(i,p,k)\ Density(i)) \le \sum_i(V_{comp}(i,p,k)\ Q_{bc}(i,s)\ Density(i)) \le$$
$$Q_{pr}^{max}(p,s)\sum_i(V_{comp}(i,p,k)\ Density(i))$$
$$\forall p,s = linear\ blended\ qualities\ (weight\ basis)\ ,k\epsilon KC \tag{14a) (14b}$$

The volumetric basis linear quality constraints for blending with uncertainty in components qualities using chance constraint formulation, Eq. (15)

$$V_{blend\_agg}(p,k)\ Q_{pr}^{min}(p,s) + CI(s,p,k)\sqrt{\sum_i(V_{comp}(i,p,k)\ ^2Q_{st}(i,s)^2)} \le$$
$$\sum_i\left(V_{comp}(i,p,k)\ Q_{bc}(i,s)\right) \le V_{blend_{agg}}(p,k)\ Q_{pr}^{max}(p,s) -$$
$$CI(s,p,k)\sqrt{\sum_i(V_{comp}(i,p,k)\ ^2Q_{st}(i,s)^2)} \qquad \forall p,s\epsilon SU,k\epsilon KC \tag{15a) (15b}$$

$CI(s,p,k)$ is the inverse cumulative normal distribution for all uncertain qualities *s* of product *p* during period *k*. One important thing to notice in Eq. (15) is that the term $V_{blend\_agg}(p,k)$ is a parameter since in the supply-demand pinch formulation the amount blended is fixed based on aggregated demand.

Eq. (16) enforces summation of violation of each individual chance constraint to not exceed the violation of the joint chance constraint. Eq. (17) represents the outer approximation constraints for the inverse cumulative distribution function.

$$\alpha_{JC}(p,k) = \sum_{s\epsilon SU} \alpha_{IC}(s,p,k) \qquad \forall p,k\epsilon KC \tag{16}$$

$$CI(s,p,k) \geq a(n) * \left(1 - \alpha_{IC}(s,p,k)\right) + b(n) \quad \forall p,s\epsilon SU,k\epsilon KC,n \tag{17}$$

The set of equations (3) - (14) describes the top-level model when uncertainty is not considered, while equations (3) - (17) represent the top-level model when uncertainty in qualities is considered. Solving the top-level computes optimal blend recipes which are fixed when solving the bottom-level model. The bottom-level model is used to compute the volume blended in each period of the bottom level subject to inventory constraints and threshold blend amount. The bottom-level objective function is given by Eq. (18). The objective is to minimize the blend cost and ensure that the blend recipes computed at the top level are feasible. The slack variables are forced to be positive and the penalty weight of the associated with the product infeasibility increase with the number of periods to delay any potential product inventory infeasibility as far into the future as possible. Also, the penalty weights associated with the components inventory slack variables are much greater than the penalty weights associated with the products inventory slack variables to for inventory infeasibilities to be on the products' side.

$$BlendCost_{L2} = \sum_{m\epsilon MK}\left[\sum_{(bl,p)\epsilon BP}\sum_i Cost(i)V_{comp}(i,p,m,bl)\right] + \sum_{m\epsilon MK}[\sum_i Penalty_{bc} * (S_{bc}^{+}(i,m) + S_{bc}^{-}(i,m))] + \sum_{m\epsilon MK}\left[\sum_p Penalty_{pr}(m) * (S_{pool}^{+}(p,m) + S_{pool}^{-}(p,m))\right] + \sum_{m\epsilon MK}\left[\sum_p\sum_j Penalty_{pr}(m) * (S_{pr}^{+}(j,p,m) + S_{pr}^{-}(j,p,m))\right] \tag{18}$$

Subject to:

The inventory balance equations for components, products and tanks with slack variables, Eq. (19) to (21). The slack variables should indicate the period where the inventory balance equation is violated to indicate the period where infeasibility occurs.

$$\sum_{a\epsilon AM} F_{bc}(a,i)t_{bc}(a,m) + V_{bc}(i,m-1) - V_{bc}(i,m) - \sum_{(bl,p)\epsilon BP} V_{comp}(i,p,m,bl) + S_{bc}^{+}(i,m) - S_{bc}^{-}(i,m) = 0 \qquad \forall i,m\epsilon MK \tag{19}$$

$$\sum_{bl\epsilon BP} V_{blend}(p,m,bl) + V_{pool}(p,m-1) - V_{pool}(p,m) - Deliver_{pool}(p,m) + S_{pool}^{+}(p,m) - S_{pool}^{-}(p,m) = 0 \qquad \forall p,m\epsilon MK \tag{20}$$

$$\sum_{bl\epsilon BP} V_{trans}(j,p,m,bl) + V_{pr}(j,p,m-1) - V_{pr}(j,p,m) - Deliver_{pr}(j,p,m) + S_{pr}^{+}(j,p,m) - S_{pr}^{-}(j,p,m) = 0 \qquad \forall (j,p)\epsilon JP, m\epsilon MK \tag{21}$$

Eq. (22) fixes the blend recipe to the recipe computed at the top-level where *bl* is the set of blenders, BP is set of blender *bl* that can blend product *p*, and *KM* is set of second level periods that correspond to the top level period.

$$V_{comp}(i,p,m,bl) = r(i,p,k)V_{blend}(p,m,bl) \quad \forall i,(bl,p)\epsilon BP,(k,m)\epsilon KM \tag{22}$$

Eq. (23) ensures that a blender can blend $np(bl)$ products at max in each period, where the binary variable $x(p,bl,m)$ takes value of 1 if product *p* is blended in blender *bl* during period *m* and the parameter $cit_{blend}^{min}(p,bl)$ is the minimum idle time required by blender *bl* to process product *p*. Eq. (24) to (27) enforce blender capacity constraints. First one enforces the maximum blender capacity assuming some idle time equivalent to the product of the maximum blending capacity and the minimum idle time required by the blender. The other three equations impose constraints on the volume blended due to the minimum and maximum blending rate for the blender and some minimum threshold pre-specified by the planner.

$$\sum_{p\epsilon BP} x(p,bl,m) \leq np(bl) \qquad \forall bl,m\epsilon MK \tag{23}$$

$$\sum_{p\epsilon BP} V_{blend}(p,m,bl) + F_{blend}^{max}(bl) \sum_{p\epsilon BP}(cit_{blend}^{min}(p,bl)\, x(p,bl,m)) \leq F_{blend}^{max}(bl)\, t(m) \qquad \forall bl,m\epsilon MK \tag{24}$$

$$V_{blend}(p,m,bl) \leq F_{blend}^{max}(bl)\, t(m)\, x(p,bl,m) \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{25}$$

$$V_{blend}(p,m,bl) \geq F_{blend}^{min}(bl)\, ct_{blend}^{min}(p,bl)\, x(p,bl,m) \quad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{26}$$

$$V_{blend}(p,m,bl) \geq V_{blend}^{min}(bl)\, x(p,bl,m) \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{27}$$

Eq. (28) to (31) enforce constraints on the blender running times, $t_{blend}(p,m,bl)$. First constraint forces the running time of a blend to be greater than or equal than the minimum running time for the blender. The parameter $ct_{blend}^{min}(p,bl)$ represent the minimum running time required by blender *bl* when processing product *p*. Eq. (29) and (30) set the limits on the upper and lower running time for the blend, while Eq. (31) ensures that running time of the blend at *m* period plus the product changeover times is less than or equal the length of period *m*.

$$t_{blend}(p,m,bl) \geq ct_{blend}^{min}(p,bl)\, x(p,bl,m) \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{28}$$

$$t_{blend}(p,m,bl) \geq \frac{V_{blend}(p,m,bl)}{F_{blend}^{max}(bl)} \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{29}$$

$$t_{blend}(p,m,bl) \leq \frac{V_{blend}(p,m,bl)}{F_{blend}^{min}(bl)} \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{30}$$

$$\sum_{p\epsilon BP} t_{blend}(p,m,bl) + \sum_{p\epsilon BP} cit_{blend}^{min}(p,bl)\, x(p,bl,m) \leq t(m) \;\; \forall bl, m\epsilon MK \tag{31}$$

Eq. (32) forces the amount of volume blended of product *p* in blender *bl* sent to tank *j* to be equal or less than the maximum volume possible to blend in the blender. $v(j,p,m,bl)$ is a binary variable that takes value of 1 if tank *j* is receiving product *p* from blender *bl* during period *m*. Eq. (33) ensure that volume receive by the tanks from blenders is equal to the volume blended for all products during all periods. Eq. (34) and (35) state that the volume blended of product *p* in blender *bl* to be sent to only one product tank during period *m* and that tank should be either empty or already stores that product. $u(j,p,m)$ is binary variable that takes value of 1 if tank *j* is storing product *p* during period *m*.

$$V_{trans}(j,p,m,bl) \leq F_{blend}^{max}(bl)\, t(m)\, v(j,p,m,bl) \;\; \forall (bl,p)\epsilon BP, (j,p)\epsilon JP, m\epsilon MK \tag{32}$$

$$\sum_{(j)\epsilon JP} V_{trasn}(j,p,m,bl) = V_{blend}(p,m,bl) \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{33}$$

$$\sum_{(j)\epsilon JP} v(j,p,bl,m) \leq 1 \qquad \forall (bl,p)\epsilon BP, m\epsilon MK \tag{34}$$

$$v(j,p,bl,m) \leq u(j,p,m) \qquad \forall (bl,p)\epsilon BP, (j,p)\epsilon JP, m\epsilon MK \tag{35}$$

Eq. (36) ensures that the only one product can be stored in component tank *j* during m period. Eq. (37) ensures that the component tank *j* maximum capacity constraint is satisfied. Eq. (38) relates the product pool inventories to the individual product tank inventories.

$$\sum_{(p)\epsilon JP} u(j,p,m) = 1 \qquad \forall\, j,m \tag{36}$$

$$V_{pr}(j,p,m) \leq V_{pr}^{max}(j)\, u(j,p,m) \qquad \forall (j,p)\epsilon JP, m \tag{37}$$

$$V_{pool}(p,m) = \sum_{(j)\epsilon JP} V_{pr}(j,p,m) \qquad \forall\, p,m \tag{38}$$

Eq. (39) and (40) introduce the binary variable $ue(j,m)$ which take value of 1 if a product transition has taken a place in product tank *j* at period *m*. Eq. (41) forces a product transition in product tank *j* to occur if the tank is empty at the end of the previous period.

$$ue(j,m) \geq u(j,p,m) - u(j,p,m-1) \qquad \forall (j,p)\epsilon JP, m\epsilon MK \tag{39}$$

$$ue(j,m) \geq u(j,p,m-1) - u(j,p,m) \qquad \forall (j,p)\epsilon JP, m\epsilon MK \tag{40}$$

$$V_{pr}(j,p,m-1) \leq V_{pr}^{max}(j)\left(1 - ue(j,m)\right) \quad \forall (j,p)\epsilon JP, m\epsilon MK \tag{41}$$

Eq. (42a) and (42b) are minimum and maximum inventory constraints for the component's tanks. Eq. (43) is a minimum pool product inventory level constraint. Eq. (44) ensures that the component tank *j* minimum capacity constraint is met.

$$V_{bc}^{min}(i) \leq V_{bc}(i,m) \leq V_{bc}^{max}(i) \qquad \forall i,m \qquad \text{(42a) (42b)}$$

$$V_{pool}(p,m) \geq V_{pool}^{min}(p) \qquad \forall p,m \qquad (43)$$

$$V_{pr}(j,p,m) \geq V_{pr}^{min}(j)\, u(j,p,m) \qquad \forall (j,p)\epsilon JP, m \qquad (44)$$

Eq. (45) states which product *p* is stored in tank components *j* at the start of time horizon. Eq. (46) and (47) set the volume of the component and product tanks at the start of the time horizon respectively. Eq. (48) set the initial blender state, whether the blender is running to blend a product *p* or is idle at the start of time horizon.

$$u(j,p,m=0) = u^{initial}(j,p) \qquad \forall (j,p)\epsilon JP \qquad (45)$$

$$V_{bc}(i,m=0) = V_{bc}^{initial}(i) \qquad \forall i \qquad (46)$$

$$V_{pr}(j,p,m=0) = V_{pr}^{initial}(j,p) \qquad \forall (j,p)\epsilon JP \qquad (47)$$

$$x(p,bl,m=0) = x^{initial}(p,bl) \qquad \forall (j,p)\epsilon JP \qquad (48)$$

Eq. (49) ensures that the delivery amount of product *p* from product tank *j* during period m does not exceed the maximum delivery rate possible. Eq. (50) relates the sum of product volume delivered from all product tanks to the amount of product delivered during period *m*. Eq. (51) set the amount of product delivered in period *m* is equal to a fraction of the total demand. Eq. (52) ensures that only the contracted fraction of the demand for order *o* is shipped. Eq. (53) ensures the completion of the order *o*. Eq. (54) ensures that the fraction delivered of order *o* is less than or equal to the maximum possible delivery.

$$Deliver_{pr}(j,p,m) \leq D_{pr}^{max}(j)t(m)\, u(j,p,m) \qquad \forall (j,p)\epsilon JP, m\epsilon MK \qquad (49)$$

$$Deliver_{pool}(p,m) = \sum_{(j)\epsilon JP} Deliver_{pr}(j,p,m) \qquad \forall p,m\epsilon MK \qquad (50)$$

$$Deliver_{pool}(p,m) = \sum_{(O)\epsilon OP} Demand(o)\, of(o,m) \qquad \forall p,m\epsilon MK \qquad (51)$$

$$of(o,m) \leq uof(o,m) \qquad \forall o,m\epsilon MK \qquad (52)$$

$$\sum_{(m)\epsilon MK} of(o,m) = 1 \qquad \forall o \qquad (53)$$

$$Demand(o)\, of(o,m) \leq D_{order}^{max}(o)dt(o,m) \qquad \forall o,m\epsilon MK \qquad (54)$$

The set of equations (18) to (54) represent the bottom-level model which used to compute the volume blended during each period of the second level.

2.4.2. Mathematical model for the Full Space Model

The full-space model under uncertainty can be obtained from the second level model by removing Eq. (22) and adding Eq. (11) - (17), and the two following equations:

$$\sum_i V_{comp}(i,p,m,bl) = V_{blend}(p,m,bl) \qquad \forall (bl,p) \epsilon BP, m \epsilon MK \tag{55}$$

$$r^{min}(i,p)\, V_{blend}(p,m,bl) \leq V_{comp}(i,p,m,bl) \leq r^{max}(i,p)\, V_{blend}(p,m,bl)$$

$$\forall i, (bl,p) \epsilon BP, m \epsilon MK \qquad \text{(56a) (56b)}$$

2.5. Case Studies

In this work. Two sets of case studies have been solved; the first set (example 1 and 2) considers small scale test problems with 3 or 4 periods and 1 blender, while the second set (example 3,4 and 5) considers large scale test problems with 14 periods and 1 to 3 blenders. For all test problems three different models have been solved to study the effect of considering different uncertainties in components qualities on optimal solutions and computation times. The first model assumes all components qualities to be deterministic, the second model consider uncertainty in RON and MON quality, while the third model considers uncertainty in RON, MON and BEN. The uncertainty in component's qualities is assumed to follow a normal distribution, with mean values equal their deterministic values and standard deviation equal 1% of their mean values. The other qualities are assumed to be known for all cases. The large test problems used are taken from (Castillo & Mahalec, 2014a).

The components qualities and products minimum and maximum qualities are shown in Table 1. The qualities specifications for all products are the same except for the minimum RON and MON qualities for the three products.

Table 1. Component's qualities and products minimum and maximum qualities

Property	Blend Components						
	ALK	BUT	HCL	HCN	LCN	LNP	RFT
ARO (%vol aromatics)	0	0	0	25	18	2.974	60.9
BEN (%vol benzene)	0	0	0	0.5	1	0.595	7.5
MON	93.7	90	79.8	75.8	81.6	66	90.8
OLF (%vol olefin)	94	92.8	82.3	86.7	92.2	67.8	102
RON	0	0	0	14	27	0	0
RVP (psi)	5.15	138	22.335	2.378	13.876	19.904	3.622
SPG	0.703	0.584	0.695	0.791	0.744	0.677	0.818
SUL (%vol sulfur)	0	0	0	0.485	0.078	0.013	0
	Product Specifications [Min, Max]						
	P1		P2		P3		
ARO (%vol aromatics)	[0,60]		[0,60]		[0,60]		
BEN (%vol benzene)	[0,4.4]		[0,4.4]		[0,4.4]		
MON	[81.5,200]		[85.5,200]		[87.5,200]		
OLF (%vol olefin)	[90.4,200]		[92.5,200]		[95.5,200]		
RON	[0,24.2]		[0,24.2]		[0,24.2]		
RVP (psi)	[0,15.6]		[0,15.6]		[0,15.6]		
SPG	[0.73,0.81]		[0.73,0.81]		[0.73,0.81]		
SI (%vol sulfur)	[0,0.1]		[0,0.1]		[0,0.1]		

Figures 5 to 9 show the cumulative total demand (CTD) and cumulative average total production (CATP) for examples 1 to 5. the figures are used to obtain the supply-demand pinch points and determine the number of top-level periods and the amount of products blended in each top level periods. Examples 1 and 3 have no supply-demand pinch points so the top-level model will have only single period, while examples 2 and 4 have one supply-demand pinch point so two top level periods will be used. Example 5 cumulative total demand curves resulted in 3 supply-demand pinch points, therefore the planning horizon at top level will have 4 periods.

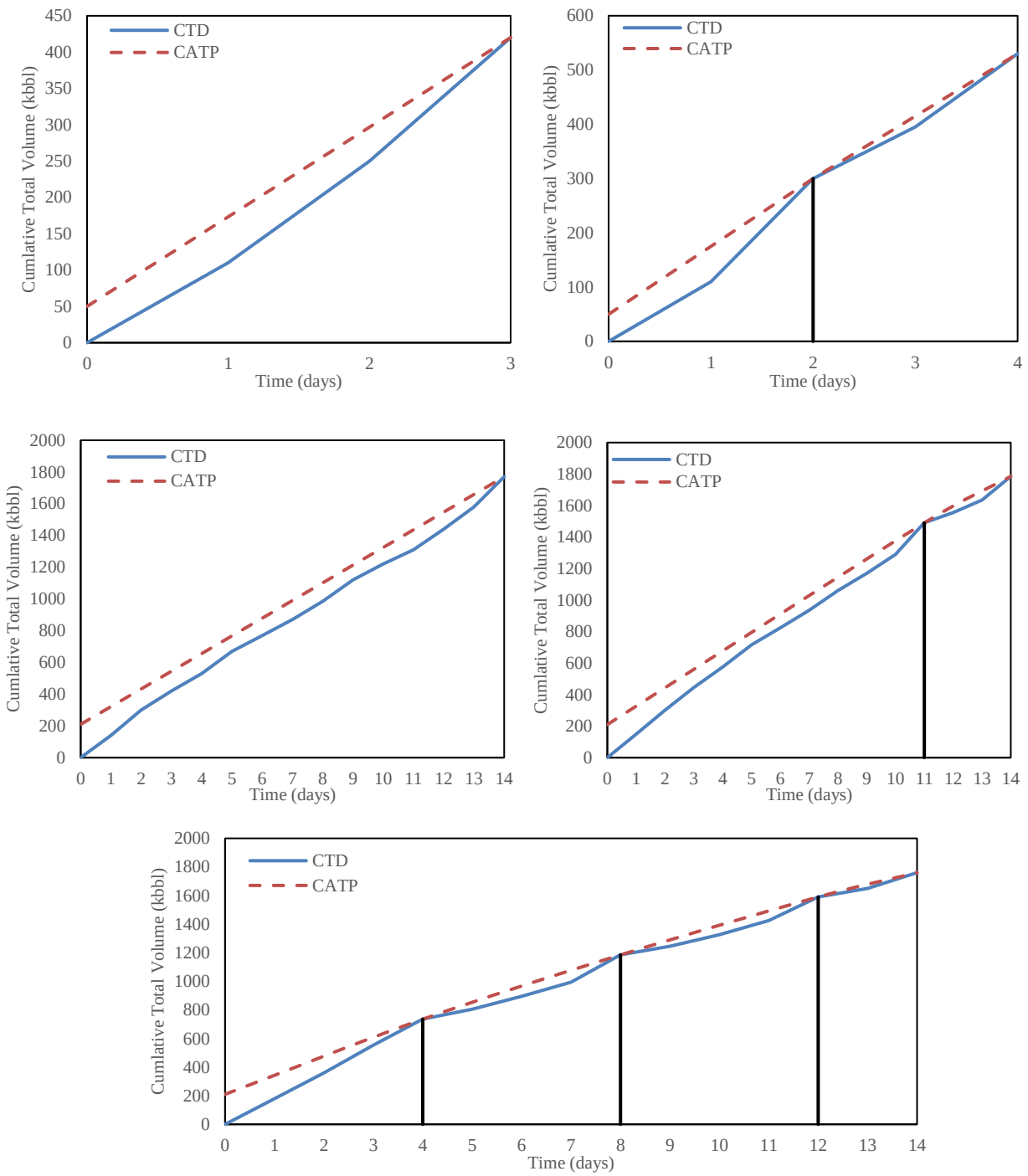

Figure. 5– 9. Cumulative total demand and average total production with supply-demand pinch points for examples 1 to 5

2.6. Results and Discussion

All case studies have been implemented in GAMS 24.4.1 software. LP and MILP models have been solved using CPLEX 12.6 solver, while NLP and MINLP models have been solved using Baron 16.8.24 solver. All problems have been solved using desktop computer (Intel ® Core™ i5-4690K CPU, 3.50 GHz, and 8.0 GB RAM) running Windows 7 Ultimate. All examples have been solved assuming deterministic components qualities, uncertainty in RON and MON components qualities, and uncertainty in RON, MON, and BEN components qualities. The stopping criteria of 0.01% optimality gap or maximum computational time of 3600 seconds have been used.

Table 2 shows the model size for each test problem for the full-space and the supply-demand pinch top and bottom levels. When solving the problem without incorporating uncertainty all three models are linear, while inclusion of uncertainty in qualities for the blending components results in nonlinear models for the full-space and top-level supply-demand pinch, but not the bottom-level since no quality specification constraints are included in that model. Example ID # 2 and 4 required two iterations for the deterministic case due to infeasibility of the optimal blend recipe computed at top-level in the first iteration. Notice that the top-level model was larger in size for the second iteration compared to first iteration since the number of periods increased by one due to subdividing the single period in top level.

When solving Example # 2 for the deterministic case with two periods at the top level the recipes computed at the top level was infeasible at the third period of the bottom level in product 2 because we run out of component C3. Therefore, the top-level planning horizon was subdivided at the end of period 3 and the top level has been resolved using the blended volume computed at the bottom level adjusted by the infeasibility values. The recipes computed at the second iteration with the 3 periods at the top level gave a feasible production plan at the bottom level. When solving Example # 4 for the deterministic case with two periods at the top level the recipes computed shows multiple infeasibilities and the first infeasibility occurs for product 3 at period 2 due to violating blender capacity. Therefore, the first period of the top level was subdivided, and the products blended volume at the top-level periods was obtained from the bottom level solution adjusted by the infeasibility. The top level should be resolved using 3 periods which resulted in infeasibility in the bottom level in product 1 at period 2, but this blend amount can be moved to next period since it is not required for the current period demand. The top-level model should be

resolved with the new volume blended for product 1 using 3 periods as in iteration 2. In the third iteration the blend recipes computed at top level were feasible and the bottom level model gave the optimal production plan.

Table 2. Full space and supply-demand pinch (two-level) model sizes for all different cases and examples

Example ID	1	2	3	4	5
Full space (deterministic cases)					
# periods	3	4	14	14	14
# equations	1,085	1,442	5582	5,597	9,601
# continuous variables	411	547	2,237	2,237	3,469
# discrete variables	102	132	432	432	768
# nonlinear terms	0	0	0	0	0
Full space (uncertainty in RON and MON cases)					
# periods	3	4	14	14	14
# equations	3596	4790	17,300	17,315	36,971
# continuous variables	447	595	2,405	2,405	3,973
# discrete variables	102	132	432	432	768
# nonlinear terms	288	384	1,344	1,344	4,032
Full space (uncertainty in RON, MON and BEN cases)					
# periods	3	4	14	14	14
# equations	4847	6458	23138	23,153	50,593
# continuous variables	465	619	2489	2,489	4,225
# discrete variables	102	132	432	432	768
# nonlinear terms	378	504	1764	1,764	5,292

Top-level Pinch Point (deterministic cases)							
Iteration	1	1	2[a]	1	1	2,3[a]	1
# periods	1	2	3	1	2	3	4
# equations	170	308	446	170	308	446	584
# continuous variables	64	116	168	64	116	168	220
# discrete variables	0	0	0	0	0	0	0
# nonlinear terms	0	0	0	0	0	0	0

Top-level Pinch Point (uncertainty in RON and MON cases)					
Iteration	1	1	1	1	1
# periods	1	2	1	2	4

# equations	1007	1982	1007	1,982	3,932
# continuous variables	76	140	76	140	268
# discrete variables	0	0	0	0	0
# nonlinear terms	96	192	96	192	384
Top-level Pinch Point (uncertainty in RON, MON and BEN cases)					
Iteration	1	1	1	1	1
# periods	1	2	1	2	4
# equations	1424	2816	1424	2,816	5,600
# continuous variables	82	152	82	152	292
# discrete variables	0	0	0	0	0
# nonlinear terms	126	252	126	252	504
Bottom-level Pinch Point (All cases)					
# periods	3	4	14	14	14
# equations	816	1083	4,323	4,338	5,710
# continuous variables	571	760	2,980	2,980	4,184
# discrete variables	102	132	432	432	768
# nonlinear terms	0	0	0	0	0

a: Second iteration is required since recipes computed at top-level was infeasible when used in the bottom-level, the problem was solved again after subdividing the top-level period where infeasibility occurred.

Table 3 summarizes computational results for all examples with deterministic and uncertainty in components qualities cases using the full space and supply-demand pinch models. For deterministic case, both models are linear and computations times are less than 5 seconds for all examples. For uncertainty in components qualities cases, the supply-demand pinch model computed the optimal solution and closed the gap for all examples. The full space model took longer to reach the optimal solution and failed in closing the optimality gap for most examples. The supply-demand pinch model computed a slightly better solution than the full space model for examples 4 and 5. These results shows how effective the supply-demand pinch model in reducing the computational times required to compute an optimal solution and closing the gap for large size problems.

Table 3. Results for supply-demand pinch and full space models

Example ID	Uncertainty Model	Supply-demand Pinch			Full Space (BARON)			
		Total Cost ($)	CPU time (s)		Total Cost ($)	CPU time (s)	Gap%	Time to best solution
			Top	Bottom				
Demand 1	Deterministic	8387.0	0.01	0.09	8387.0	0.10	0.010	-
	Uncertainty in RON and MON	8483.4	0.76	0.08	8483.4	53.28	0.010	6.13
	Uncertainty in RON, MON and BEN	8486.2	1.66	0.08	8486.2	117.64	0.010	13.20
Demand 2	Deterministic	10882.5	0.02[a]	0.42[a]	10882.5	0.18	0.010	-
	Uncertainty in RON and MON	11000.8	4.92	0.11	11000.8	1393.19	0.010	20.72
	Uncertainty in RON, MON and BEN	11004.1	16.69	0.11	11004.1	3600.00	0.036	73.06
Demand 3	Deterministic	37193.7	0.01	0.55	37193.7	2.07	0.010	-
	Uncertainty in RON and MON	37621.7	8.40	0.55	37621.9	3600.00	0.18	713.21
	Uncertainty in RON, MON and BEN	37622.1	6.53	0.56	37622.3	3600.00	0.18	885.06
Demand 4	Deterministic	37575.6	0.05[a]	1.22[a]	37575.6	1.65	0.010	-
	Uncertainty in RON and MON	38084.6	68.85	0.37	38084.8	3600.00	0.134	601.70
	Uncertainty in RON, MON and BEN	38085.0	98.65	0.39	38085.2	3600.00	0.168	1163.77
Demand 5	Deterministic	36970.5	0.04	0.58	36970.5	4.50	0.010	-
	Uncertainty in RON and MON	37527.8	1714.13	0.50	37528.5	3600.00	0.181	737.95
	Uncertainty in RON, MON and BEN	37528.3	3308.99	2.23	37528.5	3600.00	0.179	2558.29

a: computation times are the sum of all iterations.

Another advantage of using the supply-demand pinch algorithm is to minimize the number of blending recipes for each product along the planning horizon. The number of recipes computed in the supply-demand pinch algorithm are bounded by the number of periods in the top-level model. While the full space model might compute the same optimal solution but using a larger

number of different blend recipes along the planning horizon. Figures 10 and 11 shows the blending recipes computed for example 4 under the case where qualities of RON, MON and BEN are assumed to be uncertain for both full space and supply-demand pinch models. The figure show that supply-demand pinch model requires blending with two different blend recipes over the planning horizon, while the full space model requires four different blending recipes along the planning horizon. During period 11, product 2 recipes are not shown since product 2 is not blended during this period.

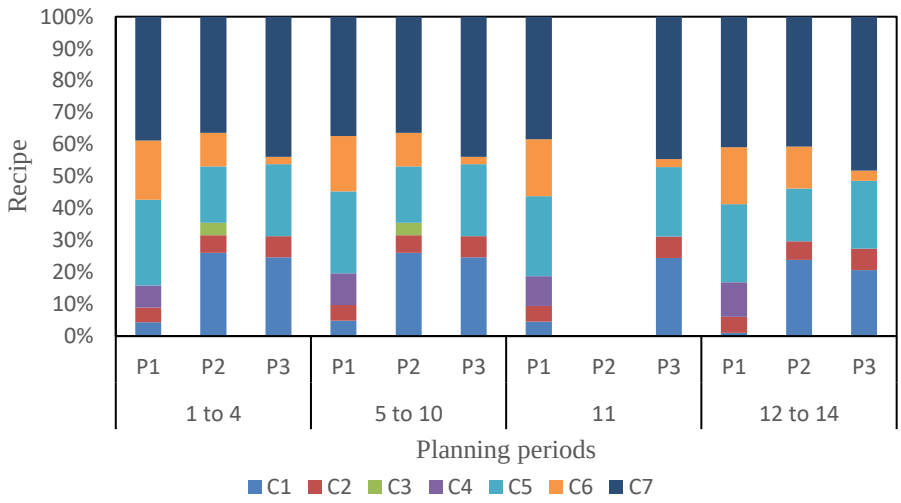

Figure 10. optimal blending recipes computed from full space model for example 4

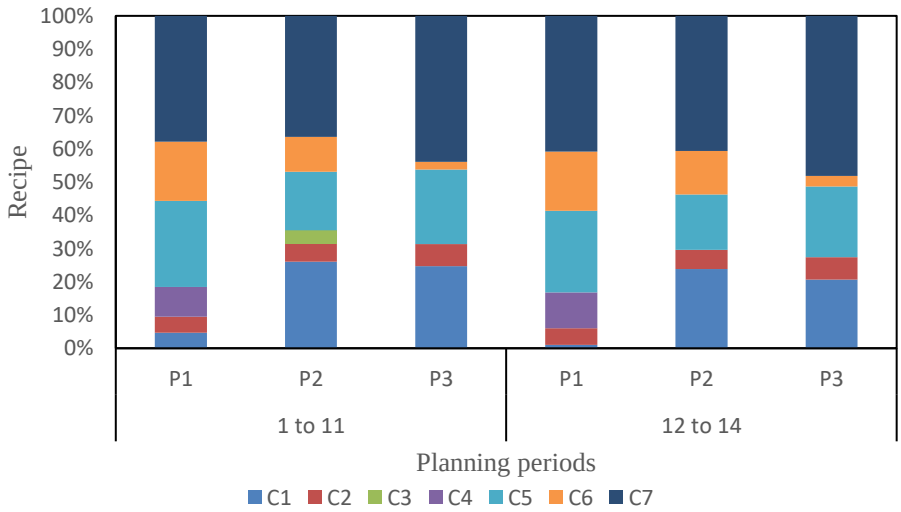

Figure 11. optimal blending recipes computed from the supply-demand pinch model for example 4

Figure 12 shows the relationship between blending cost and probability of producing on-spec products for example 4. Higher probability of meeting products specifications specified by the operator comes at the cost of higher blending cost. The cost of blending changes at an increasing rate when probability of meeting products specification required is higher, that happens due to the need of using the expensive components to guarantee meeting the products specifications. Optimizer must specify the parameter of probability blending on-spec products after evaluating the cost of re-blending for the refinery.

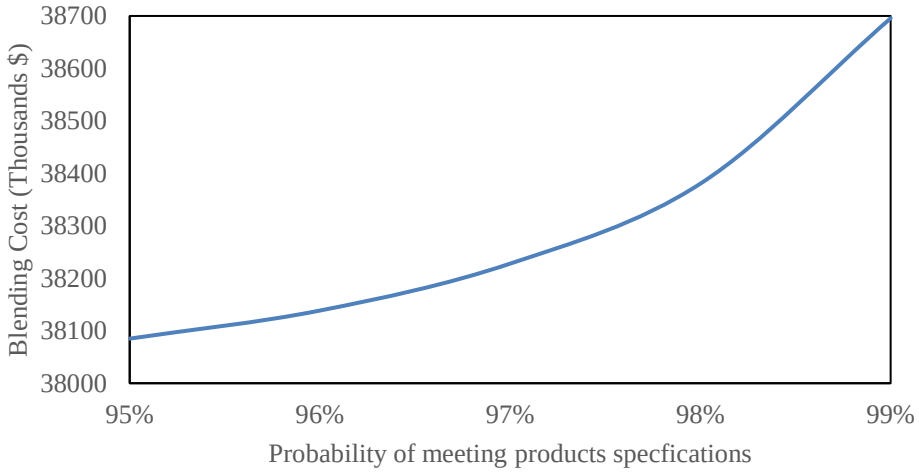

Figure 12. Blending cost ($) versus probability of meeting products specifications (%) for example 4 under uncertainty in RON, MON and BEN components qualities

2.7. Conclusion

In this paper, joint chance constrained formulation was utilized to solve the gasoline blend problem at the planning level (offline optimization) when uncertainty in components qualities are considered. By giving some quality away we can eliminate the high cost required for re-blending when products do not meet the specifications due to these uncertainties. Incorporating chance constrained formulation makes the multi-period full space model difficult and often not possible to solve for large size problems. On the other hand, supply-demand pinch-based decomposition of the planning problem enables great reduction in the execution time required to find the optimum solution and reduces the blending recipes along the planning horizon. Small size problems require orders of magnitude shorter computational times when solved via demand-pinch based decomposition, while large problems are solved reliably in acceptable execution times. The pinch point algorithm closed the optimality gap at least five-fold faster than the full

space model with BARON solver. Results of this work indicate that the supply-demand pinch-based approach is likely to be effective in solving the refinery planning problems with uncertainties in the crude qualities and the components qualities. Our further work will explore this hypothesis.

Acknowledgments

Support by McMaster Advanced Control Consortium is gratefully acknowledged.

Nomenclature

Sets and subsets

i – set of blend components

p – set of different products

k – set of periods at top-level

m – set of periods at bottom-level

s – set of qualities

o – set of demand orders

bl – set of blenders

j – set of product tanks

a – set of different supply flow rates of blend components

n – set of linear function for outer approximation for the inverse cumulative normal distribution

KC – subset of periods at top-level excluding period 0

MK – subset of periods at bottom-level excluding period 0

JP – subset of blender bl that can feed tank j

BP – subset of blender bl that can produce product p

KM – subset of periods m (bottom-level periods) that correspond to periods k (top-level periods)

Continuous variables

$BlendCost_{L1}$ – component cost for top-level model

$BlendCost_{L2}$ – component cost for bottom-level model

$V_{comp}(i,p,k)$ – volume of component i into product p at period k

$V_{comp}(i,p,m,bl)$ – volume of component i into product p in blender bl at period m

$V_{blend}(p,m,bl)$ – volume blended of product p in blender bl at period m

$V_{trans}(j,p,m,bl)$ – volume of product p in blender bl transferred to product tank j at period m

$V_{pr}(j,p,m)$ – volume of product p stored in tank j during period m

$V_{bc}(i,k)$ – volume stored of component i in period k

$V_{bc}(i,m)$ – volume stored of component i in period m

$V_{pool}(p,k)$ – volume stored in product pool p at the end of period k

$V_{pool}(p,m)$ – volume stored in product pool p at the end of period m

$r(i,p,k)$ - fraction of component i into product p during period k

$Q_{pr}(p,s,k)$ – quality value of property s for product p in period k

$Deliver_{pool}(p,m)$ – volume of product p shipped at the end of period m

$Deliver_{pr}(j,p,m)$ – volume of product p shipped from product tank j at the end of period m

$of(o,m)$ – fraction of order o to be delivered during period m

$S_{bc}^{+}(i,m)$, $S_{bc}^{-}(i,m)$ – positive and negative inventory slack variable of components i during period m

$S_{pool}^{+}(p,m)$, $S_{pool}^{-}(p,m)$ – positive and negative inventory slack variable of product pool p during period m

$S_{pr}^{+}(j,p,m)$, $S_{pr}^{-}(j,p,m)$ – positive and negative inventory slack variable of product p in product tank j during period m

Integer variables

$t_{blend}(p,m,bl)$ – time required for blend run

Binary variables

$x(p,bl,m)$ – defines if product p is blended in blender bl during period m

$v(j,p,m,bl)$ – defines if product p is transferred from blender bl to product tank j during period m

$u(j,p,m)$ – defines if product p is stored in product tank j during period m

$ue(j,m)$ – defines if a product transition occurs at product tank j at the beginning of period m

Parameters

$Cost(i)$ – cost of component i

$t(m)$ – duration of period m

$F_{bc}(a,i)$ – supply flow rate of component i during time interval a

$t_{bc}(a,k)$ – duration of time interval where component supply flow rate a at period k

$t_{bc}(a,m)$ – duration of time interval where component supply flow rate a at period m

$V_{blend\_agg}(p,k)$ – volume blended of product p at period k

$Demand_{agg}(p,k)$ – product p demand at period k

$Q_{bc}(i,s)$ – quality value of property s for component I

$V_{bc}^{min}(i)$ – minimum capacity of tank with blend component i

$V_{bc}^{max}(i)$ – maximum capacity of tank with blend component i

$V_{pr}^{min}(p)$ – minimum capacity of tank with product p

$V_{pr}^{max}(p)$ – maximum capacity of tank with product p

$r^{min}(i,p)$ – minimum fraction of component i in product p

$r^{max}(i,p)$ – maximum fraction of component i in product p

$Q_{pr}^{min}(p,s)$ – minimum quality value of property s in product p

$Q_{pr}^{max}(p,s)$ – maximum quality value of property s in product p

$F_{blend}^{min}(bl)$ – minimum blending rate of blender bl

$F_{blend}^{max}(bl)$ – maximum blending rate of blender bl

$cit_{blend}^{min}(p,bl)$ - minimum idle time required by blender to process product p

$ct_{blend}^{min}(p,bl)$ – minimum running time of blender bl for product p$V_{blend}^{min}(bl)$ – minimum volume blended in blender bl

$t_{blend}^{min}(bl)$ – maximum volume blended in blender bl

$V_{pr}^{max}(j)$ – maximum capacity of tank product j

$V_{pool}^{min}(p)$ – minimum capacity of product pool p

$D_{pr}^{max}(j)$ – maximum delivery rate of tank j

$D_{order}^{max}(o)$ – maximum delivery rate of order o

$V_{bc}^{initial}(i)$ – component i starting inventory

$V_{pool}^{initial}(p)$ – product p starting inventory

$u^{initial}(j,p)$ – initial product p in product tank j

$V_{pr}^{initial}(j,p)$ – initial volume of product p in product tank j

$x^{initial}(p,bl)$ – initial product b in blender bl

$Density(i)$ – density of component i

CI – z-score of the required confidence interval

$Q_{st}(i,s)$ – standard deviation for uncertain qualities for component i

$Demand(o)$ – volume of demand o

$np(bl)$ – number of products that can be blended in blender bl

$uof(o,m)$ – determine if order o can be delivered during period m

$dt(o,m)$ – time available to deliver order o during period m

$Penalty_{bc}$ – penalty associated with components inventory infeasibilities

$Penalty_{pr}(m)$ – penalty associated with products inventory infeasibilities

Literature Cited

ASTM international. (2018). ASTM D2699 - 18 Standard Test Method for Research Octane Number of Spark-Ignition Engine Fuel. Retrieved August 30, 2018, from https://www.astm.org/Standards/D2699.htm

Biegler, L. T. (2010). *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes*. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898719383

Castillo, P. A. C., & Mahalec, V. (2014). Inventory pinch based, multiscale models for integrated planning and scheduling-part I: Gasoline blend planning. *AIChE Journal*, *60*(6), 2158–2178. https://doi.org/10.1002/aic.14423

Castillo, P. A. C., & Mahalec, V. (2014). Inventory pinch based, multiscale models for integrated planning and scheduling-part II: Gasoline blend scheduling. *AIChE Journal*, *60*(7), 2475–2497. https://doi.org/10.1002/aic.14444

Castillo, P. A. C., Mahalec, V., & Kelly, J. D. (2013). Inventory pinch algorithm for gasoline blend planning. *AIChE Journal*, *59*(10), 3748–3766. https://doi.org/10.1002/aic.14113

Cerdá, J., Pautasso, P. C., & Cafaro, D. C. (2016). Optimizing Gasoline Recipes and Blending Operations Using Nonlinear Blend Models. *Industrial & Engineering Chemistry Research*, *55*(28), 7782–7800. https://doi.org/10.1021/acs.iecr.6b01566

Glismann, K., & Gruhn, G. (2001). Short-Term Planning of Blending Processes: Scheduling and Nonlinear Optimization of Recipes. *Chemical Engineering & Technology*, *24*(3), 246–249. https://doi.org/10.1002/1521-4125(200103)24:3<246::AID-CEAT246>3.0.CO;2-8

Jia, Z., & Ierapetritou*, M. (2003). Mixed-Integer Linear Programming Model for Gasoline Blending and Distribution Scheduling. https://doi.org/10.1021/IE0204843

Li, J., & Karimi, I. A. (2011). Scheduling Gasoline Blending Operations from Recipe Determination to Shipping Using Unit Slots. *Industrial & Engineering Chemistry Research*, *50*(15), 9156–9174. https://doi.org/10.1021/ie102321b

Li, J., Karimi, I. A., & Srinivasan, R. (2009). Recipe determination and scheduling of gasoline blending operations. *AIChE Journal*, *56*(2), NA-NA. https://doi.org/10.1002/aic.11970

Méndez, C., Grossmann, I., Harjunkoski, I., & Kaboré, P. (2006). A simultaneous optimization approach for off-line blending and scheduling of oil-refinery operations. *Computers & Chemical Engineering*, *30*(4), 614–634. https://doi.org/10.1016/J.COMPCHEMENG.2005.11.004

Merberg, G. N. (n.d.). *Evaluation of the ZX101™ Octane Analyzer*. Retrieved from

https://www.zeltex.com/papers/MerbergOctane.pdf

Monder, D. S. (2001). Real-Tome Optirniration of Gasoline Blending with Uncertain Parameters. Retrieved from http://www.nlc-bnc.ca/obj/s4/f2/dsk3/ftp04/MQ60472.pdf

Wei Wang, Zefei Li, Qiang Zhang, & Yankai Li. (2007). On-line optimization model design of gasoline blending system under parametric uncertainty. In *2007 Mediterranean Conference on Control & Automation* (pp. 1–5). IEEE. https://doi.org/10.1109/MED.2007.4433757

Yang, Y., & Barton, P. I. (2016). Integrated crude selection and refinery optimization under uncertainty. *AIChE Journal*, *62*(4), 1038–1053. https://doi.org/10.1002/aic.15075

Zhang, Y., Monder, D., & Fraser Forbes, J. (2002). Real-time optimization under parametric uncertainty: A probability constrained approach. *Journal of Process Control*, *12*(3), 373–389. https://doi.org/10.1016/S0959-1524(01)00047-6

Zhao, X., & Wang, Y. (2009). Gasoline Blending Scheduling Based on Uncertainty. In *2009 International Conference on Computational Intelligence and Natural Computing* (pp. 84–87). IEEE. https://doi.org/10.1109/CINC.2009.206

Chapter 3: Gasoline Blend Planning under Demand Uncertainty: Aggregate Supply-Demand Pinch Algorithm with Rolling Horizon

The contents of this chapter have been published in the *Industrial & Engineering Chemistry Research Journal*. Reprinted with permission from

Jalanko, M., & Mahalec, V. (2019). Gasoline Blend Planning under Demand Uncertainty: Aggregate Supply–Demand Pinch Algorithm with Rolling Horizon. *Industrial & Engineering Chemistry Research*, *59*(1), 281-29

Abstract

While most products from oil refineries are produced to meet contracted known demand, there is an additional uncertain demand which refineries can satisfy to generate extra profit. Using a deterministic model leads to suboptimal solutions since such a model fails to account for future additional uncertain demand when making a production plan. In this paper, a rolling horizon optimization approach is utilized to develop a production planning model under time-varying uncertainty in demand and applied to the gasoline blending problem. The model utilizes loss function formulation to account for expected revenue generated from meeting future uncertain demand when making a production plan for the current period. Our model considers uncertainty to vary with time; demand uncertainty for periods further into the future is higher. In the gasoline production planning application under demand uncertainty, our stochastic model makes the current period decisions (i.e. blend recipes) based on action of future uncertain demands, resulting in meeting higher products demands and higher profits compared to deterministic models. The model proposed is a mixed integer nonlinear programming (MINLP), and its size depends on the number of periods in the production horizon which leads to computational difficulties for cases with large number of periods. Difficulties are resolved by applying a supply-demand pinch algorithm to decompose the large MINLP model into two smaller models solved in sequence. The supply-demand pinch algorithm allows using local solver which results in 2000 to 3000-fold reduction in computation times compared to the full-space algorithm, while still achieving solutions within 0.04% from the full space algorithm solutions.

3.1. Introduction

Oil refinery plants are key elements in the supply chain of the petroleum industry. In the final section of the refinery, components produced by various upstream processes are mixed together to produce different gasoline products that meet varying specification. Gasoline products can yield 60%-70% of the total revenue of refineries; therefore, decreasing the gasoline blending costs or increasing revenues by satisfying higher product demands level can greatly impact the profitability of the refinery[1 2]. The gasoline blending production plan needs to consider multiple external and internal sources of uncertainty. The internal uncertainties come from variation in the upstream processes which result in variations in the feedstock flows or qualities over the planning horizon. The external uncertainties arise from fluctuations in the crudes market prices, gasoline products prices and demands. Therefore, it is important to consider these uncertainties when making a production plan for the gasoline blending section. This paper considers the gasoline blend planning problem under uncertainty in products demand. Considering uncertainty in product demand gives a better estimation of model parameters, which allows improvement of the solution validation and practicality. The products demand of each period consists of known contracted demand plus additional uncertain demand (spot market). The uncertainty in the additional products demand is time-varying uncertainty; where the uncertainty in each period increases as we look further into the future. Under the assumption that the additional demand in each period follows a normal distribution, the time-varying uncertainty can be modeled using higher and higher variances for subsequent time periods. In this paper we deal with the production planning problem, under the assumption that the scheduling problem can be solved as a second level problem using results obtained from the planning problem.

The gasoline blend planning problem requires computing optimal blend recipes and production plan for the planning horizon that minimizes the operation cost and meets the products demands, products qualities specifications, while being constrained by components supply rates and capacity limitations on the tanks and the blenders. Modeling an accurate gasoline blend problem leads to a large mixed integer nonlinear programming (MINLP) for cases with large number of periods; since the model size is dependent on the number of periods. The nonlinearity of the model stems from the nonlinear blending properties rules for some of the properties. The large MINLP problem leads to difficulties in computing the optimal solution with commercial solvers. Different deterministic gasoline blend planning and scheduling problems have been introduced

in the literature; Li et al.[3] developed a slot-based MILP formulation to solve the gasoline blend problem as a MILP instead of MINLP using blending indices and linear blending correlation to address nonlinear blending properties. Another approach to solve the gasoline blending problem is to decompose the problem into planning and scheduling. Glismann and Gruhn[4] presented an integrated optimization model of planning and scheduling. Jia and Ierapetritou[5] presented a MILP model for gasoline blending and distribution schedule with the assumption of having fixed preferred blend recipe. This allowed them to avoid the complexity of the MINLP model while opening the possibility of their solution being not optimal. That is because such an approach does not guarantee that blend recipe chosen is an optimum recipe. Mendez et al.[6] presented a novel MILP formulation that addresses the simultaneous optimization of blending and scheduling in oil refinery using discrete or continuous-time representation. Their original large MINLP problem formulation, where nonlinearity comes from nonlinear blending properties, is replaced by a sequential MILP approximation. Cerdá et al.[7] presented a novel continuous-time MILP formulation based on floating time slots to simultaneously optimize both blend recipes and scheduling operations. Their proposed approach computes optimal solutions at much lower computational costs compared to other previous work. Castillo and Mahalec[8] introduced a MINLP model to solve the gasoline blend planning problem and used the concept of supply-demand pinch (also known as inventory pinch) to decompose the MINLP planning model into two level models (NLP and MILP) solved in sequence to allow a great reduction in execution times compared to solving the full-space MINLP model. The supply-demand pinch concept uses an aggregation technique based on the aggregate demand curve to reduce the number of periods of top-level model which computes blending recipes, then the bottom-level model computes detailed production plan using the recipes computed at the top-level. The supply-demand pinch concept is explained further in section 5.

Rolling horizon procedures have proved their efficiency in solving production planning problem when uncertainty in the demands is considered. Rolling horizon approach solves the production planning problem in sequence of iterations. In each iteration, the production plan is solved for the current and future periods, and current period decisions are implemented, then a one timestep forward move is applied and the uncertain parameters are updated based on new available information before solving the next iteration. Two rolling horizon strategies have been used in the literature based on the prediction horizon length, a fixed-end strategy where the prediction

length is varying and moving-end strategy where the prediction length is constant[9]. The first approach suffers from a problem known "End of Horizon Effect"[10] which occurs because the last iteration has a single period and does not consider any future periods which is not realistic. In this work, a fixed end rolling horizon strategy is used, where products volume of the final period is assumed to be fixed to allow satisfying demands in future periods. Modeling the production plan as a rolling horizon problem without considering the uncertainty in products demands might lead to high inventory and low service level[11]. One way to improve the rolling horizon model under uncertainty in demands is to hold safety inventory stocks. The safety inventory stocks can improve the task of achieving the service levels required[12], but a constant stock inventory level might lead to high inventory level[13]. Another approach to deal with demands uncertainty in rolling horizon model is to use stochastic programming with discrete scenarios of the future demands[14], but such approach might lead to computational difficulties since the problem size grows rapidly with increasing number of planning periods and demands scenarios. Chance constraint formulation which requires satisfying constraint with specific probability is another approach to deal with uncertain demands in a rolling horizon model which assumes that the distribution of the uncertain demands is known[10].

In this paper, a fixed-end rolling horizon formulation is used to solve the gasoline blend planning problem under uncertainty in products demands. One of the paper's novelty is the way the uncertainty in demand is modeled: it is assumed that products demand consists of contracted demand which is certain and must be met, plus additional demand which is uncertain and follows a normal distribution. Also, the model considers time-varying uncertainty by assuming that the variance of the uncertain additional demand increases for periods further into the future, since it is harder to accurately predict the actual products demand for periods far into the future. The rolling horizon approach iteratively solves the planning problem in a rolling time horizon mode. In every iteration, a new production plan is computed for the planning horizon based on newly available information about the products demand. The purpose of using rolling horizon model is that the planning decisions for future periods cannot be optimal due to the uncertainty in future periods demands, therefore obtaining an initial plan for future periods and updating it every time new information about the demands are available can lead to near-optimal solutions. The optimality and feasibility of the updated plan are highly impacted by the previous plan decisions since these previous decisions determine both products and components inventory levels at the

beginning of the current horizon. Another contribution is extending the supply-demand pinch concept introduced by Castillo and Mahalec[8] from the deterministic case to the case where uncertainty in demands are considered and make it suitable for a rolling horizon formulation. Also, the model modification introduced allows using the supply-demand pinch concept even when not all demands can be satisfied which was an assumption required by the model Castillo and Mahalec[8] developed under the deterministic demand case. In summary, the aim of this work is to propose a rolling horizon approach that uses the supply-demand pinch aggregation algorithm and apply this approach to a large-scale gasoline blend planning MINLP model under products demand uncertainty.

The rest of this paper is organized as follows. The gasoline blend planning problem addressed in this work is presented in section 2, including detailed description and assumptions. Sections 3 and 4 introduce the revenue calculation under the demand uncertainty and the rolling horizon formulation, respectively. The full-space algorithm mathematical model of the gasoline blend planning problem is introduced in section 5. Section 6 introduces the supply-demand pinch concept, its mathematical models, and our proposed algorithm. Different case studies are introduced in section 7. Computations results and discussions for all case studies are presented in section 8. Finally, section 9 draws some conclusions and future directions.

3.2. Problem Statement

The gasoline blend planning problem presented by Castillo and Mahalec[8] is the base of our work, with modification to incorporate uncertainty in demands. The system studied in this paper falls under the application of short-term planning with 14-period horizon each can be considered as one day. Figure 1 shows the gasoline blending system which has seven feedstocks produced by upstream processes with fixed flow rate sent to be stored in individual component tanks. These components are mixed in a blender to produce three different products (U87, U91, U93), subject to meeting their quality specifications. Blended products are sent to six storage tanks before being shipped to satisfy products demand. Three of the products storage tanks are dedicated to the three products, while the other three tanks can store any product, but only one product at a certain time. The qualities monitored and required to be within some bounds for the product to be considered on-specs are aromatic content (ARO), benzene content (BEN), olefin content (OLF), research octane number (RON), motor octane number (MON), rapid vapour pressure (RVP), sulfur content (SI), and Specific gravity (SGI). The product demands are considered to

have contracted demand and additional demand that can be sold in the spot market. While contracted demand is certain and known at the point when the production plan is made and represents the minimum delivered demand required by the plant, the additional demand represents the spot market demand for future periods which is uncertain and cannot be predicted accurately. In this work, additional products demands are assumed to follow normal distribution. Furthermore, our model considers time-varying uncertainty where the variance of the normal distribution increases as we look further into the future. This is more suitable to represent the real-life case since our prediction of the additional demand is more likely to deviate from the realized demand for periods further into the future.

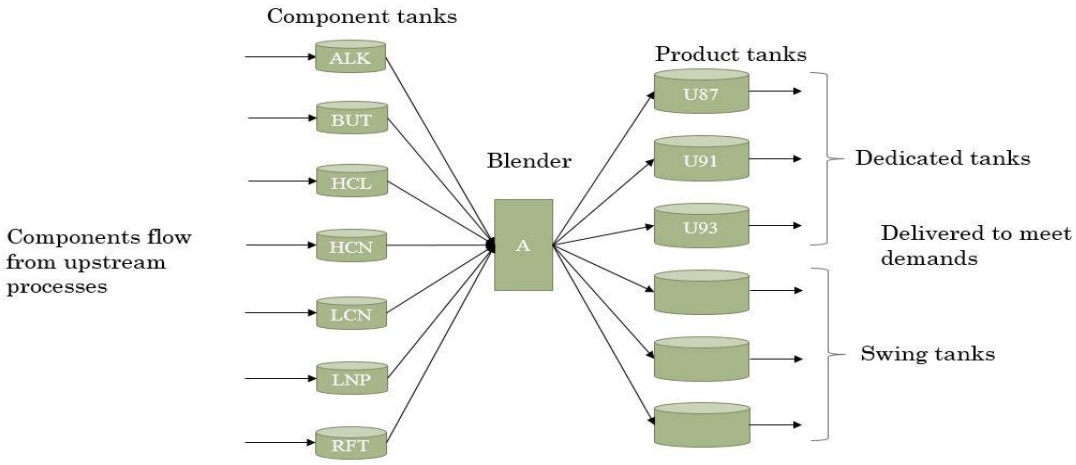

Figure 1. The gasoline blending system

The gasoline blend planning problem under demand uncertainty considered in this work can be described as follows.

Given:

(1) A predefined short-term planning horizon (0, H) that is divided into fixed durations time periods 1, 2… *M*.

(2) A set of blend components with known cost, initial inventories, fixed supply flow rates, and fixed quality properties along the planning horizon.

(3) A set of products with known selling prices, initial inventories, and their maximum and minimum quality specifications.

(4) A set of contracted demands for each product along the planning horizon.

(5) A set of components and product storage tanks with their minimum and maximum hold-ups.

(6) A set of blenders' maximum and minimum blending capacity.

Assumptions are:

(1) Refinery production plan model has determined components volumetric flow rates that are capable of meeting products contracted demands but does not have to meet all demand (contracted plus additional demands).

(2) Contracted products demand along the planning horizon are hard constraints (they must be met).

(3) Additional products demands are assumed to follow a normal distribution with increasing variance along the planning horizon and are not necessarily satisfied.

(4) Each order involved only one product and all orders are completed during the blending period.

(5) Perfect mixing occurs in the blenders.

(6) Blenders can blend all products, but only one product at a time.

(7) Component and product tanks may receive and feed simultaneously.

The objective of the algorithm is:

To maximize the refinery profit by maximizing the revenue from selling products and minimizing the cost of blending.

We need to compute:

(1) Volumes of each grade of gasoline produced and delivered at each period.

(2) Volumes of each component used to blend each gasoline grade product and which blender will carry out the blending process in each period (blending recipes).

(3) Inventory profiles for the components and gasoline products.

Subject to constraints on:

(1) Minimum delivered demands (contracted demands).

(2) Minimum and maximum components and products volumes in the tanks.

(3) Minimum and maximum products qualities specifications.

(4) Minimum and maximum blending capacity (including the idle time required to switch a blender from one service to another)

(5) Maximum delivery rate from blenders to product tanks.

3.3. Revenue Calculation Using Loss Function Under Demand Uncertainty

The gasoline blend model under uncertainty in products demands requires computing the expected revenue of future periods since there is no information available about the exact demand. The expected revenue under the case where we hold inventory can be represented as follow.

$$\text{revenue} = E[\sum_{x=0}^{\infty} C \min(P+I,x)]$$

Where E is the expectation operator, C is the selling price, P is the production, I is the inventory, and x is the products demand. Under the case where the selling prices are fixed, and demand follow a density distribution $\rho(x)$, the expected revenue can be reformulated as follows

$$\text{revenue} = C \int_0^{\infty} \min(P+I,x)\, \rho(x)\, dx$$

Under the case where the demand is lower than or equal production ($x \leq P+I$), the revenue can be computed as $C \int_0^{P+I} x\, \rho(x)\, dx$. While under the case where demand is higher than production ($x > P+I$), the revenue can be computed as $C \int_{P+I}^{\infty} (P+I)\, \rho(x)\, dx$. Therefore, the equation above can be reformulated as follows

$$\text{revenue} = C[\int_0^{P+I} x\, \rho(x)\, dx + \int_{P+I}^{\infty} (P+I)\, \rho(x)\, dx]$$

Assuming the mean of demand is $\mu = \int_0^{\infty} x\, \rho(x)\, dx$, the revenue equation can be reformulated as follows

$$\text{revenue} = C[\mu - \int_{P+I}^{\infty} x\, \rho(x) dx + \int_{P+I}^{\infty} (P+I)\, \rho(x)\, dx]$$

$$\text{revenue} = C[\mu - \int_{P}^{\infty} (x-(P+I))\, \rho(x)\, dx]$$

The term $\int_P^{\infty} (x-(P+I))\, \rho(x)\, dx$ is called the loss function. Under the case where the density function $\rho(x)$ is normally distributed with mean μ and standard deviation σ, the loss function can be written as follow.

$$\int_{P}^{\infty} (x-(P+I))\, \rho(x)dx = \sigma L(\frac{P+I-\mu}{\sigma})$$

where L(·) represents the standard loss function. The revenue can be formulated as follow.

$$\text{revenue}= C[\mu-\sigma L(\frac{P+I-\mu}{\sigma})]$$

From the above equation, the actual amount delivered to customers, S, is calculated as follow.

$$S = \mu-\sigma L(\frac{P+I-\mu}{\sigma})$$

The loss function $L(\frac{P+I-\mu}{\sigma})$ cannot be implemented in the optimization model directly. Therefore, the loss function should be approximated by a function that can be implemented in the optimization problem. Li and Hui[15] approximated the standard loss function using four polynomial function that has the form of a sixth-order polynomial, where $z = \frac{P+I-\mu}{\sigma}$ as shown below

$$L_l(z) = a_l+b_l z+ c_l z^2+d_l z^3+e_l z^4+f_l z^5+ g_l z^6 \qquad l = 1\text{-}4$$

Figure 2 shows that the four approximated functions bound the actual loss function from below. For the revenue maximization problem, the value of the loss function L(·) is to be minimized, therefore the loss function can be formulated as follow.

$$L(z) \geq L_l(z) \qquad l = 1\text{-}4$$

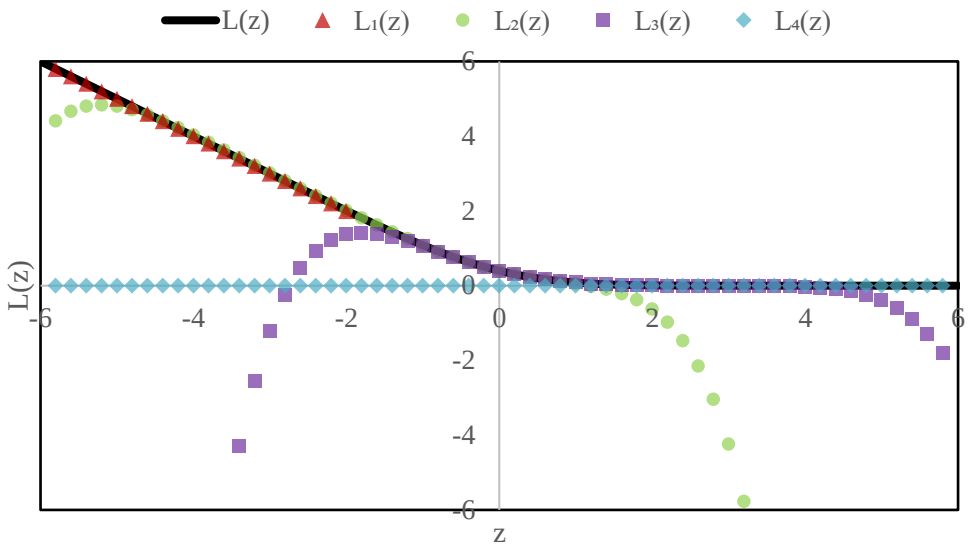

Figure 2. Nonlinear approximation functions of the standard loss function. The standard loss function (▬), and its four polynomial approximation functions: L_1 (▲), L_2 (●) L_3 (■) L_4 (♦).The revenue and loss function approximation equations introduced earlier can be integrated into the full-space and the supply-demand pinch models to make them consider uncertainty in the products demands.

