CLOSED-LOOP PREDICTION FOR OPTIMIZATION AND CONTROL

CLOSED-LOOP PREDICTION FOR ROBUST AND STABILIZING OPTIMIZATION AND CONTROL

By LLOYD ALASTAIR MACKINNON, M.Sc.,B.Eng.

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

McMaster University

DOCTOR OF PHILOSOPHY (2023) McMaster University

(Chemical Engineering) Hamilton, Ontario, Canada

TITLE:	Closed-Loop Prediction for Robust and Stabilizing Optimization and Control
AUTHOR:	Lloyd Alastair MacKinnon M.Sc., Imperial College London, London, UK B.Eng., McMaster University, Hamilton, Canada
SUPERVISOR:	Dr. Christopher L.E Swartz
NUMBER OF PAGES:	xxi, 202

Lay Abstract

It is common for control and optimization of chemical plants to be performed in a multi-layered hierarchy. The ability to predict the behavior of other layers or the future behavior of the same layer can improve overall plant performance. This thesis presents optimization and control frameworks which use this concept to more effectively control and economically optimize chemical plants which are subject to uncertain behavior or instability. The strategy is shown, in a series of simulated case studies, to effectively control chemical plants with uncertain behavior, control and optimize unstable plant systems, and economically optimize uncertain chemical plants. One of the drawbacks of these strategies is the relatively large computation time required to solve the optimization problems. Therefore, for uncertain systems, the problem is separated into smaller pieces which are then coordinated towards a single solution. This results in reduced computation time.

Abstract

The control and optimization of chemical plants is a major area of research as it has the potential to improve both economic output and plant safety. It is often prudent to separate control and optimization tasks of varying complexities and time scales, creating a hierarchical control structure. Within this structure, it is beneficial for one control layer to be able to account for the effects of other layers. A clear example of this, and the basis of this work, is closed-loop dynamic real-time optimization (CL-DRTO), in which an economic optimization method considers both the plant behavior and the effects of an underlying model predictive controller (MPC). This technique can be expanded on to allow its use and methods to be employed in a greater diversity of applications, particularly unstable and uncertain plant environments.

First, this work seeks to improve on existing robust MPC techniques, which incorporate plant uncertainty via direct multi-scenario modelling, by also including future MPC behavior through the use of the CL modelling technique of CL-DRTO. This allows the CL robust MPC to account for how future MPC executions will be affected by uncertain plant behavior. Second, Lyapunov MPC (LMPC) is a generally nonconvex technique which focuses on effective control of plants which exhibit open-loop unstable behavior. A new convex LMPC formulation is presented here which can be readily embedded into a CL-DRTO scheme. Next, uncertainty handling is

incorporated directly into a CL-DRTO via a robust multi-scenario method to allow for the economic optimization to take uncertain plant behavior into account while also modelling MPC behavior under plant uncertainty. Finally, the robust CL-DRTO method is computationally expensive, so a decomposition method which separates the robust CL-DRTO into its respective scenario subproblems is developed to improve computation time, especially for large optimization problems.

Acknowledgements

I would like to first express my profound gratitude to my supervisor, Dr. Christopher Swartz, for his guidance, support, and understanding. Without his expertise this thesis would not have been possible. I would also like to thank Dr. Prashant Mhaskar and Dr. Shahin Sirouspour for being on my supervisory committee. They did a phenomenal job thoroughly discussing and challenging all our findings to ensure the project's success. I would additionally like to thank Dr. Mhaskar for his guidance and support during our collaboration.

I would also like to acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC) and the McMaster Advanced Control Consortium (MACC).

I would like to thank my friends and colleagues, Daniela Dering, Carlos Rodriguez, Anthony Quarshie, and Praveen Ramesh. You helped me in countless ways throughout the last few years and made my time here infinitely more enjoyable. I would like to thank my family for their love and support, especially my parents who have been there for me every day. Also, thanks to my dad who continues to help me with my math homework. And I would like to thank all my friends, who keep me sane, grounded, and looking forward to every day. To everyone in my life, thank you.

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviations

BFGS	Broyden-Fletcher-Goldfarb-Shanno algorithm
CL	Closed-Loop
CL-DRTO	Closed-Loop Dynamic Real-Time Optimization
CSTR	Continuous Stirred-Tank Reactor
DAE	Differential Algebraic system of Equations
DMC	Dynamic Matrix Control
DRTO	Dynamic Real-Time Optimization
KKT	Karush-Kuhn-Tucker optimality conditions
LMI	Linear Matrix Inequality
LMPC	Lyapunov-based Model Predictive Control
LP	Linear Program

MILP	Mixed-Integer Linear Program
MIMO	Multi-Input-Multi-Output
MINLP	Mixed-Integer NonLinear Program
MPC	Model Predictive Control
MPCC	Mathematical Program with Complementarity Constraints
NAMW	Number Average Molecular Weight
NLP	NonLinear Program
NMPC	Nonlinear Model Predictive Control
OL	Open-Loop
PCA	Principal Component Analysis
QDMC	Quadratic Dynamic Matrix Control
QP	Quadratic Program
RTO	Real-Time Optimization
SISO	Single-Input-Single-Output
SP	Set-Point
SSE	Sum of Squared Errors

Declaration of Academic Achievement

The following is a declaration that the research described in this thesis was completed by Lloyd MacKinnon and recognizes the contributions of Hao Li, Praveen Ramesh, Dr. Prashant Mhaskar, and Dr. Christopher Swartz. Lloyd MacKinnon contributed to the research directions, the design of the research, the formulation of Chapter 1, the formulation of Chapter 2 and was responsible for the formulation of Chapter 3, the formulation of Chapter 4, the design of the case studies for all chapters, the implementation of the case studies, the data analysis, and the writing of the manuscript. Hao Li contributed to the design and formulation of Chapter 1. Praveen Ramesh contributed to the design and formulaiton of Chapter 2. Dr. Prashant Mhaskar contributed to the research direction and design of Chapter 2, and the review of the manuscript of Chapter 2. Dr. Christopher Swartz contributed to the research directions, design of the research, and the review of the manuscript.

Chapter 1

Introduction

1.1 Background and Motivation

The control and optimization of chemical plants is an ongoing and active area of research as modern chemical plants become increasingly complex and dynamic. Many modern applications require the plant to respond to frequent changes in external conditions, including product demand and raw material cost and availability. For this reason, the methods used to control and optimize these plants should be capable of directing the plant to new economically optimal states quickly and effectively while also ensuring maximum economic productivity during the transition itself. While this optimization could be implemented offline, the decisions become suboptimal as soon as process and economic conditions change. For this reason, economic optimization is preferably implemented in a real-time optimization (RTO) framework.

Additionally, it is common for control strategies to be designed hierarchically, with large-scale, long-term problems handled at the top and local, short-term problems at the bottom. This allows the different tasks to be solved separately and

asynchronously, reducing complexity and computation time of each layer. As such, optimization methods should be able to take advantage of this hierarchical control strategy. A typical hierarchical automation architecture is illustrated in Figure 1.1. The relevant layers to this work are the (Dynamic) Real-Time Optimization (DRTO) and Model Predictive Control (MPC) layers.

The use of RTO systems for the purpose of determining economically optimal operating points for chemical plants is a well-studied technique in the field of optimization and control. Most commonly, RTO is performed in a steady state paradigm, where the optimization problem considers only the ideal operating point, and not how to transition the plant to that point [48, 12]. Initial work focused on using a high fidelity, steady state model at the optimization level in conjunction with a low fidelity, dynamic model at a lower control level (such as in an MPC). A key advantage of the real-time nature of the RTO strategy is that it can respond quickly and without intervention to unexpected disturbances, and then adjust the optimal plant operating points accordingly. Ying and Joseph [79] examine some of the stability properties and economic properties of versions of this system where the supervisory optimization layer is a Linear Program (LP) or Quadratic Program (QP). An excellent review of RTO implementation strategies is provided in Darby et al. [12].

An important development in RTO is the incorporation of dynamic models into the optimization formulation, rather than simply using a steady state model. Such dynamic models allow the DRTO to predict how the plant will perform during transitions and can therefore optimize for economic performance both during steady state operation and during transition phases. This is especially useful for applications

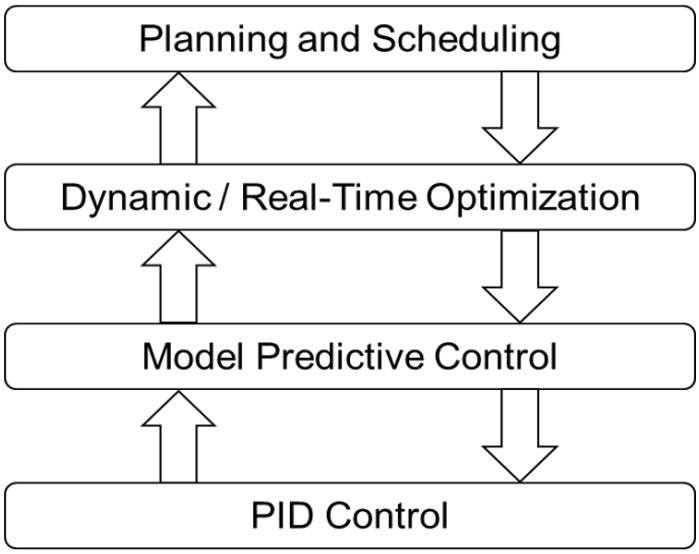

Figure 1.1: Illustration of a typical control hierarchy. Long-term supervisory and economic optimization occurs near the top with short-term control and plant action occurring near the bottom.

where the plant undergoes frequent or slow transitions.

Nath and Alzein [61] use linear dynamic models to develop a DRTO system for coordinating distributed MPC problems. The use of linear process models allows the problem to be formulated as a positive semi-definite quadratic program, keeping the optimization problem tractable for on-line solution. This DRTO was then applied to an olefin plant and was found to significantly improve ethylene production capacity over the previous strategy of steady state RTO. Tosukhowong et al. [75] propose a DRTO system with a reduced order model, which is executed at a substantially lower frequency than the underlying MPC. This system was applied to two example problems, the second of which comprised a CSTR, storage tank, and a separator with recycle stream. The proposed scheme was found to significantly outperform a steady state RTO in terms of economics both with and without disturbances present in the simulation. This was accomplished because the DRTO was able to improve

the economic conditions during transition and cause the plant to reach the optimal steady state more quickly than with a steady state RTO.

Kadam et al. [30], rather than simply using a low frequency DRTO, trigger the DRTO execution only when necessary, such as during transitions, thereby reducing the overall computational demand but allowing the DRTO to effectively optimize during transitions. They combine this with an MPC at the lower level and show the benefits of such a two-level system over determining optimal operating points offline with a methyl acetate semi-batch distillation process. Würth et al. [78] expand on the execution frequency aspect of DRTO implementation by using a sensitivity analysis to determine whether the DRTO needs to be executed at a given time point, with steady operation leading to less frequent execution and more transient behavior resulting in more frequent execution. Swartz and Kawajiri [74] provide an excellent review of existing DRTO strategies and implementations.

In addition to modeling the dynamic behavior of the plant for computing optimal transitions, it is also possible to simultaneously consider the behavior of the MPC which controls the plant. MPC, as its own optimization problem, independently determines the input moves which will drive the system towards the set-point. This behavior affects the conditions of the plant during transition, thus taking it into account at the DRTO level can improve the overall economics of the plant. Jamaludin and Swartz [27] directly embed the MPC optimization subproblems into the DRTO, resulting in a multilevel optimization problem. They show that such a closed-loop (CL) prediction of MPC behavior improves economic performance over a similar DRTO without explicit consideration of the MPC. The DRTO problem is solved by reformulating the MPC optimization subproblems as algebraic equations

corresponding to the first-order Karush-Kuhn-Tucker (KKT) optimality conditions, and including them as constraints in the DRTO economic optimization problem. A representation of a CL-DRTO architecture is shown in Fig. 1.2.

Alternative solution strategies that have been used for solving optimization problems that account for the predicted plant response under the action of constrained MPC include embedded closed-loop simulation [15], and use of a multiparametric MPC formulation [8]. Jamaludin and Swartz [29] examine the practicality of different approximations of the MPC behavior to reduce the overall computation time of the DRTO. Following this, Jamaludin et al. [25] and Li and Swartz [39] expand on the CL-DRTO by applying it to distributed MPC systems where the DRTO is also coordinating different MPC subsystems.

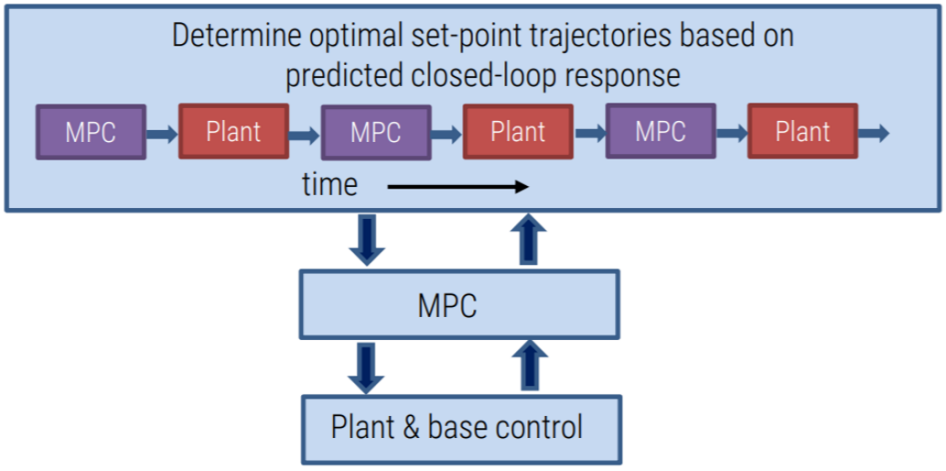

Figure 1.2: Visual representation of the CL-DRTO algorithm.

1.2 Research Problem Statement

The goal of this research is to extend a closed-loop dynamic real-time optimization (CL-DRTO) strategy for use in applications where stability and uncertainty handling are required at either the supervisory control or economic optimization level. The objectives of this research are as follows:

- Apply the CL prediction strategy to a single-level robust multi-scenario MPC to allow the MPC to predict its own future responses uncertain plant behavior.
- Formulate a convex Lyapunov MPC (LMPC) strategy which is effective at controlling an open-loop (OL) unstable plant.
- Embed the convex LMPC directly into a CL-DRTO strategy and evaluate the ability of the overall scheme to optimize and control an unstable system.
- Incorporate uncertainty handling into the CL-DRTO by using a multi-scenario modelling approach.
- Evaluate the ability of the multi-scenario CL-DRTO method to optimize an uncertain system across a range of possible model realizations.
- Improve the computation time of the robust CL-DRTO method for large problems by decomposition of the singular optimization problem into scenario subproblems.

1.3 Thesis Outline

The thesis is organized into the following chapters:

- In **Chapter 2**, we seek to combine a scenario-based approach with embedded closed-loop prediction under future MPC control action by directly incorporating MPC subproblems into the overall robust MPC formulation. This allows the current MPC to predict closed-loop MPC responses to a range of uncertain future plant realizations. The resulting multilevel programming problem is solved by reformulating the inner MPC optimization subproblems as algebraic constraints corresponding to their first-order optimality conditions, resulting in a single-level mathematical program with complementarity constraints (MPCC). The performance of the robust MPC scheme is evaluated against a standard MPC formulation in linear and nonlinear case studies.

- In **Chapter 3**, we address the problem of stabilizing dynamic real-time optimization and control of nonlinear systems. The chapter seeks to bring the improved performance of a CL-DRTO to stabilizing control systems by formulating a CL-DRTO which utilizes and explicitly models an underlying Lyapunov stabilizing MPC to achieve stabilization for nonlinear systems. The proposed formulation is compared to a previously developed endpoint penalty formulation to demonstrate the improved closed-loop control and performance.

- In **Chapter 4**, we extend the formulation of CL-DRTO for direct inclusion of uncertainty handling. A robust multi-scenario CL-DRTO scheme which models the dynamic behavior of the plant and its MPC system under uncertainty is introduced. The method is applied and its performance evaluated in two nonlinear case studies, where an input clipping approximation scheme is used to reduce the computation time. The effects of number of scenarios and multiple sources of uncertainty are also investigated.

- In **Chapter 5**, we present a decomposition approach for the robust closed-loop dynamic real-time optimization method of the previous chapter in order to reduce the computation time of the optimization problem. The method uses a primal decomposition approach which separates the problem into individual scenarios and fixes the non-anticipativity constraint variables for all subproblems. The decomposition method is compared to the monolithic robust CL-DRTO of the previous chapter in two case studies. The method is shown to be effective at reducing computation time when the problem size is sufficiently large, particularly as the number of modelled scenarios increases.

- In **Chapter 6**, we conclude the work by covering the overall findings of the research. The key contributions are summarized and the future research directions are presented.

Chapter 2

Robust Model Predictive Control with Embedded Multi-Scenario Closed-Loop Prediction

The formulations and results in this chapter have been published and presented in:

[1] MacKinnon, L., Li, H., Swartz, C.L.E., 2021. Robust model predictive control with embedded multi-scenario closed-loop prediction. Computers & Chemical Engineering 149, 107283.

[2] MacKinnon, L., and Swartz, C. L. E., 2019. Robust Model Predictive Control with Embedded Multi-Scenario Closed-Loop Prediction. Presented at the 69th Canadian Chemical Engineering Conference (CCEC 2019), Halifax, NS, Canada.

[3] MacKinnon, L., and Swartz, C.L.E., 2020. Robust Multi-Scenario MPC with Embedded Closed-Loop Prediction. Presented at the 2020 Virtual AIChE

Annual Meeting (AIChE 2020).

2.1 Introduction

Model Predictive Control (MPC) is a standard technique for control of chemical plants, but relies on an accurate model of the plant behavior to perform efficiently. An inaccurate model, whether due to structural mismatch, parameter uncertainty, or unknown disturbances, can lead to suboptimal plant performance. This can include overly aggressive or conservative behavior, constraint violation, large overshoot, or slow settling time. In order to mitigate these effects, robust MPC methods have been developed to deal with this plant uncertainty. The main drawbacks of robust MPC methods are the increased complexity and solution time, and the possibility of inferior performance relative to standard MPC if the model is highly accurate. The former of these problems is rapidly being mitigated by the development of efficient numerical computation and optimization algorithms, and increasing computational capabilities. The latter is case specific and, for many applications, unlikely, since the plant model is typically identified from plant data and chosen to be of relatively simple structure. Therefore, robust MPC has a wide range of possible applications and its use should only increase over time.

A number of robust MPC schemes have been proposed in the literature over the past two to three decades, many of which can be categorized into a few key paradigms. Bemporad and Morari [5] provide an excellent review of robust MPC approaches. One key distinction is between the use of open-loop versus closed-loop prediction of the propagation of the uncertainty over time. In the former, there is no provision for future control action as the uncertainty propagates, whereas in the latter, some form

of feedback action is considered, resulting in less conservative control action. Some of the key robust MPC paradigms are briefly reviewed in the following paragraphs.

Campo and Morari [10] propose a robust MPC scheme in which the predicted worst-case tracking error over a family of possible plants is minimized. An open-loop prediction approach is utilized, and the control problem is formulated as an LP. Kothare et al. [31] minimize an upper bound on the worst-case performance, with the problem formulated as a linear matrix inequality (LMI) by computing a linear state feedback controller at every control execution. Lee and Yu [38] propose a robust MPC scheme that minimizes a worst-case quadratic performance objective. Formulations based on both open-loop and closed-loop prediction are presented, with the latter based on Bellman's principle of optimality and dynamic programming. Kouvaritakis et al. [32] capture future feedback action through a linear state feedback law, similar to Kothare et al. [31], but add a vector which is included in the optimization decision space. They furthermore propose that the feedback gain be determined off-line, with the online calculation of the feedback law bias term determined through the solution of a single LMI, resulting in a significant reduction in computation time. Wan and Kothare [76] propose a robust output feedback MPC scheme that includes a sequence of state feedback control laws and a state estimator that are designed off-line through LMI formulations. Online, a specific control law is determined based on the current state through a bisection search.

Bemporad et al. [4] develop a robust MPC scheme in which the control input is determined as an explicit function of the state through multiparametric programming. A worst-case linear performance objective (infinity or 1-norm based) is optimized. Both open- and closed-loop prediction are considered, the latter utilizing dynamic

programming. Sakizlis et al. [70] similarly utilize a multiparametric formulation for robust MPC, and optimize a quadratic performance objective based either on nominal performance or expected performance over the uncertainty set.

Langson et al. [37] develop a tube-based robust MPC method for linear time invariant (LTI) systems with additive disturbances, and describe its extension to uncertain plants. The objective is to determine a tube and a control policy, where a tube is defined as a sequence of sets of states within which the system states will lie for all admissible disturbance sequences. The method is extended in Mayne et al. [53] to nonlinear plants with additive disturbances. A detailed description of tube-based MPC schemes is given in Rawlings et al. [68].

Multi-stage stochastic programming is another tool used for the handling of uncertainty within an MPC. In this stochastic MPC paradigm, the expected performance of the controller across multiple possible plant scenarios is optimized, rather than a particular (often worst-case) realization. While the optimization of the expected behavior of the MPC differs from the strict definition of robust MPC, it is often termed as a sub-type of robust MPC as it is still an uncertainty handling technique. Muñoz de la Peña et al. [63] present a scheme of this type, where the uncertainty is discretized and a scenario tree generated. With linear dynamic models, the problem becomes a linear program (LP) or quadratic program (QP), depending on whether an infinity-norm, 1-norm or squared Euclidean norm is used in the cost function. Lucia et al. [44] apply a nonlinear MPC (NMPC) algorithm of this type to a semi-batch polymerization reactor. They explore the impact of the robust horizon, defined as the number of stages for which branching of the scenario tree is considered, and compare the performance of standard, multi-stage and min-max

NMPC. A potential drawback of the multi-stage stochastic MPC approach is that the scenario tree grows exponentially with the number of stages, the number of uncertain parameters, and the number of discrete parameter realizations. This has led to several approaches focused on improving computational efficiency, while retaining performance advantages offered by this robust MPC scheme. Lucia et al. [45] propose a scenario decomposition approach in which the non-anticipativity constraints on the inputs at each node of the scenario tree are relaxed, permitting independent solution of the uncertainty scenario subproblems, with iteration until the non-anticipativty constraints are satisfied. Mastragostino et al. [50] obtain good performance using two-stage stochastic programming in robust MPC to approximate future feedback action in an application to supply chain operation. They consider, in addition, discrete decision variables that arise in production scheduling. Marti et al. [49] compare the performance of different scenario decomposition approaches, including bundle decomposition based on subsets of scenarios. Krishnamoorthy et al. [34] propose a scenario decomposition scheme for scenario-based robust MPC based on primal decomposition in which the non-anticipativity constraints are always feasible. Holtorf et al. [23] consider a scenario-based approach to robust NMPC, in which the scenario tree, based on worst-case uncertain parameter realizations, is adaptively generated at each time step based on a constraint sensitivity analysis. Krishnamoorthy et al. [36] propose the use of principal component analysis (PCA) for the selection of scenarios in multi-stage robust MPC, which allows for the relationship between uncertain parameters to be accounted for in order to reduce conservatism.

One of the advantages of the multi-scenario method is that it computes future

control moves under a range of future possible plant conditions. This can greatly improve the performance of the controller in the presence of plant model mismatch. However, beyond the robust horizon, the branching action of the multi-scenario MPC ends and feedback response to uncertain plant behavior past that time step is not captured. This is a significant limitation as the computational requirements of the multi-scenario method increase exponentially with the robust horizon. The approach proposed in this chapter is scenario-based, but follows a somewhat different paradigm in that the predicted closed-loop response of uncertain plant realizations under nominal MPC is utilized, thereby extending the functional length of control actions under uncertain plant behavior. This draws on the work of Jamaludin and Swartz [26, 27] where embedded MPC optimization problems are included in a dynamic real-time optimization (DRTO) problem. The robust MPC approach proposed in this article uses such embedded MPC subproblems to predict the MPC response under uncertain parameter realizations in a multi-scenario framework. Closed-loop dynamics are considered throughout the prediction horizon, and the number of branches scales linearly with the number of scenarios.

2.2 Problem Formulation

The robust MPC approach used here is a multi-scenario approach where the scenarios, other than the nominal, contain embedded MPC subproblems to generate closed-loop predictions of the plant response under nominal MPC. This is accomplished in the following way. For each uncertain plant realization, a closed-loop response is generated under the action of the standard MPC algorithm utilizing the nominal model. The control input at every time step is applied to the plant model realization

to generate a predicted response which serves as a surrogate measurement to its associated MPC to calculate the next input, and the sequence is repeated. A closed-loop response under nominal MPC is thus generated for each uncertain plant realization with the exception of the nominal plant, for which an open-loop response is utilized as in the standard MPC formulation.

A visualization of this process can be seen in Figure 2.1, shown for three uncertainty scenarios. The initial inputs to all scenarios are forced to be the same through the imposition of non-anticipativity constraints.

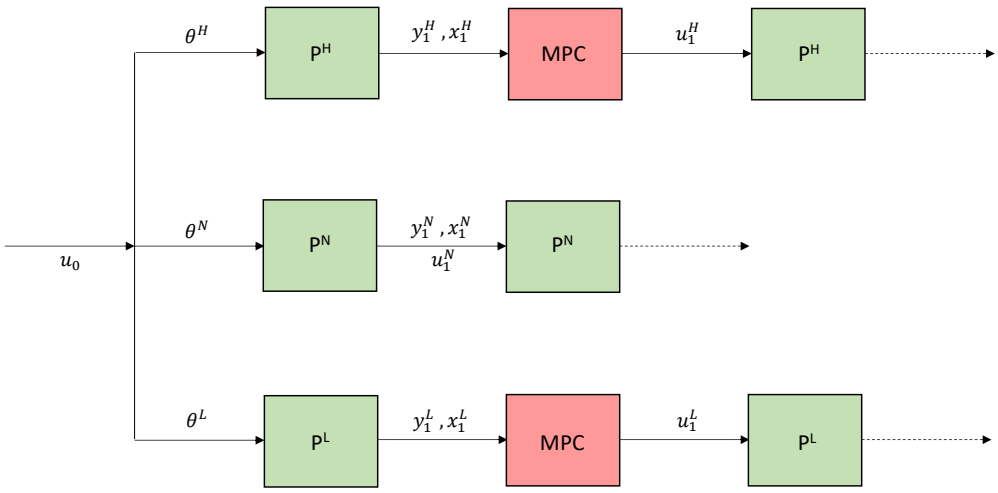

Figure 2.1: Visualization of embedded closed-loop control action within a robust MPC problem. The P blocks represent plant models associated with uncertain parameter realizations; the MPC blocks represent MPC calculations using the nominal model; the subscripts are the time step; the superscripts are the scenario (low, nominal, high); θ is the uncertain parameter.

The overall objective of the robust MPC is similar to that of a standard MPC. It seeks to minimize a combined target tracking and input move suppression performance objective by manipulating the input moves and predicting the resulting

output trajectories. In this case, however, it optimizes a weighted performance objective over the uncertain plant realizations. Choosing the weights as the probability of occurrence of the scenarios yields the expected performance over the uncertainty set; however, other choices such as nominal performance can be selected. Additionally, the input moves beyond the first time step of the non-nominal scenario models are computed by the inner MPC subproblems, and therefore do not constitute optimization degrees of freedom. Therefore, the robust MPC problem seeks to optimize the composite performance by varying the first input move of all the scenarios and the subsequent input moves of the nominal model scenario, with the inputs of the other plant scenarios (beyond the initial time step) determined by the inner MPC subproblems. Input, state and output constraints are also imposed as required.

The robust MPC interacts with the real plant in much the same way that any standard MPC would. It determines one input move which will be implemented on the plant and uses the current measured value of the output variables to estimate the disturbance for the next MPC problem. A key difference is that a disturbance estimate will be calculated for each of the scenario models; they will compute the predicted outputs and compare these to the actual measured output, resulting in a disturbance estimate. The embedded MPCs, on the other hand, generate a disturbance estimate by comparing the prediction made by the inner MPC model to the prediction made by the associated plant scenario model, which serves as a surrogate plant measurement. This is done so that the inner MPC executions can have an updated disturbance estimate in accordance with the scenario model being used, which simulates an MPC updating its disturbance estimate for each execution in accordance with the most

recent plant measurement. In other words, for the inner MPC executions and for the purposes of disturbance estimation, the outer MPC scenario model predictions represent the plant measurement while the inner MPC represents an MPC execution.

Let $\theta \in \Re^p$ denote the vector of uncertain parameters, N_s denote the number of uncertainty scenarios indexed by i, and let $i = N$ represent the scenario corresponding to the nominal model parameters. Then the robust MPC problem described above can be represented mathematically as follows:

$$\min \quad \phi = \sum_{i=1}^{N_s} w^i \varphi^i(x^i, y^i, u^i) \tag{2.2.1a}$$

s.t.

$$x^i_{j+1} = f_p(x^i_j, u^i_j, \theta^i) \quad j = 0, \ldots, p-1 \quad i = 1, \ldots, N_s \tag{2.2.1b}$$

$$y^i_j = g_p(x^i_j, \theta^i) \quad j = 1, \ldots, p \quad i = 1, \ldots, N_s \tag{2.2.1c}$$

$$0 \leq h_p(x^i_j, y^i_j, u^i_j) \quad j = 0, \ldots, p \quad i = 1, \ldots, N_s \tag{2.2.1d}$$

$$u^i_j = f_c(y^i_j, \theta^N) \quad j = 1, \ldots, p-1 \quad i = 1, \ldots, N_s, \quad i \neq N \tag{2.2.1e}$$

In the above, φ^i represents the performance objective associated with scenario i with associated weight w^i. $x^i_j \in \Re^{n_x}$, $y^i_j \in \Re^{n_y}$, and $u^i_j \in \Re^{n_u}$ are the states, outputs and inputs, respectively, at time step j, defining the dynamic response of scenario i. For $i = N$, this corresponds to the open-loop response; it otherwise represents a closed-loop response with the inputs u^i generated by the solution of an inner MPC subproblem using the nominal model, defined by the mapping in Eq. (2.2.1e). x^i is a composite vector of the x^i_j, with y^i and u^i similarly defined. f_p and g_p are mappings that define the open-loop dynamic model, and h_p represents constraints

on the states, outputs and inputs, as well as non-anticipativity constraints to ensure that all scenarios have the same initial input.

We remark that the proposed robust MPC scheme reduces to the standard MPC formulation under the assumption of a nominal plant with no uncertainty scenarios.

In the next two subsections, we describe the formulation in more detail, where for simplicity we confine the description to linear dynamic systems. We also assume that the nominal and uncertain plants are open-loop stable, and follow the QDMC formulation of MPC [18, 46]. In Section 2.2.1 we focus on Eqs. (2.2.1a)-(2.2.1d), and in Section 2.2.2 we describe the inner MPC subproblems represented by Eq. (2.2.1e). In Section 2.2.3 we describe how the integrated multilevel optimization problem is solved.

2.2.1 Primary MPC

The primary MPC contains the multiple scenarios necessary for this approach. It is similar in design to previous multi-scenario methods, but with a robust horizon of 1. The single branching point is all that is required since the embedded MPC problems provide all the necessary future predicted MPC responses. As the inner MPC subproblems are solved at every time step, all future control actions (up to the prediction horizon) are therefore predicted, negating the need for further branching. Using the scenario models involving the extreme parameter realizations to propagate the plant prediction for the non-nominal scenarios allows for the extremes of the uncertain model to continue to be considered without the need for additional branching. However, the extreme parameter realizations do not necessarily correspond to the extremes of the plant behavior. The behavior of

the plant could respond non-linearly to the change in parameter value, potentially causing an interior value of the uncertain parameter to correspond to extreme plant behavior. Alternatively, plant-model mismatch has the potential to cause the most extreme possible plant behavior to be outside the range of scenarios considered by the MPC. In these cases, the proposed robust MPC formulation would not guarantee feasibility or constraint satisfaction, but would still be expected to outperform a nominal MPC.

A more detailed expression of Eqs. (2.2.1a)-(2.2.1d) for the linear dynamic plant case is as follows.

$$\min_{u_0,u_j^N} \quad \phi = w^N \left(\begin{array}{c} \sum_{j=1}^{p} (y_j^N - y_j^{\text{sp}})^T Q (y_j^N - y_j^{\text{sp}}) \\ + \sum_{j=0}^{m-1} (\Delta u_j^N)^T R (\Delta u_j^N) \\ + \sum_{j=0}^{m-1} (u_j^N - u_j^{\text{sp}})^T S (u_j^N - u_j^{\text{sp}}) \end{array} \right) + \sum_{\substack{i=1 \\ i \neq N}}^{N_s} w^i \left(\begin{array}{c} \sum_{j=1}^{p} (y_j^i - y_j^{\text{sp}})^T Q (y_j^i - y_j^{\text{sp}}) \\ + \sum_{j=0}^{p-1} (\Delta u_j^i)^T R (\Delta u_j^i) \\ + \sum_{j=0}^{p-1} (u_j^i - u_j^{\text{sp}})^T S (u_j^i - u_j^{\text{sp}}) \end{array} \right) \tag{2.2.2a}$$

$$\text{s.t.} \quad x_{j+1}^i = A^i x_j^i + B^i u_j^i, \quad j = 0, \ldots, m-1 \quad i = N \tag{2.2.2b}$$

$$x_{j+1}^i = A^i x_j^i + B^i u_{m-1}^i, \quad j = m, \ldots, p-1 \quad i = N \tag{2.2.2c}$$

$$x_{j+1}^i = A^i x_j^i + B^i u_j^i, \quad j = 0, \ldots, p-1 \quad i \neq N \tag{2.2.2d}$$

$$y_j^i = C^i x_j^i + d_j^i, \quad j = 1, \ldots, p \quad i = 1, \ldots, N_s \tag{2.2.2e}$$

$$\Delta u_j^i = u_j^i - u_{j-1}^i, \quad j = 0, \ldots, p-1 \quad i = 1, \ldots, N_s \tag{2.2.2f}$$

$$u_{\min} \leq u_j^i \leq u_{\max}, \quad j = 0, \ldots, p-1 \quad i = 1, \ldots, N_s \tag{2.2.2g}$$

$$y_{\min} \leq y_j^i \leq y_{\max}, \quad j = 1, \ldots, p \quad i = 1, \ldots, N_s \tag{2.2.2h}$$

In the above, ϕ is the objective function; u represents the vector of inputs, x represents the states, y represents the outputs; w is the weight of each scenario; p is the prediction horizon, m is the control horizon; Δu is the change in input; Q, R, S are diagonal weighting matrices for the output deviation, input move, and input deviation terms, respectively; $y^{\text{sp}}, u^{\text{sp}}$ are the output and input set-points, respectively; A, B, C are the state space model matrices; d is the disturbance estimate; $y_{\min}, y_{\max}, u_{\min}, u_{\max}$ are the constraint limits on the outputs and inputs.

The output constraints can alternatively be treated as soft constraints with the slack variables, $e^i_{\text{min,j}}$ and $e^i_{\text{max,j}}$, added to the inequality constraints. A weighted sum of squares of the slack variables is added to the previous objective function, ϕ, to form an adjusted objective function, $\hat{\phi}$, to keep the constraint violation at or close to zero. This can be expressed as follows,

$$\hat{\phi} = \phi + \sum_{i=1}^{N_s} w^i \sum_{j=1}^{p} \left[(e^i_{\text{min,j}})^T \Omega_{\min} (e^i_{\text{min,j}}) + (e^i_{\text{max,j}})^T \Omega_{\max} (e^i_{\text{max,j}}) \right] \tag{2.2.3a}$$

$$y_{\min} - e^i_{\text{min,j}} \leq y^i_j \leq y_{\max} + e^i_{\text{max,j}}, \quad j = 1, \ldots, p \quad i = 1, \ldots, N_s \tag{2.2.3b}$$

$$0 \leq e^i_{\text{min,j}}, e^i_{\text{max,j}}, \quad j = 1, \ldots, p \quad i = 1, \ldots, N_s \tag{2.2.3c}$$

where $\Omega_{\min}$ and $\Omega_{\max}$ are diagonal weighting matrices associated with the minimum and maximum constraints, respectively. The nominal scenario MPC respects the specified control horizon and halts control action beyond that time step. However, the other model scenarios compute inputs for all time steps up to the prediction horizon, not the control horizon. The reason for this is that these specified inputs are not true degrees of freedom within the primary MPC optimization problem; they are solutions to the inner MPC subproblems which are solved algebraically. Therefore,

the only optimization degrees of freedom are the inputs at the first time step and the subsequent inputs along the nominal MPC scenario up to the control horizon. The objective function reflects this difference in horizons between the scenarios, with the nominal scenario including a more standard setup with the output deviation being considered through the prediction horizon and the input move and input deviation terms being considered only for the control horizon. The other scenarios instead include the input move and deviation terms for the entire prediction horizon. This difference in horizons is why the objective function is shown as two separate summations, one for the nominal scenario and one for the other scenarios. The different scenarios are weighted in the objective function, in order to balance the relative importance or expected frequency of occurrence of the scenarios.

The disturbance estimate used here is assumed constant over the prediction horizon and is set as the difference between the last measured output value, y^{m}, and the corresponding predicted output, as in the DMC and QDMC formulations used in Cutler and Ramaker [11] and Garcia and Morshedi [18], respectively.

$$d_0^i = y^{\mathrm{m}} - C^i x_0^i, \quad i = 1, \ldots, N_s \tag{2.2.4}$$

$$d_j^i = d_0^i, \quad j = 1, \ldots, p \;\; i = 1, \ldots, N_s \tag{2.2.5}$$

The current predicted states are estimated using the MPC scenario models, the states used as initial conditions in the previous MPC iteration, and the implemented inputs from the previous time step (previous states and inputs are denoted with a -1

subscript).

$$x_0^i = A^i x_{-1}^i + B^i u_{-1}^i, \quad i = 1, \ldots, N_s \tag{2.2.6}$$

Importantly, in order for the MPC to give one input move to the plant for the current time, the input moves across the MPC scenarios must be the same for the first time step.

$$u_0^i = u_0, \quad i = 1, \ldots, N_s \tag{2.2.7}$$

2.2.2 Inner MPC

The inner, or embedded, MPCs simulate future control action corresponding to the uncertain plant scenarios. The objective function of each inner MPC subproblem corresponds to the nominal term of the primary MPC objective function. The inner MPCs are solved for each non-nominal scenario at each time step in the prediction horizon, and as with standard MPC, the solution that is implemented (in this case, applied to the associated scenario plant model) is the first input along the internal input move horizon. The inner MPC subproblems are not solved for the nominal scenario of the primary MPC because doing so would add no new information as the primary and inner MPCs would use the same model. The closed-loop response generated by the inner MPCs would therefore coincide with that determined by the open-loop calculation for the nominal scenario of the primary MPC, except for possible minor discrepancies due to the finite MPC horizon length.

The states and outputs are predicted within the inner MPC using the nominal scenario MPC model. As in the primary MPC, the change in input is simply

calculated as the difference in input value between successive time steps. Output constraints may also be implemented in the inner MPC, again either as soft or hard constraints. The inner MPC subproblem for scenario i, $i \neq N$, with hard constraints, takes the following form for each step $j = 1, \ldots, p-1$:

$$\min_{\tilde{u}^i_{j,k}} \quad \tilde{\phi}^i_j = \sum_{k=1}^{p} (\tilde{y}^i_{j,k} - y^{\text{sp}}_{j+k})^T Q (\tilde{y}^i_{j,k} - y^{\text{sp}}_{j+k}) + \sum_{k=0}^{m-1} (\Delta \tilde{u}^i_{j,k})^T R (\Delta \tilde{u}^i_{j,k}) + \sum_{k=0}^{m-1} (\tilde{u}^i_{j,k} - u^{\text{sp}}_{j+k})^T S (\tilde{u}^i_{j,k} - u^{\text{sp}}_{j+k}) \tag{2.2.8a}$$

$$\text{s.t.} \quad \tilde{x}^i_{j,k+1} = A^N \tilde{x}^i_{j,k} + B^N \tilde{u}^i_{j,k}, \quad k = 0, \ldots, m-1 \tag{2.2.8b}$$

$$\tilde{x}^i_{j,k+1} = A^N \tilde{x}^i_{j,k} + B^N \tilde{u}^i_{j,m-1}, \quad k = m, \ldots, p-1 \tag{2.2.8c}$$

$$\tilde{y}^i_{j,k} = C^N \tilde{x}^i_{j,k} + \tilde{d}^i_{j,k}, \quad k = 1, \ldots, p \tag{2.2.8d}$$

$$\Delta \tilde{u}^i_{j,k} = \tilde{u}^i_{j,k} - \tilde{u}^i_{j,k-1}, \quad k = 0, \ldots, m-1 \tag{2.2.8e}$$

$$u_{\min} \leq \tilde{u}^i_{j,k} \leq u_{\max}, \quad k = 0, \ldots, m-1 \tag{2.2.8f}$$

$$y_{\min} \leq \tilde{y}^i_{j,k} \leq y_{\max}, \quad k = 1, \ldots, p \tag{2.2.8g}$$

The tilde (~) denotes a variable corresponding to an embedded MPC subproblem, distinct from the primary MPC variables, but with the same model meaning (ie. $\tilde{u}$ is the vector of inputs for the inner MPC subproblem, just as u is the vector of inputs for the primary MPC problem). Note that some of the parameters used in this problem are the same as those of the primary MPC problem.

It should be noted that the first subscript for the inner MPC input, output, and state variables corresponds to the time step of the primary MPC problem, or, in other words, to the time position when this inner MPC subproblem is being executed. The second subscript is the internal time step, or the time step of the inner MPC

subproblem.

The inner MPC disturbance estimate uses the scenario model output as the simulated measured output and its internal nominal model to generate its predicted output. As in the primary MPC, the disturbance is assumed constant over the internal MPC prediction horizon, and is given by

$$\tilde{d}^i_{j,0} = y^i_j - C^N \tilde{x}^i_{j,0}, \quad i \neq N \quad j = 1, \ldots, p-1 \tag{2.2.9}$$

$$\tilde{d}^i_{j,k} = \tilde{d}^i_{j,0}, \quad i \neq N \quad k = 1, \ldots, p \quad j = 1, \ldots, p-1 \tag{2.2.10}$$

The value of the predicted state for the inner MPC at $k = 0$ corresponds to the state at time step $k = 1$ of the previous inner MPC subproblem. For the first inner MPC execution, the current predicted states are taken as the predicted states at time $j = 1$ within the nominal model scenario of the primary MPC.

$$\tilde{x}^i_{1,0} = x^N_1, \quad i \neq N \tag{2.2.11}$$

$$\tilde{x}^i_{j,0} = \tilde{x}^i_{j-1,1}, \quad i \neq N \quad j = 2, \ldots, p-1 \tag{2.2.12}$$

Similarly, the input that the inner MPC considers to be the most recent implemented input (required for $k = 0$ in Eq. (2.2.8e) is simply the computed first input move from the previous inner MPC execution. For the first inner MPC execution, the first input move of the primary MPC is used.

$$\tilde{u}^i_{1,-1} = u^i_0, \quad i \neq N \tag{2.2.13}$$

$$\tilde{u}^i_{j,-1} = \tilde{u}^i_{j-1,0}, \quad i \neq N \quad j = 2, \ldots, p-1 \tag{2.2.14}$$

Finally, the purpose of using the embedded MPC subproblems is to generate a predicted closed-loop response for the scenario plant models. Therefore, the first input move from each inner MPC is applied to the scenario plant models at the primary MPC level in at the appropriate point the prediction horizon via Eq. (2.2.2d):

$$u_j^i = \tilde{u}_{j,0}^i, \quad i \neq N \quad j = 1, \ldots, p-1 \tag{2.2.15}$$

2.2.3 Solution Strategy

The primary and inner MPC optimization problems are connected through Eq. (2.2.9) in which the primary problem provides surrogate measurements to the inner MPC subproblems, Eq. (2.2.15) which provides predicted future control inputs to the scenario plant models in the primary problem, and Eqs. (2.2.11) and (2.2.13) which provide initial conditions for the first inner MPC execution. Eqs. (2.2.2) and (2.2.4)-(2.2.15) comprise a composite multilevel optimization problem which needs to be solved to generate the robust MPC input that is applied to the actual plant.

