State and Parameter Estimation in Closed-Loop Dynamic Real-Time Optimization

State and Parameter Estimation in Closed-Loop Dynamic Real-Time Optimization

by

Andrew Solano, B.Ch.E

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree
Master of Applied Science

McMaster University

MASTER OF APPLIED SCIENCE (2024)	McMaster University
(Chemical Engineering)	Hamilton, Ontario, Canada

TITLE:	State and Parameter Estimation in Closed-Loop Dynamic Real-Time Optimization
AUTHOR:	andrew Solano B.Ch.E. (Brigham Young University Provo, United States)
SUPERVISOR:	Dr. Christopher L.E. Swartz
NUMBER OF PAGES	xi, 127

Abstract

To adapt to overarching objectives and changing demands, a plant automation system capable of real-time optimization and dynamic model predictions is desirable. Dynamic real-time optimization (DRTO) can achieve higher level objectives such as profitability, however RTO and DRTO schemes require a mechanism to utilize plant measurements to adapt the model to reflect changing conditions. This study proposes a novel integration of Kalman filter state and parameter estimation in which the impact of the controller and the plant response is accounted for in the DRTO. This closed-loop DRTO (CL-DRTO) approach is used to control a multi-input multi-output CSTR where a critical parameter is not measurable. The CSTR is optimized under economic and target tracking objectives, and is tested using two different control layers, PI-based and MPC-based. In the PI controlled CSTR, the proposed solution was compared to the ideal case of full state feedback and a common approach to dealing with mismatch: bias updating. The proposed Kalman filter estimator effectively handles noise and infeasible targets, surpassing bias updating in scenarios involving input saturation and increased measurement noise. The PI controlled CSTR is also tested with nonlinear models and an extended Kalman filter, demonstrating a method for controlling even highly nonlinear systems. In the MPC controlled CSTR, the Kalman filter is tested under input saturation and various disturbance sources. By using DRTO setpoints to guide the MPC towards targets, inputs can be maintained at their constrained bounds without directly accounting for these constraints in the MPC formulation or clipping the inputs directly. Under every scenario tested, the Kalman filter successfully estimated the unknown parameter and demonstrated excellent robustness. The proposed strategy's ability to control nonlinear plants using linear models suggests potential scalability for larger, more complex systems.

Acknowledgments

I'd like to thank Dr. Christopher L.E. Swartz for his insights and guidance as I studied the subject matter and pursued my research goals. His expertise and willingness to help me find research I would enjoy made this project possible and fulfilling.

I want to thank José Matias for his detailed explanations on the inner mechanics of the complex algorithms and code structures. His example and tutelage helped me learn the subject matter quickly and develop skills in fields I could not have learned otherwise. There are many obstacles I overcame in my research only because José was willing to walk through the problem with me and help troubleshoot as long as was needed.

I'm very grateful for the financial assistance I received from the McMaster University to cover my expenses and tuition while I focus my energy on research. The generous funding drew me to Canada and set the university apart from other research alternatives.

I could not have done this without my family, namely my wife Frances. Without her support and encouragement, I could not have pursued my dreams and achieved my research goals.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Research Objectives

Optimizing chemical processing facilities and their interconnected elements is critical, but often complex. These facilities have design limitations, operational goals, and local control objectives to account for. To optimize these systems, optimization structures must consider high frequency control targets as well as low frequency plant-wide objectives. An example of this type of tiered control is given in Figure 1.1, adapted from Marlin and Hrymak [1]. Typically, these methods achieve broader, long-term objectives by optimizing targets sent to local elements in the facility. This work delves into dynamic real-time optimization (RTO) of PI-controlled and MPC-controlled plants, coupled with state and parameter estimation to achieve these objectives.

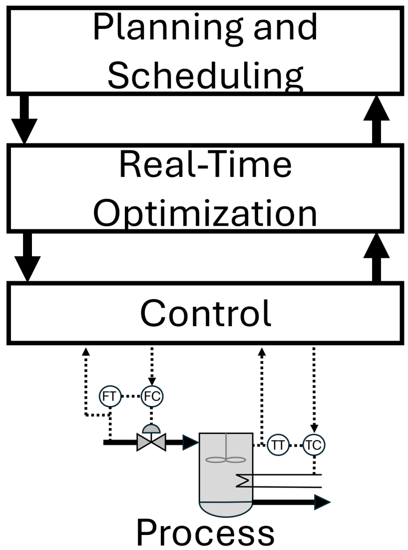

Figure 1.1: Multi-layered Control Strategy

Dynamic RTO (DRTO) can be run open-loop: without consider the plant response to the control actions, however if the DRTO is able to predict the effect of its set-points it will have a more comprehensive understanding as to how those set-points affect the objective function. To accomplish this, the DRTO can run a model of plant and controller dynamics within the optimizer to predict the effects of the set-points it chooses. This may include a prediction for how the controller will respond to the set-point, and the predicted controller action will be sent to the model for a predicted plant response. This work will seek to approach DRTO using closed-loop optimization, and will be giving particular focus to how the controller dynamics are included in the optimization problem. Even when control architecture can accurately control the plant, there may be times when the control layer receives a set-point from the DRTO that it cannot reach. During these times, the control layer must clip its control action to the maximum or minimum values determined by the controller's practical limitations. To avoid this scenario, constraints may be added to the MPC layer, but approximating constrained MPC in DRTO requires additional complexity in the optimization problem, and a constrained controller is not possible with a standard PI control formulation. Instead, the DRTO can either approximate the effect of input clipping or attempt to avoid it entirely. This work will employ the latter choice by adding the maximum and minimum controller values to the DRTO optimization problem. Plant-model mismatch can come from many sources: noise, disturbances, poor model choice or linearization of a highly nonlinear process. The control and optimization layers are affected by these disturbances, and there are various approaches to dealing with these inaccurate values. While bias updating would be considered standard practice, its method simply applies a correcting "bias" term to compensate for a poor model or inaccurate measurements. Even when working successfully, bias updating cannot predict the true value of the states, let alone any parameters that are not known to the control architecture. The Kalman filter can estimate these states as well as unknown parameters, and offers a promising alternative to bias updating in closed-loop DRTO. This work will explore Kalman filter estimation as a method of dealing with disturbances, noise, and the plant-model mismatch caused by input saturation. It will also explore the extended Kalman filter as a means of estimating the states and parameters of nonlinear plants. The purpose of this research is to test the implementation of the Kalman filter in a closed-loop dynamic RTO (CL-DRTO) setting using various forms of control and testing the performance under input saturation. This will be tested using a CSTR case study, and will be compared to alternative methods and a full state feedback baseline to determine its merit.

1.2 Main Contributions

In this work, the main contributions are as follows:

1. CL-DRTO model with integrated controller equations. CL-DRTO by nature

has a built in model that approximates the plant response to set-points chosen by the DRTO. This improves DRTO control, however typically this is done by predicting the controller response to the set-point in a separate equation. This work demonstrates a method for incorporating controller dynamics directly into the DRTO model, a method that allows the controller dynamics to be linearized and discretized along with the model. While ideally this method would be equivalent to calculating the controller dynamics in an auxiliary function, it provides the opportunity to include controller dynamics into the Kalman filter's prediction step: a novel technique that gives the Kalman filter more information in its predictions.

2. Kalman filter estimation with CL-DRTO and PI control under input saturation. This work demonstrates a method for using Kalman filter estimation in a CL-DRTO setup with PI control. The PI controller dynamics are directly incorporated into the predictions made by the estimator and the CL-DRTO. By adding the input bounds from the PI layer to the DRTO layer's constraints, this work shows that the effects of input saturation on estimator performance can be significantly reduced without using a constrained control layer such as MPC. In this work, the mismatch arising while the estimator converges will specifically be parametric plant-model mismatch, though Chapter 4 will explore other sources of mismatch. Under these conditions, the CL-DRTO is able to perform well even when given infeasible targets or placed under heavy noise.

3. Extended Kalman filter to help CL-DRTO control nonlinear systems. This work proposes a control architecture for controlling nonlinear systems. This work demonstrates how the extended Kalman filter can be used along with nonlinear models in the optimization and control layers to control a nonlinear system. This contribution expands the scope of the strategy proposed in Section 3.2 by offering a version usable in nonlinear systems.

4. Kalman filter estimation with CL-DRTO and MPC control under input saturation. MPC offers a more sophisticated form of control, and this work also

contributes an architecture for implementing the Kalman filter in MPC controlled systems. This work also demonstrates that by adding the input bounds to the DRTO optimization problem it is possible to obey controller constraints without using a constrained MPC layer. The MPC and Kalman filter will attempt to handle structural plant-model mismatch arising from various sources described in Chapter 4 as well as the parametric plant-model mismatch studied in Chapter 3. The proposed structure also demonstrates that by adding controller constraints to the DRTO, satisfactory control can be achieved without incorporating input saturation into the DRTO.

1.3 Thesis Summary

Chapter 2: Literature Review. This chapter explores the field of Real-Time Optimization and the advancements made to improve the economic and safety performance of RTO control structures. There are several challenges with RTO, and this chapter discusses some of those challenges, such as the need for steady-state conditions and difficulty during transient plant dynamics. State and parameter estimation has been proposed as a method of improving RTO during these conditions and even improving the performance of dynamic RTO, although estimators have not been tested on DRTO that incorporates controller dynamics (CL-DRTO). Additionally, there are significant issues with estimators when plant-model mismatch occurs, and research is needed to study the effect of various controllers coupled with CL-DRTO and state and parameter estimation.

Chapter 3: Kalman filter estimation with CL-DRTO and PI control layers under Input Saturation This chapter studies Kalman filter estimation coupled with CL-DRTO and PI control layers to control a multi-input multi-output CSTR under input saturation. The control structure optimizes operation of the CSTR under economic and target tracking objectives. The proposed structure integrates the controller dynamics directly into the model used in the DRTO layer and estimator, giving the estimator more informed predictions. This chapter will induce input saturation by using infeasible targets (in the target tracking objective) or by severely limiting the input clipping bounds on the controller (in the economic objective). The Kalman filter must estimate an unmeasured efficiency parameter despite significant input saturation. In Chapter 3, the input saturation directly affects the estimator which models the controller dynamics without accounting for input saturation. However, in Chapter 4, the estimator does not include controller dynamics, and the estimator performance is instead affected indirectly by the mismatch within the DRTO and MPC layers. To mitigate this effect, the DRTO predicts the PI layer's response to DRTO set-points and constrains them between the plant input bounds. By avoiding input saturation in this way, plant-model mismatch is significantly reduced. This chapter compares the proposed approach to the common approach to plant-model mismatch, bias updating, as well as a baseline of full state feedback. A nonlinear architecture is also proposed for controlling a nonlinear plant, including the extended Kalman filter, a nonlinear variation of the standard Kalman filter. This chapter also studies how the proposed approach behaves under significant noise compared to other techniques.

Chapter 4: State and Parameter Estimation in closed-loop dynamic RTO of MPC-controlled plants This chapter explores Kalman filter estimation and its interaction with another controller type: model predictive control (MPC). This chapter's architecture also uses CL-DRTO, however the DRTO predicts controller dynamics using an auxiliary function instead of directly integrating them into the model. An unconstrained MPC formulation is run on the same CSTR as Chapter 3, optimizing the system under economic and target tracking objectives. Input clipping is minimized by predicting the MPC controller within the DRTO and constraining the actions within the input bounds. MPC controllers have their own built-in disturbance term for handling plant-model mismatch, and testing how the architecture handles disturbances is studied under several disturbance types.

Chapter 5: Conclusions and Recommendations This chapter consolidates the conclusions from Chapters 3 and 4, noting significant improvements in the performance of Kalman filter estimation with CL-DRTO. In both chapters the DRTO was able to significantly minimize input clipping by constraining the predicted controller behavior within the DRTO. In the PI controlled system, the Kalman filter and CL-DRTO showed excellent performance in economic and set-point tracking, especially under conditions of input saturation and increased measurement noise. The extended Kalman filter was also able to control the nonlinear plant effectively, even under input saturation. This chapter also suggests applying the architecture to constrained controllers such as MPC or approximating input clipping within the DRTO. In the MPC controlled system, the Kalman filter effectively adhered to constraints and handled mismatch and input saturation well. The MPC based architecture also demonstrated an ability to control a nonlinear plant with linear models and estimators, suggesting a potential for scalability to larger or more complex systems. The findings presented in this chapter indicate promising potential for Kalman filter estimation coupled with CL-DRTO, and suggest that even under noise, input saturation, nonlinearity, or linearization mismatch, the Kalman filter can perform well in both PI and MPC controlled systems.

1.4 Relations to Previous Work

This work employs multilayered control, and the components comprising these layers have been studied in different settings elsewhere in literature. For example, Dering and Swartz [2], [3], and MacKinnon [4] have employed CL-DRTO, and several works testing CL-DRTO also used predicted control actions to constrain the DRTO as this study does [5] [6] [7] [8] [9]. However, none of these studies employed any form of estimator, they each used bias updating to send feedback to the DRTO. The Kalman filter has also been studied in literature [10] [11], however these studies did not use CL-DRTO, nor did they attempt to constrain controller behavior from within the optimizer. This work seeks to offer a unique combination of these elements by combining CL-DRTO with Kalman filter estimation and using input constraints in the optimizer to avoid input saturation. This work also embeds the PI controller dynamics into the nonlinear model, allowing the controller dynamics to be included in the Kalman filter: a feature not studied anywhere in literature to date. By studying the effects of noise and nonlinearity this work offers a material contribution to existing literature on control theory.

References

[1] T. E. Marlin and A. N. Hrymak. "Real-time operations optimization of continuous processes". In: *AIChE Symposium Series*. Vol. 93. 316. New York, NY: American Institute of Chemical Engineers, 1971-c2002. 1997, pp. 156–164.

[2] D. Dering and C. L. E. Swartz. "Dynamic Real-Time Scheduling and Control Under Uncertainty". In: *2022 AIChE Annual Meeting*. AIChE. 2022.

[3] D. Dering and C. L. E. Swartz. "A Closed-Loop Integrated Scheduling and Control Formulation for Processes Controlled By Distributed MPC Systems". In: *2023 AIChE Annual Meeting*. AIChE. 2023.

[4] L. A. MacKinnon. "Closed-Loop Prediction for Robust and Stabilizing Optimization and Control". Ph.D. Thesis. Hamilton, ON: McMaster University, 2023.

[5] M. Z. Jamaludin and C. L. E. Swartz. "Approximation of closed-loop prediction for dynamic real-time optimization calculations". In: *Computers & Chemical Engineering* 103 (2017), pp. 23–38.

[6] H. Li and C. L. E. Swartz. "Approximation techniques for dynamic real-time optimization (DRTO) of distributed MPC systems". In: *Computers & Chemical Engineering* 118 (2018), pp. 195–209.

[7] H. Li and C. L. E. Swartz. "Dynamic real-time optimization of distributed MPC systems using rigorous closed-loop prediction". In: *Computers & Chemical Engineering* 122 (2019), pp. 356–371.

[8] J. E. Remigio and C. L. E. Swartz. "Production scheduling in dynamic real-time optimization with closed-loop prediction". In: *Journal of Process Control* 89 (2020), pp. 95–107.

[9] M. Z. Jamaludin and C. L. E. Swartz. "Dynamic real-time optimization with closed-loop prediction". In: *AIChE Journal* 63.9 (2017), pp. 3896–3911.

[10] J. O. Matias and G. A. Le Roux. "Real-time Optimization with persistent parameter adaptation using online parameter estimation". In: *Journal of Process Control* 68 (2018), pp. 195–204.

[11] J. Matias, J. P. Oliveira, G. A. Le Roux, and J. Jäschke. "Steady-state real-time optimization using transient measurements on an experimental rig". In: *Journal of Process Control* 115 (2022), pp. 181–196.

Chapter 2

Literature Review

2.1 Real-Time Optimization

Real-Time Optimization (RTO) has been thoroughly studied both in practical application and academic research, playing a pivotal role in meeting a plant's economic and safety requirements [1]. RTO continuously monitors and updates process conditions using a feedback loop, offering a real-time response to changing demands and constraints [2] [3]. RTO can correct for disturbances and uncertainty using model biases, and a detailed approach on how these bias terms function can be found in Forbes and Marlin [4]. In self-optimizing control (SOC), combinations of measurements are used to compute self-optimizing variables, which are maintained at optimal set-points to minimize economic loss under disturbances, as demonstrated by Skogestad [5]. In this case, the model is not updated, but instead the self-optimizing variables are controlled to meet this objective given an expected disturbance range. To update these manipulated variables, RTO must have steady-state conditions, a significant drawback of the methodology [3].

Despite these benefits, RTO comes with significant limitations. RTO control requires steady state conditions to execute the optimization problem. Determining this steady state is difficult, but the primary issue is RTO's inability to update during transient operation. RTO is especially ill-suited in systems with constant disturbances, slow dynamics, or frequent changes, resulting in poor system operation [6]. Most integrated plants have transient periods lasting up to multiple days due to transportation delays, intermediate storage, and recycle loops [7]. Embedding the economic objective within the model predictive controller (MPC) layer can address this issue, and creates an economic MPC (EMPC) layer. A detailed explanation of EMPC systems can be found in Ellis et al. [8].

2.2 DRTO

Another method of dealing with the steady-state drawback to RTO is to include the system dynamics in the upper RTO layer, creating a dynamic RTO (DRTO) layer [9]. There have been several advancements to this technique, and several are listed here. Würth et al. [10] proposed a method of dealing with dynamics at different speeds. Würth's two-layer approach used a higher level DRTO operating at a lower frequency to capture slow process dynamics, and a lower level neighboring-extremal (NE) controller to track the trajectory and account for faster dynamics. Tosukhowong et al. [7] also found it advantageous to operate the DRTO layer at a reduced frequency, and developed a method of choosing a reduced-order model for a given DRTO frequency and discussed its implementation in a system of controllers. Biegler et al. [11] noted that unknown inputs and model parameters affect steady state values in a significant way, and suggested using on-line state and parameter estimation where the states and parameters are updated for each optimization problem. Pontes et al. [12] suggested a method of DRTO that could be easily used on processes with rapid transitions. Their objective function prioritized economics and used both dynamic and static stages to handle rapid transitions. The control layer that executes the set-points from the optimizer can take several forms: PI control and MPC control are two popular examples of control strategies. This work will look at both and evaluate how closed-loop DRTO coupled with state and parameter estimation performs in each.

2.3 CL-DRTO with PI Control

Typical two-layer DRTO and PI models use an open-loop dynamic prediction for the DRTO, however Jamaludin and Swartz [6] demonstrated improved economics using closed-loop DRTO (CL-DRTO), and this model will be considered here. A closed loop prediction can be computationally expensive, and literature has instead developed methods of approximating the closed-loop prediction. A detailed explanation of the variables and equations governing rigorous CL-DRTO structure is given in Sections 3.1.4 and 4.1.5. PI control offers a simple approach to reaching setpoints from the DRTO, however the control laws are reactionary; they do not consider future plant dynamics and cannot optimize an objective function within the controller subproblem. These drawbacks make model predictive control a desirable improvement to control architecture.

2.4 CL-DRTO with MPC Control

Model predictive control is frequently used industrially, and significant research has focused on improving its performance. Qin and Badgwell [13] details many of these MPC approaches that have been used commercially. Some notable advancements in MPC approaches are given here. Bemporad and Morari [14] integrated logic rules into MPC using mixed integer programming, and Biegler [15] implemented discrete-time optimization using simultaneous collocation. These advancements give greater flexibility in how DRTO can approach optimization. Some innovative techniques have been developed for applying MPC such as Bemporad et al. [16] including explicit state-feedback solutions for quadratic control or Amrit et al. [17] directly optimizing process economics to improve profitability. Several advancements have also been made in MPC's ability to handle nonlinearity; Rawlings et al. [18] improved MPC performance in complex nonlinear systems, and Zavala and Biegler [19] further improved nonlinear MPC with sensitivity-based predictions to reduce their computational expense. These are some examples of the innovation and interest invested in improving MPC control, and these advancements highlight MPC's growing adaptability and efficiency in diverse and complex industrial applications.