3.4. Rolling Horizon Formulation

The rolling horizon model under products demand uncertainty assumes fixed-end planning horizon with discrete time periods, $m = 1, \ldots, M$. In fixed end rolling horizon, the number of periods decreases as we roll forward in the horizon. At the start of each period m, the additional demand of period m becomes known, and future periods demand get updated and the variance of these demands shrink since we are getting closer to reach these periods. A production plan for the current and future periods are computed to maximize the profit of current period and expected profit of future periods, and decisions of the current period m are implemented based on the result of the optimization problem. The time advances to the beginning of period m+1, with the final inventories of the previous period m becoming the initial inventory of the period m+1. Let s be defined as the current period and H defined as the last period in the production horizon, the rolling horizon algorithm is described as follows:

Step 1: s = 0, set the initial products and components inventories at period 0.

Step 2: s = s + 1, where products demand at period s are realized.

Step 3: solve the finite production planning horizon with the planning window H-s+1 for periods s to H.

Step 4: The decision variables obtained for period s are final decisions and the inventories at the end of period s are considered the initial inventories for the next planning window s+1 to H. While the decisions for the periods s+1 to H are tentative decisions.

Step 5: Return to step 2 and repeat until s = H

Our planning model considers planning horizon length with 14 periods (H=14) at the first iteration; therefore we require solving 14 optimization problems before stopping. Figure 3 shows the fixed end rolling horizon formulation used in our work.

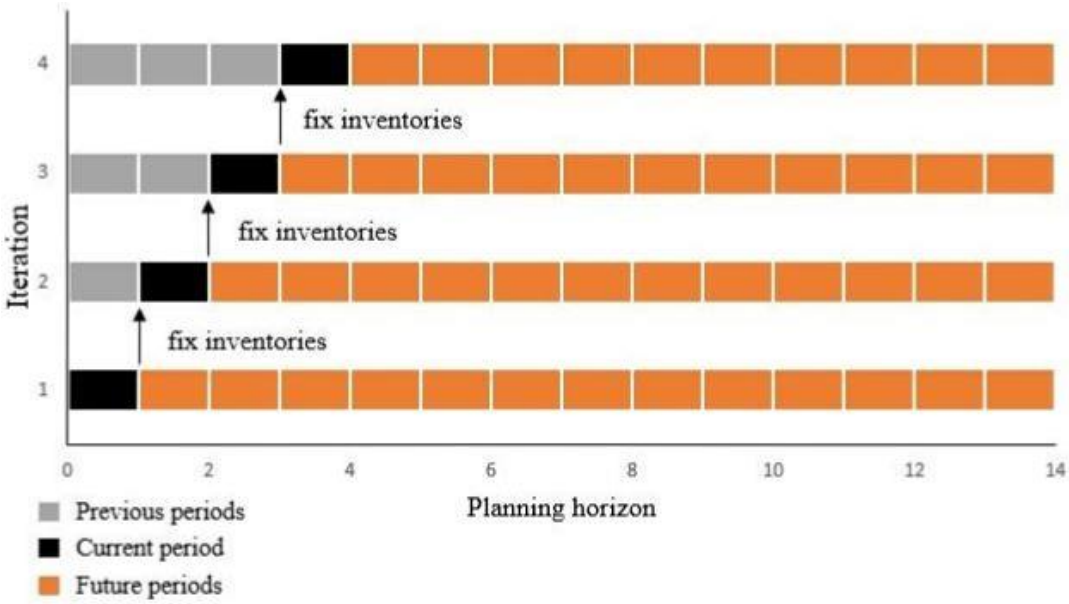

Figure 3. Fixed end rolling horizon algorithm

3.5. Full-Space Algorithm Mathematical Model

In this section, the full-space algorithm model under uncertainty in demands is presented. The full-space algorithm aims to compute a detailed production plan for the whole horizon using a single large model.

Our problem faces the challenge of having multiple optimal solutions; the same optimal profit from a single period run can be achieved using different blend recipes, production plan, and whether to satisfy product demands for current period or later future periods. Therefore, we solve two models in sequence; the first model aims to compute the optimal profit and the second fixes

that profit and maximize the amount delivered at the current period in order to maximize current period revenue. This is realistic since it is preferable to satisfy current period additional demand over satisfying additional future possible demand.

The full-space model considers a detailed model for both the current period and future periods. The objective function of the full space model, Eq. (1), maximizes the refinery profit. The revenues of the current and future periods are given by Eqs. (2) and (3) respectively. The cost of blending for the current and future periods are given by Eqs. (4) and (5) respectively.

$$\text{max. Profit} = \text{Current Revenue} + \text{Future Revenue} - \text{Current Blend Cost} - \text{Future Blend Cost} \tag{1}$$

Subject to

$$\text{Current Revenue} = \sum_{m\in \mathrm{MC}} \sum_{p} \left[\mathrm{SP}_p \ \mathrm{Deliver}_{p,m}^{\mathrm{pool}}\right] \tag{2}$$

$$\text{Future Revenue} = \sum_{m\in \mathrm{MF}} \sum_{\mathrm{p}} \left[\mathrm{SP}_p \ \mathrm{Deliver}_{p,m}^{\mathrm{pool}}\right] \tag{3}$$

$$\text{Current Blend Cost} = \sum_{m\in \mathrm{MC}} \sum_{(bl,p)\epsilon \mathrm{BP}} \sum_{i} \left[\mathrm{CC}_i \ \mathrm{V}_{i,p,m,bl}^{\mathrm{comp}}\right] \tag{4}$$

$$\text{Future Blend Cost} = \sum_{m\in \mathrm{MF}} \sum_{(bl,p)\epsilon \mathrm{BP}} \sum_{i} \left[\mathrm{CC}_i \ \mathrm{V}_{i,p,m,bl}^{\mathrm{comp}}\right] \tag{5}$$

The inventory balance equations for blending components, products pools, and products tank are given by Eqs. (6-8) respectively. The inventory constraints on components storage tanks are enforced by Eq. (9). The minimum products pool inventories are forced by Eq. (10). The inventory constraint in each products storage tanks is enforced by Eq. (11), where u(*j*,*p*,*m*) is a binary variable that takes a value of 1 if tank *j* is storing product *p* during period *m* and 0 otherwise. Eq. (12) ensures that only one product *p* is stored at tank *j* during period *m*. The products pool inventories are related to the individual product tanks inventories by Eq. (13).

$$\mathrm{V}_{i,m}^{\mathrm{comp\_in}} + \mathrm{V}_{i,m-1}^{\mathrm{bc}} - \mathrm{V}_{i,m}^{\mathrm{bc}} - \sum_{(bl,p)\epsilon \mathrm{BP}} \mathrm{V}_{i,p,m,bl}^{\mathrm{comp}} = 0 \qquad \forall i,\ m \in \mathrm{MA} \tag{6}$$

$$\sum_{bl\ \epsilon \mathrm{BP}} \mathrm{V}_{p,m,bl}^{\mathrm{blend}} + \mathrm{V}_{p,m-1}^{\mathrm{pool}} - \mathrm{V}_{p,m}^{\mathrm{pool}} - \mathrm{Deliver}_{p,m}^{\mathrm{pool}} = 0 \qquad \forall p,\ m \in \mathrm{MA} \tag{7}$$

$$\sum_{bl\ \epsilon \mathrm{BP}} \mathrm{V}_{j,p,m,bl}^{\mathrm{trans}} + \mathrm{V}_{j,p,m-1}^{\mathrm{pr}} - \mathrm{V}_{j,p,m}^{\mathrm{pr}} - \mathrm{Deliver}_{j,p,m}^{\mathrm{pr}} = 0 \qquad \forall (j,p) \in \mathrm{JP},\ m \in \mathrm{MA} \tag{8}$$

$$\mathrm{V}_{i}^{\mathrm{bc\_min}} \le \mathrm{V}_{i,m}^{\mathrm{bc}} \le \mathrm{V}_{i}^{\mathrm{bc\_max}} \qquad \forall i,\ m \in \mathrm{MA} \tag{9}$$

$$V_{p,m}^{pool} \geq V_{p}^{pool\_min} \qquad \forall p, m \in MA \qquad (10)$$

$$V_{j}^{pr\_min}\, u_{j,p,m} \leq V_{j,p,m}^{pr} \leq V_{j}^{pr\_max}\, u_{j,p,m} \qquad \forall (j,p) \in JP, m \in MA \qquad (11)$$

$$\sum_{p \in JP} u_{j,p,m} = 1 \qquad \forall j, m \in MA \qquad (12)$$

$$V_{p,m}^{pool} = \sum_{j \in JP} V_{j,p,m}^{pr} \qquad \forall p, m \in MA \qquad (13)$$

The number of products allowed to be blended in each period is enforced by Eq. (14). The binary variable $x_{p,bl,m}$ takes a value of 1 if product *p* is blended in blender *bl* during period *m* and 0 otherwise, while np_{bl} is an integer parameter that represents the allowed number of products to be blended. The blender capacity constraints are given by Eqs. (15-17), where the parameter $cit_{p,bl}^{blend\_min}$ represents the minimum idle time required by blender *bl* to process product *p*. The first equation enforces the maximum blender capacity assuming some idle time equivalent to the product of the maximum blending capacity and the minimum idle time required by the blender. The second and third equations impose constraints on the minimum and maximum blending rate for the blender respectively, where the parameter $ct_{p,bl}^{blend\_min}$ represents the minimum running time of blender *bl* for product *p*.

$$\sum_{p \in BP} x_{p,bl,m} \leq np_{bl} \qquad \forall bl, m \in MA \qquad (14)$$

$$\sum_{p \in BP} V_{p,m,bl}^{blend} + F_{bl}^{blend\_max} \sum_{p \in BP} (cit_{p,bl}^{blend\_min}\, x_{p,bl,m}) \leq F_{bl}^{blend\_max}\, t_{m} \qquad \forall bl, m \in MA \qquad (15)$$

$$V_{p,m,bl}^{blend} \geq F_{bl}^{blend\_min}\, ct_{p,bl}^{blend\_min}\, x_{p,bl,m} \qquad \forall (bl,p) \in BP, m \in MA \qquad (16)$$

$$V_{p,m,bl}^{blend} \leq F_{bl}^{blend\_max}\, t_{m}\, x_{p,bl,m} \qquad \forall (bl,p) \in BP, m \in MA \qquad (17)$$

In order to guarantee feasibility, the sum of running time of each blend plus the idle time during a single period *m* must be equal or less than the duration of this *m* period. $t_{p,m,bl}^{blend}$ is an integer variable representing the time required for a blend run, which is constrained by given Eqs. (18-21). The first constraint forces the running time of a blend to be greater than or equal than the minimum running time of blender *bl* for product *p*. Limits on the upper and lower running time for the blend run are given by Eqs. (19-20). while Eq. (21) ensures that running time of the blend

at *m* period plus the product changeover times (idle time) is less than or equal the length of period *m*.

$$t^{blend}_{p,m,bl} \geq ct^{blend\_min}_{p,bl}\, x_{p,bl,m} \qquad \forall(bl,p) \in BP,\ m \in MA \qquad (18)$$

$$t^{blend}_{p,m,bl} \geq \frac{V^{blend}_{p,m,bl}}{F^{blend\_max}_{bl}} \qquad \forall(bl,p) \in BP,\ m \in MA \qquad (19)$$

$$t^{blend}_{p,m,bl} \leq \frac{V^{blend}_{p,m,bl}}{F^{blend\_min}_{bl}} \qquad \forall(bl,p) \in BP,\ m \in MA \qquad (20)$$

$$\sum_{p \in BP} t^{blend}_{p,m,bl} + \sum_{p \in BP} (cit^{blend\_min}_{p,bl}\, x_{p,bl,m}) \leq t_m \qquad \forall bl,\ m \in MA \qquad (21)$$

Eq. (22) forces the volume of product p sent from blender *bl* to tank *j* to be equal or less than the maximum volume possible to blend in the blender. $v_{j,p,m,bl}$ is a binary variable that takes a value of 1 if tank *j* is receiving product *p* from blender *bl* during period *m*. Eq. (23) ensure that all blended products are sent to product tanks for all products during all periods. Eq. (24) states that volume blended of product *p* in blender *bl* can be sent to only one product tank during period *m*, while Eq. (25) ensures that this tank should either be empty or already stores that product.

$$V^{trans}_{j,p,m,bl} \leq F^{blend\_max}_{bl}\, t_m\, v_{j,p,m,bl} \qquad \forall(bl,p) \in BP,\ (j,p) \in JP,\ m \in MA \qquad (22)$$

$$\sum_{j \in JP} V^{trans}_{j,p,m,bl} = V^{blend}_{p,m,bl} \qquad \forall(bl,p) \in BP,\ m \in MA \qquad (23)$$

$$\sum_{j \in JP} v_{j,p,m,bl} \leq 1 \qquad \forall(bl,p) \in BP,\ m \in MA \qquad (24)$$

$$v_{j,p,m,bl} \leq u_{j,p,m} \qquad \forall(bl,p) \in BP,\ (j,p) \in JP,\ m \in MA \qquad (25)$$

Since the swing tanks can be used to store different products during different time periods, the binary variable $ue_{j,m}$ is used in Eqs. (26-27) which takes a value of 1 if a product transition has taken a place in product tank *j* at period *m*. Eq. (28) forces a product transition in product tank *j* to occur if the tank is empty at the end of the previous period.

$$ue_{j,m} \geq u_{j,p,m} - u_{j,p,m-1} \qquad \forall(j,p) \in JP,\ m \in MA \qquad (26)$$

$$ue_{j,m} \geq u_{j,p,m-1} - u_{j,p,m} \qquad \forall(j,p) \in JP,\ m \in MA \qquad (27)$$

$$V^{pr}_{j,p,m-1} \leq V^{pr\_max}_{j}\,(1 - ue_{j,m}) \qquad \forall(j,p) \in JP,\ m \in MA \qquad (28)$$

Eq. (29) states which product *p* is stored in tank components *j* at the start of the current period. Eq. (30) and (31) set the volume of the component and individual product tanks at the start of the current period respectively. Eq. (32) set the initial blender state, whether the blender is running to blend a product *p* or is idle at the start of time horizon.

$$\mathrm{u}_{j,p,m} = \mathrm{u}_{j,p}^{\mathrm{initial}} \qquad \forall(j,p) \in \mathrm{JP},\ m \in \mathrm{M0} \qquad (29)$$

$$\mathrm{V}_{i,m}^{\mathrm{bc}} = \mathrm{V}_{i}^{\mathrm{bc\_initial}} \qquad \forall i,\ m \in \mathrm{M0} \qquad (30)$$

$$\mathrm{V}_{j,p,m}^{\mathrm{pr}} = \mathrm{V}_{j,p}^{\mathrm{pr\_initial}} \qquad \forall(j,p) \in \mathrm{JP},\ m \in \mathrm{M0} \qquad (31)$$

$$\mathrm{x}_{p,bl,m} = \mathrm{x}_{p,bl}^{\mathrm{initial}} \qquad \forall(j,p) \in \mathrm{JP},\ m \in \mathrm{M0} \qquad (32)$$

The blend recipe computations are given by Eq. (33), which defines the blend recipe for each product (the fraction of the component used to blend each product). The fraction of each component *i* in a product *p* should sum to 1 for all products, periods, and blenders which is forced by Eq. (34). The minimum and maximum blend recipe for each product is forced by Eq. (35).

$$\mathrm{V}_{i,p,m,bl}^{\mathrm{comp}} = \mathrm{r}_{i,p,m,bl}\, \mathrm{V}_{p,m,bl}^{\mathrm{blend}} \qquad \forall i,(bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (33)$$

$$\sum_{\mathrm{i}} \mathrm{r}_{i,p,m,bl} = 1 \qquad \forall i,(bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (34)$$

$$\mathrm{r}_{i,p}^{\min} \leq \mathrm{r}_{i,p,m,bl} \leq \mathrm{r}_{i,p}^{\max} \qquad \forall i,(bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (35)$$

The volumetric and weight basis linear quality constraints are described by Eq. (36) and (37) respectively. $\mathrm{Q}_{i,s}^{\mathrm{bc}}$ represents the value of quality *s* of component *i*, where $\mathrm{Q}_{p,s}^{\mathrm{pr\_min}}$ and $\mathrm{Q}_{p,s}^{\mathrm{pr\_max}}$ are the minimum and maximum quality *s* specification for product *p* respectively. For the case where linear blending rules are assumed, only these two constraints are required for products qualities.

$$\mathrm{V}_{p,m,bl}^{\mathrm{blend}}\, \mathrm{Q}_{p,s}^{\mathrm{pr\_min}} \leq \sum_i (\mathrm{V}_{i,p,m,bl}^{\mathrm{comp}}\, \mathrm{Q}_{i,s}^{\mathrm{bc}}) \leq \mathrm{V}_{p,m,bl}^{\mathrm{blend}}\, \mathrm{Q}_{p,s}^{\mathrm{pr\_min}} \qquad \forall s = \text{linear blended qualities (volumetric basis)},(bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (36)$$

$$\sum_i (\mathrm{V}_{p,m,bl}^{\mathrm{blend}}\, \mathrm{Density}_i)\, \mathrm{Q}_{p,s}^{\mathrm{pr\_min}} \leq \sum_i (\mathrm{V}_{i,p,m,bl}^{\mathrm{comp}}\, \mathrm{Q}_{i,s}^{\mathrm{bc}}\, \mathrm{Density}_i) \leq \sum_i (\mathrm{V}_{p,m,bl}^{\mathrm{blend}}\, \mathrm{Density}_i)\, \mathrm{Q}_{p,s}^{\mathrm{pr\_max}} \qquad \forall s = \text{linear blended qualities (weight basis)},(bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (37)$$

For the case where nonlinear blending rules are considered, RVP is assumed to blend nonlinear based on Eq. (38), which can be transformed into linear form as shown in Eq. (39) to reduce computations times. While RON and MON properties are assumed to blend nonlinearly following the ethyl RT-70 models[16]. In the ethyl RT-70 model, the products RON and MON quality value are functions of the blend components sensitivity ($\text{sens}_i = \text{Q}^{\text{bc}}_{i,RON} - \text{Q}^{\text{bc}}_{i,MON}$), OLF, and ARO content. The model parameter values used by Singh et al.[16] are: $a_1 = 0.03224$, $a_2 = 0.00101$, $a_3 = 0$, $a_4 = 0.04450$, $a_5 = 0.00081$, and $a_6 = -0.0645$

$$\text{Q}^{\text{pr}}_{p,s,m,bl} = \left[\sum_i \text{r}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s}{}^{1.25}\right]^{0.8} \qquad \forall s = \text{"RVP"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (38)$$

$$\text{V}^{\text{blend}}_{p,m,bl}\ \text{Q}^{\text{pr\_min}}_{p,s}{}^{1.25} \le \sum_i \left(\text{V}^{\text{comp}}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s}{}^{1.25}\right) \le \text{V}^{\text{blend}}_{p,m,bl}\ \text{Q}^{\text{pr\_max}}_{p,s}{}^{1.25} \qquad \forall s = \text{"RVP"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (39)$$

$$\text{Q}^{\text{pr\_min}}_{p,s} \le \text{Q}^{\text{pr}}_{p,s,m,bl} \le \text{Q}^{\text{pr\_max}}_{p,s} \qquad \forall s = \text{"RON"},\ \text{"MON"},(bl,p) \in \text{BP},\ m \in \text{MA} \qquad (40)$$

$$\text{Q}^{\text{pr}}_{p,s,m,bl} = r^{\text{RON\_avg}}_{p,m,bl} + \text{a}_1\left[\text{sens}^{\text{RON\_avg}}_{p,m,bl} - r^{\text{RON\_avg}}_{p,m,bl}\ \text{sens}^{\text{avg}}_{p,m,bl}\right] + \text{a}_2\left[\text{Ol}^{\text{sq\_avg}}_{p,m,bl} - \text{Ol}^{\text{avg}}_{p,m,bl}{}^2\right] + \text{a}_3\left[\text{Ar}^{\text{sq\_avg}}_{p,m,bl} - \text{Ar}^{\text{avg}}_{p,m,bl}{}^2\right]^2 \qquad \forall s = \text{"RON"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (41)$$

$$\text{Q}^{\text{pr}}_{p,s,m,bl} = r^{\text{MON\_avg}}_{p,m,bl} + \text{a}_4\left[\text{sens}^{\text{RON\_avg}}_{p,m,bl} - r^{\text{MON\_avg}}_{p,m,bl}\ \text{sens}^{\text{avg}}_{p,m,bl}\right] + \text{a}_5\left[\text{Ol}^{\text{sq\_avg}}_{p,m,bl} - \text{Ol}^{\text{avg}}_{p,m,bl}{}^2\right] + \frac{a_6}{10000}\left[\text{Ar}^{\text{sq\_avg}}_{p,m,bl} - \text{Ar}^{\text{avg}}_{p,m,bl}{}^2\right]^2 \qquad \forall s = \text{"MON"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (42)$$

$$r^{\text{RON\_avg}}_{p,m,bl} = \sum_i \text{r}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s} \qquad \forall s = \text{"RON"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (43)$$

$$r^{\text{MON\_avg}}_{p,m,bl} = \sum_i \text{r}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s} \qquad \forall s = \text{"MON"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (44)$$

$$\text{sens}^{\text{avg}}_{p,m,bl} = \sum_i \text{r}_{i,p,m,bl}\ \text{sens}_i \qquad \forall (bl,p) \in \text{BP}, m \in \text{MA} \qquad (45)$$

$$\text{sens}^{\text{RON\_avg}}_{p,m,bl} = \sum_i \text{r}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s}\ \text{sens}_i \qquad \forall s = \text{"RON"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (46)$$

$$\text{sens}^{\text{MON\_avg}}_{p,m,bl} = \sum_i \text{r}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s}\ \text{sens}_i \qquad \forall s = \text{"MON"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (47)$$

$$\text{Ol}^{\text{avg}}_{p,m,bl} = \sum_i \text{r}_{i,p,m,bl}\ \text{Q}^{\text{bc}}_{i,s} \qquad \forall s = \text{"OLF"},\ (bl,p) \in \text{BP},\ m \in \text{MA} \qquad (48)$$

$$\mathrm{Ol}_{p,m,bl}^{\mathrm{sq\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,m,bl}\, \mathrm{Q}_{i,s}^{\mathrm{bc}^2} \qquad \forall s = \text{"OLF"},\ (bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (49)$$

$$\mathrm{Ar}_{p,m,bl}^{\mathrm{avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,m,bl}\, \mathrm{Q}_{i,s}^{\mathrm{bc}} \qquad \forall s = \text{"ARO"},\ (bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (50)$$

$$\mathrm{Ar}_{p,m,bl}^{\mathrm{sq\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,m,bl}\, \mathrm{Q}_{i,s}^{\mathrm{bc}^2} \qquad \forall s = \text{"ARO"},\ (bl,p) \in \mathrm{BP},\ m \in \mathrm{MA} \qquad (51)$$

The last set of equations considers the constraints required for modeling the demand uncertainty and the delivered amount of products. These equations are associated with the loss function derivation introduced in section 3, which allows us to consider the loss associated with not delivering future uncontracted products demand. In other words, these equations allow the model to compute the optimal plan, one that maximize the profit, while considering the uncertainty in the uncontracted demand. Eq. (52-53) are used to compute the loss sale expected in future periods, where the first equation computes the z-score and the second uses the 6th order polynomial approximation to compute the loss value of future demands. Since the uncontracted demand is assumed to be normal, the four 6th order polynomial approximation equations that compute the loss value of future uncontracted demand can be integrated into our model. Eq. (54) considers the current period volume delivered of products, which is equal to the contracted demand plus the certain additional demand minus unmet demand. Eq. (55) considers future periods delivered volumes of product based on loss function in additional demand and its variance. Eq. (56) ensures that our plan will satisfy the contracted demand in current and future periods. Eq. (57) ensures that the delivery amount of product *p* from product tank *j* during period m does not exceed the maximum delivery rate possible. Eq. (58) relates the sum of product volume delivered from all product tanks to the amount of product delivered during period *m*.

$$\mathrm{z}_{p,m} = \frac{\sum_{\mathrm{bl} \in \mathrm{BP}} \mathrm{V}_{p,m,bl}^{\mathrm{blend}} + \mathrm{V}_{p,m-1}^{\mathrm{pool}} - \mathrm{Demand}_{p,m}^{\mathrm{cont}} - \mathrm{Demand}_{p,m}^{\mathrm{add\_mean}}}{\sqrt{\mathrm{Demand}_{p,m}^{\mathrm{add\_Variance}}}} \qquad \forall p,\ m \in \mathrm{MF} \qquad (52)$$

$$\mathrm{L}_{p,m} = \mathrm{a}_{\mathrm{l}} + \mathrm{b}_{\mathrm{l}} \mathrm{z}_{p,m} + \mathrm{c}_{\mathrm{l}} \mathrm{z}_{p,m}^{2} + \mathrm{d}_{\mathrm{l}} \mathrm{z}_{p,m}^{3} + \mathrm{e}_{\mathrm{l}} \mathrm{z}_{p,m}^{4} + \mathrm{f}_{\mathrm{l}} \mathrm{z}_{p,m}^{5} + \mathrm{g}_{\mathrm{l}} \mathrm{z}_{p,m}^{6} \qquad \forall p,\ m \in \mathrm{MF},\ \mathrm{l}=1\text{-}4 \qquad (53)$$

$$\mathrm{Deliver}_{p,m}^{\mathrm{pool}} = \mathrm{Demand}_{p,m}^{\mathrm{cont}} + \mathrm{Demand}_{p,m}^{\mathrm{add\_mean}} - \mathrm{Demand}_{p,m}^{\mathrm{unmet}} \qquad \forall p,\ m \in \mathrm{MA} \qquad (54)$$

$$\mathrm{Deliver}_{p,m}^{\mathrm{pool}} = \mathrm{Demand}_{p,m}^{\mathrm{cont}} + \mathrm{Demand}_{p,m}^{\mathrm{add\_mean}} - \mathrm{L}_{p,m} \sqrt{\mathrm{Demand}_{p,m}^{\mathrm{add\_Variance}}} \qquad \forall p,\ m \in \mathrm{MF} \qquad (55)$$

$$\mathrm{Deliver}_{p,m}^{\mathrm{pool}} \geq \mathrm{Demand}_{p,m}^{\mathrm{cont}} \qquad \forall p, m \in \mathrm{MA} \qquad (56)$$

$$\mathrm{Deliver}_{j,p,m}^{\mathrm{pr}} \leq \mathrm{D}_{j}^{\mathrm{pr\_max}}\, \mathrm{t}_{m}\, \mathrm{u}_{j,p,m} \qquad \forall (j,p) \in \mathrm{JP}, m \in \mathrm{MA} \qquad (57)$$

$$\mathrm{Deliver}_{p,m}^{\mathrm{pool}} = \sum_{j \in \mathrm{JP}} \mathrm{Deliver}_{j,p,m}^{\mathrm{pr}} \qquad \forall p, m \in \mathrm{MA} \qquad (58)$$

The set of Eqs. (1-37) and (52-58) represents the full-space model for the linear blending rules case. While the set of Eqs. (1-58) not including Eq. (38) represents the full-space model for the nonlinear blending rules case.

3.6. Supply-Demand Pinch Algorithm and its Mathematical Models

In this section, the supply-demand pinch algorithm which decomposes the full-space MINLP model into two-level models in order to reduce execution times while still achieving an optimal or close to optimal solutions. The top-level of the supply-demand pinch reduces the number of planning periods by aggregating future periods based on the pinch points in the cumulative products demands. After computing the recipes from the top-level, these recipes are fixed and passed to the bottom-level model to compute optimal production plan if the recipes are feasible. The objective function of the top-level model is similar to the full space objective function, Eq. (1); it tries to maximize the profit of the refinery.

There are three sets of constraints in the top-level of the supply-demand pinch model. The first set of constraints represents the current detailed period equations, the second set of constraints represents the future aggregated period's equations and the third set of constraint link current and future period's equations.

The set of constraints that represents the current detailed period equations are similar to the constraints of the full space model, but they are written for the current period only (set MC) instead of all periods (set MA) and the constraints associated with only future periods are dropped. Therefore, the set of constraints of the detailed current period are represented by Eqs. (2), (4), (6-37), (54), and (56-58) for the linear blending rules case. The set of constraints of the detailed current period are represented by Eqs. (2), (4), (6-51), (54), and (56-58) and not including Eq. (38) for the nonlinear blending rules case.

The set of constraint that represents the future aggregated periods equations which replace all detailed future periods constraints are shown here. The future revenue and cost constraints are shown in Eqs. (59) and (60) respectively.

$$\text{Future Revenue} = \sum_{k}\sum_{p}\left[SP_p\ \text{Deliver}_{p,k}^{\text{pool\_agg}}\right] \tag{59}$$

$$\text{Future Blend Cost} = \sum_{k}\sum_{p}\sum_{i}\left[CC_i\ V_{i,p,k}^{\text{comp\_agg}}\right] \tag{60}$$

The inventory balance for the components and products pool inventories is shown in Eqs. (61) and (62) respectively. The components and pool products inventories capacities constraints are enforced by Eqs. (63) and (64) respectively.

$$V_{i,k}^{\text{comp\_in\_agg}} + V_{i,k-1}^{\text{bc}} - V_{i,k}^{\text{bc}} - \sum_{p} V_{i,p,k}^{\text{comp\_agg}} = 0 \qquad \forall i, k \tag{61}$$

$$V_{p,k}^{\text{blend\_agg}} + V_{p,k-1}^{\text{pool}} - V_{p,k}^{\text{pool}} - \text{Deliver}_{p,k}^{\text{pool\_agg}} = 0 \qquad \forall p, k \tag{62}$$

$$V_{i}^{\text{bc\_min}} \le V_{i,k}^{\text{bc}} \le V_{i}^{\text{bc\_max}} \qquad \forall i, k \tag{63}$$

$$V_{p}^{\text{pool\_min}} \le V_{p,k}^{\text{pool}} \le V_{p}^{\text{pool\_min}} \qquad \forall p, k \tag{64}$$

The blend recipe computations for future aggregated periods are enforced by Eqs. (65-67). The set of constraints are written in a way that enforces detailed periods within an aggregated period to have the same recipe.

$$V_{i,p,k}^{\text{comp\_agg}} = r_{i,p,k}\ V_{p,k}^{\text{blend\_agg}} \qquad \forall i, p, k \tag{65}$$

$$\sum_{i} r_{i,p,k} = 1 \qquad \forall i, p, k \tag{66}$$

$$r_{i,p}^{\min} \le r_{i,p,k} \le r_{i,p}^{\max} \qquad \forall i, p, k \tag{67}$$

The volumetric and weight basis linear quality constraints are described by Eq. (68) and (69) respectively. For the linear blending rules case, only these two constraints are required for products qualities.

$$V_{p,k}^{\text{blend\_agg}}\ Q_{p,s}^{\text{pr\_min}} \le \sum_{i}(V_{i,p,k}^{\text{comp\_agg}}\ Q_{i,s}^{\text{bc}}) \le V_{p,k}^{\text{blend\_agg}}\ Q_{p,s}^{\text{pr\_min}} \qquad \forall s = \text{linear blended qualities (volumetric basis)}, p, k \tag{68}$$

$$\sum_{i}(V_{p,k}^{\text{blend\_agg}}\ \text{Density}_i)\ Q_{p,s}^{\text{pr\_min}} \le \sum_{i}(V_{i,p,k}^{\text{comp\_agg}}\ Q_{i,s}^{\text{bc}}\ \text{Density}_i) \le \sum_{i}(V_{p,k}^{\text{blend\_agg}}\ \text{Density}_i)\ Q_{p,s}^{\text{pr\_max}} \qquad \forall s = \text{linear blended qualities (weight basis)}, p, k \tag{69}$$

For the nonlinear blending rules case, the nonlinear blending rules for RVP, RON, and MON are given by Eqs. (70-82).

$$\mathrm{V}_{p,k}^{\mathrm{blend\_agg}}\ \mathrm{Q}_{p,s}^{\mathrm{pr\_min}^{1.25}} \leq \sum_{\mathrm{i}} \left(\mathrm{V}_{i,p,k}^{\mathrm{comp\_agg}}\ \mathrm{Q}_{i,s}^{\mathrm{bc}^{1.25}}\right) \leq \mathrm{V}_{p,k}^{\mathrm{blend\_agg}}\ \mathrm{Q}_{p,s}^{\mathrm{pr\_max}^{1.25}} \qquad \forall s = \text{"RVP"}, p, k \qquad (70)$$

$$\mathrm{Q}_{p,s}^{\mathrm{pr\_min}} \leq \mathrm{Q}_{p,s,k}^{\mathrm{pr}} \leq \mathrm{Q}_{p,s}^{\mathrm{pr\_max}} \qquad \forall s = \text{"RON", "MON"}, p, k \qquad (71)$$

$$\mathrm{Q}_{p,s,k}^{\mathrm{pr}} = r_{p,k}^{\mathrm{RON\_avg}} + \mathrm{a}_1\left[\mathrm{sens}_{p,k}^{\mathrm{RON\_avg}} - r_{p,k}^{\mathrm{RON\_avg}}\ \mathrm{sens}_{p,k}^{\mathrm{avg}}\right] + \mathrm{a}_2\left[\mathrm{Ol}_{p,k}^{\mathrm{sq\_avg}} - \mathrm{Ol}_{p,k}^{\mathrm{avg}2}\right] + \mathrm{a}_3\left[\mathrm{Ar}_{p,k}^{\mathrm{sq\_avg}} - \mathrm{Ar}_{p,k}^{\mathrm{avg}2}\right]^2 \qquad \forall s = \text{"RON"}, p, k \qquad (72)$$

$$\mathrm{Q}_{p,s,k}^{\mathrm{pr}} = r_{p,k}^{\mathrm{MON\_avg}} + \mathrm{a}_4\left[\mathrm{sens}_{p,k}^{\mathrm{RON\_avg}} - r_{p,k}^{\mathrm{MON\_avg}}\ \mathrm{sens}_{p,k}^{\mathrm{avg}}\right] + \mathrm{a}_5\left[\mathrm{Ol}_{p,k}^{\mathrm{sq\_avg}} - \mathrm{Ol}_{p,k}^{\mathrm{avg}2}\right] + \frac{a_6}{10000}\left[\mathrm{Ar}_{p,k}^{\mathrm{sq\_avg}} - \mathrm{Ar}_{p,k}^{\mathrm{avg}2}\right]^2 \qquad \forall s = \text{"MON"}, p, k \qquad (73)$$

$$r_{p,k}^{\mathrm{RON\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}} \qquad \forall s = \text{"RON"}, p, k \qquad (74)$$

$$r_{p,k}^{\mathrm{MON\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}} \qquad \forall s = \text{"MON"}, p, k \qquad (75)$$

$$\mathrm{sens}_{p,m,bl}^{\mathrm{avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,\mathrm{k}}\ \mathrm{sens}_i \qquad \forall p, k \qquad (76)$$

$$\mathrm{sens}_{p,k}^{\mathrm{RON\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,\mathrm{k}}\ \mathrm{Q}_{i,s}^{\mathrm{bc}}\ \mathrm{sens}_i \qquad \forall s = \text{"RON"}, p, k \qquad (77)$$

$$\mathrm{sens}_{p,k}^{\mathrm{MON\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}}\ \mathrm{sens}_i \qquad \forall s = \text{"MON"}, p, k \qquad (78)$$

$$\mathrm{Ol}_{p,k}^{\mathrm{avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}} \qquad \forall s = \text{"OLF"}, p, k \qquad (79)$$

$$\mathrm{Ol}_{p,k}^{\mathrm{sq\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}^2} \qquad \forall s = \text{"OLF"}, p, k \qquad (80)$$

$$\mathrm{Ar}_{p,k}^{\mathrm{avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}} \qquad \forall s = \text{"ARO"}, p, k \qquad (81)$$

$$\mathrm{Ar}_{p,k}^{\mathrm{sq\_avg}} = \sum_{\mathrm{i}} \mathrm{r}_{i,p,k}\ \mathrm{Q}_{i,s}^{\mathrm{bc}^2} \qquad \forall s = \text{"ARO"}, p, k \qquad (82)$$

The last set of equations for the top-level supply-demand pinch algorithm considers the constraints required for modeling the demand uncertainty and the delivered products volume for aggregated future periods. Eq. (83-84) are used to compute the loss sale expected in future aggregated. Eq. (85) considers future periods delivered volumes of product based on loss function in aggregated additional demand and its aggregated variance. Eq. (86) ensures that our plan will satisfy the aggregate contracted demand in future periods.

$$z_{p,k} = \frac{V_{p,k}^{blend\_agg} + V_{p,k-1}^{pool} - Demand_{p,k}^{cont\_agg} - Demand_{p,k}^{add\_agg\_mean}}{\sqrt{Demand_{p,k}^{add\_agg\_Variance}}} \qquad \forall p, k \qquad (83)$$

$$L_{p,k} = a_l + b_l z_{p,k} + c_l z_{p,k}^2 + d_l z_{p,k}^3 + e_l z_{p,k}^4 + f_l z_{p,k}^5 + g_l z_{p,k}^6 \qquad \forall p, k, l=1\text{-}4 \qquad (84)$$

$$Deliver_{p,k}^{pool\_agg} = Demand_{p,k}^{cont\_agg} + Demand_{p,k}^{add\_agg\_mean} - L_{p,k}\sqrt{Demand_{p,k}^{add\_agg\_Variance}} \qquad \forall p, k \qquad (85)$$

$$Deliver_{p,k}^{pool\_agg} \geq Demand_{p,k}^{cont\_agg} \qquad \forall p, k \qquad (86)$$

The next set of equations represents the linkage between the detailed current period and aggregated future periods. The components and products pool inventories at the end of the current detailed period should equal to the initial inventories of the first aggregated period which enforced by Eqs. (87-88). Also, the current period is enforced to have the same recipe as the first aggregated period if there is no pinch point between them by Eq. (89).

$$V_{i,k-1}^{bc} = V_{i,m}^{bc} \qquad \forall i, k \in KI, m \in MC \qquad (87)$$

$$V_{p,k-1}^{pool} = V_{p,m}^{pool} \qquad \forall p, k \in KI, m \in MC \qquad (88)$$

$$r_{i,p,k} = r_{i,p,m,bl} \qquad \forall i, (bl,p) \in BP, k \in KI, m \in MC \qquad (89)$$

The objective function with the set of constraints (2), (4), (6-37), (54), and (56-58) written for current period (set MC), with constraints (59-69) and (83-89) represents the top-level model for the supply-demand pinch algorithm under the linear blending rules case. The objective function with the set of constraints (2), (4), (6-37), (39-51), (54), and (56-58) written for the current period (set MC), with constraints (59-89) represents the top-level model for the supply-demand pinch algorithm under the nonlinear blending rules case.

The bottom level of the supply-demand pinch algorithm uses blend recipes, products inventories at the end of supply-demand pinch periods, and amount of delivered products at current period computed from top-level period to check the feasibility of blend recipes and compute an optimal production plan for the detailed periods model. The bottom-level objective function is to maximize the refinery profit and ensure that the blend recipes computed at the top level are feasible as shown in Eq. (90). The slack variables introduced are forced to be positive and used

to detect if there are infeasibilities in components, products pool, or products tanks inventories. The penalty weight associated with components inventory slack variables are much greater than penalty weights associated with product inventory slack variables to force inventory infeasibilities to be on the products' sides. Also, the penalty weight associated with product infeasibility increase with the number of periods to delay any potential product inventory infeasibility as far into the future as possible.

$$\text{max. Profit} = \text{Current Revenue} + \text{Future Revenue} - \text{Current Blend Cost} - \text{Future Blend Cost} + \sum_{m \in \mathrm{MA}}\left[\sum_{i}\left[\mathrm{Penalty}^{\mathrm{bc}}\left(\mathrm{S}_{i,m}^{\mathrm{bc+}} + \mathrm{S}_{i,m}^{\mathrm{bc-}}\right)\right]\right] + \sum_{m \in \mathrm{MA}}\left[\sum_{p}\left[\mathrm{Penalty}_{m}^{\mathrm{pr}}\left(\mathrm{S}_{p,m}^{\mathrm{pool+}} + \mathrm{S}_{p,m}^{\mathrm{pool-}}\right)\right]\right] + \sum_{m \in \mathrm{MA}}\left[\sum_{p}\sum_{j}\left[\mathrm{Penalty}_{m}^{\mathrm{pr}}\left(\mathrm{S}_{j,p,m}^{\mathrm{pr+}} + \mathrm{S}_{j,p,m}^{\mathrm{pr-}}\right)\right]\right] \quad (90)$$

The constraints of the bottom-level model are similar to the constraints of the full space model with three modifications: (1) drop the constraints associated with products qualities, recipe computation and products amount delivered in current period, since recipes used at this level are fixed from the top-level model solution which is known to satisfy the qualities constraints and amount delivered of the current period are fixed. (2) Modify some of the constraints to include the slack variables to detect where inventory infeasibility occurs. (3) Add constraints to ensure that the sum of delivered product amount in future detailed periods equals the delivered amount computed of the aggregated periods. The constraints associated with qualities computations, recipe computations and products delivered amount of current period, Eqs. (34-51) and (54) are dropped, and Eq. (33) is replaced with Eq. (91) which uses fix recipes computed at the top level.

$$\mathrm{V}_{i,p,m,bl}^{\mathrm{comp}} = \mathrm{r}_{i,p,k}^{\mathrm{fix}}\mathrm{V}_{p,m,bl}^{\mathrm{blend}} \qquad \forall i,\ (bl,p) \in \mathrm{BP}, (m,k) \in \mathrm{MK} \quad (91)$$

Eqs. (6-8) are modified to include slack variables as shown by Eq. (92-94). Eq. (95) ensures that the sum of delivered products of future detailed periods is equal to the sum of delivered products volume of future aggregated periods corresponding to them.

$$\mathrm{V}_{i,m}^{\mathrm{comp\_in}} + \mathrm{V}_{i,m-1}^{\mathrm{bc}} - \mathrm{V}_{i,m}^{\mathrm{bc}} - \sum_{(bl,p)\in \mathrm{BP}}\mathrm{V}_{i,p,m,bl}^{\mathrm{comp}} + \mathrm{S}_{i,m}^{\mathrm{bc+}} - \mathrm{S}_{i,m}^{\mathrm{bc-}} = 0 \qquad \forall i,\ m \in \mathrm{MA} \quad (92)$$

$$\sum_{bl \in \mathrm{BP}}\mathrm{V}_{p,m,bl}^{\mathrm{blend}} + \mathrm{V}_{p,m-1}^{\mathrm{pool}} - \mathrm{V}_{p,m}^{\mathrm{pool}} - \mathrm{Deliver}_{p,m}^{\mathrm{pool}} + \mathrm{S}_{p,m}^{\mathrm{pool+}} - \mathrm{S}_{p,m}^{\mathrm{pool-}} = 0 \qquad \forall p,\ m \in \mathrm{MA} \quad (93)$$

$$\sum_{bl \in \mathrm{BP}}\mathrm{V}_{j,p,m,bl}^{\mathrm{trans}} + \mathrm{V}_{j,p,m-1}^{\mathrm{pr}} - \mathrm{V}_{j,p,m}^{\mathrm{pr}} - \mathrm{Deliver}_{j,p,m}^{\mathrm{pr}} + \mathrm{S}_{j,p,m}^{\mathrm{pr+}} - \qquad \forall (j,p) \in \mathrm{JP},\ m \in \mathrm{MA} \quad (94)$$

$S_{j,p,m}^{pr-}=0$

$$\sum_{m \in MK} \text{Deliver}_{p,m}^{pool} = \text{Deliver}_{p,k}^{pool\_agg} \qquad \forall p, k \qquad (95)$$

The objective function (90) and set of constraints (2-5), (9-32), (52-53), (55-58), and (91-95) represent the bottom-level model of the supply-demand pinch algorithm for both linear and nonlinear blending rules. The bottom-level model is the same for both blending rules cases, since the fixed recipes used at this level have already been ensured to meet products qualities whether linear or nonlinear blending rules are assumed and there are no qualities computations constraints needed at this level.

The full-space MINLP algorithm introduced in section 5 can be used to solve the multi-period gasoline blend planning problem under uncertainty in demands using a rolling horizon method introduced in section 4. The full-space MINLP algorithm assumes detailed decisions for current and future periods. This algorithm faces computations difficulties when many periods are considered due to the nonlinearities stemming from the nonlinear blending rules equations and loss function associated with the uncertainty in products demands in future periods. Therefore, the number of periods can be greatly reduced by aggregating the future periods based on the supply-demand pinch concept introduced by Castillo and Mahalec[8]. The supply-demand pinch concept is an aggregation and disaggregation techniques that allow us to decompose the large MINLP model into two smaller models (top-level and bottom-level models) solved in sequence. The supply-demand pinch concept requires identifying the points where demands are at peaks by constructing a cumulative total demand (CTD) curve which is obtained by adding the cumulative demands of all products along the planning horizon. Then a cumulative total average production (CATP) curve is constructed as the tightest piecewise linear overestimation of CTD with a starting point representing the initial products inventory. The points where both curves intersect are called the supply-demand pinch point, which represents the point of times where the aggregate demands have peaked. At the supply-demand pinch points, the slope of the CATP curve changes as shown in Figure 4. The hypothesis is that the optimal blend recipe for time periods between the supply-demand pinch points (demands peek) is constant. For the nonlinear blending rules model, this leads to solutions which are very close to the global optimum[8]. The aggregate blend between the two pinch points is the lowest possible cost blend since it assumes that the blend components will be available just in time when they are required to produce the

required amount of products to meet the demands at a given point in time between the pinch points. If the recipe computed from an aggregate solution can be used all along the horizon between the two pinch points, we are guaranteed to have the lowest cost blend. If it is not possible at any point in time between the pinch points to produce the required amount of product based on the aggregate recipe due to constraints on blend rates, it means that at that time there is not enough of one or more blend components. In order to resolve this infeasibility, the aggregate time period is subdivided at the point of infeasibility and the problem is resolved. In this work, since products demand consist of contracted demand (certain demand) and additional demand (uncertain part), the CTD curve is constructed from contracted demand plus the mean of the additional demand (expected uncertain demand), and the supply-demand pinch points are obtained based on that CTD curve.

The production plan is solved at each iteration using the supply-demand pinch algorithm, where the top-level model is constructed using a detailed model for the current period and aggregated periods based on the supply-demand pinch concept for future periods. This allows us to reduce the number of future periods greatly compared to the number of periods of the full-space model as shown in Figure 5. The gasoline blend planning problem solved using the supply-demand pinch concept frameworks is shown in Figure 6. The top-level model is referred to as PT(MC, k), with the MC and k representing the parameters for the current detailed period and future aggregate periods respectively. The middle-level model is referred to as $PM(MC)$, which maximize the profit of the current detailed period MC. The bottom-level model is referred to as $PB(m)$, which ensures that feasibility of blending recipes of detailed periods m.

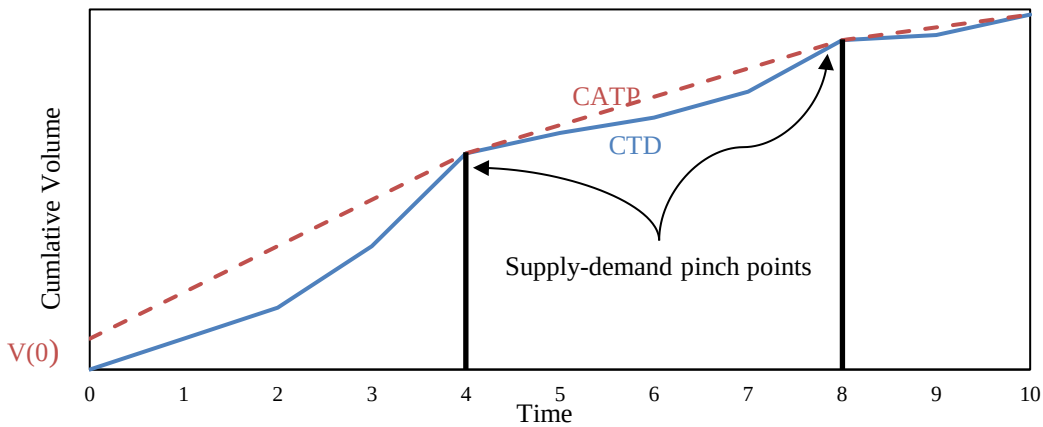

Figure 4. Example of supply-demand pinch points obtained from constructing cumulative total demand (CTD) curve and cumulative average total production (CATP) curve

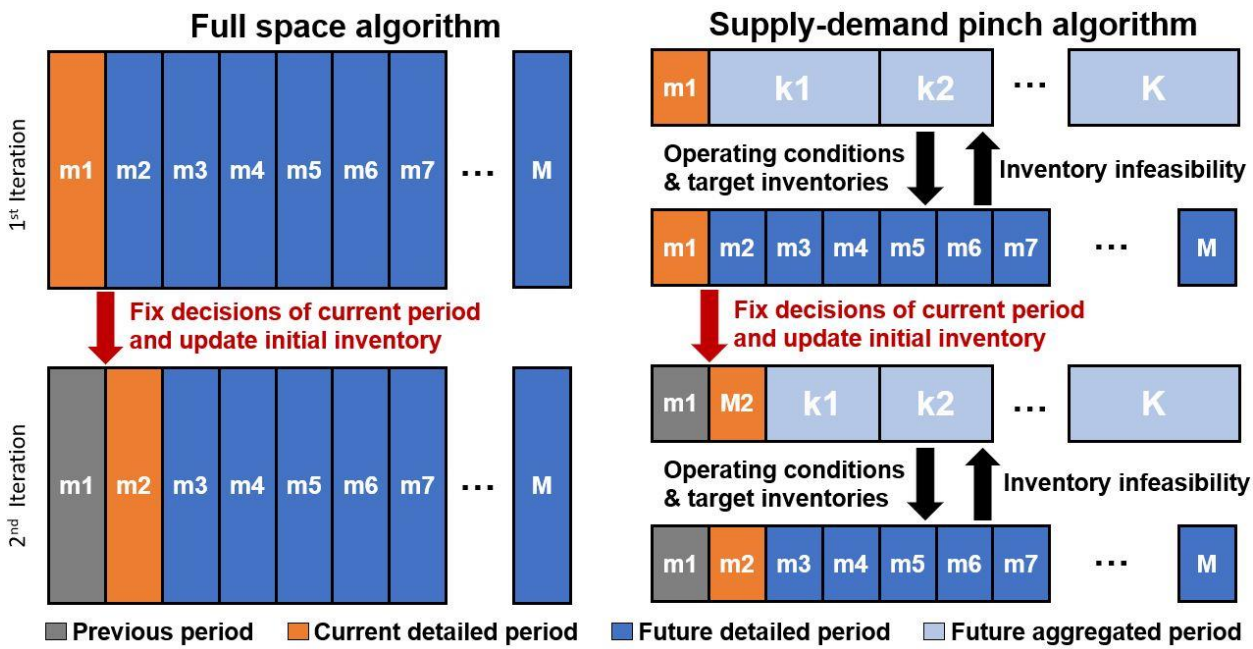

Figure 5. Full-space and supply-demand pinch algorithm with rolling horizon

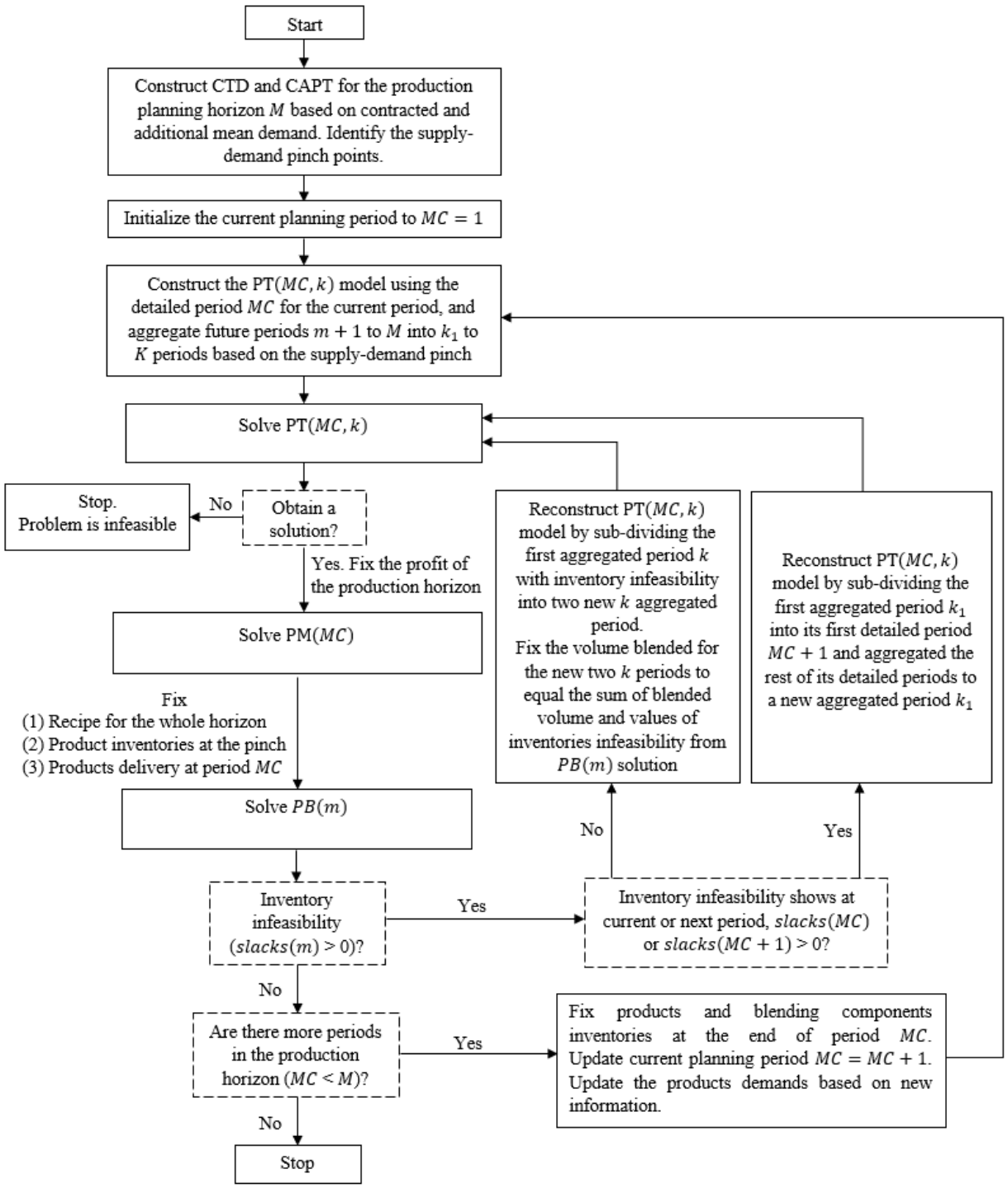

Figure 6. Flow chart of the rolling horizon supply-demand pinch algorithm under demand uncertainty

3.7. Case Studies

In this work, three different contracted demand patterns have been considered with different number of supply-demand pinch points. For each contracted demand pattern, three different cases solved with different additional demand generated randomly for both linear and nonlinear blending rules. Therefore, 18 different cases have been solved (9 assumes linear blending rules and 9 assumes nonlinear blending rules). All these cases have been solved using the full space algorithm and the supply-demand pinch algorithm with different solvers to compare the solution and computation times of both algorithms. All cases consider 14-period horizon with only 1 blending unit. The additional demand means values have been assumed to be 20% of the contracted demand initially. Then these additional demands for each period have been adjusted by generating a random number from their current mean and variance. Therefore, products demand further into the future gets adjusted more times compared to demand closer into the future. The mean additional demand generated at each iteration is used to generate demand of the second iteration (when the second period of the horizon is our current period). One assumption that has been considered is that the demand of the last period does not get updated from its value from the previous iteration. The reason for this assumption is that when we reach to the last period, revenue computed from that period is highly impacted by previous periods recipes used. Therefore, the final revenue from the rolling horizon model might highly vary based on what recipes have been used in previous periods, which is out of the scope of this paper.

The nonlinear blending rules cases have been solved using the actual RON, MON and RVP values provided by Castillo and Mahalec[8]. While the linear blending rules cases have been solved using the qualities indices by first converting the actual values into their index's values using octane and RVP index correlations given by Li et al.[3] Equations (96-98) represent the index correlations for octane (RON and MON) and RVP.

$$\text{Octane Index} = \text{Octane Number} + 11.5 \qquad 0 \leq \text{octane number} \leq 85 \tag{96}$$

$$\text{Octane Index} = \mathrm{e}^{0.0135(\text{Octane Number})+3.422042} \qquad \text{octane numner} \geq 85 \tag{97}$$

$$\text{RVP Index} = \mathrm{e}^{1,14\log(100\ \text{RVP})} \tag{98}$$

Figures 7-9 present the cumulative contracted plus additional mean total demand (CTD) and cumulative average total production (CATP) for the three different demand patterns. These CTD and CATP enable us to identify the supply-demand pinch points. The CTD curve includes the

inventory required at the end of the rolling horizon, and the initial CATP point is determined from the initial inventory minus the minimum tank inventory. Note that under the two assumptions (1) additional demand is only 20% of contracted demand and (2) additional demand is initially proportional to contracted demand, the supply-demand pinch points obtained from considering both contracted and additional demand are most likely to have the same position as when only contracted demand is considered. This allows us to construct the curve only once at the beginning of the horizon without the need to reconstruct after every iteration since the pinch points are most likely to stay in their position. Even if that was not the case, the algorithm itself will handle a slight movement in pinch by showing infeasibilities when the bottom level of the supply-demand pinch model is solved. The three demand patterns considered have zero, one, and two supply-demand pinch points respectively as shown in figures 7-9.