This multilevel programming problem could be solved sequentially, where the primary MPC problem and inner MPC subproblem alternate in their solutions, each giving information to the other. However, in this chapter, the simultaneous approach presented in Jamaludin and Swartz [28] is used instead. Such an approach allows for the multilevel problem to be solved as a single optimization problem, simplifying the overall simulation. This is accomplished by reformulating the inner MPC subproblems as their corresponding first-order Karush-Kuhn-Tucker (KKT) conditions as proposed in Baker and Swartz [2]. These conditions are represented simply as algebraic equations, and so can be included in the primary MPC problems

as constraints. Additionally, since the inner MPC subproblems are convex QPs, the first-order KKT conditions are necessary and sufficient for optimality. The reformulation results in a mathematical program with complementarity constraints (MPCC), which can be, and is here, solved using an exact penalty approach.

The general form of a QP, of which type the inner MPC subproblems are, can be written in the following form.

$$\min_{z} \quad \frac{1}{2} z^T H z + g^T z \tag{2.2.16}$$

$$s.t. \quad Az = b \tag{2.2.17}$$

$$z \geq 0 \tag{2.2.18}$$

The KKT conditions which correspond to this generalized problem may be written as:

$$Hz + g - A^T \boldsymbol{\lambda} - \boldsymbol{\nu} = 0 \tag{2.2.19}$$

$$Az - b = 0 \tag{2.2.20}$$

$$z_i \nu_i = 0, \quad \forall i \tag{2.2.21}$$

$$(z, \boldsymbol{\nu}) \geq 0 \tag{2.2.22}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ are the Lagrange multipliers for the equality and inequality constraints, respectively.

The KKT conditions of a standard MPC problem may be found in Baker and Swartz [2] and Jamaludin and Swartz [28]. The latter additionally shows the MPC as a subproblem of a larger optimization problem, in a similar manner to how it

is used here. The only substantial changes made to those previous formulations is the addition of output constraints. This simply results in additional minimum and maximum inequality constraints for the outputs which are similar in form to those used for the inputs. Also, the addition of slack variables for the soft output constraints require the addition of appropriate Lagrange gradients and primal feasibility constraints for these variables.

The complementarity constraints created by presenting the inner MPC subproblems as their KKT conditions are handled here with an exact penalty approach [66, 29]. The product of the primal and dual KKT variables are summed and multiplied by an appropriate weighting factor and then included in the objective function.

$$\rho \sum_i z_i \nu_i \tag{2.2.23}$$

With a sufficiently large weight, ρ, this ensures the complementarity constraints are satisfied. The resulting problem may then be solved by a standard NLP solver.

2.3 Case Studies

2.3.1 Case Study 1: SISO First-Order Plus Dead Time System

The first case study investigated is a simple Single Input Single Output (SISO) system represented by the following first-order plus dead time transfer function,

$$G(s) = \frac{K}{5s+1}e^{-\theta s} \tag{2.3.1}$$

The purpose of such a basic system as a case study is to demonstrate the viability of the proposed MPC formulation for a simple approximation of dynamics that is nevertheless widely used. It also allows for a tightly controlled environment in which to investigate the effects of changing circumstances on the robust MPC formulation. The MPC parameters for this case study are shown in Table 2.1.

The output constraints can be included in the above problem formulation at the primary MPC or inner MPC level. For this case study, the output constraints are included only in the primary MPC, and are not activated for the inner MPC subproblems. These output constraints are formulated as soft constraints, with the soft constraint penalty term shown in Table 2.1. This soft constraint formulation is also used in the nominal MPC with the same penalty term in order to properly compare the two methods. Soft constraints are used partially to ensure the optimization problems return solutions and do not simply state that the problem is infeasible, which can happen in the case of plant-model mismatch if hard constraints are violated at the plant level but not predicted by the controller. Additionally, hard

Table 2.1: MPC parameters for Case Study 1.

Parameter	**Description**	**Value**
y^{sp}	Output set-point	2
$y_{\min}$	Output constraint - minimum	0
$y_{\max}$	Output constraint - maximum	2.1
$u_{\min}$	Input constraint - minimum	-100
$u_{\max}$	Input constraint - maximum	100
p	Prediction horizon	10
m	Control horizon	2
Q	Output tracking weight	10
R	Move suppression weight	1
S	Input tracking weight	0
N	Simulation horizon	50
ρ	Complementarity Penalty	1000
$\Omega_{\min}$, $\Omega_{\max}$	Soft constraint penalty	200
K_{nom}	Nominal transfer function gain	0.4
$K_{\min}$	Minimum transfer function gain	0.2
$K_{\max}$	Maximum transfer function gain	0.6
θ_{nom}	Nominal transfer function dead time	2
$\theta_{\min}$	Minimum transfer function dead time	0
$\theta_{\max}$	Maximum transfer function dead time	4

constraints, if desired, can be approximated by soft constraints by using a very large penalty term, while still keeping the optimization problem feasible.

In order to fully investigate the performance of the MPC formulation, five subcases are performed with variations in the above parameters. In each subcase, the robust MPC formulation presented here is compared to a nominal MPC in three potential plant scenarios, one with the uncertain parameter at the nominal value, one at the minimum value, and one at the maximum. Each subcase also compares the robust MPC to a nominal MPC under 20 different random plant scenarios where the uncertain parameter is between the minimum and maximum values for that parameter. Then the average performance metrics for these random

plants are compared. Three scenarios are used in the robust MPC as it corresponds to an appropriate balance of performance and computation time for this (and the subsequent) case study. It was found that, for these case studies, increasing the number of scenarios did not significantly improve the controller performance but did increase the computation time. However, other applications with potentially more nonlinear behavior may benefit from including additional scenarios.

Subcase 1: Unconstrained, uncertain gain

The first subcase involves uncertainty in the gain and no output constraints. It also uses three scenarios for the robust MPC, one with the nominal gain, one with the minimum gain, and one with the maximum gain. Plots of the plant trajectories are shown in Figure 2.2 and a summary of the sum of squared errors, which is used to evaluate the MPC performance, is shown in Tables 2.2 and 2.3. As can be seen from Figure 2.2, the responses appear similar for the nominal and robust MPCs when applied to the nominal and minimum gain plants, but the robust MPC noticeably outperforms the nominal MPC in the maximum plant gain case. In terms of deviation from the set-point, the sum of squared errors (SSE) is less in each scenario for the robust MPC than the nominal MPC, showing that the robust MPC is approaching the set-point more quickly and/or reducing overshoot of the set-point. Therefore, even in the nominal gain plant scenario, the robust MPC outperforms the nominal. However, the computation time for the robust MPC is higher than for the nominal, which is the result of the optimization problem which must be solved being significantly more complex. Specifically, the average solution time for the robust MPC in this subcase was 0.915 seconds for one MPC execution, compared to 0.013 seconds for the nominal

MPC. The longer computation time is still substantially less than the MPC sample time used in typical chemical process applications [64]. The solution times for the subsequent subcases in this case study are of a similar order as those described here.

Table 2.2: Unconstrained, uncertain gain subcase sum of squared errors (SSE).

Case	**Nominal**	**Robust**
Nominal	17.3254	16.1128
Minimum	27.1568	25.6982
Maximum	15.3896	14.0759

Table 2.3: Unconstrained, uncertain gain subcase statistics for 20 plants with random gain.

Metric	**Nominal**	**Robust**
SSE Mean	17.5499	16.2749
SSE Standard Deviation	2.9819	2.9620

Subcase 2: Constrained, uncertain gain

The second subcase is identical to the first but with the output constrained according to the constraints stated in Table 2.1. The results, shown in Figure 2.3 and Tables 2.4 and 2.5, are similar to the first subcase. The nominal and minimum gain plant results are especially similar as neither the robust nor the nominal MPC violates the constraints for these plants even in the unconstrained subcase. However, in the maximum gain plant, the nominal MPC has a larger overshoot and therefore significantly violates the maximum output constraint. The robust MPC violates the constraint by a nearly insignificant margin and for a shorter time period, thus outperforming the nominal MPC in terms of reducing constraint violation. The SSE for the robust MPC also remains less than that of the nominal MPC for all plants.

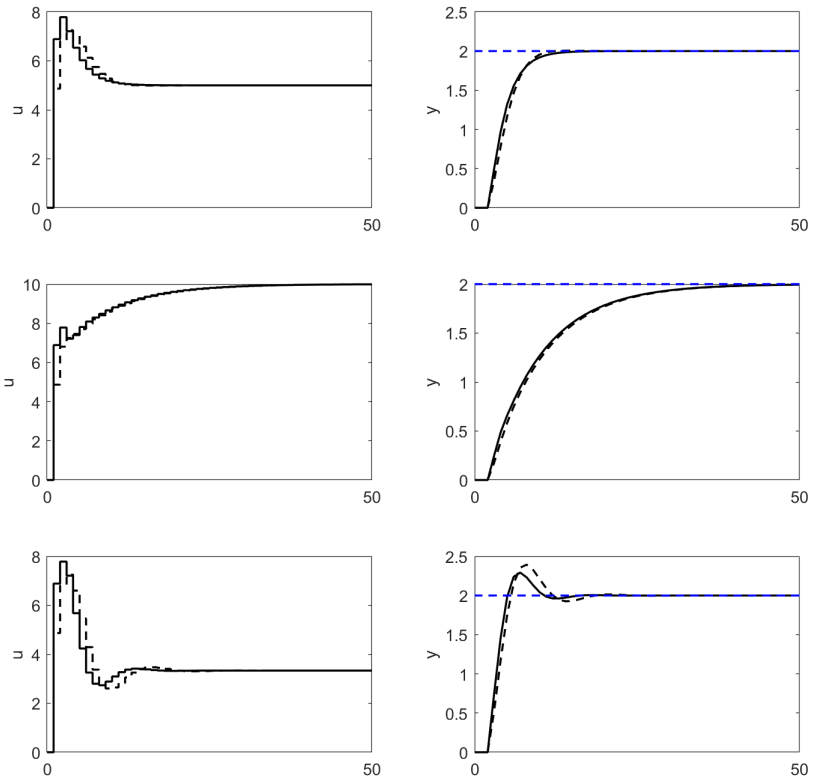

Figure 2.2: Input and output trajectories for the uncertain gain and unconstrained SISO subcase, for the nominal plant gain (top), minimum plant gain (middle), and maximum plant gain (bottom). Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point.

Subcase 3: Unconstrained, uncertain dead time

The third case study is also similar to the first, but with the dead time as the uncertain parameter rather than the gain. The output is unconstrained. As with the gain, the three scenarios for the robust MPC use the nominal dead time, minimum dead time, and maximum dead time. The three plants also use the nominal, minimum, and maximum dead time. The results for this subcase are shown in Figure 2.4 and Tables 2.6 and 2.7. The robust MPC has a faster initial response than the nominal. As a result, the SSE is markedly lower for the robust MPC across all scenarios,

Table 2.4: Constrained, uncertain gain subcase metrics.

Case	**Metric**	**Nominal**	**Robust**
Nominal	SSE	17.6467	16.9346
	Constraint Violation	0	0
Minimum	SSE	27.8114	27.0991
	Constraint Violation	0	0
Maximum	SSE	15.3545	14.3245
	Constraint Violation	0.0995	0.0026

Table 2.5: Constrained, uncertain gain subcase statistics for 20 plants with random gain.

Metric	**Nominal**	**Robust**
SSE Mean	17.5499	16.2749
SSE Standard Deviation	2.9819	2.9620
Constraint Violation Mean	0.0067	8.2810e-8
Constraint Violation Standard Deviation	0.0167	3.034e-7

demonstrating the superiority of the robust MPC over the nominal MPC for this subcase in terms of set-point deviation.

Table 2.6: Unconstrained, uncertain dead time subcase sum of squared errors (SSE).

Case	**Nominal**	**Robust**
Nominal	17.3254	15.5694
Minimum	11.7638	10.2638
Maximum	27.5550	25.2032

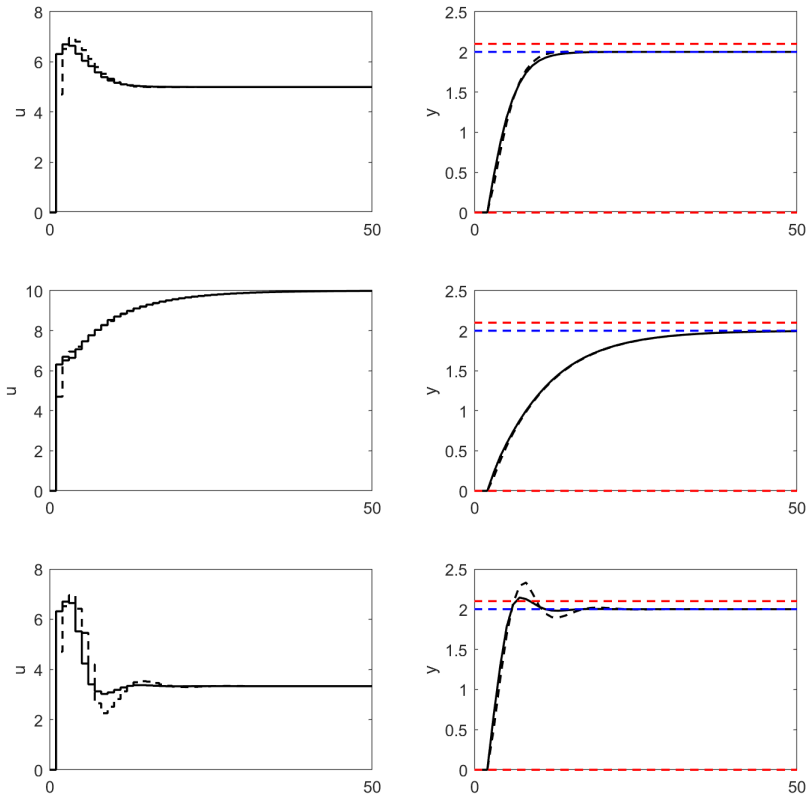

Figure 2.3: Input and output trajectories for the uncertain gain and constrained SISO subcase, for the nominal plant gain (top), minimum plant gain (middle), and maximum plant gain (bottom). Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point; Red dashed line: constraints.

Subcase 4: Constrained, uncertain dead time

The next subcase which is investigated involves uncertain dead time and constrained output according to the constraints in Table 2.1. The results are shown in Figure 2.5 and Tables 2.8 and 2.9. The robust MPC is more conservative in the nominal and minimum dead time plants, more noticeably for the nominal plant. However, the nominal MPC significantly violates the upper bound output constraint in the maximum dead time plant, whereas the robust MPC shows only a slight violation. By largely avoiding the constraint violation in this plant scenario, the robust MPC

Table 2.7: Unconstrained, uncertain dead time subcase statistics for 20 plants with random dead time.

Metric	**Nominal**	**Robust**
SSE Mean	17.5639	15.7276
SSE Standard Deviation	4.9322	4.6512

also reduces oscillations. However, it does show slight oscillations in the minimum and nominal dead time plant scenarios during approach to the set-point. As a result, the SSE for the robust MPC is similar to that of the nominal MPC in the minimum and nominal scenarios (lower in the nominal and higher in the minimum), but substantially lower in the maximum dead time scenario. For a random value of the dead time, the robust MPC, on average, outperforms the nominal MPC in terms of set-point deviation, as seen in Table 2.9. As with the constrained, uncertain gain subcase, the robust MPC significantly reduces the constraint violation in the maximum scenario compared to the nominal MPC.

Table 2.8: Constrained, uncertain dead time subcase metrics.

Case	**Metric**	**Nominal**	**Robust**
Nominal	SSE	17.6467	17.6230
	Constraint Violation	0	0
Minimum	SSE	12.0591	12.2667
	Constraint Violation	0	0
Maximum	SSE	27.3839	24.0755
	Constraint Violation	0.9455	6.3085e-4

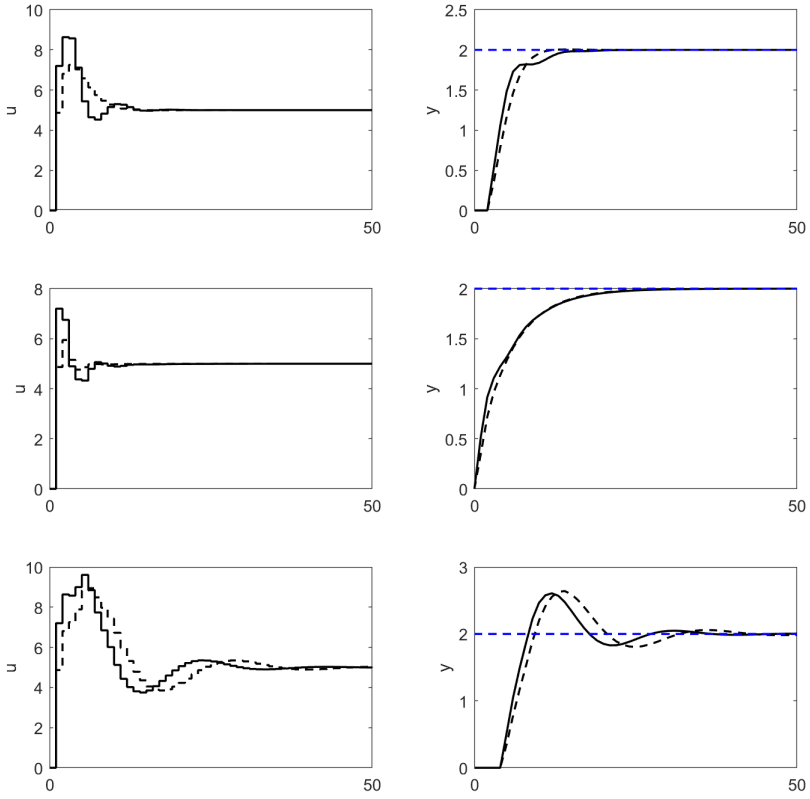

Figure 2.4: Input and output trajectories for the uncertain dead time and unconstrained SISO subcase, for the nominal plant dead time (top), minimum plant dead time (middle), and maximum plant dead time (bottom). Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point.

Subcase 5: Unconstrained, uncertain dead time with adjusted parameters for instability

The final subcase investigated considers a different set of plant parameters to demonstrate the ability of the robust MPC to maintain stability in cases where the nominal MPC is unable to do so. The plant considered here has a nominal gain of 0.4 as before, but a time constant of 2 and nominal dead time of 3. Uncertainty in the dead time is considered with a minimum value of 1 and a maximum of 5. Additionally, the input move suppression weight (R) and the prediction horizon (p)

Table 2.9: Constrained, uncertain dead time subcase statistics for 20 plants with random dead time.

Metric	**Nominal**	**Robust**
SSE Mean	17.8781	17.3771
SSE Standard Deviation	4.3030	3.3041
Constraint Violation Mean	0.0892	1.9435e-5
Constraint Violation Standard Deviation	0.2240	8.6914e-5

are chosen as 0.1 and 8, respectively. It is shown in Figure 2.6 and Tables 2.10 and 2.11 that with these parameters, the nominal MPC is unstable in both the minimum and maximum plant dead time scenarios while the robust MPC is stable. In the nominal plant dead time scenario, the nominal MPC outperforms the robust MPC substantially. This is to be expected since the robust MPC is attempting to stabilize all three scenarios while the nominal MPC is free to aggressively reach the output set-point. However, that same characteristic results in instability in the minimum and maximum dead time plant scenarios for the nominal MPC.

Table 2.10: Unconstrained, uncertain dead time with adjusted parameters for instability subcase sum of squared errors (SSE).

Case	**Nominal**	**Robust**
Nominal	16.387	17.620
Minimum	355570	10.994
Maximum	512.23	30.757

Table 2.11: Unconstrained, uncertain dead time with adjusted parameters for instability subcase statistics for 20 plants with random dead time.

Metric	**Nominal**	**Robust**
SSE Mean	10378	18.079
SSE Standard Deviation	21713	5.6920

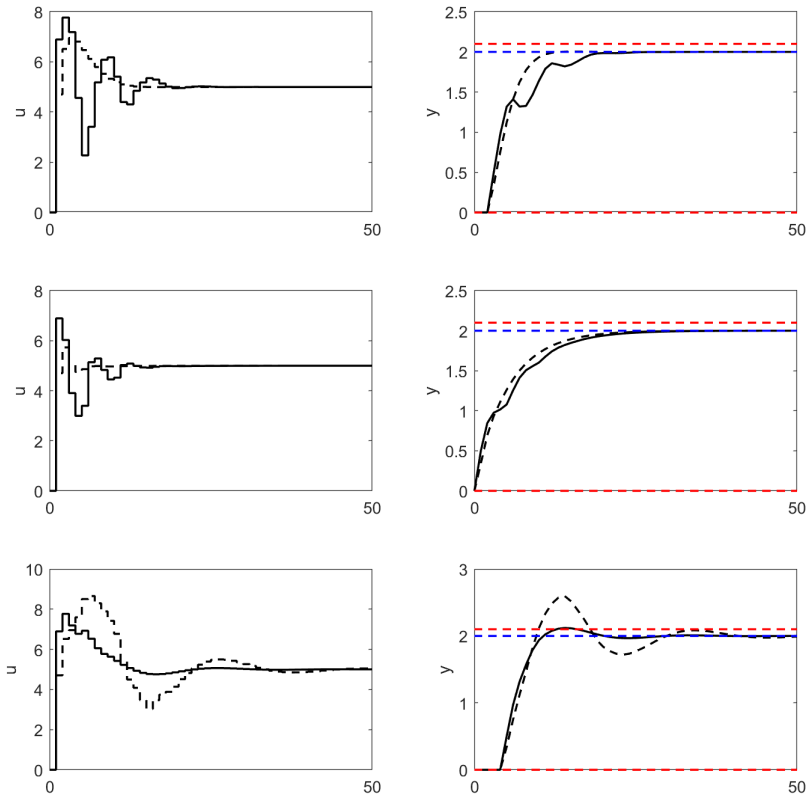

Figure 2.5: Input and output trajectories for the uncertain dead time and constrained SISO subcase, for the nominal plant dead time (top), minimum plant dead time (middle), and maximum plant dead time (bottom). Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point; Red dashed line: constraints.

Discussion of Case Study 1

Based on the above case study, it is clear that the robust MPC outperforms the nominal for the metrics of SSE and constraint violation under nearly all conditions tested. Specifically, the robust MPC substantially reduces constraint violation compared to the nominal MPC, and also reduces the overshoot that both controllers display when the plant dead time or gain is larger than in the MPC model. The reduction in overshoot is particularly pronounced in the output constrained case,

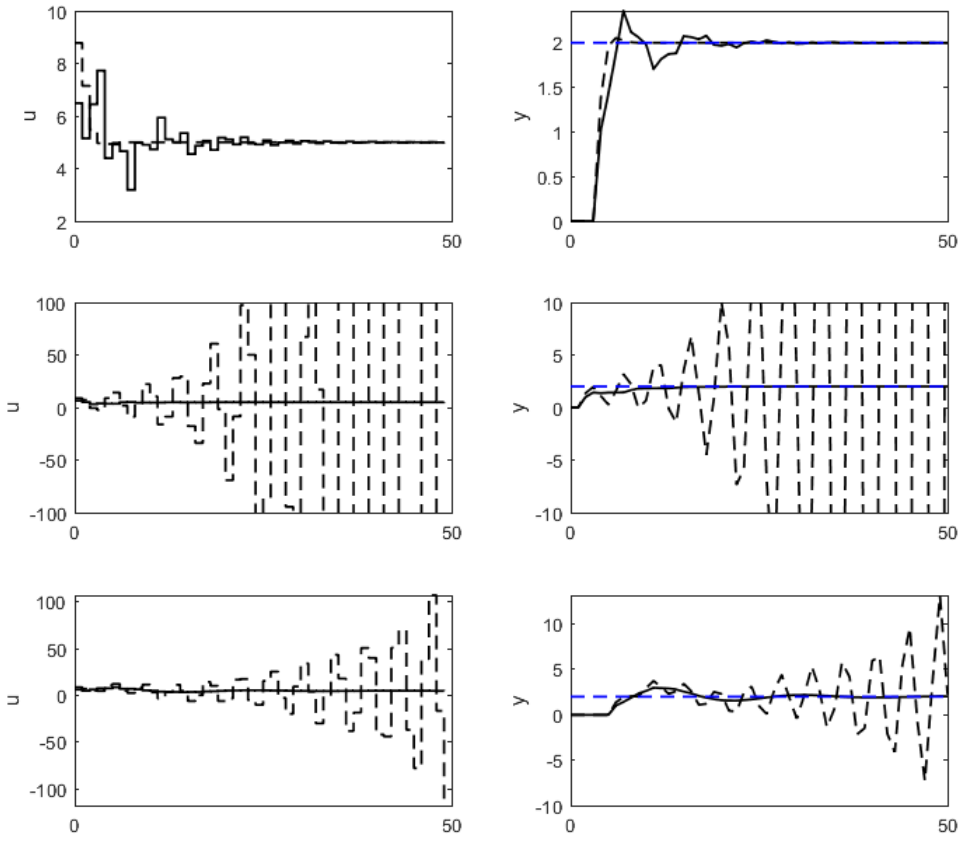

Figure 2.6: Input and output trajectories for the uncertain dead time and unconstrained SISO subcase with adjusted parameters for instability, for the nominal plant dead time (top), minimum plant dead time (middle), and maximum plant dead time (bottom). Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point.

where the nominal MPC continues to significantly overshoot the set-point, thereby also violating the upper bound constraint on the output. The robust MPC is able to further reduce the overshoot in order to nearly completely avoid constraint violation. It should be noted that increasing the soft constraint penalty term causes the constraint violation by the robust MPC to approach 0 but has no significant effect on the nominal MPC. This is because the nominal MPC does not predict the plant to go above the bound, while the robust MPC can account for the extreme

scenario and keeps the plant far from the constraint in order to avoid violation. The sum of squared set-point tracking errors provides a quantitative metric by which to compare the performance of the two controllers. The robust MPC achieves smaller SSEs, on average, by responding more quickly initially than the nominal MPC, but appropriately slowing when near the set-point. This reduces overshoot when the plant responds more quickly than predicted and reduces the time spent very far from the set-point. The nominal MPC, however, does require significantly less computation time.

2.3.2 Case Study 2: Nonlinear Polymerization Reaction with Linear MPC

The second case study investigated is a grade transition problem of a styrene polymerization reaction in a jacketed CSTR, studied by Maner et al. [47]. The plant model used is a nonlinear index-1 DAE system with six differential states and corresponding differential equations. The model equations are unchanged from the case study in Maner et al. [47] and the values of the model parameters are similarly unchanged, with the exception of the uncertain parameter, which is varied from the value specified in order to create the robust scenarios for the MPC, as described below.

The control problem has three inputs and three outputs; two outputs and one input are controlled. The chosen manipulated inputs for this problem are the initiator inlet flow rate, the coolant flow rate, and the monomer inlet flow rate. The monomer flow rate is the controlled input. The controlled outputs are the number average molecular weight (NAMW) and the temperature in the reactor. The concentration

of monomer in the reactor is a measured, uncontrolled output.

The setup for this MPC problem is to use a linearization of the nonlinear plant model as the MPC dynamic model. This maintains the MPC as a linear MPC. However, the MPC control actions will be implemented on the nonlinear plant model to simulate real plant behavior. This creates a structural plant-model mismatch inherent in running a linear MPC on a plant with nonlinear behavior. Additionally, the pre-exponential reaction rate constant for the polymerization reaction is chosen to have a $\pm 10\%$ uncertainty and additional linearizations of the system were created around the same steady state using the minimum and maximum values of this parameter. The result of this is to have three linear models of the system which will form the basis of the multi-scenario robust MPC prediction. A summary of the input and output variables is shown in Table 2.12.

The performance of the controller is evaluated on three potential nonlinear plant models, one with the nominal value of the uncertain parameter, one with the minimum, and one with the maximum. This determines whether the controller is effective at various plant models, simulating its effectiveness controlling a plant under a range of possible parameter realizations. Additionally, the linearization of the nonlinear model inherently creates plant model mismatch and so the robust MPC will be tested in its ability to control a plant under both parameter error and structural model error.

The set-points and constraints on the inputs and outputs are shown in Table 2.13. The output constraints are only applicable to the second subcase, where constraints are considered. The MPC parameters used for this case study are shown in Table 2.14.

Table 2.12: Summary of input and output variables used in Case Study 2.

Variable	**Description**	**Initial Value**	**Units**
Manipulated Inputs			
Q_i	Flow rate of initiator inlet stream	108	L/hr
Q_c	Flow rate of coolant inlet stream	471.6	L/hr
Q_m	Flow rate of monomer inlet stream	378	L/hr
Outputs			
$NAMW$	Number average molecular weight (controlled)	58.481	kg/mol
T	Temperature (controlled)	323.56	K
$[M]$	Concentration of monomer (measured)	3.3245	mol/L

Table 2.13: Input and output set-points and constraints for Case Study 2.

Variable	**set-point**	**Minimum**	**Maximum**	**Units**
Q_i	-	0	300	L/hr
Q_c	-	0	1000	L/hr
Q_m	400	0	600	L/hr
$NAMW$	68.9	58.381	70.9	kg/mol
T	325	323.56	325.5	K
$[M]$	-	3	4	mol/L

This case study involves an investigation into two specific subcases of the MPC problem, both involving a transition problem where the set-points on the controlled variables change at the outset of the simulation and the controller must direct the system to a new steady state. The first is with no output constraints or disturbance and the second is with both output constraints and a state disturbance. The MPC parameters, including weights and horizons, remain the same in both subcases. The only difference is the output constraint weight is set to 0 for the unconstrained subcase. Within each subcase, the robust MPC formulation is compared against a nominal MPC in three plant scenarios, each corresponding to the nonlinear plant used to

Table 2.14: MPC parameters for Case Study 2.

Parameter	**Description**	**Value**
p	Prediction horizon	5
m	Control horizon	2
Q	Output tracking weight	diag(0.1,5,0)
R	Input move suppression weight	diag(5e-4,1.5e-4,1e-5)
S	Input tracking weight	diag(0,0,5e-3)
N	Simulation horizon	50
ρ	Complementarity Penalty	1000
$\Omega_{\min}$,$\Omega_{\max}$	Output soft constraint penalty	diag(1,50,40)
T_s	Time Step	1 hr

generate the linear robust MPC scenarios. These scenarios represent the expected range of potential plant models.

Subcase 1: Unconstrained

The first subcase investigated is the unconstrained case. The sum of squared errors (SSE) for each controlled output variable of the nominal and robust MPCs across the different plant scenarios is shown in Table 2.15. The SSE is the primary metric used to determine the effectiveness of the MPC methods. Figure 2.7 shows the performance of the nominal and robust MPCs when implemented on the nominal plant (ie. the nonlinear plant with the nominal value of the uncertain parameter). Figures 2.8 and 2.9 show the performances on the minimum and maximum plants, respectively.

Table 2.15 clearly shows that the robust MPC significantly outperforms the nominal MPC across all plant scenarios. This can be seen by the decrease in SSE by approximately 20%, 29%, and 46% for the molecular weight, temperature, and monomer flow rate, respectively, from the nominal to the robust MPC. The

Table 2.15: sum of squared errors (SSE) of controlled variables for nominal and robust MPC methods with unconstrained outputs.

Plant Model		**Nominal MPC**	**Robust MPC**	**Percent Decrease**
Nominal	$NAMW$	235.57	185.71	21.17
	T	6.6255	4.6476	29.85
	Q_m	773.06	433.08	43.98
Minimum	$NAMW$	430.04	342.69	20.31
	T	11.078	8.2093	25.90
	Q_m	773.06	433.08	43.98
Maximum	$NAMW$	126.54	101.88	19.49
	T	4.3447	3.0137	30.64
	Q_m	435.75	230.69	47.06

improvement in SSE is also fairly consistent for each variable between scenarios, with the robust MPC producing a smaller error, and by a similar amount, for each variable in every scenario.

With a nominal plant model, it can be seen in Figure 2.7 that the overshoot of the set-point is smaller for the robust MPC for all three controlled variables. The overshoot beyond the steady state is also smaller for the other two inputs and slightly smaller for the uncontrolled output. The robust MPC also settles more quickly, as all input and output variables reach a new steady state with no oscillation prior to the nominal doing so.

The behavior of the controllers changes very little from the nominal to the minimum plant scenarios, as can be seen in Figure 2.8. The robust MPC again settles faster than the nominal MPC, with a smaller overshoot overshoot for the controlled variables.

For the maximum plant model, shown in Figure 2.9 the robust MPC responses settle faster, with a smaller overshoot than the nominal MPC for the controlled

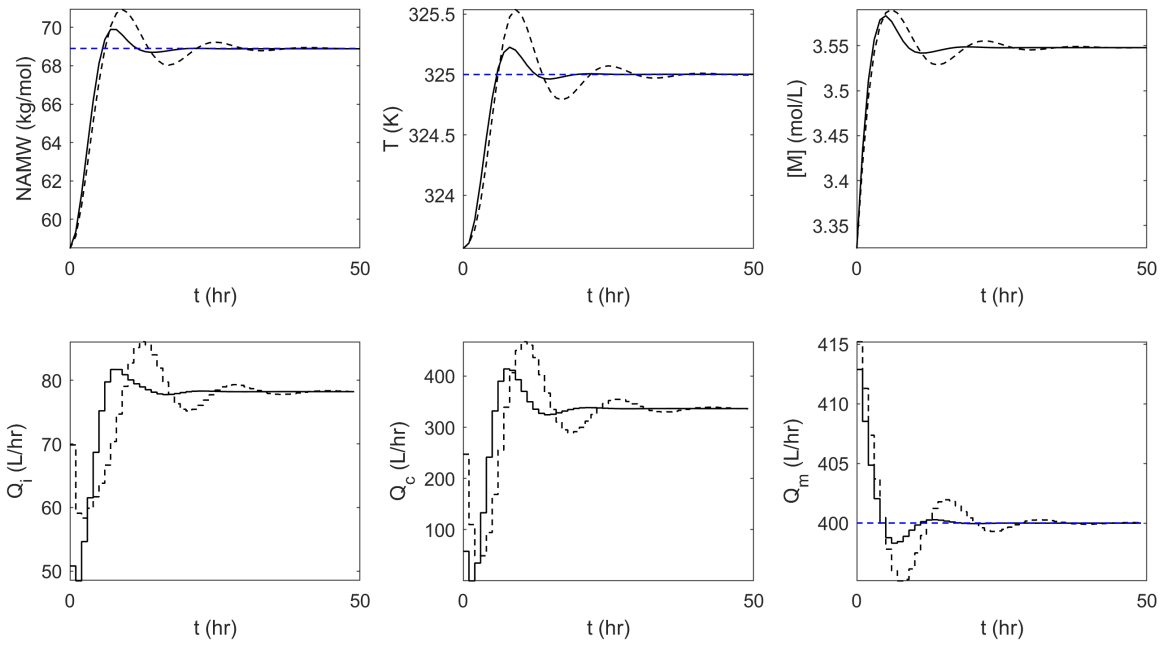

Figure 2.7: Input and Output trajectories for the unconstrained nonlinear MIMO subcase, nominal plant model. Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point.

outputs.

Subcase 2: Constrained

The second subcase investigated includes output constraints, as specified in Table 2.13, which adds an element which is likely to be present in a real plant. It would be reasonable to include temperature or composition constraints on a reactor to ensure that it is running efficiently, safely, and economically.

Additionally, it is desirable that these constraints be met even in the presence of plant model mismatch or disturbances. Therefore, both of these elements are tested here. Specifically, a state disturbance is included in this subcase where a fixed

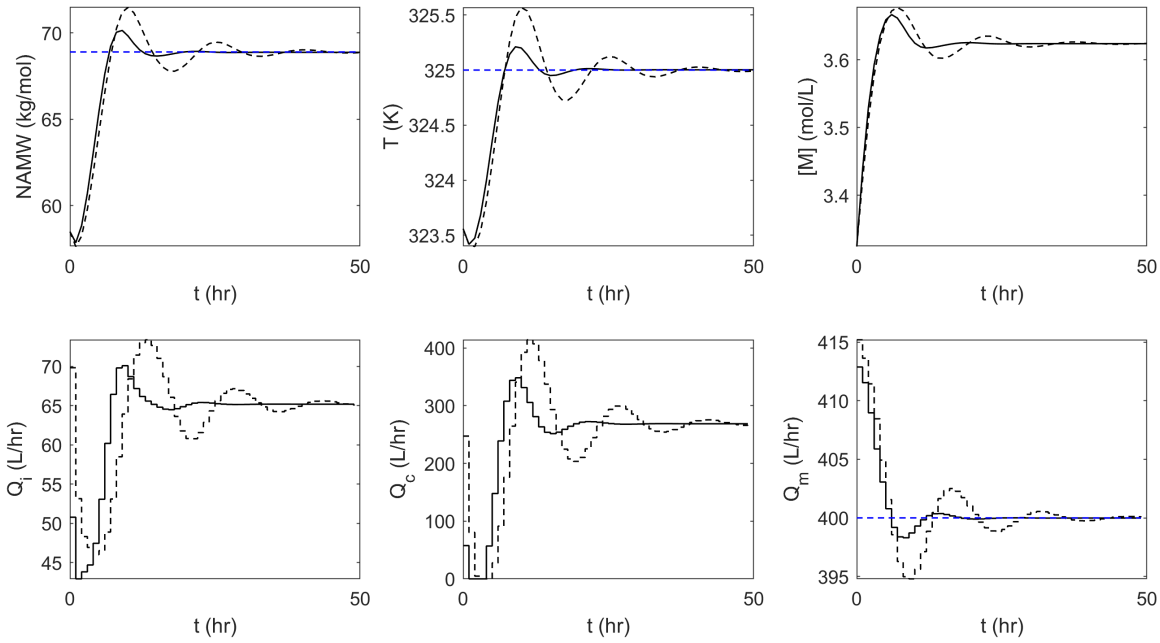

Figure 2.8: Input and Output trajectories for the unconstrained nonlinear MIMO subcase, minimum plant model. Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point.

value (+3%) is added to one of the simulation response states (the concentration of monomer) at the end of each sample time. This type of fixed disturbance could represent a number of physical disturbances present in the reactor, such as a systematic measurement error, an impurity in the monomer feed, or simply additional plant model mismatch. The purpose of adding the disturbance is to test the robust MPC in a scenario where the plant model (including disturbance) does not match exactly with one of the robust scenarios.

As in the previous subcase, the effectiveness of the robust MPC is determined by comparison with a nominal MPC in terms of SSE and qualitatively by examining the input and output trajectories. Additionally, the constraint violation produced by the

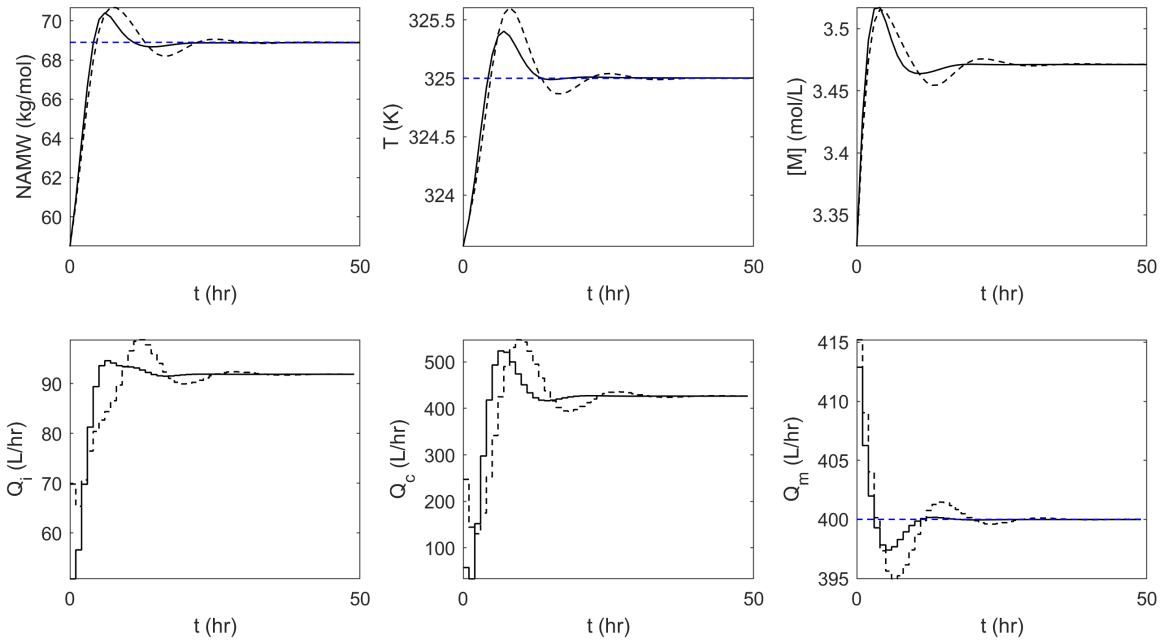

Figure 2.9: Input and Output trajectories for the unconstrained nonlinear MIMO subcase, maximum plant model. Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point.

control actions of the two controllers is also used as a metric. A summary of the SSE of the two controllers by variable across the three plant scenarios is shown in Table 2.16. A similar summary of the constraint violation is shown in Table 2.17. The input and output trajectories in the three plant scenarios is shown in Figures 2.10 through 2.12.

In the constrained subcase, the robust MPC continues to improve on the nominal MPC with respect to SSE in all scenarios. The robust MPC decreases the SSE by approximately 21%, 25%, and 62% for the molecular weight, temperature, and monomer flow rate, respectively. This improvement is similar to that of the unconstrained case, with more improvement in the molecular weight error, less for

Table 2.16: sum of squared errors (SSE) of controlled variables for nominal and robust MPC methods with constrained outputs.

Plant Model	Variable	Nominal MPC	Robust MPC	Percent Decrease
Nominal	$NAMW$	228.93	178.43	22.06
	T	7.0077	4.9394	29.51
	Q_m	793.68	289.66	63.50
Minimum	$NAMW$	388.54	306.68	21.07
	T	10.362	7.5551	27.09
	Q_m	945.92	412.81	56.36
Maximum	$NAMW$	150.68	119.99	20.37
	T	6.0489	4.8653	19.57
	Q_m	1043.5	344.36	67.00

the temperature, and substantially more for the monomer flow rate.

Table 2.17: Output Variable Constraint Violation for nominal and robust MPC methods with constrained outputs.

Plant Model	Output	Nominal MPC	Robust MPC	Percent Decrease
Nominal	$NAMW$	1.9583	0.058547	97.01
	T	0.014155	0	100
Minimum	$NAMW$	2.9848	0.39831	86.66
	T	0.11369	0.022674	80.06
Maximum	$NAMW$	4.0875	1.6176	60.43
	T	0.45473	0.14921	67.19

The robust MPC reduces the constraint violation of the controlled outputs substantially across all scenarios when compared to the nominal. The percent decrease from the nominal to the robust MPC is approximately 81% and 82% for the molecular weight and temperature, respectively.

The constrained trajectories with a nominal plant model, shown in Figure 2.10, retain many similarities to the respective scenario in the unconstrained case. The

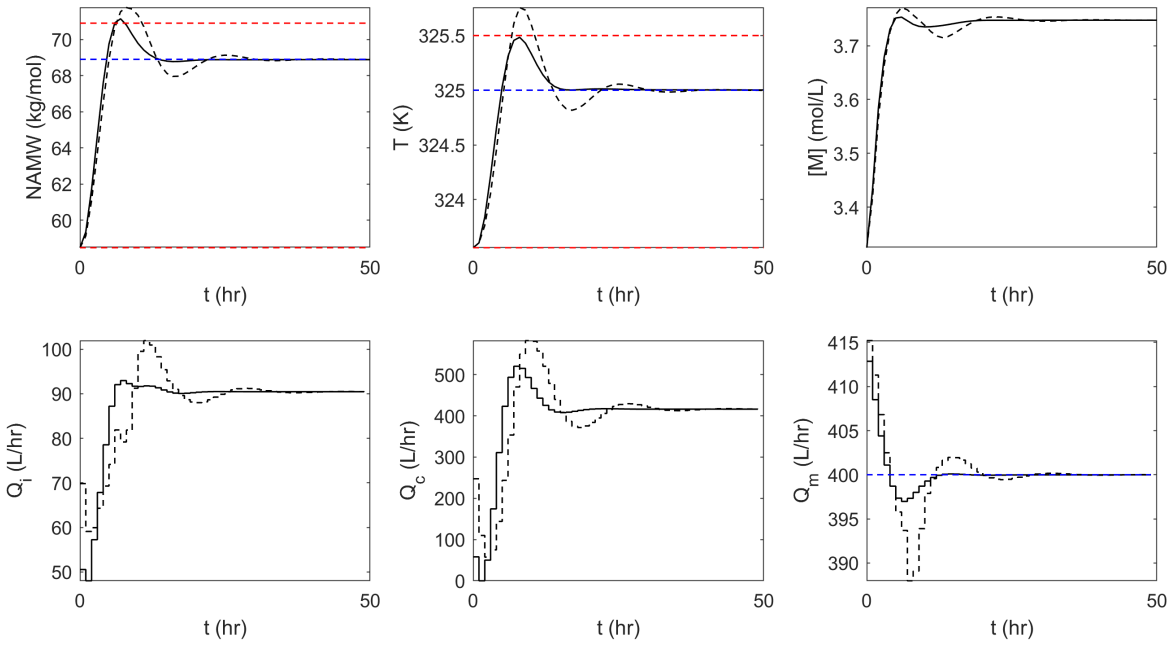

Figure 2.10: Input and Output trajectories for the constrained nonlinear MIMO subcase, nominal plant model. Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point; Red dashed lines: output constraints.

robust MPC has smaller overshoot in the controlled variables, and settles to steady state more quickly. Additionally, the robust MPC manages to nearly eliminate any constraint violation in this scenario, with only a small violation of the upper bound on the molecular weight and no violation for temperature. This is contrasted to the relatively substantial violation of the two upper bounds by the nominal MPC produced by the larger overshoot of that controller trajectory.