2.5 State and Parameter Estimation

Dynamic state and parameter estimation can also be used in RTO as a method of overcoming the steady-state drawback to RTO [20]. Matias et al. [20] mentioned that steady-state detection methods can be easily misled by complex issues such as slow signals out of phase with measurements, large processes, or high frequency disturbances. Failing to properly detect steady-states may result in inaccurate parameter estimation if a steady-state model updating strategy is used. As a consequence, the optimality of the solution computed by RTO can be affected. To help models improve the accuracy of their states and parameters, dynamic state and parameter estimation can be used in RTO and has also been included in implementations of dynamic RTO, such as in Matias et al. [20]; however Matias et al. [20] only demonstrated this setup on classical DRTO; controller dynamics were not taken into account.

While adding state and parameter estimation to DRTO structures improves model accuracy, it can lead to plant-model mismatch if the system undergoes input saturation. A method that mitigates this effect could improve overall performance and make the Kalman filter less vulnerable. This unfavorable behavior is due to incompatibility between Boolean logic at the controller level and optimizers at the DRTO level. This problem is a significant obstacle to optimizers with estimated states and parameters, as any constraints at the DRTO level would be based on estimated values and may not accurately predict plant behavior. In a highly constrained system, this can lead to poor state and parameter estimation and subsequent inefficiencies in the economic objective of the plant. Note that even with perfect state and parameter estimation, if the DRTO is unable to approximate or avoid input saturation the estimated values will be wasted, used by the DRTO to send inaccurate set-points. As input saturation is a common occurrence in practice, it is of particular interest to find a method to deal with input saturation for DRTO optimizers using estimated information.

The Kalman filter has not yet been tested on CL-DRTO combined with MPC setups, let alone under the effects of input saturation. This is particularly interesting for MPC controllers with a disturbance model, as the disturbance term must converge while the Kalman filter parameters converge, and the interaction may impact control. The Kalman filter has also not been studied under the effect of input saturation, a situation that may derail estimators and requires mixed integer optimization architecture. Input saturation will directly affect the estimator when controller dynamics are included in the model, as in Chapter 3, and may also affect the estimator indirectly when the mismatch causes poor performance in the MPC and DRTO layers the estimator interacts with, as in Chapter 4. This paper will be studying all these effects, using a Kalman filter estimator on a DRTO layer and MPC layer under significant input saturation. This work will evaluate the performance of the estimator as the control structure using it achieves economic and target tracking objectives, and will test the robustness of the structure by introducing disturbances and attempting to control a nonlinear plant with linear architecture.

References

[1] V. Adetola and M. Guay. "Integration of real-time optimization and model predictive control". In: *Journal of Process Control* 20.2 (2010), pp. 125–133.

[2] T. E. Marlin and A. N. Hrymak. "Real-time operations optimization of continuous processes". In: *AIChE Symposium Series*. Vol. 93. 316. New York, NY: American Institute of Chemical Engineers, 1971-c2002. 1997, pp. 156–164.

[3] M. L. Darby, M. Nikolaou, J. Jones, and D. Nicholson. "RTO: An overview and assessment of current practice". In: *Journal of Process control* 21.6 (2011), pp. 874–884.

[4] J. F. Forbes and T. E. Marlin. "Model accuracy for economic optimizing controllers: The bias update case". In: *Industrial & engineering chemistry research* 33.8 (1994), pp. 1919–1929.

[5] S. Skogestad. "Self-optimizing control: From key performance indicators to control of biological systems". In: *Computer Aided Chemical Engineering*. Vol. 15. Elsevier, 2003, pp. 340–345.

[6] M. Z. Jamaludin and C. L. E. Swartz. "Approximation of closed-loop prediction for dynamic real-time optimization calculations". In: *Computers & Chemical Engineering* 103 (2017), pp. 23–38.

[7] T. Tosukhowong, J. M. Lee, J. H. Lee, and J. Lu. "An introduction to a dynamic plant-wide optimization strategy for an integrated plant". In: *Computers & Chemical Engineering* 29.1 (2004), pp. 199–208.

[8] M. Ellis, H. Durand, and P. D. Christofides. "A tutorial review of economic model predictive control methods". In: *Journal of Process Control* 24.8 (2014), pp. 1156–1178.

[9] H. Li and C. L. E. Swartz. "Dynamic real-time optimization of distributed MPC systems using rigorous closed-loop prediction". In: *Computers & Chemical Engineering* 122 (2019), pp. 356–371.

[10] L. Würth, R. Hannemann, and W. Marquardt. "A two-layer architecture for economically optimal process control and operation". In: *Journal of Process Control* 21.3 (2011), pp. 311–321.

[11] L. T. Biegler, X Yang, and G. Fischer. "Advances in sensitivity-based nonlinear model predictive control and dynamic real-time optimization". In: *Journal of Process Control* 30 (2015), pp. 104–116.

[12] K. V. Pontes, I. J. Wolf, M. Embirucu, and W. Marquardt. "Dynamic real-time optimization of industrial polymerization processes with fast dynamics". In: *Industrial & Engineering Chemistry Research* 54.47 (2015), pp. 11881–11893.

[13] S. J. Qin and T. A. Badgwell. "A survey of industrial model predictive control technology". In: *Control engineering practice* 11.7 (2003), pp. 733–764.

[14] A. Bemporad and M. Morari. "Control of systems integrating logic, dynamics, and constraints". In: *Automatica* 35.3 (1999), pp. 407–427.

[15] L. T. Biegler. "Advanced optimization strategies for integrated dynamic process operations". In: *Computers & Chemical Engineering* 114 (2018). FOCAPO/CPC 2017, pp. 3–13.

[16] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. "The explicit linear quadratic regulator for constrained systems". In: *Automatica* 38.1 (2002), pp. 3–20.

[17] R. Amrit, J. B. Rawlings, and L. T. Biegler. "Optimizing process economics online using model predictive control". In: *Computers & Chemical Engineering* 58 (2013), pp. 334–343.

[18] J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model predictive control: theory, computation, and design*. Vol. 2. Nob Hill Publishing Madison, WI, 2017.

[19] V. M. Zavala and L. T. Biegler. "The advanced-step NMPC controller: Optimality, stability and robustness". In: *Automatica* 45.1 (2009), pp. 86–93.

[20] J. Matias, J. P. Oliveira, G. A. Le Roux, and J. Jäschke. “Steady-state real-time optimization using transient measurements on an experimental rig”. In: *Journal of Process Control* 115 (2022), pp. 181–196.

Chapter 3

State and Parameter Estimation in CL-DRTO and PI-Controlled Plants

3.1 Preliminaries

This section will explore the equations and concepts comprising the control architecture of this study. After describing the system, the equations governing the CL-DRTO used in this chapter will be demonstrated. The Kalman filter and the observability of the system will also be demonstrated, alongside an alternative approach to handling disturbances: bias updating. After describing these layers in detail, this section will propose adaptations to the control architecture for controlling nonlinear systems, namely the extended Kalman filter. Input saturation plays a significant role in how these architectures perform, and this section will close with an explanation of these effects and how they affect the system.

3.1.1 Background

Linearized control architecture offers a less computationally expensive means of control than nonlinear architecture. Equation Set 3.1 gives a general representation of a nonlinear plant.

$$\begin{aligned} \dot{\boldsymbol{x}}_{nl} &= \frac{d\boldsymbol{x}_{nl}}{dt} = f(\boldsymbol{x}_{nl}(t), \boldsymbol{u}_{nl}(t)) \\ \boldsymbol{y}_{nl}(t) &= g(\boldsymbol{x}_{nl}(t)) \end{aligned} \tag{3.1}$$

Here $\boldsymbol{x}_{nl}(t)$ and $\boldsymbol{u}_{nl}(t)$ represent the current state and input values at time t, respectively. These are used to calculate the change in the states ($\dot{\boldsymbol{x}}_{nl}$) and used to map the states to their measured output vector ($\boldsymbol{y}_{nl}(t)$). The vectors used in Equation Set 3.1 are given the subscript "*nl*" to indicate the nonlinear vectors are not in deviation form. This was done for simplicity of notation, as the remainder of this work will deal with variables in deviation form, this will be treated as the default form of the vectors. Based on these values, the nonlinear plant can determine the rate at which the states change ($\dot{\boldsymbol{x}}_{nl}$). In many cases a sufficiently accurate approximation of the nonlinear process can be achieved with linearization, though if the process is significantly nonlinear or deviates from the nominal value of the states enough this approximation can lead to poor control. The nonlinear model shown in Equation 3.1 would take the form of the linearized state space model in Equation Set 3.2.

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x}(t) + B\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= C\boldsymbol{x}(t) \end{aligned} \tag{3.2}$$

where $\dot{\boldsymbol{x}}$ and $\boldsymbol{y}$ represent the derivative of the states and measurements, respectively. A and B are matrices that describe the relationship between the states and inputs, and C is used to select states that correspond to the measurement of the system in practice.

After demonstrating proficiency with linear systems, the strategy proposed in this chapter will also be tested on a nonlinear plant and model as shown in Equation 3.1. However as linearized systems are more common in practice, the primary focus of this chapter and its formulations will be on linear systems.

3.1.2 Deviation Form

When the state space model matrices are linearized, a steady state condition is chosen for the states and manipulated variables. The act of linearization places the linearized model in deviation form, and all state and input vectors must be adjusted to conform with deviation form as well. This is accomplished by subtracting the steady state values used in linearization from the current values of those states in their vector. For example, state vector $\boldsymbol{x}(t)$ composed of n states and input vector $\boldsymbol{u}(t)$ composed of m

inputs are set in deviation form as in Equation Set 3.3:

$$\boldsymbol{x}(t) = \begin{bmatrix} x'_1 \\ x'_2 \\ \vdots \\ x'_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} - \begin{bmatrix} x_{1,ss} \\ x_{2,ss} \\ \vdots \\ x_{n,ss} \end{bmatrix} \boldsymbol{u}(t) = \begin{bmatrix} u'_1 \\ u'_2 \\ \vdots \\ u'_m \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} - \begin{bmatrix} u_{1,ss} \\ u_{2,ss} \\ \vdots \\ u_{m,ss} \end{bmatrix} \tag{3.3}$$

Typically, vectors in deviation form are denoted with an apostrophe, however nearly all vectors in this work will be in deviation form, so to reduce unnecessary notation it will be assumed that any vectors that interact with state space model matrices are in deviation variable form unless stated otherwise.

3.1.3 System Description

Unlike other research, this work seeks to integrate the behavior of the PI-level into the DRTO model of the plant to account for the impact of controller dynamics on the response, resulting in a model that is a function of the set-point. This poses a challenge due to input saturation issues described later, but as this chapter mitigates those effects, the PI-level can be safely integrated into the model, though input saturation will not be directly integrated in this way. The combined model can now predict the plant behavior and PI behavior together. This is first done by considering the PI-level's relationship between the set-points it receives ($\boldsymbol{y}_{sp,nl}$) and the control actions it chooses as a result ($\boldsymbol{u}_{nl}(t)$). This relationship can be simply described by Equation Set 3.4.

$$\begin{aligned} \boldsymbol{u}_{nl}(t) &= \boldsymbol{u}_{\text{bias}} + K_c \boldsymbol{e}_{nl}(t) + \frac{K_c}{\tau_I} \boldsymbol{I}_{e,nl}(t) \\ \boldsymbol{e}(t) &= \boldsymbol{y}_{\text{sp,nl}}(t) - \boldsymbol{y}_{nl}(t) \end{aligned} \tag{3.4}$$

Where $\boldsymbol{u}_{bias}$ is a vector of input values at steady state conditions, $\boldsymbol{e}_{nl}(t)$ is the error of the system measured as the distance between the set-point ($\boldsymbol{y}_{sp,nl}(t)$) and the measured states ($\boldsymbol{y}_{nl}(t)$). $\boldsymbol{I}_{e,nl}(t)$ is the integral of the error accumulated by the system, which is initially 0. In this nonlinear equation set, $\boldsymbol{I}_{e,nl}(t)$ is calculated using the following ODE:

$$\frac{d\boldsymbol{I}_{e,nl}}{dt} = \boldsymbol{e}_{nl}(t) \tag{3.5}$$

The ODE from Equation 3.5 will be added to the state vector to calculate the integral terms alongside the other states. The gain (K_c) and time constant (τ_I) of Equation Set 3.4 are tuned to control the behavior of the controller. Anti-reset windup (ARW) was employed in the PI layer, but due to the computational expense of boolean logic in optimization, the prediction of the PI layer within the DRTO will not take anti-reset windup into account. Substituting Equation 3.4 into Equation 3.1, we can instead describe $\dot{\boldsymbol{x}}_{nl}$ as a function of the current states, set-points of the outputs, and the integral of the error between them. This new model includes the dynamics of the PI-layer in its model, and Equation Set 3.6 gives a generic example of the nonlinear and linearized versions of this process.

$$\dot{\boldsymbol{x}}_{nl} = h(\boldsymbol{x}_{nl}(t), \boldsymbol{y}_{sp,nl}(t), \boldsymbol{I}_{e,nl}(t))$$

The derivative of the states ($\dot{\boldsymbol{x}}_{nl}$) contains both states and the integral of error for the states involved in PI control. The linearization of $\dot{\boldsymbol{x}}_{nl}$ yields Equation Set 3.6.

$$\begin{aligned} \begin{bmatrix} \frac{d\boldsymbol{x}}{dt} \\ \frac{d\boldsymbol{I}_e}{dt} \end{bmatrix} &= \begin{bmatrix} A\boldsymbol{x}(t) + B\boldsymbol{y}_{\text{sp}}(t) \\ \boldsymbol{y}_{\text{sp}}(t) - \boldsymbol{y}(t) \end{bmatrix} \\ \boldsymbol{y}(t) &= C\boldsymbol{x}(t) \end{aligned} \tag{3.6}$$

Note that in Equation Set 3.6, the generic nonlinear model given in Equation Set 3.2 will be expanded to include integral terms as additional rows from this point onward. This work will be concerned with parameter estimation, so the model will be a function of the changing parameters as well. To accomplish this, an additional term, $D_{\theta,m}\boldsymbol{\theta}(t)$, is included to represent the effect of changing parameters on the states of the model and plant. The aforementioned concepts are combined to create the continuous model used for this work, given in Equation Set 3.7.

$$\begin{aligned} \dot{\boldsymbol{x}}_m &= A_m \boldsymbol{x}_m(t) + B_m \boldsymbol{y}_{sp}(t) + D_{\theta,m}\boldsymbol{\theta}(t) \\ \boldsymbol{y}_m &= C_m \boldsymbol{x}_m(t) \end{aligned} \tag{3.7}$$

In Equation Set 3.7, $\boldsymbol{y}_{sp}(t)$ represents the set-points sent from the DRTO layer to the PI controller. $\dot{\boldsymbol{x}}_m$ and $\boldsymbol{y}_m$ are the states and measurements from the model, respectively. A_m, B_m and C_m are the state space model matrices of the model. An estimator will be used to predict the parameters of the plant, so it is desirable to describe the states as a function of the parameter value as well. The matrix $D_{\theta,m}$ represents how the parameters affect the states, and $\boldsymbol{\theta}(t)$ represents the parameters as a function of time. This value will be constant for the plant as the parameter values will be fixed for the case study, but due to the estimator described in Section 3.1.6 this value will change in the model, so it is included as a variable here. Using Equation Set 3.7,, the plant response due to the DRTO set-point can be simulated for the optimizer. A detailed explanation of the linearization that enables matrices A_m, B_m and C_m to describe the PI control and the model is given in the proposed solution. It should be noted that although the DRTO will incorporate the PI layer into its model, the plant is still distinct from the PI level, and as such the plant will be a function of controller action ($\boldsymbol{u}(t)$). The change in plant states ($\dot{\boldsymbol{x}}_p$) and measurements ($\boldsymbol{y}_p$) can be described as follows:

$$\begin{aligned} \dot{\boldsymbol{x}}_p &= A_p \boldsymbol{x}_p(t) + B_p \boldsymbol{u}(t) + D_{\theta,p}\boldsymbol{\theta} \\ \boldsymbol{y}_p &= C_p \boldsymbol{x}_p(t) \end{aligned} \tag{3.8}$$

Notice here that the parameter values $\boldsymbol{\theta}$ do not change as a function of time for the current case study, however this may not always be the case. Since the plant is distinct from the PI level, A_p, B_p, C_p and $D_{\theta,p}$ are used to describe the plant on its own. To incorporate the system into optimizers and DRTO architecture, the plant and model are discretized using the same time interval as the optimization interval (Δt_{opt}). The choice of this interval is especially important for the optimizer, and Tosukhowong et al. [1] gave an example of how to choose an appropriate Δt_{opt} for DRTO. The resulting discretized model for the DRTO is given in Equation Set 3.9 below.

$$\begin{aligned} \boldsymbol{x}_{m,(k+1)} &= A_{m,d}\boldsymbol{x}_{m,k} + B_{m,d}\boldsymbol{y}_{sp,k} + D_{\theta m,d}\boldsymbol{\theta}_k \\ \boldsymbol{y}_{m,(k+1)} &= C_{m,d}\boldsymbol{x}_{m,(k+1)} \end{aligned} \tag{3.9}$$

The PI level takes more frequent actions than the DRTO level, and this must be accounted for when discretizing the matrices. To account for these more frequent actions, the plant matrices have been discretized using $\Delta t_{\text{PI/plant}}$ which is equal to $\frac{\Delta t_{\text{opt}}}{N}$ where N is number of control actions taken between points on the DRTO horizon. Once discretized at this shorter interval, the plant model from Equation Set 3.8 becomes the new discrete form in Equation Set 3.10.

$$\begin{aligned} \boldsymbol{x}_{p,(k+1)} &= A_{p,d}\boldsymbol{x}_{p,k} + B_{p,d}\boldsymbol{y}_{sp,k} + D_{\theta p,d}\boldsymbol{\theta}_k \\ \boldsymbol{y}_{p,(k+1)} &= C_{p,d}\boldsymbol{x}_{p,(k+1)} \end{aligned} \tag{3.10}$$

3.1.4 CL-DRTO

Tosukhowong et al. [1] gave an example of DRTO implementation which is adapted for use here. Their system was a function of controller action, whereas this model is a function of the set-points. Their system also used an MPC layer that will be not be used in this study but will be integrated in the next chapter. The adapted system comprises the optimization setup in Equation Set 3.11 in addition to the model in Equation Set 3.9 and is optimized for each Δt_{opt} interval.

$$
\begin{aligned}
&\min_{Y_{\text{sp}}} \phi \\
\text{s.t. } & X_{m,\min} \leq X_{m,(k+1|k)} \leq X_{m,\max} \\
& Y_{\text{sp},min} \leq Y_{\text{sp}} \leq Y_{\text{sp},max} \\
& U_{\min} \leq U \leq U_{\max} \\
& Y_{\text{sp}} = \begin{bmatrix} \boldsymbol{y}_{\text{sp},k}^T & \boldsymbol{y}_{\text{sp},(k+1)}^T & \cdots & \boldsymbol{y}_{\text{sp},(k+M-1)}^T \end{bmatrix}^T \\
& X_{m,(k+1|k)} = \begin{bmatrix} \boldsymbol{x}_{m,(k+1|k)}^T & \boldsymbol{x}_{m,(k+2|k)}^T & \cdots & \boldsymbol{x}_{m,(k+P|k)}^T \end{bmatrix}^T
\end{aligned}
\tag{3.11}
$$

Here ϕ is the objective function of the DRTO, Y_{sp} is a vector of set-points to send to the PI layer starting at time step k over the control horizon of length M, subject to $Y_{sp,min}$ and $Y_{sp,max}$, the respective lower and upper limits on that vector. U represents a vector of the predicted control actions from the PI layer as a result of the chosen set-points. U may not directly appear in the objective function, but it's values must be constrained to ensure the optimal set-point is feasible for the PI layer. The notation $(k+n|k)$ indicates the variable was calculated for time step $k+n$, and the most recent measurements came from time step k. These measurements are obtained from the state-to-output mapping given in Equation 3.9. X_m represents a vector of predicted states starting at time step k and continuing over prediction horizon P. The vector Y_{sp} is then used by the model to determine $X_{m,(k+1|k)}$, the predicted states resulting from those set-points. The composite vector Y_{sp} is an array of optimized set-points which can be sent directly to the PI layer. Often states or parameters are not available as measurements for the model, or the measurements contain significant levels of noise. In this case, it is desirable to correct for resulting disturbance in some way. In CL-DRTO the current standard practice of dealing with these issues is bias updating.