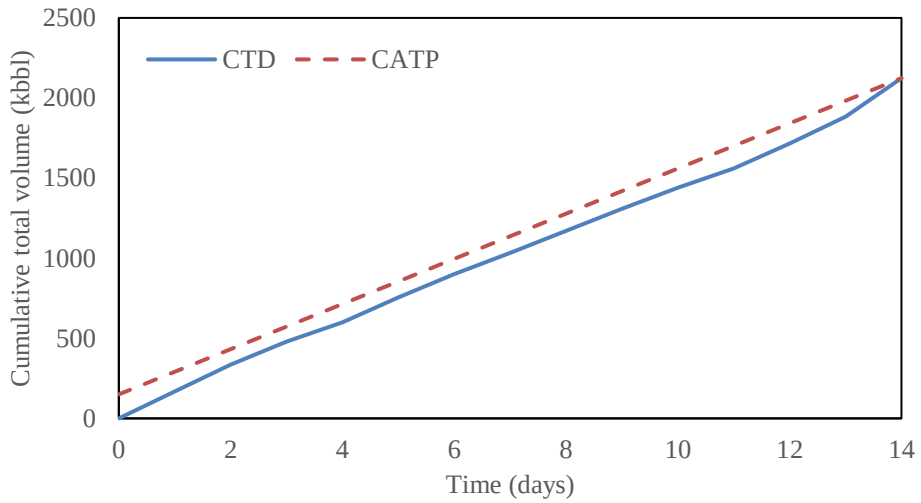

Figure 7. CTD and CATP for demand pattern 1 (case 1-3)

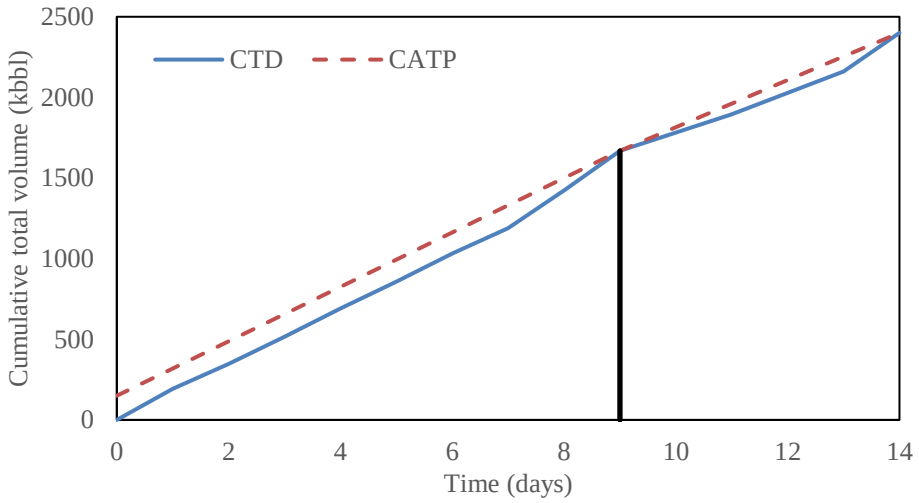

Figure 8. CTD and CATP for demand pattern 2 (case 4-6)

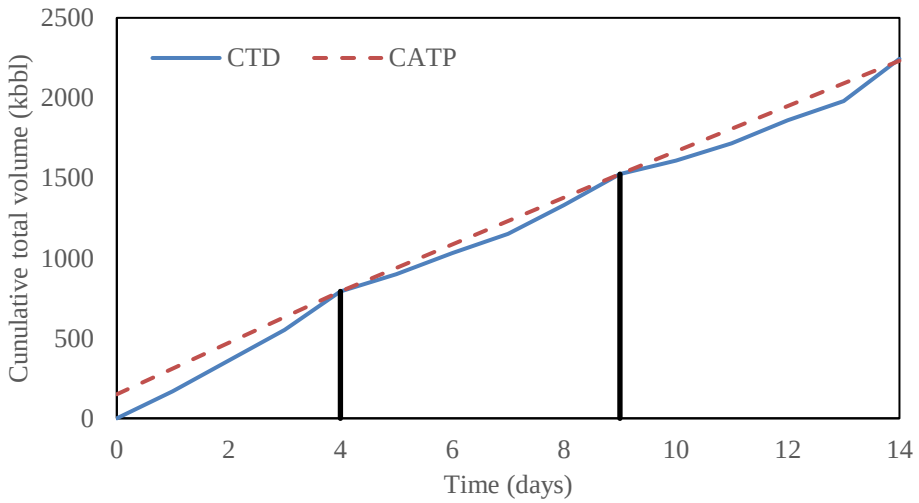

Figure 9. CTD and CATP for demand pattern 3 (case 7-9)

3.8. Results and Discussion

All case studies have been implemented in GAMS 25.1.3 software. MINLP models have been solved using ANTIGONE 1.1 as a global solver or DICOPT 2 as a local solver, with CPLEX 12.8 being used as the MILP solver and CONOPT 4.06 as the NLP solver. All problems have been solved using a desktop computer (Intel ® Core™ i5-4690K CPU, 3.50 GHz, and 8.0 GB RAM) running Windows 10. The stopping criteria of 0.001% optimality gap or maximum computational time of 3600 seconds have been used.

The linear blending rules cases have been solved using the full-space algorithm and the supply-demand pinch algorithm using Antigone solver. The nonlinear blending rules cases were solved using 4 different strategies as explained here:

- First strategy: solves the production planning problem using the full space algorithm, where the top and bottom models are solved using Antigone solver.
- Second strategy: an approximation strategy which solves the production planning problem using the full-space algorithm, where top and bottom models are solved using DICOPT local solver. This strategy solves the top-level of the full space algorithm then fix the profits and the volume of products blended compute from top-level model then solves the bottom-level model which maximize the demand delivered in current period (maximize current period revenue). The reason for fixing the volume blended is that the DICOPT solver fails to converge to a feasible solution when the amount of products blended were left as variable.
- Third strategy: solves the production planning problem using supply-demand pinch algorithm, where all three models are solved using Antigone solver.
- Fourth strategy: solves the production planning problem using supply-demand pinch algorithm, where all the three models are solved using DICOPT solver.

Tables 1 and 2 present the models' size for the first iteration of all different cases for linear and nonlinear blending rules, respectively. These tables show comparison between the models' size for the two algorithms (full-space and supply demand pinch) and for the case of linear and nonlinear blending rules. For the full-space algorithm, both first and second models have the same model size for all cases since they all use 14 periods model (1 current detailed period plus 13 future detailed periods). The second model in the full-space algorithm has one more equation and one more continuous variable compared to the first model which stems from the equation associated with maximizing the revenue of the current period. For the supply-demand pinch algorithm, the model size of the first and second models in the supply-demand pinch algorithm increases with the number of supply-demand pinch points. The number of periods of each first and second model in the supply-demand pinch algorithm represents the current detailed period, plus the future aggregated periods (ie: cases 4-6 have three total periods; one current detailed period and two aggregated future periods, since we have one supply-demand pinch point). For

the third model of the supply-demand pinch algorithm, all cases have the same model size since they all have 14 periods. Also, the model size for the nonlinear blending rules models compared to linear blending rules models have higher number of equations, continuous variables, and nonlinear terms. Since all model size shown are for the model in the first iteration, it is important to note that the model size of all detailed models reduces as we roll forward in the horizon, while the model size of all aggregated models reduces when the current period represents the last period of an aggregated period.

Table 1. Full-space and supply-demand pinch models' size in the first iteration for linear blending rules

Algorithm	**Model**	**Cases**	**# periods**	**# equations**	**# continuous variables**	**# nonlinear term**	**# Discrete variables**
Full-Space	First Model	All cases	14	4358	1982	666	504
	Second Model		14	4359	1983	666	504
Supply-demand Pinch	First Model	Cases 1-3	2	498	255	90	39
	Second Model		2	499	256	90	39
	Third Model		14	3022	2432	78	504
	First Model	Cases 4-6	3	687	343	138	60
	Second Model		3	688	344	138	60
	Third Model		14	3028	2432	78	504
	First Model	Cases 7-9	4	876	431	186	81
	Second Model		4	877	432	186	81
	Third Model		14	3034	2432	78	504

Table 2. Full-space and supply-demand pinch models' size in the first iteration for nonlinear blending rules

Algorithm	Model	Cases	# periods	# equations	# continuous variables	# nonlinear term	# Discrete variables
Full-Space	First Model	All cases	14	4610	2402	1002	504
	Second Model		14	4611	2403	1002	504
Supply-demand Pinch	First Model	Cases 1-3	2	558	315	138	39
	Second Model		2	559	316	138	39
	Third Model		14	3022	2432	78	504
	First Model	Cases 4-6	3	777	433	210	60
	Second Model		3	778	434	210	60
	Third Model		14	3028	2432	78	504
	First Model	Cases 7-9	4	996	551	282	81
	Second Model		14	4610	2402	1002	504
	Third Model		14	4611	2403	1002	504

Tables 3 and 4 summarize computational results for both full-space and supply-demand pinch algorithms/strategies for all cases under linear and nonlinear blending rules respectively. The first iteration solutions represent a single iteration run which can give a fair comparison between the different solution algorithms/strategies since they all have the same initial components and products inventories, while the rolling horizon solutions represent the solutions obtained from rolling forward in the horizon until the final planning period is reached. The computational results aim to evaluate the advantage of using the supply-demand pinch algorithm in terms of profit solutions and execution times over the full-space algorithm under linear and nonlinear blending rules. When comparing the profit solutions and execution times obtained under linear blending rules versus nonlinear blending rules, cases under linear blending rules give higher profits but have shorter execution times. This means that using linear blending rules might result

in producing off-specs products since nonlinear blending rules are more accurate model for how the blending in real-life occurs.

When comparing the profit solutions and execution times under linear blending rules obtained from the first iteration, the profits solutions are the same for cases 1-3 and 7-9 and are within 0.001% for cases 4-6 between the two algorithms, but the supply-demand pinch algorithm gives 5 to 150-fold reduction in execution times compared to the full-space algorithm for different cases. The slight difference in the profits between the two algorithms for cases 4-6 are within the optimality gap defined. For rolling horizon solutions, the supply-demand pinch algorithm profit solutions are the same as the full-space algorithm for cases 1-3 and 6-9, and within 0.001% for cases 4 and 5. The supply-demand pinch algorithm gives 5 to 30-fold reduction in execution times compared to the full-space algorithm for different cases. These results show that using the supply-demand pinch algorithm under rolling horizon formulation can compute profit solutions similar to the full-space algorithm with great reduction in execution times under the linear blending rules.

Under the nonlinear blending rules, the first iteration profit solutions and execution times results can give a fair comparison between the four different solution strategies. The first iteration solutions obtained from the four different strategies will be compared then the rolling horizon solutions will be compared after. Strategy 1 which is based on the full-space algorithm shows that the global solver (Antigone) has failed to close the optimality gap for all different cases. The optimality gap ranges from 0.8 to 1.8% for the different cases. Strategy 1 is used as the base case, so the other three strategies profit solutions and execution times will be compared to strategy 1. Strategy 2 which is based on the full-space algorithm shows that the local solver (DICOPT) can achieve profits solutions within the optimality gap (0.001%) from strategy 1 for all cases. Also, the solver can close the optimality gap and the execution times are less than 100 seconds for all cases. Strategy 3 which is based on the supply-demand pinch algorithm shows that the Antigone solver might still struggle to close the optimality gap even with the supply-demand pinch algorithm for most cases (cases 1-3 and 4-6) and profits solutions computed are lower compared to strategy 1. Strategy 4 which is based on the supply-demand pinch algorithm shows that the DICOPT solver can close the optimality gap for all cases and provide the lowest execution times compared to all the other three strategies but has lower profits solutions compared to strategy 1. The optimal solutions for strategies 3 and 4, which uses the supply-

demand pinch algorithm, for all cases are within 0.04% lower compared to strategy 1. This deteriorates in profits solutions for strategies 3 and 4 stems from the constraint of the supply-demand pinch model that restrict all aggregated periods to single recipe which impacts the expected revenue and cost computed at the first iteration. This reduction in optimal solutions is small and come with the advantage of great reductions in execution times, especially when DICTOP solver is used.

For the rolling horizon solutions, the solutions from all strategies are impacted by the amount of products blended and delivered, and the blending recipes used in previous periods since they impact components and products inventory levels. For that reason, optimal solution of the rolling horizon for all four strategies are not consistent with the different cases. In general, the strategy 1 gives the best profit solutions in most cases, but this strategy suffers from difficulty in closing the optimality gap for all cases causing it to have high execution times. strategy 2 can close the optimality gap and allows great reduction in execution times compared to strategy 1, but this strategy fixes the volume blended at bottom level model which is used to maximize the current period revenues based on the top-level solution. This limitation results in not satisfying early possible demands compared to strategy 1 as shown in figure 10. Therefore, while strategy 2 gives great reductions in execution times compared to strategy 1, it suffers from lower optimal solutions in general since it might not be able to prioritize current period guaranteed revenues over future uncertain revenues. This can be seen in the profit solutions computed for cases 2 and 7 which are 0.18% and 0.14% lower compared to strategy 1 respectively. The rolling horizon solutions for strategy 3 shows that execution times are reduced by 2 to 5-fold only compared to strategy 1 for the different cases, since this strategy struggles in closing the optimality gap. This can deteriorate the profit solutions computed as shown in case 7 which is 0.15% lower compared to strategy 1. Strategy 4 which uses the supply-demand pinch with local solver (DICOPT) allows on average 500 to 1000-fold reduction in execution times compared to strategy 3 and compute better profit solutions for most cases. Also, strategy 4 achieves solutions within 0.04% compared to strategy 1 for all cases, which shows it is more consistent then strategies 2 and 3. This small reduction in profit solutions comes with the advantage of 2000 to 3000-fold reduction in execution times. This shows that the big advantage of using the supply-pinch algorithm comes from enabling the use of local solver to compute solutions close to optimal with great reduction in execution times.

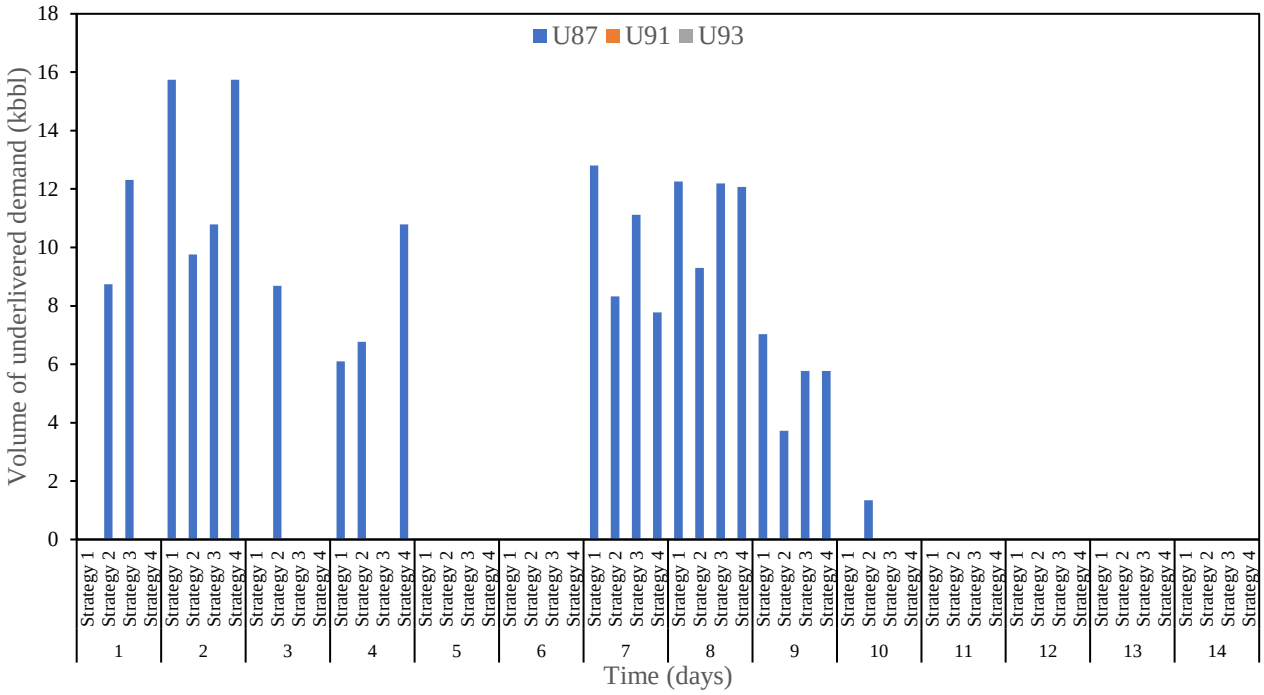

Figure 10. Products undelivered demands for case # 9 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

Table 3. Results for supply-demand pinch and full-space model for linear blending rules

Demand Pattern		Full-space algorithm			Supply-demand pinch algorithm		
1	**First Iteration**	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
	Revenue (thousands $)	67,668.6	67,617.1	67,625.0	67,668.6	67,617.1	67,625.0
	Cost (thousands $)	44,897.8	44,841.7	44,855.3	44,897.8	44,841.7	44,855.3
	Profit (thousands $)	22,770.8	22,775.4	22,769.6	22,770.8	22,775.4	22,769.6
	Execution time (s)	22.0	17.1	22.8	1.2	1.1	1.1
	Rolling Horizon	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
	Revenue (thousands $)	67,396.5	67,634.1	67,575.2	67,396.5	67,634.1	67,575.2
	Cost (thousands $)	44,678.7	44,862.3	44,798.6	44,678.7	44,862.3	44,798.6
	Profit (thousands $)	22,717.8	22,771.7	22,776.6	22,717.8	22,771.7	22,776.6
	Execution time (s)	99.8	102.1	167.2	10.5	12.0	11.3
2	**First Iteration**	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6
	Revenue (thousands $)	76,960.7	76,657.1	76,745.0	76,961.8	76,658.1	76,746.0
	Cost (thousands $)	51,289.0	51,023.1	51,091.1	51,290.2	51,024.3	51,092.3
	Profit (thousands $)	25,671.7	25,633.9	25,653.9	25,671.6	25,633.8	25,653.8
	Execution time (s)	239.4	62.7	73.7	3.0	2.6	2.8
	Rolling Horizon	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6
	Revenue (thousands $)	76,870.1	76,513.6	76,780.4	76,855.9	76,514.6	76,778.8
	Cost (thousands $)	51,223.9	50,896.1	51,128.2	51,209.4	50,897.0	51,126.6
	Profit (thousands $)	25,646.2	25,617.5	25,652.2	25,646.5	25,617.6	25,652.2
	Execution time (s)	336.9	377.8	276.1	15.8	17.9	14.5
3	**First Iteration**	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9
	Revenue (thousands $)	70,010.7	70,048.7	70,026.3	70,010.7	70,048.7	70,026.3
	Cost (thousands $)	46,491.6	46,529.3	46,512.0	46,491.6	46,529.3	46,512.0
	Profit (thousands $)	23,519.1	23,519.4	23,514.4	23,519.1	23,519.4	23,514.4
	Execution time (s)	364.1	19.9	33.1	2.4	2.7	3.4
	Rolling Horizon	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9
	Revenue (thousands $)	70,120.3	70,069.6	70,088.5	70,120.3	70,058.3	70,088.5
	Cost (thousands $)	46,612.4	46,548.6	46,577.8	46,612.4	46,537.3	46,577.8
	Profit (thousands $)	23,507.9	23,521.0	23,510.7	23,507.9	23,521.0	23,510.7
	Execution time (s)	451.9	126.5	135.0	13.3	15.7	22.5

Table 4. Results summary for the four different solutions strategies for the nonlinear blending rules cases

Demand			Strategy 1			Strategy 2			Strategy 3			Strategy 4		
			Case 1	Case 2	Case 3	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
1	First Iteration	Profit (thousands $)	22559.5	22563.5	22559.7	22559.5	22558.3	22559.7	22551.4	22555.8	22551.8	22551.4	22555.8	22551.8
		Execution time (s)	3648.5	3710.1	3619.9	10.2	20.3	13.3	3601.2	3601.4	3602.1	1.4	1.5	1.4
		Gap (%)	1.68	1.74	1.52	-	-	-	-	-	-	-	-	-
	Rolling Horizon	Profit (thousands $)	22510.0	22560.8	22564.9	22510.1	22520.6	22564.4	22504.7	22553.8	22557.5	22505.0	22554.2	22557.0
		Execution time (s)	37238.5	34981.9	37343.3	89.4	148.7	165.0	25223.6	10827.8	10839.7	9.6	13.1	10.5
			Case 4	Case 5	Case 6	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6
2	First Iteration	Profit (thousands $)	25369.7	25338.0	25381.8	25369.8	25338.7	25381.9	25368.2	25337.6	25380.3	25368.2	25337.6	25380.3
		Execution time (s)	3629.8	7200.2	3623.5	52.1	26.8	22.4	5.4	3.9	7.2	2.9	1.9	1.5
		Gap (%)	0.86	0.91	0.92	-	-	-	-	-	-	-	-	-
	Rolling Horizon	Profit (thousands $)	25337.1	25337.9	25372.3	25336.3	25355.3	25372.1	25333.7	25350.1	25375.8	25327.6	25351.7	25375.3
		Execution time (s)	30980.2	39974.4	47822.7	159.8	141.2	142.5	5453.9	14484.3	14845.0	15.7	12.9	13.0
			Case 7	Case 8	Case 9	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9
3	First Iteration	Profit (thousands $)	23303.2	23303.1	23300.5	23303.7	23303.0	23300.5	23298.7	23298.2	23295.3	23298.7	23298.2	23293.6
		Execution time (s)	7245.7	3629.5	3629.5	13.3	14.4	91.1	3613.6	3633.6	3728.3	2.0	2.4	3.8
		Gap (%)	1.59	1.47	1.52	-	-	-	-	-	-	-	-	-
	Rolling Horizon	Profit (thousands $)	23288.6	23304.2	23285.2	23255.0	23303.8	23271.8	23253.0	23289.9	23292.8	23289.8	23296.7	23291.4
		Execution time (s)	36845.7	59489.1	46524.6	107.1	163.8	193.6	11419.2	12453.1	11689.3	26.9	16.3	22.3

3.9. Conclusions

In this work, a production plan model for gasoline blend under demand uncertainty has been introduced. The formulation utilizes a fixed-end rolling horizon and loss function formulation to maximize the refinery ability to satisfy uncertain demand and maximizes its profit. The products demand is modeled as two parts; contracted demand (certain) and additional spot market demand (uncertain). In addition, the time-varying uncertainty is considered, where the uncertainty in demand of the spot market is higher for periods further into the future. The rolling horizon formulation is essential to capture the real-life where new information about the future uncertain demand is available as we roll forward along the horizon. Two models based on linear and nonlinear blending rules are proposed and solved using a full-space algorithm which uses a detailed model for all periods in the planning horizon. The full-space algorithm is effective in solving the linear blending model but struggles in closing the optimality gap under the nonlinear blending causing high execution times. A new supply-demand pinch algorithm is proposed to solve the linear and nonlinear blending models more efficiently, which decompose the problem into two sub problems solved in sequence using aggregation and disaggregation technique based on the supply-demand pinch concept. Under linear blending rules, the supply-demand pinch algorithm computes the same profit solutions compared to full space algorithm with 5 to 30-fold reduction in execution times. While for nonlinear blending rules, the supply-demand pinch algorithm with global solver (Antigone) computes profit solutions within 0.05% compared to full space algorithm solutions for most cases, with only 2 to 5-fold reduction in execution times. The big advantage of the supply-demand pinch algorithm is its ability to simplify the large MINLP model so that local solvers such as DICOPT can be used to compute solutions close to optimal. The supply demand pinch algorithm with DICOPT achieved solutions within 0.04% compared to full-space algorithm solutions while reducing execution times by 2000 to 3000-fold.

Supporting Information.

Detailed Results for the four different solutions strategies for the case of nonlinear blending rules, Components costs and products prices, contracted demand data, product and component tanks data, components qualities and products qualities specifications under both linear and nonlinear blending rules, blender capacity, penalty coefficients for inventory infeasibility on the components and products sides, coefficients for loss function approximation, actual products demands for all cases, and undelivered product demands figures (Supporting Info PDF file).

Acknowledgments

Authors would like to thank the McMaster Advanced Control Consortium and Imperial Oil Ltd. for their financial support.

Nomenclature

Sets and indices

I = {*i*}	Set of blend components
P = {*p*}	Set of different products
K = {*k* | 1,2, …, *K*}	Set of aggregated future periods at top-level
M = {*m* | 0,1, …, *M*}	Set of detailed periods at bottom-level (or periods of the full-space model)
S = {*s*}	Set of qualities
BL = {*bl*}	Set of blenders
J = {*j*}	Set of product storage tanks
MA = {*m* | 1,2, …, *M*}	Subset of **M**, detailed periods excluding period 0
M0 = {*m* | 0}	Subset of **M**, previous period in detailed periods
MC = {*m* | 1}	Subset of **M**, current detailed period
MF = {*m* | 2,3, …, *M*}	Subset of **M**, future detailed periods
KI = {*k* | 1}	Subset of **K**, the first aggregated period
JP = {(*j*, *p*)}	Tank *j* can store product *p*

BP = {(*bl, p*)}	Blender *bl* that can produce product *p*
MK = {(*m, k*)}	Periods *m* (bottom-level periods) that contained in each aggregated period *k* (top-level periods)

Continuous variables

Profit	Refinery profit
Current Revenue	Sales revenue from the current period
Future Revenue	Sales revenue from future periods
Current Blend Cost	blending cost from current period
Future Blend Cost	blending cost from future periods
$V^{comp}_{i,p,m,bl}$	Volume of component *i* into product *p* in blender *bl* at period *m*
$V^{blend}_{p,m,bl}$	Volume blended of product *p* in blender *bl* at period *m*
$V^{comp\_agg}_{i,p,k}$	Aggregated volume of component *i* into product *p* at period k
$V^{blend\_agg}_{p,k}$	Aggregated volume blended of product *p* at period *k*
$V^{trans}_{j,p,m,bl}$	Volume of product *p* in blender *bl* transferred to product tank *j* at period *m*
$V^{pr}_{j,p,m}$	Volume of product *p* stored in tank *j* during the period *m*
$V^{bc}_{i,k}$	Volume stored of component *i* in period *k*
$V^{bc}_{i,m}$	Volume stored of component *i* in period *m*
$V^{pool}_{p,k}$	Volume stored in product pool *p* at the end of period *k*
$V^{pool}_{p,m}$	Volume stored in product pool *p* at the end of period *m*
$r_{i,p,m,bl}$	Fraction of component *i* into product *p* at blender *bl* during period *m*
$r_{i,p,k}$	Fraction of component *i* into product *p* during period *k*
$Q^{pr}_{p,s,m,bl}$	Quality value of property *s* for product *p* in blender *bl* during period *m*
$Q^{pr}_{p,s,k}$	Quality value of property *s* for product *p* in period *k*

$\mathrm{Deliver}_{p,m}^{\mathrm{pool}}$	Volume of product *p* shipped at the end of period *m*
$\mathrm{Deliver}_{p,k}^{\mathrm{pool\_agg}}$	Volume of product *p* shipped at the end of period *k*
$\mathrm{Deliver}_{j,p,m}^{\mathrm{pr}}$	Volume of product *p* shipped from product tank *j* at the end of period *m*
$\mathrm{Demand}_{p,m}^{\mathrm{unmet}}$	Unmet demand of product *p* during period *m*
$\mathrm{z}_{p,m}$	z-score for product *p* during period *m*
$\mathrm{L}_{p,m}$	Loss value of product *p* during period *m*
$\mathrm{z}_{p,k}$	z-score for product *p* during period *k*
$\mathrm{L}_{p,k}$	Loss value of product *p* during period *k*
$\mathrm{S}_{i,m}^{\mathrm{bc+}}$, $\mathrm{S}_{i,m}^{\mathrm{bc+}}$	Positive and negative inventory slack variable of components *i* during period *m*
$\mathrm{S}_{p,m}^{\mathrm{pool+}}$, $\mathrm{S}_{p,m}^{\mathrm{pool-}}$	Positive and negative inventory slack variable of product pool *p* during period *m*
$\mathrm{S}_{j,p,m}^{\mathrm{pr+}}$, $\mathrm{S}_{j,p,m}^{\mathrm{pr-}}$	Positive and negative inventory slack variable of product *p* in product tank *j* during period *m*

Integer variables

$\mathrm{t}_{p,m,bl}^{\mathrm{blend}}$	Time required for blend run

Binary variables

$\mathrm{x}_{p,bl,m}$	Defines if product *p* is blended in blender *bl* during period *m*
$\mathrm{v}_{j,p,m,bl}$	Defines if product *p* is transferred from blender bl to product tank *j* during period *m*
$\mathrm{u}_{j,p,m}$	Defines if product *p* is stored in product tank *j* during period *m*
$\mathrm{ue}_{j,m}$	Defines if a product transition occurs at product tank *j* at the beginning of period *m*

Parameters

SP_p	Selling Price of product *p*

CC_i	Cost of component *i*
$\mathrm{V}_{i,m}^{\mathrm{comp\_in}}$	Volume of components *i* produced during the period *m*
$\mathrm{V}_{i,k}^{\mathrm{comp\_in\_agg}}$	Aggregated volume of components *i* produced during period *k*
t_m	Duration of period *m*
$\mathrm{V}_{p,k}^{\mathrm{blend\_agg}}$	Aggregated volume blended of product *p* at period k
$\mathrm{Q}_{i,s}^{\mathrm{bc}}$	Quality value of property *s* for component *i*
$\mathrm{V}_i^{\mathrm{bc\_min}}$	Minimum capacity of the tank with blend component *i*
$\mathrm{V}_i^{\mathrm{bc\_max}}$	Maximum capacity of the tank with blend component *i*
$\mathrm{V}_p^{\mathrm{pr\_min}}$	Minimum capacity of the tank with product *p*
$\mathrm{V}_p^{\mathrm{pr\_max}}$	Maximum capacity of the tank with product *p*
$\mathrm{r}_{i,p}^{\mathrm{min}}$	Minimum fraction of component i in product *p*
$\mathrm{r}_{i,p}^{\mathrm{max}}$	Maximum fraction of component i in product *p*
$\mathrm{Q}_{p,s}^{\mathrm{pr\_min}}$	Minimum quality value of property *s* in product *p*
$\mathrm{Q}_{p,s}^{\mathrm{pr\_max}}$	Maximum quality value of property *s* in product *p*
$\mathrm{F}_{bl}^{\mathrm{blend\_min}}$	Minimum blending rate of blender *bl*
$\mathrm{F}_{bl}^{\mathrm{blend\_max}}$	Maximum blending rate of blender *bl*
$\mathrm{cit}_{p,bl}^{\mathrm{blend}\_min}$	Minimum idle time required by blender to process product *p*
$\mathrm{ct}_{p,bl}^{\mathrm{blend\_min}}$	Minimum running time of blender bl for product *p*
$\mathrm{V}_{bl}^{\mathrm{blend\_min}}$	Minimum volume blended in blender *bl*
$\mathrm{t}_{bl}^{\mathrm{blend\_min}}$	Maximum volume blended in blender *bl*
$\mathrm{V}_j^{\mathrm{pr\_min}}$	Minimum capacity of tank product *j*
$\mathrm{V}_j^{\mathrm{pr\_max}}$	Maximum capacity of tank product *j*

$V_p^{pool\_min}$	Minimum capacity of product pool *p*
$D_j^{pr\_max}$	Maximum delivery rate of tank *j*
$V_i^{bc\_initial}$	Component *i* starting inventory
$V_p^{pool\_initial}$	Product *p* starting inventory
$u_{j,p}^{initial}$	Initial product *p* in product tank *j*
$V_{j,p}^{pr\_initial}$	Initial volume of product *p* in product tank *j*
$x_{p,bl}^{initial}$	Initial product *p* in blender *bl*
$Density_i$	Density of component *i*
$Demand_{p,m}^{cont}$	Contracted demand of product *p* during period *m*
$Demand_{p,m}^{add\_mean}$	Mean additional demand for product *p* during period *m*
$Demand_{p,m}^{add\_Variance}$	Variance of additional demand for product *p* during period *m*
$Demand_{p,k}^{cont\_agg}$	Aggregated contracted demand of product *p* during period *k*
$Demand_{p,k}^{add\_agg\_mean}$	Aggregated mean additional demand of product *p* during period *k*
$Demand_{p,k}^{add\_agg\_Variance}$	Aggregated variance of additional demand of product *p* during period *k*
np_{bl}	Number of products that can be blended in blender *bl*
$Penalty^{bc}$	Penalty associated with components inventory infeasibilities
$Penalty_m^{pr}$	Penalty associated with products inventory infeasibilities

References

(1) Biegler, Lorenz T. *Nonlinear programming: concepts, algorithms, and applications to chemical processes*. Vol. 10. Siam, 2010. https://doi.org/10.1137/1.9780898719383

(2) Wang, Wei, Zefei Li, Qiang Zhang, and Yankai Li. "On-line optimization model design of gasoline blending system under parametric uncertainty." In *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pp. 1-5. IEEE, 2007. https://doi.org/10.1109/MED.2007.4433757

(3) Li, Jie, I. A. Karimi, and Rajagopalan Srinivasan. "Recipe determination and scheduling of gasoline blending operations." *AIChE Journal* 56, no. 2 (2010): 441-465. https://doi.org/10.1002/aic.11970

(4) Glismann, Klaus, and Günter Gruhn. "Short-term planning of blending processes: scheduling and nonlinear optimization of recipes." *Chemical Engineering & Technology: Industrial Chemistry-Plant Equipment-Process Engineering-Biotechnology* 24, no. 3 (2001): 246-249. https://doi.org/10.1002/1521-4125(200103)24:3<246::AID-CEAT246>3.0.CO;2-8

(5) Jia, Zhenya, and Marianthi Ierapetritou. "Mixed-integer linear programming model for gasoline blending and distribution scheduling." *Industrial & Engineering Chemistry Research* 42, no. 4 (2003): 825-835. https://doi.org/10.1021/IE0204843

(6) Mendez, Carlos A., Ignacio E. Grossmann, Iiro Harjunkoski, and Pousga Kaboré. "Optimization techniques for blending and scheduling of oil-refinery operations." *Pittisburg, USA*(2004).

(7) Cerdá, Jaime, Pedro C. Pautasso, and Diego C. Cafaro. "A cost-effective model for the gasoline blend optimization problem." *AIChE Journal* 62, no. 9 (2016): 3002-3019. https://doi.org/10.1002/aic.15208

(8) Castillo, Pedro A. Castillo, and Vladimir Mahalec. "Inventory pinch based, multiscale models for integrated planning and scheduling-part I: Gasoline blend planning." *AIChE Journal*60, no. 6 (2014): 2158-2178. https://doi.org/10.1002/aic.14423

(9) Hamisu, Aminu Alhaji. "Petroleum refinery scheduling with consideration for uncertainty." (2015).

(10) Lin, Po-Chen, and Reha Uzsoy. "Chance-constrained formulations in rolling horizon production planning: an experimental study." *International Journal of Production Research* 54, no. 13 (2016): 3927-3942. https://doi.org/10.1080/00207543.2016.1165356

(11) WEMMERLÖV, URBAN. "A time-phased order-point system in environments with and without demand uncertainty: a comparative analysis of non-monetary performance variables." *International journal of production research* 24, no. 2 (1986): 343-358. https://doi.org/10.1080/00207548608919733

(12) Yano, Candace Arai, and Robert C. Carlson. "Interaction between frequency of rescheduling and the role of safety stock in material requirements planning

systems." *International journal of production research* 25, no. 2 (1987): 221-232. https://doi.org/10.1080/00207548708919835

(13) Campbell, Gerard M. "Master production scheduling under rolling planning horizons with fixed order intervals." *Decision Sciences* 23, no. 2 (1992): 312-331. https://doi.org/10.1111/j.1540-5915.1992.tb00391.x

(14) Higle, Julia L., and Karl G. Kempf. "Production planning under supply and demand uncertainty: A stochastic programming approach." In *Stochastic Programming*, pp. 297-315. Springer, New York, NY, 2010.

(15) Li, Wenkai, Chi-Wai Hui, Pu Li, and An-Xue Li. "Refinery planning under uncertainty." *Industrial & engineering chemistry research* 43, no. 21 (2004): 6742-6755. https://doi.org/10.1021/ie049737d

(16) Singh, A., J. Fraser Forbes, P. J. Vermeer, and S. S. Woo. "Model-based real-time optimization of automotive gasoline blending operations." *Journal of process control* 10, no. 1 (2000): 43-58. https://doi.org/10.1016/S0959-1524(99)00037-2

Chapter 4: Adaptive System Identification of Industrial Ethylene Splitter: A comparison of Subspace Identification and Artificial Neural Networks

The contents of this chapter have been published in the *Computer & Chemical Engineering Journal*. Reprinted with permission from

Jalanko, M., Sanchez, Y., Mahalec, V., & Mhaskar, P. (2021). Adaptive system identification of industrial ethylene splitter: A comparison of subspace identification and artificial neural networks. *Computers & Chemical Engineering*, *147*, 107240

Abstract

The manuscript considers the problem of data-driven modeling of an ethylene splitter (from an industrial plant). The process presently operates with end composition controllers that does not work well during process transition. The objective of the present work is to investigate the use of different data-driven techniques such as subspace identification and neural network-based methods for the purpose of developing a dynamic data-driven model. To this end, first an ethylene splitter simulation model is built that replicates industrial operation. The ability of the simulation model to capture the key traits of the process dynamics are first established by comparing it with data from the plant operation. The simulation model is subsequently utilized to work as a test bed for future control purposes and to serve as an additional test of the modeling approaches. An online model adaptation scheme is developed to improve the model's prediction capabilities under new operation patterns.

Keywords:

Modeling industrial separation unit; system identification; subspace identification; artificial neural network; time series prediction

4.1. Introduction

Ethylene (C_2H_4) is one of the most versatile and widely used petrochemicals in the world today and is primarily being used for the manufacturing of polyethylene. The separation of ethylene from ethane by the C2 splitter is normally the final step in the production of ethylene, where the final products are primarily ethylene in the top stream and ethane in the bottom stream. The top ethylene product is then sold, while the bottom ethane is recycled back to upstream processes. Separation of ethylene from ethane is one of the most energy intensive separations, which uses large distillation column with over 100 trays due to the small difference in relative volatilities of ethane and ethylene [1]. Another contributor to the high energy consumption is that the ethylene splitter is commonly operated at high-pressure, utilizing closed-cycle propylene refrigeration. Optimization of the C2 splitter process is critical to meet desired ethylene product purity while maximization the ethylene-ethane separation and the hydraulic processing capacity of the tower. Continued push towards operational efficiency of the ethylene splitter process, in turn, continues to motivate advanced modeling and control efforts in this direction. Of particular interest are modeling approaches that can readily adapt the model to new operating conditions as new data becomes available.

The complexity of C2 splitter operation stems from many reasons: (1) high interaction between the top and bottom purities, (2) small difference in relative volatilities of ethane and ethylene, (3) slow dynamics of the column, (4) nonlinear dynamic behavior which changes in the plant conditions (such as pressure and temperature), and (5) continuous change in feed flow and composition. C2 splitter operates with limited capacity and high feed flow which can result in exceeding tower capacity causing tower flooding. The ethylene splitter can be modeled using first-principles models or data-driven models. The advantage of first-principles models is that they can work in a relatively broad range of operations, but such models are expensive, hard to develop, and harder to maintain. Therefore, industrial practice has sought to pursue data-driven models that are relatively easier to develop and maintain [2]. These challenges make the problem of building an accurate model for C2 splitter a hard task to accomplish [3]. The critical metric in evaluating the usefulness of such data driven models is model validity and extrapolation capabilities, especially when being used for the purpose of control and optimization (and not just process monitoring).

In the direction of utilizing first principles model for ethylene splitter, Salerno et al. [1] developed a rigorous first principles model in Aspen Plus for ethylene splitter. Borralho [4] developed a first principles model for the ethylene plant in gPROMS. Yan [5] developed a simplified first principles ethylene plant model which includes a thermal cracking section, a separation system, and an integrated refrigeration. Wang [6] developed a rigorous dynamic simulation for the startup operation of C2 splitter. Choe and Luyben [7] developed rigorous dynamic models of C2 splitter that considers the effect of vapor holdup in high pressure columns, which is normally ignored. Eliceche et al. [8] studied ethylene plant units' capacities for bottlenecks in the furnaces and ethylene splitter sections caused by changes in the feed flow rate and composition. All these contributions, however, used only simulations and have not validated their results against plant operation.

Friedman [9] reviewed the existing contributions in modeling ethylene plant units using various data driven models, including subspace identification and artificial neural network-based approaches. Huang et al. [10] proposed a closed-loop subspace identification approach through an orthogonal projection. Several examples exist of the application of subspace identification methods for data driven modeling of distillation columns using simulation data [11, 12, 13]. Kanthasamy et al. [14] developed a nonlinear system identification model for pilot plant distillation column based on Hammerstein model. The model consists of a nonlinear static element followed by a linear dynamic model. The data generated from their first principles model was first validated using experimental data before being used as the process model in the Hammerstein model parameter estimation. Norquay et al. [15] developed a nonlinear system identification model for an industrial ethylene splitter based on Wiener model, which consists of a linear dynamic element followed by a nonlinear static element. In their work, they used plant data with only two outputs and three inputs along with generated simulation to develop the Wiener model and compared it to real plant data.

Artificial neural networks (ANN) models have been used by many researchers in the context of model identification of a distillation column. Many works have utilized simulation data to model distillation column using feed forward neural network [16] [17] [18], recurrent neural network [19] [20] and nonlinear autoregressive with exogenous inputs (NARX) network architecture [21] [22]. Relatively fewer contributions model industrial distillation columns using real process data. Savkovic-Stevanovic [23] used plant data to develop a feed forward neural network model for an

industrial distillation column. Singh et al. [24] used laboratory data to model a 9-tray binary distillation column available in the laboratory using both, feed forward neural network and recurrent neural network. Abdullah et al. [25] proposed a feed forward neural network to predict the top and bottom product composition of a pilot plant distillation column using simulation data and validation against plant data. ANN based Model Predictive Controller (MPC) has also been used in other application [26] [27]. In summary, while many contributions have utilized ANN models based on either simulation data or real data, limited work has been carried out to validate the model with real data from industrial distillation columns and to compare ANN based approaches against alternatives such as subspace identification.

In practice, process operation changes over time due to internal and external conditions which can cause deterioration in model predictions. Therefore, the need to continuously updating process models, via adaptive modeling algorithm, as time evolves might be necessary to sustain the model predictions accuracy. Alanqar et al. [28] proposed an error-triggered on-line model identification strategy for linear state-space model to obtain more accurate state predictions for nonlinear process systems. The error-triggering was conducted by a moving horizon error detector that quantifies the relative prediction error within its horizon and triggers model re-identification using recent operations when the prediction error exceeds a threshold. Wu et al. [29] proposed a machine learning-based predictive control scheme that utilizes an online update of the Recurrent Neural Network RNN models to capture process nonlinear dynamics in the presence of model uncertainty.

Motivated by these considerations, in the present manuscript, the ethylene splitter, a distillation column with a large number of inputs and output is modelled using different system identification methods. The system identification methods studied in this work are subspace identification, NARX neural network, and nonlinear RNN. This is achieved in two steps. First, a simulation model in Aspen Dynamics is developed to work as a test bed for future control purposes and to serve as an additional test of the modeling approaches. The Aspen simulator is developed for testing our implemented control strategy, since it cannot be demonstrated directly using the real plant. Subsequently, various data driven models are developed and tested against both the simulation and the plant data. Finally, two online modelling schemes are developed for the purpose of continuously updating the models with new available data to improve their predictions capabilities. The two adaptive modeling algorithms are introduced to keep the

dynamic model up to date with the most recent operations and these algorithms are adapted for our three system identification methods. The rest of the paper is organized as follows. Section 2 presents an overview of the ethylene plant,the C2 splitter, and the system identification methods. Section 3 presents the developed Aspen dynamics-based simulation model and compares the simulation results against the plant operation. Section 4 presents the system identification results. Section 5 presents the two adaptive strategies for three different system identification methods that allow more recent data to be incorporated into the training approaches. Section 6 demonstrates the importance of using an adaptive modeling scheme when modeling the ethylene splitter for the three system identification methods. Finally, concluding remarks are presented in section 7.

4.2. Preliminaries

In this section, a detailed description of the C2 splitter is provided, followed by a brief review of the different system identification methods utilized.

4.2.1. C2 Splitter Process

In this section, we describe the C2 splitter at Joffre site that separates ethylene from ethane, which is normally the final step in the production of ethylene. For a detailed description of the ethylene production plant, its main sections, and its chemistry, the reader is referred to [30]. The required purity of ethylene usually exceeds 99%. To meet this product specification, the quality level is set at a purity level greater than what is required to prevent production of off-spec products. To operate at a minimum energy consumption, however, the column is operated as close as possible to the maximum impurity level. [31]

A schematic of the C2 splitter at the Joffre site without the end composition controllers is shown in Figure 1. The tower has a temperature controller on tray 85, which measures the temperature at that tray and adjusts the reboiler. It also has an ethane composition controller at tray 24, which measures the ethane composition at that tray and adjusts the reflux. The tower sump level controller is used to adjust the bottom product flow rate to maintain the level of liquid at the bottom tower to its set point. The reflux drum level controller adjusts the distillate flow to maintain the reflux drum level at its set point.

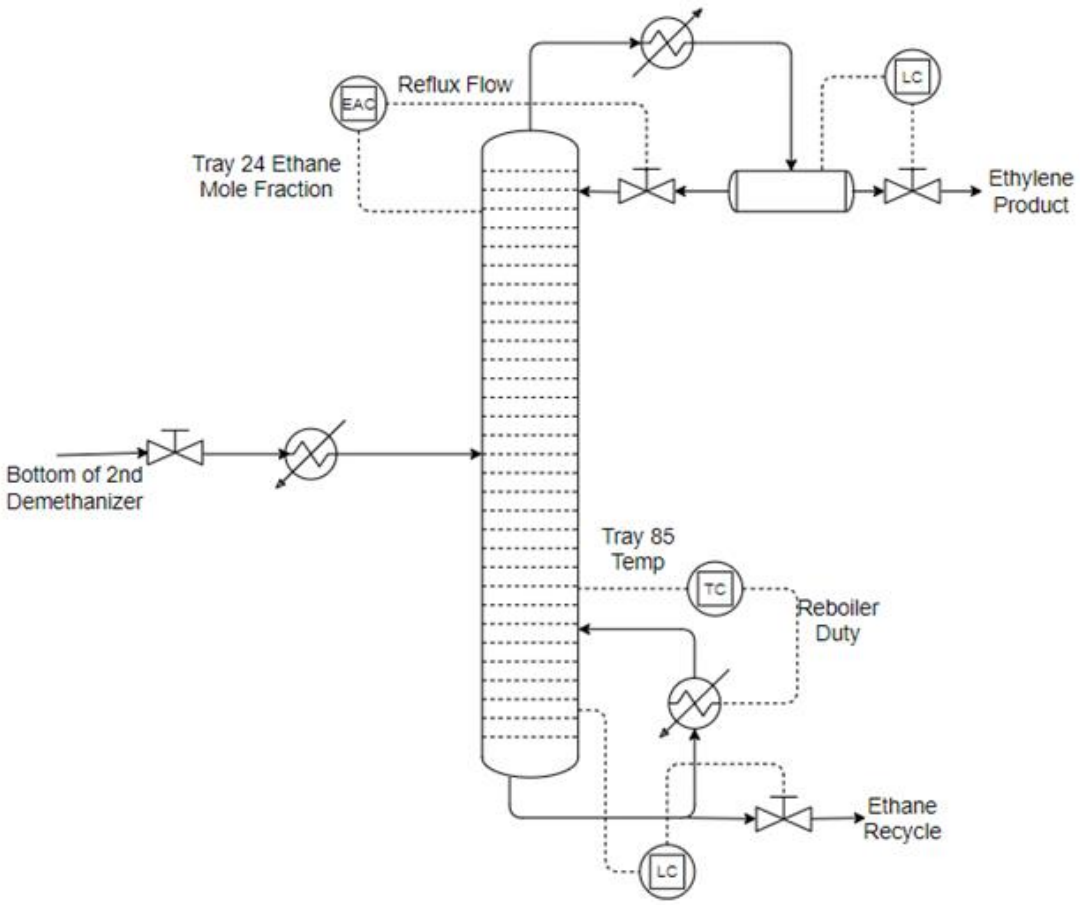

Figure 1. C2 splitter current control strategy

Table 1 gives a summary of the C2 splitter input (manipulated variables and disturbances) variables and output variables.

Table 1. C2 splitter input and output variables

Category	Variable	Description	Units
Disturbance	$\dot{m}_{feed}$	Feed flow from upstream process	kg/hr
	T_f	Feed temperature	°C
	$x_{C_2H_4}$	Mole fraction of ethylene	%
	$x_{C_2H_6}$	Mole fraction of ethane	%
	P_f	Feed pressure	bar
	v_f	Vapor Fraction	%
	$\dot{Q}_C$	Condenser duty	Gj/hr
Manipulated variables	$\dot{m}_R$	Reflux flow	kg/hr
	$\dot{Q}_R$	Reboiler duty	Gj/hr
Outputs	T_{t_24}	Temperature at tray 24	°C
	T_{t_37}	Temperature at tray 37	°C
	T_{t_51}	Temperature at tray 51	°C
	T_{t_56}	Temperature at tray 56	°C
	T_{t_85}	Temperature at tray 85	°C
	P_{t_1}	Pressure at tray 1	bar
	ΔP_{top}	Top tower differential pressure	bar
	ΔP_{bottom}	Bottom tower differential pressure	bar
	$x_{C_2H_4\ t\_97}$	Mole fraction of ethylene at tray 97	%
	$x_{C_2H_6\ t\_24}$	Mole fraction of ethane at tray 24	%
	$\dot{m}_{top}$	Flow rate of top product (ethylene)	kg/hr
	T_{top}	Temperature of top product (ethylene)	°C
	$x_{C_2H_6\ top}$	Mole fraction of ethane of the top product (ethylene)	%
	$\dot{m}_{bottom}$	Flow rate of bottom product (ethane)	kg/hr
	T_{bottom}	Temperature of bottom product (ethane)	°C
	$x_{C_2H_6\ bottom}$	Mole fraction of ethylene of the bottom product (ethane)	%

4.2.2. System identification methods

Based on how the multi-step prediction is generated, these methods can be classified into two categories: direct prediction methods and recursive prediction methods. In the direct prediction methods, the model is trained to directly make a multi-step prediction and then used directly to make multi-step ahead predictions. However, in recursive prediction methods, the model is trained to make a one-step prediction and then used to make multi-step ahead predictions recursively. To achieve this, the one step ahead prediction provided by the model is fed back into the model as to make the next step prediction. These two strategies normally have trade-off between bias and estimation variance. The direct prediction strategy has higher variance because it uses fewer observations when estimating the model, while the recursive prediction strategy is biased when the process is nonlinear [32]. Recursive models that are built to perform one step ahead predictions are generally simpler compared to direct models that are built to perform multi-step ahead predictions. Therefore, recursive models generally have a smaller number of parameters, making it less powerful and less able to fit any shape, and thus lead to a larger bias. On the other hand, the smaller number of parameters lead to smaller sensitivity to the data and therefore smaller variance. The direct method normally requires larger number of parameters make multi-step ahead predictions, which causes the model to have low bias but large variance [33]. Another way to classify the models is whether they map the inputs to outputs directly or define states that relate the inputs to outputs. Next, we briefly review the subspace identification, the NARX neural network, and the RNN modeling approaches.

4.2.2.1. Subspace identification

In this section, a brief description of the subspace identification method is presented. For a detailed description of the subspace identification method utilized in this work, the reader can refer to the work of Moonen and De Moor [34], the work of Corbett and Mhaskar [35] and Garg and Mhaskar [36] for adaptation of the modeling approach for batch and batch like processes. This method falls under state-space models and gathers the input-output trajectories of the process to construct a linear time invariant discrete time model. Ultimately, the goal is to determine the system matrices (*A*, *B*, *C* and *D*) for a discrete linear time invariant model of the following form:

$$x_{k+1} = Ax_k + Bu_k \tag{1}$$

$$y_k = Cx_k + Du_k$$

where $x \in \mathbb{R}^{n_x}$ denotes the vector of state variables, $y \in \mathbb{R}^{n_y}$ denotes the vector of measured output variables and $u \in \mathbb{R}^{n_u}$denotes the vector of manipulated input variables. The sampling instance (k) represents the sampling time of the process. The subspace identification algorithm consists of two steps. In the first step, a state vector sequence is realized as the intersection of the row spaces of two block Hankel matrices constructed from the input and output data. In the second step, the system matrices are obtained from the least-square solution of a set of linear equations. The subspace identification method has only two hyperparameters that need to be chosen- these are the number of states, and the history length of the batch. For prediction for a new operation, the states of the underlying model must be estimated using any state estimator. A Luenberger observer is utilized in this work.

4.2.2.2. Artificial Neural Network

In this section, we review two different neural network architectures that have been commonly used in system identification and time series prediction: the NARX neural network and the RNN. The NARX neural network is an input-output system identification method.

The NARX neural network is trained using a feed forward network architecture to perform a single step prediction using past output measurements as additional inputs (along with the past and current inputs) as shown in eq (2). Once the training is complete, the NARX neural network uses closed loop architecture to perform multi-step predictions as shown in eq (3), where the predicted outputs are fed back as additional inputs (along with the past and current inputs). There are two advantages of training the NARX neural network as a single step with open loop architecture. First, the input to the feedforward network is more accurate since we are using the true past output values as inputs instead of the predicted. Second, the open-loop structure can be modelled as purely feedforward architecture which allows using traditional training algorithm such as static backpropagation during training.

$$\hat{y}_k = f(y_{k-1}, \dots, y_{k-N_y}, u_k, \dots, u_{k-N_u}) \tag{2}$$

$$\hat{y}_k = f(\hat{y}_{k-1}, \dots, \hat{y}_{k-N_y}, u_k, \dots, u_{k-N_u}) \tag{3}$$

Where $f(.)$ is the mapping function of the neural network, $\hat{y}_k$ is the predicted output of the NARX neural network at the time $k-1$ for the time k. $y_{k-1}, \dots, y_{k-N_y}$ are the past outputs, $\hat{y}_{k-1}, \dots, \hat{y}_{k-N_y}$ are predicted past outputs, and $u_k, \dots, u_{k-N_u}$ are the current and past output. N_y and N_u are the number of output and inputs delays. In state-space approach, the relationship that maps the input variables to output variables is described by intermediate state variables. In an input-output model, the output is expressed directly in terms of the inputs, bypassing the state variables. While an input-output model can be expressed in the form of an equivalent state-space model and vice versa, the difference in the training algorithm of each model can result in different models.

RNN models use internal states (memory) to process a sequence of inputs and these states recursively updates themselves throughout the prediction horizon. Therefore, RNN models, similar to linear subspace identification, require a state initialization in order to produce accurate predictions. The RNN architecture is considered a generalization of feedforward neural network with internal memory and must have feedback connection while training which enables the network to do temporal processing and learn time sequences. RNN uses the feedback connection from its past decisions to ingest its own outputs as inputs, step after step. The past sequential information is preserved in the recurrent network's hidden state as a memory, which is used along with the current input to predict the output. The RNN equation can be defined as follows:

$$\begin{aligned} \hat{x}_k &= g(\hat{x}_{k-1}, u_k) \\ \hat{y}_k &= C\hat{x}_k \end{aligned} \tag{4}$$

where $g(.)$ is a function that takes current inputs u_k, and previous states $\hat{x}_{k-1}$, as inputs to predict current state $\hat{x}_k$. C is a matrix that transform the current states $\hat{x}_k$ to the predicted output $\hat{y}_k$.

4.3. Modeling Ethylene Splitter

As noted earlier, the first step was to create a dynamic model for testing of the modeling procedure and as a future test bed for feedback control purposes. Since the tower has been simulated as an isolated system, the dynamic model needed adjustment to replicate the industrial process shown previously in Figure 1 due to two issues. First, the model does not consider the effect of the propylene stream through the heat exchanger used to preheat the feed. Second, the

tower operates with pressure floating on propylene system capacity which is not included in the simulation since no information is available on the propylene stream. The required modifications needed to walkaround these two issues are discussed later in this section.

To develop a dynamic model of our process, the process was first modeled as a steady state process in Aspen Plus V10 using the Peng Robinson model for property estimation. The tower stage Murphree efficiency has been tuned and a value of 0.9 across the tower was found to produce results that are close to the real plant operation. The Aspen Plus model has been exported to Aspen Dynamics and then adapted to create a closed-loop process that replicates the plant operation. The Aspen Dynamics model generated includes the default tower sump level controller and the reflux drum level controller. The two other controllers, tray 85 temperature and tray 24 ethane composition controllers, have been implemented in Aspen Dynamics to replicate plant's operation. To deal with the two issues mentioned earlier, a temperature controller was added to match the plant feed temperature measurements by adjusting the reboiler duty in the heat exchanger before the C2 splitter. Additionally, a simple linear model that correlates the tower's feed rate to the tower's top pressure based on plant's measurements has been developed and implemented to give the top tower pressure set point based on the flow by adjusting the condenser duty. The full process used in of the C2 splitter in Aspen Dynamics is shown in Figure 2. Table 2 shows the parameters used in each controller.

Table 2. Summary of the parameters used in each controller/block in Aspen Dynamics model

Controller	PV Range	Output Range	Gain	Integral Time (min)	Comment
Tower Sump Level Controller	0-100%	0-100%	0.55	55	Tuned to be slow and not very reactive. In the plant it is configured as a gap controller
Reflux Level Controller	0-100%	0-100%	2	60	Configured same as plant
Tray 85 Temp Controller	-40 – 0 C	0 – 165 GJ/hr	0.3	20	Model simplification manipulates duty. In the plant, it manipulates propylene flow. Tuned to mimic plant dynamic behavior
Tray 24 Ethane Controller	0 – 5 %	0 – 481.11 Mgh	0.19	15	Configured same as plant
Feed Temperature Controller	-50-0 C	-100-100 GJ/hr	1	60	Tuned to be relatively fast to match the feed temperature to the real plant temperature
OVHD Pressure Controller	0 – 2500kPa	-320 – 0 GJ/hr	1	60	Tuned to be relatively fast. This controller gets remote SP from correlation to mimic plant behavior
f(x)					Linear correlation developed that relates tower feed flow and pressure.

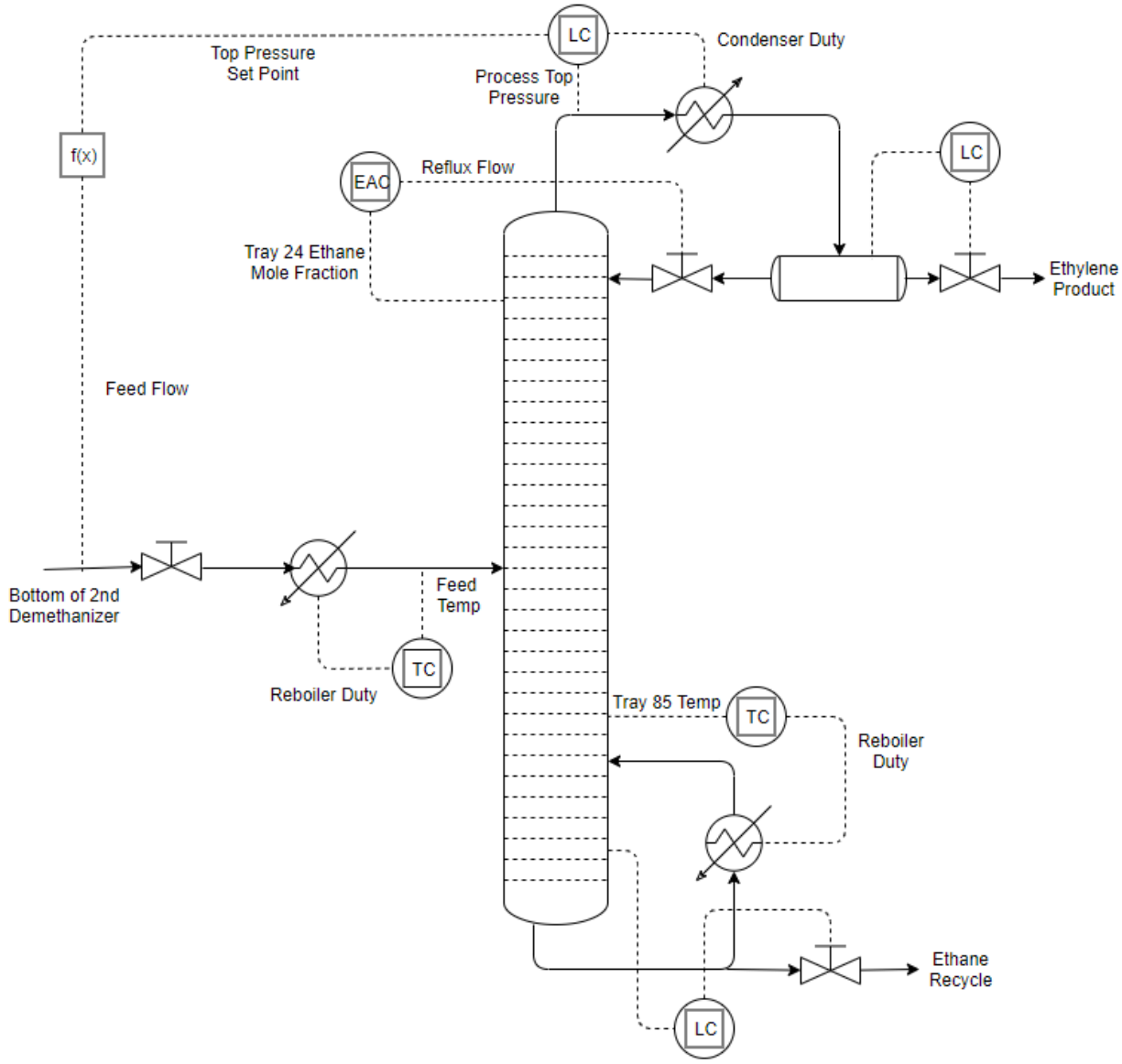

Figure 2. Detailed diagram for the Aspen Dynamics Model

One of the main difficulties involved with validating our simulation model with the plant's ethylene splitter is that no gas chromatography (GC) is available on the ethylene splitter feed. As a result, the feed composition measurements are not available. To address this problem, we inferred the feed composition from upstream composition information. Figure 3 shows the C2 splitter with the relevant upstream processes used to infer the feed composition of the C2 splitter. The composition of the deethanizer top stream and the composition of the recycling stream of the top second demethanizer are measured and used to infer the feed composition of the C2 splitter. The overhead deethanizer stream has ethane, ethylene, acetylene, methane, and propylene

components. We assume that the liquid feed stream composition of the second demethanizer is the same as the overhead deethanizer stream composition subtracting the acetylene. Then, component balance calculations are done around the second demethanizer to obtain the composition of the second demethanizer bottom. While this procedure helps us to compute the C2 splitter feed composition, it is still not completely accurate and might contribute to the plant-model mismatch in the C2 splitter output measurements.