The minimum plant scenario in the constrained case, shown in Figure 2.11, is quite similar to the nominal scenario, just as in the unconstrained case. The robust MPC still has smaller overshoot than the nominal and settles faster. The nominal MPC again overshoots the set-points of the controlled outputs sufficiently to violate

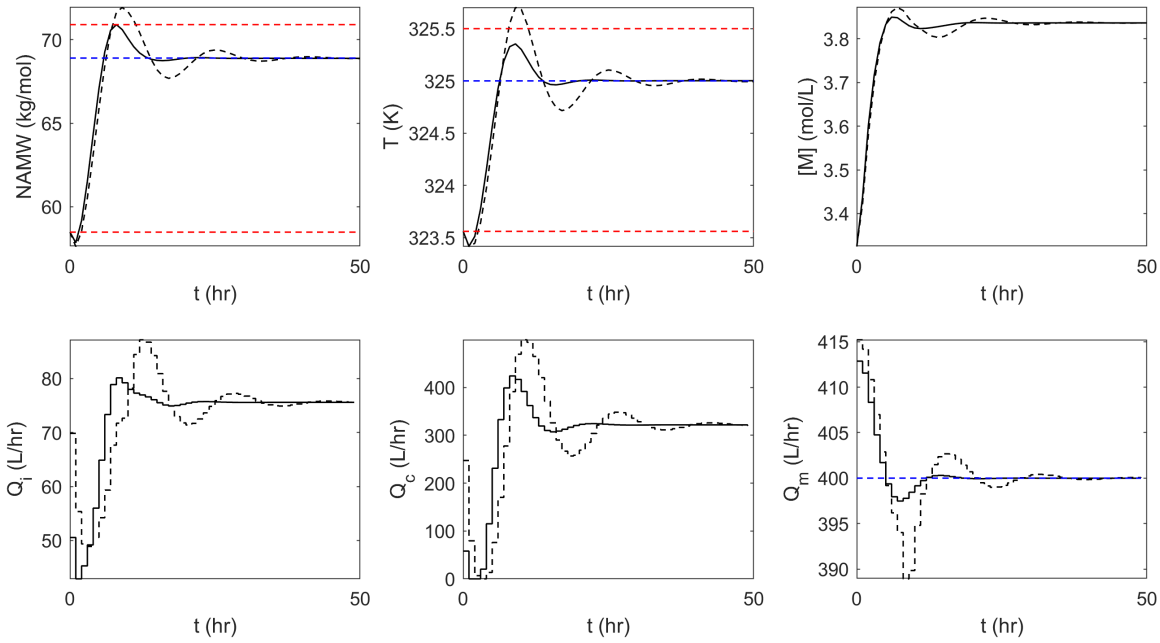

Figure 2.11: Input and Output trajectories for the constrained nonlinear MIMO subcase, minimum plant model. Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point; Red dashed lines: output constraints.

the upper bounds, and by a similar margin. The robust MPC, on the other hand, reduces the overshoot such that neither controlled output violates the upper bound. However, both controllers violate the lower bound for both variables in the initial time steps of the simulation. Nevertheless, the robust MPC still manages to substantially reduce the constraint violation in the two controlled outputs by reducing the overshoot so as not to exceed the upper bound.

The maximum plant scenario, seen in Figure 2.12, still retains some elements seen in the previous scenarios. The robust MPC still has smaller overshoot than the nominal and settles faster. However, the degree of these improvements, especially the overshoot reduction, is noticeably smaller in this scenario. This can be seen

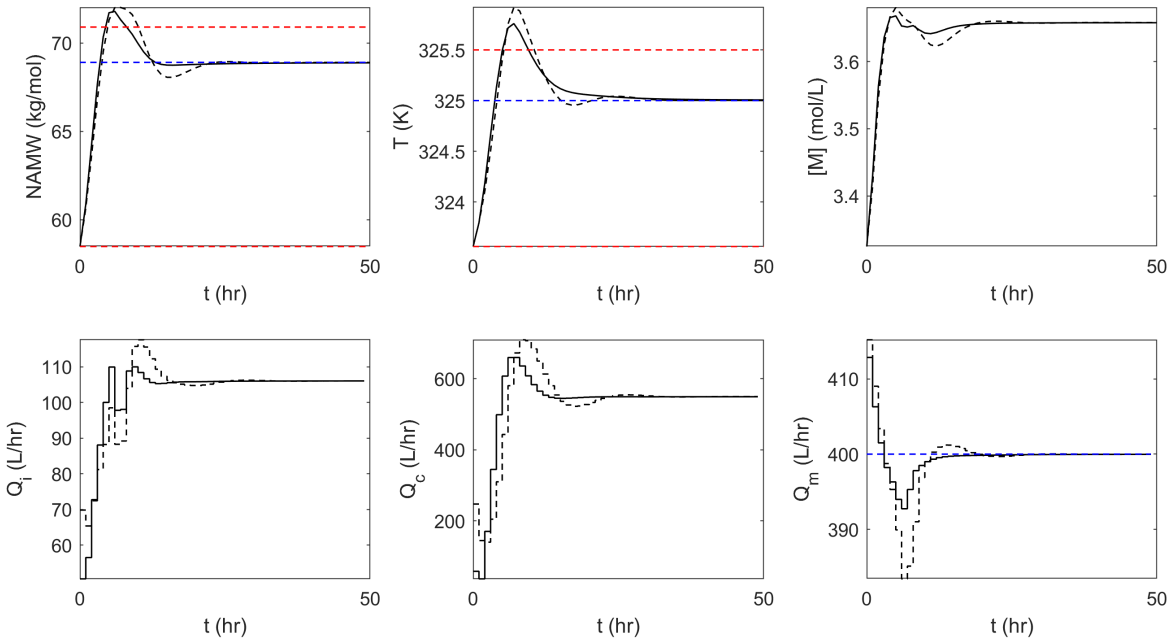

Figure 2.12: Input and Output trajectories for the constrained nonlinear MIMO subcase, maximum plant model. Solid line: Robust MPC; Black dashed line: Nominal MPC; Blue dashed line: set-point; Red dashed lines: output constraints.

most clearly in the two controlled outputs, where the overshoot produced by both controllers exceeds the upper bound. However, the robust MPC does manage to reduce the amount of violation by decreasing the overshoot and returning to below the upper bound more quickly than the nominal.

Discussion of Case Study 2

The second case study investigated here evaluates the performance of the robust MPC formulation on a more realistic system than the simple linear SISO system of the first case study. The disagreement between the linear MPC models and the nonlinear plant simulation models emulates the plant model mismatch likely to be present when an

MPC is applied to a real plant. Therefore, the effectiveness of the robust MPC in this case study should more closely reflect its ability to control real plants.

The robust MPC in this case study clearly outperforms the nominal MPC in both subcases, under all three plant scenarios, and in all measured metrics. The robust MPC output responses have smaller overshoot than that of the nominal MPC. The robust MPC settles and reaches the new steady state more rapidly; it reduces constraint violation; and it maintains a lower error in all controlled variables in all scenarios. Unlike in the first case study, this remains true even when the nominal MPC model uses the same parameter values as the plant model. This is likely because, even with identical parameters, the nominal MPC model and the plant simulation model are different, as one uses a linearized model and the other a nonlinear model. Therefore, including the uncertain parameter to produce linearizations, as in the robust MPC, improves performance even in nominal plant model scenarios.

Additionally, the robust MPC is able to better handle the state disturbance included in the second subcase, both in terms of reducing constraint violation and set-point error. In fact, the robust MPC manages to nearly eliminate upper bound violations in the first two plant scenarios, with only the maximum plant scenario resulting in substantial upper bound violation. This is likely because the maximum plant scenario with state disturbance represents a plant model which is functionally outside the robust scenarios included in the robust MPC. As such, it is unable to correctly predict the constraint violation and adjust accordingly to avoid it. Instead, it must react to the constraint violation after it has already occurred. Despite this, the robust MPC still outperforms the nominal in this scenario, by reacting faster to the violation and bringing the system to inside the constraints and then to steady

state more quickly. It is likely able to do this because its outermost scenario is still nearer to the true plant performance than the nominal MPC model is, thus giving it better predictive abilities even in this scenario.

2.4 Conclusions

In this chapter, a novel multi-scenario robust MPC method with embedded closed-loop prediction was proposed and tested against a standard MPC architecture. The closed-loop prediction allows the MPC to predict the future behavior of uncertain plant realizations while taking the effect of feedback action into account. By only branching at the first time step but continuing the embedded MPC problems along the prediction horizon, the future behavior of control action can be predicted throughout the prediction horizon without creating an exponentially increasing number of scenarios. The resulting multi-level optimization problem was reformulated as a single-level MPCC by expressing the embedded MPC subproblems as their associated first-order optimality conditions, which, since the inner MPC is a convex QP, are necessary and sufficient for optimality.

The robust MPC formulation presented here was tested in two main case studies against a standard QDMC formulation. The first case study involved a linear SISO transfer function model with either the gain or the dead time uncertain. For all subcases, the average SSE and constraint violation of the robust MPC across a range of uncertain parameter values was less than that produced by the standard MPC. The second case study involved a three input, three output polymerization reaction with a nonlinear plant model. In this case study, the robust MPC was shown to outperform the standard implementation in both a constrained and unconstrained

context regardless of the plant realization, even in the case where the standard MPC model used the same value of the uncertain parameter as the plant model. This was likely the result of the structural plant model mismatch produced by using a nonlinear plant model with a linear MPC model.

Chapter 3

Closed-Loop Stabilizing Dynamic Real-Time Optimization for Nonlinear Systems

The formulations and results in this chapter have been published and presented in:

[1] MacKinnon, L., Ramesh, P.S., Mhaskar, P., Swartz, C.L.E., 2022. Dynamic real-time optimization for nonlinear systems with Lyapunov stabilizing MPC. Journal of Process Control 114, 1-15.

[2] MacKinnon, L., Ramesh, P.S., Mhaskar, P., and Swartz, C. L. E., 2021. Closed-Loop Stabilizing Dynamic Real-Time Optimization for Nonlinear Systems. Presented at the 71st Canadian Chemical Engineering Conference (CCEC 2021), Montreal, QC, Canada (virtual).

[3] MacKinnon, L., Ramesh, P.S., Mhaskar, P., and Swartz, C. L. E., 2022. Dynamic Real-Time Optimization with Embedded Closed-Loop Lyapunov

Stabilizing MPC. Presented at the 2022 AIChE Annual Meeting (AIChE 2022), Phoenix, AZ, USA.

3.1 Introduction

Stability is an important aspect of both the economic and regulatory operation of chemical plants. This is because many chemical processes exhibit open-loop (OL) unstable characteristics where standard control and optimization techniques may be insufficient to bring the system to a desired steady state. These systems require additional control strategies to ensure they remain in stable operation.

Almost all existing closed-loop DRTO formulations, however, are designed for operation around open-loop stable points. In contrast, MPC has often been used for stability. For an introduction to many of the standard and well-established techniques for stabilizing MPC, Mayne et al. [52] provide an excellent review. Two key strategies discussed therein are endpoint constraint MPC and endpoint penalty MPC. Michalska and Mayne [56] propose a variation of the endpoint constraint method where the horizon of the MPC is variable. This allows the MPC to be feasible but computationally intensive when the system is far from the set-point and the required horizon is long, and to reduce the horizon and computational complexity when close to the set-point. Muske and Rawlings [58] discuss the technique of utilizing endpoint penalties and its stabilizing characteristics.

A third technique of importance to this work is that of Lyapunov based MPC. In such a design, the MPC contains constraints which utilize a Lyapunov function which must always be decreasing until it reaches a suitable region characterized by the system being in the vicinity of the desired set-point. This technique proposed

in Mhaskar et al. [54] (for input constraints) and Mhaskar et al. [55] (for state and input constraints) has the advantage that it continuously drives the system towards the set-point and therefore does not require a long horizon.

The use of a stabilizing MPC as part of a closed-loop DRTO strategy has potential to improve economic performance while handling possible instability. This technique has been partially explored by Ramesh et al. [67], who designed a CL-DRTO algorithm which directly models the behavior of an underlying endpoint penalty MPC. The DRTO MPC two-layer system was able to effectively stabilize an OL unstable process while improving on economic performance compared to a standalone stabilizing MPC. The formulation of Ramesh et al. [67], however, utilizes a linear model in the underlying MPC, limiting the performance of the DRTO for nonlinear systems.

This work aims to continue with this development by designing a CL-DRTO algorithm which functions with a Lyapunov-based stabilizing MPC, specifically able to handle process nonlinearity (from a stability stand-point), while still retaining the lower computational complexity of the formulation of Ramesh et al. [67]. To achieve this, the existing paradigm for Lyapunov MPC of nonlinear plant models both for prediction and Lyapunov constraints is adapted. This is to enable the calculations of the CL-DRTO, with its current implementation requiring the underlying MPC to be a convex QP. Thus, the present manuscript first presents a LMPC algorithm that allows the use of a linear plant prediction model and a nonlinear model in the Lyapunov constraints, allowing the overall optimization problem to be convex quadratic, and yet retains the stability property of the nonlinear LMPC. The resulting LMPC is tested in standalone operation, and the improved performance compared to a linear MPC

demonstrated. Following this, a CL-DRTO algorithm is presented which predicts the behavior of this convex LMPC; the CL-DRTO with LMPC is then tested for economic performance and stability characteristics, and significant improvements over the formulation in Ramesh et al. [67] demonstrated.

3.2 Preliminaries

3.2.1 System Description

We consider nonlinear plants affine in the control input described by state-space equations of the form:

$$\dot{x} = f(x) + g(x)u \tag{3.2.1}$$

where f and g are continuous nonlinear functions of the states, x, at time t and u is the control input constrained between $u_{\min}$ and $u_{\max}$.

3.2.2 CL-DRTO

This section briefly reviews the Closed-Loop DRTO (CL-DRTO) scheme of Jamaludin and Swartz [27] designed to ensure that the two layers of DRTO and MPC work together effectively. This DRTO computes reference trajectories, used as set-points for the MPC, which are economically optimal based on predicted plant and MPC behavior. As would be expected, ensuring that the predictions of the MPC are consistent with the actual MPC controlling the plant is paramount for the CL-DRTO to be effective.

The solution of the CL-DRTO involves using a primary economic optimization problem at the DRTO level which explicitly includes MPC subproblems for each of the MPC executions which take place along the length of the DRTO horizon. This creates a multi-level optimization problem. It should be noted that the time step of the DRTO and MPC are free to be different from one another, as well as the respective horizons. The CL-DRTO problem may be expressed as follows:

$$\min_{y^{Ref},u^{Ref}} \quad \phi_{econ}^{DRTO}(\hat{x}^{DRTO},\hat{u}^{DRTO},\hat{y}^{DRTO}) \tag{3.2.2a}$$

$$\text{s.t.} \quad \hat{x}_{j+1}^{DRTO} = f^{DRTO}(\hat{x}_j^{DRTO},\hat{u}_j^{DRTO}), \quad j=0,\ldots,N-1 \tag{3.2.2b}$$

$$\hat{y}_j^{DRTO} = h^{DRTO}(\hat{x}_j^{DRTO}), \quad j=1,\ldots,N \tag{3.2.2c}$$

$$0 \leq g^{DRTO}(\hat{x}_j^{DRTO},\hat{y}_j^{DRTO}), \quad j=1,\ldots,N \tag{3.2.2d}$$

$$0 = h^{Ref}(y^{Ref},u^{Ref},y^{SP},u^{SP}) \tag{3.2.2e}$$

$$0 \leq g^{Ref}(y_{j+1}^{Ref},u_j^{Ref}), \quad j=0,\ldots,N-1 \tag{3.2.2f}$$

$$\hat{u}_j^{DRTO} = \tilde{u}_{j,0}, \quad j=0,\ldots,N-1 \tag{3.2.2g}$$

$$\begin{aligned}\tilde{u}_{j,0} \in \arg \min_{\tilde{u}_{j,k}} \quad \phi_j = &\sum_{k=1}^{p}(\tilde{y}_{j,k}-y_{j,k}^{\text{sp}})^T Q(\tilde{y}_{j,k}-y_{j,k}^{\text{sp}}) + \sum_{k=0}^{m-1}(\Delta\tilde{u}_{j,k})^T R(\Delta\tilde{u}_{j,k}) \\ &+ \sum_{k=0}^{m-1}(\tilde{u}_{j,k}-u_{j,k}^{\text{sp}})^T S(\tilde{u}_{j,k}-u_{j,k}^{\text{sp}})\end{aligned} \tag{3.2.2h}$$

$$\text{s.t.} \quad \tilde{x}_{j,k+1} = A\tilde{x}_{j,k} + B\tilde{u}_{j,k}, \quad k=0,\ldots,m-1 \tag{3.2.2i}$$

$$\tilde{x}_{j,k+1} = A\tilde{x}_{j,k} + B\tilde{u}_{j,m-1}, \quad k=m,\ldots,p-1 \tag{3.2.2j}$$

$$\tilde{y}_{j,k} = C\tilde{x}_{j,k}, \quad k=1,\ldots,p \tag{3.2.2k}$$

$$\Delta\tilde{u}_{j,k} = \tilde{u}_{j,k} - \tilde{u}_{j,k-1}, \quad k=0,\ldots,m-1 \tag{3.2.2l}$$

$$u_{\min} \leq \tilde{u}_{j,k} \leq u_{\max}, \quad k=0,\ldots,m-1 \tag{3.2.2m}$$

where ϕ_{econ}^{DRTO} is the economic objective function of the primary DRTO; u,x,y are the system inputs, states, and outputs, respectively; the hat ($\hat{}$) and $DRTO$ superscript denote predicted values used at the primary DRTO optimization level; the Ref superscript denotes reference points; the SP superscript denotes set-point trajectories

for the MPC; f^{DRTO} is the set of dynamic equations for the plant model; h^{DRTO} is the algebraic equation system for the plant model; g^{DRTO} is the set of inequality process constraints; h^{Ref} is the functional relationship between the reference trajectory and the set-point trajectories for each MPC subproblem; g^{Ref} is the set of bounds on the reference points; the tilde ($\tilde{}$) denotes an MPC subproblem variable; ϕ_j is the MPC objective function corresponding to step j of the DRTO prediction horizon; m and p are the MPC control and prediction horizons, respectively; Q,R,S are the output deviation weighting matrix, input move penalty matrix, and input deviation weighting matrix, respectively; A,B,C are the linear state-space matrices for the linear plant system dynamic model; $u_{\min}$,$u_{\max}$ are the input bounds; and j,k are the time steps for the DRTO and MPC, respectively.

It is important to note that the input moves throughout the DRTO horizon are not decision variables at the DRTO level, but are rather computed by the MPC subproblems (they are the decision variables of the MPC subproblems). The DRTO degrees of freedom are the reference trajectories used to produce the set-points for the MPC. The set-point variables have two time dimensions, one for the time point for which the MPC is being executed and the other for the interior time point of the MPC. The reference trajectories, on the other hand, have a single time index that is associated with the DRTO horizon. The set-point variables are extracted subsets of the reference trajectory, such that the equivalent single time point in the reference trajectory is represented by the double time point of the set-point variable. For example, a specific instance of equation 3.2.2e may be expressed as $y^{SP}_{j,k} = y^{Ref}_{j+k}$. This communication is identical to that between the CL-DRTO and the actual MPC controlling the plant, with the DRTO providing set-points for the MPC and the MPC

providing the DRTO with implemented input moves.

The resulting multi-level optimization problem may be solved with many different strategies. Jamaludin and Swartz [27] chose to employ a simultaneous solution approach, and so this chapter will follow the same strategy. For the simultaneous method, as explored in Baker and Swartz [3], the MPC subproblems are each transformed into their respective first-order Karush-Kuhn-Tucker (KKT) optimality conditions. These equations are then included in the primary DRTO optimization problem as algebraic constraints. This creates a single-level mathematical program with complementarity constraints (MPCC). As mentioned previously, it is important for the CL-DRTO to accurately model the MPC behavior for effective optimization. MPC problems which are convex Quadratic Programs (QPs) can be exactly represented by their first-order KKT conditions. Therefore, it is desirable for the CL-DRTO to be utilized with a convex MPC, as this allows the CL-DRTO to exactly predict the MPC behavior via KKT constraints.

The KKT conditions of a general QP of the form,

$$\min_{z} \quad \frac{1}{2} z^T H z + g^T z \tag{3.2.3a}$$

$$s.t. \quad Az = b \tag{3.2.3b}$$

$$z \geq 0 \tag{3.2.3c}$$

can be written as:

$$Hz + g - A^T\lambda - \nu = 0 \tag{3.2.4a}$$

$$Az - b = 0 \tag{3.2.4b}$$

$$z_i\nu_i = 0, \quad \forall i \tag{3.2.4c}$$

$$(z, \nu) \geq 0 \tag{3.2.4d}$$

The complementarity constraints are handled with the use of an exact penalty approach [66], where a weighted penalty term of the product of the respective complementarity variables is included in the objective function of the primary DRTO problem.

3.2.3 CL-DRTO with Stabilizing MPC

The applicability of the above CL-DRTO strategy is predicated on the ability of the DRTO to accurately predict the behavior of the underlying MPC. For some applications, however, the MPC in use requires additional complications beyond what was considered by Jamaludin and Swartz [27]. One such requirement is the inclusion of stability constraints in the MPC for use with open-loop (OL) unstable processes. Therefore, Ramesh et al. [67] adjusted the CL-DRTO formulation for use with an endpoint penalty stabilizing MPC. The basic concept of the CL-DRTO remains the same as that for previous work, but with the addition of stabilizing considerations within the MPC subproblems of the CL-DRTO.

Specifically, the stabilizing MPC uses endpoint penalty terms in the objective function. These terms, therefore, also have to be included in the objective function of

the MPC subproblems of the CL-DRTO. The work also assumes a linear state-space plant model at the MPC level. The DRTO formulation itself remained unchanged from previous work. The MPC subproblems with endpoint penalty constraints may be expressed as follows:

$$\tilde{u}_{j,0} \in \arg\min_{\tilde{u}_{j,k}} \phi_j = \sum_{k=0}^{m-1} (\tilde{x}_{j,k})^T C^T Q C (\tilde{x}_{j,k}) + (\tilde{u}_{j,k})^T R (\tilde{u}_{j,k}) + \Delta\tilde{u}_{j,k}^T S \Delta\tilde{u}_{j,k} + (\tilde{x}_{j,m})^T \overline{Q} (\tilde{x}_{j,m}) + \Delta\tilde{u}_{j,m}^T S \Delta\tilde{u}_{j,m} \tag{3.2.5a}$$

$$\text{s.t. } \tilde{x}_{j,k+1} = A\tilde{x}_{j,k} + B\tilde{u}_{j,k} \ , \ k = 0, \ldots, m-1 \tag{3.2.5b}$$

$$\Delta\tilde{u}_{j,k} = \tilde{u}_{j,k} - \tilde{u}_{j,k-1} \ , \ k = 0, \ldots, m \tag{3.2.5c}$$

$$\tilde{u}_{\min} \leq \tilde{u}_{j,k} \leq \tilde{u}_{\max} \ , \ k = 0, \ldots, m-1 \tag{3.2.5d}$$

$$\tilde{u}_{j,k} = 0 \ , \ k \geq m \tag{3.2.5e}$$

$$\tilde{V}_u \left[A^{m-1}B, A^{m-2}B, \ldots, B\right] \tilde{u}_{j,m} = -\tilde{V}_u A^m (\tilde{x}_{j,0}), \tag{3.2.5f}$$

$$j = 0, \ldots, N-1 \tag{3.2.5g}$$

where $\overline{Q}$ is the endpoint penalty weighting matrix, $\tilde{V}_u$ is a matrix containing the unstable mode eigenvectors of the system.

The above formulation assumes the MPC specific variables are in deviation form relative to the set-points of the MPC. However, the DRTO does not assume this, and therefore equation 3.2.2g in the original CL-DRTO formulation must be altered to the following for consistency.

$$\hat{u}_j^{DRTO} = \tilde{u}_{j,0} + u^{SP}, \quad j = 0, \ldots, N-1 \tag{3.2.6}$$

For this MPC, to ensure stability of the underlying process, the objective function has additional terms for the endpoint penalty weighting. There is also a new constraint which is present for characterizing the future system conditions as unstable or stable. It should be noted that the matrices $\overline{Q}$ and $\tilde{V}_u$ can be determined offline Muske and Rawlings [59].

As in Jamaludin and Swartz [27] and Baker and Swartz [3], the multi-level optimization problem is solved using the simultaneous method where the MPC subproblems are transformed to their respective KKT conditions and included in the primary DRTO problem as constraints. These KKT conditions must simply be adjusted to reflect the addition of the endpoint penalty terms in the objective function and the new constraint. The resulting MPCC is again solved with the use of an exact penalty approach for the complementarity variables [66].

The objective function additions for this stabilizing MPC are themselves convex quadratic, the new constraint is linear, and the plant model of the MPC remains linear. Therefore, the stabilizing MPC used here remains a convex QP, and, as such, the KKT conditions are an exact representation of the MPC problem.

It should be noted that the stability properties of the resulting CL-DRTO problem are posited to be inherited from those of the underlying stabilizing MPC. This is because all feasible solutions of the CL-DRTO also represent feasible solutions to the MPC subproblems which are themselves exact representations of the actual MPC implementations.

3.2.4 Lyapunov-based Stabilizing MPC

Systems which can be represented by a model of the form of equation 3.2.1 can be effectively controlled using a stabilizing Lyapunov-based MPC (LMPC), as presented by Mhaskar et al. [55]. The MPC formulation, to compute the control action for a current state x_0, can be represented as follows

$$\min_{u} \quad \int_{t}^{t+T} x(s)^T Q x(s) + u(s)^T R u(s)\, ds \tag{3.2.7a}$$

$$\text{s.t.} \quad \dot{x} = f(x) + g(x)u \tag{3.2.7b}$$

$$\dot{V}(x(\tau)) \leq -\varepsilon \quad \forall \tau \in [t, t+\Delta) \quad \text{if} \quad V(x(t)) > \delta' \tag{3.2.7c}$$

$$V(x(\tau)) \leq \delta' \quad \forall \tau \in [t, t+\Delta) \quad \text{if} \quad V(x(t)) \leq \delta' \tag{3.2.7d}$$

where V is an appropriately chosen Lyapunov function, ε and δ' are positive values, Δ is the input hold time, and T is the horizon.

This formulation guarantees feasibility of the optimization problem and closed-loop stabilization from a region of initial conditions that can be determined a priori. Specifically, Mhaskar et al. [55] posit that any starting condition, x_0, of a system which can be represented by equation 3.2.1, within a well characterized region, will result in a feasible and stabilizing result from the LMPC formulation of equation 3.2.7. The rigorous proof and more detailed explanation thereof can be found in Mhaskar et al. [55].

It is important to note a few of the remarks concerning this proof of stability and feasibility for the LMPC. One is that this LMPC is also stabilizing for discrete implementations for sufficiently small values of the discrete time step. Secondly, the

Lyapunov constraints, equations 3.2.7c-3.2.7d, are the key to the stability properties of the LMPC. In other words, they alone drive the system to stability, with the objective function and other constraints only affecting the performance of the LMPC, not its stability. Finally, the Lyapunov constraints, in discrete implementation with sufficiently small time step, are only required to be active for the initial time step of the LMPC execution, rather than its entire horizon, for stability to be maintained. This is due to the receding horizon implementation practice, where each time step inevitably is used as the initial time step of subsequent MPC executions, thereby ensuring that the Lyapunov constraints are active for every implemented time step.

3.3 Convex Lyapunov-based Stabilizing MPC for Nonlinear Systems

The Lyapunov MPC formulation presented here is intended to maintain the key positive attributes of the LMPC algorithm developed previously while adjusting it such that it results in a convex quadratic program. The reason for this is that the CL-DRTO formulation previously developed is generally solved in such a way that it assumes the underlying MPC is convex quadratic. Incorporating a non-convex MPC problem into the existing structure would either require the CL-DRTO to only approximate the MPC, rather than predicting its exact behavior, or solving the multi-level optimization problem using an alternative method. In order to retain the advantages of the simultaneous solution approach, the LMPC was instead adapted to be convex such that it could be easily incorporated into the CL-DRTO method.

3.3.1 Proposed LMPC Formulation

The proposed LMPC formulation fundamentally relies on and adapts the Lyapunov constraints from Mhaskar et al. [55]. The proposed MPC formulation takes the form shown in equation 3.3.1:

$$\min_{\tilde{u}} \ \phi = \sum_{k=0}^{m-1} (\tilde{x}_{k+1})^T C^T Q C (\tilde{x}_{k+1}) + (\tilde{u}_k)^T R (\tilde{u}_k) + \Delta \tilde{u}_k^T S \Delta \tilde{u}_k \tag{3.3.1a}$$

$$\text{s.t.} \ \ \tilde{x}_{k+1} = A\tilde{x}_k + B\tilde{u}_k \ , \ k = 0, \ldots, m-1 \tag{3.3.1b}$$

$$\Delta \tilde{u}_k = \tilde{u}_k - \tilde{u}_{k-1} \ , \ k = 0, \ldots, m-1 \tag{3.3.1c}$$

$$\tilde{u}_{\min} \leq \tilde{u}_k \leq \tilde{u}_{\max} \ , \ k = 0, \ldots, m-1 \tag{3.3.1d}$$

$$\left.\frac{\partial V(x)}{\partial x}\right|_{\tilde{x}_k} \cdot g(\tilde{x}_k)\tilde{u}_k \leq - \left.\frac{\partial V(x)}{\partial x}\right|_{\tilde{x}_k} f(\tilde{x}_k) - \varepsilon V(\tilde{x}_k), \quad k = 0 \tag{3.3.1e}$$

The feasibility and stability properties are established next. To that end, we first define the following set:

Definition 1. For a given Δ and a choice of $V(\cdot)$, let Π be the connected set of states x for which there exists a $\tilde{u}_{\min} \leq \tilde{u}_k \leq \tilde{u}_{\max}$ such that $\left.\frac{\partial V(x)}{\partial x}\right|_{\tilde{x}_k} \cdot g(\tilde{x}_k)\tilde{u}_k + \left.\frac{\partial V(x)}{\partial x}\right|_{\tilde{x}_k} f(\tilde{x}_k) \leq -\varepsilon V(\tilde{x}_k)$ implies $\dot{V}(\tilde{x}(t)) < 0 \ \ \forall \ \ 0 \leq t \leq \Delta$, and let Ω be the largest level set of $V(x)$ within Π.

The above set is independent of the controller, and only depends on the system dynamics, choice of the control Lyapunov function, and the size of the region one ultimately wants the system to converge to. Thus, for a given set of these choices, the set Π captures the set of states for which, if the Lyapunov function decays, is made appropriately negative definite at the initial value of the state by the control

action; then for that fixed value of the control action over the hold time, the Lyapunov function value will continue to decrease for the rest of the hold time. Furthermore, picking Ω as the largest level set of $V(x)$ within Π renders Ω as the stability region for the system under an appropriate choice of the controller, shown next.

Theorem 1. *For any initial states, $x_0 \in \Omega$, the LMPC controller given by the optimization problem of Eqs. 3.3.1 is feasible for all times; the states remain bounded, $x(t) \in \Omega \quad \forall \quad t \geq 0$; and the states approach the set-point, $\lim_{t\to\infty} \|x(t)\| = 0$.*

Proof. The proof of the above theorem can be broken into the following points: the LMPC is always feasible for all initial states x_0 within Ω; the LMPC remains feasible for all subsequent time; and the Lyapunov function and therefore the states are decreasing over time and approach the set-point.

Recall that the constraint of Eq. 3.3.1e is chosen to be in line with the definition of Π, and since Ω is a subset of Π (specifically the largest level set of Π), any initial states within Ω result in an initially feasible LMPC. Furthermore, the definition of Π implies that the implementation of the control action results in the decay of the Lyapunov function over time, and since Ω is a level set entirely within Π, then any initial state $x_0 \in \Omega$ will produce a decreasing Lyapunov function and states which will produce subsequent states also within Ω. Therefore, for any starting position within Ω, the LMPC will be feasible for any subsequent time. Additionally, for all subsequent time, the feasibility of the LMPC and the definition of Π imply that the Lyapunov function and states will be monotonically decreasing, and therefore the states will be approaching the set-point for all time.

□

Remark 1. *The above requires the hold time, Δ, to be sufficiently small that the*

gradient of the Lyapunov function does not change sign during the hold time. This then ensures that the Lyapunov function is always decreasing. Note that this is not due to the control design but simply a result of the compute and hold nature of the control action, and as such the MPC using the nonlinear equations can also only guarantee practical stability.

Remark 2. *The decreasing Lyapunov function is necessary only for the initial time point as the receding horizon of the MPC causes the Lyapunov constraint to be active for all future time points as the LMPC executes for subsequent time points. In other words, all future time points of the system will eventually be the initial time point of an MPC execution. Therefore, the Lyapunov constraint will inevitably be in effect for every time point of the system. It is valid to use a horizon larger than 1 for the Lyapunov constraint, but doing so will cause the problem to be non-convex unless a linear model is used in the Lyapunov constraint. Instead, a nonlinear model is here used with a horizon of 1 thus keeping the optimization problem a convex optimization problem.*

Remark 3. *The model discrepancy between the Lyapunov constraint and the plant prediction implies that the two will predict different behaviors for the system. However, because the model prediction directly affects only the objective function, the use of the linear model only affects the performance of the LMPC, not its stability. The Lyapunov constraint determines the stability of the system, and therefore using the nonlinear model for this constraint implies that the system will be stable. This is an advantage over other stabilizing MPC systems which use only the linear model for both prediction and stability. This is because if the linear model becomes inaccurate, linear model based MPC will lose performance or may no longer be stabilizing, but*

the LMPC will remain stable and drive the system towards the set-point. Similarly, this LMPC is a convex problem, and is therefore much easier and faster to solve than a nonlinear problem. This is an advantage over MPC designs which use a nonlinear model for both stability and prediction.

Remark 4. *A consequence of the Lyapunov constraint being applied only for the initial time of the LMPC execution is that the stabilizing ability of the LMPC is independent of the horizon of the LMPC. This is because the horizon only affects the objective function and model prediction of the system, not the Lyapunov constraint. As the Lyapunov constraint is the only part of the LMPC necessary for stability, a change to the horizon of the LMPC does not affect the stability properties of the LMPC. A change in horizon, therefore, only affects the objective function performance of the LMPC, not its stability properties.*

The above Lyapunov constraint of equation 3.3.1e is adapted from the Lyapunov constraints from Mhaskar et al. [55] in two ways. First, to turn the constraint into a linear equation in the control action, the constraint is imposed only on the initial state. Thus, since the initial or current state of the plant is known (or estimated), the constraint is linear in the input for control affine systems described by Eq. 3.2.1.

The second adaptation from Mhaskar et al. [55] is as follows: In Mhaskar et al. [55], the constraint essentially required $\dot{V}$ to be less than $-\varepsilon$, with $\varepsilon > 0$ being a design parameter. Thus, as the Lyapunov function approaches 0, it may not be possible for the Lyapunov gradient to continue to be smaller than $-\varepsilon$. In the present formulation, an additional adjustment is made, specifically multiplying the right hand side by the value of the Lyapunov function such that the value the Lyapunov gradient must be below 0 as the Lyapunov function approaches 0, thus reducing the size of the

neighborhood around the origin to which the controller achieves practical stability.

The Lyapunov function for the case studies of this chapter was chosen to be of the following form.

$$V(x) = x^T P x \tag{3.3.2}$$

where P is an appropriately tuned positive-definite weighting matrix.

3.3.2 Case Study 1 - Standalone LMPC

The case study presented here to illustrate the effectiveness of the proposed linear Lyapunov MPC method is a two input, two output reaction jacketed CSTR model where the reaction is assumed to be second order and irreversible. The inputs for the control system are the inlet concentration of reactant and the net heat provided to the system by way of the heating/cooling jacket. The outputs of the system are the concentration of reactant in the CSTR and the temperature of the CSTR. The states and outputs are the same for this system. This case study was taken from Heidarinejad et al. [22] and is also used in Ramesh et al. [67]. The dynamic model used for this case study is given by Eqs. (3.3.3) and (3.3.4) below,

$$\frac{dC_A}{dt} = \frac{F}{V}(C_{A,0} - C_A) - k_0 e^{\frac{-E}{RT}} C_A^2 \tag{3.3.3}$$

$$\frac{dT}{dt} = \frac{F}{V}(T_0 - T) - \frac{\Delta H}{\rho C_p} k_0 e^{\frac{-E}{RT}} C_A^2 + \frac{Q}{\rho C_p V} \tag{3.3.4}$$

where C_A represents the concentration of reactant, T is the temperature in the reactor, $C_{A,0}$ is the inlet concentration of reactant, Q is the heat input to the system, F is the

Table 3.1: Model Parameters for MIMO Case Study

Parameter	**Description**	**Value**	**Units**
F	Inlet Flowrate	5	m^3/h
V	Volume of CSTR	1	m^3
E	Reaction Activation Energy	5e4	kJ/kmol
k_0	Reaction Rate Constant	8.46e6	h^{-1}
ΔH	Enthalpy of Reaction	-1.15e4	kJ/kmol
C_p	Heat Capacity of Reactor Fluid	0.231	kJ/kgK
ρ	Density of Reactor Fluid	1000	kg/m^3
R	Gas Constant	8.314	kJ/kmolK

inlet flowrate, V is the volume of the CSTR, k_0 is the pre-exponential rate constant of the reaction, E is the activation energy of the reaction, R is the ideal gas constant, T_0 is the inlet temperature, ΔH is the molar enthalpy of reaction, ρ is the density of the fluid in the reactor, and C_p is the heat capacity of the fluid. The model parameter values used are given in Table 3.1.

This system is used in three separate case studies where, in each, the proposed Lyapunov MPC is compared to an endpoint penalty MPC formulation. The first is a comparison of the MPC methods with no supervisory RTO or DRTO present and simply a static set-point provided to the MPC. This simulates a control hierarchy where the upper layer optimization method is a steady state optimizer and operates on much larger time scales than the MPC (such that the MPC will reach steady state before the RTO determines a new steady state operating point). The initial conditions of every simulation correspond to the unstable steady state at $u_{ss} = (4, 0), x_{ss} = (1.954, 401.9)$.

The first version of the case study investigated is one in which there is no supervisory optimization present, and demonstrates the superior control capability of

Table 3.2: MPC Parameters for MIMO Case Study

Parameter	**Description**	**Value**
m	Control Horizon	5
Q	Output Tracking Weight	diag(1,1)
R	Input Move Penalty Weight	diag(0,0)
S	Input Tracking Weight	diag(100,100)
T_S	Time Step	0.01 h
ΔT_{MPC}	MPC Execution Interval	0.01 h
$u_{\min}$	Input Constraints - Minimum	(0,-150)
$u_{\max}$	Input Constraints - Maximum	(10,150)
P	Lyapunov Function Weight	$\begin{bmatrix} 260 & 24.6 \\ 24.6 & 4.24 \end{bmatrix}$
$\overline{Q}$	Endpoint Penalty Weight	$\begin{bmatrix} 39700 & 797 \\ 797 & 16.0 \end{bmatrix}$
$\tilde{V}_u$	Unstable Mode Eigenvector	(61.4,2.23)

the proposed MPC compared to a standard linear MPC. In both MPC formulations, the MPC uses a linear state-space model linearized around the initial unstable steady state. The plant simulation uses the nonlinear model equations shown previously. For the proposed Lyapunov based linear MPC, the Lyapunov gradient function utilizes the nonlinear model equations, rather than the linearized equations, and evaluates the gradient only at the first time step of the MPC in accordance with the Lyapunov MPC formulation explained previously. For this case study, the model equations are linear in the inputs, allowing this technique to be used while the resulting MPC formulation is a convex quadratic problem. The MPC parameters used are the same for both MPC formulations and are shown in Table 3.2.

The Lyapunov function was found according to the previously explained formulation using the Ricatti equations to determine a suitable P matrix. The tolerance on the Lyapunov gradient, ε, was set to 3, as this was found to offer a

Variable	Endpoint Penalty MPC	Lyapunov MPC	Percent Decrease
x_1 (C_A)	0.579	0.107	81.5
x_2 (T)	179	69.3	61.3
u_1 $(C_{A,0})$	4.41	0.309	93.0
u_2 (Q)	1816	183	90.0

Table 3.3: Sum of Squared Error (SSE) by variable for standalone Lyapunov and endpoint penalty MPC in small transition subcase.

reasonable balance of aggressiveness and smoothness in the resulting MPC.

This case study involves two subcases. The difference between the two subcases is simply the set-point provided to the MPC. In the first subcase, the set-point is relatively close to the initial steady state, resulting in a relatively small transition time and distance, as well as good accuracy of the linearized model. The set-point for the second is further from the initial steady state, which increases the transition time, and decreases the accuracy of the linearized model, and illustrates the improved stabilizing region of the proposed MPC.

Subcase 1 - Small Transition

The linearized state space model is expected to be reasonably accurate in the relatively tight range of the first subcase and both MPC formulations are expected to perform well. The set-points for this subcase are $y_{SP} = (2.454, 406.9)$ and a corresponding steady state input set-point determined using the nonlinear model equations. The sum of squared errors (SSE) of the two approaches are summarized in Table 3.3 and the resulting plant trajectories are shown in Figure 3.1.

As can be seen in Figure 3.1, both MPC formulations perform well, reaching the desired set-point quickly and without substantial oscillations. It could be said that the Lyapunov MPC reaches the steady state somewhat faster than the endpoint penalty

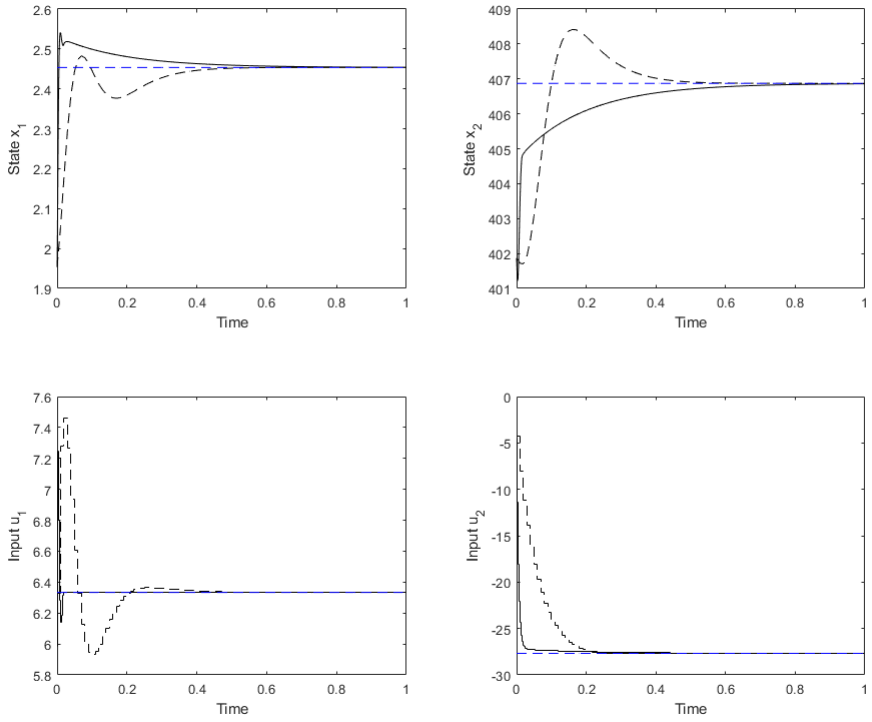

Figure 3.1: Input and Output Plots of the Lyapunov based MPC (solid black) and endpoint penalty MPC (dashed black) in standalone operation for small transition to set-point (blue dashed).