3.1.5 Bias Updating

The bias term is simply the difference between the value of the measured states and the value of those states predicted by the model. After measuring the plant and using the DRTO model to predict it's value, the bias is calculated for the following time step: $k+1$, as shown in Equation 3.12.

$$\boldsymbol{\beta}_{\text{bias},(k+1)} = \boldsymbol{y}_{p,(k+1)} - \boldsymbol{y}_{m,(k+1)} \tag{3.12}$$

Bias updating deals with plant-model mismatch by adjusting the constraints and objective function of the optimizer to better agree with plant measurements.

$$\begin{aligned} &\min_{Y_{\text{sp}}} \phi_\beta \\ \text{s.t. } & X_{m,\beta,\min} \leq X_{m,(k+1|k)} \leq X_{m,\beta,\max} \\ & Y_{\text{sp},min} \leq Y_{\text{sp}} \leq Y_{\text{sp},max} \\ & U_{\min} \leq U \leq U_{\max} \\ & Y_{\text{sp}} = \begin{bmatrix} \boldsymbol{y}^T_{\text{sp},k} & \boldsymbol{y}^T_{\text{sp},(k+1)} & \cdots & \boldsymbol{y}^T_{\text{sp},(k+M-1)} \end{bmatrix}^T \\ & X_{m,(k+1|k)} = \begin{bmatrix} \boldsymbol{x}^T_{m,(k+1|k)} & \boldsymbol{x}^T_{m,(k+2|k)} & \cdots & \boldsymbol{x}^T_{m,(k+P|k)} \end{bmatrix}^T \end{aligned} \tag{3.13}$$

In the adjustment shown in Equation 3.13, ϕ_β represents the objective function where the measured states are adjusted with their own individual bias term, $\boldsymbol{\beta}$. As with Equation Set 3.11, the optimization problem for bias updating comprises of Equation Set 3.13 and model in Equation Set 3.9 after discretization. $X_{m,\beta,min}$ and $X_{m,\beta,max}$ represent the minimum and maximum constraints on the predicted plant response, with constraints on measured states adjusted by their respective bias term. By adjusting the values of the states, the bias terms apply an "incentive" to the objective function and allow a deviation in the model constraints corresponding to the degree of mismatch between the plant and model.

While bias updating corrects the behavior of the plant towards a desirable objective, it does not directly predict the values of the estimated states and unknown parameters. This prediction is a desirable feature, particularly for non-measured parameters that require maintenance when they drop below a particular value, such as the efficiency of a heater or the fouling factor of a heat exchanger. This makes estimators especially valuable for dealing with missing measurements. The Kalman filter estimator is a good candidate for this task and has already been used to approximate missing states and parameters in DRTO [2].

3.1.6 Kalman filter Estimation

The Kalman filter can estimate states and parameters, and Song and Grizzle [3] demonstrated that over time this method will converge to the true states and parameters depending on certain noise and model conditions. This technique will be adapted to CL-DRTO using the following Kalman filter algorithm, which is derived from the set-point model in Equation 3.7. In this chapter the controller dynamics are incorporated directly into the estimator model, a novel enhancement offering the Kalman filter improved accuracy. The estimator will estimate any states or parameters of interest before sending them to the DRTO layer for optimization.

The state vector used by the estimator ($\tilde{\boldsymbol{x}}_e$) expands on the state vector ($\boldsymbol{x}_m$) from Equation 3.7 by adding the estimated parameters as pseudo-states. Naturally, $\tilde{\boldsymbol{x}}$ will have n_{ps} more rows than $\boldsymbol{x}_m$; where n_{ps} is the number of pseudo-states estimated by the Kalman filter.

This estimator model will be a function of the set-points and the parameter values, shown in Equation Set 3.14.

$$\begin{aligned} \dot{\tilde{\boldsymbol{x}}}_e &= A_e \tilde{\boldsymbol{x}}_e(t) + B_e \boldsymbol{y}_{sp}(t) \\ \tilde{\boldsymbol{y}}_e &= C_e \tilde{\boldsymbol{x}}_e(t) \end{aligned} \tag{3.14}$$

Equation Set 3.14 is very similar to the model in Equation Set 3.7, and it's A_e, B_e, and C_e are identical to their counterparts in the DRTO model except for an added row or column for each pseudostate parameter. This new structure for Equation Set 3.14 is given in greater detail in Equation Set 3.15.

$$\begin{aligned} \begin{bmatrix} \dot{\boldsymbol{x}}_m \\ \dot{\boldsymbol{x}}_{ps} \end{bmatrix} &= \begin{bmatrix} A_m & A_{ps_1} \\ A_{ps_2} & A_{ps_3} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_{ps} \end{bmatrix} + \begin{bmatrix} B_m \\ B_{ps} \end{bmatrix} \boldsymbol{y}_{sp} \\ \tilde{\boldsymbol{y}}_e &= \begin{bmatrix} C_m & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_{ps} \end{bmatrix} \end{aligned} \tag{3.15}$$

Here the model from Equation Set 3.6 is expanded to include the pseudostate parameters estimated by the Kalman filter. Vectors and matrices associated with these added pseudo-states are given the subscript "ps". Together the $\dot{\boldsymbol{x}}_m$ and $\dot{\boldsymbol{x}}_{ps}$ vectors comprise the larger estimator vector $\dot{\tilde{\boldsymbol{x}}}_e$ from Equation Set 3.14. However as pseudo-states are not measured, $\tilde{\boldsymbol{y}}_e$ must be equal to $\boldsymbol{y}_m$ and the composition of the model vector remains unchanged.

As the estimated parameters are treated as pseudo-states in the estimator model, the effect of the pseudostate on the other states is described by A_{ps1}, so the $D_{\theta,e}\boldsymbol{\theta}(t)$ term used in the DRTO model is not needed.

The estimator model given in Equation Set 3.14 is then discretized using Δt_{opt}, resulting in the discretized state space model in Equation Set 3.16.

$$\begin{aligned} \tilde{\boldsymbol{x}}_{e,(k+1)} &= A_{e,d}\tilde{\boldsymbol{x}}_{e,k} + B_{e,d}\boldsymbol{y}_{sp,k} \\ \tilde{\boldsymbol{y}}_{e,(k+1)} &= C_{e,d}\tilde{\boldsymbol{x}}_{e,k} \end{aligned} \tag{3.16}$$

Equation Set 3.16 is used to estimate the states and unknown parameters. The Kalman filter algorithm from Walter et al. [4] is given in Equation Set 3.17.

$$
\begin{aligned}
&1. \quad \tilde{\boldsymbol{x}}_{e,(k+1|k)} = A_{e,d}\tilde{\boldsymbol{x}}_{e,(k|k)} + B_{e,d}\boldsymbol{y}_{sp,k} \\
&2. \quad P_{(k+1|k)} = A_{e,d}P_{(k|k)}A_{e,d}^T + V_k \\
&3. \quad K_{f,(k+1)} = P_{(k+1|k)}C_{e,d}^T\left(W_{(k+1)} + C_{e,d}P_{(k+1|k)}C_{e,d}^T\right)^{-1} \\
&4. \quad \tilde{\boldsymbol{x}}_{e,(k+1|k+1)} = \tilde{\boldsymbol{x}}_{e,(k+1|k)} + K_{f,(k+1)}\left(\boldsymbol{y}_{p,(k+1)} - C_{e,d}\tilde{\boldsymbol{x}}_{e,(k+1|k)}\right) \\
&5. \quad P_{(k+1|k+1)} = P_{(k+1|k)} - K_{f,(k+1)}C_{e,d}P_{(k+1|k)}
\end{aligned}
\tag{3.17}
$$

The equations above are solved sequentially, and the algorithm is repeated at each time step. The subscript $(g|h)$ indicates that the corresponding value(s) at time step g were calculated using data from time step h. The estimated state vector $\tilde{\boldsymbol{x}}_e$ is similar to the model state vector $\boldsymbol{x}_p$ except that pseudo-state parameters are included in $\tilde{\boldsymbol{x}}_e$. $A_{e,d}$, $B_{e,d}$ and $C_{e,d}$ are the linearization matrices of the estimator, discretized using Δt_{opt}. The estimator has its own linearization matrices to accommodate the pseudo-states it uses in the Kalman filter. P and K_f represent the noise covariance of the estimates and gain of the Kalman filter, respectively. These matrices converge on their stationary values as Equation Set 3.17, iterates. The initial value of the covariance matrix, $P_{(0|0)}$ is an initial guess for the algorithm. W and V represent the measurement noise and process noise of the system, respectively. The V matrix needed to be extended to include the covariance of the pseudostate estimates. This gives the pseudostate the ability to change values even if its dynamics are static. The values comprising W and V are chosen to represent the system's noise sources and are important to gurantee good performance from the estimator. In Step 1 the Kalman filter predicts the future state of the system using a state space model. Similar to the DRTO model these states are a function of the set-point. The covariance of the estimates $P_{(k+1|k)}$ is then calculated in Step 2, and the Kalman filter gain $K_{f,(k+1)}$ is calculated in Step 3. The values from the first 3 steps are then used to update the states and parameters in step 4. The new Kalman filter gain is also used to update the covariance matrix in step 5. This process returns a vector of estimated states and an improved gain and covariance matrix for the next iteration of the Kalman filter.

In the context of CL-DRTO, the controller dynamics can have a significant impact on the predictions of how the plant and controller respond to set-points. Traditionally the control architecture using the Kalman filter estimator does not take into account the dynamics of the control layer in the DRTO constraints. Without a means of preventing or approximating input-clipping this setup may perform poorly in highly-constrained systems. An example of this traditional Kalman filter setup can be found in Matias and Le Roux [5] applied to RTO control. This traditional approach does not use controller constraints (U_{min} and U_{max} from Equation Set 3.11) in the optimization layer. The proposed solution will be compared against this unconstrained implementation to determine its merit.

3.1.7 Observability

Each system has limits on its observability. While the controllability of the system is a function of state space model matrices A_e and B_e, the observability is a function of matrices A_e and C_e. This is intuitive as the matrix A_e represents state dynamics and C_e represents which of these states are visible to the estimator. The system in Equation Set 3.16 is considered observable if it is possible to estimate any states in $\tilde{\boldsymbol{x}}_e$ given the past trajectories of the measurements in $\tilde{\boldsymbol{y}}_e$. There are many types of tests to determine observability. Here the observability matrix O_e is used to determine observability

$$O_e = \begin{bmatrix} C_e \\ C_e A_e \\ C_e A_e^2 \\ \vdots \\ C_e A_e^{n-1} \end{bmatrix} \tag{3.18}$$

where n is the dimension of the full state, including any pseudo-states estimated by the Kalman filter. In an estimator with 4 states and 1 estimated parameter, n would be 5. If the rows of this matrix span $\mathbb{R}^{n_x+n_{ps}}$ dimensions, any state vector can be estimated from the past trajectories of $\tilde{\boldsymbol{y}}_e$ and the system is observable [6].

3.1.8 Nonlinear Adaptations to Control Architecture

Julier and Uhlmann [7] discussed the necessary conditions for an estimation to be considered consistent; a property described in part by the mean and covariance of that estimate. While Julier and Uhlmann [7] demonstrated the mean and covariance can be recursively obtained in linear systems using the Kalman filter, they noted a new method would be needed for nonlinear systems. Approximating these terms using a truncated Taylor expansion leads to the extended Kalman filter. This technique has been used by Julier and Uhlmann [7], and Julier and Uhlmann [7] warned that negelcting these terms may lead to inadequate estimator behavior in nonlinear systems. This extended Kalman filter (EKF) technique has been used in the past to improve the performance of KF estimation [4] on nonlinear systems. The extended Kalman filter approximates this missing nonlinearity by updating the linear state space matrix A_e each iteration using the nonlinear model equations.

Equipped with the EKF, the estimator can estimate the states and parameters of a nonlinear plant with improved accuracy by obtaining a better approximation of the mean and covariance of the estimate when compared to the linear Kalman filter. Using this adaptation, this chapter will demonstrate the proposed strategy's performance in nonlinear systems. To do this several adaptations are made to the above setup. As mentioned in Section 3.1.1, the nonlinear comparison will use a nonlinear plant shown generically in Equation 3.1. The DRTO will use Equation Set 3.4 to predict the PI-level behavior based on the DRTO set-points and then use these PI-level actions as inputs to a nonlinear model. This is slightly different than the linearized setup which incorporates PI-level behavior directly into the model.

Using Equation 3.1 and Equation Set 3.4, the DRTO will optimize a nonlinear model while constraining the actions taken at the PI-level.

The CL-DRTO used in the nonlinear study will use a single-shooting NL optimization technique for integrating the ODE system to obtain an NLP suitable for the optimization solver. The gain and time constant of the PI equation also needed to be retuned to control the nonlinear plant.

These changes serve to create a fully nonlinear system in an attempt to test how the control architecture handles nonlinearity. The most significant change to the control architecture in this nonlinear study is the use of an extended Kalman filter over a linear Kalman filter. The way this EKF is applied to the proposed solution is detailed below.

Extended Kalman filter

To incorporate the nonlinear nature of the system, the linear matrix A_e from Equation Set 3.17 is recalculated at each iteration of the estimator. This dynamic matrix is now a function of time: $A_{e,t}$. The EKF will also use a nonlinear model to increment the estimated states. The remaining steps remain unchanged from Equation Set 3.17. $A_{e,t}$ is updated using Equation 3.19 below:

$$A_{e,k} = \frac{\partial f(\tilde{\boldsymbol{x}}_{e,k}, \boldsymbol{y}_{sp,k})}{\partial \tilde{\boldsymbol{x}}_{e,k}^T}_{|\tilde{\boldsymbol{x}}_{e,(k|k)}} = \begin{bmatrix} \frac{\partial f(\boldsymbol{x}_{m,k}, \boldsymbol{x}_{ps,k}, \boldsymbol{y}_{sp,k})}{\partial \boldsymbol{x}_{m,k}^T} & \frac{\partial f(\boldsymbol{x}_{m,k}, \boldsymbol{x}_{ps,k}, \boldsymbol{y}_{sp,k})}{\partial \boldsymbol{x}_{ps,k}^T} \\ 0 & I \end{bmatrix}_{|\tilde{\boldsymbol{x}}_{e,(k|k)}} \tag{3.19}$$

where $\boldsymbol{x}_{m,k}$, $\boldsymbol{x}_{ps,k}$, and $\boldsymbol{y}_{sp,k}$ represent the states, pseudostates, and set-points of the system at time k, respectively. Instead of being linearized around the steady-state conditions like the linear KF, the EKF linearizes the $A_{e,k}$ matrix around the current state values calculated by the estimator, $\tilde{\boldsymbol{x}}_{e,(k|k)}$. The subscript "$k|k$" denotes that the values used here are before any Kalman filter correction in this iteration. The " ˜ " symbol denotes the new vector structure, which now includes the parameters as pseudo states. As a result of including these pseudo states, $\tilde{\boldsymbol{x}}_{e,k}$ has m more rows than

$\boldsymbol{x}_k$, so the zero and identity matrices $\boldsymbol{0}$ and $\boldsymbol{I}$ are used to pad $A_{e,k}$ to its new size. This is explained in greater detail in Section 3.1.6.

The $A_{e,k}$ matrix is then discretized using Δt_{opt} as described in Section 3.1.3, resulting in the discretized matrix $A_{e,d,k}$. After updating the $A_{e,d,k}$ matrix, the steps proceed very similarly to Equation 3.17, except that Step 1, which increments the states, is now a nonlinear step. The resutlting algorithm for an extended Kalman filter is given below in Equation Set 3.20.

$$
\begin{aligned}
&0. \quad A_{e,d,k} = \frac{\partial f(\tilde{\boldsymbol{x}}_{e,k}, \boldsymbol{y}_{sp,k})}{\partial \tilde{\boldsymbol{x}}_{e,k}^T}_{|\tilde{\boldsymbol{x}}_{e,(k|k)}} \\
&1. \quad \tilde{\boldsymbol{x}}_{e,(k+1|k)} = f(\tilde{\boldsymbol{x}}_{e,(k|k)}, \boldsymbol{y}_{sp,k}) \\
&2. \quad P_{(k+1|k)} = A_{e,d} P_{(k|k)} A_{e,d}^T + V_k \\
&3. \quad K_{f,(k+1)} = P_{(k+1|k)} C_{e,d}^T \left(W_{(k+1)} + C_{e,d} P_{(k+1|k)} C_{e,d}^T \right)^{-1} \\
&4. \quad \tilde{\boldsymbol{x}}_{e,(k+1|k+1)} = \tilde{\boldsymbol{x}}_{e,(k+1|k)} + K_{f,(k+1)} \left(\boldsymbol{y}_{p,(k+1)} - C_{e,d} \tilde{\boldsymbol{x}}_{e,(k+1|k)} \right) \\
&5. \quad P_{(k+1|k+1)} = P_{(k+1|k)} - K_{f,(k+1)} C_{e,d} P_{(k+1|k)}
\end{aligned}
\tag{3.20}
$$

Notice in Equation Set 3.20 the EKF still uses linearized matrices, but nonlinear behavior is captured by the nonlinear model by updating $A_{e,d,k}$, the relationship between the inputs and the outputs, in real-time and computing $\tilde{\boldsymbol{x}}_{e,(k+1|k)}$ using the nonlinear model in Step 1. For a complete description on how these modifications turn the Kalman filter into an extended Kalman filter, see Walter et al. [4]. For further details on steps 1 through 5, please refer to the description of 3.17 as they remain unchanged.

3.1.9 Input Saturation

While a system may be observable according to the test described above, an estimator may still have difficulty predicting plant behavior if significant input saturation occurs. As shown in Figure 3.1, if the optimizer sends an unreachable set-point to the PI layer, it will be clipped by constraints the DRTO layer in unaware of. This causes significant plant-model mismatch, especially during the time the estimator is converging as it's estimated $\tilde{x}_e$ vector will be less accurate.