The simulation data has been generated by providing the 2nd demethanizer bottom flow rate, temperature, and pressure provided by the plant measurements, along with the feed composition inferred from upstream processes as mentioned earlier. The tower feed temperature measurements of the plant have been used as the set point for the feed temperature controller. Tray 85 temperature and tray 24 ethane composition plant set points have been fixed based on the real plant set points. The plant-model mismatch is evaluated by generating plots for all the inputs and outputs of the real plant and the model for data from (2019-01-31 2:50) to (2019-02-17 11:20) with sampling rate of 10 minutes. Figure 4 compares plant measurements to simulation results for some of the key variables in the process. It is important to note that due to confidential reasons, the feed flow, reflux flow, products flows, and reboiler duty values are normalized based on the minimum and maximum value of the plant measurements. The plots of the other measurements are shown in the Appendix.

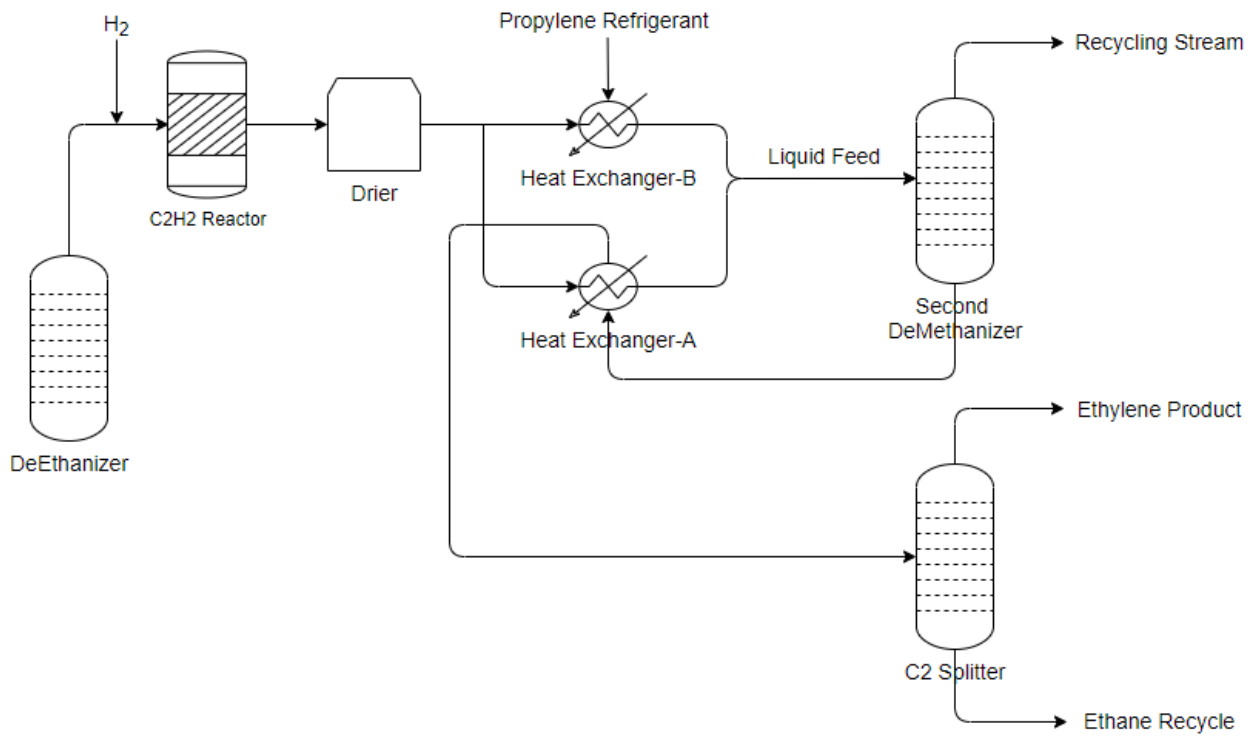

Figure 3: Ethylene splitter and relevant upstream processes

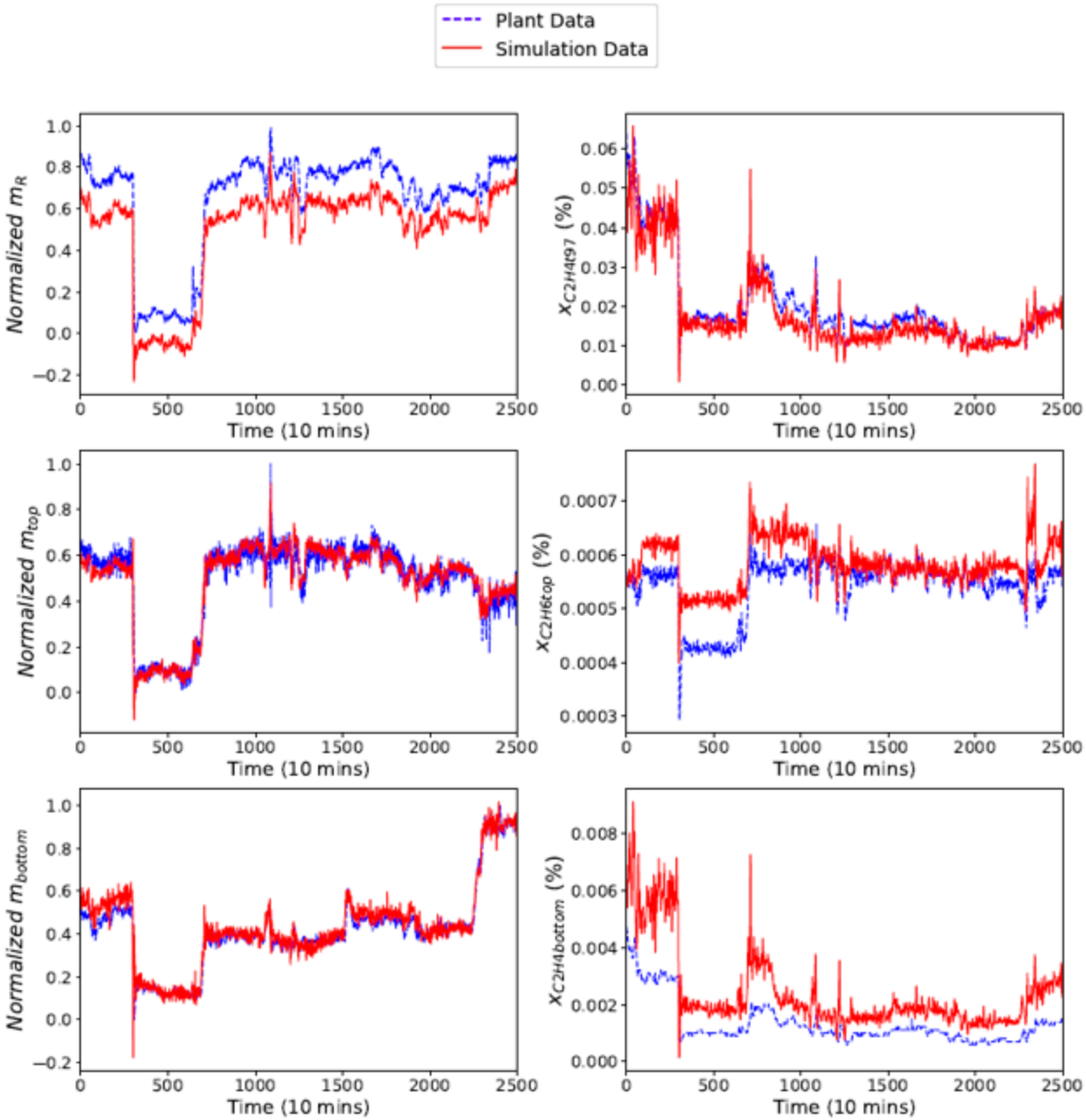

Figure 4. plant measurements versus simulation results

Since the simulation uses controllers set points as given to generate closed-loop data, the behavior of the manipulated variables is used as a measure of the plant-model mismatch. The reflux flow, which is one of the manipulated variables, shows that the simulation data has a similar pattern to the plant measurement with some bias. The reboiler duty plant-model result (included in the Appendix) is less reliable in evaluating the plant-model mismatch since the reboiler duty plant measurement is not measured in the plant but computed using energy balance

on the side stream. Tray 97 ethylene composition simulation data seems to closely match the plant measurements. The top and bottom flow rates of the plant measurement and simulation results match qualitatively. The products composition, ethane mole fraction of ethylene stream and ethylene mole fraction of ethane stream, have the same behavior overall with regions showing a mismatch between the plant and the simulation. Beyond the possible parameter and model structure issues, there are two other key reasons for the plant-model mismatch: (1) the feed composition of the tower is not measured, but instead is computed from upstream process information. As a result, it might not be the actual tower feed composition. (2) tower separation efficiency was assumed to be 0.9 for all trays which is not the case for the real plant. This can be confirmed from comparing the ethylene composition mismatch at tray 97 to the ethylene composition mismatch at the bottom product. The simulation seems to have higher separation compared to the plant at the middle section of the tower which explains why reflux and reboiler duty of the plant measurement are higher than the simulation data. The simulation seems to have less separation compared to the plant at the tower ends, which can be seen in the top and bottom products composition.

The key illustration (and objective) for the use of the dynamic simulator was to develop a test bed that behaves similarly to (and captures the complex behavior of) the plant. Thus, the intent is not to perform parameter estimation using plant data and the simulator model structure to calibrate the simulator. Rather, it simply establishes that the simulator can be a reasonable stand-in for plant operation for testing modeling and control implementations. The figures in this section establish the success of this objective.

4.4. Data driven Modeling of the Ethylene Splitter

In this section, we model the ethylene splitter to evaluate the different system identification methods for three different cases: (1) simulated data with 9 inputs shown in Table 1 (Sim9), (2) simulated data with only 4 inputs (feed flow, feed temperature, reboiler duty, and reflux flow) (Sim4), and (3) plant data with the same 4 inputs as case 2 (Plant4). The Sim9 case study uses all of the available tower inputs to model the ethylene splitter, including inputs that are not measured in the plant. As for the Sim4 case, we model the ethylene splitter using only the inputs measured in the plant to replicate the real ethylene splitter case. Note that in the Sim4 case, the other five inputs essentially act as disturbances. In our analysis, we compare Sim9 and Sim4 to evaluate the benefits of having more input measurements (such as feed composition, pressure,

and vapor fraction) compared to less input measurements. Sim4 and Plant4 are utilized to demonstrate the ability of the approach for modeling the simulator and the plant data. For all three case studies, the 16 output variables shown earlier in Table 1 are being predicted. The metric to evaluate the prediction capability of the methods (and also as a loss function in the NN based approaches) is the weighted mean squared error (WMSE) as shown in eq (5). Both training and testing datasets were normalized using the maximum and minimum values of each variable in the training set as shown in eq (6), so that the WMSE is computed using the normalized predicted and actual values.

$$WMSE = \sum_i W_i \, (\hat{y}_i - y_i)^2 \tag{5}$$

$$y_i = \frac{y_{measured} - y_{min}}{y_{max} - y_{min}} \tag{6}$$

where i is an index of the output, W_i is the weight associated with the output i. $\hat{y}_i$,y_i, and $y_{measured}$ are the scaled predicted, the actual scaled, and the measured outputs respectively. Each one of the 16 outputs have an assigned weight which is included in the Appendix to represents how important this output in our prediction model. The products compositions are assigned the highest weights to emphasize their importance compared to the other outputs. In order to build a data driven model for our C2 splitter based on the available data and to evaluate the impact of the different hyperparameters associated with each model, the first 2,000 data points shown in the Figure 4 with sampling rate of 10 minutes are used. More details about the training of each system identification method are provided in the corresponding subsections below. For clarity, we will use the term training set to refer to the data used in training, the term validation set to refer to the data used in estimate the performance of the models with different hyperparameters, and testing set to estimate the final performance of our model. In this section where different models hyperparameter are investigated we only use training and validation datasets which are used for training and evaluating our model's predictions performance.

Remark – It is important to note that our system identification goal is to predict the output for multi-step ahead using the inputs at these steps. In particular, the aim is to evaluate our model's capability of predicting 20 steps into the future (equivalent to 3 hours and 20 minutes). The ability of the model to provide accurate multi-step ahead predictions is a key to its potential use in an MPC. In particular, an MPC formulation utilizes the model and

candidate future inputs to predict multi-step future outputs where the number of steps represent the prediction horizon. The ability to predict (accurately enough) the future process outputs for a candidate future input trajectory, therefore, implies that the model can be used as part of a successful MPC implementation.

All of the training and prediction results have been generated using a desktop computer with Windows 10 Pro (Intel ® Xeon ® W-2135 @ 3.70GHz CPU, 128.0 GB RAM, and NVIDIA GeForce RTX 2080 Ti GPU). The subspace identification algorithm has been implemented in MATLAB R2018b and it uses the CPU during training. The neural network algorithms have been implemented in Python using Keras framework and run using Jupyter Notebook (anaconda 3) and uses the GPU during training.

4.4.1. Subspace Identification

The subspace identification approach requires only two hyper-parameters: the number of states and the Hankel matrix size (history length). Note that the Hankel matrix size should be slightly higher than number of states. One of the benefits of the subspace identification approach is the uniqueness of solution. Thus, a given data set, with a given set of hyper parameters gives the same solutions (unlike neural network-based approaches). A single run has therefore been used for each different choice of these hyperparameters. The training dataset for all case studies are 1,500 data points, while the validation dataset is 500 data points. Also, the last 20 data points from the training dataset are included in the validation dataset to allow states convergence by the observer before making predictions as shown in eq 5. The WMSE for the training and validation sets have been computed using a 20-step ahead predictions. After every 20 step predictions, the measurements at these future time points are utilized to update the states using a Luenberger observer and then the predictions are carried out again. The Luenberger observer takes the following standard form:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - \hat{y}_k) \tag{6}$$

where A and B are the discrete dynamic system matrices and L is the observer gain matrix. The observer gain matrix was computed using pole placement. In this work, we choose to place the observer pole for each state on the real axis using a normal distribution and a random location between 0 and 0.01. Table 3 reports the WMSE for the different choice of states and Hankel Matrix size. The best hyperparameters (number of states-Hankel matrix size) for the Sim9 and

Plant4 cases are 8 states and 10 Hankel matrix size. The best hyperparameters for the Sim4 case are 12 states and 18 Hankel matrix size. One of the key metrics in comparing our system identification methods is the CPU time and complexity of the training process. The training time for a single model in all three cases was approximately 22 seconds. Thus, the total training time needed to search over the hyperparameters was approximately 308 seconds for each case.

Table 3. Summary of subspace identification results for hyperparameter optimization

Number of states	Hankel Matrix Size	WMSE								
		Sim9			Sim4			Plant4		
		Train	Validation	Training Time	Train	Validation	Training Time	Train	Validation	Training Time
2	4	0.068	0.204	21.8	0.340	0.435	21.7	0.337	0.552	21.8
2	6	0.065	0.160	21.8	0.439	0.393	21.5	0.402	0.641	21.8
2	8	0.072	0.156	22.0	0.422	0.341	21.7	0.375	0.513	21.9
4	6	0.061	0.167	21.8	0.076	0.494	21.7	0.163	0.531	22.1
4	10	0.059	0.157	21.8	0.073	0.500	21.6	0.153	0.498	21.5
4	14	0.047	0.112	21.8	0.076	0.478	21.5	0.174	0.584	21.5
8	10	0.051	0.035	21.7	0.147	0.272	21.6	0.083	0.071	21.6
8	14	0.046	0.040	21.9	0.106	0.208	21.6	0.088	0.073	21.5
8	18	0.051	0.037	21.8	0.093	0.138	21.7	0.090	0.075	21.7
12	14	0.035	0.084	21.9	0.101	0.148	21.7	0.164	0.159	21.6
12	18	0.041	0.060	22.0	0.085	0.109	21.6	0.159	0.121	21.6
16	18	0.091	0.075	21.9	0.195	0.107	21.7	28.84	44.59	21.7
16	22	0.318	0.323	22.1	0.217	0.103	21.8	155.0	222.7	21.8
20	22	0.111	0.098	21.9	0.191	0.162	21.8	0.483	0.448	21.8

The subspace identification method seems to generate reasonably good predictions for all three cases. The subspace identification simulation WMSE results show that Sim9 case predictions are

better compared to Sim4. This is expected since the Sim9 model has more inputs information in comparison to Sim4, such as feed composition which improves the model prediction capability.

4.4.2. NARX Neural Network

The NARX neural network has multiple hyperparameters that need to be optimized such as: number of past inputs and output, number of layers, number of cells in each layer, and activation function of hidden layers. We considered different architectures with 1, 2, and 3 layers, different numbers of past inputs and outputs (6, 10, and 14), and three activation functions in the hidden layers which are relu, sigmoid, and tanh. The number of data used to find the best model is the same 2,000 data points used in the subspace identification. Similar to the subspace identification, the first 1,500 data points were used for training, and the last 500 data points were used for validation. The only difference is that a portion of this training set (the last 200 data points) was not included directly in the training but used to avoid overfitting (we call it holdout set). Note that the holdout set is used to return the model that performed the best on this holdout set so as such it is part of the training dataset. The holdout set was chosen to include the last 200 data points from the training dataset to ensure that the model is performing well over the most recent operations.

The loss function picked is the weighted mean squared error (WMSE) and the inputs and outputs were normalized between values of 0 and 1. Adam optimizer has been adopted to train the neural network with initial learning rate of 0.001 and a batch size of 128 was used. The training process stops when the number of epochs reaches 5,000. The parameters obtained in a certain epoch that perform best on the holdout set are used to evaluate the performance on the validation set. Since the neural networks results can vary based on the initialization of the weights, 3 independent runs for each case were conducted to evaluate the performance of each case. The four models with the best average WMSE on the validation set over the three runs are evaluated further by running an additional 30 runs for each case to find the model with the best hypermeters. Figure 5 summarizes the results obtained for all the best four cases for the 30 independent runes for the Sim9, Sim4, and Plant4 cases, respectively. To establish a fair comparison of the NARX neural network with the subspace identification results, the WMSEs of the NARX neural network model is evaluated on its capability to predict 20-step ahead. Therefore, once the 20-step ahead are predicted recursively, the real values of these predictions are realized and used to predict the first step from the next prediction horizon.

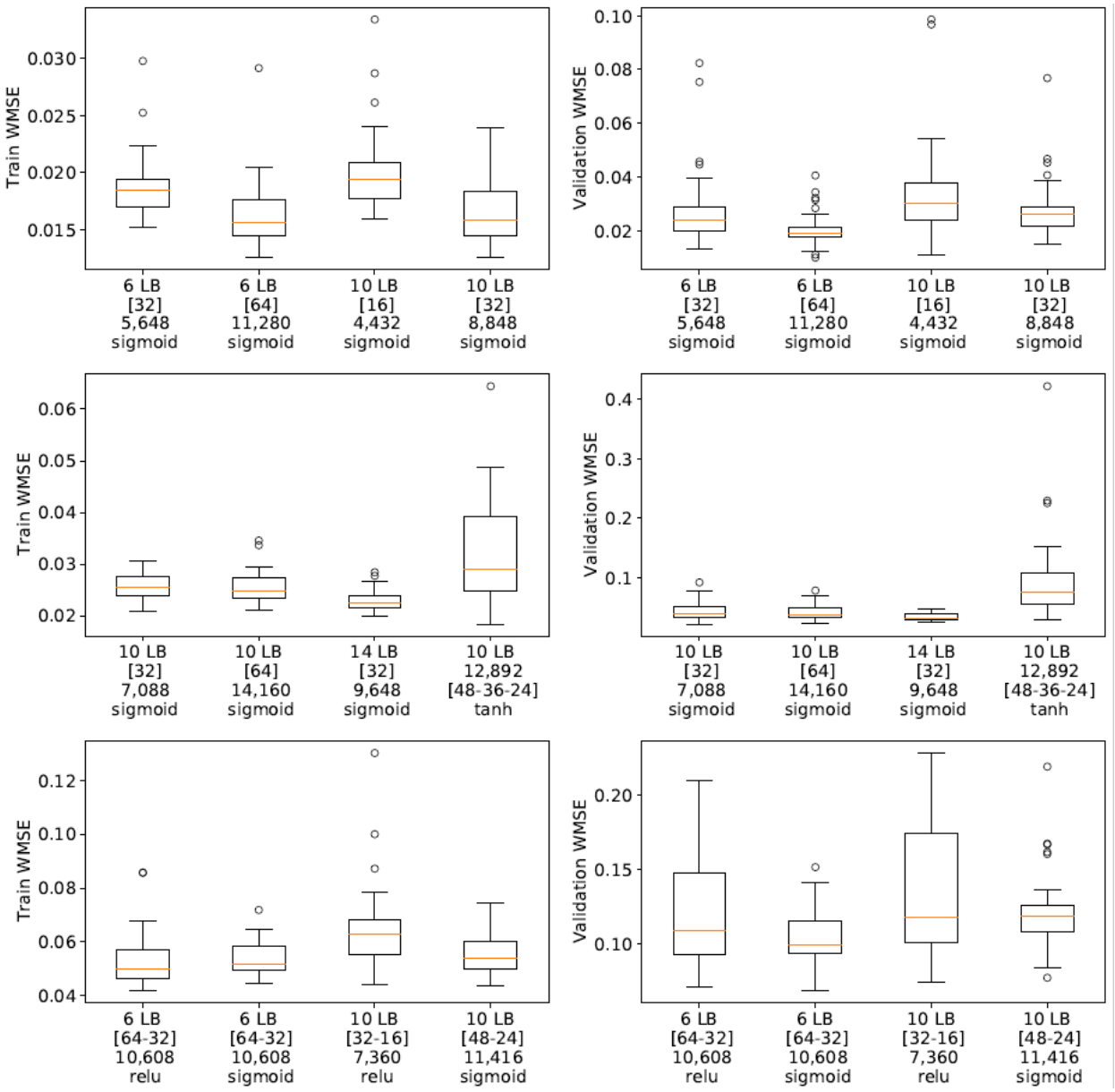

Figure 5: Train and validation WMSE for the best four NARX neural network architecture for Sim9, Sim4, and Plant4

The best model for each case was chosen based on the validation WMSE mean and variance results for the 30 runs. The model with the best results for Sim9 case uses 6 past information, one hidden layer with 64 cells, and the sigmoid activation function in the hidden layers. The model with the best results for Sim4 case uses 14 past information, one hidden layer with 32 cells, and the sigmoid activation function in the hidden layers. The model with the best results for Plant4 case uses 6 past information, two hidden layers with 64 and 32 cells, and the sigmoid activation function in the hidden layers. The sigmoid activation function is more suitable for modeling our

process due to its ability to provide higher nonlinearity fitting compared to relu and lower range compared to tanh. Also, it is important to note that the trained WMSE is lower than the validation WMSE which indicates that the model is overfitting on the training sets even with using part of it as holdout set. The training time for a single model was roughly 260 seconds. The total training time needed to search over the hyperparameters was roughly 28 hours for each case. Note that the training for the NARX neural network was performed on the GPU.

Once the optimal hyperparameters for each case has been found, all the 1,500 data points in the training data were used to train the model and a holdout set was not used. The NARX neural network method seems to generate reasonably good predictions for all three cases. The NARX neural network simulation WMSE results show that Sim9 case predictions are better compared to Sim4. This is expected since the Sim9 model has more inputs information such as feed composition compared to Sim4, which improves the model prediction capability.

4.4.3. Recurrent Neural Network

In this work, our aim is to model the recurrent neural networks (RNNs) as a state-space model (similar to subspace identification). While the use of subspace identification and NARX neural network methods for dynamic modeling in chemical engineering processes has been well studied, the use of RNNs for the same purpose has not been sufficiently pursued. Therefore, different RNN training architectures have been investigated before selecting the best architecture to use to model our distillation columns. In this section, we will discuss the different strategies to train the RNN and introduce our proposed training architecture and its results.

RNN can be trained as a single-step model or multi-step model. In a single-step setup, the model is trained to predict a single output in the future by feeding it the current input and the previous state as inputs and the current output as output as shown in eq (6). A single-step model can still be used recursively to perform a multi-step prediction. In a multi-step setup, the model is trained to predict a sequence of range future outputs using a range of future inputs and previous states. The multi-step model can used to do multi-step predictions directly. In this work, we train our model as a multi-step prediction model.

Since the RNN is modeled as a state space model, the initial state of the RNN has a direct effect on the immediate and transient response of the network. Since these states do not have any meaningful physical property, physical measurements cannot be used directly as the initial

values of the hidden states, similar to subspace identification, and state initialization is needed. It is important to note that a state-space model is equivalent to an input-output model, but the training procedure of the two models generate different results [37]. The work of Sum et. al [38] have showed that every RNN can be transformed to a NARX model, and vice versa, under some conditions. The common approach to initialize the states of the RNN is the washout method, In the washout method, the states are initialized with values of zero or random values and then the RNN is run for several steps until the effect of the initial state values wash out. This method gives poor state initialization due to two main reasons. First, the predictions during the washout period can be too inaccurate to use since they can vary for each input trajectory based on the output trajectory. Second, the RNN may experience some unstable situations during the training which causes the sates to explode within the washout period. [39].

The RNNs is normally trained using the Back-Propagations Through Time (BPTT) algorithm which can suffer from a vanishing or exploding gradient [40]. Therefore, Long Short-Term Memory cells (LSTMs) are used in this work instead of simple RNN since they can resolve these issues [41]. LSTMs facilitate the flow of information throughout a network by employing cells equipped with gates to remember, forget, or output information. This causes the LSTM networks to have two types of states: cell states and hidden states, where simple RNN networks have only hidden states. Throughout the paper we use states to refer to both cell states and hidden states. More details about the LSTM networks used in this work can be found in [42].

In this work, we utilized two RNNs; the first RNN is used as an observer to find the initial states and the second RNN is used as a predictor of future outputs by using the initial states and the future inputs. Figure 6 shows the proposed RNN which includes the observer and the predictor RNN.

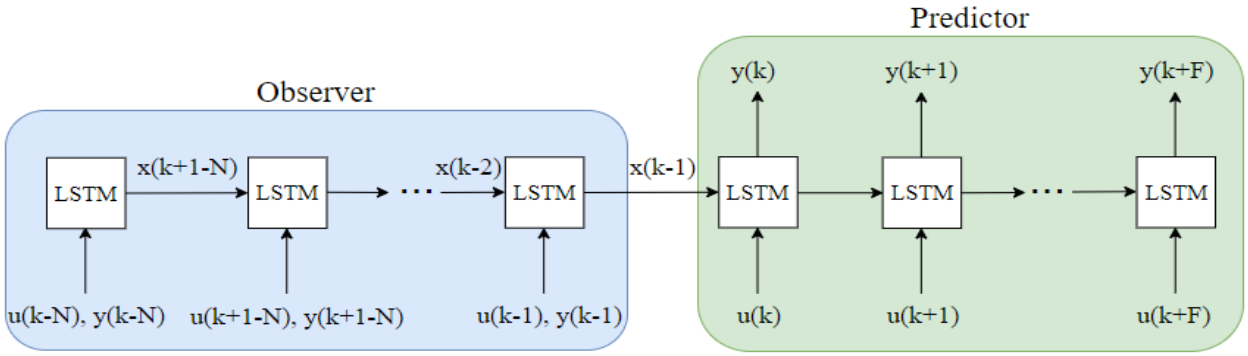

Figure 6: RNN architecture with observer and predictor

The observer takes both multi-step of past inputs and outputs to build the states, which are passed as initial states to the predictor. The predictor uses these initial states along with multi-step future inputs to predict multi-step future outputs. In this work, we used the past multi-step of the observer (N) to be equal to 10 (similar to the subspace identification), and the multi-step predictions of the predictor (F) to be equal to 20 (number of multi-step ahead predictions). We chose 20 as the multi-step ahead predictions to make the results comparable to the subspace identification and the NARX neural network methods, since both methods were used to recursively predict 20 steps ahead.

RNNs have multiple hyperparameters that need to be optimized such as: number of layers, number of cells in each layer, and number of past inputs and outputs used in the observer RNN. The number of past inputs and outputs used in the observer RNN is chosen to be 10 to fairly compare to the subspace identification. Also, the LSTM implementation used in this work is cuDNNLSTM which can only have tanh as the activation function. We have considered different architectures for the observer and predictor with 1 and 2 layers. The number of data used to find the best model is the same 2,000 data points used in the first two approaches. The first 1,500 data points were used for training, where a portion of this training set (the last 200 data points) were used as holdout set, similarly to the NARX neural network method. The last 500 data points were used for validation. The loss function is the weighted mean squared error (WMSE) and the inputs and outputs were normalized between values of 0 and 1. Adam optimizer has been adopted to train the neural network with initial learning rate of 0.001, and a batch size of 128. The training process stops when the number of epochs reaches 5,000. The parameters obtained in a certain epoch that perform best on the holdout set are used to evaluate the performance on the validation set. Since the neural networks results can vary based on the initialization of the weights, 3 independent runs for each case were conducted to evaluate the performance of each case. The three cases with the best average WMSE on the validation set over the three runs are evaluated further by running 30 additional runs for each case to find the model with the best hypermeters. Figure 7 summarizes the results obtained for all the best three cases for the 30 independent runs for the Sim9, Sim4, and Plant4 cases respectively.

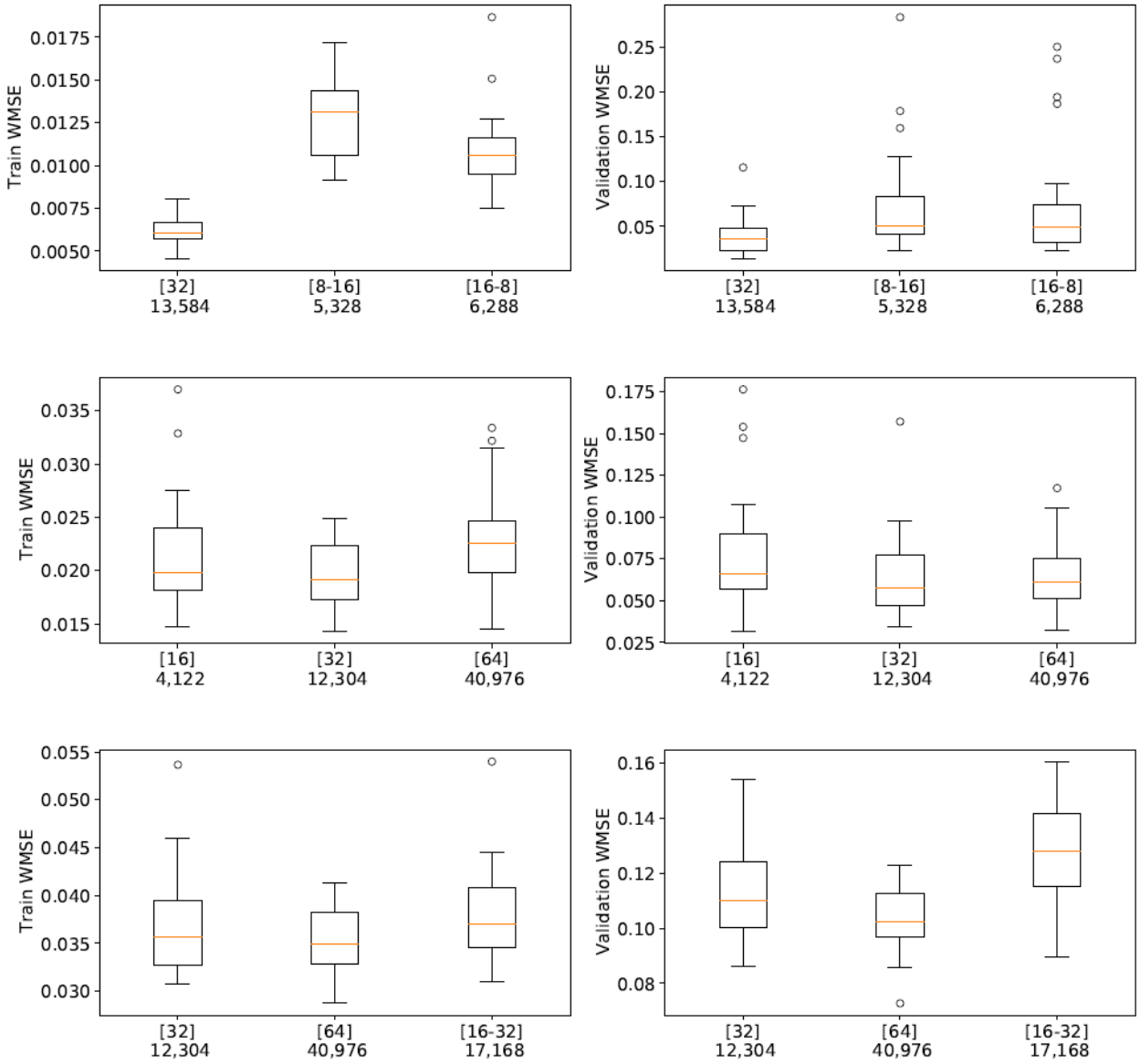

Figure 7: Train and validation WMSE for the best three RNN architectures for Sim9, Sim4, and Plant4

The best model for each case was chosen based on the validation WMSE mean and variance results for the 30 runs. The model with the best results for Sim9 and Sim4 cases use one hidden layer with 32 cells. The model with the best results for Plant4 uses one hidden layer with 64 cells. The train WMSE is much lower than the validation WMSE, which indicates that the RNN is overfitting on the training data. The training time for a single model was between 600 and 1,000 seconds based on the model size. The total training time needed to search over the

hyperparameters was roughly 28 hours for each case. Note that the training for the RNN was performed on the GPU.

Once the optimal hyperparameters for each case has been found, all of the 1,500 data points in the training data have been used to train the model, without the use of holdout set. To alleviate overfitting, a dropout technique was used to regularize our neural network training. We applied dropout to the hidden states between the LSTM cells between the time steps (horizontal connection in Figure 6) on the observer and the predictor. Also, since cuDNNLSTM implementation does not support dropout within the recurrent cells, the final LSTM implementation used simple LSTM instead of cuDNNLSTM and it was trained on the CPU since it was faster than training it on the GPU. The values, 0.8 and 0.2, were used as dropout rates on the observer and predictor, respectively.

4.5. Results and Discussion

In this section we compare and discuss the results and challenges of the three system identification methods. Table 4 gives a summary of the results achieved by the system identification methods on the three different case studies. The subspace identification and the RNN methods were trained on the CPU, while the NARX neural network was trained on the GPU. When comparing each case individually, it can be observed that a higher number of parameters results in longer training time and lower WMSE on the training data. The neural network methods (NARX and RNN) have better results on the validation dataset compared to subspace identification for the Sim9 and Sim4 cases. As for the Plant4, the subspace identification showed better results on the validation dataset compared to the neural network methods. These results indicate that subspace identification can deal better with measurement noise associated with the Plant4 case compared to the simulation cases. Also, it shows that neural networks-based approaches (NARX and RNN) are capable of outperforming subspace identification when their architectures are optimized to predict these specific data. This shows that neural network models with large number of parameters can provide good predictions even when trained using small amount of data.

Table 4: Summary of the three system identification results on the three different case studies on training and validation dataset

Case	Modeling Approach	# Parameters	WMSE Train	WMSE validation	Training Time
Sim9	Subspace Identification	408	0.051	0.035	21.7
	NARX Neural Network	11,280	0.012	0.018	187.5
	RNN	13,584	0.005	0.008	1174.4
Sim4	Subspace Identification	400	0.085	0.109	21.6
	NARX Neural Network	9,648	0.024	0.021	215.5
	RNN	12,304	0.011	0.034	1152.9
Plant4	Subspace Identification	288	0.083	0.071	21.6
	NARX Neural Network	10,608	0.047	0.125	215.6
	RNN	40,976	0.018	0.104	1264.9

Neural network methods (NARX and RNN) require more time to optimize the hyperparameters and obtain the final architecture of the model due to two reasons: (1) the training time for a single model using neural network methods required approximately an order of magnitude larger compared to subspace identification. (2) the training of subspace identification model is stable, and the solution generated is unique compared to neural network methods solutions. Therefore, a single run is needed to evaluate a specific architecture of subspace identification, while multiple runs are needed to evaluate a specific architecture of neural network methods. Also, it is important to understand how the testing dataset behaves compared to the training dataset to further understand the comparison of the three system identification approaches.

In order to explain the difference in operations between the training and validation dataset used for hyperparameter optimization purposes the score plots of the principal components analysis (PCA) based on all the input variables for the Sim9, Sim4, and Plant4 cases are shown in Figure 8. In this figure, the first two components are used to illustrate the nature of the operating patterns in the training and validation dataset. Samples closer to each other represent similar operating patterns, while samples farther away from each other have distinct operation patterns.

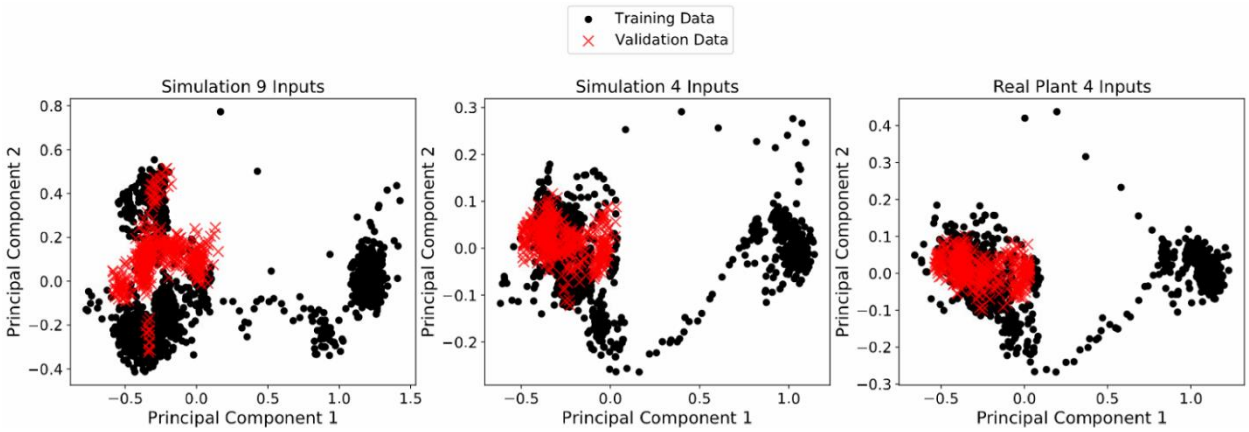

Figure 8. PCA scores plot of inputs variables for the three studied cases for hyperparameters optimization

The figure shows that the validation dataset has similar operating patterns to the training dataset for all three cases. Also, it can be observed that simulation and real plant cases with four inputs look similar which confirms that while model-plant mismatch exists, as observed in the previous section, both data share similar key patterns. This indicates that the results in Table 4 are evaluating the models' capability at interpolating (validation dataset is similar to the training dataset).

Remark – The neural network system identification methods (NARX and RNN) have the capability to outperform subspace identification for ethylene splitter when they are trained and optimized to predict a specific dataset. This does not show that neural network methods could provide better results on "true testing data" (data that has not been used in training or optimizing hyperparameters), it only shows that when dealing with steady state data where the dynamic is not changing, the neural network methods have the capability to outperform subspace identification.

Next, we evaluate our models' performance on new data that have not been used in the training including hyperparameter optimization, which we call the testing dataset. It is important to note that the training data is also different than the one used in the previous results since we always aim to use the most recent data available for training. To provide an understanding of the change in operations of our training and testing data the score plots of the principal components analysis (PCA) based on all the of input variables for the three case studies are shown in Figure 9. In this figure, the first two components are used to demonstrate the information about the operating patterns in the training and testing data. Samples closer to each other represent similar operating

patterns, while samples farther away from each other have distinct operation patterns. This shows that for all case studies, the first 260 samples of the testing data are similar to the training samples, meaning they have similar operation patterns. While the last 240 samples of the testing data are different from the training samples meaning the process is transitioning to new operating conditions. Therefore, model capability over the first 260 and last 240 samples are being denoted as `interpolation' and `extrapolation', respectively.

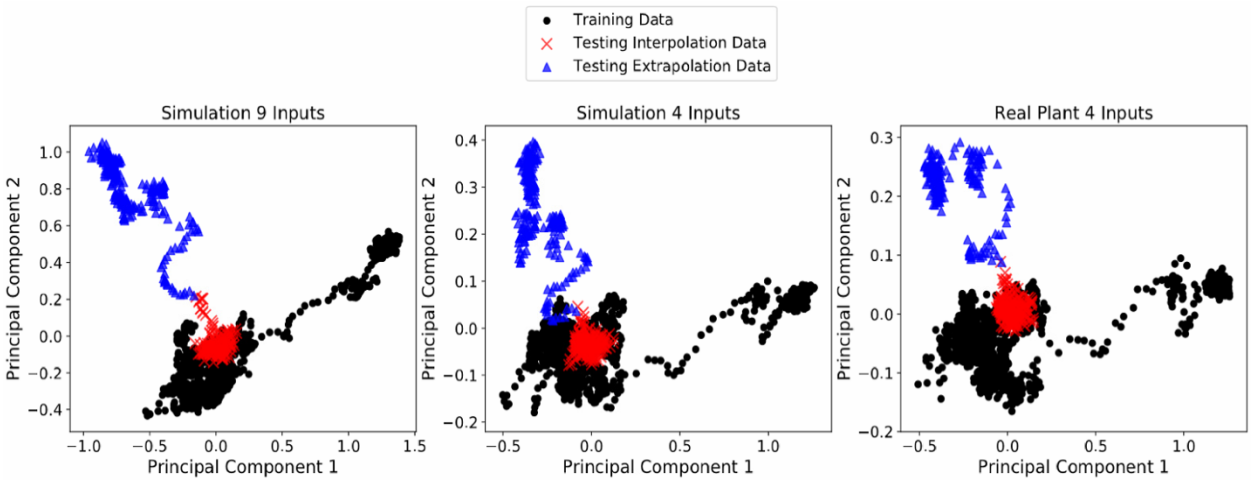

Figure 9. PCA scores plot of inputs variables for the three studied cases for online

Table 5 gives a summary of the results WMSE achieved by the system identification methods on the three different case studies for the new unseen testing data. Figures 10 to 12 demonstrate the predictions of the top product flow rate and composition on the testing dataset for the three different system identification models for the cases Sim9, Sim4, and Plant4 respectively. Note that the predictions are updated every 20 sampling instances. Thus, the outputs converge to the measured variables every 20 sampling instances, but if the process dynamics are mismatched from the model, they diverge again. The subspace identification has better results for Sim9 and Plant4 cases compared to neural network methods (NARX and RNN), while neural network methods have better results for Sim4 case. For Sim9, subspace identification outperforms neural network methods in both interpolation and extrapolation. One reason for such results is that the interpolation data still represents slightly different operations compared to the training data as can be observed from the Figure 9 PCA scores for the Sim9 case. This means that the neural network methods results can deteriorate greatly if the testing data is slightly different than most of the training dataset, where subspace identification performance is not impacted as much by such small different between the training and testing datasets. For Plant4, the neural network

methods seem to outperform subspace identification for interpolation data, but subspace outperform neural network for extrapolation data. These results confirm the tendency of NARX neural network and RNN to overfit (and thus be better at interpolation) and thus, are not as suitable for extrapolation purposes.

The above also points to a somewhat fundamental limitation with neural network models (NARX and RNN). It is observed that they are unable to revert to or naturally result in a simpler model when a smaller dataset is available, and more importantly, when the dynamics being expressed in the smaller data set are relatively less complex (in the sense that they can be adequately expressed using a linear model). In general, neural network models (NARX and RNN) have better predictions when using smaller number of input cases (Sim4 and Plant4) as opposed to subspace identification. This is mainly due to the small number of training examples causing the model to be more prone to overfitting when more input information is available.

Table 5: MSE of the three system identification results and training times on the three different case studies on testing dataset

Case Studies	Data	Subspace Identification	NARX Neural Network	RNN
Sim9	All WMSE	0.518	3.752	4.948
	Interpolation WMSE	0.004	0.017	0.040
	Extrapolation WMSE	1.075	7.798	10.265
	Training Time	21.7	241.3	1060.4
Sim4	All WMSE	26.452	0.958	2.419
	Interpolation WMSE	0.025	0.018	0.037
	Extrapolation WMSE	55.080	1.976	4.999
	Training Time	21.6	289.8	1089.7
Plant4	All WMSE	0.630	1.499	1.541
	Interpolation WMSE	0.113	0.079	0.088
	Extrapolation WMSE	1.191	3.037	3.115
	Training Time	21.6	313	1858.8

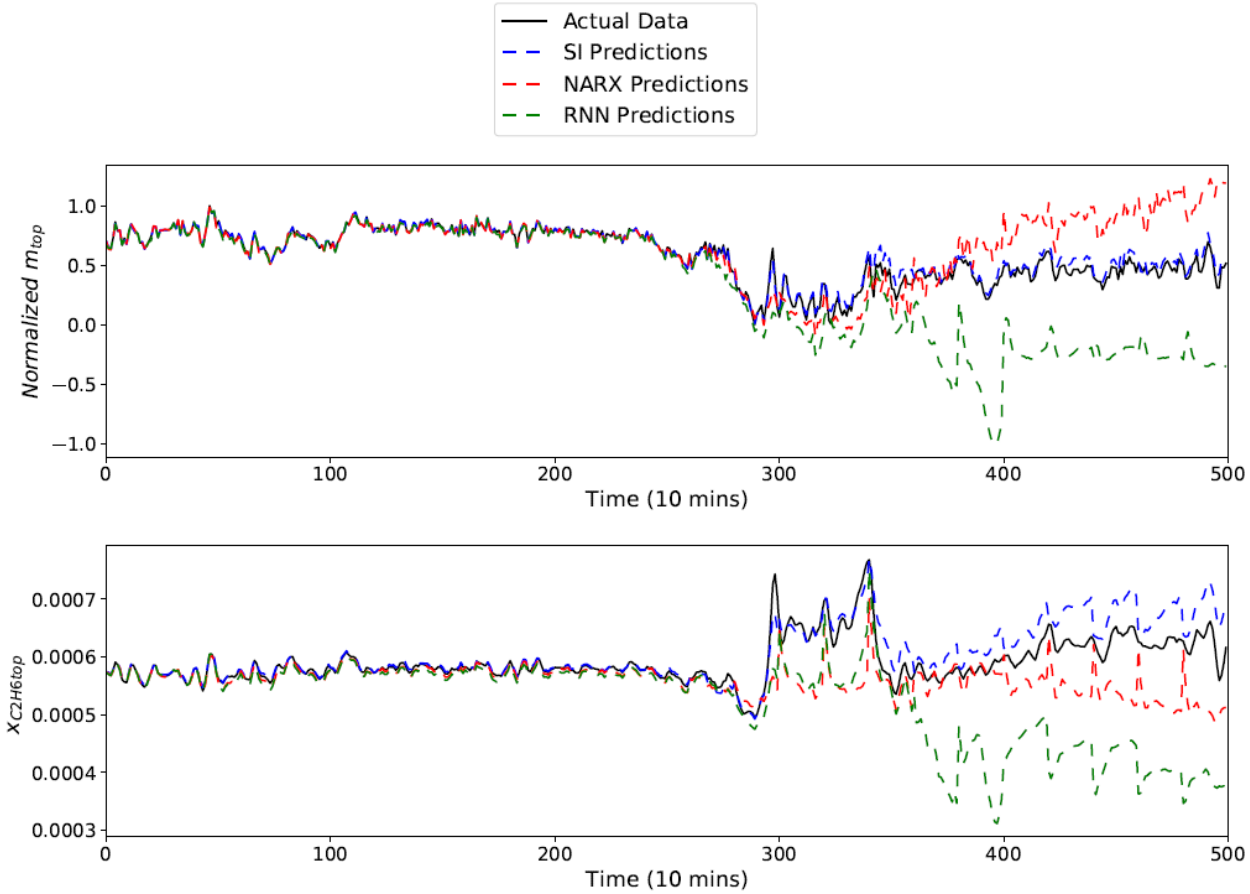

Figure 10: Results of the three system identification methods on testing dataset for Sim9 case

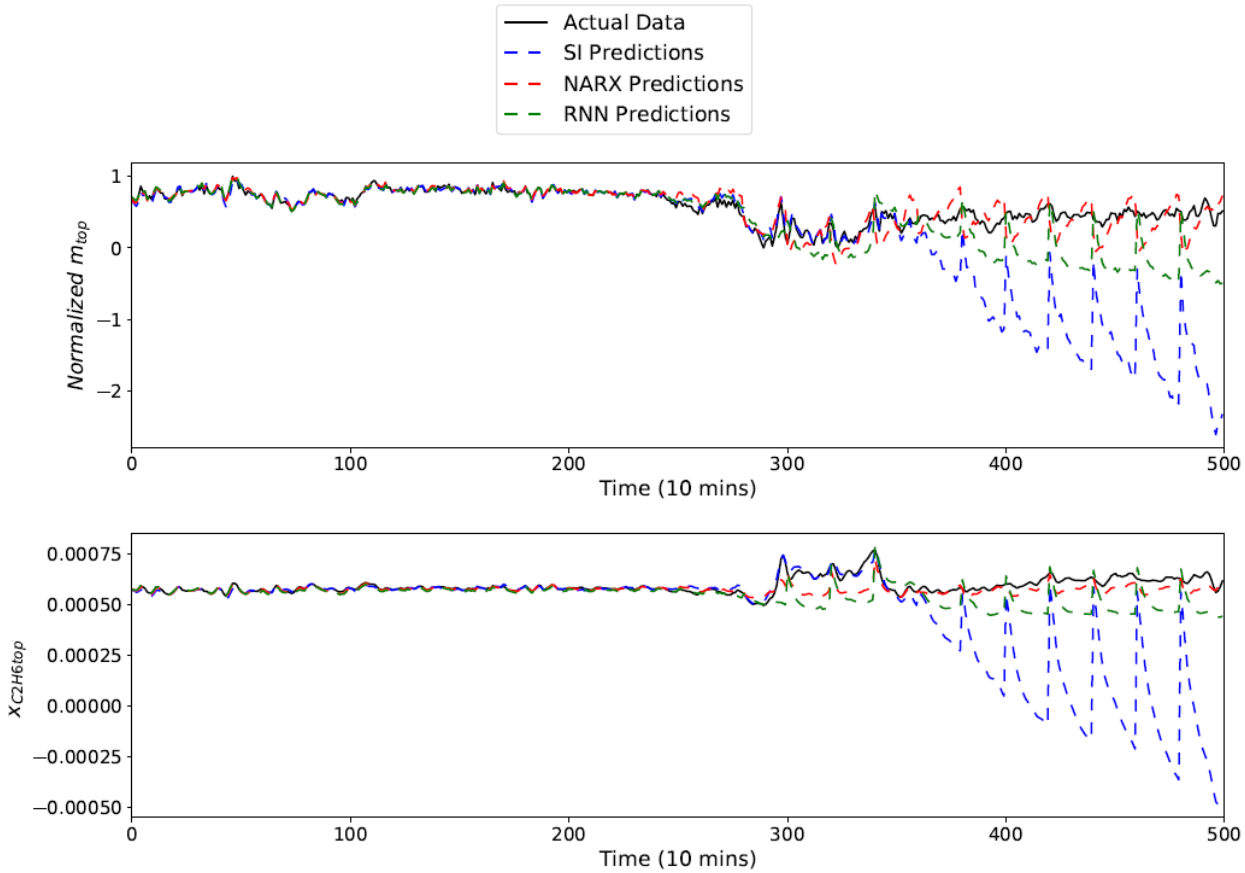

Figure 11: Results of the three system identification methods on testing dataset for Sim4 case

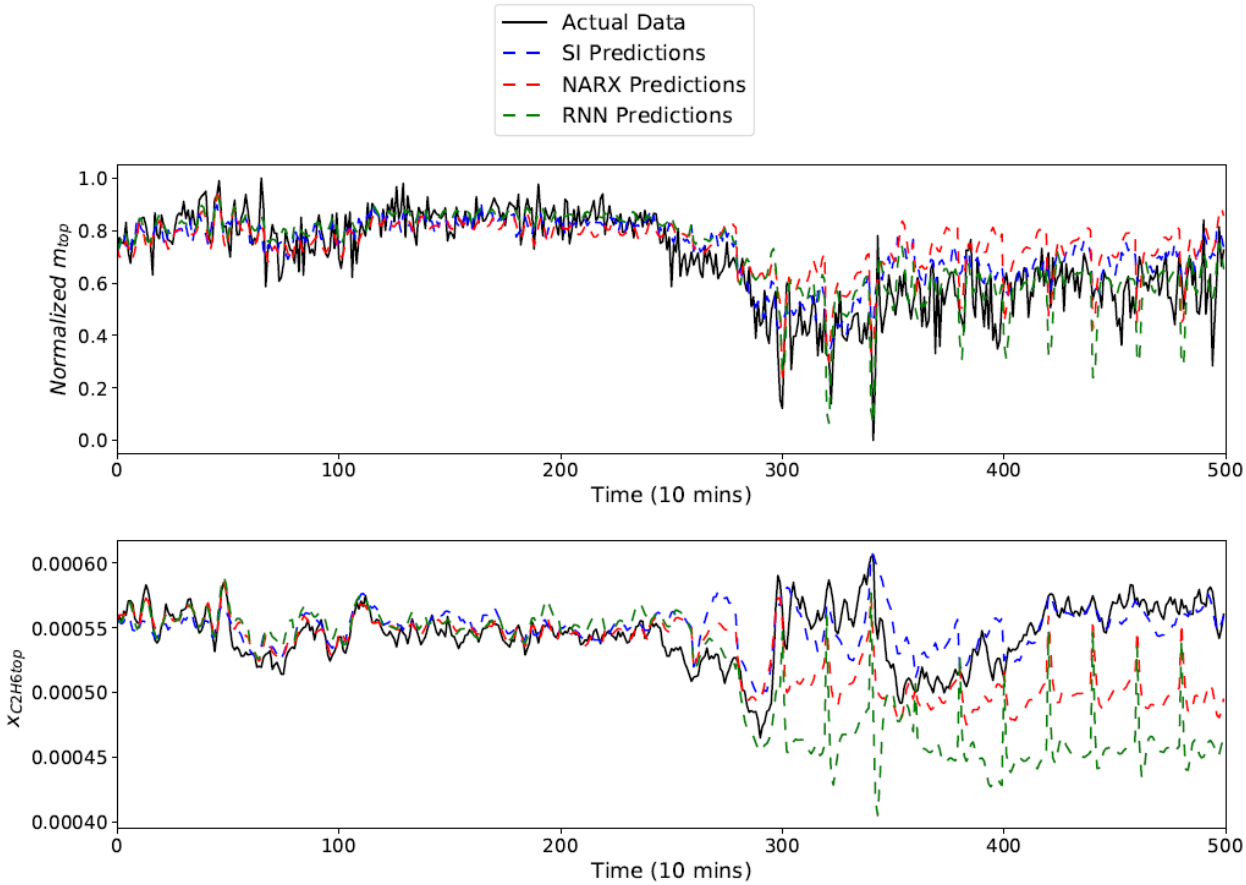

Figure 12: Results of the three system identification methods on testing dataset for Plant4 case

4.6. Online Model Updating scheme

In the previous section, we illustrated our different system identification methods for the ethylene splitter and analyzed the influence of different hyperparameters on each model's performance. Finally, we evaluated our developed model on new data (testing dataset). In this section, we propose different online system identification algorithms to improve the model predictions capability and evaluate it. One of the key things to consider when implementing system identification online is model validity over time and when it is necessary to update the model. An online modeling scheme to keep our model relevant to the most recent operation is needed because our model is built based on limited data and there are frequent changes in operating patterns causing models' prediction capability to degrade. Therefore, updating the model based on the most recent measurements might server to improve model validity for the current process operation. Even in the case where our training data span the full range of process operation, some of our disturbance variables are not measured and not considered when building

the model. This can cause model degradation when the unmeasured disturbances differ from the training data considerably. Finally, the model update becomes fairly useful for the subspace identification method since the ethylene separation process is nonlinear and subspace identification method generates an inherently linear model.

Consider n is the set of the initial training samples, and Δn is subset of the training samples that corresponds to the initial 20 samples in n. m is the set of the whole testing samples, and Δm is subset of the testing samples that corresponds to the initial 20 samples in m. The Δm samples correspond to the prediction horizon needed in the MPC implementation, which is assumed to be 20 in this work. Our online model updating algorithm starts by using the initial training samples n to determine the minimum and maximum values of the inputs and outputs variables for normalizing purposes. Then the n training samples are normalized and used to build our system identification model. Next, the identified model is used to predict the Δm samples outputs using their inputs. After the actual values of the testing sample Δm are available, they are added to training data n and the first Δn samples of the training data are removed (so that we always have 1500 samples for training). The model is then re-identified using the new normalized training samples n before being used to predict the next Δm testing samples. The training algorithm for online scheme is summarized in Algorithm 1.

Algorithm 1 online model update scheme

- Use the initial training data (n) to determine the minimum and maximum values of all the inputs and outputs variables for normalizing purposes.

while termination criterion not fulfilled **do**:

- Normalize the inputs and output variables of the training data (n) and testing data (Δm) using the minimum and maximum value of the initial training data.
- Train the model using the normalized input variables and normalized output variables of the training data.
- Use the normalized input variables of the Δm testing data to predict the normalized output variables.
- Anti-normalize the normalized output variables to obtain nominal predictions of the output variables.
- Update the training data by removing the first Δn samples and adding Δm samples of actual realized values of the testing data.
- Update the testing data by considering the next Δm samples of the testing data.

end while

While Algorithm 1 can improve the model prediction capability by re-identifying the model based on the most recent operations after every Δm samples, a rapid change in the operations, unmeasured disturbances specifically, might still result in poor performance since the model is only updated after every Δm samples. Therefore, we propose another algorithm based on model monitoring and error tracking by comparing model predictions with observed outputs. The idea is to trigger model re-identification when the error between the model prediction and observed value exceed a specific threshold or after every Δm samples. Since a big change in operation conditions can results in triggering re-identification after only a single prediction and recognizing that adding that single point to the training data may (and should not) impact the model greatly, we put a lower limit (k) on how many new data points are needed before triggering model re-identification. The online model update with error tracking scheme is summarized in Algorithm 2.