MPC, but it should be noted that this is dependent on the choice of the Lyapunov gradient tolerance, ε. Larger values of the tolerance will result in faster and less smooth performance. In Table 3.3, it can be seen that the SSE of the Lyapunov MPC is substantially smaller than that produced by the endpoint penalty MPC, suggesting that, in this case, the Lyapunov MPC is able to outperform the endpoint penalty MPC.

Variable	Endpoint Penalty MPC	Lyapunov MPC	Percent Decrease
x_1 (C_A)	64.1	1.75	97.3
x_2 (T)	50800	902	98.2
u_1 $(C_{A,0})$	9.71	0.0930	99.0
u_2 (Q)	17400	1560	91.1

Table 3.4: Sum of Squared Error (SSE) by variable for standalone Lyapunov and endpoint penalty MPC in large transition subcase.

Subcase 2 - Large Transition

The second subcase involves a somewhat larger transition with a set-point further from the steady state. Here, the linearized model is expected to be less accurate, resulting in poorer performance of the standard linear MPC. The proposed Lyapunov-based MPC uses the nonlinear equations in its Lyapunov gradient constraint, which is expected to improve the performance of the LMPC compared to the endpoint penalty MPC as the set-point moves away from the linearization point. The set-points for this subcase are $y_{SP} = (2.954, 411.9)$ and a corresponding steady state input set-point. The SSE results are summarized in Table 3.4 and the resulting plant trajectories shown in Figure 3.2.

As seen in Figure 3.2, the Lyapunov MPC is still able to reach the set-point provided and does so rapidly and smoothly. The time to reach steady state is larger than for the first subcase, but this is expected given the larger transition required. The endpoint penalty MPC, on the other hand, is unable to reach the desired set-point. Instead, it reaches a different steady state well away from the set-point. This is a result of the MPC only having the linearized state space model to make predictions, causing it to predict the system to move towards the set-point when in reality the input moves it is providing are driving the system to a different steady state. This

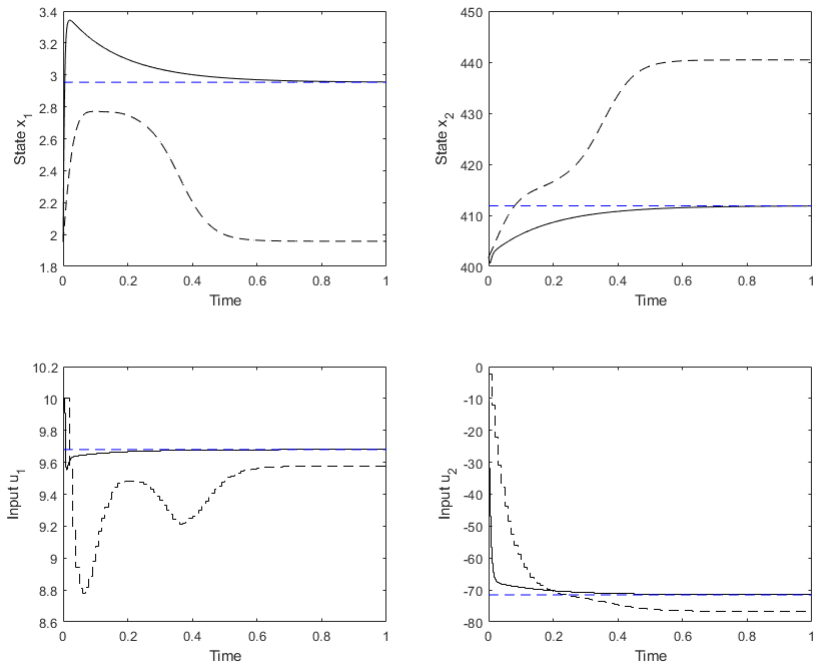

Figure 3.2: Input and Output Plots of the Lyapunov based MPC (solid black) and endpoint penalty MPC (dashed black) in standalone operation for large transition to set-point (blue dashed).

is a substantial drawback to stabilizing MPC methods (and to some extent, all MPC methods) which only use a linearized model. It is a drawback which is mitigated by the inclusion of the nonlinear Lyapunov constraints, while retaining the quadratic nature of the resultant MPC formulation.

Table 3.4 shows that the inability of the endpoint penalty MPC to reach the desired set-point results in the SSE of this MPC to be very high, and the corresponding improvement of the LMPC is therefore also very high.

The first case study shows that the presented LMPC formulation is an effective stabilizing MPC technique able to achieve significant improvements over a standard linear model based stabilizing MPC formulation. When the set-point is close to the

linearization point, both techniques reach the desired set-point quickly and without substantial oscillations. The LMPC is able to produce significantly lower error than the endpoint penalty MPC in this scenario, though this result is dependent on the values of the tuning parameters used. When the set-point is far from the linearization point, the LMPC greatly outperforms the endpoint penalty MPC, both in terms of error reduction and because the endpoint penalty MPC is unable to reach steady state at the desired set-point while the LMPC is able to do so. This is a significant advantage of the LMPC, as it suggests the LMPC has a larger stabilizable region than the endpoint penalty MPC due to its inclusion of the nonlinear plant model in its Lyapunov gradient constraint.

3.4 CL-DRTO with Lyapunov MPC

The CL-DRTO formulation follows the same structure as previous versions of the CL-DRTO algorithm with embedded prediction of an underlying MPC. The key distinction here is that the underlying MPC includes Lyapunov stability constraints. This is in contrast to previous work by Jamaludin and Swartz [27], which used an MPC with no stability constraints, and Ramesh et al. [67], which used a linear MPC with an endpoint penalty approach for stability. As mentioned previously, the underlying LMPC used in this chapter is constructed to be convex quadratic, which allows the embedded CL prediction of the underlying MPC to still be accomplished using a simultaneous approach by including the first order KKT optimality conditions of the MPC as algebraic constraints in the CL-DRTO problem, and enabling stability for the original nonlinear system (instead of for the linear approximation). The adjustment required is that the KKT conditions will change with the addition of the Lyapunov

stability constraints. The complementarity constraints can still be handled with an exact penalty approach.

3.4.1 Stabilizing CL-DRTO Formulation

The closed-loop DRTO approach presented here differs from previous work in its model of the underlying MPC. As the MPC now contains Lyapunov stability constraints, the embedded MPC subproblems must, therefore, also contain Lyapunov stability constraints. The overall DRTO with embedded MPC subproblems, including stability constraints, can be expressed in a general form as follows.

$$\underset{y^{Ref},u^{Ref}}{\text{maximize}} \quad \phi_{econ}^{DRTO}(\hat{x}^{DRTO},\hat{u}^{DRTO},\hat{y}^{DRTO}) \tag{3.4.1a}$$

$$\text{s.t.} \quad \hat{x}_{j+1}^{DRTO} = f^{DRTO}(\hat{x}_j^{DRTO},\hat{u}_j^{DRTO}) \quad , \quad j=0,\ldots,N-1 \tag{3.4.1b}$$

$$\hat{y}_j^{DRTO} = h^{DRTO}(\hat{x}_j^{DRTO}) \quad , \quad j=1,\ldots,N \tag{3.4.1c}$$

$$y_{\min} \leq \hat{y}_j^{DRTO} \leq y_{max} \tag{3.4.1d}$$

$$y_{\min}^{Ref} \leq y^{Ref} \leq y_{max}^{Ref} \tag{3.4.1e}$$

$$h^{Ref}(y^{Ref},u^{Ref},y^{SP},u^{SP}) = 0 \tag{3.4.1f}$$

$$\hat{u}_j^{DRTO} = \tilde{u}_{j,0} + u^{SP} \quad , \quad j=0,\ldots,N-1 \tag{3.4.1g}$$

$$\tilde{u}_{j,0} \in \underset{\tilde{u}_{j,k}}{\arg\min} \quad \Phi_j^{MPC} \quad , \quad j=0,\ldots,N-1 \tag{3.4.1h}$$

$$\text{s.t.} \quad \tilde{x}_{j,k+1} = A\tilde{x}_{j,k} + B\tilde{u}_{j,k} \quad , \quad k=0,\ldots,m-1 \tag{3.4.1i}$$

$$\Delta\tilde{u}_{j,k} = \tilde{u}_{j,k} - \tilde{u}_{j,k-1} \quad , \quad k=0,\ldots,m-1 \tag{3.4.1j}$$

$$\tilde{u}_{\min} \leq \tilde{u}_{j,k} \leq \tilde{u}_{\max} \quad , \quad k=0,\ldots,m-1 \tag{3.4.1k}$$

$$\tilde{u}_{j,k} = 0 \quad , \quad k \geq m \tag{3.4.1l}$$

$$\left.\frac{\mathrm{d}V(\tilde{x})}{\mathrm{d}\tilde{x}}\right|_{\tilde{x}_{j,k}} * [f(\tilde{x}_{j,k}) + g(\tilde{x}_{j,k})\tilde{u}_{j,k}] \leq -\varepsilon V(\tilde{x}_{j,k}), \quad k=0 \tag{3.4.1m}$$

The objective function used in the DRTO is intended to be an economic objective function, although a target tracking objective also works if the desired steady state of the system is already known. The plant model used in this chapter is a nonlinear model for improved performance and accuracy. The DRTO contains output constraints for the system, which may be hard or soft constraints, while the MPC (and MPC subproblems) contain the input constraints. Imposing

input constraints within the MPC optimization problem is consistent with typical MPC implementations; imposing these constraints as well at the DRTO primary optimization level (represented by Eqs. 3.4.1a to 3.4.1g) would be redundant. Hard output constraints within the MPC formulation are typically avoided in practice due to considerations of feasibility and stability, thus output constraints are applied at the DRTO primary optimization level. The degrees of freedom of the DRTO are the reference trajectories from which the MPC set-points are ultimately extracted; the reference trajectories are also constrained at the DRTO level. Finally, in accordance with previous CL-DRTO approaches, a set-point hold can also be applied such that the set-point provided by the DRTO to the MPC must be the same for a certain number of consecutive time points.

The communication between the DRTO and the MPC subproblems mirrors the communication between the DRTO and the actual plant MPC executions, and therefore the same information is exchanged in both cases. Specifically, the set-points used for the MPC are extracted from the reference trajectories which are decision variables of the DRTO, as expressed in equation 3.4.1. In the other direction, the inputs determined by the embedded MPC subproblems are used as the inputs at the DRTO primary at the appropriate time position. The states for the MPC subproblems are assumed to be fully observed, and therefore would update with plant measurements in a real MPC. However, as these MPC subproblems have no interaction with the real plant, the DRTO predictions of the states are used instead (these are different from the MPC predicted states as the MPC uses a linear prediction model and the DRTO a nonlinear model). The MPC also uses the last MPC execution determined initial input as the previous input of the plant (for

purposes of calculating the change in input).

3.4.2 Solution Approach

The resulting CL-DRTO is a multi-level optimization problem represented by the primary DRTO problem and the embedded MPC subproblems. For this chapter, this problem is solved using the simultaneous method used in Baker and Swartz [3], Jamaludin and Swartz [27], and Ramesh et al. [67]. Specifically, as alluded to previously, the MPC subproblems are reformulated to their associated Karush-Kuhn-Tucker (KKT) first order optimality conditions. This set of equations is then included in the DRTO as algebraic constraints, resulting in a single-level mathematical program with complementarity constraints (MPCC). The complementarity constraints are handled with an exact penalty approach, as presented in Ralph and Wright [66]. Due to the addition of the Lyapunov constraints, the KKT conditions for the embedded MPC subproblems are different here from those previously derived. The KKT conditions of this problem can be expressed as follows (using the specific choice of Lyapunov function from equation 3.3.2, rather than the general form, for simplicity).

The Lagrangian of the convex LMPC is given by

$$
\begin{aligned}
L_j = & \sum_{k=0}^{m-1} (\tilde{x}_{j,k+1})^T C^T Q C (\tilde{x}_{j,k+1}) + (\tilde{u}_{j,k})^T R (\tilde{u}_{j,k}) + (\Delta \tilde{u}_{j,k})^T S (\Delta \tilde{u}_{j,k}) \\
& - \sum_{k=0}^{m-1} (\lambda^1_{j,k+1})^T (\tilde{x}_{j,k+1} - A\tilde{x}_{j,k} - B\tilde{u}_{j,k}) - \sum_{k=0}^{m-1} (\lambda^2_{j,k})^T (\tilde{u}_{j,k} - \tilde{u}_{j,k-1} - \Delta \tilde{u}_{j,k}) \\
& - \sum_{k=0}^{m-1} (\eta^1_{j,k})^T (\tilde{u}_{j,k} - \tilde{u}_{\min}) - \sum_{k=0}^{m-1} (\eta^2_{j,k})^T (\tilde{u}_{\max} - \tilde{u}_{j,k}) \\
& + (\eta^3_{j,0})(\tilde{x}^T_{j,0} P g(\tilde{x}_{j,0}) \tilde{u}_0 + \tilde{x}^T_{j,0} P f(\tilde{x}_{j,0}) + \varepsilon V(\tilde{x}_{j,0}))
\end{aligned}
\tag{3.4.2}
$$

From this, the KKT optimality conditions are:

- Lagrange gradients, $\nabla L_j = 0$

with respect to $\tilde{x}_{j,k}$

$$2C^T Q C(\tilde{x}_{j,k}) + A^T \lambda^1_{j,k+1} - \lambda^1_{j,k} = 0, \quad k = 1, \ldots, m-1 \tag{3.4.3a}$$

$$2C^T Q C(\tilde{x}_{j,k}) - \lambda^1_{j,k} = 0, \quad k = m \tag{3.4.3b}$$

with respect to $\Delta \tilde{u}_{j,k}$

$$2S(\Delta \tilde{u}_{j,k}) + \lambda^2_{j,k} = 0, \quad k = 0, \ldots, m-1 \tag{3.4.3c}$$

with respect to $\tilde{u}_{j,k}$

$$2R(\tilde{u}_{j,k}) + B^T\lambda^1_{j,k+1} - \lambda^2_{j,k} + \lambda^2_{j,k+1} - \eta^1_{j,k} + \eta^2_{j,k} + (\eta^3_{j,k})(g(\tilde{x}_{j,k}))^T(P^T\tilde{x}_{j,k}) = 0, \quad k = 0 \tag{3.4.3d}$$

$$2R(\tilde{u}_{j,k}) + B^T\lambda^1_{j,k+1} - \lambda^2_{j,k} - \lambda^2_{j,k+1} - \eta^1_{j,k} + \eta^2_{j,k} = 0, \quad k = 1, \ldots, m-2 \tag{3.4.3e}$$

$$2R(\tilde{u}_{j,k}) + B^T\lambda^1_{j,k+1} - \lambda^2_{j,k} - \eta^1_{j,k} + \eta^2_{j,k} = 0, \quad k = m-1 \tag{3.4.3f}$$

- Primal feasibility of equality constraints, input inequality constraints transformed to equalities using slack variables, $\mu^i_{j,k}, \quad i = 1, 2, 3$

$$\tilde{u}_{j,k} - \tilde{u}_{\min} - \mu^1_{j,k} = 0, \quad k = 0, \ldots, m-1 \tag{3.4.4a}$$

$$\tilde{u}_{\max} - \tilde{u}_{j,k} - \mu^2_{j,k} = 0, \quad k = 0, \ldots, m-1 \tag{3.4.4b}$$

$$\tilde{x}^T_{j,0}Pg(\tilde{x}_{j,0})\tilde{u}_0 + \tilde{x}^T_{j,0}Pf(\tilde{x}_{j,0}) + \varepsilon V(\tilde{x}_{j,0}) + \mu^3_{j,0} = 0 \tag{3.4.4c}$$

$$\mu^1_{j,k} \quad , \quad \mu^2_{j,k} \quad , \quad \mu^3_{j,0} \geq 0, \quad k = 0, \ldots, m-1 \tag{3.4.4d}$$

It should be noted that, in addition to the above reformulation of the inequality constraints, the equality constraints of the MPC must also be included in the list of primal feasibility KKT constraints, but are left out here for brevity, as they are unchanged from their form in equations 3.4.1i-3.4.1j.

- Dual feasibility of inequality constraints

$$\eta^1_{j,k} \quad , \quad \eta^2_{j,k} \quad , \quad \eta^3_{j,0} \geq 0, \quad k = 0, \ldots, m-1 \tag{3.4.5}$$

- Complementarity constraints

$$\eta^i_{j,k}\mu^i_{j,k} = 0, \quad k = 0, \ldots, m-1, \quad i = 1, 2 \tag{3.4.6a}$$

$$\eta^3_{j,0}\mu^3_{j,0} = 0 \tag{3.4.6b}$$

The primary differences from previous CL-DRTO approaches are equations 3.4.3d and 3.4.4c which have been adjusted and added, respectively, to account for the Lyapunov constraints. The complementarity constraints are handled with an exact penalty approach by multiplying the corresponding primal and dual KKT variables together, summing the products, and adding the result to the DRTO objective function with an appropriate weighting term.

$$\phi' = \phi + \rho(\sum_{j=1}^{N}\sum_{k=0}^{m-1}(\eta^1_{j,k}\mu^1_{j,k} + \eta^2_{j,k}\mu^2_{j,k}) + \eta^3_{j,0}\mu^3_{j,0}) \tag{3.4.7}$$

where ϕ is the original DRTO objective function and ϕ' is the new objective function with the exact penalty weights for the complementarity variables. With an appropriately large weighting term, the complementarity constraints are satisfied for all time.

3.4.3 Overall System Stability

The above CL-DRTO retains the stability properties of the underlying Lyapunov MPC. In other words, any system and control implementation for which a standalone LMPC with constant set-point is itself stabilizing will also be stabilizable by the CL-DRTO with underlying LMPC system.

Theorem 2. *Given any initial state $x_0 \in \Omega$, the CL-DRTO optimization problem of Equation 3.4.1 is feasible for all time (in the absence of hard output constraints, Eq. 3.4.1d) , $x(t) \in \Omega \ \ \forall \ \ t \geq 0$.*

Proof. As posited previously, any initial states $x_0 \in \Omega$ will result in initial and subsequent feasibility, for all time, $t \geq 0$. This proposition was based on a static set-point of the LMPC at the origin, which causes the Lyapunov constraint to continuously drive the system towards the origin, away from the edges of the level set, Ω. Based on this, there exists a reference trajectory in the CL-DRTO which guarantees feasibility of the underlying LMPC subproblems, so long as the initial states are within Ω. In the absence of hard output constraints, this also implies that the overall CL-DRTO will be feasible. Assuming the reference trajectories are constrained to Ω, all subsequent states of the system will also be in Ω, guaranteeing feasibility of the CL-DRTO for all time, $t \geq 0$. □

Remark 5. *The existence of hard output constraints may cause the CL-DRTO to become infeasible as they may overly restrict the possible set-points provided to the LMPC and thereby the available input moves. However, if Ω is defined with respect to both the input constraints present in the LMPC and the output constraints in the DRTO (see Mhaskar et al. [55]), the resulting formulation will be feasible and stabilizing for all $x_0 \in \Omega$.*

Remark 6. *The stability of the LMPC being independent of its horizon is particularly useful as part of the CL-DRTO. This is because the horizon of the LMPC subproblems has a significant effect on the computation time of the CL-DRTO (because multiple LMPC subproblems must be computed). Therefore, a short horizon for the LMPC can be used without loss of stability of the overall CL-DRTO. Furthermore, the loss*

in performance in the LMPC as a result of the short horizon will be predicted by the DRTO and can be compensated for. Therefore, the computation time of this system can be reduced considerably by using a short LMPC horizon and allowing the CL-DRTO to mitigate much of the resulting performance loss while the overall system remains stabilizing.

Remark 7. *The increase in the stabilizing region of the LMPC relative to stabilizing MPC formulations which only use a linear model is of substantial benefit to the CL-DRTO. This is because the CL-DRTO is free to provide set-points to the LMPC within the stabilizing region with no risk of infeasibility (set-points outside* Ω *may or may not be feasible for the LMPC, but set-points within* Ω *are guaranteed to be feasible). Therefore, a larger region* Ω *wherein the CL-DRTO can provide set-points both increases the region to which the CL-DRTO can effectively optimize a transition; and allows for more freedom of set-points for the LMPC, which can improve the resulting behavior of the LMPC. This behavior is demonstrated clearly in the following Case Study 2, specifically Subcase 2, as well as in both subcases of Case Study 3.*

Remark 8. *In the event of the set-point trajectory provided to the LMPC being a static point, the resulting system will also be stabilizing. However, there is no guarantee that the set-point trajectory will be static over the DRTO horizon. This may be the result of the CL-DRTO attempting to improve the economic behavior of the system during a transition or because the optimal economic behavior of the system is not at steady state but rather oscillatory. In these cases, the system is guaranteed to stay within* Ω*.*

3.4.4 Case Study 2 - DRTO with target tracking objective

In this case study, the system being controlled is identical to that used previously for Case Study 1 where the convex LMPC was tested in standalone operation. Now, however, the MPC formulation is embedded in a two-layer control hierarchy with the CL-DRTO formulation at the upper level. This means that the MPC will be receiving set-point trajectories provided by the DRTO, rather than a single static set-point. This is meant to improve performance of the overall control system by allowing the DRTO to predict the behavior of the MPC and adjust its provided set-points accordingly.

For this scenario, the MPC formulations use the same parameter values as in the standalone MPC scenario, with one exception being the horizon of the LMPC going from 5 to 2. This will not affect the stability of the LMPC, as the stability properties are contained in the Lyapunov constraints and which are only applied for the initial time step, thereby being independent of the MPC horizon.

This was done for improved computation time at the DRTO level (which must simulate the MPC subproblems), and was found to have negligible effect on the overall controller performance, but substantially improve the computation time of the CL-DRTO. The horizon for the endpoint penalty MPC was kept the same as the stability properties of the endpoint penalty MPC are closely tied to the hoizon - a longer horizon gives the MPC more time to drive the system to its desired endpoint conditions. This stability horizon independence for the LMPC can pose an advantage over an endpoint penalty MPC by reducing its horizon and therefore computational complexity, which is especially important when embedded in a CL-DRTO, as the DRTO must model many instances of the MPC simultaneously.

Table 3.5: DRTO Parameters for MIMO Case Study

Parameter	**Description**	**Value**
N	Optimization Horizon	20
ΔT_{DRTO}	DRTO Time Step	0.05 h
$x_{\min}$	State Constraints - Minimum	(0.454,391.9)
$x_{\max}$	State Constraints - Maximum	(5.954,411.9)
$x_{SP,\min}$	State set-point Constraints - Minimum	(0.454,391.9)
$x_{SP,\max}$	State set-point Constraints - Maximum	(5.954,421.9)

The DRTO uses the nonlinear model equations in its prediction, allowing it to more accurately predict the behavior of the system. However, its MPC subproblems use the linearized equations, the same as for the MPC itself, in order to provide the best possible prediction of the MPC behavior. While the MPC uses only input bounds, the DRTO also includes output bounds to ensure the resultant behavior of the system stays within desired constraints. These constraints are formulated in the DRTO as soft constraints with a squared penalty term added to the objective function using a weighting parameter of $\omega = 100$. Additionally, the set-points provided to the MPC are themselves constrained. Finally, a set-point hold of 2 MPC periods is applied, which requires the set-points provided to stay the same for 2 MPC iterations. A full list of the DRTO parameters is given in Table 3.5.

For this scenario, the objective function used by the DRTO is a target tracking objective. This a version of the DRTO-MPC system, where the desired operating point is known a priori, corresponds to a reference governor control system in which the set-points may be varied to provide additional degrees of freedom to shape the response (Li and Swartz [40]; Garone et al. [19]). In the present context it may be viewed as a simplified case of the DRTO which allows for easier analysis of the comparative properties of the DRTO with the proposed Lyapunov MPC and the

DRTO with an endpoint penalty MPC. The objective function is given below in Eq. 3.4.8. Note that, in practice, the full objective function also includes the soft constraint penalty term and the complementarity constraint exact penalty term for the embedded MPC subproblems, but these are not shown here for compactness.

$$\min \quad \phi = \sum_{j=1}^{N} (y_j - y_{tar})^T (y_j - y_{tar}) \tag{3.4.8}$$

As in the previous scenario of the standalone MPC, this target tracking CL-DRTO scenario involves two subcases, one with a small transition and one with a larger transition. In fact, the targets provided in each of the two subcases are the same as the set-points used in the two subcases of the standalone MPC scenario. This is meant to show the improved performance of the CL-DRTO over the standalone MPC using both the LMPC and endpoint penalty MPC formulations.

Subcase 1 - Small Transition

The first subcase involves a small transition with a DRTO target of $y_{tar} = (2.454, 406.9)$. As both MPC formulations showed good performance in this subcase as standalone MPCs, it is expected that the CL-DRTO with MPC system also offers good performance with both MPC formulations. The SSE's of the two formulations are summarized in Table 3.6 and the plant trajectories shown in Figure 3.3.

As seen in Figure 3.3, the performance of the two formulations are here very similar. Both reach the set-point quickly, though with some oscillations before reaching steady state. The LMPC here is noticeably more aggressive than the endpoint penalty MPC. This does result in a substantial overshoot of the set-point, especially in the concentration of reactant (x_1). Additionally, both results here are

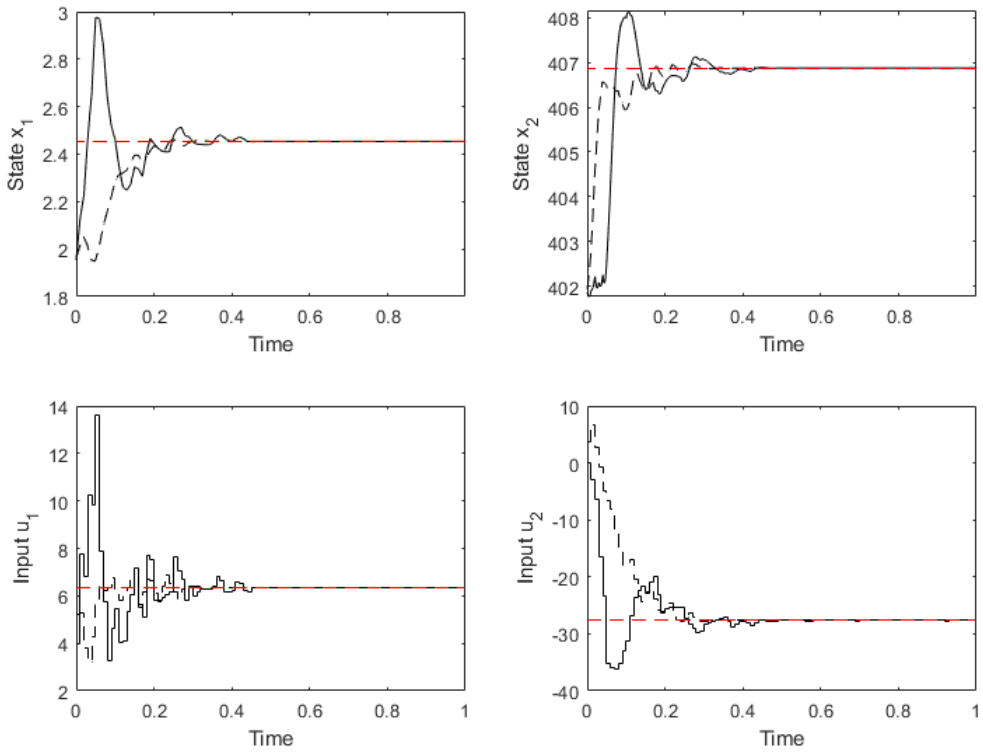

Figure 3.3: Input and Output Plots of the DRTO with Lyapunov based MPC (solid black) and DRTO with endpoint penalty MPC (dashed black) for small transition, target tracking objective (red dashed - target).

Table 3.6: Sum of Squared Error (SSE) by variable for DRTO with Lyapunov and endpoint penalty MPC in small transition subcase.

Variable	Endpoint Penalty MPC	Lyapunov MPC	Percent Decrease
x_1 (C_A)	1.75	1.20	31.3
x_2 (T)	39.0	141	-261
u_1 ($C_{A,0}$)	25.8	126	-387
u_2 (Q)	6070	2540	58.1

much less smooth than in the standalone MPC version, which is expected as the set-point provided to the MPC varies, resulting in noisy behavior. This, along with the overshoot seen in the LMPC version, can be largely mitigated by tuning the set-point hold parameter and the set-point variable bounds. Increasing the set-point hold will smooth out the trajectories of the controllers, approaching the standalone MPC behavior at a set-point hold corresponding to the DRTO horizon. Similarly, further constraining the set-points will decrease the overshoot by not allowing the set-point provided to the MPC to be too aggressive. It should be noted that the set-point trajectories determined by the DRTO are not included in the plots for this subcase for clarity.

Table 3.6 shows that the LMPC does not, overall, outperform the endpoint penalty MPC in terms of error reduction. Specifically, it reduces the error of the reactant concentration (x_1) and heat input (u_2) while increasing the error of the temperature (x_2) and the inlet reactant concentration (u_1). However, the overall SSE of both methods is relatively small compared to subsequent cases. Furthermore, the error of both methods can be reduced by appropriate tuning of the DRTO and MPC parameters, as both formulations are clearly capable of reaching steady state at the desired target value.

Subcase 2 - Large Transition

In the second subcase, as in the standalone MPC version, the transition is larger, causing the linearized MPC model to be less accurate. While the standalone endpoint penalty MPC was unable to reach the desired set-point, it is here expected that the CL-DRTO will be able to drive the MPC towards the desired set-point as it can

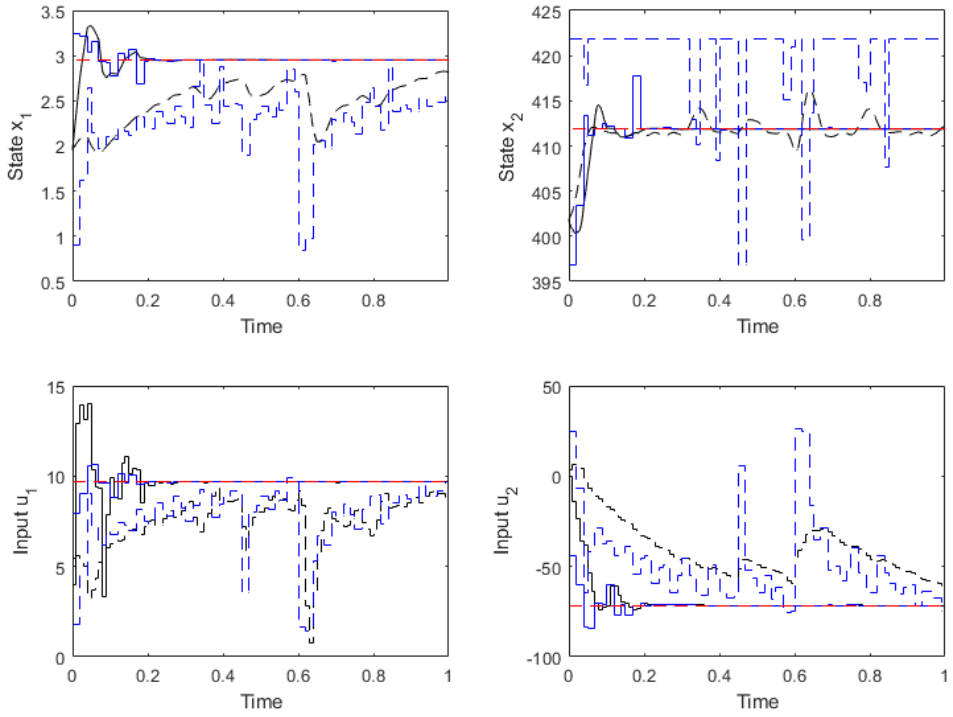

Figure 3.4: Input and Output Plots of the DRTO with Lyapunov based MPC (solid black - plant; solid blue - set-point) and DRTO with endpoint penalty MPC (dashed black - plant; dashed blue - set-point) for large transition, target tracking objective (red dashed - target).

predict the actual behavior of the plant (using the nonlinear model equations) and can provide set-points to the MPC which account for the MPC model error. The CL-DRTO with LMPC is expected to perform very similarly to the small transition subcase, as the LMPC had no difficulty reaching the set-point in standalone mode. The errors of the two methods are summarized in Table 3.7 and the resulting plant behavior shown in Figure 3.4.

As seen in Figure 3.4, the CL-DRTO with LMPC performs well, reaching the set-point and settling quickly with some overshoot. This is expected as the standalone

Table 3.7: Sum of Squared Error (SSE) by variable for DRTO with Lyapunov and endpoint penalty MPC in large transition subcase.

Variable	Endpoint Penalty MPC	Lyapunov MPC	Percent Decrease
x_1 (C_A)	27.6	1.40	94.9
x_2 (T)	325	551	-69.9
u_1 ($C_{A,0}$)	728	142	80.4
u_2 (Q)	125000	13800	88.9

LMPC did not have difficulty reaching this set-point. The CL-DRTO can be seen to noticeably increase the aggressiveness of the controller as compared to its standalone performance, as evidenced by its much shorter time to reach the set-point and its lower overall error. In contrast, the CL-DRTO with endpoint penalty MPC was able to approach the set-point in a reasonable time, though slower than with the LMPC, but was unable to control the system to steady state. Instead, the system continues to oscillate near the set-point. This is likely the result of the CL-DRTO being able to account for the MPC model error, but not fully stabilize the resulting MPC-plant system. This shows the limitations of the CL-DRTO method when the underlying MPC is unable to reach the desired set-point on its own. This attempt of the DRTO to force the endpoint penalty MPC to the desired target can be seen in the set-point trajectories provided to the MPC, particularly for the reactor temperature (x_2). The DRTO mostly provides set-points much higher than the actual target value in order to manipulate the MPC to move to the target, since providing the MPC with set-points equal to the target would result in the MPC reaching steady state away from the target, as evidenced by the standalone MPC performance.

Table 3.7 shows that the inability of the endpoint penalty MPC method to reach steady state at the desired target results in substantially higher SSE for three of the four controlled variables when compared to the LMPC method. The exception is

for the temperature (x_2), where the endpoint penalty MPC has lower error than the LMPC. This is because the endpoint penalty MPC approaches the target value more quickly initially (in the first few time stpes) than the LMPC does. As the majority of the SSE contributions come from the early time points when the variables are far from the target, this small increase in aggressiveness by the endpoint penalty MPC results in the overall SSE being lower than the LMPC, despite the former's inability to reach the desired target steady state.

Overall, when used in a two-layer control hierarchy with the upper level DRTO using a target tracking objective function, the DRTO with LMPC is able to outperform the endpoint penalty MPC when the target value is far from the linearization point. In this case, rather than the endpoint penalty MPC method not being able to reach the target, it instead reaches the region near the target but does not reach steady state, instead oscillating around the target.

3.4.5 Case Study 3 - DRTO with economic objective

In the final version of the case study, the target tracking objective is exchanged for a more realistic and commonly used economic objective function. Specifically, the rate of reaction is intended to be maximized for this scenario, as shown in Eq. (3.4.9).

$$\min \quad \phi = \sum_{j=1}^{N} -k_0 e^{\frac{-E}{R(y_{j,2})}} y_{j,1}^2 \tag{3.4.9}$$

The parameters used by the DRTO and MPC in this scenario are the same as for the target tracking DRTO scenario. The only changes are the variable and set-point

Table 3.8: Input and Output constraints for DRTO with economic objective function, subcase 1.

Variable	Minimum	Maximum
x_1 (C_A)	0.954	2.954
x_2 (T)	396.9	406.9
u_1 ($C_{A,0}$)	0	10
u_2 (Q)	-150	150

bounds, which will be mentioned as they are adjusted in each subcase.

The above objective function effectively causes the system to attempt to maximize both output variables. Therefore, the system will likely attempt to reach the upper bounds of the reactant concentration and temperature in order to maximize the objective function. In light of this, the output and input bounds were chosen separately for each subcase. Once again, soft constraints were used for the outputs at the DRTO level and the inputs were constrained at the MPC level (hard constraints).

Subcase 1 - Active State Constraints

For this subcase, the input and output bounds were chosen such that it is expected that the two output bounds are active while neither input bound is active. Additionally, the bounds are sufficiently close to the initial steady state that the linearized MPC model is expected to be able to stabilize at the new optimal steady state. Therefore, both DRTO formulations are expected to perform reasonably well. The actual economic objective values for both subcases are summarized in Table 3.9, and the corresponding state constraint violations in Table 3.10. The plant behavior for the first subcase is shown in Figure 3.5. The bounds for this subcase can be seen in Table 3.8.

The CL-DRTO with LMPC formulation performs well in this subcase, as can

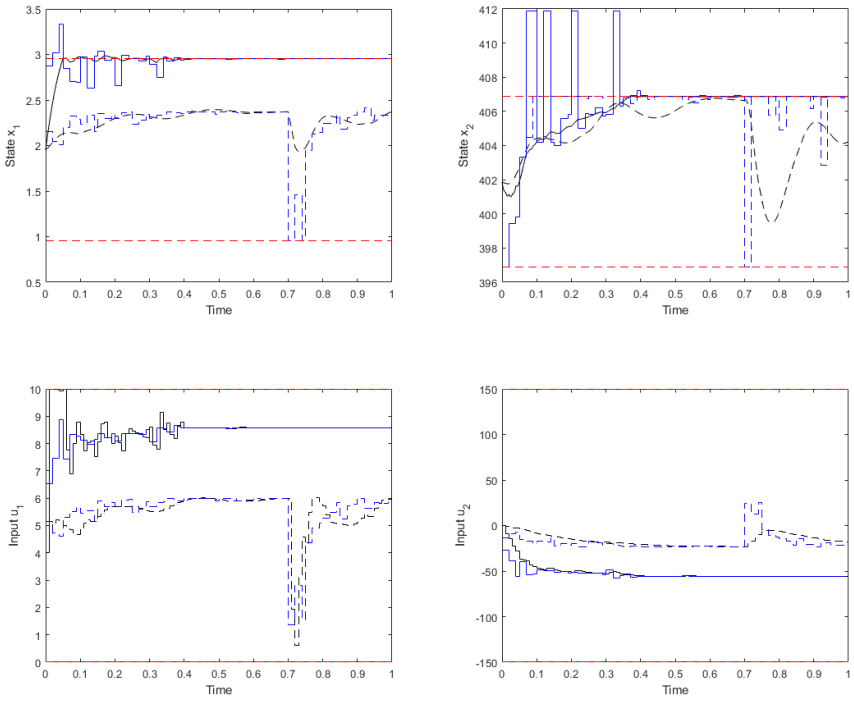

Figure 3.5: Input and Output Plots of the DRTO with Lyapunov based MPC (solid black - plant; solid blue - set-point) and DRTO with endpoint penalty MPC (dashed black - plant; dashed blue - set-point) with economic objective function (red dashed - bounds), subcase 1.

Table 3.9: Actual average economic objective value for DRTO with Lyapunov and endpoint penalty MPC with economic objective function.

Subcase	Endpoint Penalty MPC	Lyapunov MPC	Percent Increase
1	6.80	9.19	35.1
2	7.31	11.2	53.7

Table 3.10: Constraint Violation by variable for DRTO with Lyapunov and endpoint penalty MPC with economic objective function.

Subcase	Variable	Endpoint Penalty MPC	Lyapunov MPC
1	x_1 (C_A)	0	0.0258
	x_2 (T)	0	0.00401
2	x_1 (C_A)	0	0
	x_2 (T)	0	0.136

be seen in Figure 3.5, with similar performance to that seen in the first subcase of the target tracking objective scenario. It reaches the optimal steady state fairly quickly and with only small constraint violations. The overshoot is substantially reduced compared to the target tracking scenario because going above the steady state is heavily penalized in this scenario whereas it was not in the target tracking objective scenario. The CL-DRTO with endpoint penalty MPC performs much worse than the LMPC. It is unable to reach the optimal steady state of the system, instead approaching the optimal state for the temperature (x_2) but exhibiting large oscillations. For the reactant concentration (x_1), the system does not approach the optimal state but instead oscillates around a point well below the upper bound. The inputs, correspondingly, also do not approach the optimal steady state. This overall behavior results in substantially reduced economic performance (Table 3.9), though with improved constraint violation numbers (Table 3.10).

Subcase 2 - One State, One Input Active Constraints

In the second subcase, the input and output bounds are adjusted such that the optimal steady state is at the upper bounds of the first input and second output. The optimal steady state is also further from the linearization point in this subcase, which is expected to reduce the performance of the endpoint penalty MPC formulation. The

Table 3.11: Input and Output constraints for DRTO with economic objective function, subcase 2.

Variable	Minimum	Maximum
x_1 (C_A)	0.454	5.954
x_2 (T)	391.9	411.9
u_1 ($C_{A,0}$)	0	10
u_2 (Q)	-150	150

bounds for this are given in Table 3.11.

As seen in Figure 3.6, the DRTO with LMPC is able to substantially outperform the DRTO with endpoint penalty MPC by stabilizing at the optimal steady state while the endpoint penalty formulation does not. This performance is similar to that seen in subcase 1. The DRTO with Lyapunov MPC is able to quickly and with little oscillation reach the optimal steady with a small constraint violation (Table 3.10). The DRTO with endpoint penalty MPC does not drive the system to the optimal steady state defined by two upper bounds. Instead, its performance is again similar to subcase 1, though more pronounced. Only the second state approaches the optimal steady state, with all other variables staying far from their optimal conditions. This controller again shows significant oscillations rather than reaching a steady state. This behavior results in substantially improved economic performance by the DRTO with LMPC (Table 3.9), with over 50% improvement relative to the DRTO with endpoint penalty MPC.

Overall, the LMPC method substantially outperforms the endpoint penalty MPC in the cases investigated where the DRTO uses an economic objective function. This may be related to the optimal steady state being at active constraints, or may simply represent a limitation in stabilizing region for the endpoint penalty MPC. With only a slight improvement in constraint violation reduction, the endpoint penalty MPC

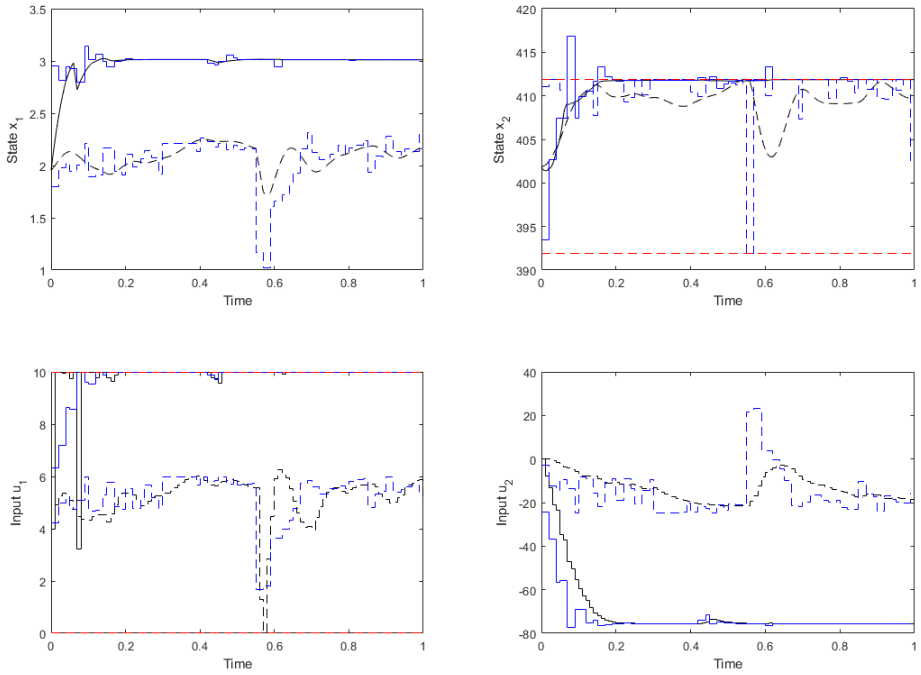

Figure 3.6: Input and Output Plots of the DRTO with Lyapunov based MPC (solid black - plant; solid blue - set-point) and DRTO with endpoint penalty MPC (dashed black - plant; dashed blue - set-point) with economic objective function (red dashed - bounds), subcase 2.

produces significantly lower economic returns than the LMPC both when the optimal steady state is close to the linearization point and when it is further away.