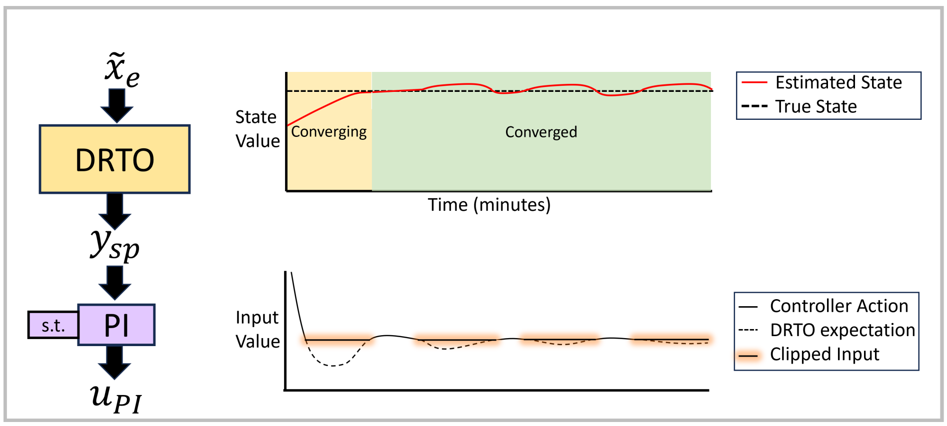

Figure 3.1: Issues with estimators and input saturation

As Figure 3.1 shows, each time input saturation occurs at the PI level the estimator is thrown off its converged value. Incorporating input saturation behavior into the DRTO layer is complicated; in simultaneous optimization the boolean logic from input saturation gives rise to mixed integer optimization or smoothing functions. It is therefore desirable to mitigate this effect by only sending set-points the PI layer can reach without input clipping.

3.2 Proposed Solution

The solution proposed here combines Kalman filter estimation with CL-DRTO and constrains the predicted behavior of the PI layer within the DRTO using estimated states from the Kalman filter or extended Kalman filter. This input constrained CL-DRTO must overcome the challenge of constraining inaccurate state estimates while the estimator is still converging. However, this work will show that even during convergence the effects of input clipping are reduced with DRTO constraints. As shown in Figure 3.2, input constrained CL-DRTO reduces the degree of plant-model mismatch from input saturation effects while the estimator is converging and eliminates it after convergence. After convergence the DRTO level will be able to accurately predict the PI level. By constraining the predicted response of the PI layer to its feasible region, the optimizer will only send set-points that the PI layer can reach without saturating the inputs.

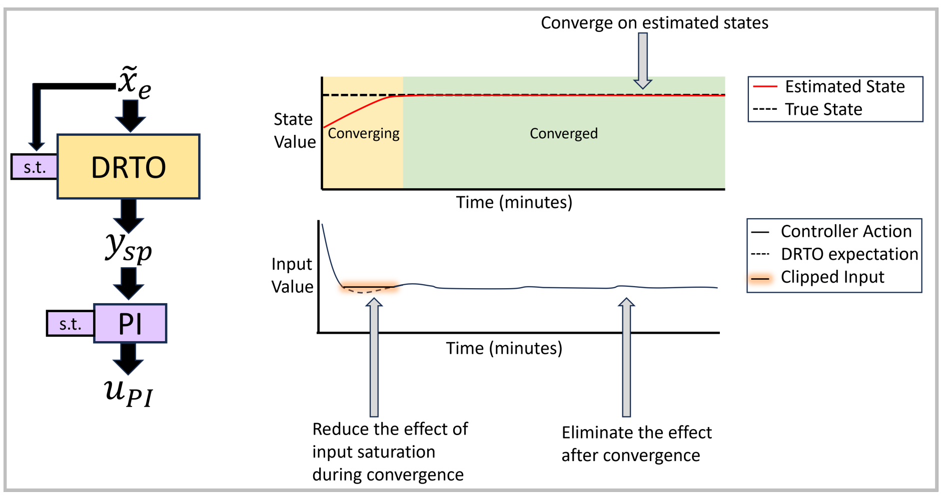

Figure 3.2: Input constrained CL-DRTO and its effect on the estimator

Once the DRTO level can avoid input saturation, the estimated states and parameters should stay at their converged values and the optimizer will send more accurate set-points to the controller. The result is a much more effective optimizer and greater economic yields for systems using this technique. The proposed solution will use a similar DRTO structure to Equation Set 3.11, which includes the optimizer equations in Equation Set 3.21 and the model in Equation Set 3.9.

$$\begin{aligned}
&\min_{Y_{\text{sp}}} \phi \\
\text{s.t. } & X_{m,\min} \leq X_{m,(k+1|k)} \leq X_{m,\max} \\
& Y_{\text{sp},min} \leq Y_{\text{sp}} \leq Y_{\text{sp},max} \\
& U_{\min} \leq U \leq U_{\max} \\
& Y_{\text{sp}} = \begin{bmatrix} \boldsymbol{y}^T_{\text{sp},k} & \boldsymbol{y}^T_{\text{sp},(k+1)} & \cdots & \boldsymbol{y}^T_{\text{sp},(k+M-1)} \end{bmatrix}^T \\
& X_{m,(k+1|k)} = \begin{bmatrix} \boldsymbol{x}^T_{m,(k+1|k)} & \boldsymbol{x}^T_{m,(k+2|k)} & \cdots & \boldsymbol{x}^T_{m,(k+P|k)} \end{bmatrix}^T
\end{aligned} \tag{3.21}$$

The significant change is the inclusion of constraints on predicted PI-level behavior: U. The model and plant will take the linearized and discretized structure given in Equation Set 3.9 and 3.10. The integral of error terms are measured from the PI layer and fed to the estimator and DRTO as pseudo states along with measured states and any parameters that need to be estimated. See Figure 3.3 for the structure of the proposed control strategy.

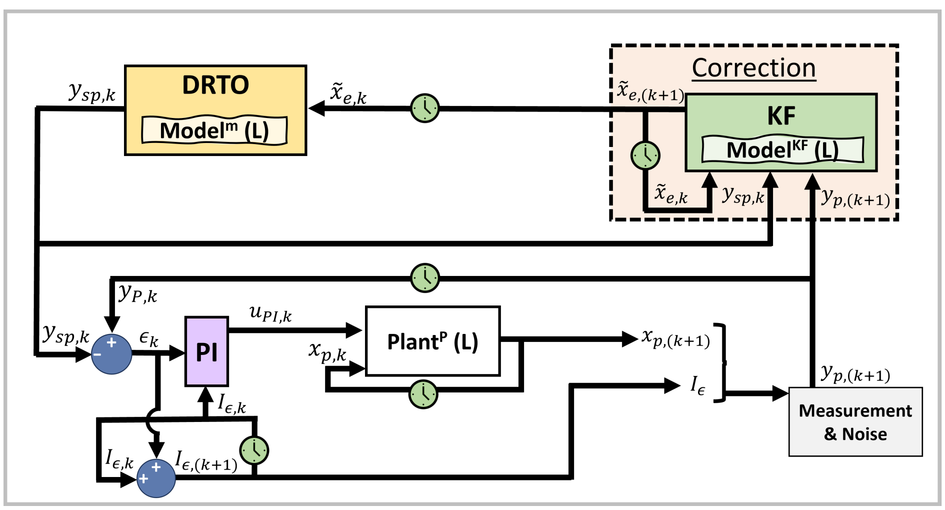

Figure 3.3: Information flow diagram of proposed solution in generalized form

Here the diagram uses a clock icon to denote the progression of time at the end of a time step and the time subscript "$k+1$" is stepped backwards to "k" for the next time step. In Figure 3.3 the pseudostate estimates of the unknown parameters are contained within $\tilde{\boldsymbol{x}}_e$. In Figure 3.3, $Model^m(L)$ is the linearized model referenced in Equation Set 3.9, the $Model^{KF}(L)$ is the linearized estimator model referenced in Equation Set 3.14 that includes additional estimated states and treats unknown parameters as pseudo-states. $Plant^P(L)$ is the linearized set of equations referenced in Equation Set 3.10 which is a function of the manipulated variable $\boldsymbol{u}_k$. Using this structure, this work seeks to show that despite the estimator taking time to converge, the CL-DRTO can still use these estimates to update the model so the optimizer can constrain the predicted controller response; in this case by only sending feasible set-points to the PI-level.

3.3 Software Implementation

The control architecture for this case study was coded using Python 3.9.6 in Jupyter Notebook, and represents the original work of the author, though several pre-built packages were employed within the code. CasADi 3.6.2 was used for optimization [8], and Control 0.9.3 was used for discretization [9]. Model discretization was performed with a zero-order holder being applied to the model inputs. The CL-DRTO is optimized using CasADi's built-in IPOPT solver [8]. Integration of the linear models was carried out using a forward Euler method, and in Chapter 3, integration of the nonlinear models was done using direct single shooting and the "CVODES" integrator method within CasADi [8] [10]. Linear simulations took an average of 5.2 seconds to compute, while nonlinear simulations took an average of 308 seconds to compute. The calculations were performed on a computer with an Intel Core i7-10750H CPU @ 2.60GHz and 32.0 GB of RAM.

3.4 Case Study

3.4.1 Nonlinear Plant

The CSTR case study used in the Economic MPC study in Li et al. [11] will be used here to evaluate the performance of the proposed method. The CSTR is a multiple-input multiple-output (MIMO) system where the process variables of concentration (C_A) and temperature (T) are controlled by the manipulated variables of inlet flowrate (F) and heater output (Q). A diagram showing the CSTR used in this case study is given below in Figure 3.4.

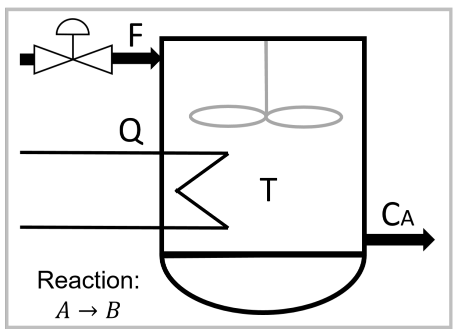

Figure 3.4: A diagram of the CSTR used in the case study

The nonlinear model from is given below, and the associated constants for this CSTR are defined in Table 3.1, and were obtained from Li et al. [11].

$$\frac{dC_A}{dt} = \frac{F}{V_R}(C_{A_0} - C_A) - k_0 e^{-\frac{E}{RT}} C_A^2 \tag{3.22}$$

$$\frac{dT}{dt} = \frac{F}{V_R}(T_0 - T) - \frac{\Delta H k_0}{\rho_R C_p} e^{-\frac{E}{RT}} C_A^2 + \frac{\eta Q}{\rho_R C_p V_R} \tag{3.23}$$

The heater efficiency will be considered unknown to the model. For the purposes of

Table 3.1: Table of values used in the nonlinear formula of case study

Symbol	Description	(Initial) Value	Units
C_A	Conc. of A in CSTR	0.339	kmol/m^3
T	Temperature of CSTR	545	K
F	Inlet Flowrate	5	m^3/h
Q	Heater Power	99,840	kJ/h
t	time	0	h
C_{A0}	Inlet Conc. of A	3.5	kmol/m^3
T_0	Inlet Temperature	300	K
k_0	Pre-exponential rate factor	8.46×10^6	m^3/kmol-h
E	Activation Energy	5×10^4	kJ/kmol
R	Ideal Gas Constant	8.314	kJ/kmol-K
ρ_R	Density of fluid in CSTR	1000	kg/m^3
C_p	Heat capacity of fluid in CSTR	0.231	kJ/kg-K
V_R	Reactor fluid volume	1.0	m^3
ΔH	Heat of reaction	-1.16×10^4	kJ/kmol
η	Heater efficiency	0.9	——

this case study, all parameters in Table 3.1 are assumed to remain constant throughout the simulation. The initial conditions of the process variables and manipulated variables were obtained by calculating the steady-state conditions corresponding to a set of specified inputs.

The nonlinear plant will be linearized in Section 3.4.2 below, however the nonlinear plant will also be used directly in Strategy 4.

3.4.2 Linearized Plant

The plant above was linearized and expressed in deviation form, represented using Equation 3.24.

$$
\begin{aligned}
\boldsymbol{x} &= \begin{bmatrix} C'_A \\ T' \end{bmatrix} = \begin{bmatrix} C_A - C_{A,\mathrm{ss}} \\ T - T_{\mathrm{ss}} \end{bmatrix} \\
\boldsymbol{u} &= \begin{bmatrix} F' \\ Q' \end{bmatrix} = \begin{bmatrix} F - F_{\mathrm{ss}} \\ Q - Q_{\mathrm{ss}} \end{bmatrix} \\
\boldsymbol{\theta} &= [\eta'] = \eta - \eta_{\mathrm{ss}} \\
\frac{d\boldsymbol{x}_p}{dt} &= \begin{bmatrix} a_p C'_A + b_p T' + e_p F' + f_p Q' + p_{1,p}\eta' \\ g_p C'_A + h_p T' + k_p F' + l_p Q' + p_{2,p}\eta' \end{bmatrix}
\end{aligned}
$$

In state space form, the linearized model is given by

$$
\begin{bmatrix} \frac{dC'_A}{dt} \\ \frac{dT'}{dt} \end{bmatrix} = \begin{bmatrix} a_p & b_p \\ g_p & h_p \end{bmatrix} \begin{bmatrix} C'_A \\ T' \end{bmatrix} + \begin{bmatrix} e_p \\ f_p \end{bmatrix} \begin{bmatrix} F' \\ Q' \end{bmatrix} + \begin{bmatrix} p_{1,p} \\ p_{2,p} \end{bmatrix} [\eta']
$$

which can be expressed compactly as

$$
\begin{aligned}
\dot{\boldsymbol{x}}_p &= A_p \boldsymbol{x}_p + B_p \boldsymbol{u} + D_{\theta,p}\boldsymbol{\theta} \\
\boldsymbol{y}_p &= C_p \boldsymbol{x}_p
\end{aligned} \tag{3.24}
$$

In Equation Set 3.17, constants a_p, b_p, e_p, f_p and $p_{1,p}$ represent the partial derivative of Equation 3.22 with respect to C_A, T, F, Q, and η, respectively. Constants g_p, h_p, k_p, l_p, and $p_{2,p}$ represent the partial derivative of Equation 3.23 with respect to C_A, T, F, Q, and η, respectively. The A_p, B_p, and $D_{\theta,p}$ matrices are used to create a linearized state space model for how the states change with time. Equation Set 3.24 is also discretized with $\Delta t_{\mathrm{PI/plant}} = 0.0\bar{3}3$ hrs, or 2 minutes. The C_p matrix contains ones and zeros that map the state space to the measurement space, effectively indicating which states are measured. Noise is also typically applied during this step, however noise was only added to the associated outputs during the noise study in Section 3.5.5. To add this noise, Gaussian random noise was applied to the outputs to simulate measurement noise. For the plant model, both states (T and C_A are measured, so C_p is the identity matrix. The integral of error terms are also calculated and treated as measurements, but they are added later as shown in Figure 3.3.

The efficiency parameter will not be measured; the estimator must predict this value for the DRTO. Additionally, the plant value of efficiency in deviation form (η') will always be zero as the true efficiency will not change with time. Once the parameters are grouped into their respective matrices, the plant can be described in state space form as described in Equation Set 3.7. For implementation in an optimizer, Equation Set 3.24 is also discretized as described in the Preliminaries section.

3.4.3 PI-Control Level

This system will utilize a PI-level controller at the plant level that changes the manipulated variables 5 times between each DRTO execution. The DRTO-level optimizer will send set-points for the next 3 time steps every 30 minutes, and the PI level will track these targets with 15 control actions taken every 2 minutes. A figure detailing the order these layers execute and their respective timing is given in Figure 3.5.

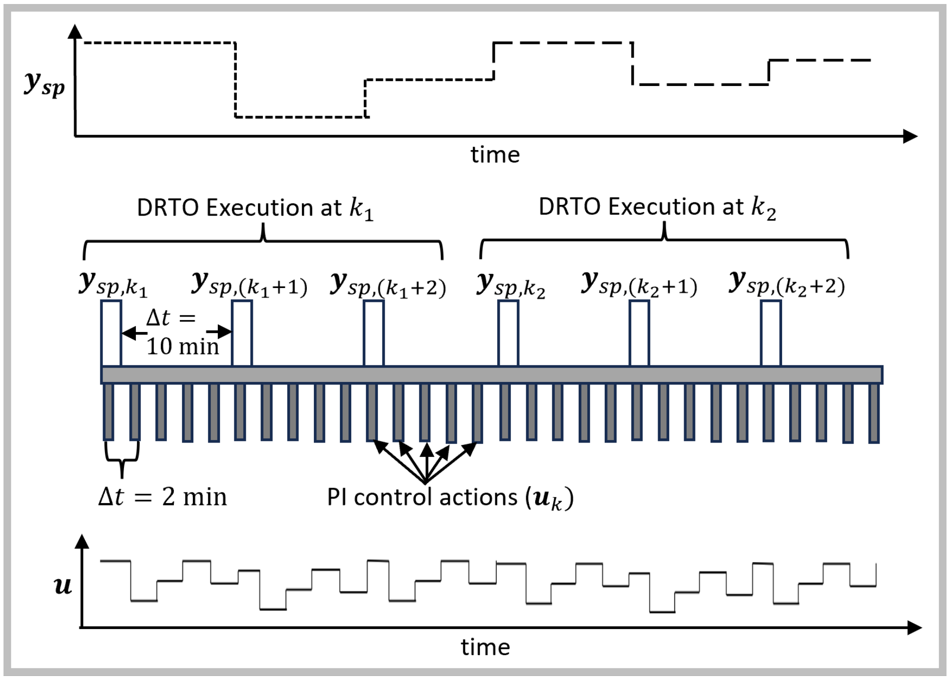

Figure 3.5: A diagram depicting the timing of the CL-DRTO layer and the PI control layer

The PI-level responds to the DRTO set-points using the following PI-equations:

$$
\begin{aligned}
F &= F_{\text{bias}} + K_{c,C_A}(C_{A,\text{sp}} - C_A) + \frac{K_{c,C_A}}{\tau_{I,C_A}} I_{C_A} \\
Q &= Q_{\text{bias}} + K_{c,T}(T_{\text{sp}} - T) + \frac{K_{c,T}}{\tau_{I,T}} I_T
\end{aligned}
\tag{3.25}
$$

where the set-points for concentration $C'_{A,SP}$ and T'_{SP} received from the DRTO are used to calculate the error and integral of the error. Anti reset windup keeps the integral terms (I_{C_A} and I_T) from accumulating during input saturation. The initial values of the manipulated variables are set as the bias of their respective PI-equation. The tuning constant values were chosen by trial and error testing to give rapid and smooth performance and are given in Table 3.2.