Algorithm 2 online model update with error tracking scheme

- Use the initial training data (n) to determine the minimum and maximum values of all the inputs and outputs variables for normalizing purposes.

while termination criterion not fulfilled **do**:

- Normalize the inputs and output variables of the training data (n) and testing data (Δm) using the minimum and maximum value of the initial training data.
- Train the model using the normalized input variables and normalized output variables of the training data.
- Use the normalized input variables of the Δm testing data to predict the normalized output variables.
- Compute the Weighted Mean Squared Error (WMSE) using the normalized predicted output and the normalized observed output for the Δm testing samples. WMSE($i = 1, \dots, \Delta m$).
- For loop ($i = \mathrm{k}, \dots, \Delta m - 1$):
    - if MWSE(i) $\geq threshold$:
        - Break and return i as I
    - else: $\mathrm{I} = \Delta m$
- Anti-normalize the normalized predicted output variables ($1, \dots, I$) to obtain nominal predictions of the output variables.
- Update the training data by removing the first I samples and adding I samples of actual realized values of the testing data.
- Update the testing data by removing the first I samples and adding the next I samples of the testing data.

end while

One challenge with the two algorithms introduced is it needs to check if the most recent data have sufficient excitation to obtain a reliable model. In this work, the subspace identification algorithm checks if there is sufficient excitation in the data before applying system identification. It is important to note that one of the tuning parameters in the subspace identification is the number of states. Therefore, for some given choice of number of states, the method may report that the data is not sufficiently excited, so it will only identify a model which is 'consistent' with the observed excitation. As for the neural network methods, model re-identification uses the previous identified model which was trained on previous data (supposedly have enough excitement) as a starting point then modifies these parameters by training the network on the more recent data. Thus, the network should still preserve some of the learning that was done on the initial data which has sufficient excitement. In the present implementation, the subspace identification has always showed that excitation was sufficient to reidentify a new model every time model re-identification was required. One reason that our data was informative to trigger system identification always is due to the relatively high disturbance (feed flow, feed temperature, feed composition, etc.).

4.7. Ethylene Splitter Online Modeling Scheme Results

To demonstrate the online modeling schemes introduced in the previous section, we use the same training and testing datasets used earlier for the three case studies. In this section, we consider 1,500 training samples (n) for case studies Sim9, Sim4, and Plant4. Also, we consider 500 testing samples (m) for testing all case studies. The 500 testing samples are predicted recursively 20 samples (Δm) at a time with an updated model. The 20 samples which correspond to a duration of 3 hours and 20 minutes represent the prediction horizon needed in the MPC implementation.

4.7.1. Online Subspace Identification Results

In this section, we implement the two proposed online model update algorithms using the subspace identification method and compare it to the results obtained without model update to show the effectiveness of our algorithms. The threshold parameter in Algorithm 2 used to trigger model re-identification before 20 steps are picked based on the WMSE computed from Section 4. The thresholds for the three cases (Sim9, Sim4, and Plant4) were chosen to be 1.5 times the WMSE value computed in Section 4. This was chosen to control the number of model re-

identification triggers since it was occurring too frequently due to high noise possibly. One thing to note is that Algorithm 2 predictions are expected to be better compared to no model update predictions for two factors; (1) updating the model based on most recent operations makes the model learns more relevant information on how the system is behaving at these operating conditions. (2) every time model re-identification is triggered; the observer is used to estimate the current state variables which improve the predictions. In order to show the impact of the first factor (model update) separately, we run another algorithm (Algorithm 3) that uses triggered steps obtained from Algorithm 2 to re-estimate the states at these steps before making predictions. Table 6 shows the WMSE for the no-model update versus our three algorithms over the 500 samples, the first 260 samples, and the last 240 samples.

For all three cases (Sim9, Sim4, and Plant4), the WMSE results show that frequently updating the model based on the more recent operations can greatly improve the predictions when the process is going to a new operation mode (extrapolation). When comparing the extrapolation WMSE results for the no update method with Algorithm 3, we can see that reinitializing the states using the observer is a factor in improving the prediction but model reupdate is still the key factor, which can be observed from comparing the WMSE from Algorithm 2 versus Algorithms 3. Also, model reupdate does not give significant (if any) improvement on interpolation dataset. Another thing to observe is that Sim9 results are greatly better than Sim4, which shows the benefits of having some future inputs information (especially tower feed composition) on the model prediction capabilities. One unforeseen key result was that the subspace identification generated better extrapolation predictions for Plant4 compared to Sim4 with the model update and without. One reason for such results might be because the real plant data goes through data compression, causing it to be constant for some period of time if it is not changing much, which is easier to predict for the linear subspace identification model. The model seems to need more frequent re-identification when going through extrapolation data compared to interpolation, as expected. Also, Sim4 and Plant 4 cases need more frequent model reupdates compared to Sim9 because they do not have information about the feed composition which is important when predicting products compositions. Figures 13 and 14 show the subspace identification results for Plant4 case on interpolation and extrapolation datasets, respectively. For brevity purposes, the results for Sim4 and Sim9 cases are not included. The dash horizontal lines in the figures represent where system re-identification occurred in Algorithm 2. Also, it is

important to note that subspace re-identification takes the same time reported in Table 5 since it requires identifying the model from scratch.

Table 6: WMSE for subspace identification method with no model update versus the other three algorithms

Case Studies	Data	No Update	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 2 re-identification trigger
Sim9	All	0.518	0.043	0.024	0.415	39
	Interpolation	0.004	0.003	0.003	0.004	13
	Extrapolation	1.075	0.087	0.046	0.861	26
Sim4	All	26.452	0.737	0.404	23.922	44
	Interpolation	0.025	0.021	0.019	0.018	13
	Extrapolation	55.080	1.512	0.822	49.819	31
Plant4	All	0.630	0.235	0.092	0.319	58
	Interpolation	0.113	0.094	0.056	0.070	17
	Extrapolation	1.191	0.387	0.130	0.588	41

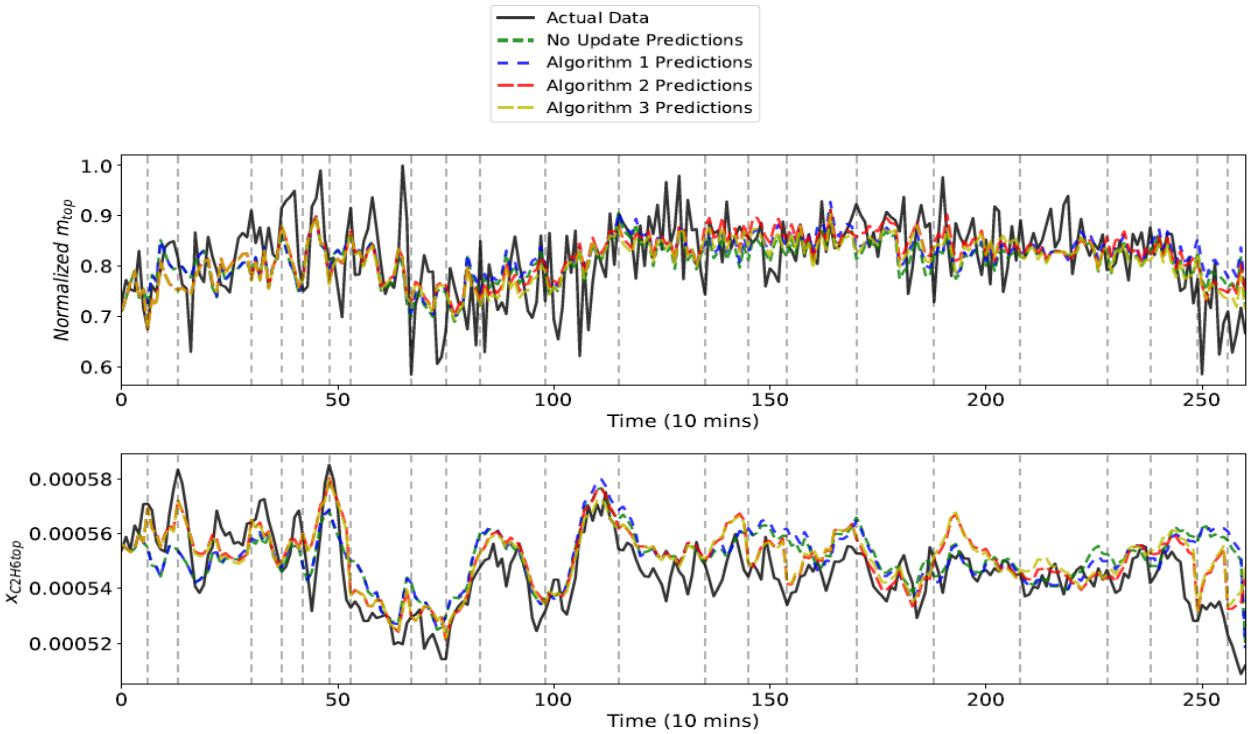

Figure 13. Extrapolation data prediction with subspace identification for the real plant data with no update and the three algorithms

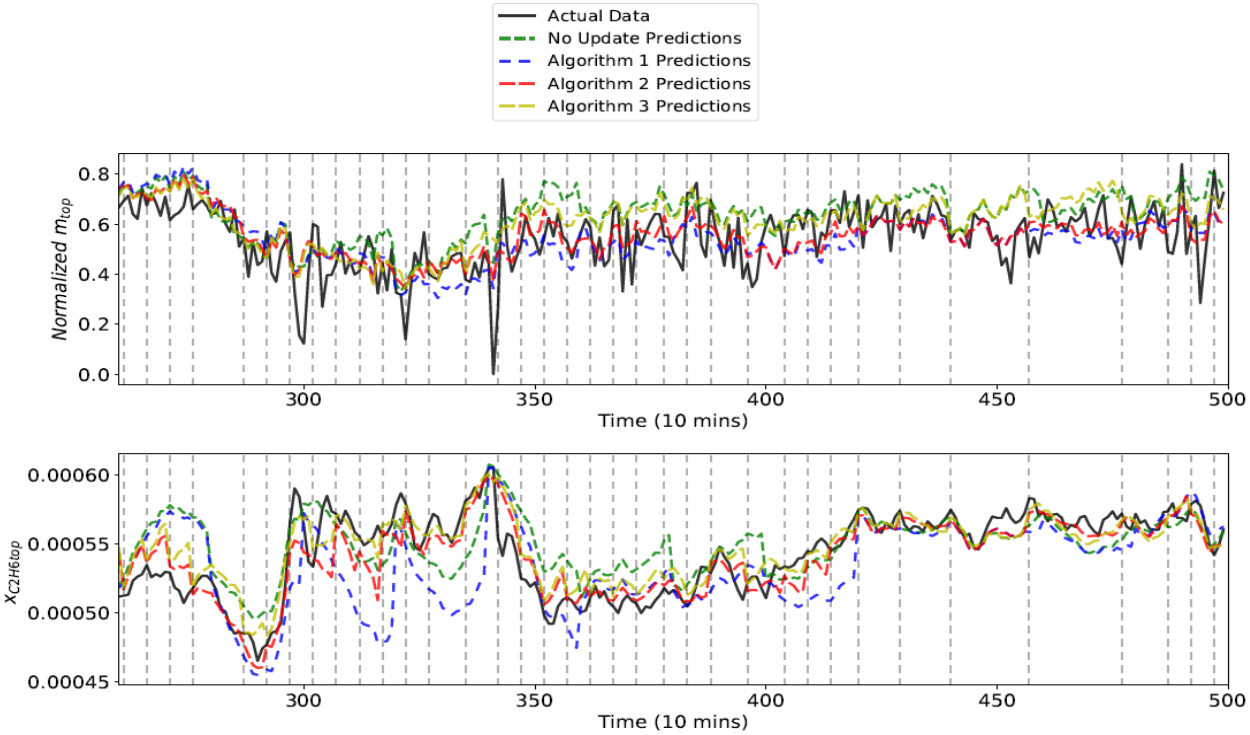

Figure 14. Interpolation data prediction with subspace identification for the real plant data with no update and the three algorithms

4.7.2. Online NARX Neural Network and RNN Results

In this section, we implement the two proposed online model update algorithms using the NARX neural network and RNN methods and compare them to the results obtained with no model update to show the effectiveness of our algorithms on these neural network system identification methods. When performing model reupdate in subspace identification, the model is identified from scratch every time the model reupdate is performed. When reupdating the model for neural network methods, the model parameters of the last identified model are used to initialize the parameters of the new model before training it on the new training data for only 500 epochs. This is done to lower training time and utilize the information that have already been captured by the previous training runs. The threshold parameter in Algorithm 2 used to trigger model re-identification before 20 steps are picked based on the WMSE computed from Section 4. The thresholds for the three cases (Sim9, Sim4, and Plant4) were chosen to be 1.5 times the WMSE value computed in Section 4. This was chosen to control the number of model re-identification triggers since it was occurring too frequently due to high noise possibly. Table 7 and 8 shows the WMSE for the no-model update versus our three algorithms for the NARX neural network and RNN methods, respectively.

Tables 7 and 8 show that for all three cases (Sim9, Sim4, and Plant4), frequently updating the model based on the more recent operations can greatly improve the predictions when the process is going to a new operation mode (extrapolation). Also, model reupdate does not give significant (if any) improvement on interpolation dataset, similar to subspace identification. One unforeseen key result was that NARX neural network and RNN gives better predictions on Sim4 compared to Sim9 for the no update method, which is opposite to the results obtained from subspace identification. One reason for such results is that the NARX neural network and RNN are overfitting when they are provided with more inputs information causing them to generate bad results and model reupdating becomes more necessary. RNN model seems to be more prone to overfitting compared to the NARX neural network model which might be due to our chosen architecture (RNN has a greater number of parameters than NARX neural network). This can be observed from the fact that both models are performing closely similar on interpolation dataset, but NARX neural network outperform RNN on extrapolation dataset. Also, both models seem to have similar performance on Plant4 and Sim4 cases, where subspace identification performance on Plant4 was greatly better than Sim4. Figures 15 and 16 show the NARX neural network

results on Plant4 case for interpolation and extrapolation respectively. Figures 17 and 18 show the RNN results on Plant4 case for interpolation and extrapolation respectively. Also, it is important to note that system re-identification takes tenth the time reported in Table 5. This is because when reupdating the model, the model parameters of the last identified model are used to initialize the parameters of the new model before training for only 500 epochs as opposed to using 5,000 epochs when training the model for the first time.

Remark – The results obtained from the three system identification methods show that subspace identification is the least prone to overfitting, while RNN is the most prone to overfitting. This is because of the structure of the model (linear versus nonlinear), the number of parameters, and the training algorithm. RNN and NARX neural network can outperform subspace identification when the testing data are similar to the majority of the training data, but their performance starts deteriorating greatly when testing data behavior discrepancy increases compared to training data. Reducing the number of parameters, increasing the number of training points that behave similar to the testing data, and decreasing the number of training points that behave differently compared to testing data, might help greatly in improving the neural network model's capability to extrapolate. Also, using data compression on the inputs before RNN model can improve model performance.

Table 7: WMSE for NARX neural network method with no model update versus the other two algorithms

Case Study	Data	No Update	Algorithm 1	Algorithm 2	Algorithm 2 re-identification trigger
Simulation 9 Inputs	All	3.752	0.137	0.064	59
	Interpolation	0.017	0.005	0.004	14
	Extrapolation	7.798	0.279	0.130	45
Simulation 4 Inputs	All	0.958	0.227	0.080	48
	Interpolation	0.018	0.019	0.012	14
	Extrapolation	1.976	0.453	0.153	34
Plant 4 Inputs	All	1.499	0.219	0.088	40
	Interpolation	0.079	0.084	0.056	15
	Extrapolation	3.037	0.365	0.123	25

Table 8: WMSE for RNN method with no model update versus the other two algorithms

Case Study	Data	No Update	Algorithm 1	Algorithm 2	Algorithm 2 re-identification trigger
Simulation 9 Inputs	All	4.948	0.575	0.272	62
	Interpolation	0.040	0.012	0.006	18
	Extrapolation	10.265	1.186	0.561	44
Simulation 4 Inputs	All	2.419	0.359	0.112	48
	Interpolation	0.037	0.017	0.009	13
	Extrapolation	4.999	0.730	0.224	35
Plant 4 Inputs	All	1.541	0.222	0.141	43
	Interpolation	0.088	0.060	0.053	15
	Extrapolation	3.115	0.398	0.236	28

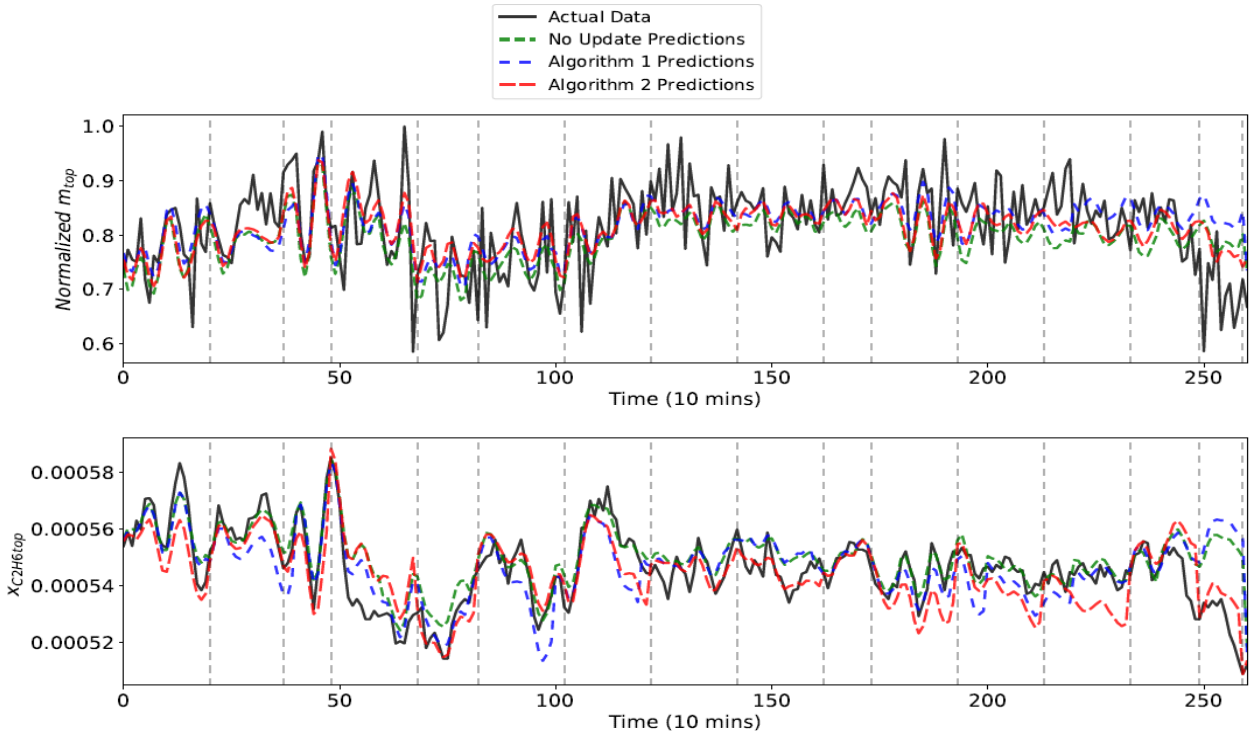

Figure 15. Interpolation data prediction with NARX neural network for the real plant data with no update and the two algorithms

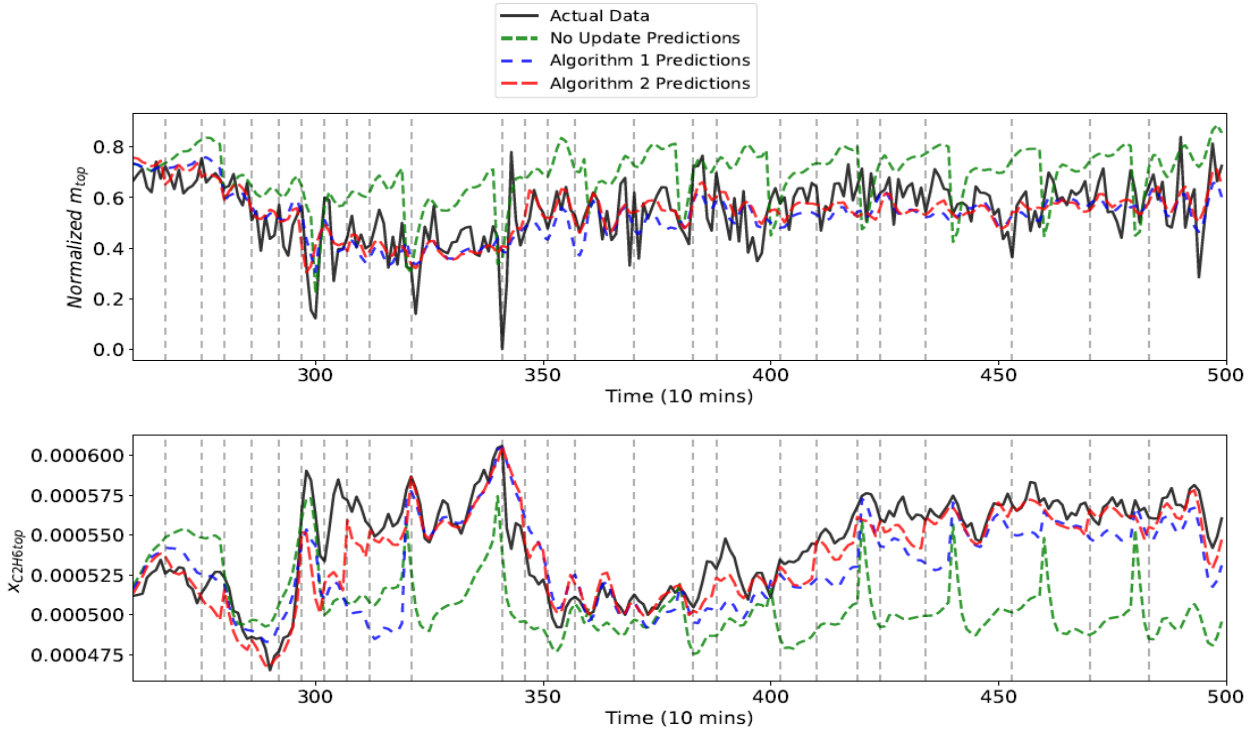

Figure 16. Extrapolation data prediction with NARX neural network for the real plant data with no update and the two algorithms

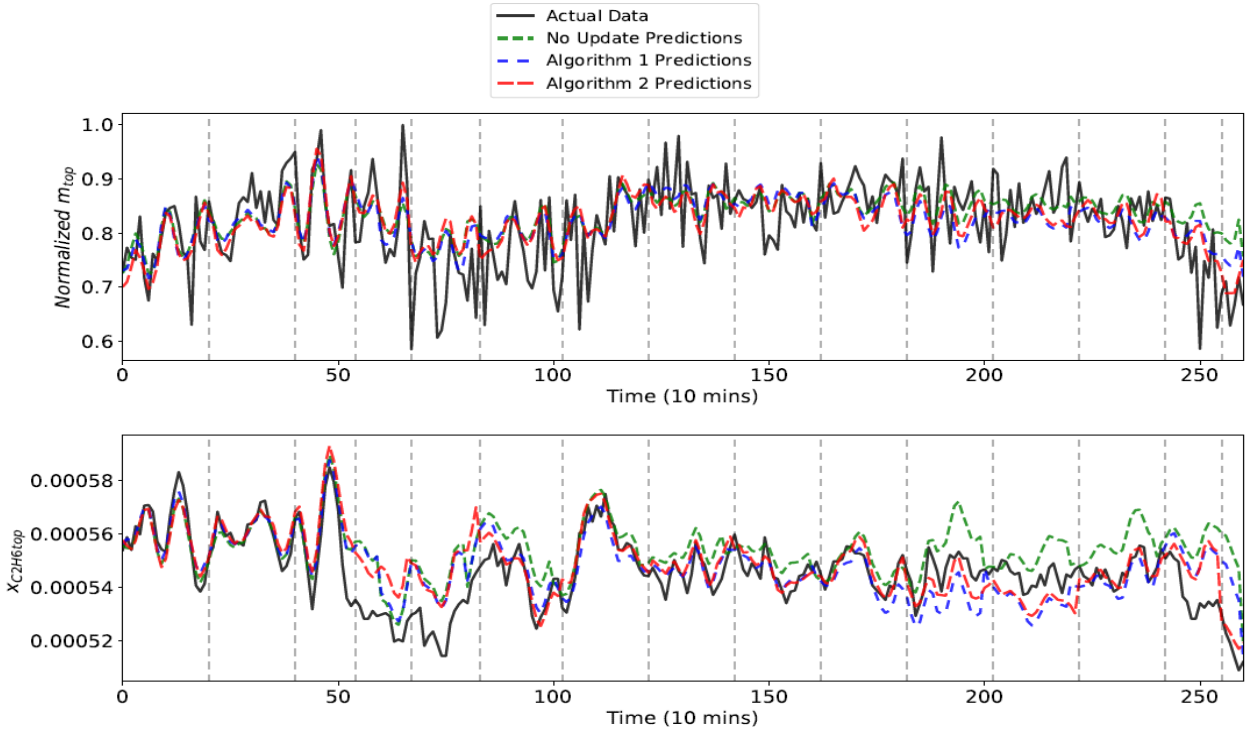

Figure 17. Interpolation data prediction with RNN for the real plant data with no update and the two algorithms

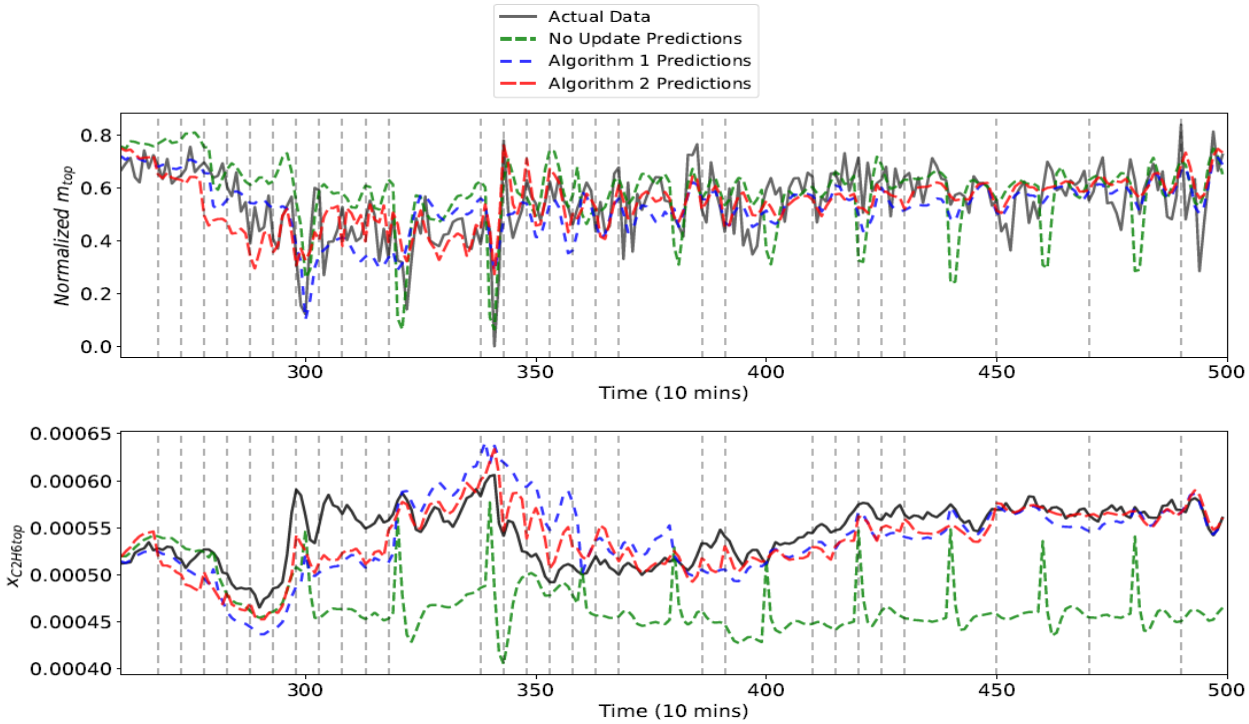

Figure 18. Extrapolation data prediction with RNN for the real plant data with no update and the two algorithms

Figure 19 demonstrates the improvement achieved from our two system adaption algorithms compared to no model update for the three system identification methods on 20 steps ahead prediction. The figure shows that frequent model updates for our three system identification methods can improve the prediction of the top ethane composition.

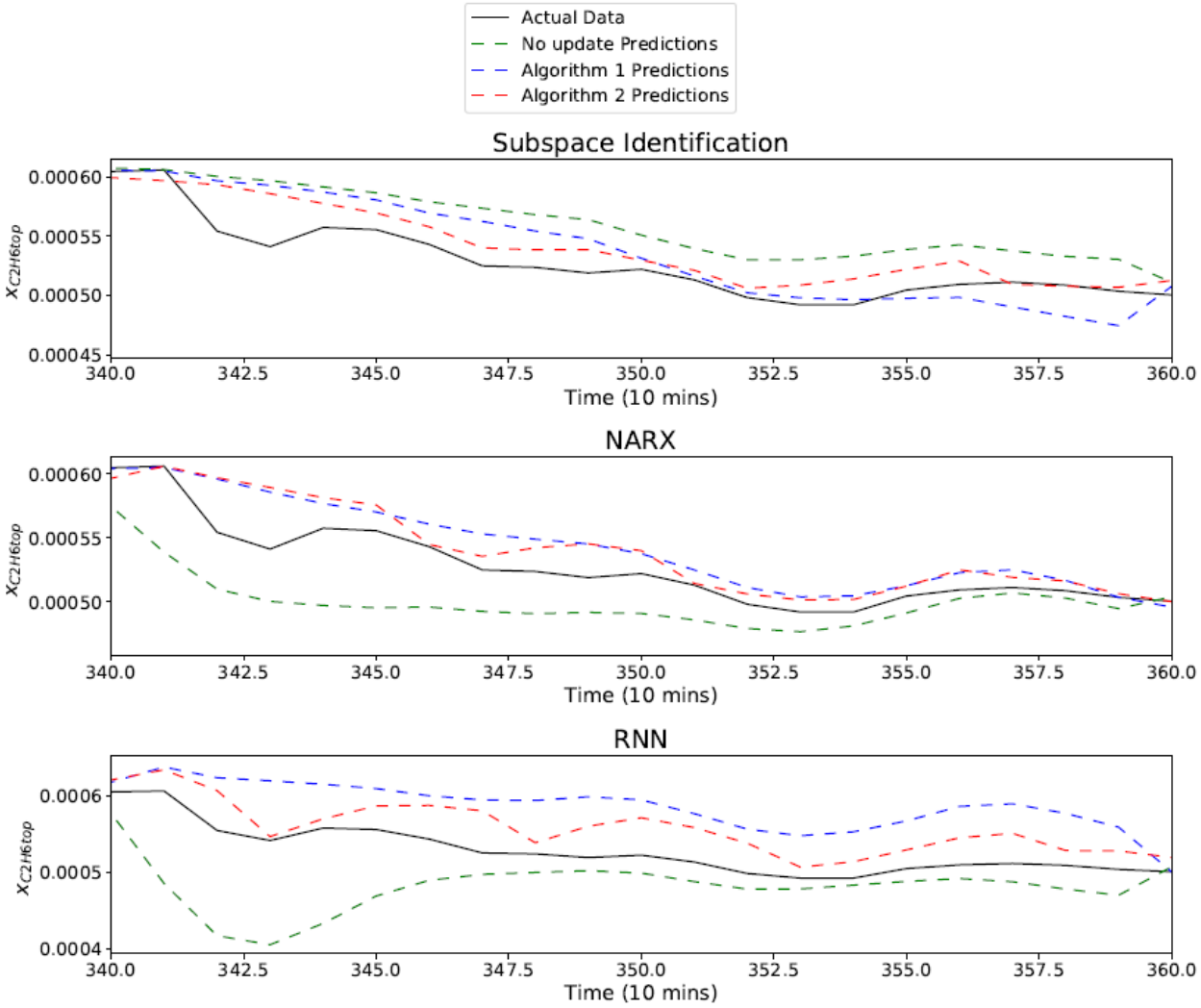

Figure 19. Compare true and predicted ethane compositions for the three different model update strategies using the different system identification methods.

4.8. Conclusions

In this paper, three different system identification methods (subspace identification, NARX neural network, and RNN) have been developed for modeling industrial ethylene splitter. The system identification methods have been adapted to fit simulated and real plant data to compare and show their capability at developing dynamic model for both data. The results show that linear subspace identification method can provide the best predictions in general for the dynamic systems due to their capability of extrapolating, while neural networks methods are highly prone to overfitting and faces issues in generalizing to new process behavior. Also, subspace identification requires greatly less time to optimize the hyperparameter compared to neural

network methods due to their uniqueness and the time required for training. Finally, two adaptive strategies have been proposed for the three different system identification methods to allow more recent data to be incorporated into the training approaches. The two adaptive strategies show great improvement in the model prediction capabilities for all the three system identification methods. Also, for the case where only a small amount of training data is available, single model architecture for subspace identification method combined with the adaptation strategy can perform well at both interpolation and extrapolation data. On the other hand, neural network methods can benefit from having large architecture (number of cells in a layer) when interpolating but this causes the results to be poor when extrapolating. This suggests that different neural network architectures may be needed for interpolation data and extrapolation data if it is possible to detect whether the current and future operation is already represented in the training data or not.

Acknowledgements

This work was supported by NOVA Chemicals Corporation, Ontario Graduate Students, and McMaster Advanced Control Consortium is greatly acknowledged

Literature Cited

[1] Salerno, D., Arellano-Garcia, H., & Wozny, G., 2011. Ethylene separation by feed-splitting from light gases. *Energy*, *36*(7), 4518-4523. https://doi.org/10.1016/j.energy.2011.03.064

[2] Zhao, H., Guiver, J., Neelakantan, R., & Biegler, L. T., 2001. A nonlinear industrial model predictive controller using integrated PLS and neural net state-space model. *Control Engineering Practice*, *9*(2), 125-133. https://doi.org/10.1016/S1474-6670(17)57130-X

[3] Kolmetz, K., Sari, R. M., 2014. Kolmetz Handbook of Process Equipment Design. *Malaysia: KLM Technology Group. Crude unit desalter system. Section Three–Refinery systems*, 1-34. Research Gate

[4] Borralho, F. J. O., 2013. Detailed Modelling and Optimisation of an Ethylene Plant (Doctoral dissertation, Master, s Thesis, Instituto Superior Técnico). https://fenix.tecnico.ulisboa.pt/downloadFile/395145831373/dissertacao.pdf

[5] Yan, M., 2000. Simulation and optimization of an ethylene plant (Doctoral dissertation, Texas Tech University). https://ttu-ir.tdl.org/handle/2346/16119

[6] Wang, Z. Y., Zhang, J., Xu, Q., & Ho, T. C., 2014. Dynamic simulation for optimal operation of distillation column startups in an ethylene plan. In *Proceedings of the 26th Ethylene Producers' Conference* (p. 56d). Research Gate

[7] Choe, Y. S., & Luyben, W. L., 1987. Rigorous dynamic models of distillation columns. *Industrial & engineering chemistry research*, *26*(10), 2158-2161. https://doi.org/10.1021/ie00070a038

[8] Eliceche, A. M., Petracci, N. C., Hoch, P., Brignole, E. A., 1995. Optimal operation of an ethylene plant at variable feed conditions. *Computers & chemical engineering*, *19*, 223-228. https://doi.org/10.1016/0098-1354(95)87040-7

[9] Friedman, Y. Z. (1999). Advanced control of ethylene plants: what works, what doesn't, and why. *Ethylene Producers' Committee*. https://petrocontrol.com/wp-content/uploads/2020/07/1999_Ethylene_APC.pdf

[10] Huang, B., Ding, S. X., Qin, S. J., 2005. Closed-loop subspace identification: an orthogonal projection approach. *Journal of process control*, *15*(1), 53-66. https://doi.org/10.1016/S1474-6670(17)31824-4

[11] Meidanshahi, V., Corbett, B., Adams II, T. A., Mhaskar, P., 2017. Subspace model identification and model predictive control based cost analysis of a semicontinuous distillation process. *Computers & Chemical Engineering*, *103*, 39-57. https://doi.org/10.1016/j.compchemeng.2017.03.011

[12] Castaño, J. E., Patiño, J. A., & Espinosa, J. J., 2011. Model identification for control of a distillation column. In *IX Latin American Robotics Symposium and IEEE Colombian Conference on Automatic Control, 2011 IEEE* (pp. 1-5). IEEE. https://doi.org/10.1109/LARC.2011.6086832

[13] Meenakshi, S., Almusthaliba, A., Vijayageetha, V., 2013. MIMO Identification and Controller design for Distillation Column. *International journal of innovative research in electrical, electronics, instrumentation and control engineering*, *1*(2), 44-48. https://ijireeice.com/wp-content/uploads/2013/03/4-Meenakshi-MIMO-Identification-and-Controller.pdf

[14] Kanthasamy, R., Anwaruddin, H., Sinnadurai, S. K., 2014. A new approach to the identification of distillation column based on hammerstein model. *Modelling and Simulation in Engineering*, *2014*. https://doi.org/10.1155/2014/813757

[15] Norquay, S. J., Palazoglu, A., Romagnoli, J. A., 1999. Application of Wiener model predictive control (WMPC) to an industrial C2-splitter. *Journal of Process Control*, *9*(6), 461-473. https://doi.org/10.1016/S0959-1524(99)00036-0

[16] Brizuela, E., Uria, M., Lamanna, R., 1996. Predictive Control of a Multi-Component Distillation Column Based on Neural Networks E. Brizuela. In *Neural Networks for Identification, Control, and Robotics, International Workshop* (pp. 0270-0270). https://doi.org/10.1109/NICRSP.1996.542769

[17] Singh, V., Gupta, I., Gupta, H. O., 2005. ANN based estimator for distillation—inferential control. *Chemical Engineering and Processing: Process Intensification*, *44*(7), 785-795. https://doi.org/10.1016/j.cep.2004.08.010

[18] Ochoa-Estopier, L. M., Jobson, M., Smith, R., 2013. Operational optimization of crude oil distillation systems using artificial neural networks. *Computers & chemical engineering*, *59*, 178-185. https://doi.org/10.1016/j.compchemeng.2013.05.030

[19] MacMurray, J., Himmelblau, D., 1993. Identification of a packed distillation column for control via artificial neural networks. In *1993 American Control Conference* (pp. 1455-1459). IEEE. https://doi.org/10.23919/ACC.1993.4793112

[20] Pan, Y., Sung, S. W., Lee, J. H., 2001. Data-based construction of feedback-corrected nonlinear prediction model using feedback neural networks. *Control Engineering Practice*, *9*(8), 859-867. https://doi.org/10.1016/S0967-0661(01)00050-8

[21] Jaleel, A., Aparna, K., 2015. Identification of ethane-ethylene distillation column using neural network and ANFIS. In *2015 Fifth International Conference on Advances in Computing and Communications (ICACC)* (pp. 358-361). IEEE. https://doi.org/10.1109/ICACC.2015.17

[22] Jaleel, E. A., Aparna, K., 2019. Identification of realistic distillation column using hybrid particle swarm optimization and NARX based artificial neural network. *Evolving Systems*, *10*(2), 149-166. https://doi.org/10.1007/s12530-018-9220-5

[23] Savkovic-Stevanovic, J., 1996. Neural net controller by inverse modeling for a distillation plant. *Computers & chemical engineering*, *20*, S925-S930. https://doi.org/10.1016/0098-1354(96)00162-7

[24] Singh, A. K., Tyagi, B., Kumar, V., 2013. Application of feed forward and recurrent neural network topologies for the modeling and identification of binary distillation column. *IETE Journal of Research*, *59*(2), 167-175. https://doi.org/10.4103/0377-2063.113038

[25] Abdullah, Z., Ahmad, Z., Aziz, N., 2009. MIMO neural network model for pilot plant distillation column. In *Computer Aided Chemical Engineering* (Vol. 27, pp. 531-536). Elsevier. https://doi.org/10.1016/S1570-7946(09)70309-8

[26] Sadeghassadi, M., Macnab, C. J., Gopaluni, B., Westwick, D., 2018. Application of neural networks for optimal-setpoint design and MPC control in biological wastewater treatment. *Computers & Chemical Engineering*, *115*, 150-160. https://doi.org/10.1016/j.compchemeng.2018.04.007

[27] Zhang, Z., Wu, Z., Rincon, D., Christofides, P. D., 2019. Real-time optimization and control of nonlinear processes using machine learning. *Mathematics*, *7*(10), 890. https://doi.org/10.3390/math7100890

[28] Alanqar, A., Durand, H., & Christofides, P. D., 2017. Error-triggered on-line model identification for model-based feedback control. *AIChE Journal*, *63*(3), 949-966.

https://doi.org/10.1002/aic.15430

[29] Wu, Z., Rincon, D., & Christofides, P. D., 2019. Real-time adaptive machine-learning-based predictive control of nonlinear processes. *Industrial & Engineering Chemistry Research*, *59*(6), 2275-2290.

https://doi.org/10.1021/acs.iecr.9b03055

[30] Abedi, A. A., 2007. Economical analysis of a new gas to ethylene technology (Doctoral dissertation, Texas A&M University). https://core.ac.uk/download/pdf/4272899.pdf

[31] Leegwater, H., 1992. Industrial experience with double quality control. In *Practical distillation control* (pp. 331-350). Springer, New York, NY. https://doi.org/10.1007/978-1-4757-0277-4_16

[32] Taieb, S. B., Hyndman, R. J., 2012. Recursive and direct multi-step forecasting: the best of both worlds (Vol. 19). Department of Econometrics and Business Statistics, Monash Univ.

[33] Taieb, S. B., & Atiya, A. F. (2015). A bias and variance analysis for multistep-ahead time series forecasting. *IEEE transactions on neural networks and learning systems*, *27*(1), 62-76.

https://doi.org/10.1109/TNNLS.2015.2411629

[34] Moonen, M., De Moor, B., Vandenberghe, L., Vandewalle, J., 1989. On-and off-line identification of linear state-space models. *International Journal of Control*, *49*(1), 219-232. https://www.tandfonline.com/doi/abs/10.1080/00207178908559631

[35] Corbett, B., Mhaskar, P., 2016. Subspace identification for data-driven modeling and quality control of batch processes. *AIChE Journal*, *62*(5), 1581-1601. https://doi.org/10.1002/aic.15155

[36] Garg, A., Mhaskar, P., 2017. Subspace identification-based modeling and control of batch particulate processes. *Industrial & Engineering Chemistry Research*, *56*(26), 7491-7502. https://doi.org/10.1021/acs.iecr.7b00682

[37] Phan, M. Q., Longman, R. W., 1970. Relationship between state-space and input-output models via observer Markov parameters. *WIT Transactions on The Built Environment*, *22*. https://www.witpress.com/Secure/elibrary/papers/DCSS96/DCSS96012FU.pdf

[38] Sum, J. P. F., Kan, W. K., Young, G. H., 1999. A Note on the Equivalence of NARX and RNN. *Neural computing & applications*, *8*(1), 33-39. https://doi.org/10.1007/s005210050005

[39] Mohajerin, N., 2017. Modeling dynamic systems for multi-step prediction with recurrent neural networks. https://uwspace.uwaterloo.ca/bitstream/handle/10012/12766/Mohajerin_Nima.pdf?sequence=3&isAllowed=y

[40] Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. https://doi.org/10.1109/9780470544037.ch14

[41] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. *Neural computation*, *9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[42] Yan, S., 2015. Understanding LSTM networks. *Online*. *Google Scholar*

Chapter 5: Flooding and Offset-Free Nonlinear Model Predictive Control of a High-Purity Industrial Ethylene Splitter Using a Hybrid Model

The contents of this chapter have been submitted to the *Computer & Chemical Engineering Journal*.

Abstract

This work aims to achieve flooding free control for an industrial ethylene splitter. The tower operates at high capacity and faces flooding issues due to feed disturbances. This work develops a replacement of the current control strategy with an offset-free Nonlinear Model Predictive Control (OF-NMPC) to improve the control and avoid flooding. The key idea is to develop an OF-NMPC that uses a hybrid model comprising of a Nonlinear Autoregressive Network (NARX) for predicting dynamics, and a first principles steady state model to capture flooding. The steady state model relates tower internal flow to the manipulated variables and is incorporated as a constraint within the OF-NMPC. The effectiveness of the proposed OF-NMPC is demonstrated on an Aspen simulation model which is shown to capture the key traits of the real plant. The simulation results demonstrate the ability of the proposed hybrid model based OF-NMPC design to achieve flooding free control.

Keywords:

Control of industrial separation unit; Hybrid model; Nonlinear model predictive control; Distillation column flooding control

5.1. Introduction

Distillation columns are commonly used in chemical plants to separate various chemical mixtures. Some distillation columns are required to have high and constant level of products' purities (Fuentes and Luyben, 1983), while keeping high level of throughputs. Frequently, industries over-reflux their distillation columns to meet product purity specifications. As a result of the over-reflux, industries use 30% to 50% more energy than necessary to meet the product specification (Dr. James B. Riggs, 2000), which may reduce their production capacity limit. Effective control of distillation columns can avoid the need to over-reflux and reduce energy usage. Another common problem facing distillation columns is flooding, which occurs due to excessive accumulation of liquid inside the column. Flooding can decrease separation efficiency and cause products' purities to go off specifications (Emerson Process Management, 2009). This work considers the ethylene splitter (C2 splitter) located at Joffre site in Alberta. A C2 splitter is used to separate ethylene from ethane during the final step in the production of ethylene. The ethylene product is drawn from the top of the tower to be sold in the market as a final product, while the ethane product is drawn from the bottom of the tower to be recycled to upstream processes. An advanced control strategy is critical to meeting the required ethylene product purity, maximizing energy usage efficiency, and considering hydraulic processing capacity of the tower. The complexity of operation and control of a C2 splitter stem from many reasons: (1) high interaction between top and bottom products' purities, (2) small difference in relative volatilities of ethane and ethylene, (3) slow and nonlinear dynamics of the column, and (4) disturbances of feed flow, composition, and enthalpy, which can strongly affect products' purities. Additionally, the C2 splitter studied operates with limited capacity and high feed flow, which can cause flooding in the tower. In summary, the C2 splitter at the Joffre site faces challenges in maintaining constant products' purities and faces flooding issues.

In the process industry, Proportional-Integral-Differential (PID) controllers are widely used due to their simplicity and effectiveness (Balaji and Rajaji, 2013) in solving most single variable control tasks with small order dynamics and small or no deadtime (Valancia-Palomo, G.; Rossiter, 2007). However, PID controllers may be ineffective in controlling complex processes with multiple inputs and outputs, and processes that exhibit large deadtime. PID controllers directly compare the measured output value with a reference output value and use the difference between these values (the error) to adjust manipulated variables in order to minimize the error

between the measured output value and its setpoint (Efheij et al., 2019). The parameters of the PID controllers used in the calculation of the manipulated variables must be tuned to provide an effective control of the output, which is not an easy task.

On the other hand, a Model Predictive Control (MPC) utilizes a dynamic model of the process, which can be identified using system identification, to predict process output variables over the prediction horizon based on the input variables (manipulated variables) (Efheij et al., 2019). The MPC's goal is to calculate a control action (i.e. manipulate variables moves) that minimizes the error between the predicted output variables and their setpoints over a finite horizon. Therefore, the effectiveness of the MPC relies heavily on the availability of a reasonably accurate process model (Bequette, 2007). MPC algorithms based on linear dynamic model may be ineffective in controlling nonlinear processes that operate over a wide range of operational conditions. Therefore, MPC algorithms based on a nonlinear dynamic model may be essential to effectively controlling such processes (Tian et al., 2014). Developing a nonlinear model that precisely matches the plant dynamics is unachievable in most cases, and a plant-model mismatch is inevitable (Badwe et al., 2010). Furthermore, unmeasured process disturbances can widen the mismatch between the plant and the model, and subsequently degrade the MPC's performance (Tian et al., 2014). Yousefi et al. (Yousefi et al., 2015) studied the effect of model-plant mismatch on the performance of control systems. Therefore, various methods have been proposed in literature to address the offset problem caused by plant-model mismatch or unmeasured disturbances (Faanes and Skogestad, 2005) (González et al., 2008) (Maeder and Morari, 2010) (Muske and Badgwell, 2002) (Huang et al., 2010). One method that has been widely used in industrial MPC implementation (Bequette, 2007) is shifting the set points to compensate for the plant-model mismatch. Qin and Badgwell (Qin and Badgwell, 2003) provided an overview of industrial MPC technology. Also, Forbes et al. (Forbes et al., 2015) examined established and emerging trends in the industrial applications of MPC.

Several works have focused specifically on the control of distillation columns. Taqvi et al. (Ammar Taqvi et al., 2019) addressed the various disturbances and operational difficulties of distillation columns. Meenakshi et al. (Meenakshi et al., 2013) used an MPC-based subspace identification method to control an ethylene splitter. The work of Riggs (Dr. James B. Riggs, 2000) determined that a MPC works better than Proportional-Integral (PI) controllers when one product purity is more important than the other, while PI controllers work better when both

product purities are equally important. Chen et al. (Chen et al., 2010) developed a nonlinear MPC strategy for distillation based on the assumption of full-state feedback and demonstrated the performance advantages of their nonlinear MPC compared to linear MPC. Ramesh et al. (Ramesh et al., 2009) developed a nonlinear model-based control scheme using Nonlinear Autoregressive Network with Exogenous Inputs (NARX) model to control a distillation column. Foss and Cong (Foss and Cong, 1999) developed a nonlinear model predictive control based on a multi-model structure through first principles modeling and estimated the model parameters using data generated by a rigorous model.

Compared to other distillation columns, high-purity distillation columns are hard to model due to their strong nonlinearity and transition dynamics between different operation points. Xiong et al. (Xiong et al., 2014) introduced multiple model-based approaches under the framework of the expectation maximization algorithm to model a high purity distillation column. Sirniwas et al. (Ravi Sriniwas et al., 1995) used nonlinear autoregressive models with exogenous inputs (NARX) to identify a model for the high purity distillation column and used nonlinear MPC to control the process. Bachans et al. (Bachnas et al., 2014) reviewed different data-driven linear parameter-varying modeling approach for a high purity distillation column.

In previous work, we studied different system identification methods to use within a MPC (Jalanko et al., 2021) to improve process control of the ethylene splitter. Also, we developed a simulation model in Aspen dynamics to use as a test bed for control purposes and demonstrated that the simulator behaves similar to the real distillation column. The previous results, however, did not consider the tower flooding. The flooding in the studied tower is caused by excessive accumulation of liquid at the bottom section of the tower. Flooding can be recognized in distillation columns by a sharp increase in differential pressure, a loss of bottom product flow, a degradation in the separation of products, or a change in the column temperature profile (Kister, 1990). The use of data-driven models to predict the previously mentioned measurements to identify tower flooding is a potential route towards flooding control. However, data-driven models may suffer from poor predictions under flooding conditions due to the limited amount of flooding operation data available to be used during model training. Therefore, this work aims to develop a hybrid model that uses first principles model to avoid tower flooding and a data-driven model to control the products' purities.

Several works have also considered the topic of hybrid modeling. Patel et al. (Patel et al., 2020) developed a hybrid model that incorporates a first principles model in the identification of subspace-based state space model. Ghosh et.al (Ghosh et al., 2019) developed a hybrid modeling framework that uses data-drive modeling approach to predict the error between the output generated by the first-principles model and the plant output. The following works have also considered hybrid modeling approaches for different processes (Su et al., 1992) (Zhang et al., 2019) (Tsen et al., 1996) (Psichogios and Ungar, 1992) (Hassanpour et al., 2020) (Wu et al., 2020). These works have either integrated first-principles knowledge directly in their data-driven models or have used data-driven models to model the error between their first principles model outputs and the process outputs. The goal of achieving flooding free control, however, is not readily achievable using these approaches. In summary, while control designs exist for simple distillation columns, the problem of control while handling flooding and accounting for nonlinear and complex behavior (all of which are present in high purity distillation columns such as the ethylene splitter) have not been addressed.

Motivated by these considerations, in the present manuscript, we develop an offset-free Nonlinear Model Predictive Control (OF-NMPC) based on NARX prediction model to improve the control of our studied distillation column and avoid tower flooding. First, data that represents flooding behavior using the Aspen simulator is generated. The generated data from the simulator is compared to the plant measurements to establish that the simulator captures the process dynamics. Then, we generate rich data by exciting the system (applying RGS signal on the manipulated variables) to develop a predictive model to the simulator. This data is first used to identify a linear model using subspace identification. However, our experiments showed that the linear subspace model is not sufficiently accurate for this application, hence a nonlinear model is developed using NARX to ensure accurate predictions. An OF-NMPC based NARX model is implemented to control the products' purities and its performance is compared to the PI controller's result. OF-NMPC performed quite well in the region away from the flowing conditions. In order to eliminate occasional excursions in the flooding region, the internal liquid flow rate at the tray where flooding occurs is calculated via a first principles model as a function of reflux flow and reboiler duty. This linear model is added as a constraint in our MPC to eliminate the internal increases to the point where the tower flooding occurs. Note that this approach is different from existing hybrid modeling approaches in the sense that the first-

principles model and the data-driven model are identified independently, and both are used models within a MPC framework. The remainder of the paper is organized as follows: Section 2 provides an overview of the preliminaries relevant to this work. Specifically, Section 2.1 presents a brief description of the C2 splitter. Section 2.2 presents the developed Aspen dynamics-based simulation model and compares the simulation results against the plant operation under flooding scenario. Section 3 introduces the hybrid modeling scheme and the hybrid model-based OF-NMPC. Specifically, Section 3.1 provides the background of system identification based on NARX. Section 3.2 presents the developed steady-state model used within our OF-MPC to avoid flooding. Section 3.3 presents the hybrid model-based OF-NMPC approach used in this work. Section 4 present the system identification and OF-NMPC results with and without flooding constraint and compares it to the PI controller strategy results. Finally, Section 5 presents concluding remarks.

5.2. Preliminaries

This section describes the C2 splitter process with its current control strategy. Next, it introduces the Aspen dynamic simulation model, then presents results generated from the Aspen dynamic simulation and compares it to the plant measurements.

5.2.1. C2 Splitter Process

This section describes the C2 splitter at Joffre site, which is the final step in the production of ethylene. The reader can refer to the work of Abedi (Abedi, 2007) for a detail description of the ethylene production plant, its main sections, and its chemistry. A schematic of the C2 splitter at the Joffre site without the end composition controllers is presented in Figure 1. The towers currently have the following PI controllers:

- Temperature controller at tray 85, which measures the temperature at that tray and adjusts the reboiler duty to maintain the temperature at its setpoint.
- Ethane composition controller at tray 24, which measures the ethane composition at that tray and adjusts the reflux flow to maintain the composition at its setpoint.
- Tower sump level controller at the bottom of the tower, which adjusts the bottom product flow rate to maintain the level of liquid at the bottom tower at its setpoint.
- The reflux drum level controller at the distillate, which adjusts the distillate flow to maintain the reflux drum level at its setpoint.

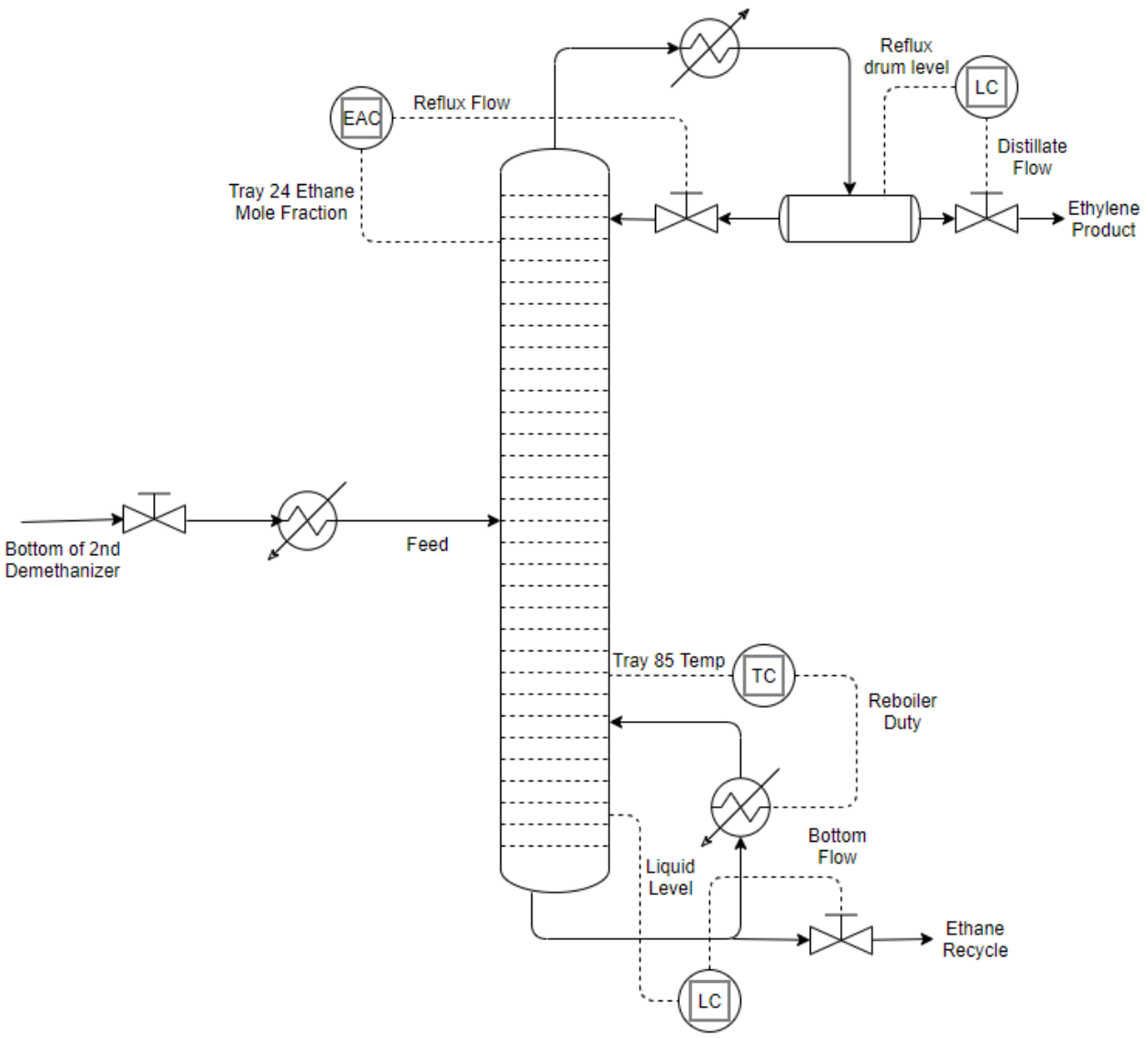

Figure 1. C2 splitter current control strategy

5.2.2. Modeling the Ethylene Splitter

As noted earlier, the first step was to create a dynamic model that captures the complex behavior of the plant to be used as a test bed to demonstrate the benefit of the proposed control strategy and show that it can avoid tower flooding. The ethylene splitter is simulated in Aspen Plus V10 as an isolated system and imported into Aspen Dynamics. Since the ethylene splitter is simulated as an isolated system, as presented in Figure 2, the dynamic model needed some adjustments to replicate the industrial process. The reader can refer to the previous work (Jalanko et al., 2021) for more details on modeling the ethylene splitter and for a detailed discussion on the reasons behind plant-simulation mismatch. Due to the lack of gas chromatography (GC) on the ethylene

splitter feed, the feed composition is not measured/known. Therefore, the feed composition was inferred using upstream process information via mass, mole, and energy balances. This can result in a mismatch between the simulation and the plant results. Another factor that contributed to the plant-simulation mismatch is that simulator assumes that the tower's feed only has ethane and ethylene, while the real process is known to have a small amount of methane, propylene, and propane. Table 1 gives a summary of the C2 splitter input variables (manipulated variables and disturbances) and output variables used in this work.