3.5 Conclusion

In this chapter, a CL-DRTO algorithm is presented which predicts the behavior of an underlying convex Lyapunov-based stabilizing MPC. The convex LMPC was adjusted from a nonlinear Lyapunov-based stabilizing MPC formulation to use a linear plant prediction model and to enforce the Lyapunov stability constraints at only the initial

time step of the MPC execution. This transformed the previously nonlinear LMPC into a convex QP, thereby allowing it to be directly embedded into the CL-DRTO algorithm as its first-order KKT conditions. The new LMPC was posited to retain all the stability properties of the nonlinear LMPC; it was also shown to be effective in controlling and stabilizing a nonlinear system over a wider range set-points than an endpoint penalty stabilizing MPC.

The resulting CL-DRTO algorithm was able to predict and account for the dynamic behavior of the plant system as well as the underlying LMPC and thereby determine economically optimal set-points to provide to the LMPC. This algorithm was posited to maintain the stability properties of the underlying LMPC, as any feasible solution to the CL-DRTO necessitated that the underlying LMPC also be feasible because the LMPC was directly embedded in the CL-DRTO. The CL-DRTO was then tested against a similar algorithm which predicted and used an underlying endpoint penalty MPC. The new algorithm with Lyapunov constraints was shown to outperform the prior endpoint penalty strategy in both target tracking and economic objective function cases. These results show that the new CL-DRTO with Lyapunov MPC algorithm is an effective economic optimization technique for OL unstable processes, and exhibits superior performance and a wider range of stable set-points than a similar strategy with an endpoint penalty MPC.

Chapter 4

Robust Closed-Loop Dynamic Real-Time Optimization

The formulations and results in this chapter have been published and presented in:

[1] MacKinnon, L. and Swartz, C.L.E., 2021. Robust Multi-Scenario Dynamic Real-Time Optimization with Embedded Closed-Loop Model Predictive Control. IFAC-PapersOnLine, 54(3), 481-486.

[2] MacKinnon, L. and Swartz, C.L.E., 2023. Robust Closed-Loop Dynamic Real-Time Optimization. Accepted (April 12, 2023) for Publication, Journal of Process Control, JPROCONT-D-22-00531R1.

[3] MacKinnon, L. and Swartz, C. L. E., 2020. Dynamic Real Time Optimization Under Uncertainty with Embedded CL Prediction. Presented at the 2020 Virtual AIChE Annual Meeting (AIChE 2020).

[4] MacKinnon, L. and Swartz, C. L. E., 2022. A Multi-Scenario Stochastic

Framework for Dynamic Real-Time Optimization under Uncertainty with Embedded Closed-Loop MPC. Presented at the 2022 AIChE Annual Meeting (AIChE 2022), Phoenix, AZ, USA.

4.1 Introduction

The handling of uncertainty has been widely incorporated into regulatory control in the form of robust MPC; it is, however, less common for it to be handled at the economic optimization level. Conceptually, the inclusion of uncertainty handling at the RTO level is just as viable as doing so at the control level. The consequence of not handling, or improperly handling, the uncertainty is potential loss of profitability. Therefore, inclusion of uncertainty handling characteristics in an RTO or DRTO framework has the potential to improve the economic performance of a system where the plant behavior is uncertain.

Traditionally, in the presence of uncertainty, steady state RTO algorithm are executed with two key steps - data reconciliation and estimation of model parameters (often performed simultaneously), followed by optimization to determine a new operating point [48, 12]. Strategies to explicitly account for uncertainty include calculation of a back-off amount from active inequality constraints into the feasible region based on disturbances and/or measurement noise [60, 42, 73], and casting the RTO problem within a stochastic optimization framework [82]. The applications considered in these studies include mineral flotation circuits [60], a fluid catalytic cracking unit [73], and gasoline blending [72, 82].

In contrast to steady state RTO, explicit consideration of uncertainty in DRTO has received relatively little attention. Würth et al. [77] use neighboring-extremal

updates of parametric sensitivities to avoid solving the rigorous optimization problem online. This method was found to be fast for small perturbations; systems that exhibit strong nonlinearities or larger perturbations require additional iterations, thereby increasing the solution time of the optimization problem. The method was applied to a simulated semi-batch reactor. Krishnamoorthy et al. [33] utilize a scenario tree-based optimization strategy in dynamic real-time optimization applied to a gas lifted well network under uncertainty. Perfect low level controllers were assumed. Müller et al. [57] apply a chance-constrained DRTO method to a constructed hydroformylation mini-plant. A dynamic model was formulated, comprising 23 process units, 12 components, and 25 streams. Three uncertain parameters were identified.

This chapter extends the MPC-aware CL-DRTO paradigm of Jamaludin and Swartz [27] to directly account for uncertainty within the DRTO optimization formulation. A scenario-based stochastic optimization formulation is used, in which the nominal plant MPC is applied to a number of discrete uncertain plant realizations, and the expected value of the objective function optimized. An input-clipping strategy is used to approximate the predicted closed-loop response under constrained MPC in order to reduce computation time, and we also explore the impact of the number of scenarios on the performance of the algorithm.

The remainder of the chapter is organized as follows. The mathematical formulation of the robust DRTO algorithm is presented in Section 4.2, where the primary economic optimization problem is laid out, followed by the setup of the embedded MPC subproblems. The solution strategy for this multi-level optimization problem is then explained, followed by a description of its modification by inclusion

of the input clipping approximation method. Section 4.3 presents two case studies, where the performance of the robust CL-DRTO scheme is evaluated against that of a nominal, single-scenario CL-DRTO. The first is a simple SISO system, and the second is a larger multi-input-multi-output (MIMO) case study. Additionally, the second case study includes an investigation into the effectiveness of different numbers of scenarios, and how the robust CL-DRTO performs with multiple sources of uncertainty.

4.2 Formulation

The objective of the robust DRTO formulation is to extend the CL-DRTO strategy previously developed in Jamaludin and Swartz [27] to effectively handle plant uncertainty at the DRTO level. This is accomplished by modeling multiple possible uncertain scenarios within the CL-DRTO algorithm. The plant prediction model differs from one scenario to the next through different uncertain parameter realizations within the model, while the MPC subproblems within each scenario utilize the same nominal plant model. This is because the actual MPC controlling the plant does not change based on the observed behavior of the plant; this phenomenon is replicated in the design of the DRTO scheme. An illustration of the multi-scenario robust CL-DRTO architecture is presented in Fig. 4.1.

This work considers nonlinear plants controlled by linear MPC, a common practice in industrial MPC applications. The DRTO uses a discretized nonlinear plant model, while the MPC subproblems utilize a linear plant model. The advantage of using two different model types is that the DRTO can maintain greater accuracy, while the MPC subproblems can remain relatively simple and maintain consistency with

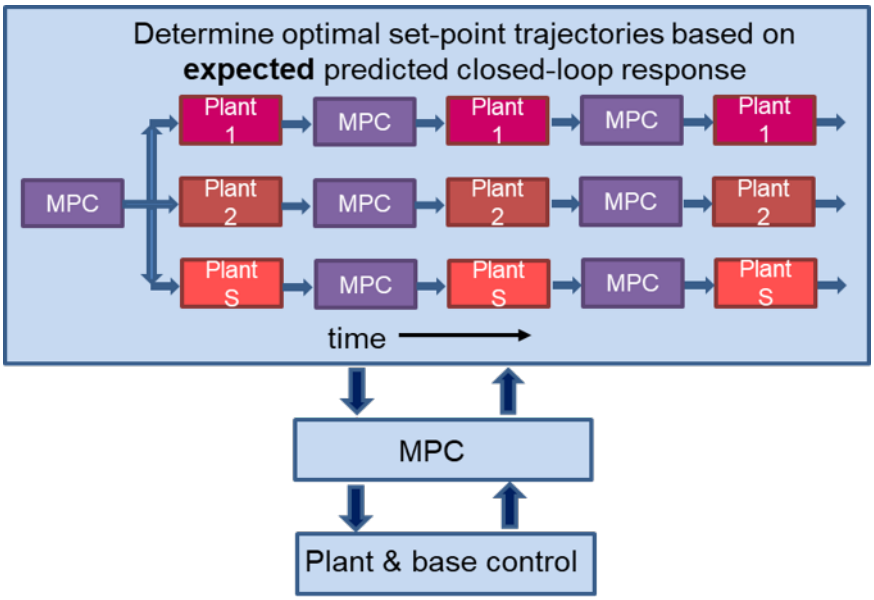

Figure 4.1: A visualization of the robust DRTO architecture.

the plant MPC. However, if the plant MPC uses a nonlinear model, it is likely to be beneficial to also use a nonlinear plant model for the MPC subproblems within the DRTO. This is a topic for future work.

The degrees of freedom in the robust DRTO optimization formulation are the set-point trajectories utilized by the MPC. These trajectories are common to all scenarios, and are determined such that the expected performance over all plant scenario closed-loop responses is optimized across the DRTO prediction horizon. In the DRTO formulation that follows, the primary optimization and MPC subproblems are presented in separate subsections for clarity; however, it is important to recognize that they constitute a single multilevel optimization problem whose components are solved simultaneously.

4.2.1 Primary DRTO problem

The primary robust CL-DRTO formulation proposed here includes an economic objective function, a nonlinear plant prediction model, and embedded prediction of the underlying MPC. The plant prediction model changes for each scenario, here expressed as a changing parameter (or parameters) within the model. The scenario tree branches only once, after the initial time point, with each scenario thereafter following a single path corresponding its predicted closed-loop response. The DRTO objective function can be specified as desired, including using a set-point target tracking function rather than an explicit economic objective, but is here expressed as the expected value of the economic performance over the uncertainty scenarios.

The DRTO problem also includes process constraints relevant to the plant being modeled. These may include safety constraints or known relations between variables which must remain true for physical realizability. In terms of variable bounds, the set-point trajectories here have hard constraints so that the solution to the optimization problem does not become unbounded and the set-points do not move to a point which is not realistic to implement on the real plant. The predicted output variables may be bounded as necessary at the DRTO level, although it should be noted that including these as hard constraints has the potential to lead to infeasibility of the overall optimization problem, thus in this work the output bounds are included as soft constraints. Input constraints are applied in the MPC subproblems, and are therefore not necessary in the DRTO primary optimization problem. Unless otherwise specified, all equations and variables indexed by i apply to scenarios, $i = 1, \ldots, N_s$. The primary DRTO problem is expressed as follows:

$$\min_{y^{Ref},u^{Ref}} \quad \phi_{econ} = \sum_i \rho^i \phi^i_{econ}(x^i, u^i, y^i) \tag{4.2.1a}$$

$$\text{s.t.} \quad x^i_{j+1} = \quad f^i(x^i_j, u^i_j), \quad j = 0, \ldots, N-1 \tag{4.2.1b}$$

$$y^i_j = \quad h^i(x^i_j), \quad j = 1, \ldots, N \tag{4.2.1c}$$

$$0 \leq \quad g(x^i_j, y^i_j), \quad j = 1, \ldots, N \tag{4.2.1d}$$

$$u^i_j = \tilde{f}_j(\tilde{y}^{SP}, \tilde{u}^{SP}, y^i_j), \quad j = 0, \ldots, N-1 \tag{4.2.1e}$$

$$0 = \quad h^{Ref}(y^{Ref}, u^{Ref}, \tilde{y}^{SP}, \tilde{u}^{SP}) \tag{4.2.1f}$$

$$0 \leq \quad g^{Ref}(y^{Ref}_{j+1}, u^{Ref}_j), \quad j = 0, \ldots, N-1 \tag{4.2.1g}$$

In the above, ϕ_{econ} is the economic objective function of the primary DRTO; ϕ^i_{econ} is the economic objective function value for scenario i; ρ^i is the objective weighting for scenario i; u, x, y are the system inputs, states, and outputs, respectively; the tilde ($\tilde{}$) denotes variables and functions associated with the MPC subproblems; the Ref superscript denotes reference trajectory variables; the SP superscript denotes set-point trajectories for the MPC; f^i is the set of dynamic equations for the plant model; h^i is the algebraic plant model relating states to outputs; g is the set of inequality process constraints; h^{Ref} is the functional relationship between the reference trajectory and the set-point trajectories for each MPC subproblem; g^{Ref} is the set of bounds on the reference points; i denotes the uncertain scenario; j denotes the DRTO time point; and u^i, x^i, y^i are composite vectors of the inputs, states, and outputs, respectively, for plant scenario i over the DRTO prediction horizon.

The degrees of freedom of the DRTO problem are the set-point trajectories which

are subsequently provided to the MPC. The set-point trajectories must be the same across all scenarios such that there is a single, unified set-point trajectory which may be applied to the actual plant MPC. The input values, including the initial input move which must be the same across all scenarios, are determined entirely by the MPC subproblems, and so are not degrees of freedom of the DRTO problem.

4.2.2 MPC subproblems

The MPC subproblems are intended to capture the behavior of the actual MPC in use in the plant control system, thus there is an MPC subproblem associated with every control interval along the DRTO horizon. The MPC used in this work constitutes a standard QDMC formulation [18], but with the use of a state space rather than step response model. In this formulation, the objective function is a sum of quadratic terms for output and input set-point tracking, and for input move suppression. The inputs are bounded at the MPC level in this formulation. The MPC subproblems take the form for $i = 1, \ldots, N_s$:

$$\min_{\tilde{u}^i_{j,k}} \quad \tilde{\phi}^i_j = \sum_{k=1}^{p} (\tilde{y}^i_{j,k} - \tilde{y}^{\text{sp}}_{j,k})^T Q (\tilde{y}^i_{j,k} - \tilde{y}^{\text{sp}}_{j,k}) + \sum_{k=0}^{m-1} (\Delta \tilde{u}^i_{j,k})^T R (\Delta \tilde{u}^i_{j,k}) + \sum_{k=0}^{m-1} (\tilde{u}^i_{j,k} - \tilde{u}^{\text{sp}}_{j,k})^T S (\tilde{u}^i_{j,k} - u^{\text{sp}}_{j,k}) \tag{4.2.2a}$$

$$\text{s.t.} \quad \tilde{x}^i_{j,k+1} = A\tilde{x}^i_{j,k} + B\tilde{u}^i_{j,k}, \quad k = 0, \ldots, m-1 \tag{4.2.2b}$$

$$\tilde{x}^i_{j,k+1} = A\tilde{x}^i_{j,k} + B\tilde{u}^i_{j,m-1}, \quad k = m, \ldots, p-1 \tag{4.2.2c}$$

$$\tilde{y}^i_{j,k} = C\tilde{x}^i_{j,k} + \tilde{d}^i_{j,k}, \quad k = 1, \ldots, p \tag{4.2.2d}$$

$$\Delta \tilde{u}^i_{j,k} = \tilde{u}^i_{j,k} - \tilde{u}^i_{j,k-1}, \quad k = 0, \ldots, m-1 \tag{4.2.2e}$$

$$u_{\min} \leq \tilde{u}^i_{j,k} \leq u_{\max}, \quad k = 0, \ldots, m-1 \tag{4.2.2f}$$

In the above, $\tilde{\phi}^i_j$ is the MPC objective function corresponding to step j of the DRTO prediction horizon; m and p are the MPC control and prediction horizons, respectively; Q, R, S are the output deviation weighting matrix, input move penalty matrix, and input deviation weighting matrix, respectively; A, B, C are the linear, or linearized, state-space matrices for the plant system dynamic model; u_{min}, u_{max} are the input bounds; $d^i_{j,k}$ is the disturbance estimate; and j, k are the time steps for the DRTO and MPC, respectively.

4.2.3 Interactions and Feedback

The set-point trajectories determined by the primary DRTO problem are provided to the MPC subproblems as the set-point to be reached, consistent with the information exchange between the DRTO and actual plant MPC. A subset of the unified reference trajectory is used as the set-point trajectory for each of the MPC subproblems, with

the specific portion of the reference trajectory extracted depending on the time point along the DRTO prediction horizon and the length of the MPC input and output horizons.

$$y^{Ref}_{j+k} = \tilde{y}^{SP}_{j,k}, \quad j = 1, \ldots, N \quad k = 1, \ldots, p \tag{4.2.3a}$$

$$u^{Ref}_{j+k} = \tilde{u}^{SP}_{j,k}, \quad j = 1, \ldots, N \quad k = 0, \ldots, m-1 \tag{4.2.3b}$$

The other primary interaction between the DRTO and MPC subproblems is the determination of the input moves by the MPC, which are also used as the input moves for the plant model at the DRTO level. Specifically, the first input determined by the MPC subproblem at a given time in a given scenario branch is used as the DRTO input for that time point.

$$u^i_j = \tilde{u}^i_{j,0}, \quad j = 0, \ldots, N-1 \tag{4.2.4}$$

Further interactions between the DRTO and MPC subproblems are interactions which emulate the utlization of plant measurements by the actual MPC. In a standard QDMC formulation, the disturbance estimate is computed as the difference between the predicted outputs of the MPC and the measured outputs of the plant, and is held constant over the MPC prediction horizon. This is emulated in the CL-DRTO fromulation as follows. The first MPC subproblem of each DRTO execution uses the plant measurement to compute the disturbance estimate as described. However, MPC subproblems within the DRTO which start after the initial time point do not

have access to future plant measurements. Therefore, they use the DRTO model output prediction as a proxy for the plant measurement. This is preferable to not updating the disturbance estimate, since the DRTO uses a nonlinear model to predict the plant outputs and should therefore be more accurate than the linear model of the MPC in predicting the actual plant behavior. The disturbance estimation within the CL-DRTO scheme takes the form, for $i = 1, \ldots, N_s$:

$$\tilde{d}^i_{0,0} = y^{\mathrm{m}} - C\tilde{x}^i_{0,0} \tag{4.2.5a}$$

$$\tilde{d}^i_{j,0} = y^i_j - C\tilde{x}^i_{j,0}, \quad j = 1, \ldots, N-1 \tag{4.2.5b}$$

$$\tilde{d}^i_{j,k} = \tilde{d}^i_{j,0}, \quad j = 1, \ldots, N-1 \quad k = 1, \ldots, p \tag{4.2.5c}$$

In the above, y^{m} is the plant measurement at the time of execution of the DRTO.

The initial states ($\tilde{x}^i_{j,0}$) and previous inputs ($\tilde{u}^i_{j,-1}$) for the MPC subproblems are also required. The MPC subproblems corresponding to the initial time step of each DRTO execution use the predicted states of the MPC subproblems in the previous DRTO execution at the time point corresponding to the current DRTO execution. For subsequent MPC subproblems, the predicted states of the previous MPC subproblem are used instead. Similarly, the previous input for the first time step MPC subproblem is the same as the input implemented on the actual plant. The next subproblems instead use the computed inputs for that time step from the previous MPC subproblems. For both of these updates, the information is taken from the MPC subproblem within that scenario prediction.

4.2.4 Solution Strategy

The DRTO with embedded MPC subproblems represents a multi-level optimization problem. There are a number of solution methods which may be used for this type of problem. Here we follow the same method used in Jamaludin and Swartz [27] and solve it simultaneousfexly by using the Karush-Kuhn-Tucker (KKT) conditions of the MPC subproblems. Since the MPC subproblems with a standard QDMC formulation are convex quadratic problems, they can be exactly represented by their first-order KKT conditions. These can then be included as constraints in the primary DRTO problem. The transformation of a convex QP to its equivalent KKT form is outlined below.

The general form of a QP, of which type the inner MPC subproblems are, can be written as

$$\min_{z} \quad \frac{1}{2} z^T H z + g^T z \tag{4.2.6a}$$

$$s.t. \quad Az = b \tag{4.2.6b}$$

$$z \geq 0 \tag{4.2.6c}$$

The corresponding KKT conditions are given by

$$Hz + g - A^T \lambda - \nu = 0 \tag{4.2.7a}$$

$$Az - b = 0 \tag{4.2.7b}$$

$$z_i \nu_i = 0, \quad \forall i \tag{4.2.7c}$$

$$(z, \nu) \geq 0 \tag{4.2.7d}$$

In the above, λ, ν are the Lagrange multipliers for the equality and inequality constraints, respectively.

The reformulation of the specific MPC problem here follows that applied in Baker and Swartz [2]. The complementarity constraints are handled here with an exact penalty approach where the products of the respective complementarity variables are summed and added to the objective function with an appropriately large weight. This method for handling complementarity constraints is proposed in Ralph and Wright [66].

4.2.5 Input Clipping Approximation

The presented formulation and solution strategy for the robust CL-DRTO is computationally intensive. Since this is a real-time optimization strategy, it is important that the solution time is reasonable compared to the frequency of execution of the process being optimized. We therefore introduce approximations of the previous formulation to reduce the computation time with limited loss of performance.

One such method is that of input clipping, wherein the input constraints within the MPC subproblems are removed the input bounds are reintroduced at the DRTO level by clipping the inputs determined by the MPC subproblem to their nearest bound if the computed value is outside those bounds. This approximation technique was proposed for CL-DRTO in Jamaludin and Swartz [29], and applied in the DRTO coordination of distributed MPC systems in Li and Swartz [41]. The input clipping can be accomplished with the use of slack variables as shown below. As this clipping only occurs when the computed inputs are outside the true input bounds, any case

where the inputs do not saturate will result in identical performance to the rigorous method. The MPC subproblems, in this formulation, are unconstrained QPs. The input clipping may be described mathematically as

$$u_j^i = \begin{cases} u_{min}, & \tilde{u}_{j,0}^i < u_{min} \\ \tilde{u}_{j,0}^i, & u_{min} \leq \tilde{u}_{j,0}^i \leq u_{max} \\ u_{max}, & \tilde{u}_{j,0}^i > u_{max} \end{cases} \tag{4.2.8}$$

Baker and Swartz [2] and Jamaludin and Swartz [29] show that this formulation can be expressed without the use of binary variables by introducing slack variables as follows

$$\tilde{u}_{j,0}^i + \eta_j^{1,i} - \eta_j^{2,i} = u_j^i \tag{4.2.9a}$$

$$u_j^i - u_{min} = \mu_j^{1,i} \tag{4.2.9b}$$

$$u_{max} - u_j^i = \mu_j^{2,i} \tag{4.2.9c}$$

$$(\eta_j^{1,i})^T \mu_j^{1,i} = 0 \tag{4.2.9d}$$

$$(\eta_j^{2,i})^T \mu_j^{2,i} = 0 \tag{4.2.9e}$$

$$(\eta_j^{1,i}, \eta_j^{2,i}, \mu_j^{1,i}, \mu_j^{2,i}) \geq 0 \tag{4.2.9f}$$

The complementarity constraints of Eqs. (4.2.9d) and (4.2.9e) can be handled with the use of an exact penalty approach as described previously with the KKT conditions.

The removal of the input constraints and clipping the resulting solution at the DRTO level reduces the number of equations to be solved for each MPC subproblem. However, more substantial improvements in computation time can be achieved by taking advantage of the fact that the MPC subproblems are now unconstrained QPs which can be solved analytically. This can be accomplished by rearranging the MPC subproblems using the equality constraints to eliminate the state variables so that the QP is expressed with solely the input changes as decision variables.

The reformulation of the unconstrained MPC subproblems in terms of the input moves can subsequently be expressed as

$$\min_{\Delta \underline{u}_j^i} \quad \phi_j^i = (\underline{y}_j^i - \underline{y}_j^{SP})^T \underline{Q} (\underline{y}_j^i - \underline{y}_j^{SP}) + (\Delta \underline{u}_j^i)^T \underline{R} (\Delta \underline{u}_j^i) + (\underline{u}_j^i - \underline{u}_j^{SP})^T \underline{S} (\underline{u}_j^i - \underline{u}_j^{SP}) \tag{4.2.10a}$$

$$\text{s.t.} \quad \underline{y}_j^i = \underline{A} \Delta \underline{u}_j^i + \underline{b}_j^i \tag{4.2.10b}$$

$$\underline{u}_j^i = \underline{u}_{j,-1}^i + I_L \Delta \underline{u}_j^i \tag{4.2.10c}$$

where $\underline{u}_j^i$, $\Delta \underline{u}_j^i$, $\underline{y}_j^i$, $\underline{u}_j^{SP}$, $\underline{y}_j^{SP}$ are time-based composite vectors of the inputs, input changes, outputs, input set-points, and output set-points, respectively; $\underline{Q}$, $\underline{R}$, $\underline{S}$ are block diagonal matrices comprised of the weighting matrices Q, R, S, respectively; $\underline{A}$ is a composite matrix of the dynamic system matrices A, B, C; I_L is a block lower triangular matrix of identity matrices; and $\underline{b}_j^i$ is a time-based composite vector of output predictions based on initial states and inputs summed with disturbance estimates. More detail on the form of the composite vectors may be found in Li and Swartz [41].

This reformulation can then be used to establish an analytical solution by substituting the above equality constraints directly into the objective and then setting the resulting gradient to zero, giving

$$\Delta \underline{u}_j^i = (\underline{A}^T \underline{Q} \underline{A} + \underline{R} + I_L^T \underline{S} I_L)^{-1} [\underline{A}^T \underline{Q} (\underline{y}_j^{SP} - \underline{b}_j^i) + I_L^T \underline{S} (\underline{u}_j^{SP} - \underline{u}_{j,-1}^i)] \tag{4.2.11}$$

$$= K_1 (\underline{y}_j^{SP} - \underline{b}_j^i) + K_2 (\underline{u}_j^{SP} - \underline{u}_{j,-1}^i) \tag{4.2.12}$$

Much of this computational work can be done a priori since the model matrices do not change over time, so the equation matrices can be pre-computed. Furthermore, only a subset of input equations needs be included, since only the first input move is required to be provided to the DRTO. Both Jamaludin and Swartz [29] and Li and Swartz [41] show that the input clipping approximation results in significant reductions in computation time with limited loss of performance. Therefore, the input clipping approximation is used in the proposed robust DRTO formulation and in the case studies that follow.

4.3 Case Studies

4.3.1 Case Study 1 - SISO

The first case study in which the robust DRTO is applied is a simple single-input-single-output (SISO) system, where the performance of the robust DRTO is compared against that of a nominal DRTO. The system is a single reaction CSTR with one inlet stream, one outlet, and one reaction. The reaction kinetics are governed by the

Michaelis-Menten equation. The control system is SISO with two states, with the governing equations taken from Gao [17] and can be expressed as

$$\frac{dC}{dt} = D \cdot (C_{in} - C) - \frac{V_m \cdot C}{K_s + C} \tag{4.3.1}$$

$$\frac{dP}{dt} = \frac{V_m \cdot C}{K_s + C} - D \tag{4.3.2}$$

where the differential states are the concentration of reactant, C, and concentration of product, P; the inlet concentration, C_{in}, is the input; and the concentration of product, P, is the output. D, V_m, K_s are the ratio of flow rate to reactor volume, the maximum reaction rate, and the reaction constant, respectively, and are parameters. The system was discretized using the implicit Euler method. The uncertain parameter was chosen to be the maximum reaction rate, V_m, with a $\pm 20\%$ uncertainty.

The DRTO uses three uncertain scenarios, corresponding to the nominal, minimum, and maximum parameter values. The parameter is assumed to have a uniform probability distribution within the expected range and so the scenarios weights are all equal to 1. The DRTO schemes are evaluated in three different realizations where the plant simulation uses the nominal, minimum, and maximum parameter values, respectively.

The objective function used here is a profit band objective, where the system generates revenue only when the product concentration is within a specified quality range, as implemented in Jamaludin and Swartz [27]. The input is also included as a cost. The profit band is approximated using hyperbolic tangent functions to approximate binary switching functions. The objective function can be expressed as

follows

$$\phi_{econ} = \sum_i \rho^i \sum_{j=0}^{N} 10 \cdot R^i_{1,j} \cdot R^i_{2,j} - u^i_j \tag{4.3.3}$$

$$R^i_{1,j} = \frac{1}{2}\tanh\left(\gamma[y^i_j - (1-\delta)y^{tar}_j]\right) + \frac{1}{2} \tag{4.3.4}$$

$$R^i_{2,j} = \frac{1}{2}\tanh\left(\gamma[(1+\delta)y^{tar}_j - y^i_j]\right) + \frac{1}{2} \tag{4.3.5}$$

where γ and δ are fixed parameters which determine the slope of the tanh functions and the half-width of the profit band, respectively; u^i_j and y^i_j are the inputs and outputs of the system, respectively, in this case they represent the inlet concentration of reactant, C_{in}, and the concentration of product, P. The product of tanh switching function approximations in 4.3.3 gives $R^i_{1,j} \cdot R^i_{2,j} \approx 1$ if $(1-\delta)y^{tar}_j \leq y^i_j \leq (1+\delta)y^{tar}_j$; and is 0 otherwise. The DRTO and MPC parameter values are shown in Table 4.1.

Table 4.1: DRTO and MPC parameters for Case Study 1.

Parameter	**Description**	**Value**
N	DRTO horizon	40
Δt_{DRTO}	DRTO sample time	2 hr
Δt_{MPC}	MPC sample time	1 hr
p	MPC prediction horizon	10
m	MPC control horizon	2
Q	Output tracking weight	10
R	Input move suppression weight	1
S	Input tracking weight	0
ρ	Complementarity Penalty	1000
T_s	Time Step	1 hr

Subcase 1: Small transition with constrained output

The output in Subcase 1 is subject to a one-sided economic penalty term for exceeding a value of 0.84, with a penalty weight of 100. The profit band is defined to be between 0.816 and 0.836 ($y_j^{tar} = 0.826$, $\delta = 0.01$) with $\gamma = 100$. The results for Case Study 1, Subcase 1 are shown in Table 4.2 and Fig. 4.2.

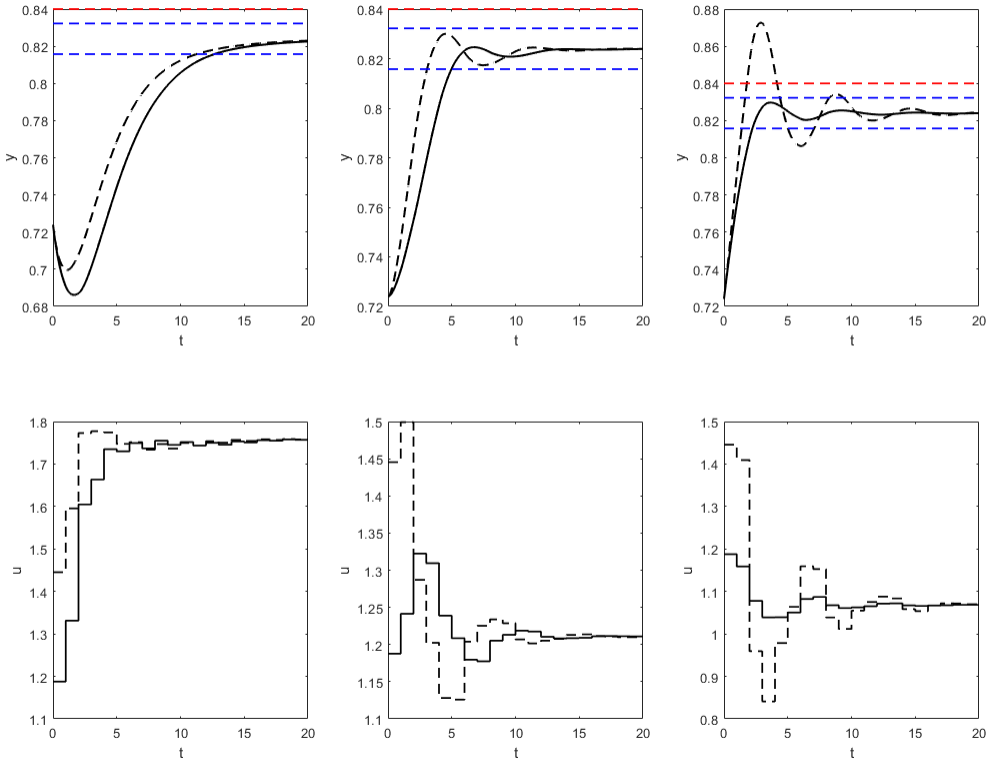

Figure 4.2: Plots of the input (bottom) and output (top) for the minimum (left), nominal (middle), and maximum (right) parameter value plant realizations for Case Study 1, Subcase 1. The red dashed line is the economic penalty bound, the blue dashed lines are the bounds of the profit band, the black dashed lines are the nominal CL-DRTO trajectory, and the solid black line is the robust CL-DRTO trajectory.

The results show the robust DRTO outperforming the nominal in terms of

Table 4.2: Summary of economic objective function values for Case Study 1, Subcase 1, comparing the nominal and robust CL-DRTO methods.

Plant Model	**Nominal DRTO**	**Robust DRTO**	**Percent Change**
Minimum	51.48	38.70	-24.8
Nominal	143.6	124.7	-13.2
Maximum	98.08	155.4	58.4
Expected Value	97.72	106.3	8.78

economics and avoiding exceeding the upper economic penalty bound of 0.84. This can be seen in the maximum parameter value scenario, where the nominal DRTO overshoots the profit band and the economic penalty bound, resulting in both reduced profit accumulation during this overshoot period and a penalty applied to the final profit value. In contrast, the robust DRTO manages to stay in the profit band once it initially reaches it, avoiding losing profit accumulation time as well as the penalty. In the minimum and nominal parameter value scenarios, neither the nominal DRTO or robust DRTO overshoot. In these scenarios, the robust DRTO reaches the profit band slightly later than the nominal DRTO, resulting in reduced economic returns. However, the expected economic performance is still greater for the robust DRTO since it outperforms the nominal DRTO by a larger margin in the maximum scenario.

Subcase 2: Large transition

The second subcase is not subject to an upper bound economic penalty term. It involves a larger transition to the profit band, with said band also being a wider target. The profit band is now defined to be from 0.95 to 1.05 ($y_j^{tar} = 1.00$, $\delta = 0.05$) with an unchanged $\gamma = 100$. The simulation time is also increased to allow for the

larger transition to occur and then reach steady state. The results for Case Study 1, Subcase 2 are shown in Table 4.3 and Fig. 4.3.

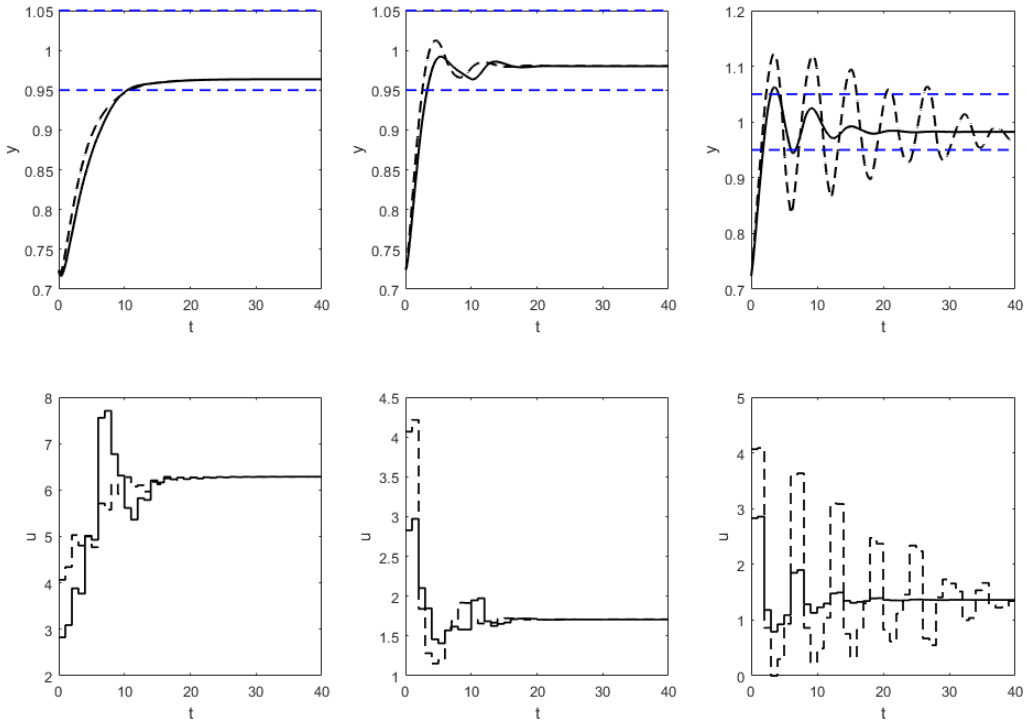

Figure 4.3: Plots of the input (bottom) and output (top) for the minimum (left), nominal (middle), and maximum (right) parameter value plant realizations for Case Study 1, Subcase 2. The blue dashed lines are the bounds of the profit band, the black dashed lines are the nominal CL-DRTO trajectory, and the solid black line is the robust CL-DRTO trajectory.

The robust DRTO here outperforms the nominal DRTO in terms of expected economic returns over the three tested realizations, as can be seen in Table 4.3. Once again, the largest difference is in the maximum parameter value scenario, where the robust DRTO substantially outperforms the nominal DRTO by reducing the overshoot and oscillation (Fig. 4.3). In contrast to the previous subcase, the

Table 4.3: Summary of economic objective function values for Case Study 1, Subcase 2, comparing the nominal and robust CL-DRTO methods.

Plant Model	**Nominal DRTO**	**Robust DRTO**	**Percent Change**
Minimum	48.03	52.87	10.1
Nominal	301.1	294.9	-2.06
Maximum	142.9	305.3	114
Expected Value	164.0	217.7	32.7

robust DRTO also outperforms the nominal DRTO in the minimum parameter value scenario. As expected, the nominal DRTO still outperforms the robust DRTO in the nominal scenario, as the plant simulation behavior in this scenario is in accordance with the behavior predicted by the nominal DRTO.

For Case Study 1, the robust DRTO outperforms the nominal DRTO in terms of expected economic performance in both subcases. This is primarily accomplished by reducing the overshoot in the maximum parameter value scenario realization. In Subcase 1, this overshoot causes an economic penalty to be incurred for the nominal DRTO in addition to reduced time in the profit band; in Subcase 2, it causes significant oscillation leading to reduced time in the profit band. The robust DRTO, in both subcases, exhibits reduced economic performance in the nominal parameter value scenario, but more than makes up for this loss in the other realizations.

4.3.2 Case Study 2 - MIMO

The case study here uses the polymerization case study presented previously in Jamaludin and Swartz [27]. It is a MIMO system with three inputs, three outputs, and six states. The system inputs are the inlet flowrates to the reactor of the monomer,

an initiator, and a solvent, respectively. The concentrations of the respective reactant in the inlet streams are assumed constant. The outputs of the system are the number average molecular weight (NAMW) of the polymer product, the temperature in the reactor, and the concentration of monomer in the reactor. The variables can range across several orders of magnitude, and so the system is first scaled to a dimensionless form to enable easier solution. This case study was chosen due to its nonlinear reactor equations, its multi-variable set-points at the MPC level, and its relatively slow transition time. The long transition time encourages improvements in transition to reduce the time the reactor is not producing polymer product. This case study again uses the profit band objective function as in Case Study 1. Specifically, the NAMW of the polymer product must be within ±1% of 68.9 kg/mol ($y_j^{tar} = 68.9$, $\delta = 0.689$) with $\gamma = 10$. All three inputs are assumed to be equal in cost. Additionally, the temperature within the reactor must be maintained between 320 and 325 K and the NAMW must stay below 70 kg/mol. These output constraint are implemented as soft constraints with a penalty weight of 100. The DRTO and MPC parameter values for Case Study 2 are shown in Table 4.4.

In this case, the MPC tracking weights are non-zero for the first two outputs, the NAMW and temperature, respectively, and the third input, the inlet flowrate of monomer. This means the DRTO provides reference trajectories to the MPC for these two outputs and one input. The third output, the concentration of monomer, and the first two inputs, the inlet flowrates of initiator and coolant, respectively, do not have reference trajectories as their MPC tracking weights are zero.

Table 4.4: DRTO and MPC parameters for Case Study 2.

Parameter	**Description**	**Value**
N	DRTO horizon	20
Δt_{DRTO}	DRTO sample time	5 hr
Δt_{MPC}	MPC sample time	1 hr
p	MPC prediction horizon	10
m	MPC control horizon	2
Q	Output tracking weight	diag(1,1,0)
R	Input move suppression weight	diag(50,15,5)
S	Input tracking weight	diag(0,0,1)
ρ	Complementarity Penalty	1000
T_s	Time Step	1 hr

Three scenario robust DRTO

The first evaluation of the robust DRTO in this case study is to evaluate the performance of the robust DRTO with three modeled scenarios against that of the nominal DRTO. The two DRTO formulations are evaluated first over three possible plant realizations corresponding to the minimum, nominal, and maximum parameter values. This is a similar setup to that shown in Case Study 1. The results for the three scenario robust DRTO and single scenario nominal DRTO evaluated over these three plant realizations are shown in Tables 4.5 and 4.6.

Table 4.5: Summary of economic objective function for nominal, and three scenario robust DRTO, in characteristic scenarios for Case Study 2.

Plant Model	**Nominal DRTO**	**Robust DRTO**	**Percent Change**
Minimum	53.80	67.61	25.7
Nominal	62.95	61.19	-2.80
Maximum	49.19	60.95	23.9
Expected Value	55.31	63.25	14.4

Table 4.6: Summary of sum of squared constraint violation for nominal, and three scenario robust DRTO, in characteristic scenarios for Case Study 2.

Plant Model	**Nominal DRTO**	**Robust DRTO**	**Percent Change**
Minimum	2.097	1.806	-13.9
Nominal	0.762	0.447	-41.3
Maximum	0	0.012	-
Expected Value	0.954	0.755	-20.9

As shown in Table 4.5, the robust DRTO outperforms the nominal DRTO in terms of economics. It should be noted that in this case study, unlike in the previous case study, the constraint violation is not included in the economic objective function values shown in the tables. The profit is higher in the minimum and maximum parameter realization scenarios for the robust DRTO and slightly lower in the nominal scenario. This is consistent with Case Study 1, and is expected given that it is in this scenario that the deterministic nominal DRTO model matches the plant simulation. In terms of expected profit, the robust DRTO improves on the nominal DRTO by about 14% when evaluated in these three scenario realizations.

The robust DRTO is also able to reduce the constraint violation (specifically the upper bound) of the NAMW relative to the nominal DRTO, as shown in Table 4.6. The constraint violation shown is only for the NAMW; the temperature constraint violation is negligible within a tolerance of 10^{-3}. Here, the robust DRTO is able to reduce the expected constraint violation by about 21%.

Plots of the inputs and outputs over time for these characteristic scenarios are shown in Figs. 4.4-4.6. In all three scenario realizations, the overall trajectories of the plant simulation under the control of the two algorithms are similar. The only noticeable difference is a slight decrease in the inlet flowrates during transition (the

ultimate steady state is not affected), resulting in a decrease in the total cost for the plant. While too slight to be visible in the plots, the robust DRTO marginally improves on the time to viable product NAMW band in the minimum and maximum parameter value scenarios, resulting in a marginal increase in revenue. However, the majority of the expected increase in profit, shown in Table 4.5, is from the decrease in input costs.

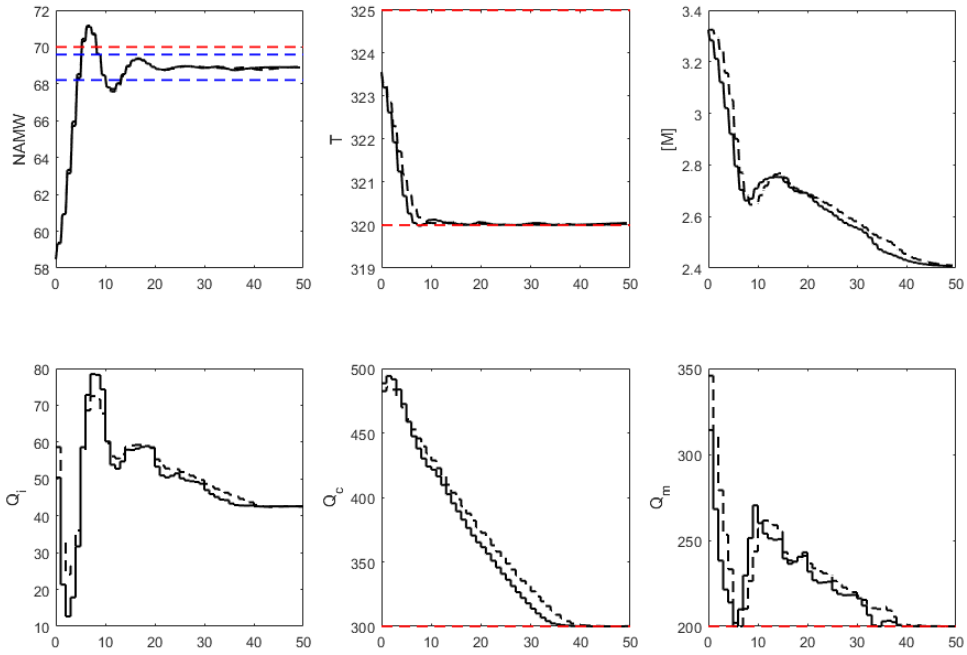

Figure 4.4: Plot of Inputs and Outputs for nominal and robust DRTO, minimum parameter value scenario.