Table 3.2: Tuning Constants used in the PI-Layer

Tuning Constant	Inlet Flowrate (F)	Heater Power (Q)
K_c	16	70
τ_I	0.05	0.01

3.4.4 DRTO Layer

The DRTO layer has its own model that incorporates the behavior of the PI layer, as shown in Equation Set 3.7. To this end, the manipulated variables in the nonlinear plant are replaced by their respective PI-equations. The new nonlinear equations for this case study are given in Equation Set 3.26.

$$\begin{aligned}
\frac{dC_A}{dt} &= \frac{(F_{\text{bias}} + K_{c,C_A}(C_{A,\text{sp}} - C_A) + \frac{K_{c,C_A}}{\tau_{I,C_A}} I_{C_A})}{V_R}(C_{A_0} - C_A) \\
&\quad - k_0 e^{-\frac{E}{RT}} C_A^2 \\
\frac{dT}{dt} &= \frac{F_{\text{bias}} + K_{c,C_A}(C_{A,\text{sp}} - C_A) + \frac{K_{c,C_A}}{\tau_{I,C_A}} I_{C_A}}{V_R}(T_0 - T) \\
&\quad - \frac{\Delta H_{k_0}}{\rho_R C_p} e^{-\frac{E}{RT}} C_A^2 \\
&\quad + \frac{\eta(Q_{\text{bias}} + K_{c,T}(T_{\text{sp}} - T) + \frac{K_{c,T}}{\tau_{I,T}} I_T)}{\rho_R C_p V_R} \\
\frac{dI_{C_A}}{dt} &= C_{A,\text{sp}} - C_A \\
\frac{dI_T}{dt} &= T_{\text{sp}} - T
\end{aligned} \tag{3.26}$$

The DRTO layer model is also linearized and expressed in deviation form. The linearization process is given in Equation Set 3.27:

$$
\begin{aligned}
\boldsymbol{x}_m &= \begin{bmatrix} C_A' \\ T' \\ I_{C_A}' \\ I_T' \end{bmatrix} = \begin{bmatrix} C_A - C_{A,ss} \\ T - T_{ss} \\ I_{C_A} - I_{C_{A,ss}} \\ I_T - I_{T_{ss}} \end{bmatrix} \\
\boldsymbol{y}_{sp} &= \begin{bmatrix} C_{A,sp}' \\ T_{sp}' \end{bmatrix} = \begin{bmatrix} C_{A,sp} - C_{A,sp,ss} \\ T_{sp} - T_{sp,ss} \end{bmatrix} \\
\boldsymbol{\theta} &= [\eta'] = [\eta - \eta_{ss}] \\
\frac{d\boldsymbol{x}_m'}{dt} &= \begin{bmatrix} a_m C_A' + b_m T' + c_m I_{C_A}' + d_m I_T' + e_m C_{A,sp}' + f_m T_{sp}{}' + p_{1,m}\eta' \\ g_m C_A' + h_m T' + i_m I_{C_A}' + j_m I_T' + k_m C_{A,sp}' + l_m T_{sp}{}' + p_{2,m}\eta' \\ m_m C_A' + n_m T' + o_m I_{C_A}' + p_m I_T' + q_m C_{A,sp}' + r_m T_{sp}{}' + p_{3,m}\eta' \\ s_m C_A' + t_m T' + u_m I_{C_A}' + v_m I_T' + w_m C_{A,sp}' + z_m T_{sp}{}' + p_{4,m}\eta' \end{bmatrix}
\end{aligned}
$$

In state space form, the linearized model is given by

$$
\begin{bmatrix} \frac{dC_A'}{dt} \\ \frac{dT_m'}{dt} \\ \frac{dI_{C_A}'}{dt} \\ \frac{dI_T'}{dt} \end{bmatrix} = \begin{bmatrix} a_m & b_m & c_m & d_m \\ g_m & m_m & s_m & j_m \\ h_m & n_m & t_m & p_m \\ i_m & o_m & u_m & v_m \end{bmatrix} \begin{bmatrix} C_A' \\ T' \\ I_{C_A}' \\ I_T' \end{bmatrix} + \begin{bmatrix} e_m & f_m \\ k_m & l_m \\ q_m & r_m \\ w_m & z_m \end{bmatrix} \begin{bmatrix} C_{A,sp}' \\ T_{sp}' \end{bmatrix} + \begin{bmatrix} p_{1,m} \\ p_{2,m} \\ p_{3,m} \\ p_{4,m} \end{bmatrix} \begin{bmatrix} \eta' \end{bmatrix}
$$

which can be expressed compactly as

$$
\begin{aligned}
\dot{\boldsymbol{x}}_m(t) &= A_m \boldsymbol{x}_m + B_m \boldsymbol{y}_{sp} + D_{\theta,m}\boldsymbol{\theta} \\
\boldsymbol{y}_m(t) &= C_m \boldsymbol{x}_m
\end{aligned} \tag{3.27}
$$

where the entries in row r and column c of matrices A_m, B_m, and $D_{\theta,m}$ describe the partial derivative of the equation from row r of $\dot{x}_m$ with respect to the state/parameter from column c of A_m, B_m, and $D_{\theta,m}$. The entries in these matrices are given subscript m to distinguish the partial derivatives of the model from those in the plant that are derived from different equations. Equation Set 3.27 is discretized with $\Delta t_{\text{opt}} = 0.1\bar{6}6$ hrs, or 10 minutes. The DRTO model may be given different efficiency parameters as mismatch and convergence change this value, so η' in the model is not necessarily zero as it is in the plant. The DRTO will measure all the plant states as well as the integral of error states from the PI controller. This is represented by C_m in Equation Set 3.27. Efficiency will be predicted using the estimator or a guess value will be used, and a bias update will compensate for the mismatch. This chapter proposes using a DRTO layer with an estimator and DRTO-level input constraints. This DRTO structure will be compared to several alternatives using two common objective functions. The first is a target tracking objective:

$$\varphi_{Tgt} = \alpha_{1,Tgt}(C_A - C_{A_{Tgt}})^2 + \alpha_{2,Tgt}(T - T_{Tgt})^2 \tag{3.28}$$

where ϕ is the objective value, and the goal of the optimizer is to minimize the difference between the current concentration and temperature and the specified targets for those states ($C_{A_{Tgt}}$ and T_{Tgt}, respectively). α_1 and α_2 are weighting constants used to give priority to either temperature target tracking or concentration target tracking. For this case study, $\alpha_{1,Tgt}$ will be 1000 and $\alpha_{2,Tgt}$ will be 1. These values were These values were shown to provide a good balance to account for the differences in the variable responses and their units. Most often, high-level optimizers are used to achieve higher-level objectives, such as the profitability of the plant. Therefore, an economic objective will also be used to quantify the performance between DRTO structures. The objective function in Equation 3.29 was derived from one by Li and Zhang to approximate the profitability of the process:

$$\varphi_{Econ} = -\alpha_{1,Econ}F(C_{A0} - C_A) + \alpha_{2,Econ}Q^2 \tag{3.29}$$

In Equation 3.29, revenue is described by the concentration of A consumed, and the cost is described by the power consumed by the heater. New tuning constants are used to make revenue the main priority while still considering the costs. The new tuning constants $\alpha_{1,Econ}$ and $\alpha_{2,Econ}$ are 10^5 and 10^{-7}, respectively. Together the function is minimized to find the profitability of the plant as F, Q, and C_A change with time. The state vector (x_p) for this case study also contains two integral terms that track the integral of error for concentration (I_{C_A}) and temperature (I_T). These terms are "measured" from the PI layer, and an information flow diagram showing these connections can be found in Figure 3.8. The DRTO layer is solved using the optimization problem given in Equation Set 3.11.

3.4.5 Constraints

To show the effects of input saturation, the system is highly constrained. The bounds on the process variables, DRTO set-points, and manipulated variables are given in Table 3.3.

Table 3.3: Constraints on the states and inputs for the case study

Symbol	Upper Bound	Lower Bound	Units
C_{A}	3.5	0.1	kmol/m^3
T	700	400	K
F_{Tgt}	13	0	m^3/h
F_{Econ}	2.8	0	m^3/h
Q	400000	0	kJ/h
$C_{\mathrm{A,sp}}$	3.5	0	kmol/m^3
T_{sp}	700	400	K
$\Delta C_{\mathrm{A,sp}}$	0.1	-0.1	kmol/m^3
ΔT_{sp}	30	-20	K

As this chapter will be studying the effects of input saturation, an infeasible target will be given to the target tracking objective from Equation 3.28. To ensure input saturation occurs when testing economic performance that lacks these specified targets, the inlet flow rate is more heavily constrained. These two constraints are given as F_{Tgt} for the target tracking objective and F_{econ} for the economic objective.

The chosen CSTR will also need to preserve at least 0.1 kmol/m^3 of the reactant for use in a downstream process. Dropping below this level will incur a financial penalty as the downstream process is affected. Dropping C_A to 0 kmol/m^3 will cost 10% of that time step's revenue. Dropping halfway to 0.05 kmol/m^3 will incur half the penalty, and so on. This penalty will be applied when the simulation is evaluated at the end and the objective functions are calculated over the entire horizon. This penalty will not occur within the DRTO as it only takes effect when the DRTO is inaccurately predicting the plant states, however the lower bound of 0.1 kmol/m^3 will be applied within the DRTO as a lower constraint on C_A.

Several strategies will be compared with the proposed Kalman filter strategy, but all will use input constrained CL-DRTO described in Section 3.2. The different strategies are given below and described in the sections following Table 3.4.

Table 3.4: A table of strategies tested

Strategy 1	Kalman filter estimation
Strategy 2	Bias updating
Strategy 3	Full state feedback
Strategy 4	Nonlinear system and extended Kalman filter

3.4.6 Strategy 1: Kalman Filter Estimation

In this strategy, the DRTO layer will avoid the obstacle of input saturation by predicting the input actions of the PI-layer (U) and constraining them within bounds of feasibility.

$$\begin{aligned}
&\min_{Y_{\text{sp}}} \phi \\
\text{s.t. } & X_{m,\min} \leq X_{m,(k+1|k)} \leq X_{m,\max} \\
& Y_{\text{sp},min} \leq Y_{\text{sp}} \leq Y_{\text{sp},max} \\
& U_{\min} \leq U \leq U_{\max} \\
& Y_{\text{sp}} = \begin{bmatrix} \boldsymbol{y}_{\text{sp},k}^T & \boldsymbol{y}_{\text{sp},(k+1)}^T & \cdots & \boldsymbol{y}_{\text{sp},(k+M-1)}^T \end{bmatrix}^T \\
& X_{m,(k+1|k)} = \begin{bmatrix} \boldsymbol{x}_{m,(k+1|k)}^T & \boldsymbol{x}_{m,(k+2|k)}^T & \cdots & \boldsymbol{x}_{m,(k+P|k)}^T \end{bmatrix}^T
\end{aligned} \tag{3.30}$$

The DRTO layer for this strategy consists of Equation Set 3.30 above and is subject to a discretization of the model in Equation Set 3.27. The estimator includes additional estimated states and treats unknown parameters as pseudo-states, including them in the model as if they were process variables. Under this premise, the model used in the DRTO Equation Set 3.27 is augmented to include the pseudo-states (in this case, η). This new estimator model is exclusively used within the estimator and feeds states and parameters to the DRTO model.

$$\begin{aligned}
\tilde{\boldsymbol{x}}_e &= \begin{bmatrix} C_A' \\ T' \\ I_{C_A}' \\ I_T' \\ \eta_{ss}' \end{bmatrix} = \begin{bmatrix} C_A - C_{A,ss} \\ T - T_{ss} \\ I_{C_A} - I_{C_{A,ss}} \\ I_T - I_{T_{ss}} \\ \eta - \eta_{ss} \end{bmatrix} \\
\boldsymbol{y}_{sp} &= \begin{bmatrix} C_{A,sp}' \\ T_{sp}' \end{bmatrix} = \begin{bmatrix} C_{A,sp} - C_{A,sp,ss} \\ T_{sp} - T_{sp,ss} \end{bmatrix} \\
\frac{d\tilde{\boldsymbol{x}}_e}{dt} &= \begin{bmatrix} a_m C_A' + b_m T' + c_m I_{C_A}' + d_m I_T' + e_m C_{A,sp}' + f_m {T_{sp}}' + p_{1,m}\eta' \\ g_m C_A' + h_m T' + i_m I_{C_A}' + j_m I_T' + k_m C_{A,sp}' + l_m {T_{sp}}' + p_{2,m}\eta' \\ m_m C_A' + n_m T' + o_m I_{C_A}' + p_m I_T' + q_m C_{A,sp}' + r_m {T_{sp}}' + p_{3,m}\eta' \\ s_m C_A' + t_m T' + u_m I_{C_A}' + v_m I_T' + w_m C_{A,sp}' + z_m {T_{sp}}' + p_{4,m}\eta' \\ p_{5,e} C_A' + p_{6,e} T' + p_{7,e} I_{C_A}' + p_{8,e} I_T' + p_{9,e} C_{A,sp}' + p_{10,e} {T_{sp}}' + p_{11,e}\eta' \end{bmatrix}
\end{aligned}$$

In state space form, the linearized model is given by

$$\begin{bmatrix} \frac{dC'_A}{dt} \\ \frac{dT'_m}{dt} \\ \frac{dI'_{C_A}}{dt} \\ \frac{dI'_T}{dt} \\ \frac{d\eta'}{dt} \end{bmatrix} = \begin{bmatrix} a_m & b_m & c_m & d_m & p_{1,m} \\ g_m & m_m & s_m & j_m & p_{2,m} \\ h_m & n_m & t_m & p_m & p_{3,m} \\ i_m & o_m & u_m & v_m & p_{4,m} \\ p_{5,e} & p_{6,e} & p_{7,e} & p_{8,e} & p_{11,e} \end{bmatrix} \begin{bmatrix} C'_A \\ T' \\ I'_{C_A} \\ I'_T \\ \eta' \end{bmatrix} + \begin{bmatrix} e_m & f_m \\ k_m & l_m \\ q_m & r_m \\ w_m & z_m \\ p_{9,e} & p_{10,e} \end{bmatrix} \begin{bmatrix} C'_A, sp \\ T'_{sp} \end{bmatrix}$$

which can be expressed compactly as

$$\begin{aligned} \dot{\tilde{\boldsymbol{x}}}_e(t) &= A_e \tilde{\boldsymbol{x}}_e + B_e \boldsymbol{y}_{sp} \\ \tilde{\boldsymbol{y}}_e(t) &= C_e \tilde{\boldsymbol{x}}_e \end{aligned} \tag{3.31}$$

As described in detail in Section 3.1.6, the model from Equation Set 3.27 is expanded to include an unknown parameter (η) in a larger state vector $\tilde{x}_e$. In Equation Set 3.31, the estimator model linearizes an additional nonlinear equation: the dynamics of pseudo-state η. While most of the matrix entries can be reused from the model, the estimator model requires an additional row of entries. However, since the true efficiency is assumed a constant value, $\frac{d\eta}{dt} = 0$, thus $\frac{d\eta'}{dt} = 0$ and $p_{5,e} = p_{6,e} = p_{7,e} = p_{8,e} = p_{9,e} = p_{10,e} = p_{11,e} = 0$. The estimator receives measurements of process variables C_A and T from the plant and the integral of error terms I_{C_A} and I_T from the PI-layer. The efficiency η is not measured. The state space model matrices used here are built by expanding the state space model matrices from the model as explained in section 3.1.6. Equation Set 3.31 is also discretized with $\Delta t_{\text{opt}} = 0.1\bar{6}6$ hrs, or 10 minutes. The observability of measuring C_A, T, I_{C_A}, and I_T to predict the efficiency η was tested using the observability test in Section 3.1.7. The rank of the resulting matrix was 5, which is equal to the dimension in this estimator, thus the estimator should be able to estimate the unknown efficiency parameter. Figure 3.6 shows a block flow diagram of the control strategy proposed here.

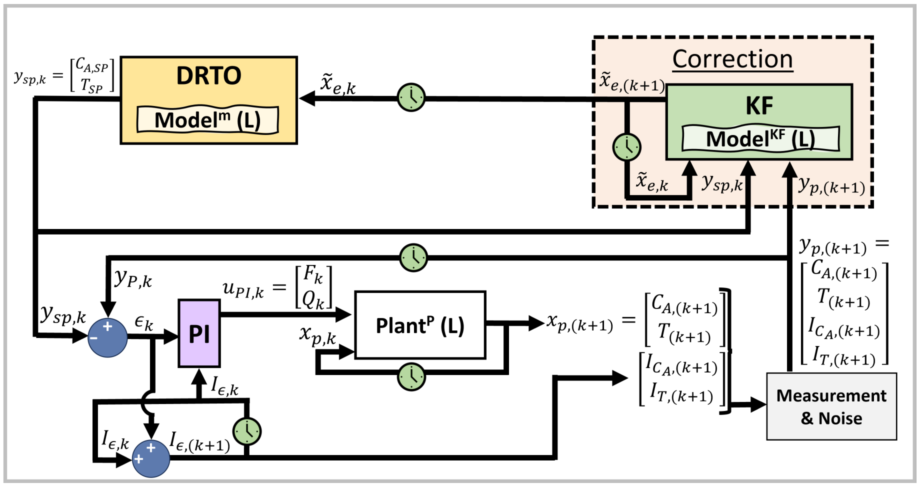

Figure 3.6: Information flow diagram for DRTO with Kalman filter

The structure shown in Figure 3.6 is consistent with Kalman filter implementations with CL-DRTO in accepted literature with one key difference: this formulation has constraints on PI-actions at the DRTO-level. These inputs are calculated using the PI control laws shown in Equation Set 3.25, and the states and parameters come from the estimator. The estimator handles measurement noise and unknown parameter values to allow a prediction of what the PI-controller will do with the set-points given from the DRTO layer. This makes DRTO optimization able to reduce and avoid input saturation by sending reasonable set-points to the plant controllers.

3.4.7 Strategy 2: Bias Updating

This strategy will also constrain DRTO set-points within feasible reach of the PI layer, but will be unable to estimate the unknown parameter η or to filter noise like the Kalman filter. Instead, bias updating corrects for plant-model mismatch by adjusting the constraints and objective function, as described in detail in Section 3.1.5. This is the typical approach for dealing with mismatch, and will be used as a reference for how the proposed solution improves on current practice in CL-DRTO.

For this case study, a bias term is created for each measured state: in this case, β_{C_A}, β_T, $\beta_{I_{C_A}}$, and β_{I_T}. These terms are the difference between the measured state and the value expected by the model. The bias updating step uses Equation Set 3.27, the same model used in the DRTO optimizer, which was discretized with $\Delta t_{\text{opt}} = 0.1\bar{6}6$ hrs, or 10 minutes. The bias terms affect the optimizer as follows:

$$\begin{gathered} \boldsymbol{x}_{UB} - \boldsymbol{\beta}_{\text{bias},(k+1)} \\ \boldsymbol{x}_{LB} - \boldsymbol{\beta}_{\text{bias},(k+1)} \end{gathered} \tag{3.32}$$

$$\begin{aligned} \varphi_{Tgt} = \alpha_{1,Tgt} \left((C_{A,k} + \beta_{\text{bias},k,CA}) - C_{A,\text{SP},k} \right)^2 \\ + \alpha_{2,Tgt} \left((T_k + \beta_{\text{bias},k,T}) - T_{\text{SP},k} \right)^2 \end{aligned} \tag{3.33}$$

$$\varphi_{Econ} = -\alpha_{1,Econ} F(C_{A0} - (C_A + \beta_{\text{bias},k,CA})) + \alpha_{2,Econ} Q^2 \tag{3.34}$$

Where the lower and upper states are shifted by the bias term so the constraints reflect the same degree of bias. Any constraints that do not involve a measured output variable (i.e. the DRTO set-points, efficiency, etc.) are not adjusted by a bias. The objective function for target tracking, (Equation 3.33) and the objective function for economic evaluation (Equation 3.34) are adjusted to reflect the true value by adding the bias term. The block flow diagram for this control strategy is shown in Figure 3.7:

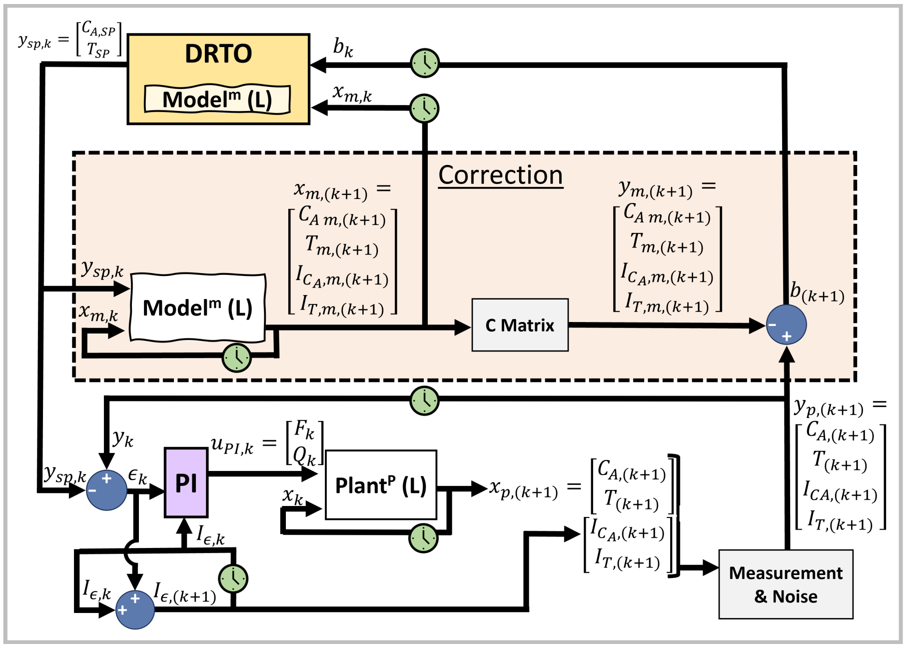

Figure 3.7: Information flow diagram for DRTO with bias updating

Note in Figure 3.7 that a copy of the model used in the DRTO layer is used in the bias updating step to predict the model values of the measured states. This strategy makes no attempt to estimate the efficiency but will instead use a nominal value for the efficiency and correct for the mismatch using the bias terms.