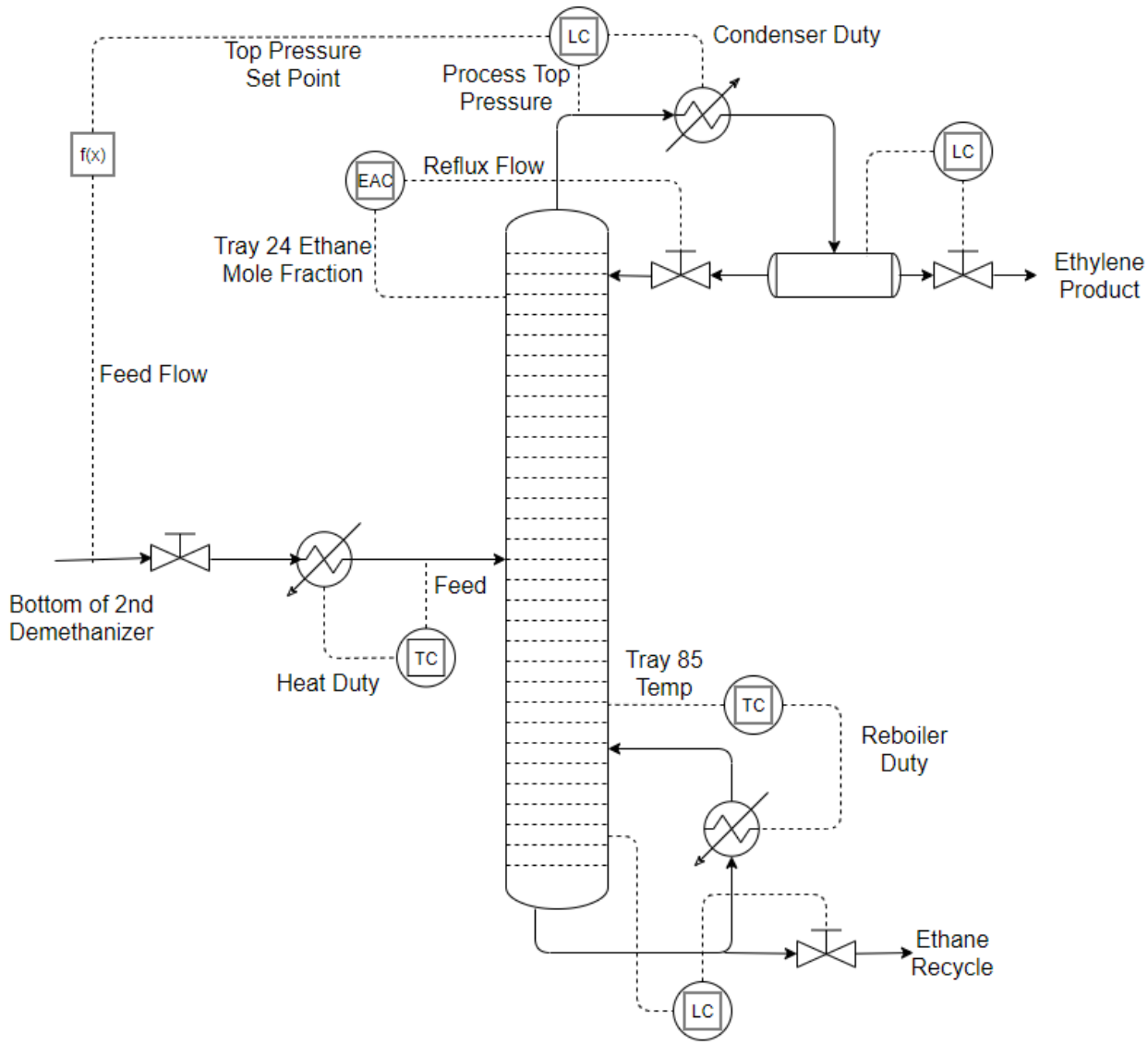

Figure 2. detailed diagram for the Aspen dynamics model

The simulation data was generated by providing the bottom of the 2nd demethanizer flow rate, temperature, and pressure provided by the plant measurements, along with the composition inferred from upstream processes. In the previous work (Jalanko et al., 2021), the tower feed temperature measurements of the plant were used as the set point for the feed temperature controller which adjusts the duty of the heat exchanger. In this work, the heat duty was fixed since it provided a better match between the simulation and the plant data results. Tray 85 temperature and tray 24 ethane composition plant setpoints were fixed based on the real plant setpoints. The plant-model mismatch was evaluated by generating plots for all the inputs and outputs of the real plant and the model under flooding operation with a sampling rate of 10 minutes. Figures 3 and 4 compare the plant inputs and outputs measurements, respectively, to the simulation results under normal and flooding conditions. It is important to note that for confidentiality purposes, the feed flow, reflux flow, and reboiler duty values were standardized. While the comparison between the plant and simulation model was presented in previous work (Jalanko et al., 2021) under free flooding operations, it was necessary to evaluate the effectiveness of the Aspen simulator to behave similarly to the real plant under flooding operations.

Table 1. C2 splitter input and output variables

Category	Variable	Description	Units
Disturbance	$\dot{m}_F$	Feed flow from upstream process	kg/hr
	T_F	Feed temperature	°C
	$x_{C_2H_4\ F}$	Feed mole fraction of ethylene	%
	P_F	Feed pressure	bar
	v_F	Feed Vapor Fraction	%
Manipulated variables	$\dot{m}_R$	Reflux flow	kg/hr
	x	Reboiler duty	Gj/hr
Outputs	T_{t24}	Temperature at tray 24	°C
	T_{t37}	Temperature at tray 37	°C
	T_{t51}	Temperature at tray 51	°C
	T_{t56}	Temperature at tray 56	°C
	T_{t85}	Temperature at tray 85	°C
	P_{t_1}	Pressure at tray 1	bar
	ΔP_{top}	Top tower differential pressure	bar
	ΔP_{bot}	Bottom tower differential pressure	bar
	$x_{C_2H_4\ t97}$	Mole fraction of ethylene at tray 97	%
	$x_{C_2H_6\ t24}$	Mole fraction of ethane at tray 24	%
	$\dot{m}_{top}$	Flow rate of top product (ethylene)	kg/hr
	T_{top}	Temperature of top product (ethylene)	°C
	$x_{C_2H_6\ top}$	Mole fraction of ethane of the top product (ethylene)	%
	$\dot{m}_{bot}$	Flow rate of bottom product (ethane)	kg/hr
	T_{bot}	Temperature of bottom product (ethane)	°C
	$x_{C_2H_4\ bot}$	Mole fraction of ethylene of the bottom product	%
	$\dot{m}_{L\ t102}$	Liquid flow at tray 102	kg/hr
	$\hat{\dot{m}}_{L\ t102}$	Predicted Liquid flow at tray 102	kg/hr

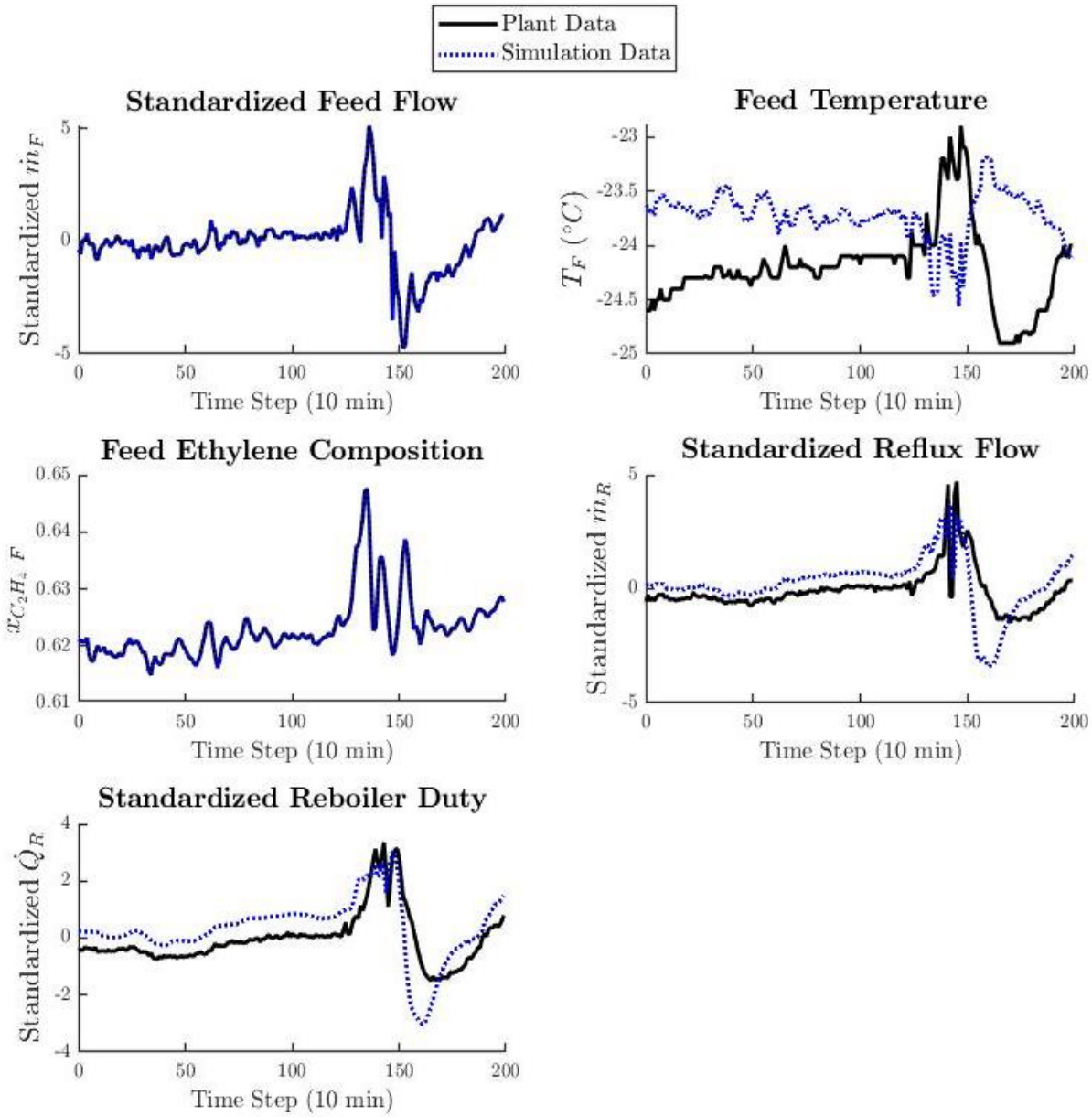

Figure 3. plant measurement inputs versus simulation inputs

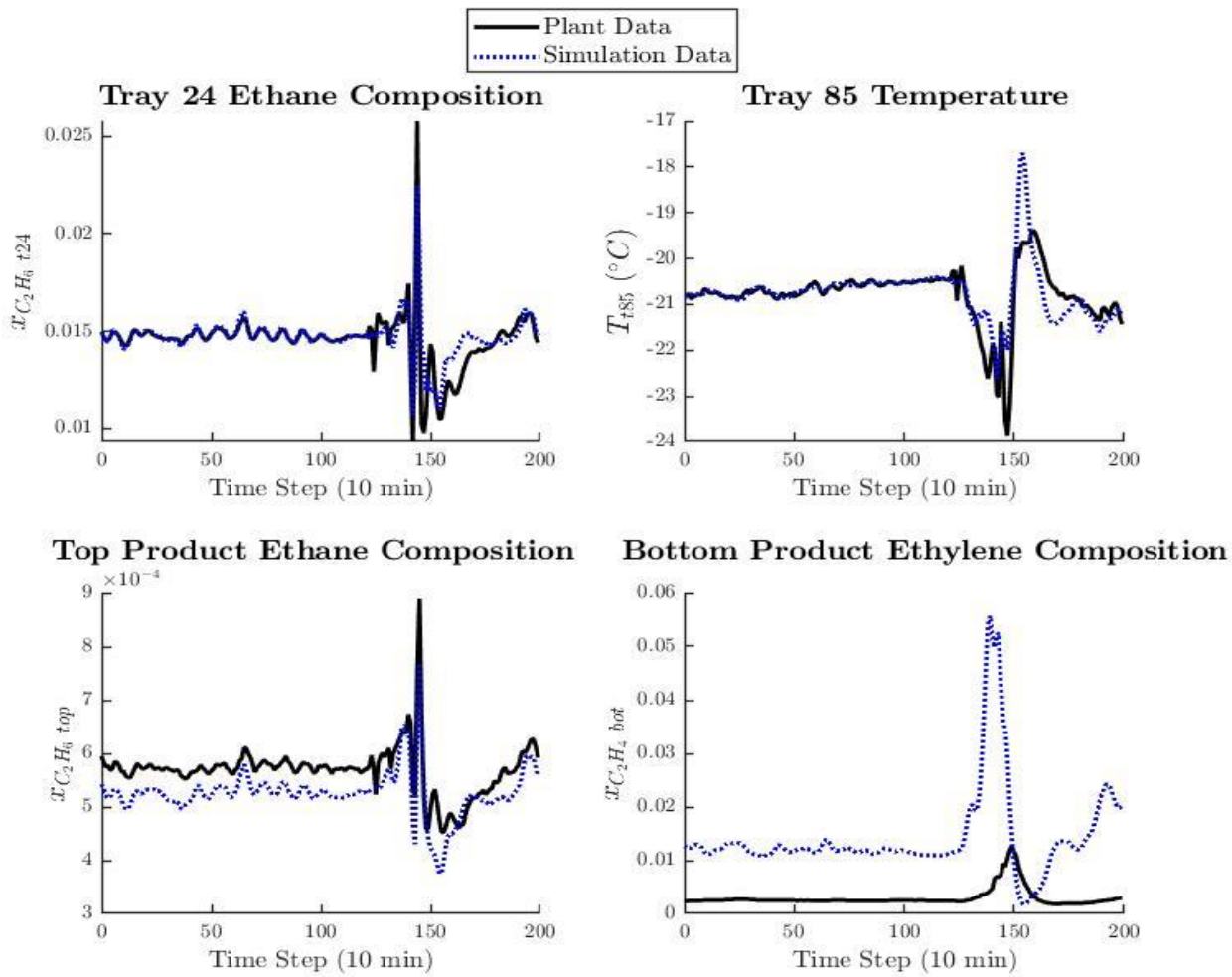

Figure 4. plant measurement outputs versus simulation outputs

The feed flow and composition of the simulation presented in Figure 3 perfectly match the plant measurements since they were specified based on the plant measurements. The feed temperature of the simulation is completely different from the plant measurements and that is due to fixing the heat duty of the heat exchanger. The heat duty of the heat exchanger was fixed since it provided closer simulation outputs results to the plant. The discrepancy between the simulation and plant feed temperature is mainly due to the fact that the simulation only considers ethylene and ethane in the feed and ignores the small amount of other compositions. Since the simulation uses controllers setpoint from the real plant to generate closed-loop data, the behaviour of the manipulated variables can be used to measure the plant-model mismatch. The standardized reflux flow and reboiler duty show that the simulation behaves similarly to the real plant with some bias. The bias is shown clearly when plotting the nominal values of the reflux flow and reboiler as opposed to the standardized values, which is not provided due to confidential reasons.

The reboiler duty plant-model is less reliable in evaluating the plant-model mismatch compared to reflux flow since the reboiler duty plant measurement is not measured in the plant but computed using energy balance on the side stream. The controlled variables, ethane composition at tray 24 and temperature at tray 85, have the same behaviour overall with regions showing a mismatch between the plant and the simulation. The mismatch may be caused by the tuning parameters used in the controllers. The product's composition, ethane mole fraction of ethylene stream, and ethylene mole fraction of ethane stream, have the same behaviour overall with bias and some regions showing a mismatch between the plant and the simulation. Beyond the possible parameter and model structure issues, there are three other key reasons for the plant-model mismatch: (1) the feed composition of the tower is not measured, but instead is inferred from upstream process information. As a result, it might not be the actual tower feed composition. (2) The feed composition considers that the feed contains only ethane and ethylene and ignores the small amount of methane, propylene, and propane which is the reason for the large bias in the bottom product composition. (3) Tower separation efficiency was assumed to be 0.9 for all trays which may not be the case for the real plant. Note that the above results are only to establish that the simulation captures the key aspects of the plant dynamics making the simulation match the plant data quantitatively is not the focus or contribution of the present work.

5.3. OF-NMPC based on Hybrid Model

In this section, OF-NMPC with its hybrid model are introduced. First, the NARX system identification method is described, and the results obtained from system identification are presented. Next, the steady-state model used to capture tower flooding is presented. Finally, the OF-NMPC based on the hybrid model is presented.

5.3.1. System Identification Using NARX Model

Artificial Neural Networks (ANNs) have received significant attention in the area of modeling and identification of dynamic systems in the recent years. Recurrent Neural Networks (RNNs) have been the main approach to model dynamic systems. RNN models are effective in modeling dynamic systems due to their structure which provides the network with a dynamic memory through delayed feedback loop. In this work, a NARX network was utilized to identify a dynamic model of the process to be used within OF-NMPC. The NARX network was trained

using a feed forward network architecture to perform a single step prediction using past output measurements as additional inputs (along with the past and current inputs) as shown in Eq. (1) below:

$$\hat{y}_k = f(y_{k-1}, \dots, y_{k-N_y}, u_{k-1}, \dots, u_{k-N_u}) \quad (1)$$

where $f(.)$ is the mapping function of the neural network, $\hat{y}_k$ is the predicted output of the NARX neural network at the time $k-1$ for the time k. $y_{k-1}, \dots, y_{k-N_y}$ are the past outputs, and $u_{k-1}, \dots, u_{k-N_u}$ are the past inputs. N_y and N_u are the number of output and input delays. Once the training is complete, the NARX network uses closed-loop architecture to perform multi-step predictions as shown in Eq. (2), where the predicted outputs are fed back as additional inputs (along with the past and current inputs). There are two advantages of training the NARX network as a single step with open-loop architecture. First, the input to the feedforward network is more accurate since the true past output values are used as inputs instead of the predicted outputs. Second, the open-loop structure can be modelled as purely feedforward architecture which allows the use of traditional training algorithm such as static backpropagation during training, and takes the following form:

$$\hat{y}_k = f(\hat{y}_{k-1}, \dots, \hat{y}_{k-N_y}, \dots, u_{k-N_u}) \quad (2)$$

where $\hat{y}_{k-1}, \dots, \hat{y}_{k-N_y}$ are predicted past outputs. The NARX model can be classified as an input-output model, where the output is expressed directly in terms of the inputs. In this work, we set N_u and N_y to equal 1. The NARX model is trained using rich data for model identification purposes by exciting the system inputs (reflux flow and reboiler duty) using the Random Gaussian Signal (RGS). In this work, the RGS signal was utilized instead of Pseudo Random Binary Sequence (PRBS) since it has been shown in practice to perform better at capturing nonlinear dynamics as opposed to PRBS, which is better for processes with linear dynamics (Shariff et al., 2013). Once the data used for system identification is generated the inputs and outputs were standardized before being fed to the NARX network. The structure of the network (number of hidden layers and number of neurons), and the activation function of the hidden layers were chosen by trial and error.

The inputs of the NARX model are the two manipulated variables (reflux flow and reboiler duty), and the disturbances (feed flow, feed ethylene composition, and feed temperature). The

outputs of the NARX model are the ethane composition at tray 18, the temperature at tray 85, the top product ethane composition, and the bottom product ethylene composition. The NARX network is employed using MATLAB machine learning and deep learning toolbox. To this end, the input and the output data samples are first standardized before training the open-loop NARX network. One hidden layer with 6 neurons is utilized in the NARX network. The hyperbolic tangent activation function (tanh) is used in the hidden layer, and the linear activation function is used in the output layer. The Levenberg-Marquardt algorithm is used to train the NARX network. 1,500 data points are used for training and 300 data points are used for validation. The validation inputs (standardized reflux flow, standardized reboiler duty, standardized feed flow, feed ethylene composition, and feed temperature) are presented in Figure 5. The measured and predicted outputs (ethane composition at tray 18, the temperature at tray 85, the top product ethane composition, and the bottom product ethylene composition) are presented in Figure 6. The results show that the disturbances have a major impact on the output measurements. Also, the NARX model has the capability to capture the process dynamics.

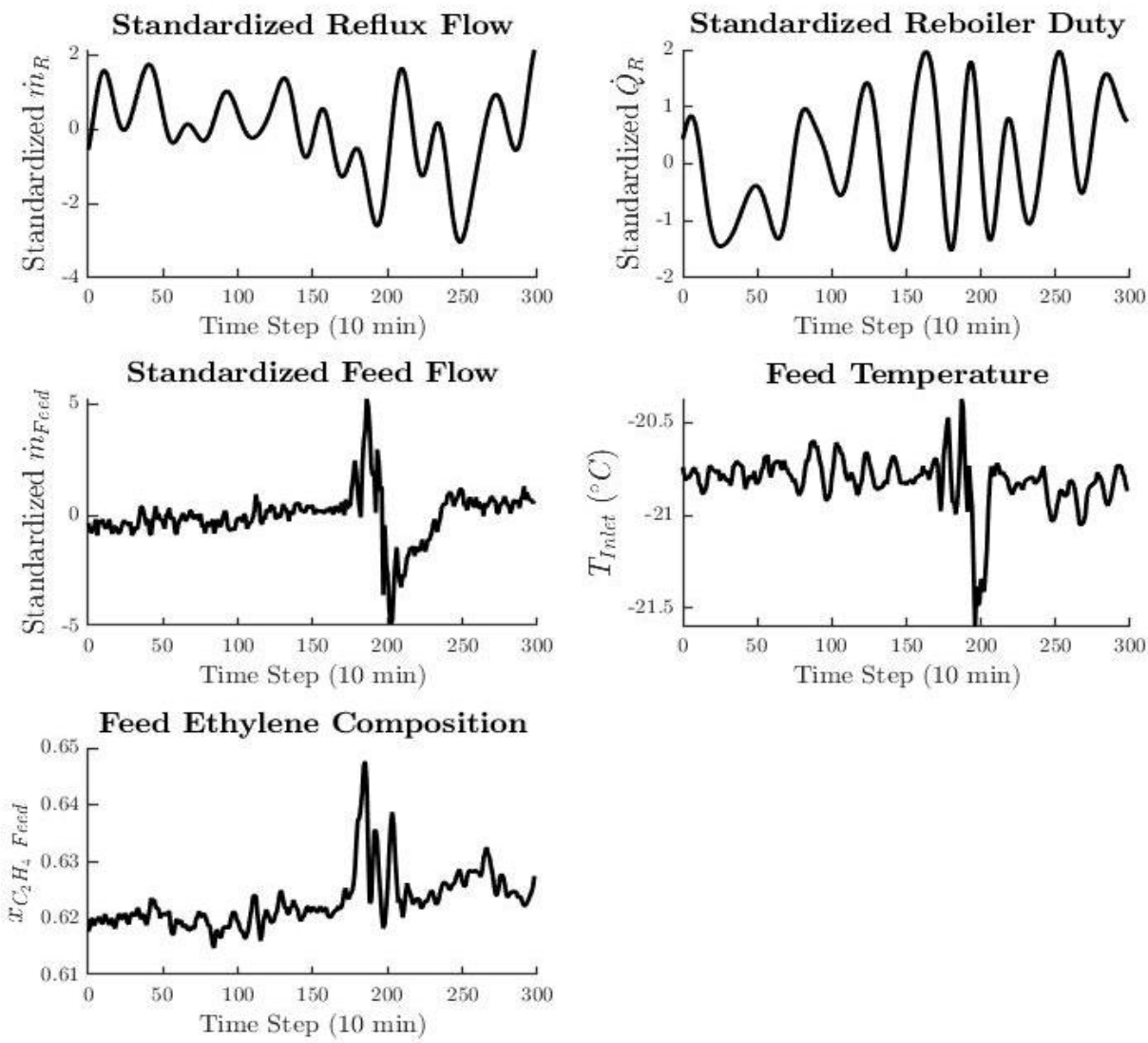

Figure 5. validation of manipulated and disturbance variables under RGS signal in the manipulated variables

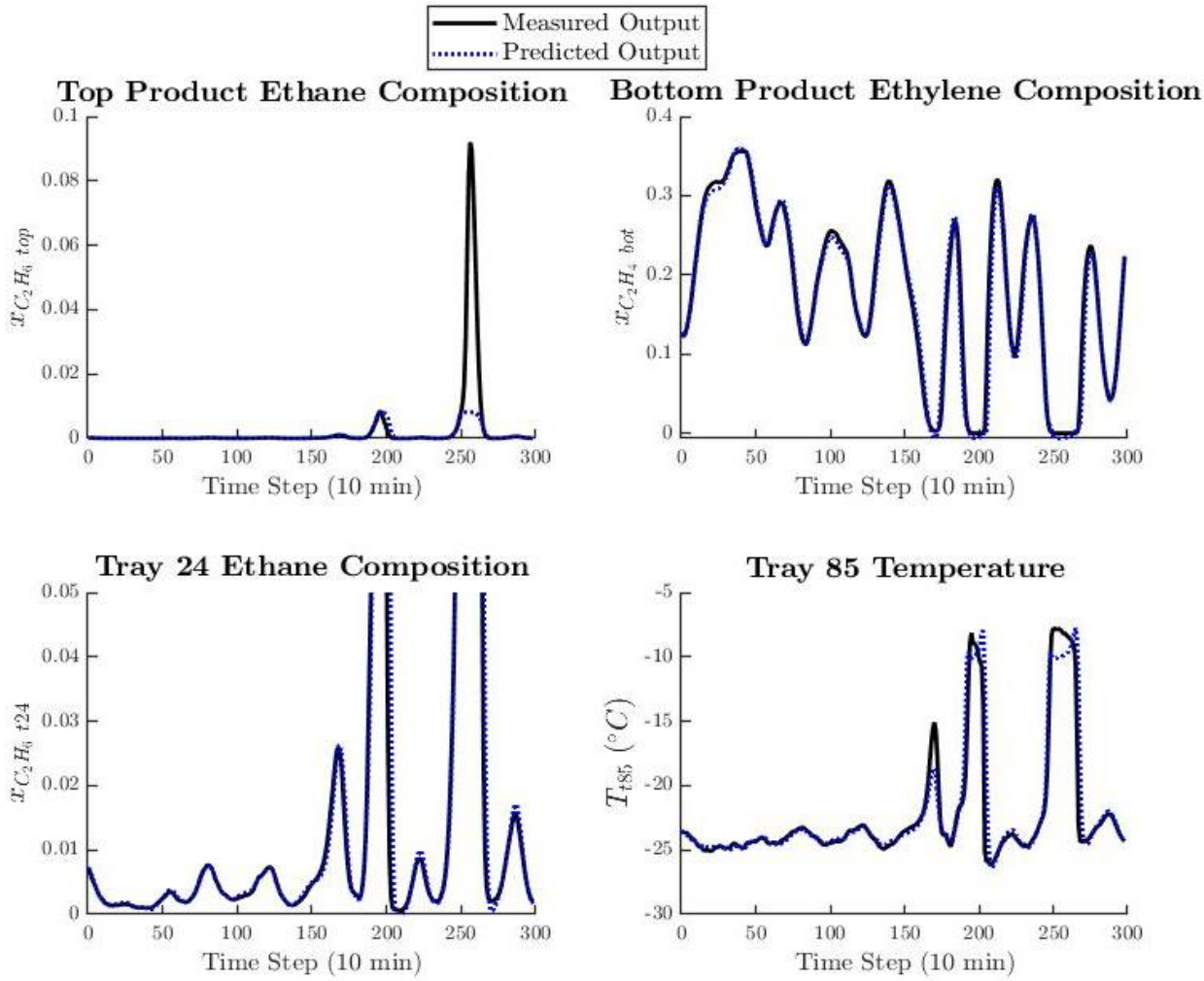

Figure 6. validation of output results under RGS signal in the manipulated variables

5.3.2. Steady State Model

In this section, the steady-state model used in the OF-NMPC controller is presented (see Figure 7 for a visual depiction of how the steady state model has been developed and the information it needs from the Aspen Plus model and the dynamic model). To this end, empirical correlations, and information from Aspen Plus model and dynamic model are used to compute the steady state liquid flow at tray 102 in the tower (the flooding tray). The steady state model is first developed in Aspen Plus V10 using the Peng Robinson model for property estimation. The tower stage Murphree efficiency is set to a value of 0.9 to give close results to the real plant operation. The Aspen Plus model is used to generate a base case set of data with normal operations for the input and output streams and internal tower liquid and vapor flows. Specifically, the data generated are: (1) base case temperatures, (2) base case liquid and vapour stream mass enthalpies, and (3)

liquid and vapour specific heat capacities. These parameters are assumed to be constant, therefore, the Aspen Plus model is only needed to run once to get these parameters. The steady state model developed assumes that: (1) the pressure across the tower is constant, and (2) the specific heat capacities of the liquid and vapour are independent of temperature. With these assumptions, the Kirchhoff's Law can be used to adjusted enthalpies of all streams with change in temperature measurements as shown below:

$$h = h^0 + C_{P,L}(T - T^0) \tag{3}$$

$$H = H^0 + C_{P,V}(T - T^0) \tag{4}$$

where h and H are the adjusted heat enthalpies of the liquid and vapour, and h^0 and H^0 are the base case liquid and vapour enthalpies obtained from Aspen Plus under normal operations. $C_{P,L}$ and $C_{P,V}$ are the heat capacities of liquid and vapour which are obtained from Aspen Plus. T, T^0 are the measured temperature and the base case temperature, respectively. The Aspen Plus model requires the following inputs for base case operation:

- Feed flow rate, temperature, composition, pressure, vapour fraction, reflux flow, and reboiler duty.
- Products flow rates and purities.

The other information needed in the steady-state model are obtained from the dynamic model or the plant measurements when implemented on the real plant. This information needs to be updated with the recent available measurements. The measurements needed from the dynamic model are the feed disturbances (feed flow, feed temperature, and feed vapor fraction), the top and product temperatures, and tray 2, 24, 85, and 102 temperatures. Temperatures of tray 2 and tray 102 are not measured in the plant, therefore, we approximate them with linear interpolation using the closest measured tray temperature and the base case temperature as shown in Eqs. (5) and (6). Tray 2 temperature is computed using top product temperature and tray 24 temperature. Tray 102 temperature is computed using bottom product temperature and tray 85 temperature.

$$T_{t2} = \left(\frac{T_{t24} - T_{top}}{T^0_{t24} - T^0_{top}}\right)\left(T^0_{t2} - T^0_{top}\right) + T_{top} \tag{5}$$

$$T_{t102} = \left(\frac{T_{bot} - T_{t85}}{T^0_{bot} - T^0_{t85}}\right)(T^0_{t102} - T^0_{t85}) + T_{t85} \tag{6}$$

The steady state model used within the OF-NMPC is developed using mass and energy balances. The mass and energy balance equations for the distillation tower are used to compute the top and bottom products flow. The mass and energy balances of the distillation column are shown in Eqs. (7) and (8), respectively.

$$\dot{m}_F = \dot{m}_{top} + \dot{m}_{bot} \tag{7}$$

$$(1 - v_F)h_F\dot{m}_F + v_F H_F \dot{m}_F + \dot{Q}_R = \dot{Q}_C + h_{top}\dot{m}_{top} + h_{bot}\dot{m}_{bot} \tag{8}$$

where h_F and H_F are the liquid and vapour heat enthalpies of the feed, respectively. h_{top} and h_{bot} are the liquid heat enthalpies of the top and bottom products, respectively, v_F is the feed vapour fraction, and $\dot{Q}_C$ is the condenser heat duty. The goal is to compute the top and bottom product flows as a function of reflux flow and reboiler duty. The feed flow rate in the plant is measured, and the vapour fraction in the plant is not measured in the plant but cab be inferred from upstream process measurements and feed tower conditions. The heat enthalpies of the feed, products, internal liquid and vapour flows are estimated using the base case data and their associated measured temperatures assuming constant pressure as shown in Eqs. (3) and (4). The condenser duty $\dot{Q}_C$ is approximated using Eq. (9).

$$Q_c = (\dot{m}_{top} + \dot{m}_R)(H_{t2} - h_{top}) \tag{9}$$

where H_{t2} and h_{top} are the vapour heat enthalpy at tray 2 and liquid heat enthalpy of the top product, respectively, which are estimated using Eqs. (3) and (4). Note that computing these enthalpies requires knowing the temperature; the temperature of the top product is measured, while the temperature of the vapour stream at tray 2 is approximated using the linear interpolation equation, Eq. (5).

Substitute Eq. (7) and Eq. (9) into Eq. (8) and solve for top product flow F_{top} gives us the following equation:

$$\dot{m}_{top} = \frac{(1 - v_F)h_F\dot{m}_F + v_F H_F \dot{m}_F + \dot{Q}_R - \left(H_{t2} - h_{top}\right)\dot{m}_R - h_{bot}\dot{m}_F}{H_{t2} - h_{bot}} \tag{10}$$

Once the top product flow $\dot{m}_{top}$ is computed, Eq. (7) can be used to compute the bottom product flow $\dot{m}_{bot}$.

Once both top and bottom products flow are known, the liquid flow at tray 102 can be computed using mass and energy balances at the bottom of the tower which are shown in Eqs. (11) and (12).

$$\dot{m}_{L\,t102} = \dot{m}_{V\_103} + \dot{m}_{bot} \tag{11}$$

$$\dot{Q}_R + h_{t102}\dot{m}_{L\,t102} = H_{103}\dot{m}_{V\,t103} + h_{bot}\dot{m}_{bot} \tag{12}$$

where $\dot{m}_{L\,t102}$ is the liquid flow at tray 102 and $\dot{m}_{V\,t103}$ is the vapour flow coming from the reboiler.

Substitute Eq. (11) into Eq. (12) gives us the following equation:

$$\dot{m}_{L\,t102}(H_{t103} - h_{t102}) = \dot{Q}_R + (H_{t103} - h_B)(\dot{m}_F - \dot{m}_{top}) \tag{13}$$

$\dot{m}_{L\,t102}$ as a function of $\dot{Q}_R$ and $\dot{m}_R$ by can be determined by using E. (10) and Eq. (13) as follows:

$$\begin{aligned}\dot{m}_{L\,t102}(H_{t103} - h_{t102}) + (H_{t103} - h_{bot})\left(\frac{((1-v_F)h_F + v_F H_F - h_{bot})\dot{m}_F}{H_{t2} - h_{bot}} - \dot{m}_F\right) \\ = \left[\frac{(H_{t103} - h_{bot})(H_{t2} - h_{top})}{H_{t2} - h_{bot}}\right]\dot{m}_R + \left[1 - \frac{(H_{t103} - h_{bot})}{H_{t2} - h_{bot}}\right]\dot{Q}_R\end{aligned} \tag{14}$$

Eq. (14) can be written in a matrix form with $\dot{m}_R$ and $\dot{Q}_R$ being the variables. This equation requires the left side to be greater than the right side to avoid flooding. The final flooding equation is shown in Eq. (15).

$$\begin{aligned}\left[\frac{(H_{t103} - h_{bot})(H_{t2} - h_{top})}{H_{t2} - h_{bot}} \quad 1 - \frac{(H_{t103} - h_{bot})}{H_{t2} - h_{bot}}\right]\begin{bmatrix}\dot{m}_R \\ \dot{Q}_R\end{bmatrix} \\ \leq \left[\dot{m}^{max}_{L\,t102}(H_{t103} - h_{t102}) + (H_{t103} - h_{bot})\left(\frac{((1-v_F)h_F + v_F H_F - h_{bot})\dot{m}_F}{H_{t2} - h_{bot}} - \dot{m}_F\right)\right]\end{aligned} \tag{15}$$

where $\dot{m}^{max}_{L\,t102}$ is the maximum liquid flow allowed at tray 102 to avoid tower flooding.

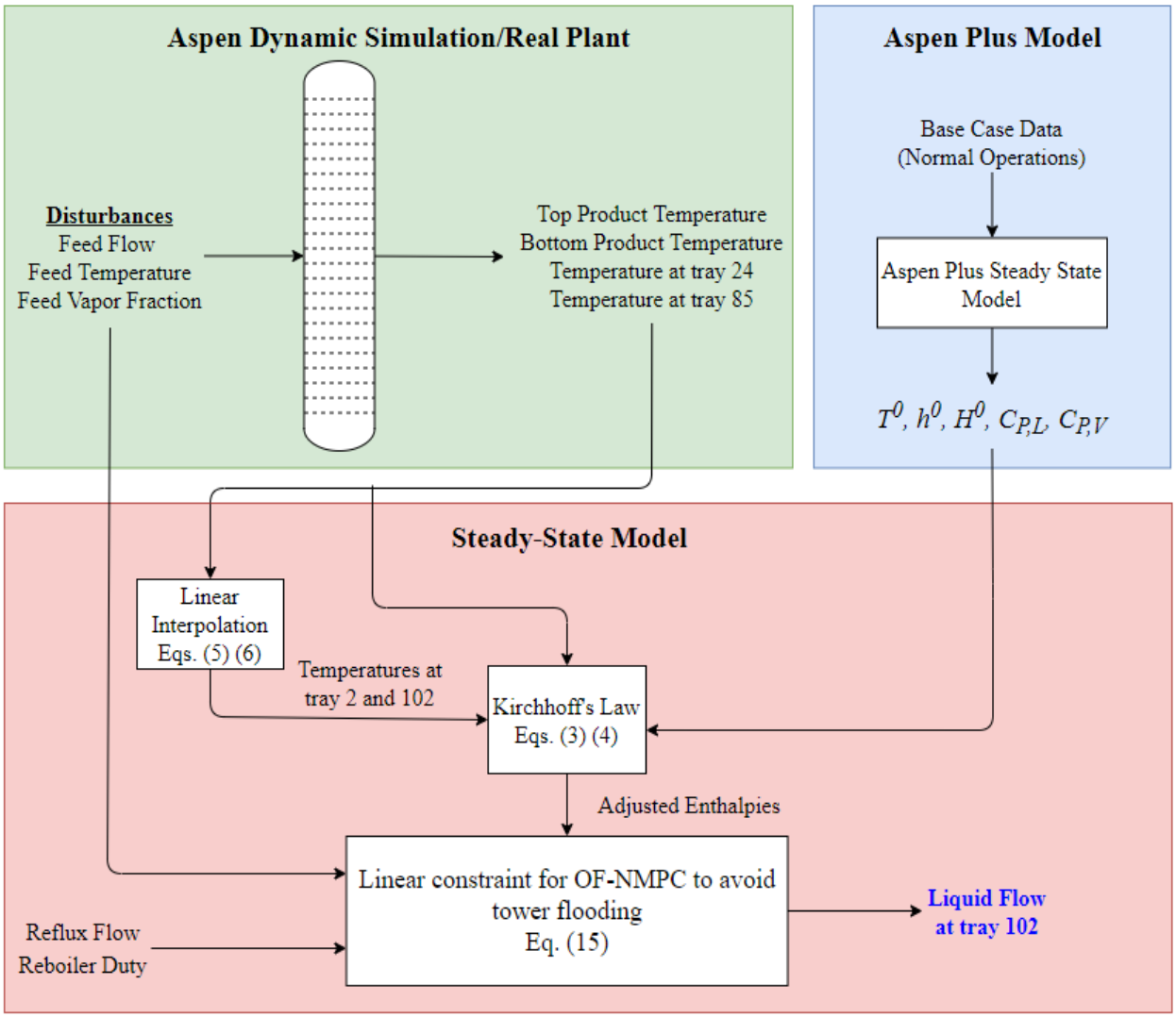

Figure 7. steady-state model implementation within OF-NMPC

In summary, the steady state model uses the following variables, parameters, and inputs to compute the liquid flow rate at tray 102, $\dot{m}_{L\,t102}$:

- Process measured variables: feed flow $\dot{m}_F$, feed vapour fraction v_F, temperatures of the feed T_F, top product T_{top}, bottom product T_{bot}, tray 24 T_{t24}, and tray 85 T_{t85}. The temperatures are used to compute the enthalpies of the streams as shown in Eqs. (3) and (4).
- Base case parameters:
  - Temperatures of the feed T_F^0, top product T_{top}^0, bottom product T_{bot}^0, tray 2 T_{t2}^0, tray 24 T_{t24}^0, tray 85 T_{t85}^0, tray 102 T_{t102}^0.

- Enthalpies of the vapour feed H_F^0, liquid feed h_F^0, liquid top product h_{top}^0, liquid bottom product h_{bot}^0, vapour at tray 2 H_{t2}^0, liquid at tray 102 h_{t102}^0, vapour at tray 103 H_{t103}^0.
- Liquid heat capacity of feed $C_{P,L\,F}$, top product $C_{P,L\,top}$, bottom product $C_{P,L\,bot}$, and tray 102 $C_{P,L\,t102}$, vapour heat capacity of feed $C_{P,V\,F}$, tray 2 $C_{P,V\,t2}$, and tray 103 $C_{P,V\,t103}$.

- The temperatures of tray 2 T_{t2}, and tray 102 T_{t102} are approximated with linear interpolation using the closest measured tray temperature as shown in Eqs. (5) and (6). Then, these temperatures are used to compute the enthalpies of the stream as shown in Eqs. (3) and (4).

It is important to note that since the steady state model requires output measurements such as product temperatures, the information used from the dynamic model is obtained from the previous timestep measurements. Figure 8 shows the actual liquid flow at tray 102, the predicted liquid flow at tray 102 computed from the steady state model using current measurement and previous timestep measurement, and the flooding flow assumed. The flooding flow was assumed to be 470,000 kg/hr. These flow profiles are based on the results obtained from controlling the tower using the two PI controllers. The results show that steady-state model can capture the trend of the actual flow, albeit with some bias. Also, the results show that using the previous timestep measurements as opposed to the current step measurements (some of them will not be available yet) do not impact the steady state liquid flow at tray 102. This demonstrates that the steady state model can be implemented within the proposed OF-NMPC to avoid tower flooding. While the computed liquid flow at tray 102 shows somewhat a similar trend to the actual liquid flow at tray 102, there is a bias in the model which is shown in Figure 9. The bias seems to be somewhat constant with some fluctuation at a value of 2000 kg/hr under normal operations. However, when the tower is reaching pre-flooding or under flooding, the bias values seem to fluctuate rapidly which is believed to be due to the accumulation of the liquid inside the tower making the steady state calculations unreliable. Further details about mitigating the issue of bias in the steady-state calculation versus the actual flow are discussed in Section 4.

Remark 1. The steady-state calculation has roughly constant bias at normal operation (similar feed flow), but this bias increases or decreases based on the feed flow due to the accumulation of the liquid inside the tower. This is mainly because the steady-state calculation does not capture

the dynamic of the process, and only uses previous timestep measurements. This shows that the steady-state calculations are unreliable under high and fast disturbance in the feed flow.

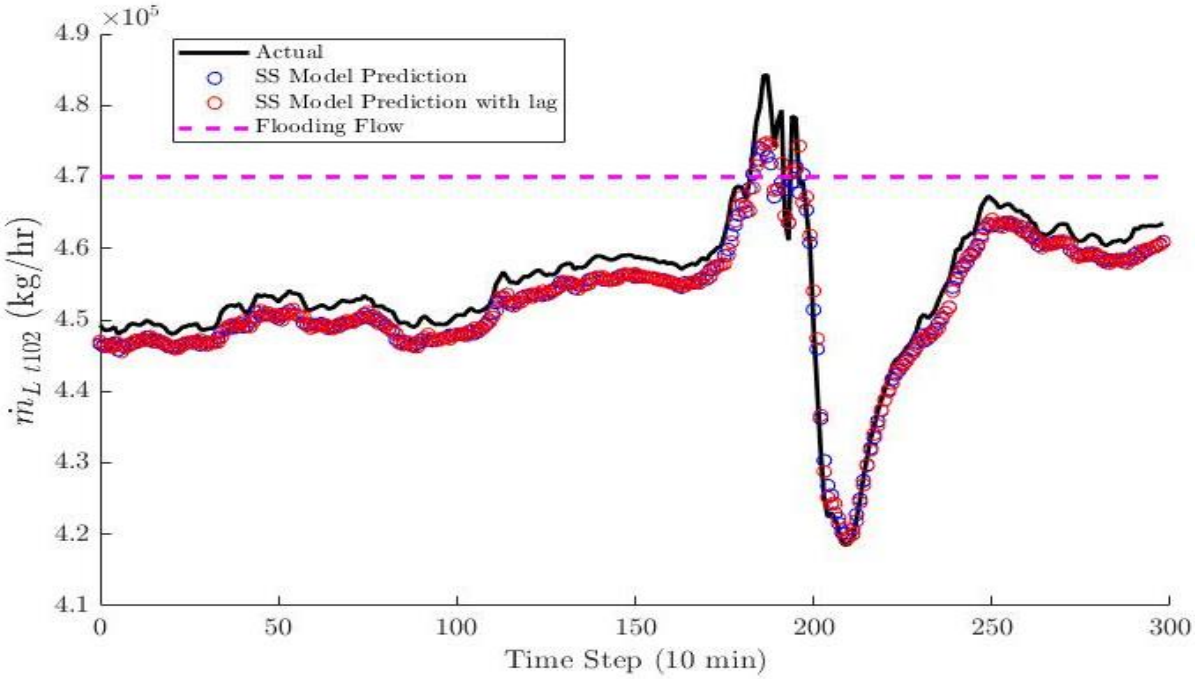

Figure 8. a comparison of the actual liquid flow and steady state computed liquid flow at tray 102

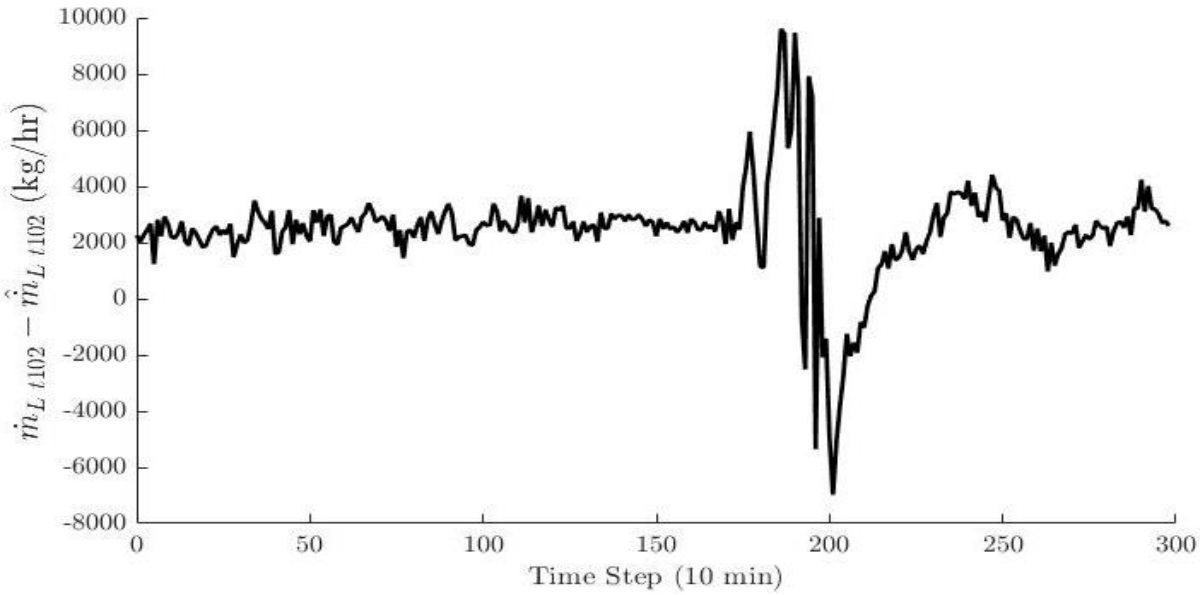

Figure 9. bias of the actual liquid flow and steady state computed liquid flow with delay at tray 102

5.3.3. OF-NMPC Formulation based on Hybrid Model

The objective of the OF-NMPC is to utilize the hybrid model (NARX model and steady-state model) in MPC implementation to achieve offset free and flooding free control. The NMPC implementation is shown in Figure 10. The NMPC based on hybrid model formulation can be described as follows:

$$\min_{u_k,..,u_{k+P}} \sum_{j=0}^{P} \left[\left\| \hat{y}_{k+j} - y_{k+j}^{SP} \right\|_{Q_y}^2 + \left\| u_{k+j} - u_{k+j-1} \right\|_{R_u}^2 \right] \tag{16}$$

$$s.t. \hat{y}_k = f\left(y_{k-1}, \dots, y_{k-N_y}, \dots, u_{k-N_u}\right) \ \forall\, k = 1 \tag{17}$$

$$\hat{y}_k = f\left(\hat{y}_{k-1}, \dots, \hat{y}_{k-N_y}, \dots, u_{k-N_u}\right) \ \forall\, k = 2, \dots, P \tag{18}$$

$$\left[\frac{(H_{t103} - h_{bot})(H_{t2} - h_{top})}{H_{t2} - h_{bot}} \quad 1 - \frac{(H_{t103} - h_{bot})}{H_{t2} - h_{bot}} \right] u_k \le \left[\dot{m}_{L\,t102}^{max}(H_{t103} - h_{t102}) + (H_{t103} - h_{bot}) \left(\frac{\left((1 - v_F)h_F + v_F H_F - h_{bot}\right)\dot{m}_F}{H_{t2} - h_{bot}} - \dot{m}_F \right) \right] \ \forall\, k = 1, \dots, P \tag{19}$$

$$u_{min} \le u_k \le u_{max} \ \forall\, k = 1, \dots, P \tag{20}$$

where P represents the prediction horizon, y_{k+j}^{SP} is the desired setpoint values for the outputs, $\hat{y}_{k+j}$ is the prediction of output, Q_y and R_u are the penalty matrices corresponding to the output deviations from the setpoints and the rate of change in the inputs, and u_{min} and u_{max} represent the lower and upper bounds of the manipulated inputs. It should be noted that the NARX-based predictive model computes multi-step ahead predictions. Thus, the initial values of inputs and outputs are fed to the model using the last current measured values and then the values of the outputs are calculated one step ahead as shown in the Eq. (17). Afterwards, the predicted outputs, along with candidate future input values are fed to the model to predict the outputs two steps ahead. Subsequently, this procedure is performed recursively to predict the outputs multi-steps ahead up to the prediction horizon as shown in the Eq. (18). Eq. (19) limits the liquid flow at tray 102 to avoid tower flooding. Eq. (20) constraint limits the inputs with upper and lower bounds.

The current control strategy may fail to effectively control the distillation column due to process-plant mismatch. Furthermore, unmeasured disturbances can result in a gap between the model and the process which may lead to degradation in the NMPC performance. In this work, we

adapt a widely used offset free scheme in industrial NMPC implementation which shifts the set points using constant bias to compensate for plant-model mismatch. The bias is computed by comparing the measured process output y_k and the predicted process output $\hat{y}_k$ at the current sampling instant k as

$$e_k = \hat{y}_k - y_k \tag{21}$$

Once the bias is computed at the sampling instant k, this bias is assumed to remain constant in the future (for the MPC calculations) and the set point is modified as:

$$y_{k+j}^{SP} = y^{SP} + e_k\,, \quad 1 \le j \le P \tag{22}$$

Subsequently, this bias is continuously updated to modify the set points accordingly.

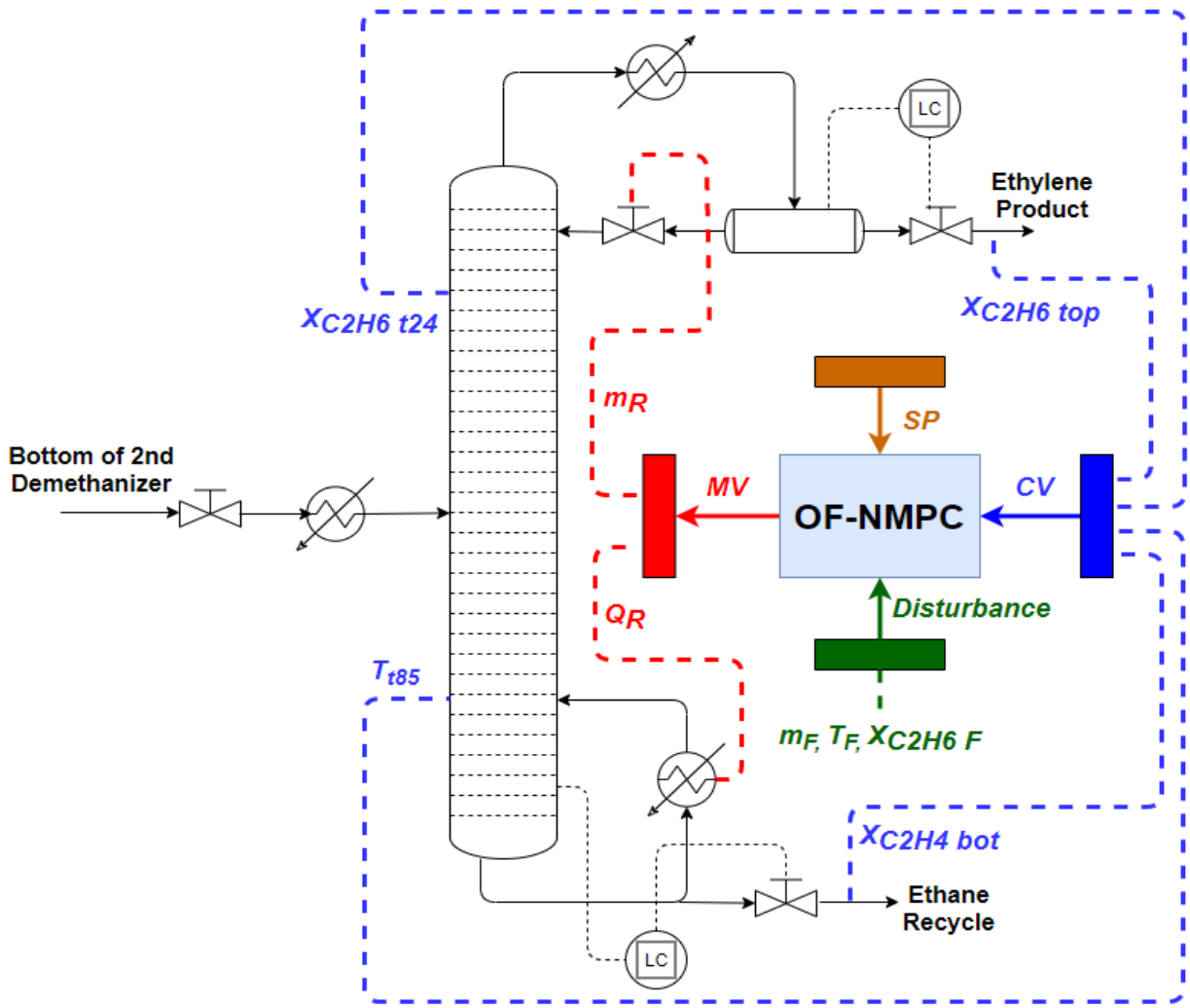

Figure 10: OF-NMPC implementation on the distillation column

5.4. Results of the hybrid model based OF-NMPC

In this section, the proposed OF-NMPC strategy is implemented on the ethylene splitter model in Aspen dynamics with the goal of maintaining ethylene purity at the desired level and avoiding flooding. The four controlled variables in our system are the top and bottom products' compositions ($x_{C_2H_6\,top}$ and $x_{C_2H_4\,bot}$) and the two controlled variables by the PI controllers ($x_{C_2H_6\,t\_24}$ and T_{t85}). The two manipulated variables are the reflux flow and reboiler duty ($\dot{m}_R$ and $\dot{Q}_R$). The optimization problem of the NMPC is solved using *fmincon* solver in MATLAB. The optimization problem uses the previous timestep manipulated variables solutions as an initial guess to reduce execution times. The NMPC parameters are chosen as follows: sampling time T_s = 10 mins, prediction horizon P = 4, Outputs penalty Q_y = diag([1e2, 1e2, 1e10, 1e5]), and change in inputs penalty R_u = diag([1e-4, 1e4]). These penalties are chosen based on the magnitude of the variable and its importance. The manipulated inputs constraints were chosen as follows: $\dot{m}_R \in$ [360,000 460,000] and $\dot{Q}_R \in$ [120 150]. The setpoints of the outputs were chosen as follows: [5e-4, 1e-2, 0.0145, -20.8]. Note that these setpoints change after every iteration due to the off-set free feature based on the difference between the model predicted output and the process actual output from the previous timestep. The OF-NMPC results is first generated without including the flooding constraint which is included in the Appendix. These results show that OF-NMPC control strategy can improve the top product purity control but fails to eliminate tower flooding. Therefore, incorporate the steady-state model within the OF-NMPC is essential to achieve flooding free control.

The results of the OF-NMPC strategy with constraint are generated using different constrained value of the highest allowed liquid flow at tray 102, which demonstrate different level of conservatism to avoid tower flooding. The values were chosen to be 470000 kg/hr (low conservatism), 464000 kg/hr (medium conservatism), and 458000 kg/hr (high conservatism). The results of the OF-NMPC strategy with flooding constraint using different level of conservatism compared to PI controller strategy are presented in Figure 11. The results show that a more conservative control results in higher deviation of the products purities from their desired set points. This is expected as a tighter constraint on the flow at tray 102 prevents the OF-NMPC from increasing the reflux flow and reboiler duty to maintain the products purities at their desired level. The results of the actual and computed liquid flow at tray 102 via the steady-state model

for the PI control strategy and OF-NMPC with flooding constraints at different maximum liquid flow threshold are presented in Figure 12. The results show that to avoid tower flooding (maintain actual liquid flow at tray 102 below 47000 kg/hr), the flooding constraint must be set at 458000 kg/hr to account for the bias between the actual and computed liquid flow at tray 102.

Remark 2. In the present formulation, a trade-off that exists between the two goals: maintain the products' purities at their desired setpoints and avoid tower flooding. At high tower feed, the reflux flow and reboiler duty should be increased to maintain the products purities at the desired level. However, increase in reflux flow and reboiler duty causes the liquid flow at tray 102 to increase, possibly causing flooding. The avoidance of tower flooding being important for the process, the present formulation ends up sacrificing control performance to achieve flooding free control, motivating the modified MPC formulation presented later in the paper.