An additional test for the robust DRTO is performed where 20 random parameter values between the minimum and maximum predicted values are chosen and the nominal and robust DRTO performances compared in these simulations. The results are summarized in Table 4.7 and show that the overall trends from the characteristic

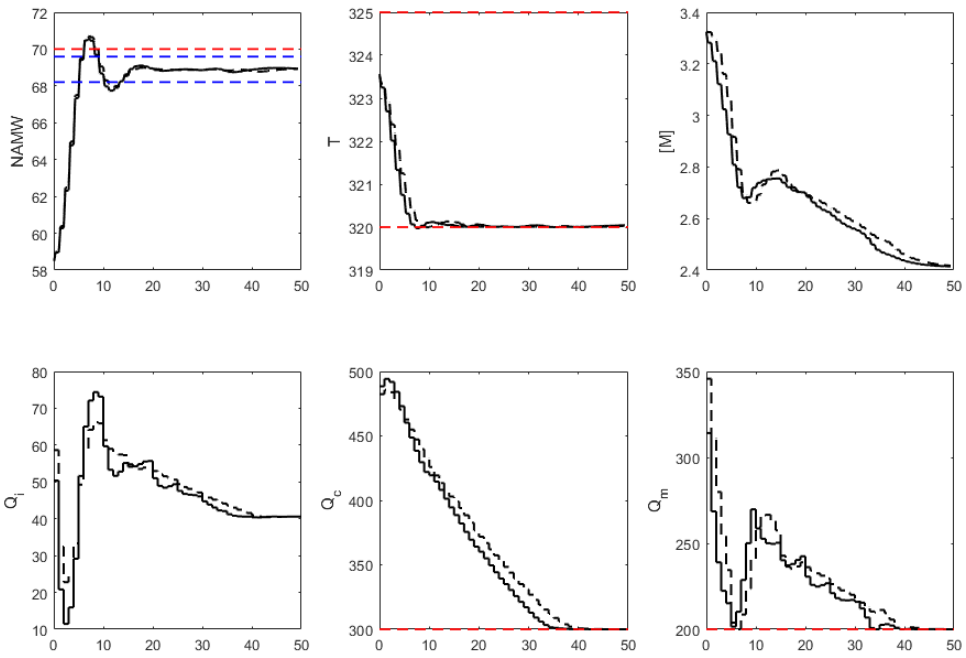

Figure 4.5: Plot of Inputs and Outputs for nominal and robust DRTO, nominal parameter value scenario.

scenarios testing are maintained. The expected economic improvement by the robust DRTO decreases, as is expected given that the previous characteristic scenarios are the exact scenarios modeled and the majority of the 20 random test cases are not directly accounted for in the robust DRTO formulation. The improvement of the robust DRTO over the nominal DRTO is still significant, however, at about 8%. The constraint violation reduction for the NAMW is actually larger for these scenarios, at about 36%, showing that the robust DRTO is still effective at avoiding constraint violation even when the plant behavior does not precisely match the predicted scenarios. The temperature constraint violation is still within the same tolerance. Finally, the solution time per DRTO execution is also shown in Table 4.7,

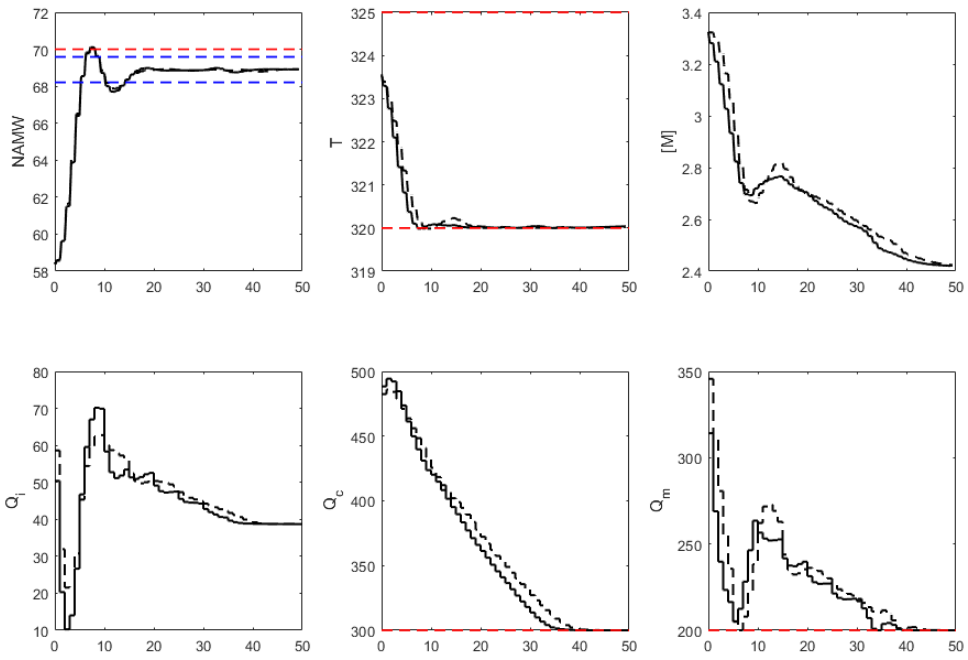

Figure 4.6: Plot of Inputs and Outputs for nominal and robust DRTO, maximum parameter value scenario.

with the nominal DRTO being approximately 5 times faster than the robust DRTO. All solution times reported are using an Intel Core i7 8700 (3.2 GHz, 6 cores, 12 processors).

Effect of number of scenarios

The above case studies all involve a robust DRTO with three modeled scenarios, the expected parameter value (nominal) and the two extremes of the parameter value (minimum, maximum). However, this may not be sufficient to capture the expected plant behavior within this range, especially for nonlinear systems. Therefore, in this section, a robust DRTO with more than three scenarios is tested on the same set of

Table 4.7: Metrics of expected performance for 20 random parameter value scenarios of nominal and three scenario robust DRTO, Case Study 2.

Metric	Nominal DRTO	Robust DRTO	Percent Change
Profit	55.47	59.94	8.05
Constraint Violation	1.009	0.6410	-36.4
Solution Time (s)	2.055	10.02	388

20 random parameter value scenario realizations. It should be noted that the overall range of parameter values remains constant, and the added scenarios are all within this range and are evenly spaced and evenly weighted in accordance with an assumed uniform distribution.

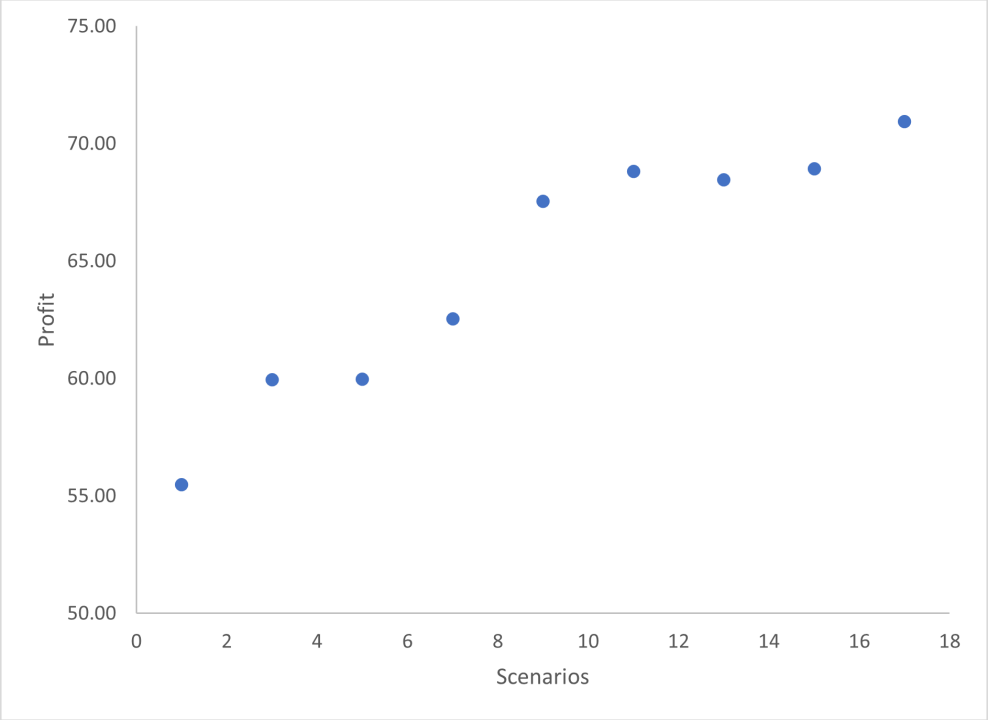

Figure 4.7: Plot of expected profit across 20 random simulated scenarios by number of DRTO modeled scenarios.

As shown in Fig. 4.7 and Table 4.8, the overall expected profit for the robust

Table 4.8: Metrics of expected performance for 20 random parameter value scenarios by number of modeled scenarios in Robust DRTO, Case Study 2.

No. Scenarios	**Profit**	**Constraint Violation**	**Sol. Time (s)**
1	55.47	1.009	2.05
3	59.94	0.641	10.02
5	59.98	0.637	25.87
7	62.55	0.582	76.58
9	67.55	0.543	134.9
11	68.82	0.529	238.9
13	68.47	0.544	299.5
15	69.94	0.528	543.7
17	70.93	0.517	803.1

DRTO shows a noticeable increase from 1 to 3 scenarios (as shown previously) and then further increases beyond 3 scenarios, especially past 5 scenarios, up to 9 modeled scenarios. Beyond 9 scenarios the expected profit seems to exhibit diminishing returns. This may suggest that 9 scenarios is the superior choice for improved economic benefit prior to diminishing returns of increased scenario modeling.

The constraint violation for the robust DRTO with varying number of scenarios shows, in Table 4.8 and Fig. 4.8, a more clear case of diminishing returns where, for the NAMW constraint violation, there is a large reduction from one scenario to three scenarios, and then smaller reductions for increased scenarios beyond three. This suggests that for the purposes of constraint violation reduction, three scenarios modeled by the robust DRTO may be adequate. Again, the constraint violation for the temperature is negligible.

The cost of this improvement in expected profit for increased number of scenarios is that the computation time also increases, as shown in Fig. 4.9. While the three

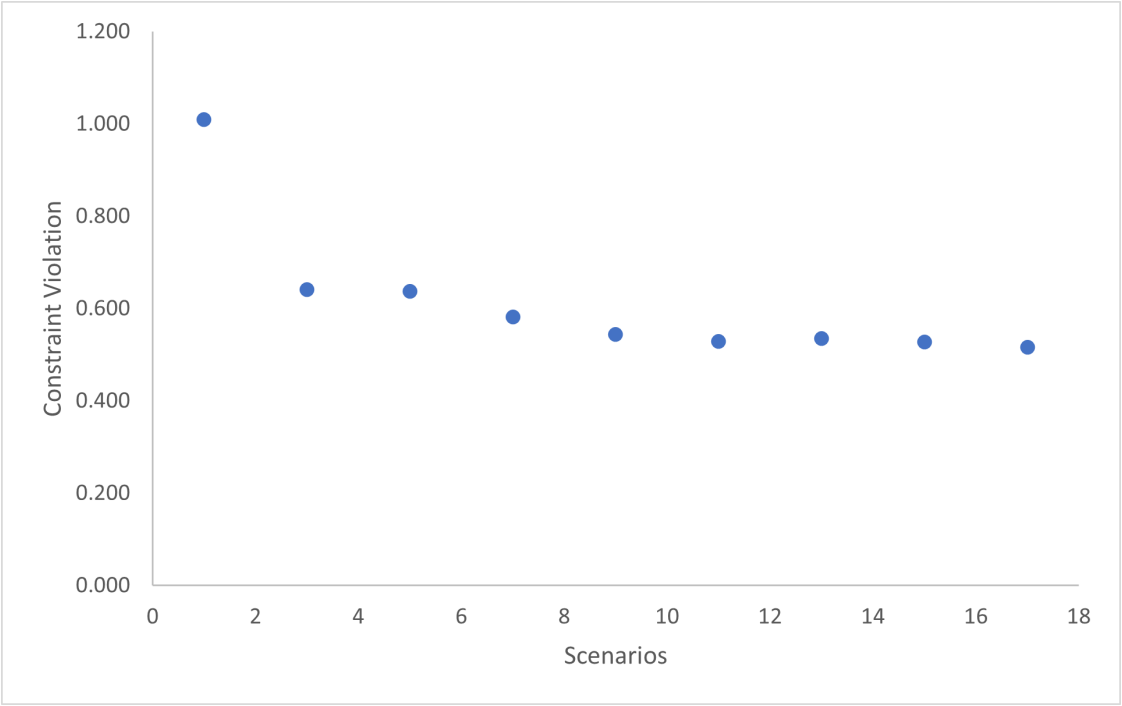

Figure 4.8: Plot of expected NAMW constraint violation across 20 random simulated scenarios by number of DRTO modeled scenarios.

scenario robust DRTO averages about 10 s per execution, the 9 scenario DRTO averages about 135 s, and the 17 scenario DRTO about 800 s. Therefore, although the performance improves in terms of economics and, to a lesser extent, constraint violation, the process to which the robust DRTO is applied must be examined to determine if a higher number of scenarios is viable to solve in the required DRTO iteration time.

Multiple uncertain parameters

In the previous case studies, it is assumed that there is one source of uncertainty in the process. For this case study, it will be assumed that there are instead two uncertain parameters. These uncertain parameters will be modeled by the robust

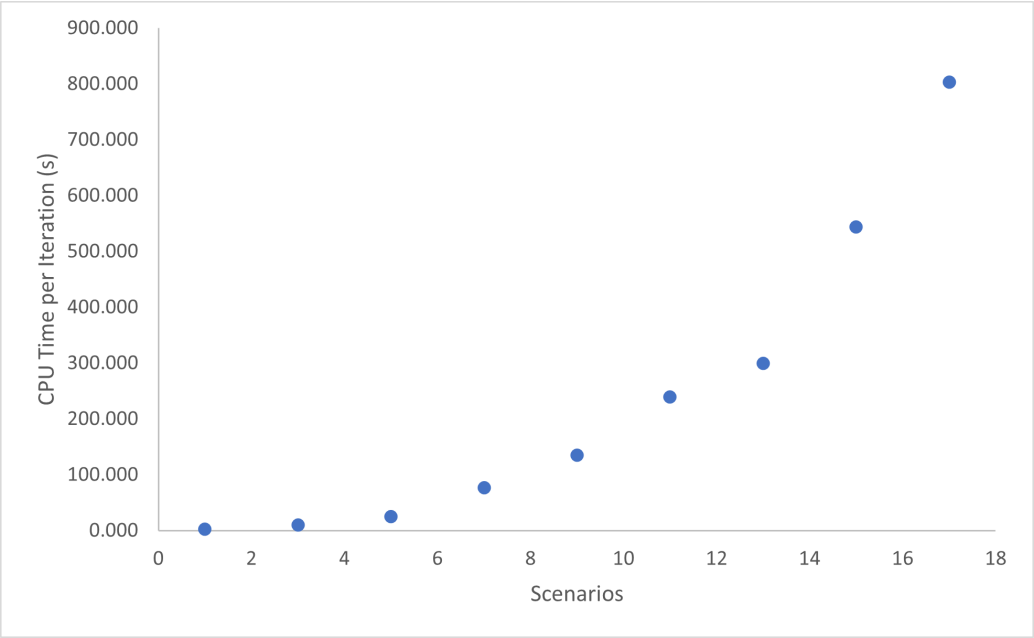

Figure 4.9: Plot of average computation time (s) per DRTO iteration by number of DRTO modeled scenarios.

DRTO by including scenario predictions for each pair of parameter values for their minimum, nominal, and maximum values. Therefore, there will be 9 total scenarios modeled with three possible values for each uncertain parameter. One of the uncertain parameters is the pre-exponential rate constant for the dimerization reaction (the same parameter considered uncertain in previous case studies) and the other is the inlet concentration of monomer. Since these two parameters are not considered to be related, they are treated as uncorrelated in this case study. The robust DRTO is again compared against a single scenario nominal DRTO in the characteristic scenario simulations (the 9 scenarios modeled by the robust DRTO) and in 20 scenarios with random parameter values within the expected range.

Overall, Table 4.9 shows that the trends seen previously in the single uncertain parameter case also hold here, but the magnitude of the changes from the nominal to

Table 4.9: Metrics of expected performance for characteristic scenarios (the scenarios modeled in the robust scenario tree) of nominal and robust DRTO with two uncertain parameters.

Metric	**Nominal DRTO**	**Robust DRTO**	**Percent Change**
Profit	48.30	75.31	55.9
Constraint Violation	1.485	0.9337	-37.1
Solution Time (s)	2.079	112.8	5330

robust DRTO are increased. The increase in profit is over 50%, compared to about 14% in the single uncertain parameter case. Furthermore, decrease in constraint violation for the NAMW has improved substantially from about 20% to about 37%. The constraint violation for temperature is still negligible.

Figs. 4.10-4.12 show the time plots of the input and output trajectories for three chosen scenarios of the nine characteristic scenarios. Specifically, the nominal values of both parameters is shown in Figure 4.10, the minimum value of the pre-exponential rate constant paired with the maximum value of the inlet concentration of monomer is shown in Fig. 4.11, and the maximum rate constant with minimum concentration in Fig. 4.12. These were chosen to be shown as they represent the central scenario (nominal-nominal) and the two scenarios which, on examination, exhibited the most extreme behavior (min-max and max-min). The differences in the trajectories of the two DRTOs are more noticeable than in the single uncertain parameter case. As before, the robust DRTO drives the system to have lower inlet flowrates during transition, though by a larger margin than previously, resulting in reduced costs. Also, the robust DRTO now noticeably reduces the overshoot and oscillations of the NAMW, particularly in the min-max and max-min scenarios (Figs. 4.11 and 4.12) thereby increasing the revenue generated by staying in the profit band for a longer

period than the nominal DRTO.

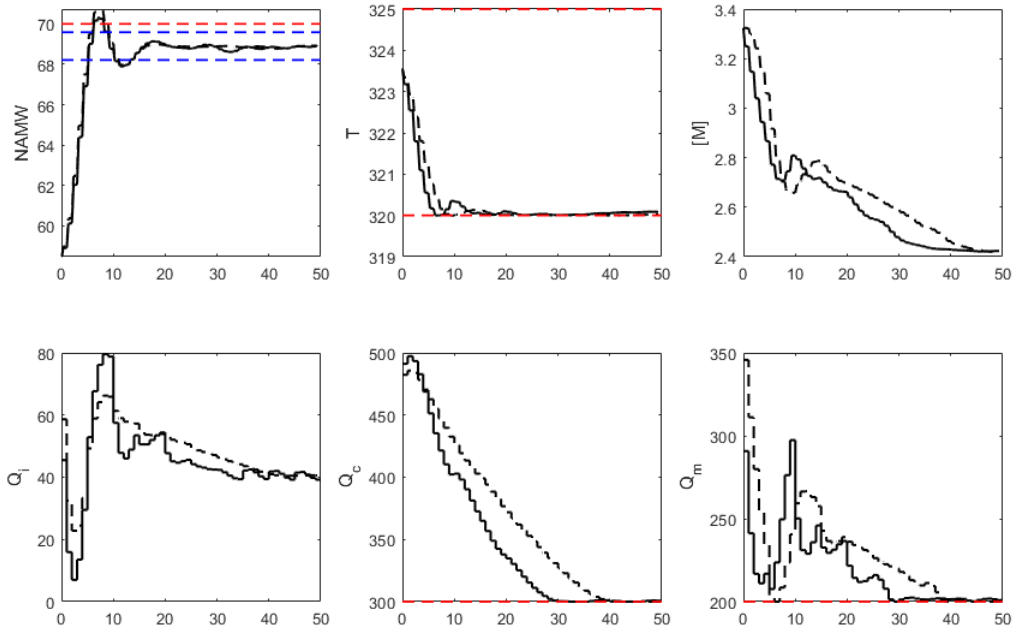

Figure 4.10: Plots of inputs, outputs for both parameters at nominal values scenario.

Table 4.10: Metrics of mean performance for 20 random parameter value scenarios of nominal and robust DRTO with two uncertain parameters.

Metric	**Nominal DRTO**	**Robust DRTO**	**Percent Change**
Profit	51.72	79.01	52.8
Constraint Violation	1.189	0.5580	-53.1
Solution Time (s)	2.080	97.60	4590

As in the previous case, the performance of the methods is also evaluated for 20 random plant uncertainty scenarios. As shown in Table 4.10, in this case, the robust DRTO still outperforms the nominal DRTO and by similar margins to the characteristic scenarios. The increase in profit is still about 50%. The decrease

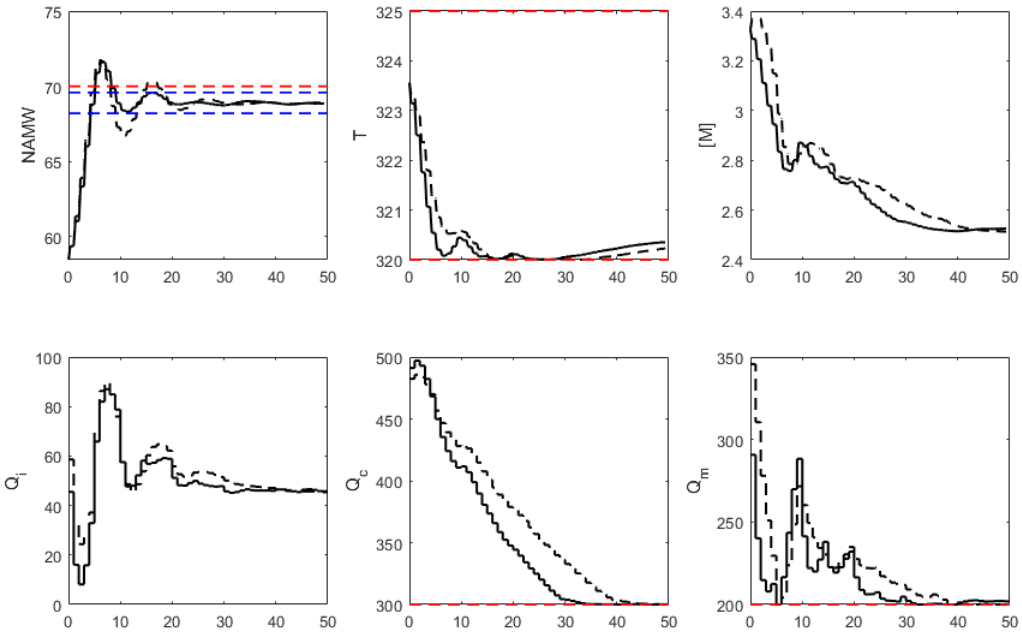

Figure 4.11: Plots of inputs, outputs for min-max scenario (pre-exponential rate constant at minimum; inlet concentration of monomer at maximum).

in NAMW constraint violation has actually improved further to about 50%. The results being very similar between the characteristic scenarios and the random scenarios shows that the robust DRTO is very capable of handling multiple sources of uncertainty, and effectively optimizing the plant even when its behavior does not exactly match its modeled scenarios.

The cost of this improvement in economic performance and constraint violation reduction is, of course, computation time. Due to modeling each pair of uncertain parameter values, the number of modeled scenarios is relatively large, at nine. Therefore, the computation time has increased by about a factor of 50, from approximately 2 s per DRTO iteration to approximately 100 s per DRTO iteration. This is one of the consequences of modeling multiple sources of uncertainty.

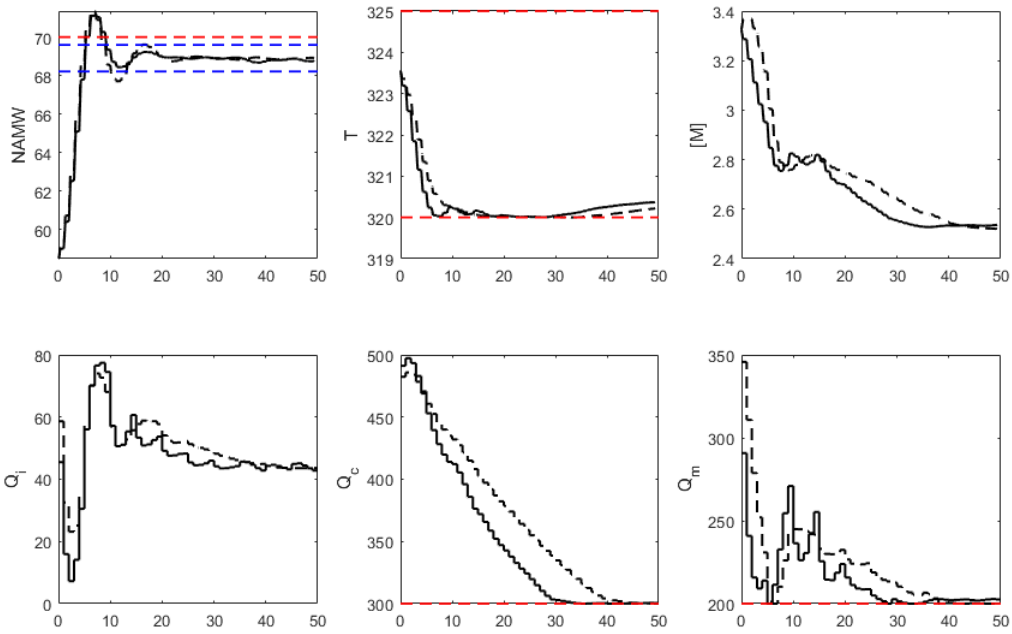

Figure 4.12: Plots of inputs, outputs for max-min scenario (pre-exponential rate constant at maximum; inlet concentration of monomer at minimum).

4.4 Conclusion

In this work, a multi-scenario closed-loop dynamic real-time optimization (CL-DRTO) formulation with input clipping approximation was presented as a means of handling uncertainty at the economic optimization layer while modeling the dynamic behavior of both the plant being optimized and the MPC controlling the plant. The method shows improvement over a single-scenario CL-DRTO in terms of economics and constraint violation in two simulated case studies. This improvement came at the cost of increased computation time.

A three-scenario CL-DRTO showed improvements over a single-scenario CL-DRTO in both a SISO and a MIMO case study. Following this, the performance

benefits of additional scenarios was investigated; it was found that additional scenarios modeled by the DRTO improves the overall economic performance of the method while also increasing the computation time. Specifically, the performance increased consistently until 9 scenarios were modeled, and then showed diminishing returns for more than 9 scenarios. Finally, the multi-scenario DRTO was shown to be effective at handling multiple sources of model uncertainty by modeling two uncertain parameters. In this case, the improvements over a single-scenario DRTO were more substantial, but at a significant increase to computation time. Overall, the method presented here has been shown to effectively handle the economic optimization of an uncertain plant while incorporating the behavior of the underlying deterministic MPC.

Chapter 5

Decomposition of Robust Closed-Loop Dynamic Real-Time Optimization

The formulations and results in this chapter will be submitted to:

[1] MacKinnon, L. and Swartz, C.L.E. Decomposition of Robust Closed-Loop Dynamic Real-Time Optimization. In preparation for journal publication submission.

5.1 Introduction

Economic optimization is often performed in a real-time setting to allow the optimizer to take into account feedback from the plant. However, this creates time constraints on the optimization process as an optimization solution must be provided to the plant and the relevant controllers at least before the next execution of the real-time

optimization (RTO). Considering this, the computationally expensive multi-scenario robust closed-loop dynamic RTO (CL-DRTO) presented in the previous chapter will have application limitations. Specifically, large DRTO problems which must be solved frequently may not be able to find a solution prior to the next execution. One strategy to mitigate this limitation is to decompose the large, monolithic robust CL-DRTO problem into its respective scenario subproblems. This chapter will propose such a decomposition method and test it against the previously presented monolithic method in terms of computation time and objective performance.

There are many possible strategies for decomposing multi-scenario dynamic optimization problems. Generally speaking, these strategies can be separated into primal and dual decomposition methods [6]. In dual methods, the variables which connect the subproblems are not fixed but rather their gradients are adjusted by the master problem and then the subproblems use those gradients to determine optimal values. Benders decomposition is a commonly used example of dual decomposition which has found use in a wide variety of applications [65]. In primal methods, the variables themselves are fixed by the master problem and the subproblems determine their gradients. Primal methods, therefore, maintain problem feasibility prior to convergence while dual methods only reach a feasible solution at convergence.

For robust MPC, decomposition methods are commonly used because handling uncertainty comes at a computational cost. De La Peña et al. [13] apply a dual decomposition strategy to reduce the computation time of a multi-stage linear min-max MPC in which the problem is separated by stage. They show the computation time of the decomposition is less than that of the single, large LP, with the gains increasing with the problem horizon. Lucia et al. [45] also use a

dual decomposition method, but for a multi-scenario nonlinear robust MPC where the problem is decomposed for each scenario, rather than by stage. A bundled decomposition approach, where scenarios are grouped together and solved, is also investigated. They find that both decomposition approaches improve computation time while finding similar solutions. Marti et al. [49] compare a number of decomposition strategies, including augmented Lagrangian, price-driven, sensitivity-based, and bundled decomposition, and their performance for robust multi-scenario nonlinear MPC. Krishnamoorthy et al. [35] instead use a primal decomposition method for multi-scenario robust MPC and argue that the feasibility guarantee at pre-convergence iterations is a substantial advantage over dual decomposition techniques. They employ a gradient descent method with a backtracking line search in the master problem. They find substantial improvement in computation time over a centralized solution method while maintaining comparable performance.

Decomposition in the field of RTO and DRTO is also used extensively, but generally not to separate robust scenarios, but rather to reduce the size of a large single-scenario problem. Zhang et al. [81] use decomposition to aid in solution of an RTO for train scheduling. They separate the coordinating problem into the respective problem types and find substantial computational advantages as a result. Luan et al. [43] also apply decomposition to RTO of train scheduling and, furthermore, compare the relative performance of separating the problem by location, by individual trains, and by time. They found that separating into individual train problems gave overall best results of the methods tested. Jalali et al. [24] solve a robust distributed RTO by decomposing the problem into individual microgrids and utilize an alternating direction method of multipliers. The method is

shown to effectively minimize operating costs of an energy distribution system under uncertain energy cost forecasts. Gunnerud et al. [21] decompose an oil well field RTO problem using the Dantzig-Wolfe strategy. They find that the decomposition method outperforms the global method for all but the smallest of their tested cases. This is because the global problem has significant difficulty finding good feasible solutions for the larger cases, specifically running out of usable memory before an optimal solution is found. The decomposition method solved in much less time and had no such issues with available memory.

This chapter proposes a primal decomposition method for robust multi-scenario CL-DRTO which separates the problem into scenario-based subproblems and coordinates the solution with a master problem algorithm which fixes the set-point trajectories for the scenario subproblems. A primal decomposition method was chosen because it maintains feasibility of solution at all iterations. In this setup, the scenario subproblems have parts of the set-point trajectories fixed by the master problem, which then uses the objective values and gradients from the subproblems to determine a new value of the set-point trajectories. The method is compared to the monolithic robust CL-DRTO method in terms of computation time and performance in two case studies.

This chapter is organized as follows. The formulation of the decomposition method is presented in Section 5.2. The monolithic, robust CL-DRTO method is first briefly outlined as it represents the base case method for this decomposition. Then the scenario subproblem formulation is explained, including the information provided by and to the master problem. Next, the master problem algorithm, how the algorithm is intended to function, and its expected limitations are explained in detail. Section 5.3

contains the case studies used to test the decomposition method. There are two case studies presented, one single-input-single-output (SISO) and one multi-input-multi-output (MIMO). The cases investigate the decomposition method solution time and performance compared to the monolithic method and how they change with problem size, complexity, nonlinearity, and objective function.

5.2 Formulation

The decomposition algorithm presented here uses a primal method as this decomposition type has previously been employed for robust multi-scenario MPC [35]. In this primal decomposition, the non-anticipativity constraints which tie the scenarios together are fixed within the scenario subproblems. Specifically, the variables relevant to the non-anticipativity constraints are chosen by the master problem and held constant within each scenario subproblem. For a robust MPC, these relevant variables are the inputs chosen by the MPC to be implemented in the plant; for a robust CL-DRTO, they are the set-points chosen by the DRTO to be used in the underlying MPC.

The master problem of this decomposition method uses objective values and gradients provided by the scenario subproblems at specific values of the set-point trajectory to determine a new set-point trajectory to implement in the subproblems. The scenario subproblems, in turn, solve the CL-DRTO problem for a single scenario realization and a fixed set-point trajectory to determine the objective value and relevant subgradients for that set-point trajectory.

5.2.1 Robust CL-DRTO

The monolithic robust CL-DRTO formulation shown here is the same as is presented in the previous chapter. It is used in this work as a base case to compare the decomposition formulation against to determine the potential efficacy of the new method. The main purpose of the robust CL-DRTO is to determine an optimal set-point trajectory to provide to an underlying MPC which will then, in turn, provide input moves to the plant itself. The robust CL-DRTO solves an optimization problem which takes into account the predicted response of the MPC via direct MPC simulation and how those MPC responses will affect the plant behavior under multiple possible realizations of an uncertain plant model. Essentially, it is a real-time optimization formulation which accounts for underlying MPC behavior, dynamic plant behavior, and plant model uncertainty. The formulation of the robust CL-DRTO is shown below.

$$\min_{y^{Ref},u^{Ref}} \quad \phi_{econ} = \sum_i \rho^i \phi^i(x^i, u^i, y^i) \tag{5.2.1a}$$

$$\text{s.t.} \quad x^i_{j+1} = \; f^i(x^i_j, u^i_j), \quad j = 0, \ldots, N-1 \tag{5.2.1b}$$

$$y^i_j = \; h^i(x^i_j), \quad j = 1, \ldots, N \tag{5.2.1c}$$

$$0 \leq \; g(x^i_j, y^i_j), \quad j = 1, \ldots, N \tag{5.2.1d}$$

$$0 = \; h^{Ref}(y^{Ref}, u^{Ref}, \tilde{y}^{SP}, \tilde{u}^{SP}) \tag{5.2.1e}$$

$$0 \leq \; g^{Ref}(y^{Ref}_{j+1}, u^{Ref}_j), \quad j = 0, \ldots, N-1 \tag{5.2.1f}$$

$$\begin{aligned} u^i_j = \arg\min_{\tilde{u}^i_{j,k}} \quad & \tilde{\phi}^i_j = \sum_{k=1}^{p} (\tilde{y}^i_{j,k} - \tilde{y}^{\text{sp}}_{j,k})^T Q (\tilde{y}^i_{j,k} - \tilde{y}^{\text{sp}}_{j,k}) \\ & + \sum_{k=0}^{m-1} (\Delta \tilde{u}^i_{j,k})^T R (\Delta \tilde{u}^i_{j,k}) \\ & + \sum_{k=0}^{m-1} (\tilde{u}^i_{j,k} - \tilde{u}^{\text{sp}}_{j,k})^T S (\tilde{u}^i_{j,k} - u^{\text{sp}}_{j,k}), \quad j = 0, \ldots, N-1 \end{aligned} \tag{5.2.1g}$$

$$\text{s.t.} \quad \tilde{x}^i_{j,k+1} = A\tilde{x}^i_{j,k} + B\tilde{u}^i_{j,k}, \quad k = 0, \ldots, m-1 \tag{5.2.1h}$$

$$\tilde{x}^i_{j,k+1} = A\tilde{x}^i_{j,k} + B\tilde{u}^i_{j,m-1}, \quad k = m, \ldots, p-1 \tag{5.2.1i}$$

$$\tilde{y}^i_{j,k} = C\tilde{x}^i_{j,k} + \tilde{d}^i_{j,k}, \quad k = 1, \ldots, p \tag{5.2.1j}$$

$$\Delta \tilde{u}^i_{j,k} = \tilde{u}^i_{j,k} - \tilde{u}^i_{j,k-1}, \quad k = 0, \ldots, m-1 \tag{5.2.1k}$$

$$u_{\min} \leq \tilde{u}^i_{j,k} \leq u_{\max}, \quad k = 0, \ldots, m-1 \tag{5.2.1l}$$

In the above, ϕ_{econ} is the economic objective function of the primary DRTO; ϕ^i_{econ} is the economic objective function value for scenario i; ρ^i is the objective weighting for scenario i; u, x, y are the system inputs, states, and outputs, respectively; the tilde ($\tilde{}$) denotes variables and functions associated with the MPC subproblems;

the Ref superscript denotes reference trajectory variables; the SP superscript denotes set-point trajectories for the MPC; f^i is the set of dynamic equations for the plant model; h^i is the algebraic plant model relating states to outputs; g is the set of inequality process constraints; h^{Ref} is the functional relationship between the reference trajectory and the set-point trajectories for each MPC subproblem; g^{Ref} is the set of bounds on the reference points; i denotes the uncertain scenario; j denotes the DRTO time point; and u^i, x^i, y^i are composite vectors of the inputs, states, and outputs, respectively, for plant scenario i over the DRTO prediction horizon; $\tilde{\phi}^i_j$ is the MPC objective function corresponding to step j of the DRTO prediction horizon; m and p are the MPC control and prediction horizons, respectively; Q, R, S are the output deviation weighting matrix, input move penalty matrix, and input deviation weighting matrix, respectively; A, B, C are the linear, or linearized, state-space matrices for the plant system dynamic model; $u_{\min}$, $u_{\max}$ are the input bounds; $d^i_{j,k}$ is the disturbance estimate; and j, k are the time steps for the DRTO and MPC, respectively.

In particular, the DRTO generally uses an economic objective function which is the expected value across all modelled scenarios, though any appropriate objective function may be used, including a target tracking objective. Additionally, it can use a linear or nonlinear plant model at the DRTO level; previous work has focused on a nonlinear model for increased fidelity, but a linear model for improved computation time may also be used. Output, process, and set-point constraints are included at the DRTO level while input constraints are handled in the MPC. At the MPC level, it is assumed that a linear plant model is used. The MPC is also assumed to have a quadratic target tracking objective function. These assumed properties of the MPC

maintain the embedded MPC problems as convex quadratic programs, significantly simplifying the solution of the overall problem.

The formulation above is a bi-level optimization problem. The DRTO determines optimal set-points based on the inputs provided by the MPC, and the MPC determines optimal input moves based on the set-points provided by the DRTO. A simultaneous solution approach is used here to handle this problem, as has been done previously [27, 2]. This approach takes advantage of the MPC's being convex quadratic problems by reformulating them as their respective first-order KKT conditions which can then be directly included in the DRTO problem as algebraic constraints. This creates a single-level mathematical program with complementarity constraints (MPCC) which can be handled with the use of an exact penalty approach [66].

Furthermore, if the input constraints in the embedded MPC problems are relaxed, the MPC problems become unconstrained convex QPs, which can be solved analytically. Doing so has been shown to significantly improve computation time and so this is done here. This has been previously accompanied by an input clipping approach at the DRTO level [29, 41].

However, the use of input clipping still involves complementarity constraints. For this work, the inputs are instead not constrained within the CL-DRTO at either the primary DRTO or MPC level. This was done to simplify the problem to make it somewhat easier for the decomposition method to solve. While it is unlikely that an MPC controlling a real plant would be implemented without input constraints, simulating the MPC as being unconstrained within the DRTO is still a reasonable approximation of a constrained MPC. For the case studies in this

work, an underlying constrained MPC is executed as part of the closed-loop plant simulation. Furthermore, if the inputs do not saturate, then the unconstrained MPC is an exact representation of the constrained MPC. This was therefore considered to be a reasonable simplification for an initial test of the decomposition method. The removal of the input constraints was done for both the monolithic and decomposition solution methods for fair comparison.

5.2.2 Primal Decomposition

A primal decomposition method was chosen here primarily because primal decomposition does not relax the non-anticipativity constraints of the multi-scenario formulation [35]. It instead uses the Lagrange multipliers of the non-anticipativity constraints to update the variables relevant to those constraints (the set-point trajectories, in this case). This contrasts with a dual decomposition method which does relax the non-anticipativity constraints, solving the overall problem by updating the Lagrange multipliers of those constraints, rather than the relevant variables directly [45].

This property of primal decomposition ensures a feasible result at all iterations, which then may be passed on to the underlying MPC regardless of convergence. This is particularly useful in a real-time optimization setting where a solution to the DRTO may be required in a relatively short time. If, in this setting, the problem size proves too substantial and the decomposition method does not converge in time, a primal decomposition method can still provide the most recent interim solution which is guaranteed to be a feasible solution to the problem. However, a dual decomposition, prior to convergence, will not have satisfied the non-anticipativity constraints and so

will not have a feasible solution to report. For a CL-DRTO, this means that there will be multiple set-point trajectories, one for each scenario, still present within the master problem as potential solutions. Therefore, the dual decomposition method will not have a single set-point trajectory which can be provided to the underlying MPC if it has not yet converged. If a situation exists where solution time of the DRTO problem has a fixed upper bound, a primal decomposition approach may be a safer strategy as it will always provide a feasible solution even if it does not converge to an optimal solution.

The above property of primal decomposition is illustrated in the case studies in the following section by solving the optimization problem to a wide range of tolerances. This range of tolerances is similar to terminating the problem at a wide range of solution times. Tolerances were used instead for easy comparison between problems of varying sizes, where different solution time limits for different problem sizes would be required to show similar accuracy. These ranges of tolerances clearly show how the dual decomposition method provides feasible solutions for relatively loose tolerances.

5.2.3 Scenario Subproblems

The subproblems of the primal decomposition problem can be expressed as a single scenario realization of the robust CL-DRTO problem. This formulation is therefore nearly the same as the single-scenario CL-DRTO formulation originally developed by Jamaludin and Swartz [27]. A new variable is introduced, θ, which is the set of fixed set-point trajectories of the required variables provided by the master problem. The length of this fixed set-point trajectory must be sufficient to provide set-points to the underlying MPC executions until the next DRTO execution. Therefore, this

fixed set-point trajectory length must be at least the length of the MPC prediction horizon plus the DRTO sample time. This is because the set-point trajectory must be provided to all the MPC executions from the time point the DRTO is executed until the next time point where the DRTO is executed. The scenario subproblems of the primal decomposition can be expressed as follows:

$$\min_{y^{Ref},u^{Ref}} \quad \phi^s = \rho^s \phi^s(x^s, u^s, y^s) \tag{5.2.2a}$$

$$\text{s.t.} \quad x^s_{j+1} = f^s(x^s_j, u^s_j), \quad j = 0, \ldots, N-1 \tag{5.2.2b}$$

$$y^s_j = h^s(x^s_j), \quad j = 1, \ldots, N \tag{5.2.2c}$$

$$0 \leq g(x^s_j, y^s_j), \quad j = 1, \ldots, N \tag{5.2.2d}$$

$$0 = h^{Ref}(y^{Ref}, u^{Ref}, \tilde{y}^{SP}, \tilde{u}^{SP}) \tag{5.2.2e}$$

$$0 \leq g^{Ref}(y^{Ref}_{j+1}, u^{Ref}_j), \quad j = 0, \ldots, N-1 \tag{5.2.2f}$$

$$y^{Ref}_j = \theta_{y,j}, \quad j = 1, \ldots, p + \Delta t_{DRTO} \tag{5.2.2g}$$

$$u^{Ref}_j = \theta_{u,j}, \quad j = 0, \ldots, p + \Delta t_{DRTO} - 1 \tag{5.2.2h}$$

$$\begin{aligned} u^s_j = \arg\min_{\tilde{u}^s_{j,k}} \quad & \tilde{\phi}^s_j = \sum_{k=1}^{p} (\tilde{y}^s_{j,k} - \tilde{y}^{\text{sp}}_{j,k})^T Q (\tilde{y}^s_{j,k} - \tilde{y}^{\text{sp}}_{j,k}) \\ & + \sum_{k=0}^{m-1} (\Delta \tilde{u}^s_{j,k})^T R (\Delta \tilde{u}^s_{j,k}) \\ & + \sum_{k=0}^{m-1} (\tilde{u}^s_{j,k} - \tilde{u}^{\text{sp}}_{j,k})^T S (\tilde{u}^s_{j,k} - u^{\text{sp}}_{j,k}), \quad j = 0, \ldots, N-1 \end{aligned} \tag{5.2.2i}$$

$$\text{s.t.} \quad \tilde{x}^s_{j,k+1} = A\tilde{x}^s_{j,k} + B\tilde{u}^s_{j,k}, \quad k = 0, \ldots, m-1 \tag{5.2.2j}$$

$$\tilde{x}^s_{j,k+1} = A\tilde{x}^s_{j,k} + B\tilde{u}^s_{j,m-1}, \quad k = m, \ldots, p-1 \tag{5.2.2k}$$

$$\tilde{y}^s_{j,k} = C\tilde{x}^s_{j,k} + \tilde{d}^s_{j,k}, \quad k = 1, \ldots, p \tag{5.2.2l}$$

$$\Delta \tilde{u}^s_{j,k} = \tilde{u}^s_{j,k} - \tilde{u}^s_{j,k-1}, \quad k = 0, \ldots, m-1 \tag{5.2.2m}$$

$$u_{\min} \leq \tilde{u}^s_{j,k} \leq u_{\max}, \quad k = 0, \ldots, m-1 \tag{5.2.2n}$$

In the above, the scenario index i is replace with a fixed scenario, s, as the scenario subproblems are solved only for one scenario at a time. Note the scenario

specific objective value of the DRTO scenario subproblem. The set-point reference trajectories, y_j^{Ref} and u_j^{Ref}, are fixed to the values of θ_y, the output subset of the fixed set-point trajectories, and θ_u, the input subset of the fixed set-point trajectories, respectively, provided by the master problem, for the required length of the MPC horizon, p, plus the DRTO sample time, Δt_{DRTO}.