3.4.8 Strategy 3: Full state feedback

While it is unlikely that any real system would be able to measure every parameter and state in its model, it gives an accurate best-case scenario to compare the other 3 strategies to. Under the best circumstances the other 3 strategies cannot achieve this level of performance, and in practice measuring all values is usually near impossible or very expensive. The control architecture for this process is simple, as no estimators or corrections are needed for it to achieve optimal performance. As with the other strategies, the DRTO will be constrained such that only feasible set-points are sent to the PI layer to avoid input saturation. The plant is described by Equation Set 3.24 and the model is described by the discretization of Equation Set 3.27. Just as in the other strategies, the DRTO model is discretized with $\Delta t_{\text{opt}} = 0.1\bar{6}6$ hrs, or 10 minutes, and the plant is discretized with $\Delta t_{\text{PI/plant}} = 0.0\bar{3}3$, or 2 minutes.

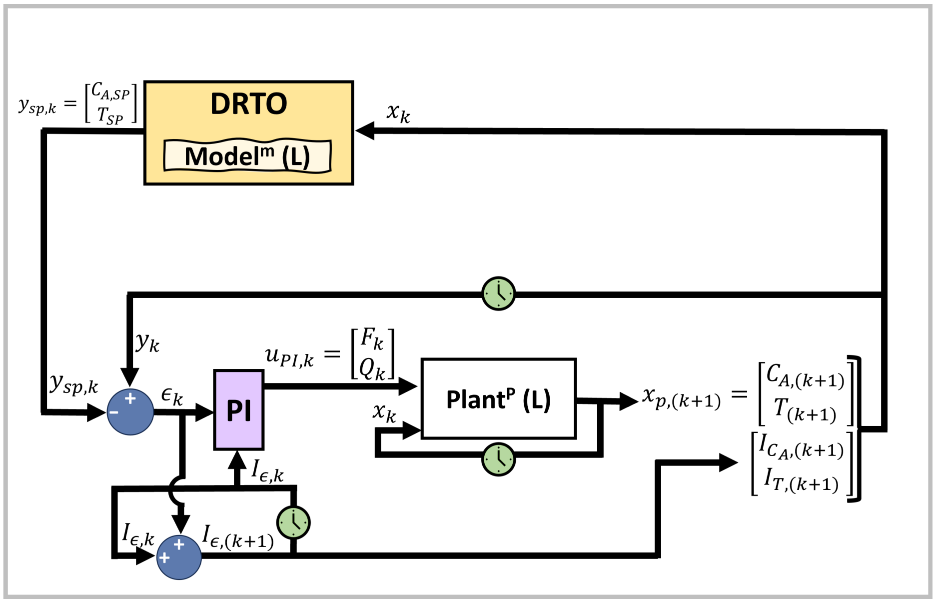

Figure 3.8: Information flow diagram for DRTO with full state feedback, where the true value of the efficiency parameter is known

In this strategy, the true value of the efficiency parameter η is known at the DRTO level, so constraints on PI actions in the DRTO-layer are particularly effective because the DRTO-layer can accurately predict plant behavior to reduce input clipping. This ideal will serve as a baseline for the other 3 methods.

3.4.9 Strategy 4: Nonlinear System and Extended Kalman Filter

Linearization significantly reduces the computational expense of optimization allowing for increased sampling frequency of the model and optimizer. This linearization uses a truncated taylor series expansion, neglecting higher order terms. However in some cases neglecting these components lead to a poor representation of the nonlinear system's behavior. These higher order components of the Taylor series are particularly important for precisely determining the mean and covariance of the estimates for nonlinear systems. Therefore as nonlinearity increases, estimation performance degrades under a linear Kalman filter [7]. More detail on the need for EKF can be found in Section 3.1.8.

To validate the Kalman filter with DRTO input constraints, its performance will also be tested in a nonlinear setting. To accomplish this, the linear plant will be replaced with a nonlinear one, the DRTO will use a nonlinear model to capture the behavior, and the Kalman filter will be extended to include nonlinearity as well. The generic formulas governing these changes are given in the Preliminaries section, and the formulas specific to this case study are given here.

Step testing indicated the nonlinear system was better able to reach targets than its linear counterpart. The target tracking constraints used in the linear system did not produce input saturation in the nonlinear system, so to ensure the nonlinear case could be evaluated in the presence of input saturation, further restrictions were needed. Experimental tests showed that restricting the step in temperature by 50% made it more difficult for the nonlinear system to achieve higher concentrations, so these restricted temperature targets were used to ensure input saturation could be studied in the nonlinear case.

While using the nonlinear plant found in Section 3.4.1, new tuning parameters will be needed for the PI controller, which are listed in Table 3.5.

Table 3.5: New tuning constants for the controller of the nonlinear plant

Tuning Constant	Inlet Flowrate (F)	Heater Power (Q)
K_c	6	70
τ_I	0.01	0.001

The DRTO for the nonlinear strategies will still optimize the best set-points to send to the PI layer, however in this strategy, the DRTO layer will predict the control action in an auxiliary function instead of including the formulas directly in the model. Equation 3.25 will be used to predict the control action based on the DRTO set-points, then those predictions will be used in Equations 3.22 and 3.23 to predict the next set of states.

Constraints and objective functions used in the linear strategies will be the same in the nonlinear strategies. As in Strategy 1, this strategy will constrain the predicted PI-level action to ensure only feasible set-points are sent to the PI-control level.

$$\begin{aligned}
&\min_{Y_{\text{sp}}} \phi_{\text{sp}} \\
\text{s.t. } & X_{m,\min} \leq X_{m,(k+1|k)} \leq X_{m,\max} \\
& Y_{sp,min} \leq Y_{\text{sp}} \leq Y_{sp,max} \\
& U_{\min} \leq U \leq U_{\max} \\
& Y_{\text{sp}} = [\boldsymbol{y}_{\text{sp},k}^T, \boldsymbol{y}_{\text{sp},(k+1)}^T, \ldots, \boldsymbol{y}_{\text{sp},(k+M-1)}^T]^T \\
& X_{m,(k+1|k)} = [\boldsymbol{x}_{m,(k+1|k)}^T, \boldsymbol{x}_{m,(k+2|k)}^T, \ldots, \boldsymbol{x}_{m,(k+P|k)}^T]^T
\end{aligned} \tag{3.35}$$

The DRTO layer will be comprised of Equation Set 3.35 and subject to a discretization of the model in Equation Set 3.27. The formulas governing the DRTO model and the plant will both be Equation Set 3.36. See Equation Section 3.4.1 for a detailed description of the variables within it.

$$\begin{aligned}
\frac{dC_A}{dt} &= \frac{F}{V_R}(C_{A_0} - C_A) - k_0 e^{-\frac{E}{RT}} C_A^2 \\
\frac{dT}{dt} &= \frac{F}{V_R}(T_0 - T) - \frac{\Delta H k_0}{\rho_R C_p} e^{-\frac{E}{RT}} C_A^2 + \frac{\eta Q}{\rho_R C_p V_R} \\
\frac{dI_{C_A}}{dt} &= C_{A,sp} - C_A \\
\frac{dI_T}{dt} &= T_{sp} - T \\
\frac{d\eta}{dt} &= 0
\end{aligned} \tag{3.36}$$

The EKF will also use the nonlinear equations above to increment the states in Step 1 of Equation Set 3.20. However, as mentioned at the end of Section 3.1.8, it is still necessary to derive all the discretized and linearized state space model matrices $A_{e,d,(t)}$, $B_{e,d}$, $C_{e,d}$, for the remaining steps of the EKF algorithm.

These state space model matrices will be structured identical to Equation Set 3.31, however the formulas being linearized here (shown in Equation Set 3.36) will not contain PI-level dynamics. The matrices will be discretized using $\Delta t_{\text{opt}} = 0.1\bar{6}6$ hours, or 10 minutes.

In Step 1 of the Kalman filter algorithm, instead of using Equation Set 3.31 to increment the states, the nonlinear model will increment the states instead. The state space model matrices of Equation Set 3.31 will instead be used to update the extended Kalman filter tuning matrices.

The nonlinear setup also passed the same observability testing done in Strategy 1, indicating that the observability is sufficient to control the system. However this was done with the nominal value of the A matrix, which updates each time step (see Section 3.1.6 for details). While in theory this could affect observability, it is assumed here that the system will remain observable within the operating range of the simulations performed. Figure 3.9 shows a block flow diagram showing the structure of the control strategy proposed here.

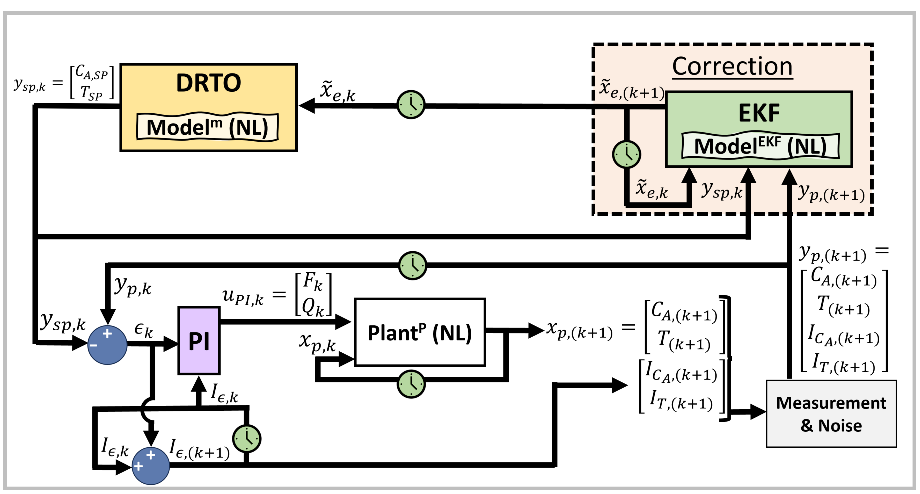

Figure 3.9: Information flow diagram for Nonlinear DRTO with Kalman filter

The hierarchy above is similar to Strategy 1 except the models used in the plant

(Plant$^{\mathrm{P}}$), the DRTO model (Model$^{\mathrm{m}}$), and the estimator model (Model$^{\mathrm{EKF}}$) are now all nonlinear. The state space model matrices used in the EKF approximate the plant using derivatives of Equation Set 3.36, however the actual integral of error terms I_{C_A} and I_T are additively calculated at each iteration as shown in Figure 3.9. The difference between these methods is negligible, and the additive calculation is a simpler method more likely found in industrial practice. A lower level of noise is applied here as with the other strategies to better distinguish between the performance of various strategies.

3.5 Results and Discussion

3.5.1 SP-Plant Based DRTO

Typically, DRTO uses a model of the plant that does not take into account the PI controller, however this work proposes using a DRTO model that incorporates both the plant and PI controller. Including the controller in the model in this way makes the model a function of the set-point. To verify this set-point based plant (SP-Plant), the plant in Strategy 3 with full state feedback is run with and without this PI-integrated DRTO model. Ideally the DRTO should behave the same whether the PI model is incorporated into the model or calculated alongside the model (PI-Plant), however discretization and linearization of the PI-layer may affect dynamics, so the SP-Plant was tested to verify unchanged dynamics. In the test shown in Figure 3.10, the targets for both controllers were stepped so the transient dynamics of each could be observed.

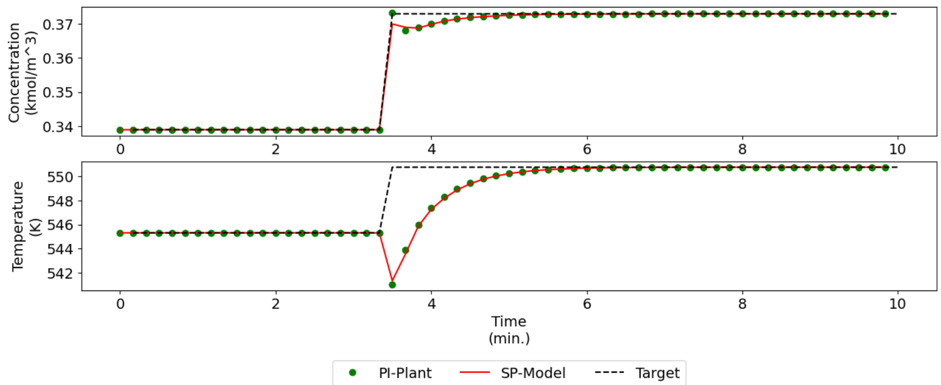

Figure 3.10: A parallel comparison of DRTO with and without a combined model and PI-layer

Figure 3.10 shows the two plants produce nearly identical results. As such it is safe to assume that using a combined model of the plant and PI-layer will predict the same values as a model of the plant alone. There is an important caveat to these findings however: the results will only match if input saturation of the PI layer does not affect plant behavior. A DRTO model that does not include the PI layer would have mismatch when the inputs saturate as it would be blind to PI-level constraints. For this reason, a DRTO model that includes the PI layer is more robust, as described in the proposed solution.

3.5.2 CL-DRTO

By incorporating the PI layer prediction into the DRTO layer, the DRTO can constrain set-points to within the feasible region of the PI controller. This minimizes input saturation during convergence of the estimator, and after convergence or with full state feedback, this method should avoid input saturation altogether. For estimators in particular this is critical, as input saturation can have a serious impact on estimator performance if not approximated. The issue can be particularly problematic in this implementation as the controller dynamics are embedded into the estimator. This effect is demonstrated in the figure below, where Strategy 1 is tested using input constrained CL-DRTO and using CL-DRTO without input constraints.

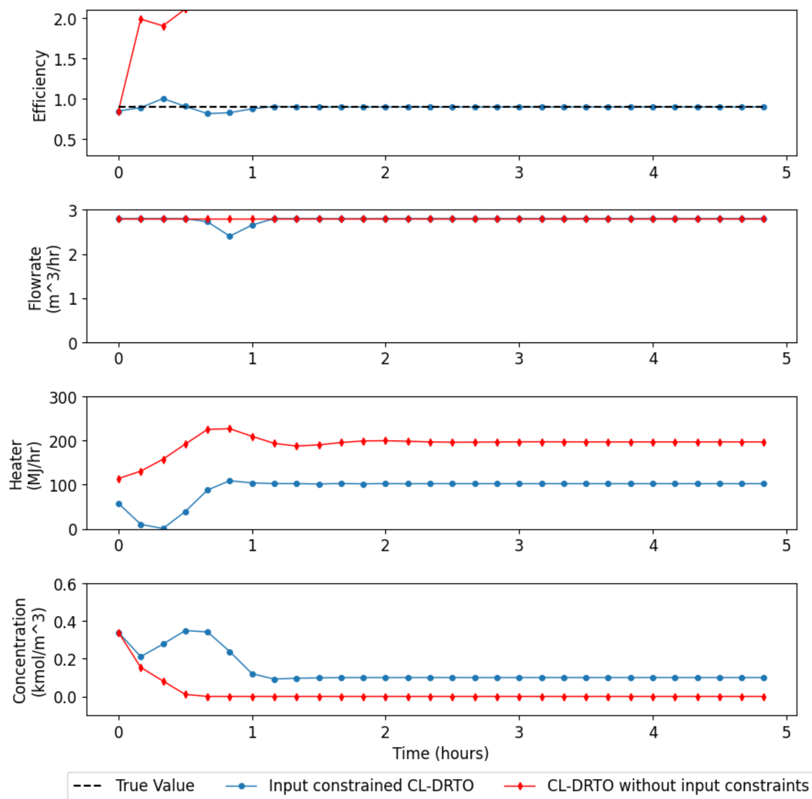

Figure 3.11: A parallel comparison of input constrained CL-DRTO and CL-DRTO without input constraints.

Figure 3.11 shows the estimated parameter η and the three states used to determine profitability (F, C_A, and Q). Under the economic objective from Equation 3.29, the need to approximate input saturation with an estimator is very apparent. As the flowrate (F) is constantly saturated, the estimator immediately fails to approximate the efficiency parameter without predicting and constraining PI set-points using the input constrained CL-DRTO. As a result, the lower bound C_A constraint of 0.1 kmol/m^3 is violated and suboptimal heater values are chosen. This is reflected by a 14.8% drop in economic performance and a 23.4% drop in target tracking performance. This demonstrates the need to approximate input saturation in some manner within the DRTO, especially when using an estimator that also considers the closed-loop behavior.

There are also alternative methods of dealing with input saturation. Instead of avoiding the event, it can be rigorously captured using methods such as complimentarity constraints, however these methods are more complex to implement, and their efficacy compared to the proposed method is a subject for future research. For the purposes of this study, the input constrained CL-DRTO will be considered adequate in achieving control amidst saturated inputs, and the architecture will instead be evaluated on its estimator performance compared to alternative disturbance handling techniques such as bias updating. Estimating input saturation in the DRTO (either through input constrained CL-DRTO or other input saturation handling methods) is critical to achieving superior performance.

3.5.3 Economic Performance

The primary function of an optimization strategy is to improve the economic performance of the plant. With estimators this is of particular interest, as there are regions of poor control while the estimator converges to the true value. The same economic objective used to verify the input constrained CL-DRTO in Figure 3.11 will be used here.

This work proposes an adjustment to a current technique: embedding the PI layer dynamics within Kalman filter estimator. With this adjustment, the models of the optimizer and estimator will take controller dynamics into account. The CL-DRTO will also use constraints to avoid input clipping, a vulnerability of the Kalman filter technique prior to this work.

In this economic evaluation, the goal will be to achieve the economic optimum as quickly as possible and with the fewest deviations from the optimal path as possible. This evaluation also pushes the inlet flowrate (F) to saturate at its maximum value of 3 kmol/m^3 for the entire simulation.