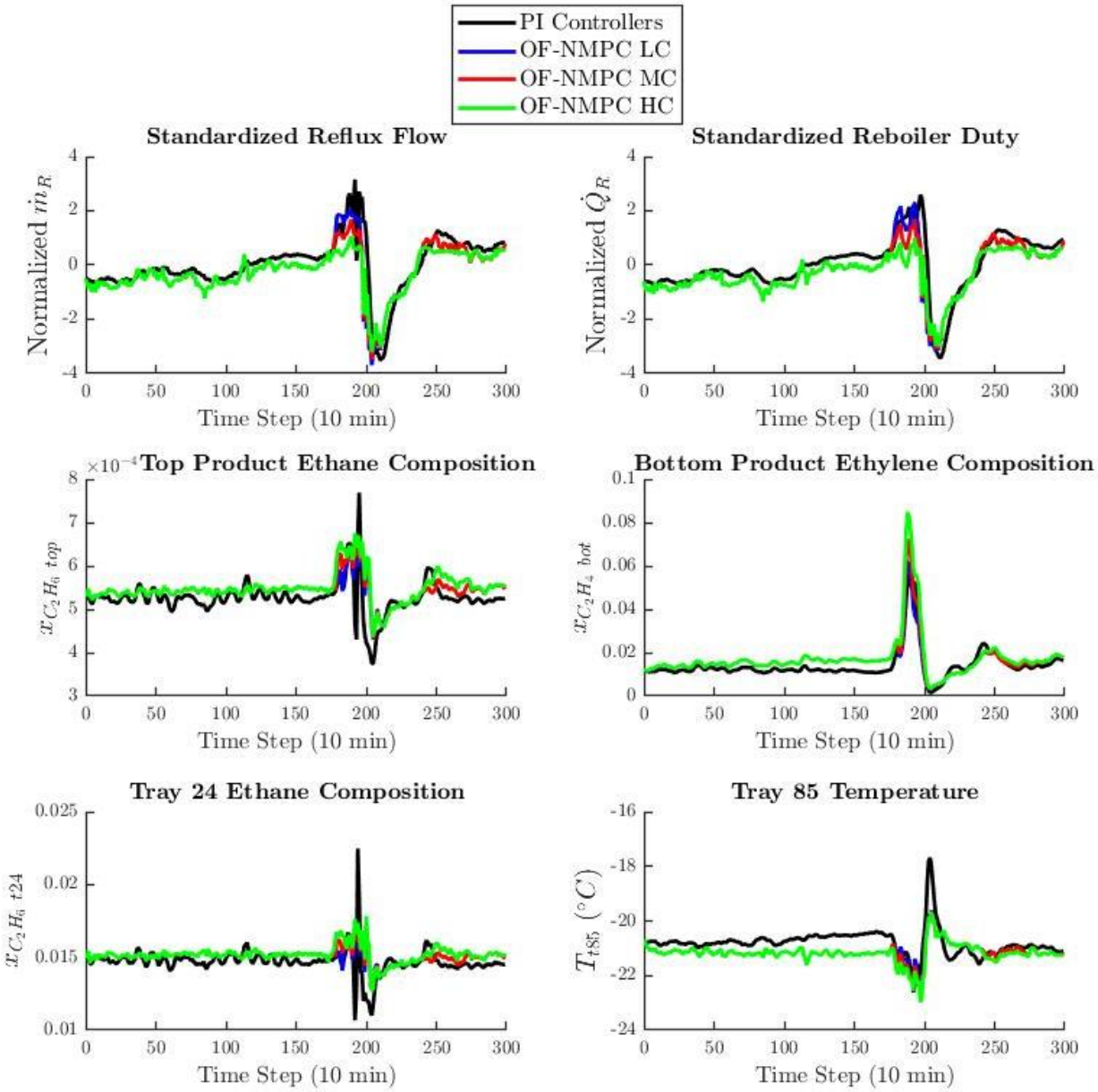

Figure 11. comparison of the manipulated and controlled variables measurements of OF-NMPC control strategy with different level of conservatism versus PI controllers strategy

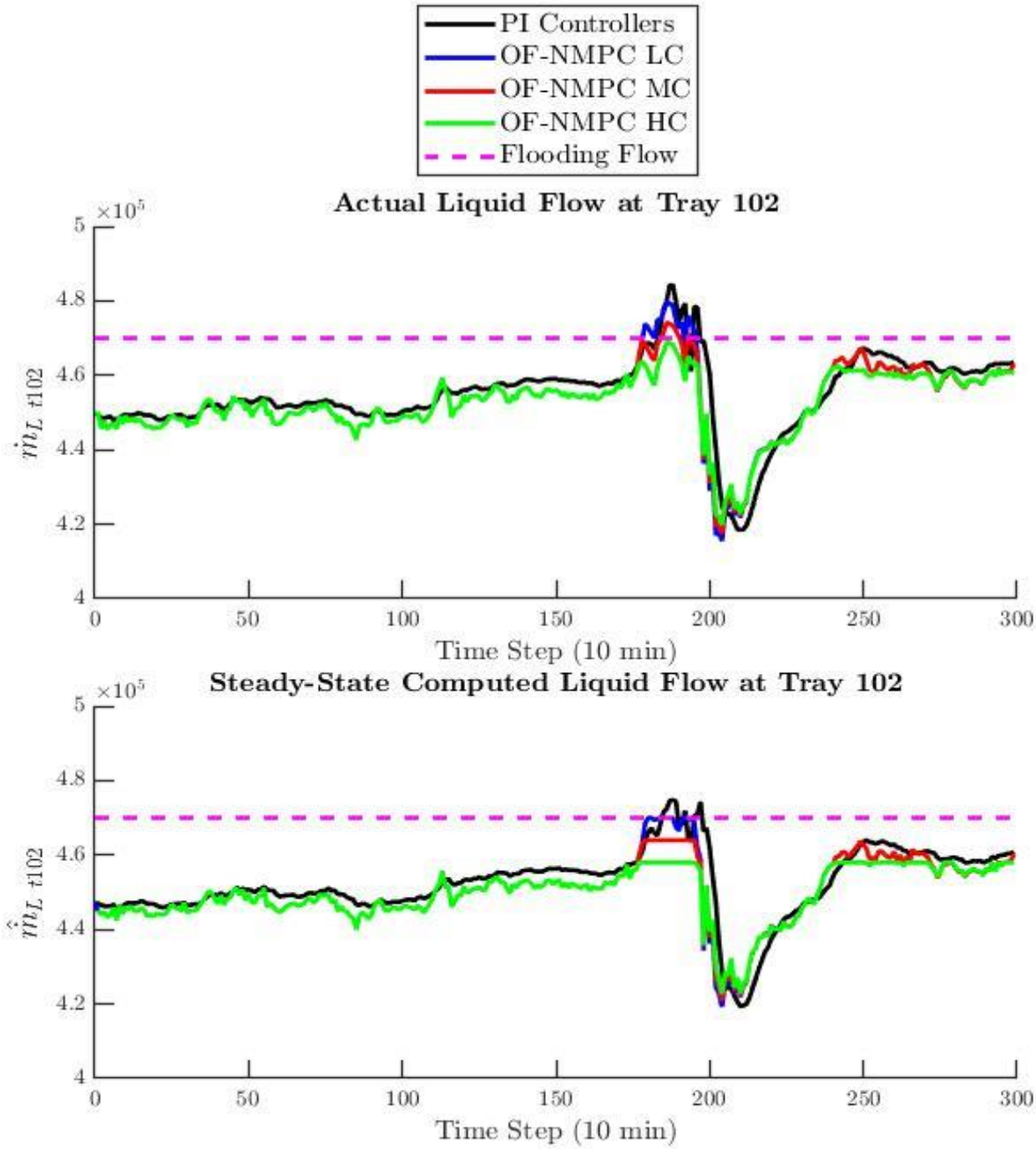

Figure 12. comparison of actual and computed liquid flow at tray 102 of OF-NMPC with different level of conservatism versus PI controllers strategy

In order to measure the effectiveness of the results shown earlier, three metrics are used to evaluate the different control strategies: (1) the variation in the ethylene product purity which was computed using the standard deviation formula. Lower variation in ethylene product purities

indicates that products purities are consistent and does not vary which is desirable. (2) The number of instances where flooding is observed, where a flooding instance corresponds to any timestep the liquid flow at tray 102 measured exceeds 470000 kg/hr. (3) The sum of tray 102 loading above the flooding threshold (470,000 kg/hr) at all flooding instances. Table 2 summarizes the performance of the PI control strategy versus the OF-NMPC strategy with and without flooding constraint using the three metrics. The results show that OF-NMPC can achieve better control of the top product purities compare to PI control strategy. Also, the results show that OF-NMPC with no constraint and low conservatism level constraint results in higher instances of flooding. However, increasing conservatism to medium can lower the instance of flooding compared to PI controllers strategy. OF-NMPC with high conservatism can avoid tower flooding, but it significantly impacts the control of the top products' purity compared to other OF-NMPC strategies.

Table 2: Performance summary of proposed OF-NMPC strategy with and without flooding constraint versus PI controllers strategy

Strategy	SD of ethylene product purity (10^{-5})	# Flooding instances	Tray loading above the flooding constraint (kg/hr)
PI controllers	3.72	11	97445
OF-NMPC with no constraint	2.09	17	208210
OF-NMPC with constraint (low conservatism)	2.52	16	76577
OF-NMPC with constraint (medium conservatism)	2.88	7	16467
OF-NMPC with constraint (high conservatism)	3.36	0	0

In order to reduce the bias between the actual liquid flow at tray 102 and the liquid flow at tray 102 computed from the steady state equation, we next propose adding a bias to the steady state calculation model. The goal of the bias is to improve the steady state calculations of liquid flow at tray 102 by considering the tower feed flow rate and whether the tower operation is under

accumulation or depletion of liquid. To indicate whether the tower's current operation is resulting in accumulation or depletion of liquid at j time step, we use the most recent measurements of the feed flow and the products flow using this formula shown below:

$$Accum(j) = \dot{m}_F(j-1) - \dot{m}_{top}(j-1) - \dot{m}_{bot}(j-1) \tag{23}$$

where $Accum$ is a variable that tracks liquid accumulation or depletion amount. If the tower feed flow rate $\dot{m}_F(j-1)$, is higher than the sum of the products flow rate $\dot{m}_{top}(j-1) + \dot{m}_{bot}(j-1)$then liquid is being accumulated in the tower. However, if the sum of the products flow rate is higher than the feed flow rate, liquid is being depleted from the tower. Under normal ranges of feed flow rate, we expect that minimal or no liquid accumulation or depletion is happening in the tower. Therefore, we only add this bias under these two conditions: (1) the feed flow rate is higher than normal operation and liquid accumulation is happening and, (2) the feed flow rate is lower than normal operation and liquid depletion is happening. The steady state calculation with added bias rules is shown below:

$$\hat{\dot{m}}_{L\ bias\ t102}(j) = \hat{\dot{m}}_{L\ t102}(j) + Accum(j),\ if\ \dot{m}_F(j-1) > 150000\ \&\ Accum(j) > 0 \tag{24}$$

$$\hat{\dot{m}}_{L\ bias\ t102}(j) = \hat{\dot{m}}_{L\ t102}(j) + Accum(j),\ if\ \dot{m}_F(j-1) < 148000\ \&\ Accum(j) < 0 \tag{25}$$

$$\hat{\dot{m}}_{L\ bias\ t102}(j) = \hat{\dot{m}}_{L\ t102}(j), \qquad \text{Otherwise} \tag{26}$$

This modification can be implemented within our OF-NMPC by replacing the variable $\dot{m}^{max}_{L\ t102}$ shown in the OF-NMPC formulation by $\dot{m}^{max}_{L\ t102} - Accum(j)$ if one of two earlier conditions apply. This modification reduced the maximum bias in the liquid flow at tray 102 from approximately 10000 kg/hr to around 6000 kg/hr.

Another concept we utilize in this work is control banding, which is used to reduce the chances of flooding by trying to stay away from the flooding constraint. The control banding considers different levels of conservatism based on how close the tower is from flooding. In this work, we consider two levels of bands which we refer to as high band and low band. High band and low band are considered when the liquid flow at tray 102 is within 0.75% and 1.5% from the flooding flow, respectively. This percentage is computed using liquid flooding flow at tray 102 and the last step predicted liquid flow at tray 102 using the following form.

$$B_{flooding} = \left(\frac{\dot{m}^{max}_{L\ t102} - \hat{\dot{m}}_{L\ bias\ t102}(j-1)}{\dot{m}^{max}_{L\ t102}}\right) 100\% \tag{27}$$

where $B_{flooding}$ is a percentage that measures how close the tower is to flooding. $B_{flooding}$ will have a value between 0% and 100%, where lower values indicates that the tower is close to flooding. When $B_{flooding}$ is lower than 1.5%, the level of conservatism is increased by adjusting $\dot{m}_{L\,t102}^{max}$ using the following formulas:

$$\left[\frac{(H_{t103}-h_{bot})(H_{t2}-h_{top})}{H_{t2}-h_{bot}} \quad 1-\frac{(H_{t103}-h_{bot})}{H_{t2}-h_{bot}}\right]u_k \leq \left[(1-\alpha_{flooding})\dot{m}_{L\,t102}^{max}(H_{t103}-h_{t102}) + (H_{t103}-h_{bot})\left(\frac{\left((1-v_F)h_F+v_F H_F-h_{bot}\right)\dot{m}_F}{H_{t2}-h_{bot}}-\dot{m}_F\right)\right] \tag{28}$$

$$\alpha_{flooding} = \left(\frac{\hat{\dot{m}}_{L\,bias\,t102}(j-1)-0.985\dot{m}_{L\,t102}^{max}}{0.985\dot{m}_{L\,t102}^{max}}\right) \tag{29}$$

The level of conservatism is increased further when $B_{flooding}$ is lower than 0.75% by constraining the upper bound of both reflux flow and reboiler duty within OF-NMPC to their current value. This will prevent OF-NMPC to make moves that increase reflux flow and reboiler duty. Throughout the rest of the paper, we will refer to the OF-NMPC control strategy with modification on the steady state calculation as strategy 1, and to OF-NMPC control strategy with modification on the steady state calculations and control banding as strategy 2.

Figures 13 and 14 present the results of the OF-NMPC with strategy 1 and strategy 2 versus the PI control strategy. The highest allowed liquid flow at tray 102 was set to a value of 464000 kg/hr, which was chosen based on the roughly 6000 kg/hr bias. The results show that both strategies 1 and 2 achieve more stable top product composition and eliminate the big spikes observed in the top product composition with PI control strategy. Also, strategy 1 eliminates most of the flooding instances and the liquid flow at tray 2 gets close to 470000 kg/hr, and slightly goes over the amount in one instance. However, strategy 2 can eliminate the flooding completely as the liquid flow at tray 102 reaches close to 470000 kg/hr but does not exceed the 470000 kg/hr limit. Table 3 summarizes the performance of the PI control strategy versus strategy 1 and strategy 2. The results show that strategy 2 can avoid tower flooding completely and achieve good control over the top product purities.

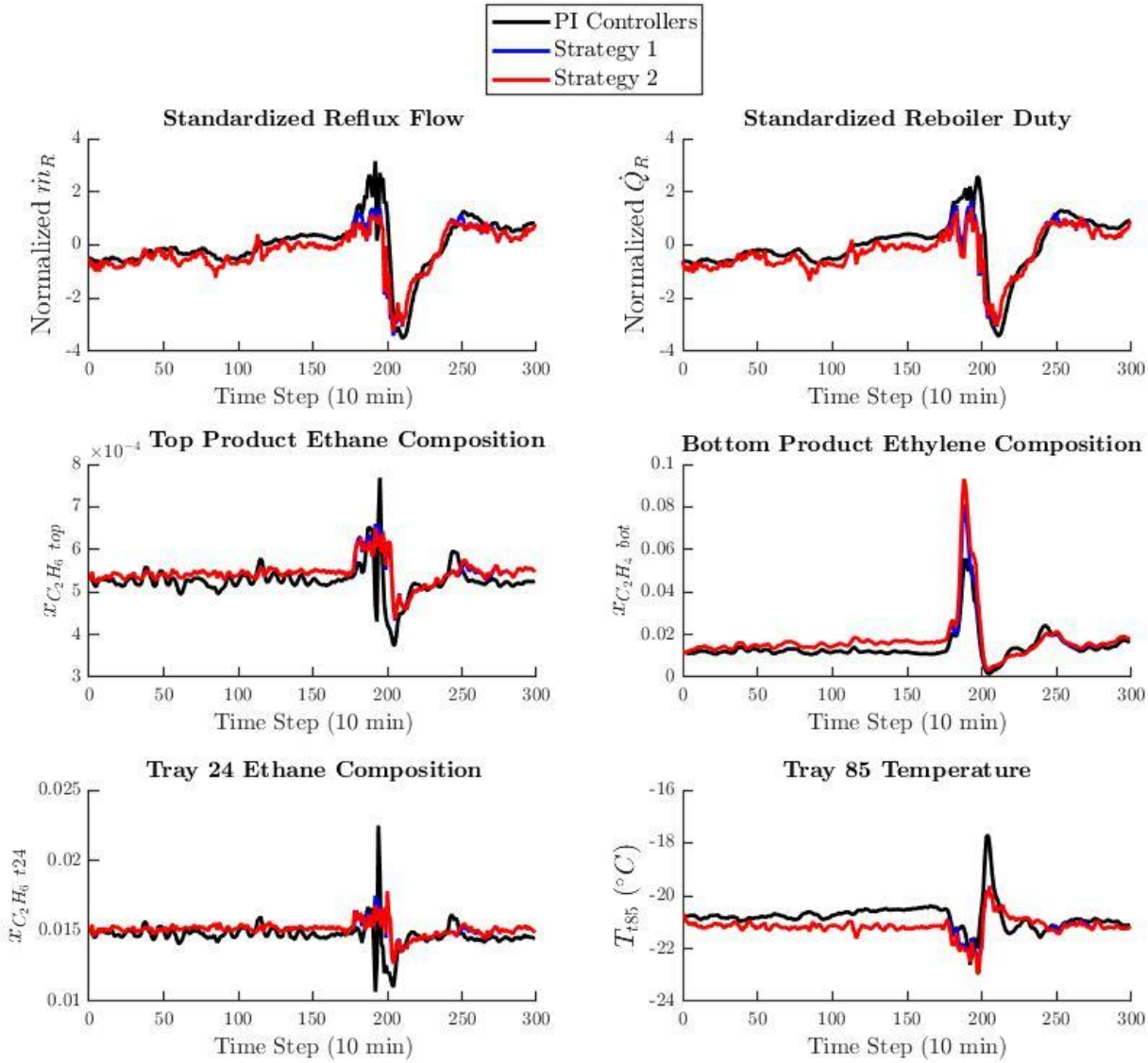

Figure 13. comparison of the manipulated and controlled variables measurements of strategies 1 and 2 versus PI controller strategy

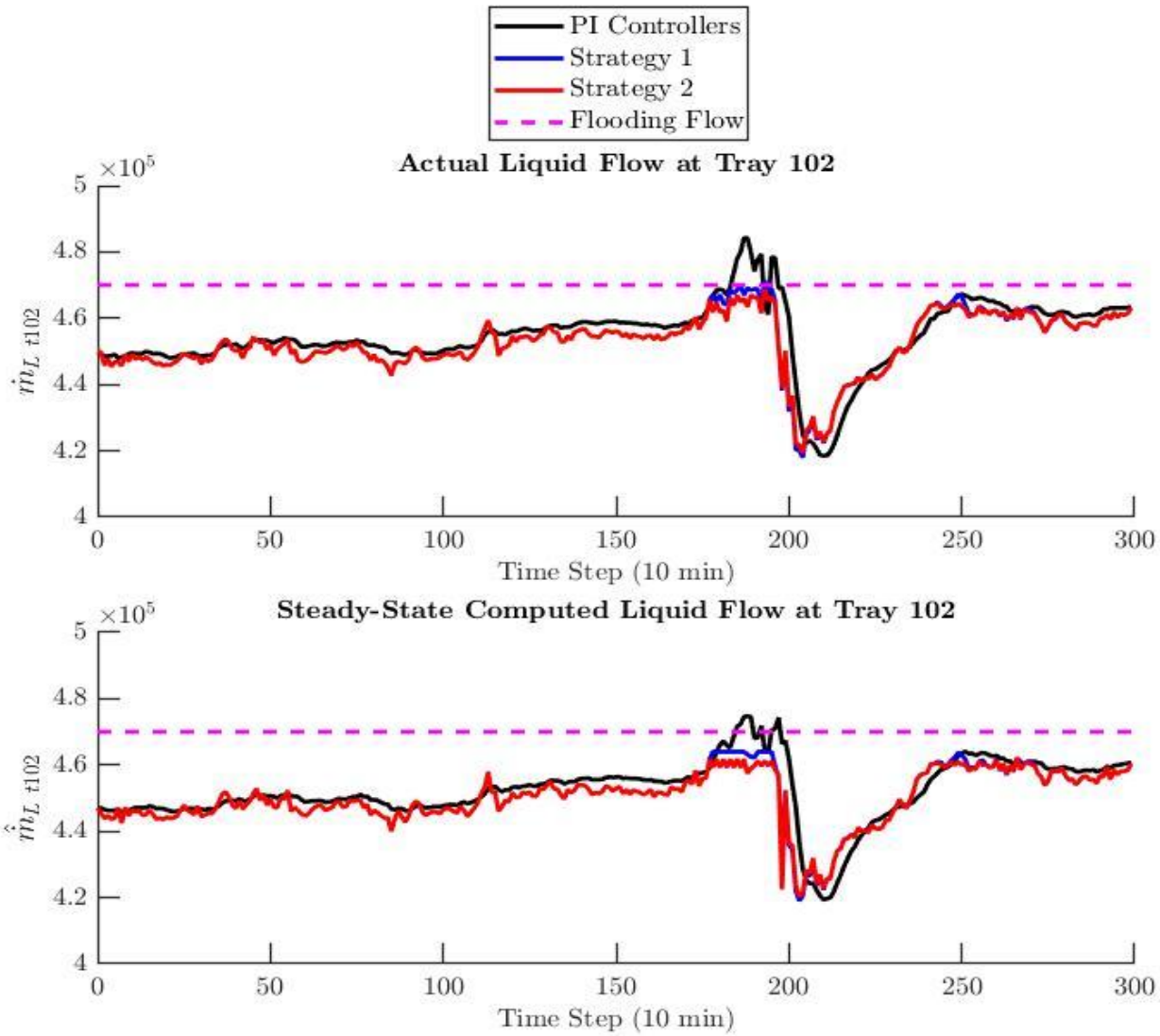

Figure 14. comparison of the actual and computed liquid flow at tray 102 of strategy 1 and 2 versus PI controllers strategy

Table 3: performance summary of the two proposed strategies versus PI controllers strategy

Strategy	SD of ethylene product purity (10^{-5})	Flooding instance	Tray loading above the flooding constraint (kg/hr)
PI controllers	3.72	11	97445
Strategy 1	2.90	1	1378.7
Strategy 2	2.77	0	0

5.5. Conclusions

This paper presents a novel hybrid OF-NMPC that utilizes dynamic NARX model, and a first principles steady-state model. The NARX model is used for predicting dynamics and control products purities, and the first principles steady-state model is used to capture and avoid tower flooding. To this end, the OF-NMPC strategy shows its superior performance in improving ethylene product composition control and avoid tower flooding when compared to the PI control strategy, which is currently used in the tower. The OF-NMPC strategy performance is improved by adding a bias to the steady state model that considers whether liquid is being accumulated or depleted in the tower. Also, control banding is included into the OF-NMPC to further improve the control of the tower and avoid flooding. The hybrid model based OF-NMPC control strategy is shown to perform better than the current PI control strategy both in terms of maintaining product purity and achieve free flooding control.

Acknowledgements

This work was supported by NOVA Chemicals Corporation, Ontario Graduate Scholarship (OGS), and McMaster Advanced Control Consortium (MACC). Their support is greatly acknowledged.

References

Abedi, A., 2007. ECONOMIC ANALYSIS OF A NEW GAS TO ETHYLENE TECHNOLOGY. Texas A&M University.

Ammar Taqvi, S.A., Zabiri, H., Tufa, L.D., Fatima, S.A., Shah Maulud, A.H., 2019. Distillation Column: Review of Major Disturbances, in: Proceedings - 9th IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2019. Institute of Electrical and Electronics Engineers Inc., pp. 168–171. https://doi.org/10.1109/ICCSCE47578.2019.9068539

Bachnas, A.A., Tóth, R., Ludlage, J.H.A., Mesbah, A., 2014. A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study. J. Process Control. https://doi.org/10.1016/j.jprocont.2014.01.015

Badwe, A.S., Patwardhan, R.S., Shah, S.L., Patwardhan, S.C., Gudi, R.D., 2010. Quantifying the impact of model-plant mismatch on controller performance. J. Process Control 20, 408–425. https://doi.org/10.1016/j.jprocont.2009.12.006

Balaji, V., Rajaji, L., 2013. Comparative Study of PID and MPC Controller Using Labview. Int. J. Adv. Res. Electr. Electron. Instrum. Energy 5545–5550.

Bequette, B.W., 2007. Non-linear model predictive control: A personal retrospective. Can. J. Chem. Eng. https://doi.org/10.1002/cjce.5450850403

Chen, Z., Henson, M.A., Belanger, P., Megan, L., 2010. Nonlinear model predictive control of high purity distillation columns for cryogenic air separation. IEEE Trans. Control Syst. Technol. 18, 811–821. https://doi.org/10.1109/TCST.2009.2029087

Dr. James B. Riggs, 2000. Comparison of Advanced Distillation Control Methods, Final Technical Report. Albuquerque, NM. https://doi.org/10.2172/780447

Efheij, H., Albagul, A., Albraiki, N.A., 2019. Comparison of Model Predictive Control and PID Controller in Real Time Process Control System, in: 19th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, STA 2019. Institute of Electrical and Electronics Engineers Inc., pp. 64–69. https://doi.org/10.1109/STA.2019.8717271

Emerson Process Management, 2009. Distillation Column Flooding Diagnostics with Intelligent Differential Pressure Transmitter.

Faanes, A., Skogestad, S., 2005. Offset-free tracking of model predictive control with model mismatch: Experimental results. Ind. Eng. Chem. Res. 44, 3966–3972. https://doi.org/10.1021/ie049422y

Forbes, M.G., Patwardhan, R.S., Hamadah, H., Gopaluni, R.B., 2015. Model predictive controlin industry: Challenges and opportunities, in: IFAC-PapersOnLine. Elsevier, pp. 531–538. https://doi.org/10.1016/j.ifacol.2015.09.022

Foss, B.A., Cong, S.-B., 1999. Nonlinear MPC Based on Multi-Model for Distillation Columns. IFAC Proc. Vol. 32, 6944–6949. https://doi.org/10.1016/s1474-6670(17)57185-2

Fuentes, C., Luyben, W.L., 1983. Control of High-Purity Distillation Columns. Ind. Eng. Chem. Process Des. Dev. 22, 361–366. https://doi.org/10.1021/i200022a004

Ghosh, D., Hermonat, E., Mhaskar, P., Snowling, S., Goel, R., 2019. Hybrid Modeling Approach Integrating First-Principles Models with Subspace Identification. Ind. Eng. Chem. Res. 58, 13533–13543. https://doi.org/10.1021/acs.iecr.9b00900

González, A.H., Adam, E.J., Marchetti, J.L., 2008. Conditions for offset elimination in state

space receding horizon controllers: A tutorial analysis. Chem. Eng. Process. Process Intensif. 47, 2184–2194. https://doi.org/10.1016/j.cep.2007.11.011

Hassanpour, H., Corbett, B., Mhaskar, P., 2020. Integrating dynamic neural network models with principal component analysis for adaptive model predictive control. Chem. Eng. Res. Des. 161, 26–37. https://doi.org/10.1016/j.cherd.2020.03.031

Huang, R., Biegler, L.T., Patwardhan, S.C., 2010. Fast offset-free nonlinear model predictive control based on moving horizon estimation. Ind. Eng. Chem. Res. 49, 7882–7890. https://doi.org/10.1021/ie901945y

Jalanko, M., Sanchez, Y., Mahalec, V., Mhaskar, P., 2021. Adaptive system identification of industrial ethylene splitter: A comparison of subspace identification and artificial neural networks. Comput. Chem. Eng. 147. https://doi.org/10.1016/j.compchemeng.2021.107240

Kister, H.Z., 1990. Distillation Operation.

Maeder, U., Morari, M., 2010. Offset-free reference tracking with model predictive control. Automatica 46, 1469–1476. https://doi.org/10.1016/j.automatica.2010.05.023

Meenakshi, S., Almusthaliba, A., Vijayageetha, V., 2013. MIMO Identification and Controller design for Distillation Column. Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng. 1, 44–48.

Muske, K.R., Badgwell, T.A., 2002. Disturbance modeling for offset-free linear model predictive control. J. Process Control 12, 617–632. https://doi.org/10.1016/S0959-1524(01)00051-8

Patel, N., Nease, J., Aumi, S., Ewaschuk, C., Luo, J., Mhaskar, P., 2020. Integrating Data-Driven Modeling with First-Principles Knowledge. Ind. Eng. Chem. Res. 59, 5103–5113. https://doi.org/10.1021/acs.iecr.0c00418

Psichogios, D.C., Ungar, L.H., 1992. A hybrid neural network-first principles approach to process modeling. AIChE J. 38, 1499–1511. https://doi.org/10.1002/aic.690381003

Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. Control Eng. Pract. 11, 733–764. https://doi.org/10.1016/S0967-0661(02)00186-7

Ramesh, K., Abd Shukor, S.R., Aziz, N., 2009. Nonlinear model predictive control of a distillation column using NARX model, in: Computer Aided Chemical Engineering. Elsevier B.V., pp. 1575–1580. https://doi.org/10.1016/S1570-7946(09)70653-4

Ravi Sriniwas, G., Arkun, Y., Chien, I.L., Ogunnaike, B.A., 1995. Nonlinear identification and control of a high-purity distillation column: a case study. J. Process Control 5, 149–162. https://doi.org/10.1016/0959-1524(95)97302-9

Shariff, H.M., Fazalul Rahiman, M.H., Tajjudin, M., 2013. Nonlinear system identification: Comparison between PRBS and Random Gaussian perturbation on steam distillation pilot plant, in: Proceedings - 2013 IEEE 3rd International Conference on System Engineering and Technology, ICSET 2013. pp. 269–274. https://doi.org/10.1109/ICSEngT.2013.6650183

Su, H.-T., Bhat, N., Minderman, P.A., McAvoy, T.J., 1992. Integrating Neural Networks with First Principles Models for Dynamic Modeling. IFAC Proc. Vol. 25, 327–332. https://doi.org/10.1016/s1474-6670(17)51013-7

Tian, X., Wang, P., Huang, D., Chen, S., 2014. Offset-free multistep nonlinear model predictive control under plant-model mismatch. Int. J. Adapt. Control Signal Process. 28, 444–463. https://doi.org/10.1002/acs.2367

Tsen, A.Y. Di, Jang, S.S., Wong, D.S.H., Joseph, B., 1996. Predictive control of quality in batch polymerization using hybrid ANN models. AIChE J. 42, 455–465. https://doi.org/10.1002/aic.690420215

Valancia-Palomo, G.; Rossiter, J.A., 2007. Comparison between an auto-tuned PI controller, a predictive controller and a predictive functional controller in elementary dynamic systems https://www.researchgate.net/publication/229006364_Comparison_between_an_auto-tuned_PI_controller_a_predictive_controller_and_a_predictive_functional_controller_in_elementary_dynamic_systems/citations (accessed 5.8.21).

Wu, Z., Rincon, D., Christofides, P.D., 2020. Incorporating Structural Process Knowledge in Recurrent Neural Network Modeling of Nonlinear Processes, in: Proceedings of the American Control Conference. Institute of Electrical and Electronics Engineers Inc., pp. 2413–2418. https://doi.org/10.23919/ACC45564.2020.9147519

Xiong, W., Chen, L., Liu, F., Xu, B., 2014. Multiple model identification for a high purity distillation column process based on em algorithm. Math. Probl. Eng. 2014. https://doi.org/10.1155/2014/712682

Yousefi, M., Gopaluni, R.B., Loewen, P.D., Forbes, M.G., Dumont, G.A., Backstrom, J., 2015. Impact of model plant mismatch on performance of control systems: An application to

paper machine control. Control Eng. Pract. 43, 59–68. https://doi.org/10.1016/j.conengprac.2015.07.005

Zhang, D., Del Rio-Chanona, E.A., Petsagkourakis, P., Wagner, J., 2019. Hybrid physics-based and data-driven modeling for bioprocess online simulation and optimization. Biotechnol. Bioeng. 116, 2919–2930. https://doi.org/10.1002/bit.27120

Chapter 6: Concluding Remarks

This chapter highlights the contributions and key findings of the thesis and directions for future research.

6.1. Contributions and Key Findings

The focus of the thesis is to improve the process automation of chemical engineering applications at the planning and control levels. Chapters 2 and 3 consider the impact of uncertainty on production planning. While chapters 4 and 5 consider advanced modeling and control strategies. The contributions and key findings of Chapter 2 can be summarized as follows:

- Develop a methodology that utilizes joint chance constrained formulation to allow the production planner to decide the certainty which they will produce on-spec products versus the cost of re-manufacturing the product if it is not on-spec.
- Adapt the supply-demand pinch algorithm to solve large-scale production planning problems under uncertainty in components qualities in efficient times.
- The supply-demand pinch algorithm reduces the computational times required to compute the optimal solutions by at least 5-fold compared to the full-space model.
- The supply-demand pinch algorithm computes slightly better optimal solutions compared to the full-space model for large-scale problems due to the incapability of commercial solvers to close the optimality gap for such large-scale problems.
- The supply-demand pinch algorithm solutions have a lower number of blending recipes for each product along the planning horizon compared to the full-space model solutions.

Chapter 3 contributions and key findings can be summarized as follows:

- Develop a methodology that utilizes rolling horizon framework and loss function formulation to maximize the refinery's ability to satisfy uncertain products demands and maximize its profit. Also, the model developed considers time-varying uncertainty in products demands to capture real-world scenarios where new information about the future uncertain demand is available as we move forward in time.

- Develop a novel supply-demand pinch algorithm to solve the production planning problem under uncertainty in products demands using linear and nonlinear blending rules in efficient times.
- Under linear blending rules, the supply-demand pinch algorithm reduces the computational times required to compute the optimal solutions by at least 5- to 30-fold compared to the full-space model.
- Under nonlinear blending rules, the supply-demand pinch algorithm allows using a local solver to reduce computation times by 2000- to 3000-fold compared to the full-space model, while still achieving solutions within 0.04% from the full-space model solutions.

Chapter 4 contributions and key findings can be summarized as follows:

- Compare three different system identification methods (Subspace identification, NARX, and RNN) for the purpose of developing a dynamic data-driven model of industrial ethylene splitter using closed-loop data.
- Propose two adaptive modeling schemes to improve the model's prediction capability by incorporating more recent data into the training approaches.
- The subspace identification method outperforms neural network-based methods due to its capability of extrapolating, while neural network-based methods are prone to overfitting and face issues in generalizing to new process behaviour.
- The subspace identification method requires significantly less time to optimize the hyperparameters (~ 6 minutes) compared to neural network-based methods (~ 28 hours).

Chapter 5 contributions and key findings can be summarized as follows:

- Develop a novel hybrid model based OF-NMPC that utilizes a dynamic NARX model to predict process dynamics and control products purities, and a first principles steady-state model to capture and avoid tower flooding.
- Demonstrate the effectiveness of the proposed control strategy to improve the control strategy of industrial ethylene splitter and avoid flooding on an Aspen simulation model which is shown to capture the key traits of the real plant.

- Improve the performance of the steady-state model used to capture tower flooding by using some heuristic to consider whether liquid is being accumulated or depleted in the tower, include control banding concept to ensure free flooding control.
- The proposed control strategy is shown to perform better than the current PI control strategy currently used in terms of maintaining ethylene product purity and achieving free flooding control.

6.2. Future Research

In this section, directions for future work in both the production planning and control are outlined. For the production planning under uncertainty:

- In chapters 2 and 3, we consider the production planning under uncertainty in components qualities and products demands separately. Future work could consider developing a single methodology that can consider both uncertainty in components qualities and products demands combined.
- In chapters 2 and 3, we show that employing the supply-demand pinch algorithm can greatly reduce execution times for production planning of gasoline blend under uncertainty. Future work could investigate the effectiveness of applying the supply-demand pinch algorithm in solving the refinery planning problem under uncertainties in crude qualities and products demands. Refinery planning problems are much larger than gasoline blend planning problem and the incorporation of uncertainty can further increase the complexity of these models making them hard to solve using commercial solvers. Therefore, the work introduced in this thesis can be a first step toward solving refinery planning problems under uncertainty.

For the advanced modeling and control:

- In chapter 4, we consider the use of different data-driven model on closed-loop data using direct method. The main problem of the closed loop system is the existence of correlation between the input and disturbances which can impact the accuracy of our identified model. Future work can consider comparing these different data-driven models using open-loop data or some other methods for closed loop identification such as indirect method, joint input output method, or two stage method. One issue with using open-loop data from the real process is the cost associated with running the plant in

open-loop to generate rich data. Some of the future work can consider building a data-driven model from open-loop data generated from the simulator since such data can be generated with no cost and use closed-loop data obtained from the real plant to modify the simulator data-driven model accordingly to match the real plant process. Another potential future work can investigate the different in performance of the three different-data driven models when integrated with first-principles knowledge.

- In chapter 5, we propose a hybrid model based OF-NMPC to improve the control strategy of industrial ethylene splitter that faces flooding issues. The proposed control strategy utilizes NARX model that uses feed disturbance measurements to predict key controlled variables. One future research direction is to incorporate a disturbance model into our proposed OF-NMPC that uses upstream processes information to improve input disturbance used in the NARX model. This should improve the NARX model capability to product future controlled variables based on the proposed manipulated variables moves, thus improve process control. Some of future work can consider implementing feedforward control strategy that manipulates the feed flow disturbance to avoid flooding and compare the effectiveness of such control strategy to our proposed strategy. Another potential future work can investigate the use of multiple first principles steady-state models for different operation patterns as opposed to the use of a single model with fixed parameters. The idea is to modify the parameters of the steady-state model by running Aspen Plus under different operations and use the parameters that correspond to the current process operations.

Appendix A: Supporting Information for Chapter 2

Table A.1. Components data: Cost, initial inventory, inventory limits, properties, supply rates for Ex. 1 & 2

Components	ALK	BUT	HCL	HCN	LCN	LNP	RFT
Cost ($/bbl)	30	12	20	22	25	20	25
Initial Inventory (kbbl)	30	20	20	10	30	20	50
Minimum Inventory (kbbl)	5	5	5	5	5	5	5
Maximum Inventory (kbbl)	150	75	50	50	150	100	150
ARO (%vol aromatics)	0	0	0	25	18	2.974	74.9
BEN (%vol benzene)	0	0	0	0.5	1	0.595	7.5
MON	93.7	90	79.8	75.8	81.6	66	90.8
OLF (%vol olefin)	0	0	0	14	27	0	0
RON	94	92.8	82.3	86.7	92.2	67.8	102
RVP (psi)	5.15	138	22.335	2.378	13.876	19.904	3.622
SPG	0.703	0.584	0.695	0.791	0.744	0.677	0.818
SI (%vol sulfur)	0	0	0	0.485	0.078	0.013	0
Supply Rate (kbbl/day) (Ex. 1 &2)	7	7	20	15	30	10	45

Table A.2. Components data: Supply rates along the planning horizon for Ex. 3 - 5

Components	**ALK**	**BUT**	**HCL**	**HCN**	**LCN**	**LNP**	**RFT**
Period	kbbl/day						
1	25	7	0	3	27	20	45
2	25	7	0	3	27	20	45
3	25	7	0	3	27	20	45
4	20	5	3	5	25	18	40
5	15	3	7	9	20	22	35
6	15	3	7	9	20	22	35
7	15	3	7	9	20	22	35
8	20	5	3	5	25	18	40
9	20	5	3	5	25	18	40
10	25	7	0	3	27	22	45
11	25	7	0	3	27	22	45
12	25	7	0	3	27	22	45
13	20	5	3	5	25	18	40
14	20	5	3	5	25	18	40

Table A.3. Products storage tanks data

Products	Storable products	Min. hold up (kbbl)	Max. hold up (kbbl)	Max. delivery rate (kbbl/hour)	Initial inventory (Ex.1 &2)	Initial inventory (Ex.3 - 5)	Initial product
Tank1	U87	10	70	10	10	40	U87
Tank2	U91	10	70	10	10	70	U91
Tank3	U93	10	70	10	20	30	U93
Tank4	U87, U91, U93	0	40	10	20	30	U87
Tank5	U87, U91, U93	0	40	10	10	40	U91
Tank6	U87, U91, U93	0	40	10	10	30	U91

Table A.4. Products data: properties

Products	Product 1	Product 2	Product 3
ARO (%vol aromatics) [min, max]	[0,60]	[0,60]	[0,60]
BEN (%vol benzene) [min, max]	[0,5.9]	[0,5.9]	[0,5.9]
MON [min, max]	[81.5,200]	[85.7,200]	[87.5,200]
OLF (%vol olefin) [min, max]	[0,24.2]	[0,24.2]	[0,24.2]
RON [min, max]	[90.4,200]	[92.5,200]	[95.5,200]
RVP (psi) [min, max]	[0,15.6]	[0,15.6]	[0,15.6]
SPG [min, max]	[0.73,0.81]	[0.73,0.81]	[0.73,0.81]
SI (%vol sulfur) [min, max]	[0,0.1]	[0,0.1]	[0,0.1]

Table A.5. Demand profile (kbbl)

Example	Product	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	**U87**	50	50	60	-	-	-	-	-	-	-	-	-	-	-
	U91	50	70	80	-	-	-	-	-	-	-	-	-	-	-
	U93	10	20	30	-	-	-	-	-	-	-	-	-	-	-
2	**U87**	50	80	40	70	-	-	-	-	-	-	-	-	-	-
	U91	50	80	40	45	-	-	-	-	-	-	-	-	-	-
	U93	10	30	15	20	-	-	-	-	-	-	-	-	-	-
3	**U87**	60	50	50	80	50	60	60	50	75	50	50	50	80	100
	U91	50	80	70	30	50	0	40	30	30	50	40	40	30	50
	U93	30	30	0	0	40	40	0	35	30	0	0	40	30	40
4	**U87**	70	70	50	70	70	60	60	60	50	70	120	0	50	70
	U91	50	50	50	30	30	50	50	30	30	50	50	30	30	50
	U93	30	30	45	30	40	0	0	35	30	0	30	35	0	30
5	**U87**	100	70	80	100	40	30	40	110	0	50	70	100	0	50
	U91	50	80	70	50	30	30	30	50	30	30	30	35	30	30
	U93	30	30	45	30	0	30	30	30	30	0	0	30	30	30

Appendix B: Supporting Information for Chapter 3

Table B.1. Detailed Results for the four different solutions strategies for the cases with nonlinear blending rules

	Strategy 1			Strategy 2			Strategy 3			Strategy 4		
First Iteration	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
Revenue (thousands $)	67597.2	67542.9	67555.8	67597.2	67577.0	67555.8	67578.3	67525.2	67537.4	67578.3	67525.2	67537.4
Cost (thousands $)	45037.6	44979.5	44996.1	45037.6	45018.6	44996.1	45026.9	44969.4	44985.6	45026.9	44969.4	44985.6
Profit (thousands $)	22559.5	22563.5	22559.7	22559.5	22558.3	22559.7	22551.4	22555.8	22551.8	22551.4	22555.8	22551.8
Execution time (s)	3648.5	3710.1	3619.9	10.2	20.3	13.3	3601.2	3601.4	3602.1	1.4	1.5	1.4
Rolling Horizon	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
Revenue (thousands $)	67396.5	67562.7	67504.1	67396.5	67374.5	67500.0	67396.5	67546.4	67489.5	67396.5	67547.5	67481.5
Cost (thousands $)	44886.4	45001.9	44939.2	44886.4	44853.9	44935.6	44891.8	44992.7	44932.1	44891.5	44993.2	44924.4
Profit (thousands $)	22510.0	22560.8	22564.9	22510.1	22520.6	22564.4	22504.7	22553.8	22557.5	22505.0	22554.2	22557.0
Execution time (s)	37238.5	34981.9	37343.3	89.4	148.7	165.0	25223.6	10827.8	10839.7	9.6	13.1	10.5
First Iteration	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6
Revenue (thousands $)	76441.9	76347.9	76450.0	76437.9	76334.2	76446.4	76458.1	76333.7	76467.0	76458.1	76333.7	76467.0
Cost (thousands $)	51072.2	51009.9	51068.2	51068.1	50995.6	51064.5	51089.9	50996.0	51086.8	51089.9	50996.0	51086.8
Profit (thousands $)	25369.7	25338.0	25381.8	25369.8	25338.7	25381.9	25368.2	25337.6	25380.3	25368.2	25337.6	25380.3
Execution time (s)	3629.8	7200.2	3623.5	52.1	26.8	22.4	5.4	3.9	7.2	2.9	1.9	1.5
Rolling Horizon	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6	Case 4	Case 5	Case 6
Revenue (thousands $)	76450.9	76264.0	76441.0	76344.5	76361.3	76428.4	76523.9	76344.1	76475.2	76443.7	76357.9	76478.1
Cost (thousands $)	51113.8	50926.1	51068.7	51008.3	51006.1	51056.3	51190.2	50994.1	51099.4	51116.1	51006.2	51102.9
Profit (thousands $)	25337.1	25337.9	25372.3	25336.3	25355.3	25372.1	25333.7	25350.1	25375.8	25327.6	25351.7	25375.3
Execution time (s)	30980.2	39974.4	47822.7	159.8	141.2	142.5	5453.9	14484.3	14845.0	15.7	12.9	13.0
First Iteration	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9
Revenue (thousands $)	69938.1	69976.0	69958.8	69939.4	69975.8	69958.8	69928.5	69965.5	69947.7	69928.5	69965.5	69975.6
Cost (thousands $)	46634.9	46672.9	46658.3	46635.6	46672.8	46658.3	46629.8	46667.3	46652.4	46629.8	46667.3	46682.0
Profit (thousands $)	23303.2	23303.1	23300.5	23303.7	23303.0	23300.5	23298.7	23298.2	23295.3	23298.7	23298.2	23293.6
Execution time (s)	7245.7	3629.5	3629.5	13.3	14.4	91.1	3613.6	3633.6	3728.3	2.0	2.4	3.8
Rolling Horizon	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9	Case 7	Case 8	Case 9
Revenue (thousands $)	70013.2	69971.1	69947.0	69794.5	69970.2	69860.5	70021.9	69965.2	70003.5	70025.2	69953.7	70004.0
Cost (thousands $)	46724.6	46666.8	46661.8	46539.4	46666.3	46588.7	46737.8	46663.5	46712.2	46735.4	46657.0	46712.6
Profit (thousands $)	23288.6	23304.2	23285.2	23255.0	23303.8	23271.8	23253.0	23289.9	23292.8	23289.8	23296.7	23291.4
Execution time (s)	36845.7	59489.1	46524.6	107.1	163.8	193.6	11419.2	12453.1	11689.3	26.9	16.3	22.3

Table B.2. Components costs and products prices

Component/Product	ALK	BUT	HCL	HCN	LCN	LNP	RFT	U87	U91	U93
Price (thousands $/kbbl)	30	12	20	22	23	18	25	32	35	37

Table B.3. Demand contracted data

Example	Product	Period	Contracted			Minimum Delivery Rate D_{order}^{max} (kbbl/h)	Maximum Delivery Rate D_{order}^{max} (kbbl/h)
			1	2	3		
Order							
O1	U87	1	60	80	60	1	10
O2	U87	2	50	60	70	1	10
O3	U87	3	40	60	60	1	10
O4	U87	4	40	80	70	1	10
O5	U87	5	50	70	40	1	10
O6	U87	6	50	65	50	1	10
O7	U87	7	40	60	50	1	10
O8	U87	8	50	85	70	1	10
O9	U87	9	55	80	60	1	10
O10	U87	10	40	35	30	1	10
O11	U87	11	40	30	40	1	10
O12	U87	12	50	40	40	1	10
O13	U87	13	80	35	50	1	10
O14	U87	14	40	40	60	1	10
O15	U91	1	50	50	40	1	10
O16	U91	2	60	40	50	1	10
O17	U91	3	50	50	60	1	10
O18	U91	4	30	30	70	1	10
O19	U91	5	40	40	30	1	10
O20	U91	6	40	50	30	1	10
O21	U91	7	40	40	30	1	10

O22	U91	8	30	70	50	1	10
O23	U91	9	30	80	50	1	10
O24	U91	10	40	30	20	1	10
O25	U91	11	30	35	30	1	10
O26	U91	12	40	35	50	1	10
O27	U91	13	30	30	30	1	10
O28	U91	14	30	30	30	1	10
O29	U93	1	30	30	40	1	10
O30	U93	2	30	30	40	1	10
O31	U93	3	30	30	40	1	10
O32	U93	4	30	35	60	1	10
O33	U93	5	40	30	20	1	10
O34	U93	6	30	30	30	1	10
O35	U93	7	30	30	20	1	10
O36	U93	8	35	40	30	1	10
O37	U93	9	30	45	50	1	10
O38	U93	10	30	30	20	1	10
O39	U93	11	30	30	20	1	10
O40	U93	12	40	35	30	1	10
O41	U93	13	30	45	20	1	10
O42	U93	14	30	30	30	1	10

Table B.4. Product and component tank data

Product or Component Tank	Initial Product	Initial Stock V^{ini} (kbbl)	Supply rate of components (kbbl/day)			Min. Capacity V^{min} (kbbl)	Max. Capacity V^{max} (kbbl)	Storable Products (Set **JP**)	Min. Delivery Rate D_{pr}^{min} (kbbl/h)	Max. Delivery Rate D_{pr}^{max} (kbbl/h)
			Case 1-3	Case 4-6	Case 7-9					
Tk1	P1	70	-	-	-	10	70	P1	2	10
Tk2	P2	40	-	-	-	10	70	P2	2	10
Tk3	P3	40	-	-	-	10	70	P3	2	10
Tk4	P1	10	-	-	-	0	40	P1-P3	1	7
Tk5	P2	10	-	-	-	0	40	P1-P3	1	7
Tk6	P2	10	-	-	-	0	40	P1-P3	1	7
ALK	ALK	30	19.5	22.5	20.25	5	150	-	-	-
BUT	BUT	20	5.2	6	5.4	5	75	-	-	-
HCL	HCL	20	7.8	9	8.1	5	50	-	-	-
HCN	HCN	10	7.8	9	8.1	5	50	-	-	-
LCN	LCN	30	20.8	24	21.6	5	150	-	-	-
LNP	LNP	20	19.5	22.5	20.25	5	100	-	-	-
RFT	RFT	50	49.4	57	51.3	5	150	-	-	-

Table B.5. Component's qualities and products qualities specifications for nonlinear blending cases

Property	Blend Components						
	ALK	BUT	HCL	HCN	LCN	LNP	RFT
ARO (%vol aromatics)	0	0	0	25	18	2.974	74.9
BEN (%vol benzene)	0	0	0	0.5	1	0.595	7.5
MON	93.7	90	79.8	75.8	81.6	66	90.8
RON	95	93.8	82.3	86.7	93.2	67.8	103
OLF (%vol olefin)	0	0	0	14	27	0	0
RVP (psi)	5.15	138	22.335	2.378	13.876	19.904	3.622
SPG	0.703	0.584	0.695	0.791	0.744	0.677	0.818
SUL (%vol sulfur)	0	0	0	0.485	0.078	0.013	0
Property	Product Specifications [Min, Max]						
	U87			U91		U93	
ARO (%vol aromatics)	[0,60]			[0,60]		[0,60]	
BEN (%vol benzene)	[0,5.9]			[0,5.9]		[0,5.9]	
MON	[81.5,200]			[85.5,200]		[87.5,200]	
RON	[91.4,200]			[94.5,200]		[97.5,200]	
OLF (%vol olefin)	[0,24.2]			[0,24.2]		[0,24.2]	
RVP (psi)	[0,15.6]			[0,15.6]		[0,15.6]	
SPG	[0.73,0.81]			[0.73,0.81]		[0.73,0.81]	
SI (%vol sulfur)	[0,0.1]			[0,0.1]		[0,0.1]	

Table B.6. Component's qualities and products qualities specifications for linear blending cases

Property	Blend Components						
	ALK	BUT	HCL	HCN	LCN	LNP	RFT
ARO (%vol aromatics)	0	0	0	25	18	2.974	74.9
BEN (%vol benzene)	0	0	0	0.5	1	0.595	7.5
MON INDEX	108.53	103.24	91.3	87.3	93.1	77.5	104.36
RON INDEX	110.45	108.67	93.8	98.74	107.8	79.3	123.04
OLF (%vol olefin)	0	0	0	14	27	0	0
RVP INDEX	2.25	11.47	4.65	1.54	3.68	4.4	1.89
SPG	0.703	0.584	0.695	0.791	0.744	0.677	0.818
SUL (%vol sulfur)	0	0	0	0.485	0.078	0.013	0
Property	Product Specifications [Min, Max]						
	U87		U91		U93		
ARO (%vol aromatics)	[0,60]		[0,60]		[0,60]		
BEN (%vol benzene)	[0,5.9]		[0,5.9]		[0,5.9]		
MON INDEX	[93,455.79]		[97.42,455.79]		[99.81,455.79]		
RON INDEX	[105.21,455.79]		[109.7,455.79]		[114.24,455.79]		
OLF (%vol olefin)	[0,24.2]		[0,24.2]		[0,24.2]		
RVP INDEX	[0,3.9]		[0,3.9]		[0,3.9]		
SPG	[0.73,0.81]		[0.73,0.81]		[0.73,0.81]		
SI (%vol sulfur)	[0,0.1]		[0,0.1]		[0,0.1]		

Table B.7. Minimum inventory for each period

Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
U87	10	10	10	10	10	10	10	10	10	10	10	10	10	40
U91	10	10	10	10	10	10	10	10	10	10	10	10	10	40
U93	10	10	10	10	10	10	10	10	10	10	10	10	10	40

Table B.8. Blender capacities

Blender	A
Maximum blending rate (kbbl/day)	9
Minimum blending rate (kbbl/day)	5
Minimum blend allowed for one product (kbbl/day)	30
Minimum waiting (idle) time before blending new product (h)	1
Minimum running time of blender for a product (h)	6

Table B.9. Penalty coefficients for inventory infeasibility on the components and products sides

Component/Product	Period	values
Components penalty	1-14	1.00E+09
Products Penalty	1	1.80E+08
	2	1.70E+08
	3	1.60E+08
	4	1.50E+07
	5	1.40E+07
	6	1.30E+07
	7	1.20E+07
	8	1.10E+06
	9	1.00E+05
	10	9.00E+04
	11	8.00E+04
	12	5.00E+04
	13	1.00E+04
	14	5.00E+03

Table B.10. The coefficient of the four polynomial functions that approximate the standard loss function

	L_1 (-∞,-3]	L_2 (-3,0]	L_3 (0,3]	L_4 (3,∞]
a_i	0	0.39893	0.398924	0
b_i	-1	-0.5004	-0.49952	0
c_i	0	0.19653	0.196346	0
d_i	0	-0.0084	0.008484	0
e_i	0	-0.028	-0.02795	0
f_i	0	-0.0078	0.007732	0
g_i	0	-0.0007	-0.0007	0

Table B.11. Contracted demand and additional demand simulation for case # 1

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	60.00	50.00	40.00	40.00	50.00	50.00	40.00	50.00	55.00	40.00	40.00	50.00	80.00	40.00
	U91	50.00	60.00	50.00	30.00	40.00	40.00	40.00	30.00	30.00	40.00	30.00	40.00	30.00	30.00
	U93	30.00	30.00	30.00	30.00	40.00	30.00	30.00	35.00	30.00	30.00	30.00	40.00	30.00	30.00
Additional Mean Demand	U87	12.00	10.00	8.00	8.00	10.00	10.00	8.00	10.00	11.00	8.00	8.00	10.00	16.00	8.00
	U91	10.00	12.00	10.00	6.00	8.00	8.00	8.00	6.00	6.00	8.00	6.00	8.00	6.00	6.00
	U93	6.00	6.00	6.00	6.00	8.00	6.00	6.00	7.00	6.00	6.00	6.00	8.00	6.00	6.00
Iteration 1	U87	13.66	8.74	7.45	7.48	8.65	9.85	5.22	10.22	11.45	7.92	8.37	10.26	16.63	7.62
	U91	9.81	10.52	9.09	6.46	7.78	8.05	8.46	6.90	5.64	8.90	4.49	7.72	6.99	3.82
	U93	4.91	6.34	3.78	3.65	7.92	5.73	5.15	7.47	6.10	9.20	5.63	7.46	5.76	5.52
Iteration 2	U87		10.65	5.75	9.19	10.18	8.02	5.42	11.61	10.88	8.07	7.84	8.12	16.60	7.41
	U91		10.68	9.49	6.50	8.60	6.98	10.17	6.85	1.86	8.50	3.91	7.04	7.10	3.45
	U93		4.47	2.44	1.79	7.46	6.42	6.36	6.04	5.59	11.39	4.85	6.01	4.23	4.83
Iteration 3	U87			7.65	8.75	10.35	8.00	6.91	12.17	10.67	9.27	7.57	8.86	14.60	6.72
	U91			9.30	6.52	5.98	6.05	9.99	8.74	2.20	8.07	4.06	7.04	8.27	4.13
	U93			2.21	2.03	7.50	6.36	7.39	6.84	4.39	11.32	4.90	5.18	3.59	4.74
Iteration 4	U87				7.03	10.63	7.67	7.05	14.07	9.38	9.60	7.75	8.62	14.30	5.94
	U91				7.42	4.44	5.54	10.56	6.73	2.90	9.15	5.29	5.07	6.52	4.69
	U93				2.30	6.20	6.37	7.34	5.32	5.30	11.49	4.67	5.11	3.85	4.45
Iteration 5	U87					7.63	7.01	6.39	13.65	8.37	9.46	6.64	8.38	15.76	5.17
	U91					4.85	4.75	12.02	7.21	1.57	8.61	5.95	5.19	5.41	3.98
	U93					6.20	6.37	7.34	5.32	5.30	11.49	4.67	5.11	3.85	4.45
Iteration 6	U87						7.03	8.87	13.52	8.00	9.86	6.05	6.90	16.43	5.29
	U91						4.75	12.02	7.21	1.57	8.61	5.95	5.19	5.41	3.98
	U93						6.81	9.89	5.88	4.94	12.05	3.80	4.89	4.91	4.55
Iteration 7	U87							8.87	13.52	8.00	9.86	6.05	6.90	16.43	5.29
	U91							11.32	7.38	1.06	7.53	5.32	4.73	5.44	4.34
	U93							10.28	4.79	4.69	12.96	5.01	6.34	4.32	6.95
Iteration 8	U87								13.31	9.02	7.53	5.96	4.34	17.59	6.22
	U91								5.99	1.10	7.60	5.07	3.89	6.02	3.24
	U93								3.09	4.86	14.01	4.36	7.08	5.02	7.12
Iteration 9	U87									9.29	6.68	7.29	5.43	17.54	4.39
	U91									1.32	8.82	4.06	3.27	5.85	2.62
	U93									5.86	13.06	3.41	7.28	4.55	7.70
Iteration 10	U87										4.93	6.14	6.06	19.28	4.33
	U91										7.75	3.54	3.05	6.01	3.35
	U93										13.41	4.03	6.79	4.57	8.82
Iteration 11	U87											4.63	5.47	19.72	6.02
	U91											2.36	2.08	5.31	2.02
	U93											3.30	6.59	3.14	10.39
Iteration 12	U87												6.56	19.08	6.32
	U91												2.13	6.13	1.70
	U93												8.70	0.64	11.23
Iteration 13	U87													19.26	6.30
	U91													4.90	1.64
	U93													0.08	12.33
Iteration 14	U87														6.30
	U91														1.64
	U93														12.33

Table B.12. Contracted demand and additional demand simulation for case # 2

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	60.00	50.00	40.00	40.00	50.00	50.00	40.00	50.00	55.00	40.00	40.00	50.00	80.00	40.00
	U91	50.00	60.00	50.00	30.00	40.00	40.00	40.00	30.00	30.00	40.00	30.00	40.00	30.00	30.00
	U93	30.00	30.00	30.00	30.00	40.00	30.00	30.00	35.00	30.00	30.00	30.00	40.00	30.00	30.00
Additional Mean Demand	U87	12.00	10.00	8.00	8.00	10.00	10.00	8.00	10.00	11.00	8.00	8.00	10.00	16.00	8.00
	U91	10.00	12.00	10.00	6.00	8.00	8.00	8.00	6.00	6.00	8.00	6.00	8.00	6.00	6.00
	U93	6.00	6.00	6.00	6.00	8.00	6.00	6.00	7.00	6.00	6.00	6.00	8.00	6.00	6.00
Iteration 1	U87	12.55	8.25	8.75	7.79	11.30	10.24	8.45	10.91	9.47	8.60	9.17	9.55	16.35	8.12
	U91	11.66	12.26	11.21	5.66	7.51	7.20	10.23	5.28	8.45	6.22	4.70	7.45	7.20	6.44
	U93	5.79	6.65	6.06	7.41	7.48	6.39	5.96	6.75	5.76	5.98	6.36	10.18	5.27	4.13
Iteration 2	U87		9.69	8.17	8.22	11.07	9.20	8.40	9.85	9.15	7.68	10.94	8.61	16.79	8.52
	U91		10.62	11.47	4.94	6.78	6.77	9.25	4.70	8.84	6.22	4.46	7.59	8.83	5.51
	U93		7.61	6.32	6.85	8.95	4.47	5.23	6.69	5.93	8.18	4.68	9.02	4.73	4.04
Iteration 3	U87			8.01	7.16	8.99	9.46	7.41	9.76	9.66	7.60	11.62	9.89	17.04	8.75
	U91			10.64	3.99	7.75	5.67	10.05	5.31	9.67	7.50	4.51	9.73	8.70	6.07
	U93			4.70	5.39	9.43	5.99	3.64	6.55	6.13	8.85	5.82	10.08	5.09	3.77
Iteration 4	U87				6.76	8.69	8.25	6.99	9.45	8.74	8.28	11.24	10.22	15.47	8.61
	U91				4.76	6.64	5.17	9.65	6.55	9.13	7.42	4.71	8.16	9.09	6.01
	U93				7.60	9.79	7.39	4.49	6.30	6.00	8.17	5.02	9.92	4.96	2.22
Iteration 5	U87					8.50	7.67	6.14	10.53	8.49	7.76	9.45	10.81	17.42	9.98
	U91					9.46	5.18	11.27	6.63	9.46	6.85	4.37	9.75	7.72	4.61
	U93					9.79	7.39	4.49	6.30	6.00	8.17	5.02	9.92	4.96	2.22
Iteration 6	U87						6.95	5.54	13.78	7.88	8.32	11.35	8.75	16.80	9.24
	U91						5.18	11.27	6.63	9.46	6.85	4.37	9.75	7.72	4.61
	U93						6.19	6.34	6.19	6.04	8.72	5.04	9.57	4.88	1.16
Iteration 7	U87							5.54	13.78	7.88	8.32	11.35	8.75	16.80	9.24
	U91							9.97	6.98	8.75	6.70	3.16	8.33	8.57	5.46
	U93							7.81	6.99	6.08	6.53	4.43	11.37	4.37	0.00
Iteration 8	U87								11.66	8.78	8.48	10.73	7.87	13.80	8.07
	U91								8.14	8.39	7.15	4.24	9.23	7.06	5.39
	U93								5.92	4.90	6.13	3.55	13.09	4.72	0.53
Iteration 9	U87									8.73	7.38	12.19	8.15	14.15	6.82
	U91									9.62	6.95	5.77	8.93	5.92	4.65
	U93									4.58	6.16	3.57	12.17	5.32	0.00
Iteration 10	U87										7.17	11.53	7.47	14.31	8.05
	U91										6.72	4.82	8.98	6.13	3.29
	U93										7.49	4.49	12.77	5.56	0.56
Iteration 11	U87											11.61	7.55	14.89	7.99
	U91											3.69	8.37	6.46	2.78
	U93											5.24	13.40	5.55	0.00
Iteration 12	U87												8.24	15.90	7.88
	U91												9.51	6.26	1.19
	U93												12.08	4.93	0.00
Iteration 13	U87													16.08	8.77
	U91													5.18	0.00
	U93													5.22	0.00
Iteration 14	U87														8.77
	U91														0.00
	U93														0.00