It should be noted that the degrees of freedom of the CL-DRTO are the set-points to be provided to the MPC. Therefore, fixing some length of the set-point trajectory reduces the degrees of freedom of the optimization problem. If the DRTO prediction horizon is less than or equal to the MPC horizon plus the DRTO frequency then the scenario subproblems of the decomposition method are deterministic and have only one feasible solution.

Overall, the scenario subproblems primarily serve to determine the objective value and gradients for a given set-point trajectory for each scenario. This information will inform the master problem as to how to adjust the set-point trajectory for the next iteration.

5.2.4 Master problem algorithm

The purpose of the master problem is to attempt to optimize the expected value of the overall optimization problem across all modelled scenarios by varying the fixed portion of the set-point trajectory, θ. This can be expressed as:

$$\min_{\theta} \quad \phi = \sum_{i} \phi^i \tag{5.2.3}$$

The scenario subproblems can provide objective values and relevant gradients for no additional computational cost as these are necessary values for the solution of

the respective optimization subproblems. Therefore, the master problem can simply extract this information from the subproblems as necessary. Given this information available from the subproblems, there are a number of strategies which have been employed to attempt to solve problems of this form. The general concerns of these methods is balancing the computational complexity of the master problem with the required number of iterations. More complex master problems often require fewer iterations but each iteration requires more computation time. A relatively simple method would be a gradient descent with backtracking line search [1], though such a method may require many iterations to converge. Krishnamoorthy et al. [35] use this approach for a robust MPC decomposition because each iteration is relatively fast to solve and the feasibility test for the backtracking step is not an optimization problem, but an algebraic computation. For a robust CL-DRTO decomposition, the subproblems are CL-DRTO problems and so are not trivial to solve; furthermore, one must be solved for each scenario that is modelled. A backtracking determination would also require additional optimization problems instead of an algebraic computation. Therefore, it is likely to be beneficial to use a somewhat more complex master problem which reduces the overall number of iterations.

In light of this, the method used here is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [7, 16, 20, 71] algorithm with a quadratic Lagrange polynomial line search. The BFGS algorithm uses two evaluations to estimate the Hessian of the function in the descent direction. It then adjusts the descent direction based on the estimated second order function information. This can be a substantial improvement over gradient methods which only use first order function information. Additionally, a

third point is here also computed within each iteration in order to produce a three point quadratic approximation of the objective function along the descent direction. Because two points are already required for the BFGS method, the additional third point only adds approximately 50% to the computation time while substantially improving the line search, and, therefore, reducing the required number of iterations. This approach was found to strike a reasonable balance of master problem complexity and required iteration count. When used in the case studies in the following section, generally fewer than 10 iterations were required for convergence. The full algorithm can be seen in detail in Algorithm 1.

In Algorithm 1, ε_1 is the convergence tolerance, ε_2 is the failure tolerance, α is the initial step size, θ is the fixed portion of the set-point trajectories to be provided to the scenario subproblems, $\Delta\phi^{rel}$ is the relative change in objective value, $\alpha^\star$ is the computed step size, c is a tuning parameter determining how quickly $\alpha^\star$ decreases in the event of a non-improving step, B is the approximated inverse Hessian used for the BFGS method, S is the total number of modelled scenarios, $\nabla_\theta\phi^i$ are the gradients of the objective with respect to the fixed set-point trajectories for scenario i.

For determining the initial guess for θ, here a single scenario CL-DRTO problem is solved using the nominal values of the uncertain parameters. This provides a good initial guess, as it represents the optimal solution for the central scenario of the uncertain region. It also does not add significantly to the computation time, as the decomposition method presented must solve 3 sets of S single scenario CL-DRTO problems for every iteration.

With sufficiently tight tolerance, this algorithm should converge to a local solution of the optimization problem. It should be noted that the algorithm is a local solution

Algorithm 1 Master Problem Algorithm

Define $c < 1$, ε_1, ε_2, α
Initialize θ, $\Delta\phi^{rel} > \varepsilon_1$, $\phi_{prev} >> 1$, $\theta_{prev} \leftarrow \theta$, $\alpha^\star \leftarrow \alpha$, $B \leftarrow I$
while $\Delta\phi^{rel} > \varepsilon_1$ **do**
  for $i = 1$ to S **do**
    Solve subproblem of scenario i with fixed θ
    Evaluate scenario objective value, ϕ^i, and gradients w.r.t. θ, $\nabla_\theta \phi^i$
  end for
  Assign $\phi \leftarrow \sum_i \phi^i$ and $\sigma \leftarrow \sum_i \nabla_\theta \phi^i$
  if $\phi > \phi_{prev}$ **then**
    Solution is not improving, attempt smaller step from previous point:
    $\alpha^\star \leftarrow c * \alpha^\star$
    $\theta \leftarrow \theta_{prev} + \alpha^\star * B * \sigma_{prev}$
    if $\alpha^\star \leq \varepsilon_2$ **then**
      BREAK, unable to find improving step
    end if
  end if
  Vary θ and test two more points along direction $B * \sigma$:
  $\theta \leftarrow \theta + \alpha * B * \sigma$
  Repeat 4-8, assigning $\phi_2 \leftarrow \sum_i \phi^i$ and $\sigma_2 \leftarrow \sum_i \nabla_\theta \phi^i$
  $\theta \leftarrow \theta + \alpha * B * \sigma$
  Repeat 4-8, assigning $\phi_3 \leftarrow \sum_i \phi^i$
  Return to original point:
  $\theta \leftarrow \theta - 2 * \alpha * B * \sigma$
  Record change in θ and gradients over distance α for computation of B:
  $\Delta\theta \leftarrow \alpha * B * \sigma$
  $\Delta\sigma \leftarrow \sigma_2 - \sigma$
  Compute optimal step size in direction $B * \sigma$ as if function were quadratic:
  $\alpha^\star \leftarrow 0.5 * \alpha * \dfrac{-3\phi + 4\phi_2 - \phi_3}{-\phi + 2\phi_2 - \phi_3}$
  Update relative change in objective, $\Delta\phi^{rel}$:
  $\Delta\phi^{rel} \leftarrow ||\phi - \phi_{prev}||/||\phi_{prev}||$
  Save information for next iteration:
  $\theta_{prev} \leftarrow \theta$, $\phi_{prev} \leftarrow \phi$, $\sigma_{prev} \leftarrow \sigma$
  Move a distance $\alpha^\star$ in direction $B * \sigma$:
  $\theta \leftarrow \theta + \alpha^\star * B * \sigma$
  Compute new approximation for inverse Hessian, B:
  $B \leftarrow B + (1 + \dfrac{\Delta\sigma^T B \Delta\sigma}{\Delta\theta^T \Delta\sigma}) \dfrac{\Delta\theta\Delta\theta^T}{\Delta\theta^T\Delta\sigma} - \dfrac{B\Delta\sigma\Delta\theta^T + \Delta\theta\Delta\sigma B}{\Delta\theta^T\Delta\sigma}$
end while

method, with no global optimization characteristics. Local NLP solvers are often used in DRTO applications and so a local solution method is likely sufficient for most applications. In light of this, a good initial guess is important for the algorithm to converge to the desired solution as well as to reduce the computation time.

The convergence criterion shown here is the relative change in objective function value. This choice was made out of practicality as it was found to produce more consistent results and generally better performance than using the change in set-point trajectory as the convergence criteria. In some test cases in this chapter, a convergence criterion was used where the relative change in objective must be less than the tolerance for two consecutive iterations. This slightly improved the solution, but significantly increased the solution time.

The algorithm, as presented, is an unconstrained NLP solution method and so does not handle the presence of hard constraints. In this chapter, this limitation is mitigated by the use of soft constraints with large penalties. This produces large gradients when the solution of a scenario subproblem violates these constraints, thereby driving the master problem back into the feasible region. The algorithm can be adjusted to be able to handle hard constraints if the application requires it. This could be done, for example, by using the BFGS-B variant of the method [9].

5.3 Case Studies

In order for the presented decomposition algorithm to be potentially useful, it should display a few key characteristics. The most important is whether the algorithm can converge to a local optimum. Additionally, there should be conditions under which it is able to find a local optimum with lower computation time than the

monolithic problem. An algorithm which exhibits these properties has potential to be a more appropriate choice of optimization method than a monolithic problem in some applications.

Secondly, the conditions under which this algorithm outperforms the monolithic problem should be determined. Therefore, the decomposition method is here evaluated on a series of case studies designed to investigate the effects of problem size, complexity, and nonlinearity on the performance of the algorithm. For the decomposition method to be potentially useful for future applications, there should be some combination of these properties under which the decomposition method outperforms the monolithic method.

For all the cases presented here, the decomposition algorithm and the monolithic problem are solved for varying numbers of modelled scenarios. This aspect of the problem size is especially important because of the difference in scaling. In the monolithic problem, increasing the number of scenarios directly increases the problem size. By contrast, in the decomposition method, the subproblems remain the same size but the number of subproblems which must be solved increases. Therefore, determining how this difference affects the relative performance of the decomposition method and the monolithic solution is an important aspect of this investigation.

The case studies all evaluate the method in a closed-loop simulation where the DRTO is executed five times over the length of the simulation. Furthermore, this simulation is carried out for three different plant realizations corresponding to the minimum, nominal, and maximum uncertain parameter values, here also called the characteristic scenarios. Therefore, each solution time data point in the below case studies is an average of the solution time of 15 DRTO executions. Each expected

objective value data point is an average of the objective evaluated using the simulation data from the three characteristic scenarios. In the simulation, an MPC is also applied to the plant to determine inputs from the set-point trajectories provided by the DRTO. This MPC still includes input constraints, even though the MPC problems embedded in the DRTO does not. The monolithic method and the decomposition scenario subproblems use CONOPT to solve the optimization problem. All solution times reported are using an Intel Core i7 8700 (3.2 GHz, 6 cores, 12 processors).

5.3.1 Case Study 1 – SISO

The first case study used to test the algorithm is a single-input-single-output (SISO) bioreactor problem with two states, where the kinetics are governed by the Michaelis-Menten equation [17]. The reaction equations may be expressed as follows:

$$\frac{dC}{dt} = D \cdot (C_{in} - C) - \frac{V_m \cdot C}{K_s + C} \tag{5.3.1}$$

$$\frac{dP}{dt} = \frac{V_m \cdot C}{K_s + C} - D \tag{5.3.2}$$

where the differential states are the concentration of reactant, C, and concentration of product, P; the inlet concentration, C_{in}, is the input; and the concentration of product, P, is the output. D, V_m, K_s are the ratio of flow rate to reactor volume, the maximum reaction rate, and the reaction constant, respectively, and are parameters. The uncertain parameter was chosen to be the maximum reaction rate, V_m, and is assumed to have a uniform probability distribution within the expected range.

This case study is the same as the first case study used in the previous chapter

and allows for a small scale example on which to test the presented decomposition algorithm. For this case, a target-tracking objective is used for simplicity. The number of scenarios modelled by the decomposition and monolithic robust CL-DRTO methods varies within each subcase. The methods are tested on three simulated plant realizations which correspond to the minimum, nominal, and maximum values of the uncertain parameter. The DRTO and MPC parameter values used for this case are shown in Table 5.1.

Table 5.1: DRTO and MPC parameters for Case Study 1.

Parameter	**Description**	**Value**
N	DRTO horizon	20
Δt_{DRTO}	DRTO sample time	4 hr
Δt_{MPC}	MPC sample time	1 hr
p	MPC prediction horizon	8
m	MPC control horizon	2
Q	Output tracking weight	1
R	Input move suppression weight	1
S	Input tracking weight	0
T_s	Time Step	1 hr

Three subcases are presented here which test different key aspects of the algorithm. First, a linear plant model single reactor subcase is used to determine the ability of the algorithm to converge to an optimal solution in a convex problem. Next, a subcase with multiple parallel reactors is used to investigate the effect of increased problem size on the performance of the algorithm without impacting the complexity or convexity of the problem. Finally, a nonlinear plant model is used to determine whether the algorithm is effective in nonconvex problems.

Subcase 1a – Single reactor, linear model, target tracking

The first subcase used to test the algorithm is designed to, first, determine the ability of the algorithm to find a local optimum and, second, to investigate the effect of number of scenarios on the computation time of the algorithm. This subcase uses a linear plant model and a quadratic target tracking objective, creating a convex optimization problem. Based on this, the decomposition method and the monolithic optimization method should reach the same solution as there is only one local optimum.

For this subcase, the uncertain parameter, V_m, is modelled and tested with an uncertainty of $\pm 20\%$. The output, P, is given a target of 1 g/L to transition to from a starting point of 0.72. A soft upper bound constraint of $P \leq 1.05$ is included in this subcase with a penalty weight of 100. There are no set-point constraints for this subcase.

Figure 5.1 shows the computation time of the decomposition and monolithic robust CL-DRTO methods for 1 to 101 modelled scenarios. Figure 5.2 shows a subset of this data, specifically for 1 to 21 modelled scenarios. It should be noted that, in these and subsequent figures, a tolerance of "10^{-6} (consecutive)" means that the convergence criterion with a tolerance of 10^{-6} had to be met for two consecutive iterations of the master problem.

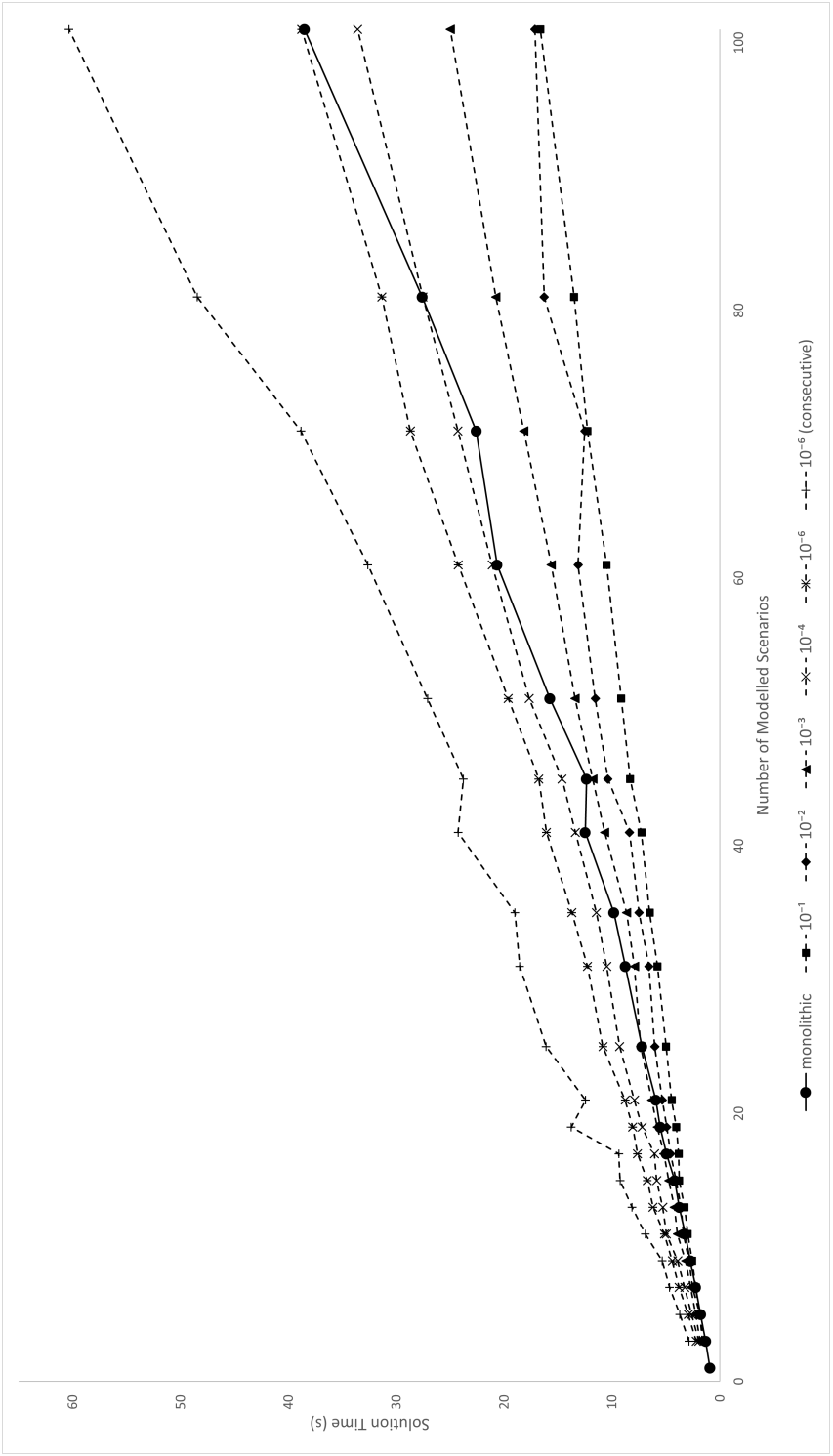

Figure 5.1: Solution time (s) of monolithic optimization problem and decomposition method using various tolerances from 1 to 101 modelled scenarios, SISO linear case.

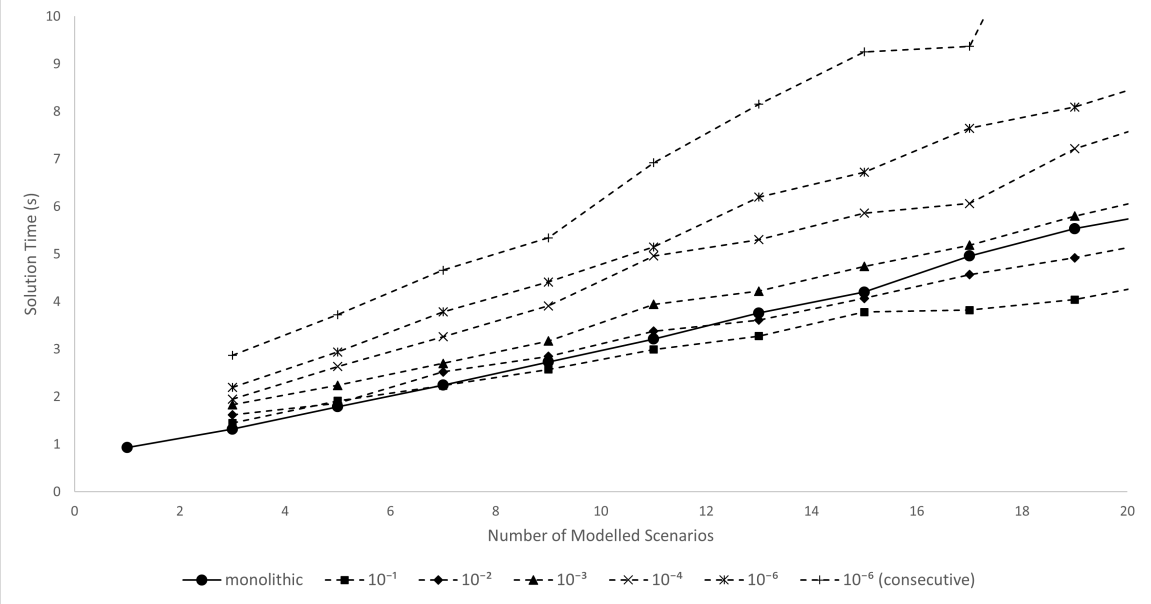

Figure 5.2: Solution time (s) of monolithic optimization problem and decomposition method using various tolerances from 1 to 21 modelled scenarios, SISO linear case.

Based on the solution time results shown in Figures 5.1 and 5.2, the decomposition method appears to scale linearly with the number of modelled scenarios while the monolithic solution method appears to scale nonlinearly (roughly proportional to the square of the number of scenarios). This is an important result as it implies that, regardless of the details of the problem, there should always be some sufficiently large number of scenarios where the decomposition method converges more quickly than the monolithic.

The investigation involves looking at the performance of the decomposition method at various convergence tolerances. This is to show the tradeoff between the computation time and the quality of solution for the decomposition method. For relatively loose tolerances, the algorithm terminates quite quickly but the solution obtained is suboptimal. Conversely, very tight tolerances provide a solution much

closer to the local optimum but take longer to converge. In accordance with this, the smallest number of scenarios modelled in the optimization problem for which the decomposition method solves more quickly than the monolithic depends on the tolerance provided to the decomposition algorithm. A table of these crossover points is shown in Table 5.2, along with their average discrepancies from the monolithic solution. The change from monolithic is relative to the nominal solution as this was the initial guess for the decomposition problem. So, for example, if the nominal solution had a value of 11, the monolithic solution 10, and the decomposition 10.1, then the change from monolithic would be reported as being 10%. For some tolerances, the observed mean expected objective value is less than for the monolithic. This is a result of the slight discrepancy between the model prediction of the DRTO and the plant simulation it is tested on. The objective value computed by the decomposition DRTO itself in these situations is still (slightly) larger than that computed by the monolithic DRTO. This can be seen in Table 5.3, where the objective computed by the first execution of the DRTO is shown for various tolerances, averaged across all modelled scenarios from 1 to 101. Figure 5.3 shows the observed expected objective function value for the monolithic method and decomposition method at various tolerances for 1 to 101 modelled scenarios.

From Figure 5.3 and Table 5.2 and 5.3, it can be seen that it requires a tolerance of 10^{-6} (consecutive) for the decomposition to achieve the same result as the monolithic. It is important that there exists a tolerance for which this is true, as it shows that the decomposition method is converging to the same solution as the monolithic (the only local optimum). However, the computation time for this convergence tolerance is large, significantly larger than the monolithic for all numbers of modelled

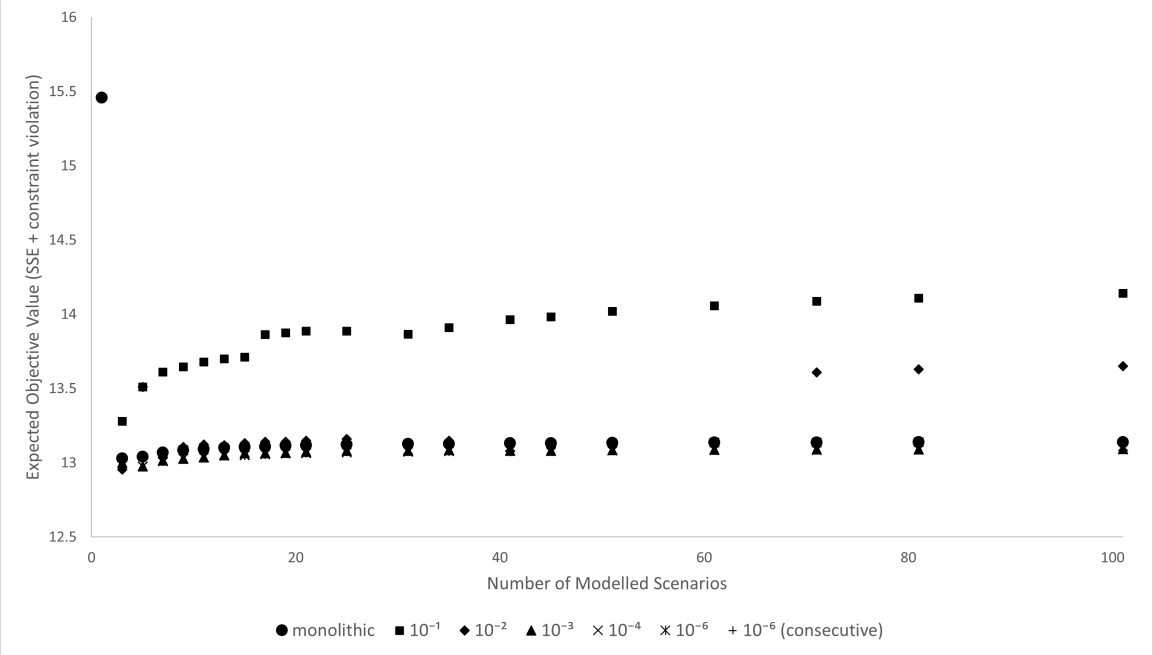

Figure 5.3: Expected objective from three characteristic scenario simulations of monolithic optimization problem and decomposition method using various tolerances by number of modelled scenarios.

scenarios tested. For looser tolerances, the solution tends to deviate from the optimal solution. At the largest tested tolerance of 10^{-1}, the solution found is about 31% worse in simulation than the monolithic solution, and 4.7% worse in terms of computed objective at the first execution. For 10^{-2}, it is about 4.1% worse in simulation and 2.2% worse of a first computed objective. For the tolerances of 10^{-3}, 10^{-4}, and 10^{-6} the relative measured performance is slightly better (approximately 1-2%). However, the computed objective is worse by a margin of 0.35%, 0.15%, and 0.068% for tolerances of 10^{-3}, 10^{-4}, and 10^{-6}, respectively. Overall, it should be considered that these sets of results are very similar and that the decomposition method does not show significant improvements in this subcase with tolerances tighter than 10^{-3}.

For this simple, small, convex problem, the number of scenarios required for the

Table 5.2: Minimum scenarios for decomposition method to solve in less time than monolithic problem (crossover point) and expected performance relative to monolithic for various tolerances, SISO linear case.

Decomposition Tolerance	**Crossover Point**	**Mean expected objective value**	**% change from monolithic**
Nominal	N/A	15.46	+100
10^{-1}	7	13.84	+31.1
10^{-2}	13	13.21	+4.14
10^{-3}	31	13.06	-2.06
10^{-4}	81	13.05	-2.23
10^{-6}	101	13.07	-1.66
10^{-6} (consecutive)	N/A (>101)	13.11	0.00
Monolithic	N/A	13.11	0

decomposition method to find a solution close to the optimum and solve in less time than the monolithic is quite large, 31 modelled scenarios with a tolerance of 10^{-3} and many more for tighter tolerances. It is unlikely that this many scenarios would be necessary. Therefore, the monolithic method is probably more appropriate for such a small, convex problem such as this. However, what this first case shows is that the solution tends to improve and the solution time increases as the tolerance decreases. It also showed that as more scenarios are modelled the decomposition method gets steadily more competitive with the monolithic method. Additionally, regardless of tolerance used, the solution provided was always better than the nominal DRTO solution (which was used as the initial guess).

Subcase 1b – Multiple parallel reactors

The next subcase investigated involves increasing the optimization problem size beyond the number of scenarios, while keeping the problem SISO. This is

Table 5.3: Computed objective value by first execution of DRTO, averaged across all modelled scenarios from 1 to 101, relative to monolithic for various tolerances, SISO linear case.

Decomposition Tolerance	**Mean computed objective value**	**% change from monolithic**
Nominal	7.0222	+100
10^{-1}	5.8551	+4.68
10^{-2}	5.8263	+2.16
10^{-3}	5.8047	+0.350
10^{-4}	5.8022	+0.149
10^{-6}	5.8013	+0.068
10^{-6} (consecutive)	5.8004	0.00
Monolithic	5.8004	0

accomplished by modelling multiple identical reactors in parallel, with one inlet stream splitting evenly into every reactor and one outlet stream being an even mix from all of the reactors. By assuming that the reactors behave identically, this subcase can test the effect of problem size (in the form of number of states and model equations) on computation time for the decomposition method without increasing the number of set-point trajectories which must be provided to the MPC. Other than increasing the number of reactors, the other details of this subcase are identical to the previous one.

As is expected, the solutions found in this subcase are identical to those found in the previous subcase. Only computation time differs by changing the number of parallel reactors. Figures 5.4, 5.5, and 5.6 show the solution time for the monolithic method and decomposition method with various tolerances by number of parallel reactors for 3, 11, and 21 modelled scenarios, respectively. The tolerance of 10^{-6} (consecutive) was not included here as it was much slower than when using the other

tolerances and did not substantially improve on those cases. The tolerance of 10^{-3} was also not included as its performance was very similar to both that of 10^{-2} and 10^{-4}. The nominal CL-DRTO results are also included here (in the previous subcase the nominal result is simply the 1 scenario data point) as a reference for the solution times of these problems by a non-robust CL-DRTO.

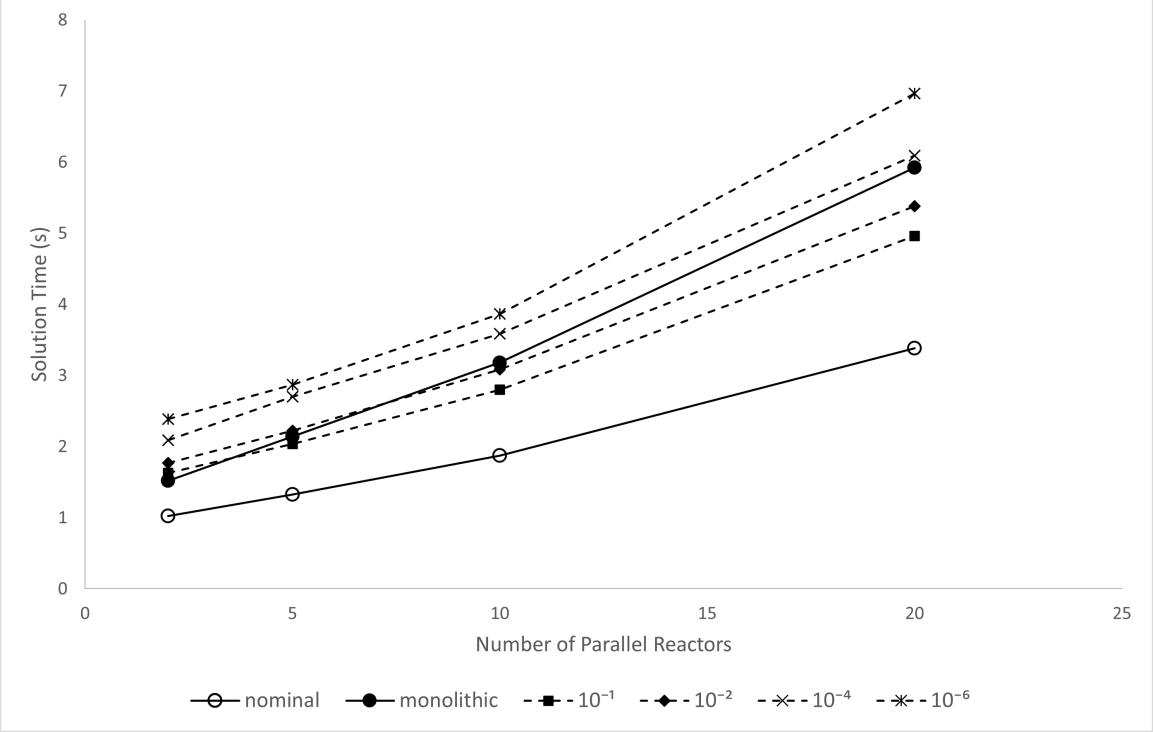

Figure 5.4: Solution time (s) of monolithic optimization problem and decomposition method at various tolerances by number of parallel reactors, for 3 modelled scenarios.

Unlike the scaling with number of scenarios, the number of reactors (acting as a proxy for number of states and model equations) affects both the monolithic and decomposition methods nonlinearly. However, the monolithic problem computation time increases more quickly with number of reactors than the decomposition method. This is likely because solution time of optimization problems generally scales

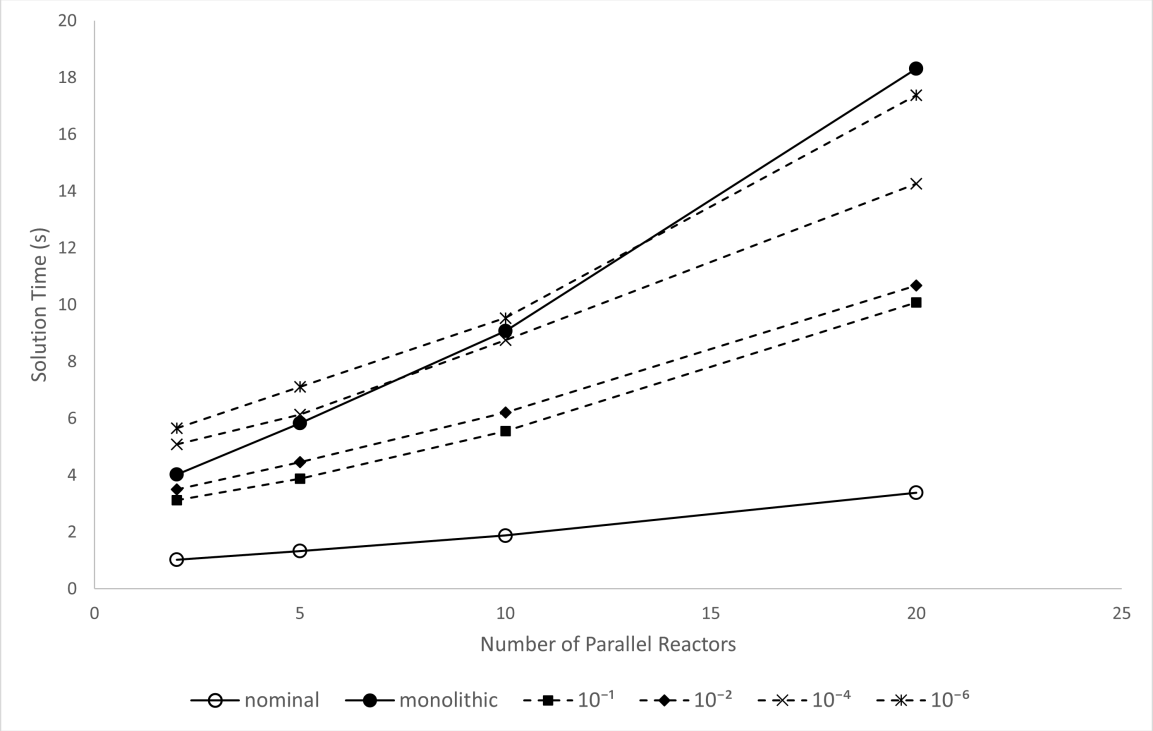

Figure 5.5: Solution time (s) of monolithic optimization problem and decomposition method at various tolerances by number of parallel reactors, for 11 modelled scenarios.

nonlinearly with problem size and so the larger initial problem size of the monolithic version causes the solution time to increase more quickly than the decomposition method. This can also be seen in that the rate of increase of computation time for the monolithic is higher with more scenarios modelled, but is not for the decomposition method.

Overall, the crossover points where the monolithic takes more time to solve than the decomposition method all occur at fewer modelled scenarios as the problem size increases. For example, the decomposition method with 11 scenarios and a tolerance of 10^{-4} is faster than the monolithic with 11 scenarios if there are 10 parallel reactors, compared to 81 scenarios being need for the decomposition to be faster when only

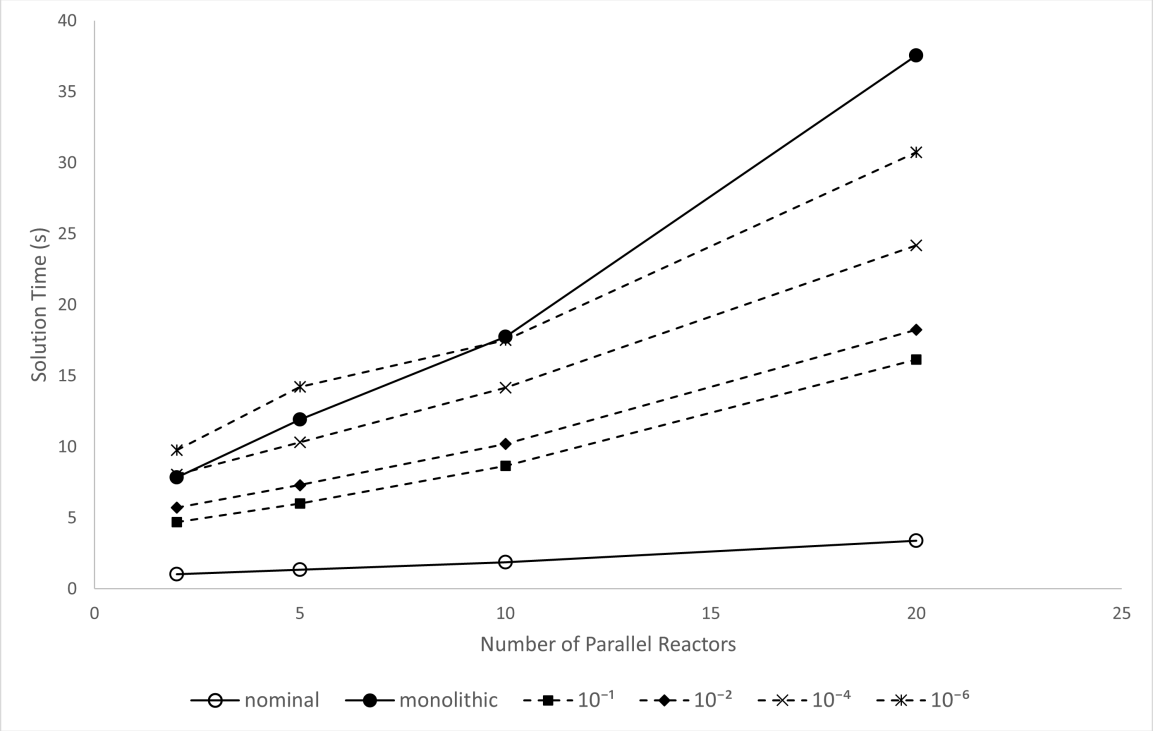

Figure 5.6: Solution time (s) of monolithic optimization problem and decomposition method at various tolerances by number of parallel reactors, for 21 modelled scenarios.

one reactor is present. This suggests that the decomposition method may be of more potential use in larger problems, as the monolithic method tends to slow in its computation time more drastically than the decomposition does as the overall problem size increases.

Subcase 1c – Nonlinear model

Thus far, the decomposition method has been tested only on a convex case. In this subcase, a nonlinear plant model is used to test the ability of the decomposition method to work on a nonconvex problem. This is relevant as convex problems are relatively easy to solve and so decomposition methods are rarely required. In order

to avoid adding much additional complexity here, this subcase still uses the same SISO case as above, with only one reactor, and a target tracking objective. The only other changes made, besides switching to a nonlinear plant model, are the addition of a soft set-point constraint of $-10 \leq y^{Ref} \leq 10$ with a penalty weight of 10^{-2} and a reduction in the magnitude of uncertainty. The set-point constraint simply keeps the set-points provided to the MPC from going too far from the desired target. The uncertain parameter is changed from having $\pm 20\%$ uncertainty to having $\pm 10\%$ uncertainty because the same change in the uncertain parameter in the nonlinear model leads to larger plant behavior changes than in the linear model. The output soft constraint upper bound is still present. Figure 5.7 shows the solution time of the monolithic method and the decomposition method with various tolerances for 1 to 25 modelled scenarios.

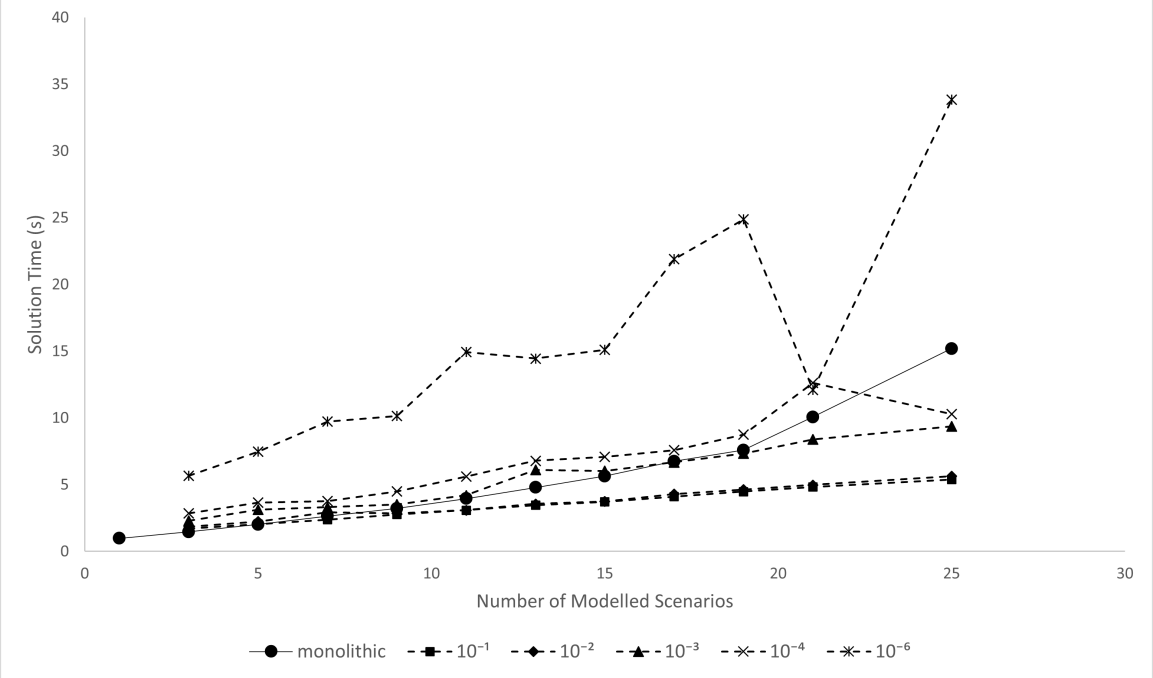

Figure 5.7: Solution time (s) of monolithic optimization problem and decomposition method at various tolerances by number of modelled scenarios, SISO nonlinear case.

Once again, it can be seen that the decomposition method solution time scales linearly with number of scenarios, while the monolithic problem scales nonlinearly. Overall, the decomposition method here solves the optimization problem in less time than the monolithic at fewer modelled scenarios than in the linear subcase above. The minimum scenarios modelled at which the monolithic is slower than the decomposition by tolerance is shown in Table 5.4, along with the average discrepancy from the monolithic solution. Figure 5.8 shows the expected objective value of the monolithic and decomposition methods at various tolerances for 1 to 25 modelled scenarios.