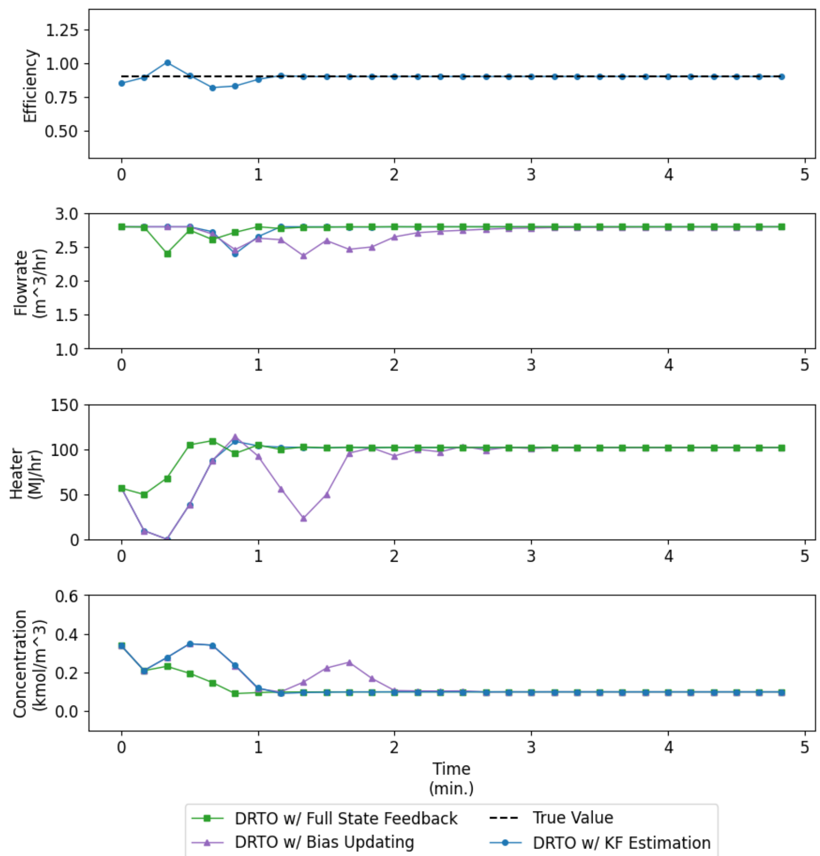

Figure 3.12: Efficiency estimation and revenue input trajectories of the new Kalman filter strategy against bias updating and full state feedback

Figure 3.12 agrees with expectations; the full state feedback scenario (Strategy 3) performs best, reaching optima fastest and smoothest and with smaller deviations. The KF estimator performs substantially better than bias updating, even while the estimator is converging. Eventually all three strategies converge to the same steady state values. These findings indicate the Kalman filter outperforms bias updating with the added benefit of having access to the unmeasured parameter, efficiency. If any further calculations based on the unmeasured states/parameters are needed, bias updating would be unable to provide these values. This makes the Kalman filter particularly valuable for unmeasurable parameters that need to be monitored for safety or profitability. Notably, the DRTO using a Kalman filter performed well even before the efficiency parameter fully converged. One hurdle for input-constrained

CL-DRTO is the use of predictions based on estimated states that have not converged yet; however these findings indicate that even before convergence, these estimates can be used for effective control. Recall that while not shown in Figure 3.12, the integral terms I_{C_A} and I_T are also being estimated by the Kalman filter here, and their success in predicting the PI controller limits indicates that the Kalman filter is also capable of estimating states as well as parameters even before complete convergence. The quantitative difference in economic performance between the 3 linear strategies is given in Table 3.6. The objective value was calculated by integrating the economic objective function over the length of the simulation.

Table 3.6: Economic objective value of the linear DRTO strategies

Strategy	Method	Objective Value
1	Kalman filter estimation	\$27,024
2	Bias updating	\$26,293
3	Full state feedback	\$27,130

Note here that the objective value in Equation 3.29 minimizes a negative value, however for demonstration the sign of this value is reversed and reported as a "\$" value, as it represents the profitability of the process. As expected, the plant performs best when all states and parameters are measurable. However the proposed strategy shows a 2.8% increase in profitability compared to bias updating, and only a 0.3% decrease in performance from full state feedback; this is especially significant as input saturation tends to cause significantly worse performance in estimator setups. Given these economic results, the proposed estimator can be expected to perform at least as well as bias updating with the added benefit of solving for missing states and parameters for use in monitoring elsewhere in the plant.

3.5.4 Target Tracking Performance

While the proposed strategy may be able to reach economic optima efficiently, in practice the optimal value will change as a function of business goals, supply and demand, and many other factors. The robustness of the strategy must be verified, and to this end the ability of the proposed strategy to track changing targets will be demonstrated alongside the other strategies. Equation 3.33 will be used as the target tracking objective function, requiring the plant to meet targets for temperature and concentration. To test how each strategy performs under changing goals, the concentration targets are stepped above the nominal values, then below the nominal values, before returning the target to nominal values. This "doublet test" consists of 3 connected step tests and should demonstrate system dynamics above and below nominal conditions. As the temperature drops when concentration rises, the temperature targets will step in the opposite direction of the concentration targets for the doublet test. To demonstrate the effect of input saturation, the first step of concentration targets is significantly larger and unreachable for the system. This will cause inputs to saturate, and the CL-DRTO will need to predict these limits within the optimizaton problem to avoid mismatch issues.

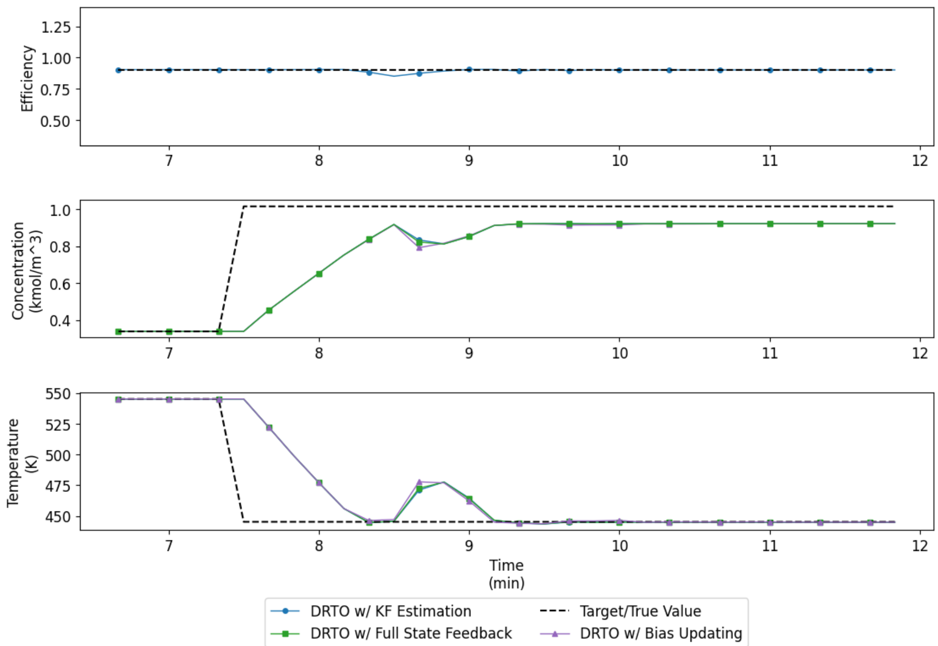

Figure 3.13: Efficiency estimation and target trajectories comparing the proposed Kalman filter with other methods

The dynamics of the three strategies above were remarkably similar. To observe the minor differences, the Figure 3.13 only shows the first step of the doublet test which contains the infeasible target. The full doublet test is not included as the dynamics of the three strategies were nearly identical for the remainder of the simulation. Even looking at this subsection there are only minor differences in the control strategies. The Kalman filter with CL-DRTO performed well even undergoing its vulnerability of saturated inputs. As with the economic performance, the objective function was integrated over the length of the simulation for each control strategy and is given in Table 3.7 for a quantitative comparison.

Table 3.7: Target tracking objective value of the linear DRTO strategies

Strategy	DRTO Method	Objective Value
1	Kalman filter estimation	16,413
2	Bias updating	16,434
3	Full state feedback	16,391

The objective values in Table 3.7 confirm that this proposed version of Kalman filter estimation performs nearly as well as full state feedback, and outperforms bias updating by a narrow margin. Large changes in targets, and even input saturation is unable to deter the convergence of the estimator on its estimated states. As expected, full state feedback performed best, but the narrow margin of improvement in Table 3.7 is likely due to bias terms and estimates converging to their true values relatively quickly over the longer simulation time used here. These findings demonstrate that even in a highly constrained system with frequent target changes the Kalman filter can still exhibit stable dynamics using a CL-DRTO that constrains the predicted PI control actions.

This rapid convergence time is a useful feature of the proposed solution; a significant drawback of estimation techniques is the poor dynamics during convergence of the estimated states and parameters, particularly when the changing targets perturb the estimated values. However these findings indicate that the Kalman filter can converge rapidly, and under these conditions the effects of mismatch during convergence are minimal.

Thus far the proposed Kalman filter has modestly outperformed bias updating in economic and tracking performance. While the use of estimated states and parameters may be desirable in certain circumstances, there is a significant financial benefit to using the Kalman filter over bias updating: filtering noise.

3.5.5 Handling Noise

Bias updating corrects for plant model mismatch in each iteration, but if the process is significantly noisy, these corrections are unable to converge to a single value effective for the whole process, and as a result bias updating handles noisy processes poorly. The Kalman filter, on the other hand, has information about the noise covariance, and is able to correct the model prediction using that information. In this case study where the model describing the plant dynamics is perfectly known (even if the values of certain states/parameters are not) the Kalman filter is especially effective at filtering out noisy measurements. To investigate these issues, four noise tests were run on the economic objective setup. The amplitude of the noise factor for these four tests is multiplied by 1, 5, 10, and 50, respectively. Each data point on Figure 3.14 represents a complete simulation, with the economic objective function integrated over the entire length of the simulation.

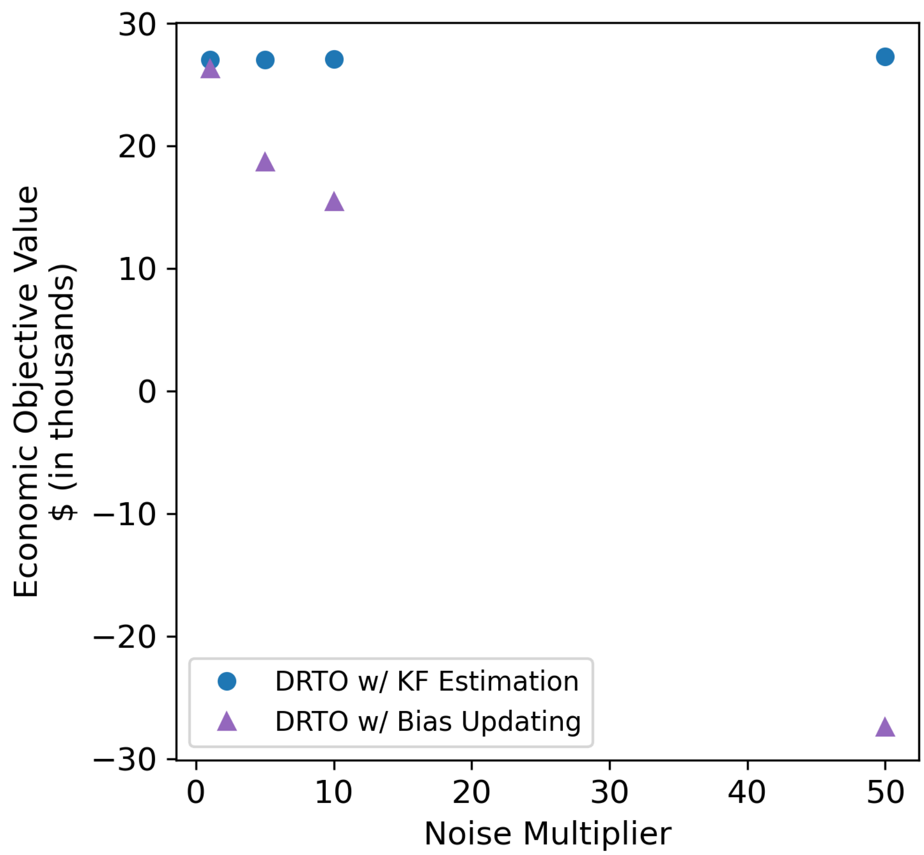

Figure 3.14: The effect of noise on the economic objective

Figure 3.14 shows as noise increases, the Kalman filter uses the predetermined noise covariance to filter out increasing levels of noise. The percent change for the Kalman filter from the first to the last test was only 1.03%. By contrast, bias updating nearly tripled its minimized value over the four noise tests. Under a low degree of noise, the Kalman filter with DRTO input constraints performs very similarly to bias updating with the same constraints. However, as noise is increased, the benefit of a Kalman filter becomes apparent as it filters out even significant levels of noise while the bias updating strategy struggles to respond to the noise. Bias updating can address significant noise by calculating the bias using the noisy measurements, then filtering the bias using an exponential filter on the noise [12].

3.5.6 Nonlinear Verification

Linear models are excellent at approximating conditions near steady state, however as the system moves further from these conditions, the approximation becomes less accurate. To verify that the proposed solution functions in systems with nonlinearity, or when linear approximations are infeasible, a nonlinear system was evaluated that uses nonlinear models, a nonlinear plant, and an extended Kalman filter (EKF) that takes into account nonlinearity.

As mentioned in Section 3.1.8, the KF can be inadequate in approximating nonlinear systems, and an EKF can help improve performance in these cases by approximating the nonlinear behavior of the plant.

Figure 3.15 demonstrates that the proposed solution can also effectively control a nonlinear plant using nonlinear models and an EKF estimator. This control was achieved despite input saturation. As was observed in the linear system, the nonlinear system demonstrated poor control while the efficiency parameter was converging, however it only took less than 2 minutes in the simulation to converge, suggesting that the effect of this poor control would be minimal. Most importantly, notice that even before the estimator converges the lower bound constraint on C_A is satisfied. This suggests that even in nonlinear systems, and even before the estimator converges, the proposed solution can effectively achieve optimal results even in the presense of saturated inputs.

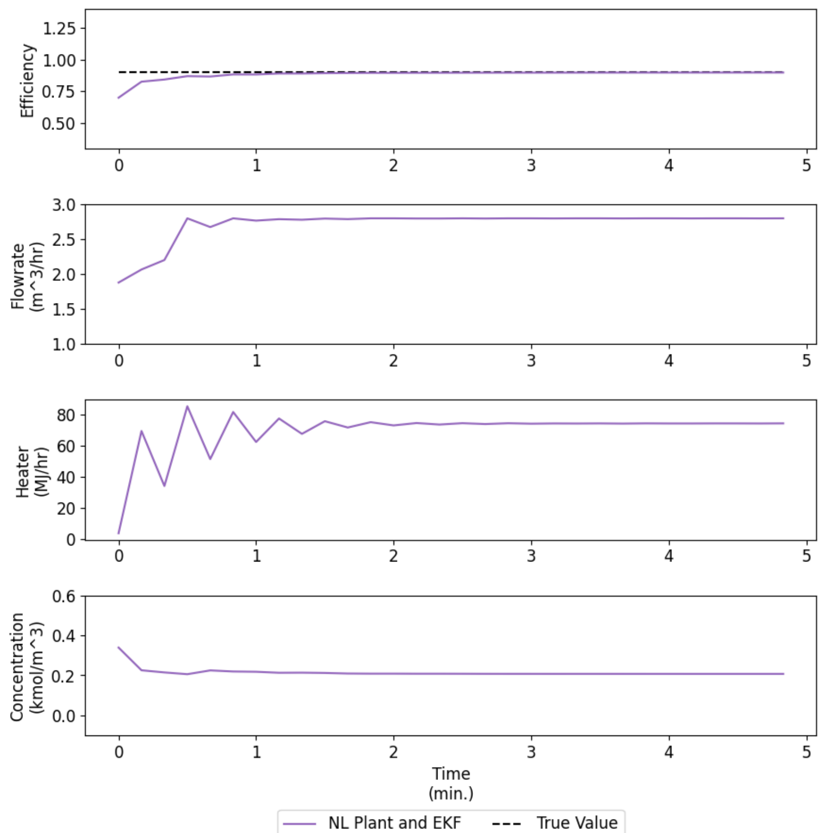

Figure 3.15: Efficiency estimation and revenue input trajectories of the EKF and nonlinear models controlling a nonlinear plant

Here the nonlinear optimum runs the system at a lower temperature (using the heater less) and does not produce as much of the product as the linear setup. As this setup controls a nonlinear plant, a quantitative comparison between these results and those of the linear plant may not be meaningful. Several qualitative comparisons can be drawn though, such as the rapid convergence time, accurate parameter estimation, and stable optimal values chosen by the economic DRTO objective. In these areas, the results in Figure 3.15 suggest that the proposed solution performs well in a nonlinear framework.

As with the linear system, it is desirable to confirm that the estimator's dynamics are favorable under changing targets, as this more closely represents practical economic operation with transient goals and constraints. The nonlinear system is tested under the same conditions as the linear one, with the exception of a slight change to the temperature targets to force input saturation to occur.

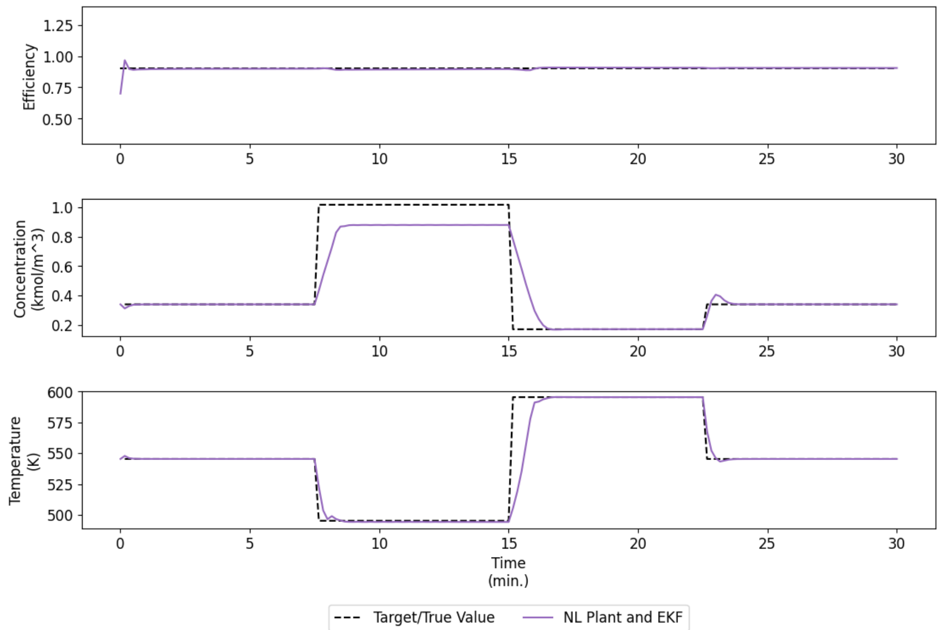

Figure 3.16: EKF and nonlinear models controlling a nonlinear plant under a target tracking objective

As shown in Figure 3.16, DRTO using an extended Kalman filter demonstrates excellent control even during input saturation. Instead of being thrown off the estimated value like the system without a Constraint Aware DRTO in Figure 3.11, the proposed solution is unaffected by the input saturation and maintains excellent control and estimation during and after the input saturation. See Equation 3.28 for the target tracking objective function used here. These findings suggest the proposed solution can be expected to track targets well under input saturation even in a nonlinear framework.

3.6 Vulnerabilities

While adding input constraints to the DRTO offers several advantages to Kalman filter estimation, there are also instances where it may not be an appropriate addition. For example, DRTO input constraints rely on a model of the PI-level behavior to predict how the PI-level responds to set-points from the DRTO. Therefore the efficacy of this method is a function of how good the PI-level model is. The estimator is also only effective if the estimated states and unknown parameters are observable using the given measurements. Systems without sufficient measurements may need alternative strategies to compensate for estimated states and unknown parameters that aren't observable. Additionally, Kalman filter estimation requires a linearized model and good nominal values for $P_{(0|0)}$, W, and V to properly converge on estimated states and parameters. If significant changes are made to constraints or the dynamics of the process have significantly changed, the Kalman filter may need a new $P_{(0|0)}$ and the linear KF may need a new linear model to use. In systems where this is a common occurrence this can become economically and computationally prohibitive. For these reasons the Kalman filter with DRTO input constraints should be considered for systems with high noise, missing states/parameters, and a reliable model of plant dynamics and PI-control dynamics.