Table B.13. Contracted demand and additional demand simulation for case # 3

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	60.00	50.00	40.00	40.00	50.00	50.00	40.00	50.00	55.00	40.00	40.00	50.00	80.00	40.00
	U91	50.00	60.00	50.00	30.00	40.00	40.00	40.00	30.00	30.00	40.00	30.00	40.00	30.00	30.00
	U93	30.00	30.00	30.00	30.00	40.00	30.00	30.00	35.00	30.00	30.00	30.00	40.00	30.00	30.00
Additional Mean Demand	U87	12.00	10.00	8.00	8.00	10.00	10.00	8.00	10.00	11.00	8.00	8.00	10.00	16.00	8.00
	U91	10.00	12.00	10.00	6.00	8.00	8.00	8.00	6.00	6.00	8.00	6.00	8.00	6.00	6.00
	U93	6.00	6.00	6.00	6.00	8.00	6.00	6.00	7.00	6.00	6.00	6.00	8.00	6.00	6.00
Iteration 1	U87	12.11	9.38	9.08	8.07	10.54	8.45	7.72	9.75	10.06	7.92	10.13	11.77	15.70	7.35
	U91	9.39	12.60	10.34	5.16	6.93	5.65	6.76	5.47	6.57	8.90	5.27	7.13	6.15	4.44
	U93	7.35	7.47	7.06	5.92	7.89	8.43	6.52	7.74	4.59	4.28	5.58	8.46	6.89	5.14
Iteration 2	U87		9.77	10.63	5.92	9.44	6.80	7.61	10.49	11.12	6.31	9.19	10.80	15.21	8.11
	U91		13.94	10.70	5.98	8.43	7.31	6.54	4.87	7.04	8.00	6.30	7.27	5.47	6.36
	U93		7.33	8.22	6.06	10.51	10.27	6.71	7.09	4.99	4.83	5.45	7.19	6.44	4.38
Iteration 3	U87			10.44	6.20	10.76	7.69	6.71	10.33	10.25	6.37	8.63	11.42	15.01	8.71
	U91			9.71	4.94	7.01	9.54	4.68	5.10	6.92	8.04	6.49	8.08	5.23	7.22
	U93			7.77	4.67	10.78	8.83	5.98	7.36	3.90	3.78	7.06	6.93	6.39	5.44
Iteration 4	U87				7.79	10.84	7.60	6.67	10.59	10.36	6.72	7.70	12.27	15.15	6.98
	U91				5.42	8.39	9.18	5.10	5.73	6.72	6.47	7.42	7.98	5.28	6.85
	U93				4.40	10.76	8.43	6.43	7.87	3.32	5.01	7.26	7.96	5.51	6.23
Iteration 5	U87					11.04	8.50	8.85	10.44	9.83	6.52	8.81	11.77	17.39	5.93
	U91					9.05	9.40	5.59	3.12	6.04	5.42	8.68	7.34	5.02	5.72
	U93					10.76	8.43	6.43	7.87	3.32	5.01	7.26	7.96	5.51	6.23
Iteration 6	U87						7.53	9.12	10.35	8.22	6.83	9.67	12.10	16.80	6.32
	U91						9.40	5.59	3.12	6.04	5.42	8.68	7.34	5.02	5.72
	U93						9.75	7.40	7.24	2.75	5.35	7.17	10.04	6.37	6.41
Iteration 7	U87							9.12	10.35	8.22	6.83	9.67	12.10	16.80	6.32
	U91							7.20	2.30	6.91	4.11	8.76	8.79	4.42	7.68
	U93							7.16	7.35	0.69	6.49	6.64	9.16	6.66	6.52
Iteration 8	U87								10.38	8.32	6.84	10.82	11.52	17.78	5.61
	U91								1.99	7.62	5.09	8.85	7.14	2.88	8.63
	U93								6.88	2.12	7.37	7.02	9.94	7.47	5.72
Iteration 9	U87									7.26	6.29	10.30	9.33	18.63	4.32
	U91									8.19	6.76	9.52	6.55	0.88	8.98
	U93									0.00	8.18	5.85	9.99	7.67	5.97
Iteration 10	U87										7.09	11.34	7.86	18.12	3.14
	U91										5.82	9.96	6.30	2.63	9.87
	U93										7.81	4.82	10.14	8.63	6.77
Iteration 11	U87											12.55	6.94	19.52	4.06
	U91											10.54	5.12	2.64	10.75
	U93											6.60	9.47	8.59	7.62
Iteration 12	U87												6.90	20.78	5.72
	U91												6.37	3.93	10.55
	U93												8.44	6.86	8.75
Iteration 13	U87													21.14	5.79
	U91													2.69	12.49
	U93													6.51	9.21
Iteration 14	U87														5.79
	U91														12.49
	U93														9.21

Table B.14. Contracted demand and additional demand simulation for case # 4

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	80.00	60.00	60.00	80.00	70.00	65.00	60.00	85.00	80.00	35.00	30.00	40.00	35.00	40.00
	U91	50.00	40.00	50.00	30.00	40.00	50.00	40.00	70.00	80.00	30.00	35.00	35.00	30.00	30.00
	U93	30.00	30.00	30.00	35.00	30.00	30.00	30.00	40.00	45.00	30.00	30.00	35.00	45.00	30.00
Additional Mean Demand	U87	16.00	12.00	12.00	16.00	14.00	13.00	12.00	17.00	16.00	7.00	6.00	8.00	7.00	8.00
	U91	10.00	8.00	10.00	6.00	8.00	10.00	8.00	14.00	16.00	6.00	7.00	7.00	6.00	6.00
	U93	6.00	6.00	6.00	7.00	6.00	6.00	6.00	8.00	9.00	6.00	6.00	7.00	9.00	6.00
Iteration 1	U87	15.68	11.13	11.12	15.99	13.92	13.17	11.52	17.80	17.87	6.38	7.64	9.91	8.47	8.15
	U91	9.43	8.41	8.72	8.52	8.31	7.57	4.94	14.04	13.91	7.80	7.02	6.43	7.21	8.13
	U93	6.30	6.11	4.15	7.94	5.54	6.50	5.40	7.50	9.25	6.12	4.89	6.18	8.86	6.94
Iteration 2	U87		11.24	11.91	16.51	14.74	12.42	10.09	18.97	17.82	5.52	7.74	9.77	9.57	7.42
	U91		7.62	9.68	8.08	6.97	7.59	6.73	15.07	15.43	6.84	6.00	8.90	5.81	6.42
	U93		6.19	4.26	6.54	5.50	6.53	5.96	5.57	8.93	6.16	5.16	7.02	9.09	7.47
Iteration 3	U87			13.43	16.21	15.84	11.69	9.78	20.48	17.70	6.70	9.05	10.28	10.67	7.69
	U91			8.05	5.82	4.81	7.99	8.87	14.79	13.81	7.17	5.24	8.51	4.74	7.74
	U93			4.32	5.56	6.05	6.69	6.25	5.04	7.52	4.58	5.29	5.49	8.13	6.98
Iteration 4	U87				16.23	17.07	12.08	7.64	20.67	19.24	8.40	9.61	10.81	11.54	7.40
	U91				7.78	6.32	7.17	10.72	13.72	14.05	6.72	5.18	8.31	5.17	8.11
	U93				5.83	4.59	7.11	4.59	7.13	7.33	3.29	6.54	5.59	7.27	7.44
Iteration 5	U87					18.03	12.40	7.59	20.67	19.15	9.74	8.46	10.90	12.18	6.58
	U91					5.11	6.80	10.06	13.05	14.40	6.19	4.40	8.63	3.78	7.13
	U93					4.59	7.11	4.59	7.13	7.33	3.29	6.54	5.59	7.27	7.44
Iteration 6	U87						11.78	5.51	20.15	18.06	9.65	9.63	11.30	12.48	5.56
	U91						6.80	10.06	13.05	14.40	6.19	4.40	8.63	3.78	7.13
	U93						7.82	4.52	7.46	8.52	2.87	7.27	7.51	7.21	7.52
Iteration 7	U87							5.51	20.15	18.06	9.65	9.63	11.30	12.48	5.56
	U91							10.45	13.28	16.48	6.25	5.46	9.33	2.04	6.79
	U93							4.15	7.35	7.31	3.75	5.67	7.60	7.20	8.07
Iteration 8	U87								20.31	18.76	9.99	9.90	11.15	12.41	6.91
	U91								14.22	17.75	8.14	6.47	8.26	2.07	5.35
	U93								9.29	8.28	4.37	6.66	5.35	5.12	8.14
Iteration 9	U87									18.84	10.83	11.17	11.58	11.28	6.11
	U91									17.84	7.44	5.64	9.85	0.49	3.91
	U93									8.53	3.66	8.24	4.81	4.67	8.87
Iteration 10	U87										11.21	10.20	10.11	11.70	8.28
	U91										8.68	4.94	9.98	0.98	4.06
	U93										2.48	9.50	3.38	1.60	8.94
Iteration 11	U87											10.95	10.40	11.13	7.38
	U91											5.42	8.48	1.38	3.88
	U93											9.97	4.60	1.55	9.62
Iteration 12	U87												8.78	12.59	7.23
	U91												6.87	1.30	5.95
	U93												5.15	0.57	9.84
Iteration 13	U87													13.64	9.32
	U91													0.25	5.52
	U93													0.11	10.13
Iteration 14	U87														9.32
	U91														5.52
	U93														10.13

Table B.15. Contracted demand and additional demand simulation for case # 5

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	80.00	60.00	60.00	80.00	70.00	65.00	60.00	85.00	80.00	35.00	30.00	40.00	35.00	40.00
	U91	50.00	40.00	50.00	30.00	40.00	50.00	40.00	70.00	80.00	30.00	35.00	35.00	30.00	30.00
	U93	30.00	30.00	30.00	35.00	30.00	30.00	30.00	40.00	45.00	30.00	30.00	35.00	45.00	30.00
Additional Mean Demand	U87	16.00	12.00	12.00	16.00	14.00	13.00	12.00	17.00	16.00	7.00	6.00	8.00	7.00	8.00
	U91	10.00	8.00	10.00	6.00	8.00	10.00	8.00	14.00	16.00	6.00	7.00	7.00	6.00	6.00
	U93	6.00	6.00	6.00	7.00	6.00	6.00	6.00	8.00	9.00	6.00	6.00	7.00	9.00	6.00
Iteration 1	U87	17.23	12.00	12.27	14.90	14.36	14.58	13.32	17.19	15.50	7.54	5.72	8.54	7.07	7.98
	U91	11.02	6.75	8.74	5.13	8.44	9.78	8.71	12.41	14.68	7.15	6.64	8.00	4.82	5.49
	U93	6.91	5.02	7.11	6.33	6.15	6.68	6.90	7.12	9.63	7.32	5.96	6.53	7.85	5.22
Iteration 2	U87		11.66	12.34	15.87	14.51	13.92	11.60	16.44	13.82	6.68	5.34	7.10	8.19	8.26
	U91		8.00	9.09	5.37	8.81	10.27	8.21	12.77	13.76	8.99	5.18	8.41	6.09	4.97
	U93		6.16	7.81	7.57	5.75	6.60	7.35	7.44	9.97	7.04	5.29	5.46	8.95	4.03
Iteration 3	U87			11.89	16.83	15.35	13.38	12.15	16.77	13.40	7.31	5.47	6.38	7.71	6.73
	U91			10.40	5.32	9.41	10.34	7.83	12.51	13.44	8.51	6.64	7.71	4.46	4.97
	U93			7.95	6.92	5.16	6.48	9.89	6.51	11.67	6.82	6.63	6.08	9.64	4.87
Iteration 4	U87				16.95	17.44	14.30	8.95	16.87	12.78	7.76	6.03	7.67	5.66	6.99
	U91				3.47	9.02	9.16	8.04	11.77	13.33	9.13	6.31	6.43	4.66	3.88
	U93				7.28	5.79	6.94	12.02	5.22	11.99	6.93	9.35	6.30	9.17	5.03
Iteration 5	U87					16.66	16.47	9.06	17.83	13.16	7.33	7.59	7.78	6.96	6.79
	U91					8.49	9.95	7.70	10.98	13.30	9.12	6.48	4.64	4.52	3.35
	U93					5.79	6.94	12.02	5.22	11.99	6.93	9.35	6.30	9.17	5.03
Iteration 6	U87						16.16	9.07	19.03	14.82	7.54	7.87	10.55	6.59	8.12
	U91						9.95	7.70	10.98	13.30	9.12	6.48	4.64	4.52	3.35
	U93						7.77	12.44	4.79	13.59	7.00	9.66	5.12	10.55	5.99
Iteration 7	U87							9.07	19.03	14.82	7.54	7.87	10.55	6.59	8.12
	U91							6.66	9.40	13.25	10.30	7.65	5.97	4.32	4.52
	U93							13.50	6.59	13.35	4.55	9.90	6.38	10.36	4.50
Iteration 8	U87								18.48	13.60	6.49	7.61	9.79	7.61	7.22
	U91								8.93	12.07	11.70	6.41	4.26	3.68	5.30
	U93								6.46	13.52	5.10	8.88	5.20	9.00	3.70
Iteration 9	U87									13.52	7.18	5.80	9.44	8.62	6.46
	U91									11.81	11.29	7.58	4.97	2.65	4.61
	U93									13.48	4.32	6.74	6.31	9.19	3.35
Iteration 10	U87										6.47	4.82	8.76	6.60	7.99
	U91										13.92	6.89	5.01	3.34	3.29
	U93										5.70	6.84	5.68	7.92	2.31
Iteration 11	U87											5.92	10.78	7.69	8.30
	U91											5.38	4.55	3.81	4.48
	U93											7.23	5.92	7.70	1.02
Iteration 12	U87												10.83	8.75	9.86
	U91												4.42	4.72	6.68
	U93												6.05	5.89	0.48
Iteration 13	U87													8.89	8.07
	U91													4.40	8.61
	U93													4.40	0.15
Iteration 14	U87														8.07
	U91														8.61
	U93														0.15

Table B.16. Contracted demand and additional demand simulation for case # 6

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	80.00	60.00	60.00	80.00	70.00	65.00	60.00	85.00	80.00	35.00	30.00	40.00	35.00	40.00
	U91	50.00	40.00	50.00	30.00	40.00	50.00	40.00	70.00	80.00	30.00	35.00	35.00	30.00	30.00
	U93	30.00	30.00	30.00	35.00	30.00	30.00	30.00	40.00	45.00	30.00	30.00	35.00	45.00	30.00
Additional Mean Demand	U87	16.00	12.00	12.00	16.00	14.00	13.00	12.00	17.00	16.00	7.00	6.00	8.00	7.00	8.00
	U91	10.00	8.00	10.00	6.00	8.00	10.00	8.00	14.00	16.00	6.00	7.00	7.00	6.00	6.00
	U93	6.00	6.00	6.00	7.00	6.00	6.00	6.00	8.00	9.00	6.00	6.00	7.00	9.00	6.00
Iteration 1	U87	18.10	11.63	12.81	16.61	13.39	13.93	11.16	15.13	17.45	8.16	6.83	7.47	6.32	7.74
	U91	9.46	7.85	10.60	5.32	7.02	12.32	7.79	14.80	16.13	5.70	6.99	7.72	6.78	6.15
	U93	6.52	4.88	6.72	8.06	5.94	5.26	6.68	8.77	10.07	4.53	5.44	7.85	10.09	7.02
Iteration 2	U87		10.33	12.06	16.74	13.55	14.69	11.79	15.02	17.19	7.44	7.07	8.12	6.22	8.26
	U91		7.06	11.18	6.12	8.16	12.28	10.45	15.57	15.08	6.67	7.50	6.93	7.52	5.39
	U93		3.84	7.51	7.21	5.70	5.97	7.89	9.32	11.28	2.61	3.62	8.84	10.18	7.71
Iteration 3	U87			11.35	15.39	13.54	13.69	12.32	14.93	16.67	7.08	7.00	7.88	7.71	8.84
	U91			10.89	6.67	9.09	12.22	9.54	14.52	14.74	5.88	6.88	7.99	8.04	5.41
	U93			5.01	6.79	6.66	5.66	8.45	9.02	11.13	2.41	5.19	10.81	10.80	6.19
Iteration 4	U87				14.76	13.80	13.22	11.12	16.85	18.97	6.91	6.75	8.33	6.50	8.60
	U91				6.04	6.98	12.39	9.91	15.13	14.32	7.22	6.39	6.88	7.92	6.17
	U93				8.23	6.65	5.49	8.63	10.37	9.99	3.54	4.25	10.58	11.09	5.34
Iteration 5	U87					13.26	12.05	11.59	17.01	18.11	5.26	8.49	7.92	6.68	9.60
	U91					6.38	13.33	9.14	12.82	15.91	8.83	5.71	5.98	9.07	6.02
	U93					6.65	5.49	8.63	10.37	9.99	3.54	4.25	10.58	11.09	5.34
Iteration 6	U87						11.41	12.43	18.96	17.84	5.12	8.41	8.07	6.53	10.52
	U91						13.33	9.14	12.82	15.91	8.83	5.71	5.98	9.07	6.02
	U93						5.03	8.03	9.50	10.08	4.42	3.87	10.07	10.66	4.71
Iteration 7	U87							12.43	18.96	17.84	5.12	8.41	8.07	6.53	10.52
	U91							8.05	13.85	15.43	8.83	6.29	7.51	8.23	6.61
	U93							7.94	8.66	9.43	3.58	2.96	8.66	9.29	5.76
Iteration 8	U87								18.52	18.22	5.67	9.10	7.11	7.08	11.11
	U91								12.44	14.03	9.72	6.69	6.41	6.89	7.85
	U93								10.03	10.65	3.57	3.49	10.04	8.79	5.14
Iteration 9	U87									18.08	5.69	8.96	7.65	7.85	12.44
	U91									13.74	11.39	6.11	7.02	6.88	6.81
	U93									10.86	3.54	3.67	10.14	8.27	4.47
Iteration 10	U87										6.94	9.11	7.39	6.01	14.51
	U91										9.91	5.08	6.82	6.76	5.22
	U93										3.95	3.54	9.20	7.36	3.47
Iteration 11	U87											8.66	7.23	4.63	14.41
	U91											4.73	7.14	7.28	5.17
	U93											3.57	9.35	8.23	1.66
Iteration 12	U87												8.10	3.43	14.38
	U91												5.30	6.26	6.40
	U93												8.37	9.25	3.02
Iteration 13	U87													3.22	15.10
	U91													7.68	7.50
	U93													10.47	2.96
Iteration 14	U87														15.10
	U91														7.50
	U93														2.96

Table B.17. Contracted demand and additional demand simulation for case # 7

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	60.00	70.00	60.00	70.00	40.00	50.00	50.00	70.00	60.00	30.00	40.00	40.00	50.00	60.00
	U91	40.00	50.00	60.00	70.00	30.00	30.00	30.00	50.00	50.00	20.00	30.00	50.00	30.00	30.00
	U93	40.00	40.00	40.00	60.00	20.00	30.00	20.00	30.00	50.00	20.00	20.00	30.00	20.00	30.00
Additional Mean Demand	U87	12.00	14.00	12.00	14.00	8.00	10.00	10.00	14.00	12.00	6.00	8.00	8.00	10.00	12.00
	U91	8.00	10.00	12.00	14.00	6.00	6.00	6.00	10.00	10.00	4.00	6.00	10.00	6.00	6.00
	U93	8.00	8.00	8.00	12.00	4.00	6.00	4.00	6.00	10.00	4.00	4.00	6.00	4.00	6.00
Iteration 1	U87	12.05	12.73	10.47	13.18	8.98	8.49	10.71	13.90	11.08	6.96	6.93	7.04	10.05	11.31
	U91	11.03	10.47	12.69	14.13	5.74	4.84	4.62	9.99	10.68	4.60	4.94	10.14	5.87	7.32
	U93	9.03	10.22	8.44	11.71	4.36	5.66	3.73	3.83	7.58	3.77	4.13	5.72	4.64	6.99
Iteration 2	U87		11.77	9.32	11.08	9.15	7.57	9.95	16.57	11.00	5.08	6.37	5.78	9.58	10.00
	U91		10.13	11.29	13.52	7.37	3.69	1.56	11.02	9.02	5.59	3.75	9.04	6.03	7.28
	U93		9.07	7.60	12.51	1.94	4.80	3.27	5.07	8.16	2.69	4.02	5.60	4.43	7.54
Iteration 3	U87			9.43	12.08	9.75	6.95	10.04	17.16	11.38	5.90	5.80	7.11	9.13	11.06
	U91			9.72	14.15	5.76	3.32	0.47	12.47	9.08	6.66	1.55	7.82	7.06	5.27
	U93			8.69	11.70	4.45	4.91	3.59	4.26	9.26	1.98	2.67	5.70	3.29	6.16
Iteration 4	U87				11.13	9.86	5.72	8.09	16.35	10.76	5.99	5.30	6.28	8.09	11.66
	U91				15.97	5.34	2.83	0.00	13.09	10.32	6.61	1.16	8.84	6.44	5.28
	U93				10.48	5.56	4.98	4.55	5.02	9.88	0.90	1.32	6.69	2.96	6.15
Iteration 5	U87					10.34	6.68	8.68	16.33	9.88	6.04	5.53	6.11	6.60	11.92
	U91					2.92	1.86	0.00	13.13	10.94	8.83	0.00	7.33	6.96	2.93
	U93					5.56	4.98	4.55	5.02	9.88	0.90	1.32	6.69	2.96	6.15
Iteration 6	U87						9.05	7.19	15.33	9.86	6.47	5.34	7.26	6.29	11.77
	U91						1.86	0.00	13.13	10.94	8.83	0.00	7.33	6.96	2.93
	U93						5.10	2.91	5.35	8.54	0.00	1.22	7.10	3.90	6.71
Iteration 7	U87							7.19	15.33	9.86	6.47	5.34	7.26	6.29	11.77
	U91							0.15	13.70	9.89	8.67	0.54	8.24	5.62	2.93
	U93							3.86	3.65	10.81	0.88	0.06	5.78	3.26	6.99
Iteration 8	U87								13.95	8.74	7.50	6.55	8.25	5.80	11.62
	U91								14.27	9.56	8.87	0.76	8.24	6.06	2.15
	U93								4.56	9.66	0.03	0.00	5.33	5.13	5.82
Iteration 9	U87									8.71	7.04	4.53	8.68	4.50	11.31
	U91									10.42	8.25	0.27	9.32	6.80	0.44
	U93									10.27	0.68	0.00	4.89	4.77	4.61
Iteration 10	U87										8.35	4.86	9.20	5.81	10.60
	U91										8.52	0.05	7.84	6.61	2.21
	U93										1.86	0.00	2.38	3.97	5.88
Iteration 11	U87											4.50	10.03	6.90	8.50
	U91											0.30	6.30	6.65	2.08
	U93											0.00	2.99	4.32	6.64
Iteration 12	U87												9.12	6.83	7.15
	U91												5.68	5.52	3.85
	U93												2.66	4.43	6.41
Iteration 13	U87													7.79	7.06
	U91													4.80	4.03
	U93													5.26	5.29
Iteration 14	U87														7.06
	U91														4.03
	U93														5.29

Table B.18. Contracted demand and additional demand simulation for case # 8

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	60.00	70.00	60.00	70.00	40.00	50.00	50.00	70.00	60.00	30.00	40.00	40.00	50.00	60.00
	U91	40.00	50.00	60.00	70.00	30.00	30.00	30.00	50.00	50.00	20.00	30.00	50.00	30.00	30.00
	U93	40.00	40.00	40.00	60.00	20.00	30.00	20.00	30.00	50.00	20.00	20.00	30.00	20.00	30.00
Additional Mean Demand	U87	12.00	14.00	12.00	14.00	8.00	10.00	10.00	14.00	12.00	6.00	8.00	8.00	10.00	12.00
	U91	8.00	10.00	12.00	14.00	6.00	6.00	6.00	10.00	10.00	4.00	6.00	10.00	6.00	6.00
	U93	8.00	8.00	8.00	12.00	4.00	6.00	4.00	6.00	10.00	4.00	4.00	6.00	4.00	6.00
Iteration 1	U87	10.79	12.89	11.57	13.57	8.25	10.12	10.08	13.12	11.12	4.91	8.23	8.31	12.54	12.67
	U91	8.66	10.57	12.70	13.37	7.82	6.40	6.07	9.40	10.45	4.17	7.31	10.98	5.79	5.38
	U93	8.99	5.47	7.91	10.91	5.12	5.37	3.51	6.11	7.67	3.50	2.29	6.17	4.79	4.99
Iteration 2	U87		14.04	10.40	11.14	9.52	9.65	10.22	12.99	11.26	3.50	9.43	7.04	12.26	13.05
	U91		9.15	11.60	14.39	7.77	7.15	7.45	7.02	11.81	4.42	7.39	9.66	6.36	2.68
	U93		5.34	8.80	10.00	6.80	4.22	4.88	5.40	6.60	3.38	1.78	6.93	7.69	3.56
Iteration 3	U87			12.08	12.80	8.17	8.14	9.39	12.67	11.65	5.22	8.25	7.43	9.90	13.84
	U91			10.15	13.75	8.75	8.02	7.74	7.09	11.71	4.88	8.65	9.72	6.99	4.19
	U93			9.64	12.55	5.79	3.20	4.28	6.46	5.55	2.07	0.00	7.23	6.79	4.98
Iteration 4	U87				13.59	7.28	9.73	8.26	11.65	12.08	5.31	8.82	8.09	10.42	14.66
	U91				13.46	9.29	8.01	6.76	6.78	11.31	5.03	9.47	10.31	7.77	5.42
	U93				11.00	5.27	3.29	5.08	6.72	5.57	0.50	0.00	7.11	7.00	6.60
Iteration 5	U87					8.04	12.72	9.39	11.57	12.78	4.91	9.83	8.00	10.01	15.46
	U91					9.21	9.46	9.35	7.35	11.16	3.56	9.82	10.61	7.66	4.77
	U93					5.27	3.29	5.08	6.72	5.57	0.50	0.00	7.11	7.00	6.60
Iteration 6	U87						13.53	7.32	12.70	13.49	4.43	10.09	6.91	10.88	15.49
	U91						9.46	9.35	7.35	11.16	3.56	9.82	10.61	7.66	4.77
	U93						2.39	5.70	8.91	5.42	0.00	0.00	6.75	6.35	7.38
Iteration 7	U87							7.32	12.70	13.49	4.43	10.09	6.91	10.88	15.49
	U91							8.69	8.61	12.14	4.46	8.46	7.95	8.59	6.32
	U93							4.54	9.32	6.69	1.77	0.14	6.18	6.98	8.94
Iteration 8	U87								10.54	13.50	3.02	10.40	7.27	12.24	14.25
	U91								8.37	12.19	3.34	8.98	8.26	7.89	6.48
	U93								9.52	6.86	1.31	0.55	6.07	5.28	8.81
Iteration 9	U87									12.88	3.53	11.42	6.55	13.03	13.60
	U91									10.71	3.49	7.79	8.52	7.85	6.33
	U93									7.17	0.00	0.12	4.54	4.70	10.00
Iteration 10	U87										3.61	14.11	5.44	13.70	13.89
	U91										3.59	9.77	7.54	7.76	5.78
	U93										0.00	0.00	4.31	5.07	10.18
Iteration 11	U87											14.57	5.25	14.22	13.92
	U91											9.84	7.91	6.45	4.80
	U93											0.00	3.33	4.43	9.84
Iteration 12	U87												5.16	15.98	13.54
	U91												9.77	8.55	4.86
	U93												4.42	1.85	10.72
Iteration 13	U87													15.33	15.65
	U91													8.14	3.66
	U93													1.83	11.56
Iteration 14	U87														15.65
	U91														3.66
	U93														11.56

Table B.19. Contracted demand and additional demand simulation for case # 9

	Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Contracted Demand	U87	60.00	70.00	60.00	70.00	40.00	50.00	50.00	70.00	60.00	30.00	40.00	40.00	50.00	60.00
	U91	40.00	50.00	60.00	70.00	30.00	30.00	30.00	50.00	50.00	20.00	30.00	50.00	30.00	30.00
	U93	40.00	40.00	40.00	60.00	20.00	30.00	20.00	30.00	50.00	20.00	20.00	30.00	20.00	30.00
Additional Mean Demand	U87	12.00	14.00	12.00	14.00	8.00	10.00	10.00	14.00	12.00	6.00	8.00	8.00	10.00	12.00
	U91	8.00	10.00	12.00	14.00	6.00	6.00	6.00	10.00	10.00	4.00	6.00	10.00	6.00	6.00
	U93	8.00	8.00	8.00	12.00	4.00	6.00	4.00	6.00	10.00	4.00	4.00	6.00	4.00	6.00
Iteration 1	U87	12.31	15.03	11.92	13.57	8.25	8.76	11.20	12.52	12.16	6.96	6.85	7.61	10.19	12.55
	U91	6.94	10.37	12.07	12.66	6.26	7.12	5.18	10.90	10.18	1.49	5.45	10.62	5.17	4.19
	U93	8.08	7.09	7.75	11.36	4.49	6.50	4.56	6.78	12.12	3.06	3.29	5.90	3.13	6.66
Iteration 2	U87		15.74	10.26	12.97	6.13	8.14	11.27	12.91	10.62	6.67	8.37	7.57	11.56	12.22
	U91		9.98	11.88	12.25	5.36	6.16	5.98	12.61	9.29	1.18	5.78	10.02	5.81	5.22
	U93		7.58	7.06	10.81	3.85	5.70	5.18	5.97	11.84	3.08	3.46	6.99	2.92	8.56
Iteration 3	U87			9.70	13.01	6.12	6.75	11.17	14.02	11.24	5.54	7.60	6.63	10.52	11.27
	U91			12.79	12.02	6.11	6.47	4.41	12.79	10.52	1.87	4.16	10.16	5.72	3.49
	U93			6.79	11.23	3.46	5.01	5.58	5.46	10.45	3.78	2.19	7.66	3.75	8.42
Iteration 4	U87				12.47	5.50	8.30	12.76	13.45	9.82	5.71	7.79	6.93	8.05	12.23
	U91				11.35	5.90	6.56	5.11	13.24	11.28	2.02	3.41	9.66	5.12	2.95
	U93				11.00	2.69	5.35	5.75	5.60	11.65	4.86	3.16	7.35	5.01	6.92
Iteration 5	U87					5.47	7.28	12.34	13.52	8.80	6.37	8.78	7.45	5.99	11.33
	U91					5.81	6.27	4.63	13.01	11.53	1.47	1.46	11.80	5.16	1.89
	U93					2.69	5.35	5.75	5.60	11.65	4.86	3.16	7.35	5.01	6.92
Iteration 6	U87						7.51	12.81	13.13	8.38	5.80	9.19	8.39	4.49	11.20
	U91						6.27	4.63	13.01	11.53	1.47	1.46	11.80	5.16	1.89
	U93						5.77	5.12	4.95	11.26	5.74	2.54	8.64	4.12	8.47
Iteration 7	U87							12.81	13.13	8.38	5.80	9.19	8.39	4.49	11.20
	U91							3.80	12.84	11.22	2.18	2.46	11.50	5.16	2.88
	U93							6.70	3.63	12.33	4.91	1.18	8.93	1.88	7.97
Iteration 8	U87								12.75	7.37	5.12	8.69	8.13	5.20	10.85
	U91								11.72	11.88	1.48	3.12	12.13	2.71	3.80
	U93								5.10	11.65	4.55	0.00	9.64	3.29	7.26
Iteration 9	U87									7.33	4.70	8.33	8.77	3.80	12.04
	U91									10.85	0.79	4.29	12.13	1.86	3.68
	U93									11.66	3.44	0.00	10.91	1.82	7.61
Iteration 10	U87										5.67	10.88	10.69	3.19	11.41
	U91										0.92	4.49	13.11	2.17	4.63
	U93										3.87	1.01	9.42	0.57	6.16
Iteration 11	U87											10.75	11.83	4.09	10.81
	U91											4.39	12.84	2.48	4.49
	U93											0.00	8.66	2.28	4.98
Iteration 12	U87												10.44	4.75	9.56
	U91												13.09	2.04	5.64
	U93												7.62	2.53	2.98
Iteration 13	U87													3.54	10.39
	U91													1.31	4.66
	U93													1.79	3.97
Iteration 14	U87														10.39
	U91														4.66
	U93														3.97

case # 1: all demands is satisfied (graph not included)

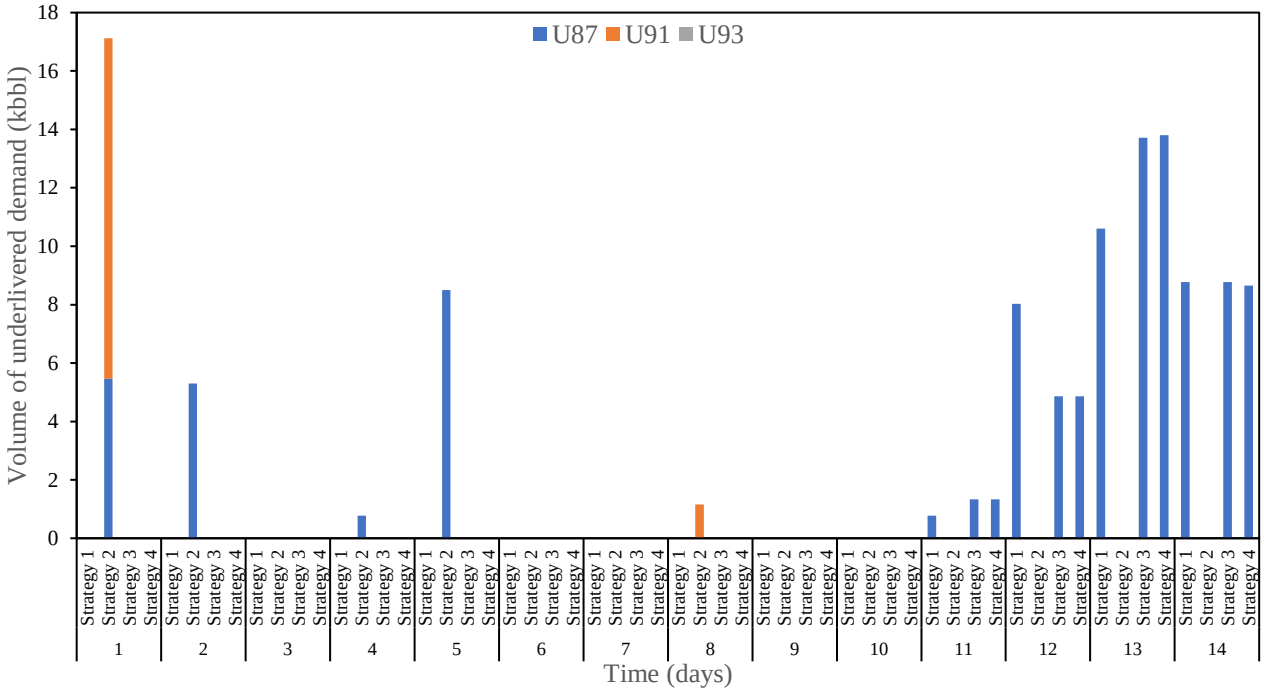

Figure B.1. Products undelivered demands for case # 2 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

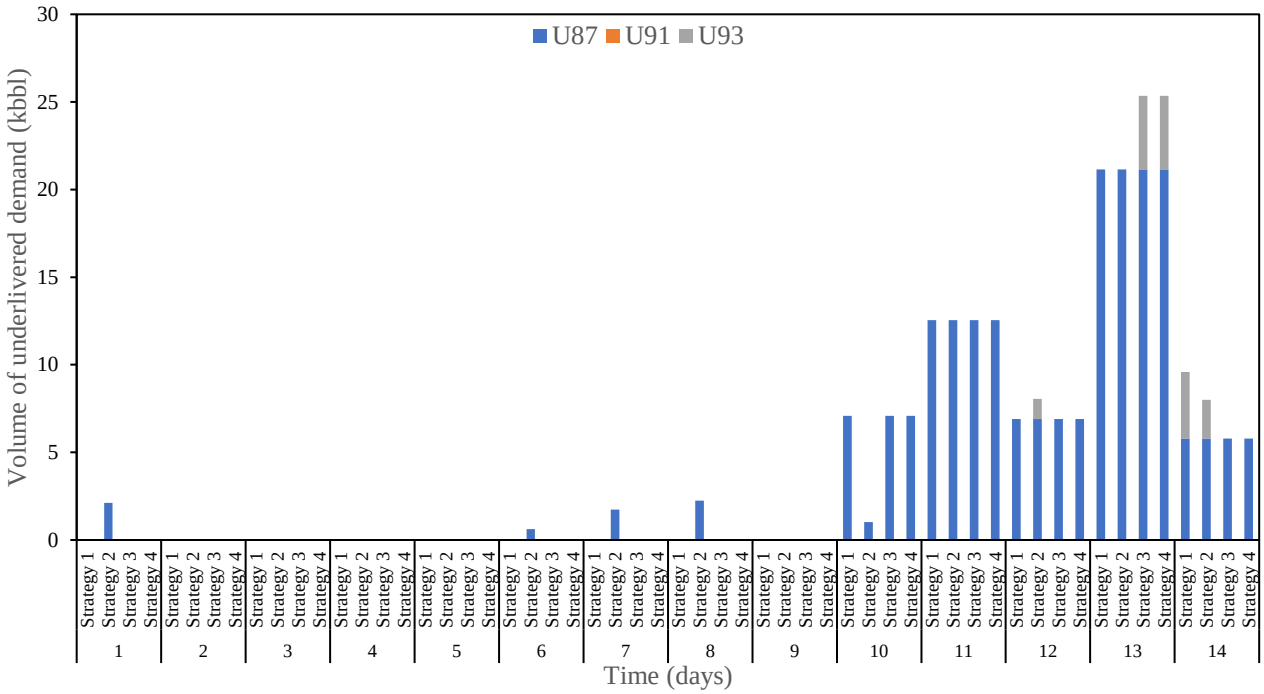

Figure B.2. Products undelivered demands for case # 3 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

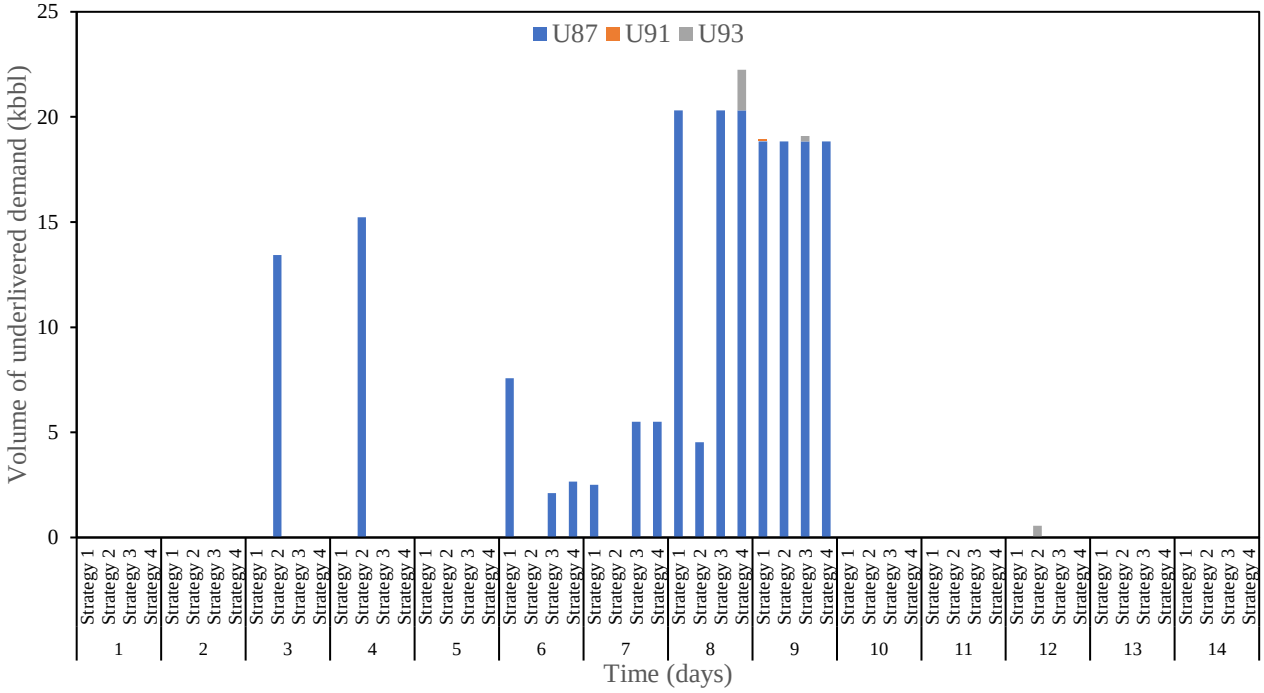

Figure B.3. Products undelivered demands for case # 4 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

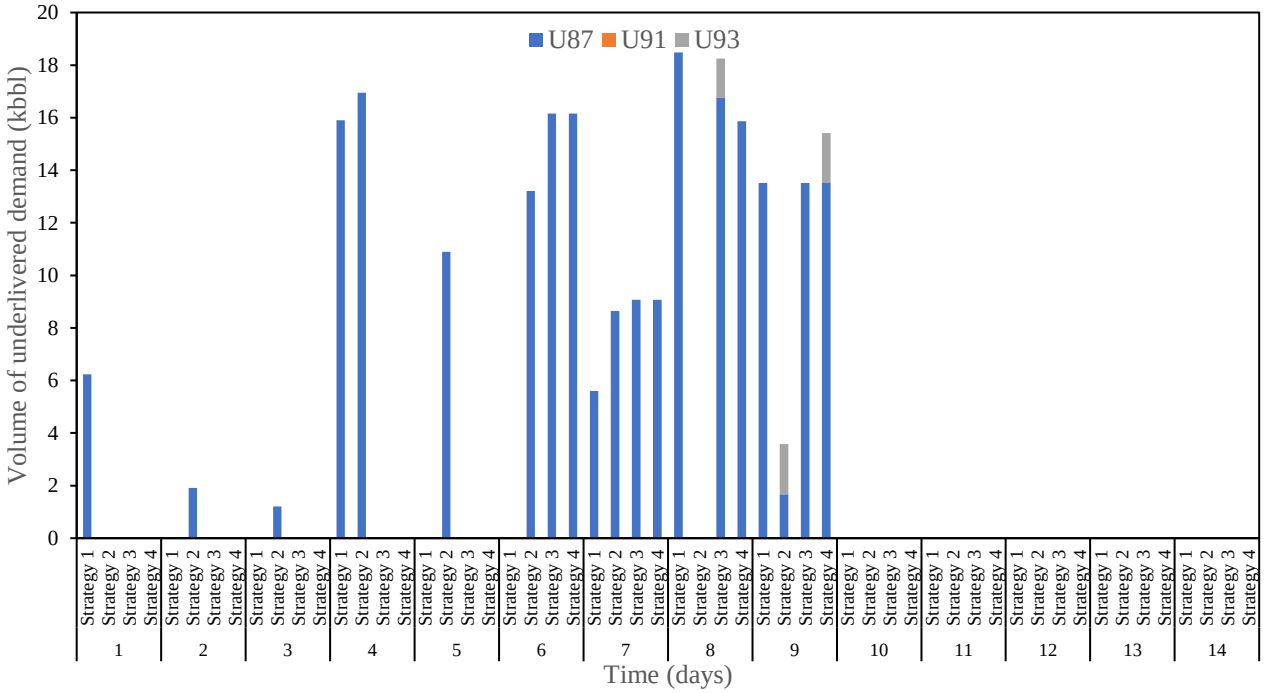

Figure B.4. Products undelivered demands for case # 5 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

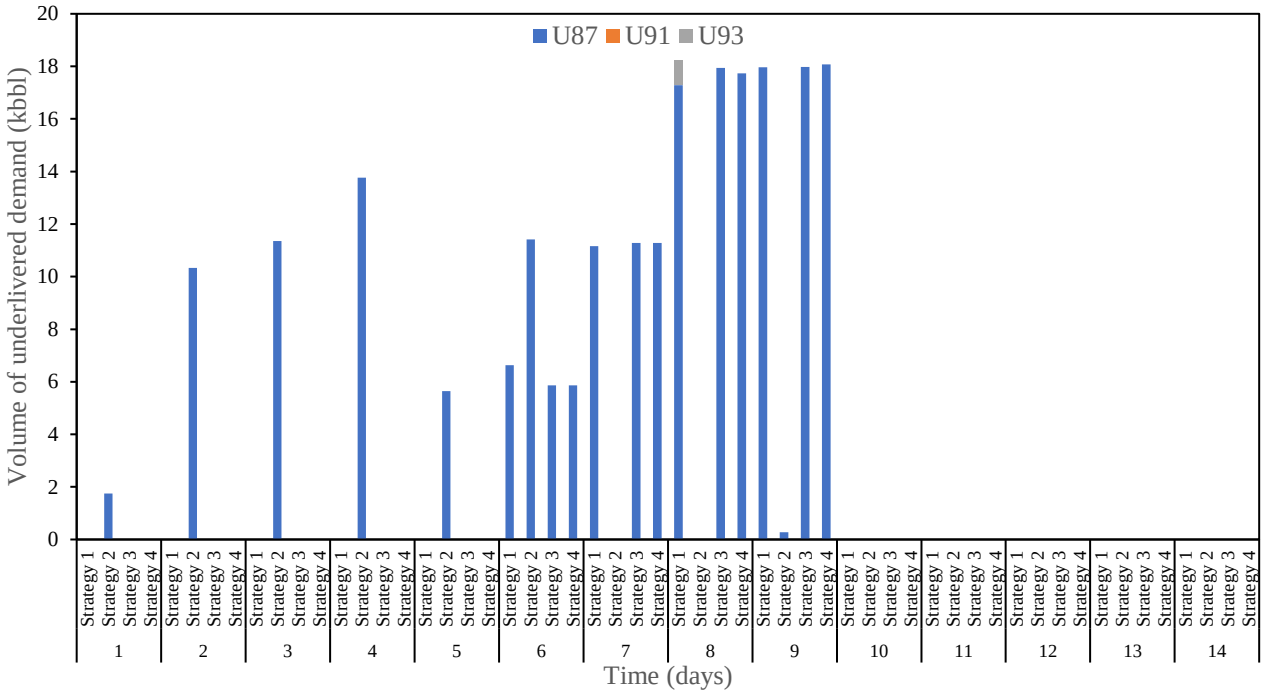

Figure B.5. Products undelivered demands for case # 6 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

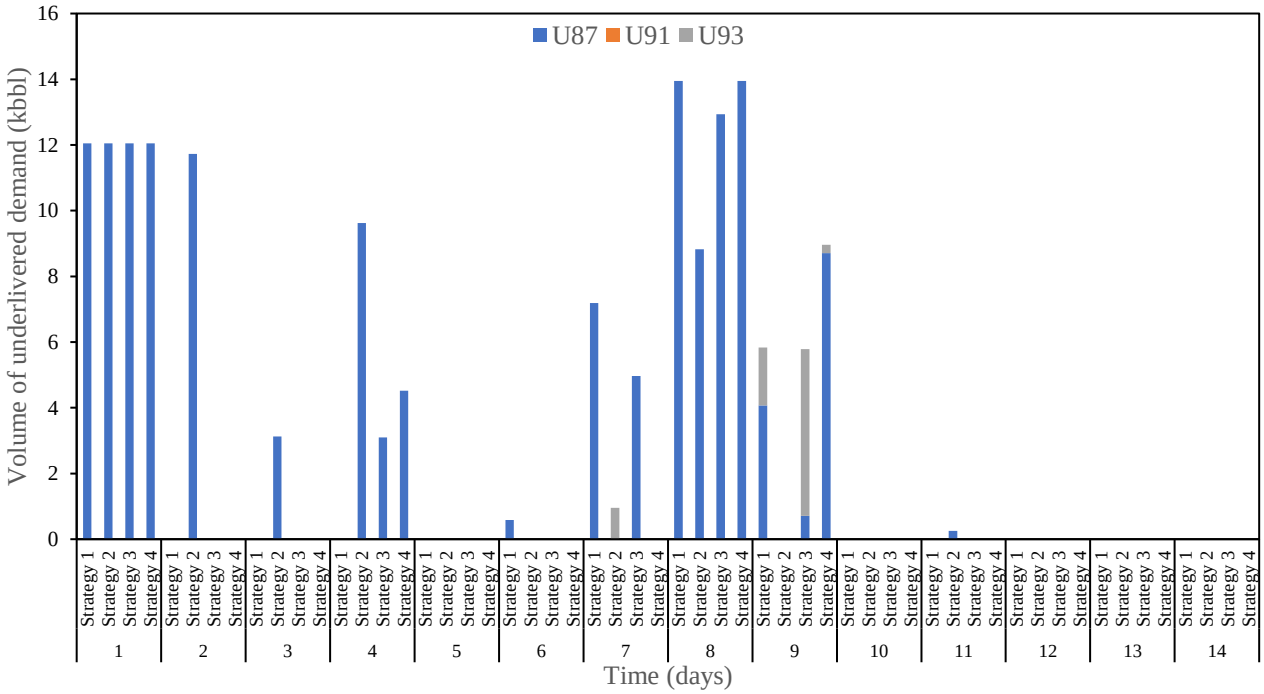

Figure B.6. Products undelivered demands for case # 7 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

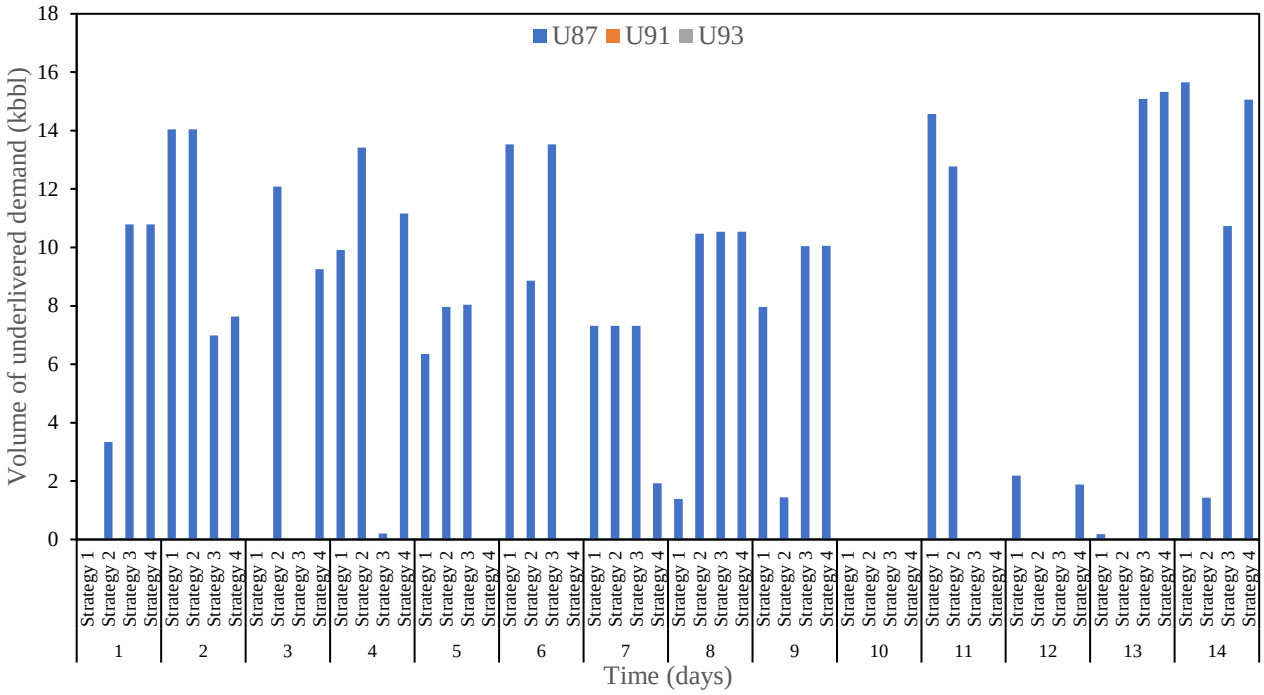

Figure B.7. Products undelivered demands for case # 8 using the four method of nonlinear blending rules for products ■ U87, ■ U91, and ■ U93

Appendix C: Supporting Information for Chapter 4

Table C.1. Weights of predicted output variables

Output	Weight
T_{t_24}	3
T_{t_37}	3
T_{t_51}	3
T_{t_56}	3
T_{t_85}	5
P_{t_1}	2
ΔP_{top}	2
ΔP_{bottom}	2
$x_{C_2H_4\ t\_97}$	5
$x_{C_2H_6\ t\_24}$	5
$\dot{m}_{top}$	3
T_{top}	3
$x_{C_2H_6\ top}$	10
$\dot{m}_{bottom}$	3
T_{bottom}	3
$x_{C_2H_4\ bottom}$	10

Appendix D: Supporting Information for Chapter 5

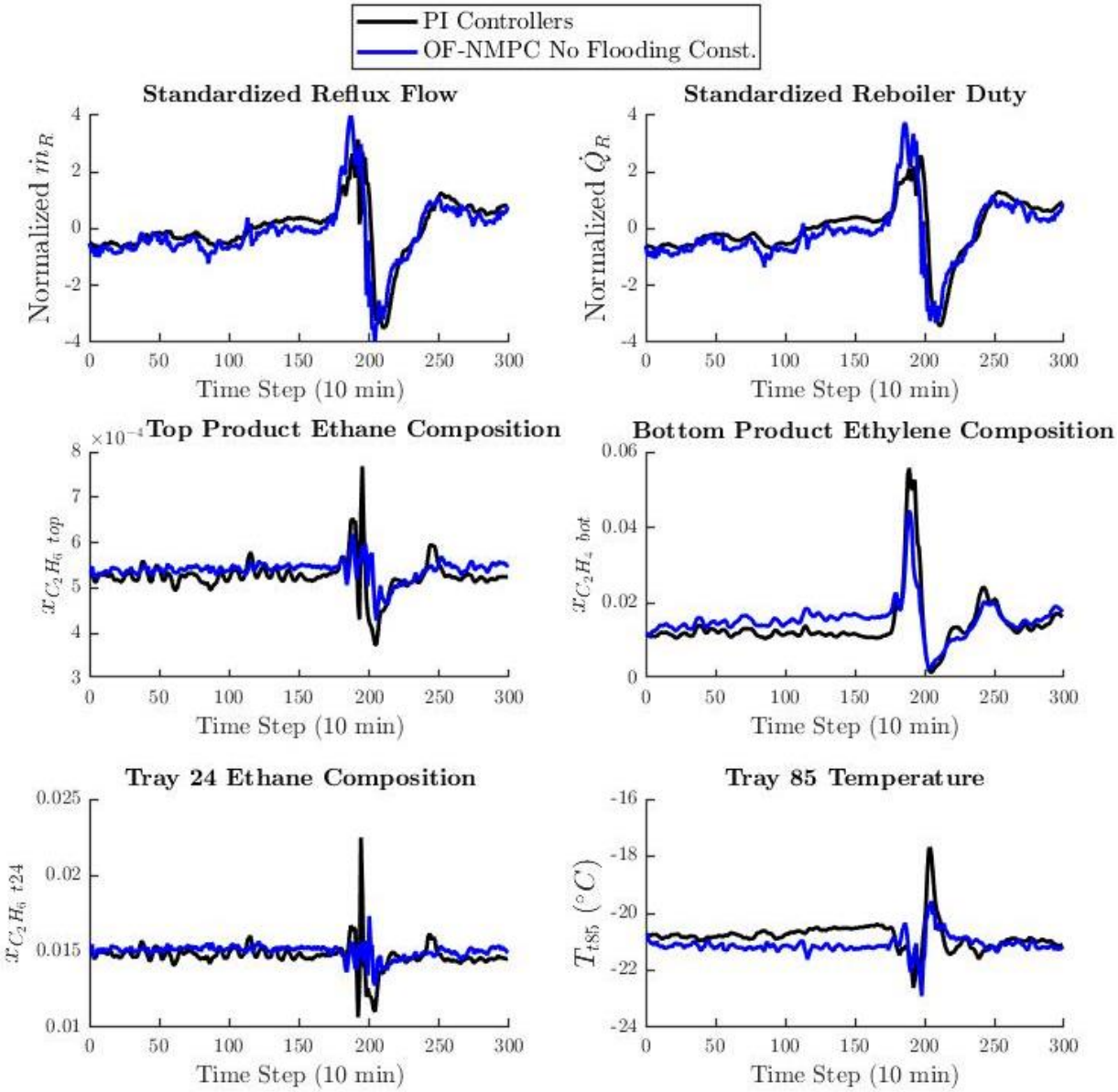

Figure D.1. comparison of the manipulated and controlled variables measurements of OF-NMPC control strategy with no flooding constraint versus PI controllers strategy

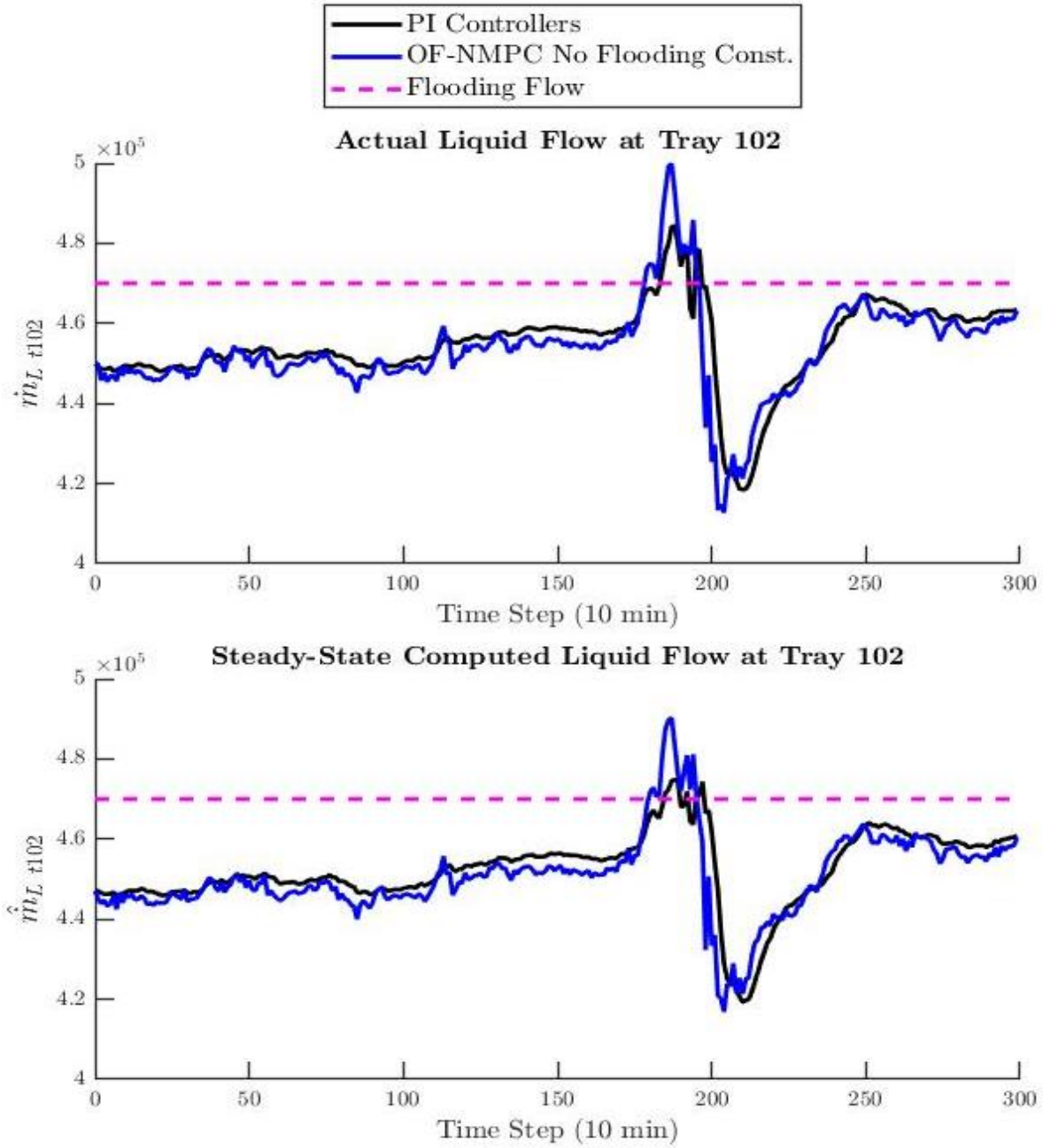

Figure D.2. comparison of the actual and computed liquid flow at tray 102 of OF-NMPC control strategy versus PI controllers strategy