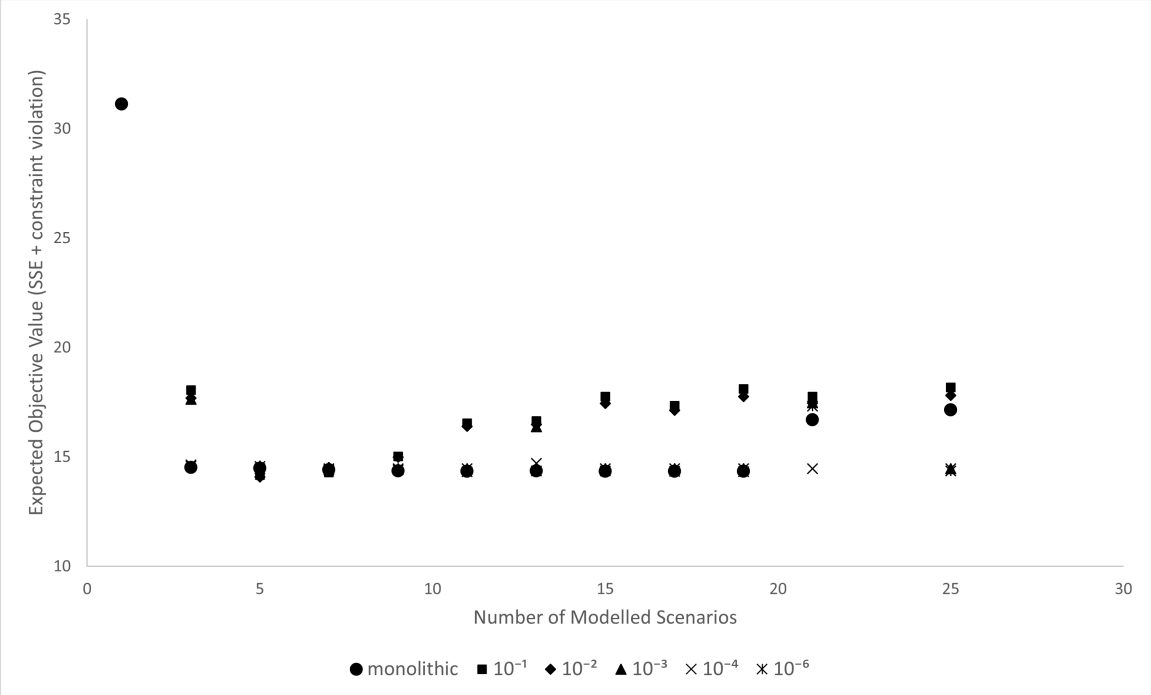

Figure 5.8: Expected objective from three characteristic scenario simulations of monolithic optimization problem and decomposition method using various tolerances by number of modelled scenarios, SISO nonlinear case.

As seen in Figure 5.8 and Table 5.4, the solutions found by the decomposition method at various tolerances once again generally improve as the tolerance tightens.

Table 5.4: Minimum scenarios for decomposition method to solve in less time than monolithic problem (crossover point) and expected performance relative to monolithic for various tolerances, SISO nonlinear case.

Decomposition Tolerance	**Crossover Point**	**Mean expected objective value**	**% change from monolithic**
Nominal	N/A	31.12	100
10^{-1}	5	16.70	+11.4
10^{-2}	9	16.51	+10.3
10^{-3}	17	15.21	+2.23
10^{-4}	25	14.50	-2.08
10^{-6}	N/A (>25)	14.65	-1.16
Monolithic	N/A	14.84	0

While the averages shown in Table 5.4 show the general improvement with tighter tolerances, this is not a consistent trend. Rather, the tighter tolerances seem to increase the likelihood that the decomposition method finds an improved solution. For example, with a tolerance of 10^{-2}, the decomposition method finds a solution within 5% of the monolithic for three cases but is over 10% worse in more than half of the cases, resulting in an overall average of about 10% discrepancy. By contrast, a tolerance of 10^{-4} results in the decomposition being only 2% worse in one case and less than 1% worse in the remaining simulations. This general improvement in consistency can be seen visually in Figure 5.8.

As this is now a nonconvex problem, it is possible for the monolithic problem to not find the optimal solution and instead converge to a different local optimum. This appears to have occurred when the monolithic problem is solved with 21 and 25 modelled scenarios. The decomposition method, on the other hand, finds a significantly better solution for these scenario cases with a tolerance of 10^{-4} and for 25 modelled scenarios with a tolerance of 10^{-6}. These results from the decomposition

method are also more similar to those found for fewer modelled scenarios, suggesting the decomposition method is converging to the desired local optimum. This is likely because the monolithic problem, as it increases in problem size, has difficulty effectively finding a solution and so can terminate at an undesirable local solution; this issue is not shared by the decomposition method due the small size of the optimization subproblems. It should be noted that the decomposition method for a tolerance of 10^{-6} with 21 scenarios finds a worse solution because this execution erroneously terminated prior to convergence and so did not find a local solution to this tolerance level. For the remaining simulations (with numbers of modelled scenarios under 21), the discrepancy between the solution found by the monolithic problem and the decomposition method with a tolerance of 10^{-4} and 10^{-6} (and often 10^{-3}) is too small ($< 1\%$, with only one exception) to be significant.

This subcase shows that the decomposition method presented works to find a local solution of nonconvex problems. It does, however, appear to require a somewhat tighter tolerance for comparable results to the monolithic method. In the linear case, the decomposition method with a tolerance of 10^{-2} finds solutions only about 4% worse than the monolithic on average, while in the nonlinear case it is about 10% worse for the same tolerance. However, there is potential in the nonconvex case to outperform the monolithic method, as mentioned previously. Even ignoring the two cases where the monolithic method doesn't find a good solution, the decomposition method with a tolerance of 10^{-4} does only 0.8% worse than the monolithic, and with a tolerance of 10^{-6} does 0.05% worse. Overall, the decomposition method generally performs better in the nonconvex case than the convex case in terms of computation time, solving faster than the monolithic at fewer scenarios while still finding similar

solutions.

5.3.2 Case 2 – MIMO with economic objective, linear model polymerization reactor

The last case study used here to investigate the effectiveness of the decomposition method seeks to determine if it is effective on problems with multiple inputs and multiple outputs (MIMO) and with an economic objective function. The first of these adds problem size both within the subproblems and in the master problem as a set-point trajectory must be provided to the MPC for each output and the master problem must evaluate the gradients of each of these set-point trajectories. This is a departure from the previous investigations into increases in problem size as previously only the subproblems were affected. The second change affects the complexity of the objective function, as the objective is no longer a sum of quadratic functions. This is important as a DRTO is unlikely to use a target tracking objective in a real application and so the decomposition method's ability to function with an economic objective is paramount to its potential future usefulness. Additionally, the sum of quadratic functions used previously lends itself well to the quadratic approximation line search used in the master problem algorithm. So it is important to show that the algorithm presented can work with other objectives. The case study here uses a linear plant model.

The exact case study used here is a three input, three output, six state polymerization reaction which has been used previously for CL-DRTO testing [27] and in the previous chapter. The system inputs are the inlet flowrates of the monomer, an initiator, and a solvent, respectively. The outputs of the system are the

number average molecular weight (NAMW) of the polymer product, the temperature in the reactor, and the concentration of monomer in the reactor. The objective is a profit band function where the polymer product can be sold when the NAMW is within a particular range, and thus revenue can be accrued. The binary nature of this objective is approximated using hyperbolic tangent functions as:

$$\phi_{econ} = \sum_i \rho^i \sum_{j=0}^{N} 10 \cdot R^i_{1,j} \cdot R^i_{2,j} - u^i_j \tag{5.3.3}$$

$$R^i_{1,j} = \frac{1}{2}\tanh\left(\gamma[y^i_j - (1-\delta)y^{tar}_j]\right) + \frac{1}{2} \tag{5.3.4}$$

$$R^i_{2,j} = \frac{1}{2}\tanh\left(\gamma[(1+\delta)y^{tar}_j - y^i_j]\right) + \frac{1}{2} \tag{5.3.5}$$

where γ and δ are fixed parameters which determine the slope of the tanh functions and the half-width of the profit band, respectively; u^i_j and y^i_j are the inputs and outputs of the system, respectively; in this case they represent the inlet concentration of reactant, C_{in}, and the concentration of product, P. The product of tanh switching function approximations in 5.3.3 gives $R^i_{1,j} \cdot R^i_{2,j} \approx 1$ if $(1-\delta)y^{tar}_j \leq y^i_j \leq (1+\delta)y^{tar}_j$; and is 0 otherwise. For this case study, a target for the NAMW of 68.9 kg/mol was used with a profit band width of $\pm 1\%$ ($y^{tar}_j = 68.9$, $\delta = 0.689$) and $\gamma = 10$.

In addition to this, there are also soft set-point constraints and output constraints whose penalties are added to the objective function (weights of 10^{-2} and 100, respectively). Finally, in order to aid in providing gradients to the master problem for the other output set-points (not the NAMW), a low weight (10^{-3}) target tracking objective was also added for all three outputs. This mainly aids in directing the master problem towards the optimal solution and, at convergence, represents $< 0.1\%$

of the total objective function value. This was also added to the objective in the monolithic problem for fair comparison. The DRTO and MPC parameters used for this case study can be seen in 5.5. Figure 5.9 shows the solution time for the monolithic method and decomposition method using various tolerances for 1 to 21 modelled scenarios.

Table 5.5: DRTO and MPC parameters for Case Study 2.

Parameter	**Description**	**Value**
N	DRTO horizon	20
Δt_{DRTO}	DRTO sample time	4 hr
Δt_{MPC}	MPC sample time	1 hr
p	MPC prediction horizon	8
m	MPC control horizon	2
Q	Output tracking weight	diag(1,1,1)
R	Input move suppression weight	diag(50,15,5)
S	Input tracking weight	diag(0,0,0)
T_s	Time Step	1 hr

Once again, it can be seen that the decomposition method solution time scales linearly with number of scenarios while the monolithic scales nonlinearly. However, in this MIMO case, the monolithic solution time increases much more quickly than the decomposition, as compared to the previous SISO case. As a result, the monolithic method is slower than even a fairly tight tolerance of 10^{-4} for 7 or more scenarios (a tolerance of 10^{-6} was not tested here as the results were nearly identical to those of 10^{-4} tolerance, with only marginally slower solution times). This shows that as the problem size and complexity increases, the decomposition method becomes more effective at reducing computation time of the optimization problem.

Specifically, the increase in problem size of the subproblems leads to lesser

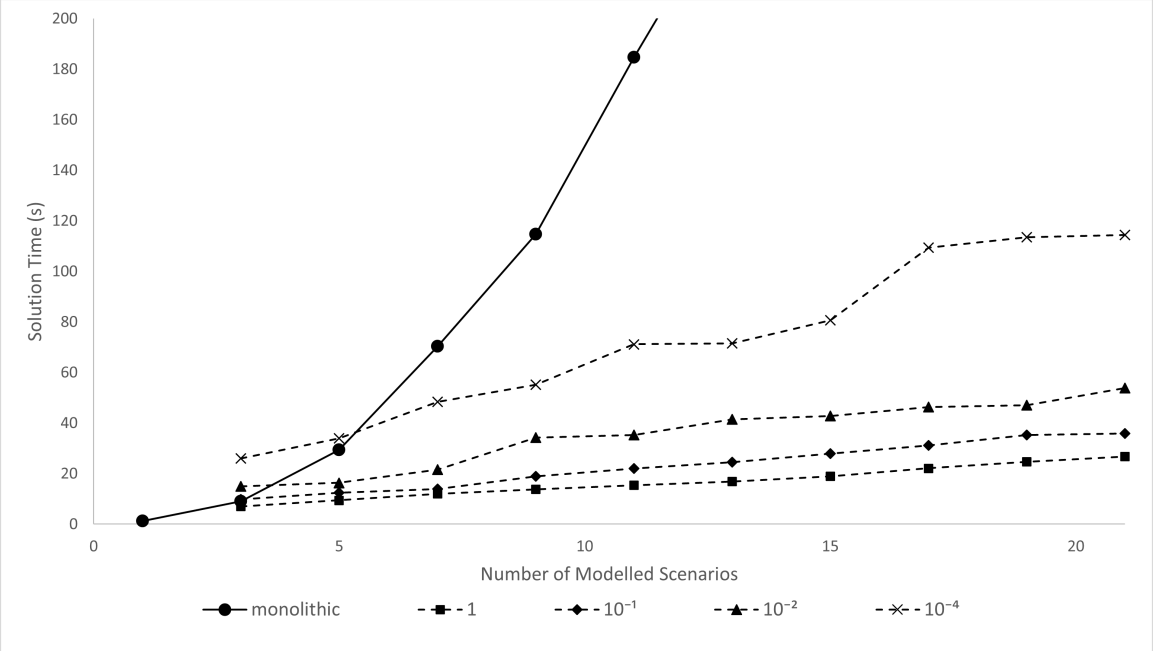

Figure 5.9: Solution time (s) of monolithic optimization problem and decomposition method at various tolerances by number of modelled scenarios, MIMO case.

increases in solution time as the subproblems are relatively small and manageable. The increase in complexity from the economic objective in conjunction with the increase in number of set-points does lead to an increase in number of required iterations, thus increasing the solution time of the decomposition method. But this increase is simply not as large, especially for many modelled scenarios, as that seen in the monolithic problem.

Figure 5.10 shows the expected value of the economic objective minus the constraint violation for the monolithic method and decomposition method for various tolerances for 1 to 21 modelled scenarios. A summary of average performance (economic objective – constraint violation) and solution time crossover point by tolerance relative to the monolithic solution is shown in Table 5.6. It should be noted that this is a maximization objective, as opposed to the minimization of the

previous cases.

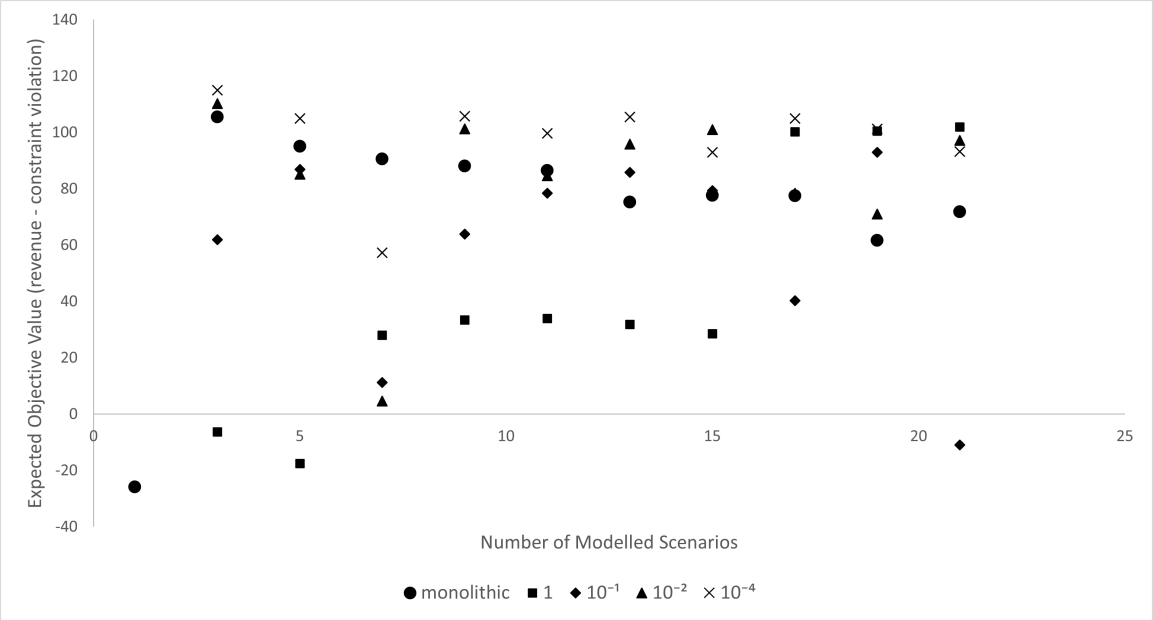

Figure 5.10: Expected objective from three characteristic scenario simulations of monolithic optimization problem and decomposition method using various tolerances by number of modelled scenarios, MIMO case.

Overall, it can be seen in Table 5.6 that a tolerance of 1 or 10^{-1} leads to substantially reduced performance compared to the monolithic, a tolerance of 10^{-2} yields very similar average results, and a tolerance of 10^{-4} produces significant improvements over the monolithic. As seen in Figure 5.10, this general trend of improvement as tolerance decreases is, once again, a result largely of consistency improvements. For example, a tolerance of 1 produces mostly worse results than the monolithic, but it does better than the monolithic in 3 cases (17, 19, and 21 scenarios). A tolerance of 10^{-1} yields 4 cases with better results than the monolithic and 6 cases of worse results. A tolerance of 10^{-2} shows 3 cases with worse results and the rest improvements. A tolerance of 10^{-4} shows all but one case having improved performance over the monolithic method. Furthermore, the decomposition method

Table 5.6: Minimum scenarios for decomposition method to solve in less time than monolithic problem (crossover point) and expected performance relative to monolithic for various tolerances, MIMO case.

Decomposition Tolerance	**Crossover Point**	**Mean expected objective value**	**% change from monolithic**
Nominal	N/A	-25.86	-100
1	3	43.36	-36.3
10^{-1}	5	58.89	-22.1
10^{-2}	5	82.86	-0.037
10^{-4}	7	97.98	+13.9
Monolithic	N/A	82.89	0

with a tolerance of 10^{-4} produced significantly improved results compared to the monolithic method on average, while solving in less time for problems with more than 5 scenarios. This suggests that for MIMO problems it is likely to be beneficial to use the decomposition method instead of the monolithic solution.

The decomposition method does not seem to have a particular difficulty with the economic objective function or the increased complexity from dealing with multiple set-point trajectories and their associated gradients. While the solutions found by the decomposition method are not very consistent across number of modelled scenarios, the solutions found by the monolithic problem are not significantly more consistent either. This is unlikely to be from the model-plant discrepancy as that was found previously to be relatively small in effect size, where these inconsistencies are quite significant; and because the discrepancy did not seem to depend on number of modelled scenarios, which is the case here. Rather, it seems likely that the nature of the MIMO problem size and complexity with many scenarios simply makes finding a good solution quite difficult regardless of method used.

In this MIMO case with an economic objective function, the decomposition approach presented here is able to effectively reduce the required computation time over the monolithic solution method, especially as the number of modelled scenarios increases. The relative improvement seen in computation time is more pronounced for the MIMO case than the SISO case, suggesting that an increase in problem size lends itself to the decomposition method. This is consistent with the result of subcase 1b which increased problem size through number of states only.

5.3.3 Discussion of Case Study Results

The primal decomposition method presented here shows potential benefits over a monolithic method for relatively large problem size, especially as a result of a large number of modelled scenarios. This is because the subproblems of the decomposition method are single scenario problems, and thus do not grow with number of modelled scenarios. Instead, the number of subproblems increases. Thus the decomposition solution time increases linearly with number of scenarios while the monolithic solution increases nonlinearly (it appears to be roughly quadratic). For problems sizes resulting not from number of scenarios (number of states, inputs, outputs, model equations, etc.), the monolithic problem and the scenario subproblems both increase nonlinearly, but the larger size of the monolithic problem results in larger proportional computation time increases.

In terms of quality of solution, the decomposition method performance depends heavily on the convergence tolerance. For convex problems, the decomposition method is capable of converging to an identical solution as the monolithic problem with a sufficiently small tolerance, though the resulting solution time is unlikely to

be smaller than the monolithic unless the problem is very large. The decomposition method can achieve similar results to the monolithic method with substantially looser tolerances, resulting in significantly reduced computation time. For nonconvex problems, the decomposition method can, again, reach similar results to the monolithic problem with sufficiently tight tolerance. Furthermore, for large nonconvex problems (whether as a result of number of scenarios or otherwise), the monolithic problem is not always able to find a good solution, and so the decomposition method is capable of outperforming it in terms of expected objective value.

As a result of the above points, there are cases shown where the decomposition method solves the optimization problem in less time than the monolithic method and produces a better solution in terms of the objective function used. In the SISO nonconvex case, a tolerance of 10^{-4} and 25 modelled scenarios yields a decomposition solution better than that found in the monolithic method, with less computation time. In the MIMO convex case, the decomposition method with a tolerance of 10^{-2} or tighter with 9 or more scenarios produces a better solution in less time the monolithic.

Therefore, there are applications which have reasonably large problem sizes and a fairly large number of scenarios for which the decomposition method would outperform the monolithic method in terms of both economic objective and solution time. Furthermore, if an application has to solve a DRTO in a relatively short time, this method may be a good choice as it will always give a feasible solution which is not worse than the initial solution, even if it does not converge. This can be seen in all case studies in the general improvements in solution as the tolerance decreases at

the cost of computation time.

5.4 Conclusion

The proposed decomposition method has been shown to effectively reduce the solution time of robust CL-DRTO problems for which the problem size is relatively large. The primary difference in computation time compared to a monolithic solution method is that the decomposition method solution time increases more slowly than the monolithic method as the number of scenarios increases. As it has been shown in the previous chapter that increasing the number of scenarios can improve performance, this is an especially important result. Additionally, the decomposition method generally solves in less time than the monolithic for large problems, even for relatively few scenarios. Finally, for large problems with many scenarios, the decomposition method not only reduces computation time dramatically, but is also able to significantly outperform the monolithic in terms of expected objective.

Chapter 6

Conclusion

6.1 Summary and Contributions

The primary purpose of this research is to extend the closed-loop prediction strategy previously employed for CL-DRTO to settings where stability and uncertainty are major concerns and so must be incorporated into the optimization and control framework. These concerns are often included in the control hierarchy of industrial chemical engineering applications and so the ability for the CL prediction strategy to also include these concerns is necessary for its employment in many of these applications. By allowing the CL prediction strategy to be used in additional applications, its usefulness is extended; and the applications to which it is applied may see improvements in performance and safety as a result. The key contributions of this research are as follows:

1. **A single-level robust multi-scenario MPC with closed-loop prediction**
   We apply the CL prediction strategy to a robust MPC framework to allow the

multi-scenario modelling of the MPC to predict both the plant behavior and the future MPC behavior subject to uncertainty. The future MPC subproblems use a non-robust MPC model to maintain convexity. This is a reasonable approximation of the future robust MPC behavior. The overall CL robust MPC strategy is shown to outperform a non-robust MPC under a range of uncertainty realizations.

2. **A convex Lyapunov-based stabilizing MPC formulation for nonlinear systems**

   A Lyapunov MPC which uses a linear plant prediction model with a nonlinear Lyapunov stability constraint is formulated to produce a convex LMPC. Such a formulation has the advantage of ease of computation compared to a nonlinear MPC while maintaining stability under a wider range of plant conditions than a linear model, linear constraint stability MPC.

3. **Lyapunov MPC embedded in a CL-DRTO framework**

   The convex LMPC formulation allows for its inclusion as the MPC subproblems in a CL-DRTO framework. This CL-DRTO with LMPC strategy models an underlying LMPC which is directly controlling an unstable plant system. The two-level optimization and control strategy is able to effectively optimize an unstable system under a wider range of plant conditions than a similar CL-DRTO strategy with an endpoint penalty stabilizing MPC.

4. **A multi-scenario robust CL-DRTO formulation**

   We present a CL-DRTO method which incorporates uncertainty handling at the economic optimization level by using a multi-scenario approach for the

CL-DRTO. This approach models multiple possible plant realizations and the associated MPC behavior in these scenario realizations. The underlying MPC is assumed to be non-robust. The robust CL-DRTO method is shown to outperform a non-robust, single-scenario CL-DRTO over a range of uncertain plant realizations.

5. **Effects of number of modelled scenarios on CL-DRTO performance**
   Investigating the performance of CL-DRTO reveals that, when executed on a random sampling of possible plant realizations, increasing the number of modelled scenarios also improves the overall performance of the economic optimization. This is because the CL-DRTO is able to predict a greater variety of plant behaviors and determine set-point moves accordingly. However, the computation time of the CL-DRTO increases significantly as the number of scenarios within the CL-DRTO increases.

6. **Decomposition of robust CL-DRTO**
   We propose a primal decomposition method for the robust CL-DRTO strategy. This method separates the multi-scenario problem into individual single-scenario subproblems where a portion of the set-point trajectory within each subproblem is fixed by the master problem. The decomposition approach substantially reduces the computation time compared to the monolithic multi-scenario method for large problem sizes without significant loss of performance, particularly for large numbers of modelled scenarios.

6.2 Future Work

There are many possible avenues for future research on the topics covered in this work. A few broad areas are identified and outlined in this section.

6.2.1 Variations on existing strategies

This is an especially broad category of future work so a few particular examples will be outlined here and briefly explored. The key characteristic of these variations is that they do not require new solution techniques to be implemented.

One aspect of uncertainty which is not explored in either chapter 1 or chapter 3 above is that of time-varying uncertainty. In both chapters, uncertainty is modelled in the form of an uncertain time-invariant parameter. However, there are many other sources of uncertainty in real applications which are time-varying, including parameters and others, such as demand uncertainty or disturbances. The ability to model this time-varying uncertainty in a CL MPC or CL-DRTO would further expand on the methods' ability to be used in applications with uncertain characteristics. However, effectively doing so would likely require additional scenario splits beyond the first time point, thereby increasing the problem size.

The use of stabilizing MPC in CL-DRTO has been explored with both endpoint penalty and Lyapunov MPC strategies. However, there are many more stabilizing MPC strategies. Incorporating any of these other stabilizing MPC strategies into the CL-DRTO algorithm would be a worthwhile endeavor as it may expand the usefulness to more systems which require stabilizing control elements. Some of these strategies may be adjustable (similar to the Lyapunov MPC in this work) to be convex quadratic, if not already convex (as with the endpoint penalty MPC investigated

previously), allowing relatively simple incorporation into the CL-DRTO system.

Similarly, a primal decomposition method is explored here for robust CL-DRTO, but many other decomposition techniques exist, including Bender's decomposition and dual decomposition techniques. While many of these techniques do not have the feasibility guarantees of the primal method, they may prove to perform better in terms of time to convergence. A full exploration of the relative performance of various decomposition techniques in a robust CL-DRTO setting would be valuable for determining the ideal method of decomposition.

6.2.2 Formulation of CL-DRTO with non-convex subproblems

For the entirety of the work here, it is assumed that the underlying MPC which is modelled in the CL-DRTO is convex. For many industrial MPC uses, this is not the case and the real MPC being implemented constitutes a non-convex problem. Therefore it would be beneficial if the CL-DRTO method were to be able to model the behavior of a non-convex underlying MPC. However, the first-order KKT conditions of a non-convex problem are not sufficient for optimality. Therefore, a different solution method, such as a sequential approach, would be required.

The implementation of non-convex subproblems into the CL-DRTO strategy allows for its use in a much wider range of applications. This includes a robust CL MPC able to model its own future behavior more accurately, as it could include a multi-scenario robust MPC as its future MPC subproblems, rather than a nominal MPC. Additionally, a CL-DRTO with a non-convex stabilizing MPC, such as a Lyapunov MPC with both nonlinear model and stabilizing constraints, is also a possible extension. For applications where uncertainty is a major concern for both

short-term control and long-term economic optimization, a robust CL-DRTO which models an underlying robust MPC may be appropriate as it would allow for both control and optimization layers to account for plant uncertainty.

6.2.3 Inclusion of scheduling decisions into CL-DRTO

The incorporation of higher order scheduling decisions into a CL-DRTO strategy is an ongoing area of research in the Swartz group. Remigio and Swartz [69] successfully combined a linear model CL-DRTO with a scheduling optimization problem. This resulted in a mixed-integer linear program (MILP). Dering and Swartz [14] extend this approach to a multi-scenario CL-DRTO to handle uncertainty and also use piecewise-affine models within the DRTO to increase fidelity while maintaining the problem as an MILP. Future research may seek to combine these efforts with the methods presented in this work to allow for a CL-DRTO problem to include scheduling decisions, underlying MPC behavior, use of a nonlinear plant model, and uncertainty handling.

However, a nonlinear CL-DRTO with scheduling decisions would result in a mixed-integer nonlinear program (MINLP) which are substantially more difficult to solve. Therefore, the use of decomposition techniques may be necessary to solve such a formulation and have it be reasonable for inclusion in a real-time setting. A method suited for decomposition by integer variables, such as Generalized Bender's Decomposition, is likely to be useful. Furthermore, inclusion of uncertainty into this problem would further increase the computational complexity, but use of a second layer of decomposition may allow for efficient solution. The resulting problem would include decomposition by integer variable at one layer and decomposition by scenario

at the next layer. The number of subproblems in such a formulation would therefore be large, but overall it may result in a much easier problem to solve than a single, large MINLP.

6.2.4 Parameter estimation for improvement of robust CL-DRTO model

Parameter estimation is another approach used to handle uncertainty at the RTO level. In such an approach, data from plant measurements is used to update the model of the RTO or DRTO at certain intervals by performing parameter estimation in conjunction with the real-time optimization. Matias and Le Roux [51] develop a strategy where this parameter estimation can occur prior to the system reaching steady state, thus allowing the RTO to improve its model during transitions. This is a significant advantage over previous methods which had to wait for steady state and thus the optimization model could be inaccurate for a potentially long transition period. Patrón and Ricardez-Sandoval [62] use an information content metric to selectively choose which measurements to use for parameter estimation in order to reduce the effects of noise. Zhang et al. [80] use a modifier adaptation principle to not only update parameters but also adjust modifiers which are designed to account for structural plant-model mismatch.

With the amount of active research surrounding parameter estimation for RTO, it stands to reason that a parameter estimation step may improve the performance of a robust CL-DRTO. The parameter estimation may allow for the size of the uncertainty region of the robust CL-DRTO to decrease as plant measurements become available. Furthermore, if the range of possible plant scenarios was chosen to be too restrictive

or was substantially different from the real plant behavior, the parameter estimation step could serve to move the entire uncertainty region towards the observed plant behavior. Therefore, the inclusion of a parameter estimation step alongside a robust CL-DRTO may allow for improved performance as plant measurements can serve to improve the plant and uncertainty models of the DRTO.

Bibliography

[1] Armijo, L., 1966. Minimization of functions having lipschitz continuous first partial derivatives. Pacific Journal of mathematics 16, 1–3.

[2] Baker, R., Swartz, C.L.E., 2008a. Interior point solution of multilevel quadratic programming problems in constrained model predictive control applications. Industrial & Engineering Chemistry Research 47, 81–91.

[3] Baker, R., Swartz, C.L.E., 2008b. Interior point solution of multilevel quadratic programming problems in constrained model predictive control applications. Industrial & Engineering Chemistry Research 47, 81 – 91.

[4] Bemporad, A., Borrelli, F., Morari, M., 2003. Min-max control of constrained uncertain discrete-time linear systems. IEEE Transactions on Automatic Control 48, 1600–1606.

[5] Bemporad, A., Morari, M., 1999. Robust model predictive control: A survey, in: Robustness in Identification and Control. Springer, pp. 207–226.

[6] Boyd, S., Xiao, L., Mutapcic, A., Mattingley, J., 2007. Notes on decomposition methods. Notes for EE364B, Stanford University 635, 1–36.

[7] Broyden, C.G., 1970. The convergence of a class of double-rank minimization algorithms 1. general considerations. IMA Journal of Applied Mathematics 6, 76–90.

[8] Burnak, B., Katz, J., Diangelakis, N.A., Pistikopoulos, E.N., 2018. Simultaneous process scheduling and control: A multiparametric programming-based approach. Industrial & Engineering Chemistry Research 57, 3963–3976.

[9] Byrd, R.H., Lu, P., Nocedal, J., Zhu, C., 1995. A limited memory algorithm for bound constrained optimization. SIAM Journal on scientific computing 16, 1190–1208.

[10] Campo, P.J., Morari, M., 1987. Robust model predictive control, in: American Control Conference, pp. 1021–1026.

[11] Cutler, C.R., Ramaker, B.L., 1979. Dynamic matrix control - a computer control algorithm, in: AIChE 86th National Meeting, Houston, TX, USA.

[12] Darby, M.L., Nikolaou, M., Jones, J., Nicholson, D., 2011. RTO: An overview and assessment of current practice. Journal of Process Control 21, 874–884.

[13] De La Peña, D.M., Alamo, T., Bemporad, A., Camacho, E.F., 2006. A decomposition algorithm for feedback min–max model predictive control. IEEE Transactions on Automatic Control 51, 1688–1692.

[14] Dering, D., Swartz, C.L., 2022. A stochastic optimization framework for integrated scheduling and control under demand uncertainty. Computers & Chemical Engineering 165, 107931.

[15] Dias, L.S., Pattison, R.C., Tsay, C., Baldea, M., Ierapetritou, M.G., 2018. A simulation-based optimization framework for integrating scheduling and model predictive control, and its application to air separation units. Computers & Chemical Engineering 113, 139–151.

[16] Fletcher, R., 1970. A new approach to variable metric algorithms. The computer journal 13, 317–322.

[17] Gao, L., 2012. Modeling and dynamics analyses of immobilized CSTR bioreactor using transfer function model, in: 2012 International Symposium on Information Technologies in Medicine and Education, IEEE. pp. 692–695.

[18] Garcia, C.E., Morshedi, A., 1986. Quadratic programming solution of dynamic matrix control (QDMC). Chemical Engineering Communications 46, 73–87.

[19] Garone, E., Di Cairano, S., Kolmanovsky, I., 2017. Reference and command governors for systems with constraints: A survey on theory and applications. Automatica 75, 306–328.

[20] Goldfarb, D., 1970. A family of variable-metric methods derived by variational means. Mathematics of Computation 24, 23–26.

[21] Gunnerud, V., Foss, B., Torgnes, E., 2010. Parallel dantzig–wolfe decomposition for real-time optimization—applied to a complex oil field. Journal of Process Control 20, 1019–1026.

[22] Heidarinejad, M., Liu, J., Christofides, P.D., 2012. Economic model predictive control of nonlinear process systems using lyapunov techniques. AIChE Journal 58, 855–870.

[23] Holtorf, F., Mitsos, A., Biegler, L.T., 2019. Multistage NMPC with on-line generated scenario trees: Application to a semi-batch polymerization process. Journal of Process Control 80, 167–179.

[24] Jalali, M., Zare, K., Seyedi, H., Alipour, M., Wang, F., 2019. Distributed model for robust real-time operation of distribution systems and microgrids. Electric Power Systems Research 177, 105985.

[25] Jamaludin, M.Z., Li, H., Swartz, C.L.E., 2017. The utilization of closed-loop prediction for dynamic real-time optimization. The Canadian Journal of Chemical Engineering 95, 1968–1978. https://onlinelibrary.wiley.com/doi/pdf/10.1002/cjce.22927.

[26] Jamaludin, M.Z., Swartz, C.L., 2016. Closed-loop formulation for nonlinear dynamic real-time optimization. IFAC-PapersOnLine 49, 406 – 411. 11th IFAC Symposium on Dynamics and Control of Process Systems 2016.

[27] Jamaludin, M.Z., Swartz, C.L., 2017a. Dynamic real-time optimization with closed-loop prediction. AIChE Journal 63, 3896–3911.

[28] Jamaludin, M.Z., Swartz, C.L.E., 2015. A bilevel programming formulation for dynamic real-time optimization. IFAC-PapersOnLine 48, 906–911.

[29] Jamaludin, M.Z., Swartz, C.L.E., 2017b. Approximation of closed-loop prediction for dynamic real-time optimization calculations. Computers & Chemical Engineering 103, 23–38.

[30] Kadam, J., Schlegel, M., Marquardt, W., Tousain, R., Van Hessem, D., van den Berg, J., Bosgra, O., 2002. A two-level strategy of integrated dynamic

optimization and control of industrial processes—a case study, in: Computer Aided Chemical Engineering. Elsevier. volume 10, pp. 511–516.

[31] Kothare, M.V., Balakrishnan, V., Morari, M., 1996. Robust constrained model predictive control using linear matrix inequalities. Automatica 32, 1361–1379.

[32] Kouvaritakis, B., Rossiter, J., Schuurmans, J., 2000. Efficient robust predictive control. IEEE Transactions on Automatic Control 45, 1545–1549.

[33] Krishnamoorthy, D., Foss, B., Skogestad, S., 2016. Real-time optimization under uncertainty applied to a gas lifted well network. Processes 4, 52.

[34] Krishnamoorthy, D., Foss, B., Skogestad, S., 2018a. A distributed algorithm for scenario-based model predictive control using primal decomposition, in: 10th IFAC International Symposium on Advanced Control of Chemical Processes, Shenyang, Liaoning, China, July 25-27, pp. 345–350.

[35] Krishnamoorthy, D., Foss, B., Skogestad, S., 2019. A primal decomposition algorithm for distributed multistage scenario model predictive control. Journal of Process Control 81, 162–171.

[36] Krishnamoorthy, D., Thombre, M., Skogestad, S., Jaschke, J., 2018b. Data-driven scenario selection for multistage robust model predictive control, in: 6th IFAC Conference on Nonlinear model Predictive Control, Madison, Wisconsin, USA, August 19-22, pp. 558–564.

[37] Langson, W., Chryssochoos, I., Raković, S., Mayne, D.Q., 2004. Robust model predictive control using tubes. Automatica 40, 125–133.

[38] Lee, J.H., Yu, Z., 1997. Worst-case formulations of model predictive control for systems with bounded parameters. Automatica 33, 763–781.

[39] Li, H., Swartz, C.L., 2019. Dynamic real-time optimization of distributed mpc systems using rigorous closed-loop prediction. Computers & Chemical Engineering 122, 356–371.

[40] Li, H., Swartz, C.L., 2021. Robust model predictive control via multi-scenario reference trajectory optimization with closed-loop prediction. Journal of Process Control 100, 80–92.

[41] Li, H., Swartz, C.L.E., 2018. Approximation techniques for dynamic real-time optimization (DRTO) of distributed MPC systems. Computers & Chemical Engineering 118, 195–209.

[42] Loeblein, C., Perkins, J.D., 1999. Structural design for on-line process optimization: I. Dynamic economics of MPC. AIChE Journal 45, 1018–1029.

[43] Luan, X., De Schutter, B., Meng, L., Corman, F., 2020. Decomposition and distributed optimization of real-time traffic management for large-scale railway networks. Transportation Research Part B: Methodological 141, 72–97.

[44] Lucia, S., Finkler, T., Engell, S., 2013a. Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. Journal of Process Control 23, 1306–1319.

[45] Lucia, S., Subramanian, S., Engell, S., 2013b. Non-conservative robust nonlinear model predictive control via scenario decomposition, in: 2013 IEEE International Conference on Control Applications (CCA), IEEE. pp. 586–591.

[46] Maciejowski, J.M., 2002. Predictive Control with Constraints. Prentice Hall, Essex, England.

[47] Maner, B.R., Doyle III, F.J., Ogunnaike, B.A., Pearson, R.K., 1996. Nonlinear model predictive control of a simulated multivariable polymerization reactor using second-order volterra models. Automatica 32, 1285–1301.

[48] Marlin, T.E., Hrymak, A.N., 1997. Real-time operations optimization of continuous processes. AIChE Symposium Series: Proceedings of the 5th International Conference on Chemical Process Control 5, 156 – 164.

[49] Marti, R., Lucia, S., Sarabia, D., Paulen, R., Engell, S., de Prada, C., 2015. Improving scenario decomposition algorithms for robust nonlinear model predictive control. Computers & Chemical Engineering 79, 30–45.

[50] Mastragostino, R., Patel, S., Swartz, C.L.E., 2014. Robust decision making for hybrid process supply chain systems via model predictive control. Computers & chemical engineering 62, 37–55.

[51] Matias, J.O., Le Roux, G.A., 2018. Real-time optimization with persistent parameter adaptation using online parameter estimation. Journal of Process Control 68, 195–204.

[52] Mayne, D., Rawlings, J., Rao, C., Scokaert, P., 2000. Constrained model predictive control: Stability and optimality. Automatica 36, 789 – 814.

[53] Mayne, D.Q., Kerrigan, E.C., van Wyk, E.J., Falugi, P., 2011. Tube-based robust nonlinear model predictive control. International Journal of Robust and Nonlinear Control 21, 1341–1353.

[54] Mhaskar, P., El-Farra, N.H., Christofides, P.D., 2005. Predictive control of switched nonlinear systems with scheduled mode transitions. IEEE Transactions on Automatic Control 50, 1670–1680.

[55] Mhaskar, P., El-Farra, N.H., Christofides, P.D., 2006. Stabilization of nonlinear systems with state and control constraints using lyapunov-based predictive control. Systems & Control Letters 55, 650–659.

[56] Michalska, H., Mayne, D.Q., 1993. Robust receding horizon control of constrained nonlinear systems. IEEE transactions on automatic control 38, 1623–1633.

[57] Müller, D., Illner, M., Esche, E., Pogrzeba, T., Schmidt, M., Schomäcker, R., Biegler, L.T., Wozny, G., Repke, J.U., 2017. Dynamic real-time optimization under uncertainty of a hydroformylation mini-plant. Computers & Chemical Engineering 106, 836–848.

[58] Muske, K.R., Rawlings, J.B., 1993a. Linear model predictive control of unstable processes. Journal of Process Control 3, 85 – 96.

[59] Muske, K.R., Rawlings, J.B., 1993b. Model predictive control with linear models. AIChE Journal 39, 262–287.

[60] Narraway, L., Perkins, J., Barton, G., 1991. Interaction between process design and process control: economic analysis of process dynamics. Journal of Process Control 1, 243–250.

[61] Nath, R., Alzein, Z., 2000. On-line dynamic optimization of olefins plants. Computers & Chemical Engineering 24, 533–538.

[62] Patrón, G.D., Ricardez-Sandoval, L., 2022. Low-variance parameter estimation approach for real-time optimization of noisy process systems. Industrial & Engineering Chemistry Research 61, 16780–16798.

[63] Muñoz de la Peña, D., Bemporad, A., Alamo, T., 2005. Stochastic programming applied to model predictive control, in: IEEE Conf on Decision and Control, and the European Control Conference 2005, Seville, Spain, Dec 12-15, pp. 1361–1366.

[64] Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. Control engineering practice 11, 733–764.

[65] Rahmaniani, R., Crainic, T.G., Gendreau, M., Rei, W., 2017. The benders decomposition algorithm: A literature review. European Journal of Operational Research 259, 801–817.

[66] Ralph, D., Wright, S.J., 2004. Some properties of regularization and penalization schemes for MPECs. Optimization Methods & Software 19, 527–556.

[67] Ramesh, P.S., Swartz, C.L.E., Mhaskar, P., 2021. Closed-loop dynamic real-time optimization with stabilizing model predictive control. AIChE Journal 67, e17308.

[68] Rawlings, J.B., Mayne, D.Q., Diehl, M.M., 2019. Model Predictive Control: Theory, Computation, and Design. Nob Hill Publishing, Santa Barbara, California. 2nd edition.

[69] Remigio, J.E., Swartz, C.L., 2020. Production scheduling in dynamic real-time optimization with closed-loop prediction. Journal of Process Control 89, 95–107.

[70] Sakizlis, V., Kakalis, N.M.P., Dua, V., Perkins, J.D., Pistikopoulos, E.N., 2004. Design of robust model-based controllers via parametric programming. Automatica 40, 189–201.

[71] Shanno, D.F., 1970. Conditioning of quasi-newton methods for function minimization. Mathematics of computation 24, 647–656.

[72] Singh, A., Forbes, J., Vermeer, P., Woo, S., 2000. Model-based real-time optimization of automotive gasoline blending operations. Journal of process control 10, 43–58.

[73] Soliman, M., Swartz, C.L.E., Baker, R., 2008. A mixed-integer formulation for back-off under constrained predictive control. Computers & Chemical Engineering 32, 2409–2419.

[74] Swartz, C.L.E., Kawajiri, Y., 2019. Design for dynamic operation-a review and new perspectives for an increasingly dynamic plant operating environment. Computers & Chemical Engineering 128, 329–339.

[75] Tosukhowong, T., Lee, J.M., Lee, J.H., Lu, J., 2004. An introduction to a dynamic plant-wide optimization strategy for an integrated plant. Computers & Chemical Engineering 29, 199–208.

[76] Wan, Z., Kothare, M.V., 2002. Robust output feedback model predictive control using off-line linear matrix inequalities. Journal of Process Control 12, 763–774.

[77] Würth, L., Hannemann, R., Marquardt, W., 2009. Neighboring-extremal updates for nonlinear model-predictive control and dynamic real-time optimization. Journal of Process Control 19, 1277–1288.

[78] Würth, L., Hannemann, R., Marquardt, W., 2011. A two-layer architecture for economically optimal process control and operation. Journal of Process Control 21, 311 – 321. Thomas McAvoy Festschrift.

[79] Ying, C.M., Joseph, B., 1999. Performance and stability analysis of lp-mpc and qp-mpc cascade control systems. AIChE Journal 45, 1521–1534.

[80] Zhang, D., Wang, K., Xu, Z., Tula, A.K., Shao, Z., Zhang, Z., Biegler, L.T., 2022. Generalized parameter estimation method for model-based real-time optimization. Chemical Engineering Science 258, 117754.

[81] Zhang, H., Li, S., Wang, Y., Yang, L., Gao, Z., 2021. Collaborative real-time optimization strategy for train rescheduling and track emergency maintenance of high-speed railway: A lagrangian relaxation-based decomposition algorithm. Omega 102, 102371.

[82] Zhang, Y., Monder, D., Forbes, J.F., 2002. Real-time optimization under parametric uncertainty: a probability constrained approach. Journal of Process Control 12, 373–389.