3.7 Chapter Summary

This chapter explores the performance of Kalman filter estimation coupled with CL-DRTO and a PI controller on a simple CSTR. The proposed solution mitigated the effects of input saturation by constraining the predicted response of the PI controller within the CL-DRTO. Including the controller dynamics in the DRTO offers an alternative method to achieving closed loop control with DRTO, and including controller dynamics in the Kalman filter allows the estimator the new feature of including controller behavior in its predictions. These effects were also studied on a nonlinear plant, controlled using nonlinear control architecture and an extended Kalman filter, and demonstrated a technique for achieving good control of a highly nonlinear process. The performance of the estimator was compared to bias updating; one method of dealing with plant-model mismatch in traditional DRTO and the standard approach in CL-DRTO. Not only did the Kalman filter outperform bias updating in both economic and target tracking objectives, the Kalman filter also obtained the true value of the unknown parameter: a result unachievable by bias updating. This chapter also demonstrated how performance rapidly declines in bias updating as noise increases, while the Kalman filter was able to achieve good control despite significant noise.

References

[1] T. Tosukhowong, J. M. Lee, J. H. Lee, and J. Lu. "An introduction to a dynamic plant-wide optimization strategy for an integrated plant". In: *Computers & Chemical Engineering* 29.1 (2004), pp. 199–208.

[2] J. Matias, J. P. Oliveira, G. A. Le Roux, and J. Jäschke. "Steady-state real-time optimization using transient measurements on an experimental rig". In: *Journal of Process Control* 115 (2022), pp. 181–196.

[3] Y. Song and J. W. Grizzle. "The Extended Kalman Filter as a Local Asymptotic Observer for Nonlinear Discrete-Time Systems". In: *1992 American Control Conference*. 1992, pp. 3365–3369.

[4] E. Walter, L. Pronzato, and J. Norton. *Identification of parametric models from experimental data*. Vol. 1. 2. Springer, 1997.

[5] J. O. Matias and G. A. Le Roux. "Real-time Optimization with persistent parameter adaptation using online parameter estimation". In: *Journal of Process Control* 68 (2018), pp. 195–204.

[6] C. Paige. "Properties of numerical algorithms related to computing controllability". In: *IEEE Transactions on Automatic Control* 26.1 (1981), pp. 130–138.

[7] S. J. Julier and J. K. Uhlmann. "Unscented filtering and nonlinear estimation". In: *Proceedings of the IEEE* 92.3 (2004), pp. 401–422.

[8] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. "CasADi: a software framework for nonlinear optimization and optimal control". In: *Mathematical Programming Computation* 11 (2019), pp. 1–36.

[9] S. Fuller, B. Greiner, J. Moore, R. Murray, R. van Paassen, and R. Yorke. "The Python Control Systems Library (python-control)". In: *60th IEEE Conference on Decision and Control (CDC)*. IEEE. 2021, pp. 4875–4881.

[10] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. "SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers". In: *ACM Transactions on Mathematical Software (TOMS)* 31.3 (2005), pp. 363–396.

[11] X. Li, L. Zhang, M. Nakaya, and A. Takenaka. "Application of economic MPC to a CSTR process". In: *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*. IEEE. 2016, pp. 685–690.

[12] J. F. Forbes and T. E. Marlin. "Model accuracy for economic optimizing controllers: The bias update case". In: *Industrial & engineering chemistry research* 33.8 (1994), pp. 1919–1929.

Chapter 4

State and Parameter Estimation in CL-DRTO and MPC-Controlled Plants

4.1 Preliminaries

4.1.1 Background

Control architecture typically uses a linearized version of nonlinear systems, as these approximations tend to be accurate enough for practical use. These nonlinear systems take the form shown in Equation 4.1

$$\begin{aligned} \dot{\boldsymbol{x}}_{nl} &= \frac{d\boldsymbol{x}_{nl}}{dt} = f(\boldsymbol{x}_{nl}(t), \boldsymbol{u}_{nl}(t)) \\ \boldsymbol{y}_{nl}(t) &= g(\boldsymbol{x}_{nl}(t)) \end{aligned} \tag{4.1}$$

where the change in the state vector ($\dot{\boldsymbol{x}}_{nl}$) is a function of the current state values $\boldsymbol{x}_{nl}(t)$ and the current value of the inputs $\boldsymbol{u}_{nl}(t)$, which tend to be controller actions. The current state values are also mapped to an vector of the measured outputs ($\boldsymbol{y}_{nl}(t)$). The subscript "*nl*" is used to denote the nonlinear vector is not in deviation form. As the majority of this work deals with linearized matrices and vectors in deviation form, this will be considered the default condition when "*nl*" is not added a subscript. In practice this nonlinear model can be computationally expensive to simulate, and it is desirable to approximate its behavior using a linearized state space model, shown in Equation Set 4.2.

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x}(t) + B\boldsymbol{u}(t) \\ \boldsymbol{y} &= C\boldsymbol{x}(t) \end{aligned} \tag{4.2}$$

Here the nonlinear model in Equation 4.1 is approximated using state space model matrices A, B, and C. These matrices convey how the current states, controller values, and measurements affect system dynamics. An additional equation is added to only select the measured states and apply any measurement noise where applicable. The resulting vector of measured states ($\boldsymbol{y}$) is used as feedback in closed-loop control. The equations are linearized using a steady state value, and their approximation is most accurate while the system remains close to that steady state condition. However, the difference in dynamics when the system steps away from steady state is typically acceptable, and this error tends to be worth the simplicity of a linear model.

4.1.2 Deviation Form

When generating the state space model matrices shown in Equation Set 4.2, a steady state condition must be chosen to linearize around. When using these state space model matrices in a linear equation such as Equation 4.2, the state ($\boldsymbol{x}(t)$), input ($\boldsymbol{u}(t)$), and any other associated vectors must be in deviation variable form, where the steady state values have been subtracted from the actual values. For example, in a system with a state vector containing n states and an input vector containing m inputs, the vectors are placed in deviation form following the formulas in Equation Set 4.3.

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1' \\ x_2' \\ \vdots \\ x_n' \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} - \begin{bmatrix} x_{1,ss} \\ x_{2,ss} \\ \vdots \\ x_{n,ss} \end{bmatrix} \quad \boldsymbol{u}(t) = \begin{bmatrix} u_1' \\ u_2' \\ \vdots \\ u_m' \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} - \begin{bmatrix} u_{1,ss} \\ u_{2,ss} \\ \vdots \\ u_{m,ss} \end{bmatrix} \tag{4.3}$$

It is common in this setting to place an apostrophe over the vector to denote it as being in deviation variable form, however as virtually all vectors in this study will be in this form, that notation will not be used here. Instead it can be assumed that any vectors interacting with state space model matrices can be assumed to be in deviation variable form unless stated otherwise.

4.1.3 Plant and Model

A model of the plant will be used in control layers to predict future behavior. Model vectors will be denoted with the subscript "m" as in Equation Set 4.4, and the actual plant dynamics will be simulated using the linear model of Equation Set 4.5 and subscript "p". To observe the effect of an estimator on the system, Equation Set 4.4 and 4.5 will be slightly modified so they are functions of the estimated parameters as well. The new component added ($D_\theta \boldsymbol{\theta}(t)$)) depicts the effect of the estimated parameter(s) ($\boldsymbol{\theta}(t)$) on the states using matrix D_θ.

$$\begin{aligned} \dot{\boldsymbol{x}}_m &= A_m \boldsymbol{x}_m(t) + B_m \boldsymbol{u}(t) + D_{\theta,m} \boldsymbol{\theta}(t) \\ \boldsymbol{y}_m &= C_m \boldsymbol{x}_m(t) \end{aligned} \tag{4.4}$$

Here $\boldsymbol{x}_m$ and $\boldsymbol{y}_m$ represent the states and their measured values respectively. Matrices A_m, B_m, C_m and D_{θ_m} will be used to model the plant dynamics. The linearization that creates these matrices is described in Section 4.2. For the purposes of demonstrating the control architecture, a linear plant will be used to receive measurements and apply control actions.

$$\begin{aligned} \dot{\boldsymbol{x}}_p &= A_p \boldsymbol{x}_p(t) + B_p \boldsymbol{u}(t) + D_{\theta,p} \boldsymbol{\theta} \\ \boldsymbol{y}_p &= C_p \boldsymbol{x}_p(t) \end{aligned} \tag{4.5}$$

For the purposes of this case study, the true values of the parameters will not change with time, and thus the parameter vector $\boldsymbol{\theta}$ will be treated as constant. This is not always the case, but for this study the value will be treated constant within the plant.

When simulating control architecture, it is often desirable to convert the continuous models above into discretized models that can be tracked iteratively. To this end, Equation Set 4.4 is discretized using the same time interval as the DRTO optimization interval (Δt_{opt}). Choosing this optimization interval requires careful consideration, as the optimizer is greatly affected by the choice of this interval length. Tosukhowong et al. [1] suggests a method for choosing an effective Δt_{opt} to use in DRTO. The DRTO will keep set-points constant across the optimization interval. The equations describing the model are given in their discretized form in Equation set 4.6 below.

$$\begin{aligned} \boldsymbol{x}_{m,(k+1)} &= A_{m,d}\boldsymbol{x}_{m,k} + B_{m,d}\boldsymbol{u}_k + D_{\theta m,d}\boldsymbol{\theta}_k \\ \boldsymbol{y}_{m,(k+1)} &= C_{m,d}\boldsymbol{x}_{m,(k+1)} \end{aligned} \tag{4.6}$$

In this study, separate time steps will be denoted with "k" and the subscript "d" will be applied to any components that have been discretized. Plants and controllers typically require much faster decisions than the complex DRTO layer is capable of reasonably supplying. To address this, the DRTO is often run at a lower frequency than the MPC control level. This becomes important in the discretization step, as the discretization interval chosen for the plant and MPC layers will be smaller than the disretization interval chosen for the DRTO layer. To address this, the MPC model and plant equations are discretized using $\Delta t_{\text{mpc/plant}}$, which is equal to $\frac{\Delta t_{\text{opt}}}{N}$ where N represents the the number of control actions taken for a given DRTO set-point. The discretized form of the plant from Equation Set 4.5 is given below in Equation Set 4.7.

$$\begin{aligned} \boldsymbol{x}_{p,(k+1)} &= A_{p,d}\boldsymbol{x}_{p,k} + B_{p,d}\boldsymbol{u}_k + D_{\theta p,d}\boldsymbol{\theta}_k \\ \boldsymbol{y}_{p,(k+1)} &= C_{p,d}\boldsymbol{x}_{p,(k+1)} \end{aligned} \tag{4.7}$$

The MPC handles mismatch using a disturbance term instead of estimated values, and as such its model requires special consideration.

4.1.4 MPC

Model predictive controllers pose several advantages over simple PI control. MPC predicts future dynamics across a set horizon length, using predictions to calculate optimal control actions. In multi-input multi-output (MIMO) systems this is especially useful, as the MPC can consider multi-variable interaction in the model and avoid the need for multiple PID controllers and the need for controller pairings. MPC can also include constraints in its optimization. The MPC controller also uses a disturbance term to update its control actions, allowing it to adjust for plant-model mismatch in real-time. This work will use the MPC equations proposed by Li and Swartz [2]. In this setup, the disturbance term is added as a separate term in the MPC calculations, Incorporating the disturbance terms directly into the model ensures the effect of the disturbance is included in predictions, MPC control, and DRTO predictions of the MPC model.

To accomplish this, this study will modify the implementation by Li and Swartz [2] using augmented state space model matrices described in Maciejowski [3]. The conventional state vector $\boldsymbol{x}_k$ will include an additional row for each disturbance term, resulting in a new vector $\tilde{\boldsymbol{x}}_{MPC,k}$. By using the Maciejowski implementation, this work can apply the disturbance term in model predictions. The disturbance terms are assumed constant in future predictions, and this is accomplished by adding identity matrices and zero matrices to the state space model matrices. Note that unlike the DRTO model, the MPC model does not receive estimated parameters from the Kalman filter. Instead, the MPC model starts with nominal values of those parameter values and uses its disturbance term to correct for offset. The Maciejowski implementation of the model is shown in Equation Set 4.8.

$$\begin{aligned}
\tilde{\boldsymbol{x}}_{mpc,(k+1)} &= \begin{bmatrix} \boldsymbol{x}_{m,(k+1)} \\ \boldsymbol{d}_{(k+1)} \end{bmatrix} = \begin{bmatrix} A_d & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m,k} \\ \boldsymbol{d}_k \end{bmatrix} + \begin{bmatrix} B_d \\ 0 \end{bmatrix} \boldsymbol{u}_k \\
\tilde{\boldsymbol{y}}_{mpc,(k+1)} &= \begin{bmatrix} C_d & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m,(k+1)} \\ \boldsymbol{d}_{(k+1)} \end{bmatrix}
\end{aligned}$$

Here the model from Equation Set 4.4 is expanded to include disturbance terms and matrices that describe how they affect the states of the model. The above equations can be expressed as Equation Set 4.8 below.

$$\begin{aligned}
\tilde{\boldsymbol{x}}_{mpc,(k+1)} &= A_{mpc,d}\tilde{\boldsymbol{x}}_{mpc,k} + B_{mpc,d}\boldsymbol{u}_k \\
\tilde{\boldsymbol{y}}_{mpc,(k+1)} &= C_{mpc,d}\tilde{\boldsymbol{x}}_{mpc,k}
\end{aligned} \tag{4.8}$$

Here the model states ($\boldsymbol{x}_{m,k}$) combine with the disturbance terms ($\boldsymbol{d}_k$) to form the state vector used in the MPC algorithm ($\tilde{\boldsymbol{x}}_{mpc,k}$). This augmentation can be performed after the discretization step, as the disturbance values are defined in discrete terms [3]. This is not always the case however, and using different matrix values can induce mismatch. This situation will be studied later in this work. The augmented shape of these new state space model matrices is determined by the number of disturbance terms (n_d). The identity matrix (I) within $A_{mpc,d}$ is of shape $n_d \times n_d$, and the identity matrix within $C_{mpc,d}$ is shaped $n_m \times n_d$; as all measured states are assigned a disturbance term, this matrix must be square. This changes the shapes of the state space model matrices A_d, B_d, and C_d to $(n_x + n_d) \times (n_x + n_d)$, $(n_x + n_d) \times n_u$, and $n_m \times (n_x + n_d)$, respectively. In this context, n_x, n_u, and n_m represent the number of states, inputs, and measured states, respectively. Once a model is developed, the new state space model matrices are used to initialize the matrices used in the MPC algorithm.

The unconstrained MPC problem can be analytically solved for control actions over the entire prediction horizon at once using a composite matrix. This composite matrix takes the matrices defined in Equation Set 4.8 and adds an additional dimension of time, adding additional rows of matrices for each step in the horizon. This methodology, described in Li and Swartz [2], is more complex to build, but many of the components can be built before the optimization problem begins, reducing the computational expense. As these matrices can be large and intricate, their exact structures and descriptions are detailed in the Appendix. The appendix also details the algorithm the MPC uses to take these matrices and compute a set of control actions based on the states, set-points, and control actions. The change in control actions calculated for the algorithm above are determined using the formulation from Li and Swartz [2] given in Equation 4.9.

$$\Delta\underline{\boldsymbol{u}}_k = \underline{K}_1(\underline{\boldsymbol{y}}_k^{SP} - (\underline{B}_1 X_k + \underline{B}_2 \boldsymbol{u}_{k-1}))) \tag{4.9}$$

Composite matrices are marked with an underbar, denoting a larger shape that accounts for how the values evolve over the prediction and control horizons. Equation 4.9 describes how the change in controller values is calculated for the MPC algorithm detailed in the Appendix.

The change in input action ($\Delta\underline{\boldsymbol{u}}_k$) is updated every time step and is a function of the set-points (($\underline{\boldsymbol{y}}_k^{SP}$), the states ($X_k$), and the control actions ($\boldsymbol{u}_{k-1}$)). The constant components used in Equation 4.9 ($\underline{K}_1$,$\underline{B}_1$,$\underline{B}_2$) and the other constants used in the algorithm ($\underline{Q}$, $\underline{R}$, $\underline{I}_L$, $\underline{A}$) are calculated once before the simulation begins and determine the attributes of the MPC layer. The Appendix contains a detailed description of each of these components as well as a description of Equation 4.9 and the other equations governing the MPC algorithm.

The MPC algorithm can be used to track set-points, but cannot optimize higher-level objectives such as profitability. This higher-level control can be achieved using a DRTO layer.

4.1.5 CL-DRTO

Tosukhowong et al. [1] proposed a DRTO architecture which utilizes an open-loop prediction of the plant dynamics. The DRTO used by Tosukhowong et al. [1] optimized a set of controller actions, but as the MPC formula for $\mathbf{u}_k$ is built into the optimizer, and $\boldsymbol{u}_k$ is a function of the set-points, the DRTO can optimize the set-points directly. This poses several advantages, including the ability to constrain both the set-points and the control actions that result from them. This enables the DRTO to overcome mismatch issues during input saturation, a subject which will be covered in more detail in Section 4.1.9. A closed-loop DRTO strategy of this type was proposed by Jamaludin and Swartz [4] in which input-constrained MPC was considered. To simply control structure, all states, inputs (including the parameters that change with time) are placed in deviation variable form. This is done by subtracting the steady state conditions used in linearization from the current values. A similar closed-loop DRTO strategy is followed in the present study, utilizing an unconstrained MPC formulation.

$$
\begin{aligned}
&\min_{Y_{sp}} \phi(Y_{\text{sp}}) \\
\text{s.t. } & X_{\text{m,min}} \leq X_{m,(k+1|k)} \leq X_{\text{m,max}} \\
& Y_{\text{sp,min}} \leq Y_{sp} \leq Y_{\text{sp,max}} \\
& U_{\text{min}} \leq U \leq U_{\text{max}} \\
& U = \begin{bmatrix} \boldsymbol{u}_k^T & \boldsymbol{u}_{(k+1)}^T & \dots & \boldsymbol{u}_{(k+M-1)}^T \end{bmatrix}^T \\
& Y_{\text{sp}} = \begin{bmatrix} \boldsymbol{y}_{\text{sp},k}^T & \boldsymbol{y}_{\text{sp},(k+1)}^T & \dots & \boldsymbol{y}_{\text{sp},(k+M-1)}^T \end{bmatrix}^T \\
& X_{m,(k+1|k)} = \begin{bmatrix} \boldsymbol{x}_{m,(k+1|k)}^T & \boldsymbol{x}_{m,(k+2|k)}^T & \dots & \boldsymbol{x}_{m,(k+P|k)}^T \end{bmatrix}^T
\end{aligned}
\tag{4.10}
$$

The DRTO layer consists of Equation Set 4.10 and the model in Equation Set 4.6. The DRTO layer has been discretized using the larger time interval Δt_{opt}. Here the objective value (ϕ) is minimized by changing Y_{sp}, a composite vector containing set-point vectors chosen across the control horizon. In the MPC subproblems, the set-points in $\underline{\boldsymbol{y}}_k^{SP}$ are held constant across the control horizon, however in the DRTO subproblems, the set-points in Y_{sp} are changed across the control horizon using the larger DRTO layer time intervals to minimize the objective function. This gives the MPC N control actions between the set-points chosen by the DRTO. See Section 4.1.3 for more detail. Equation Set 4.10 is governed by several composite vectors, which span the prediction or control horizons. Composite vectors $X_{m,(k+1|k)}$, Y_{sp}, and U contain the model states, set-points, and control actions, respectively. The control horizon is set at length M, and in setups where the length of the prediction horizon P is greater than M, $\boldsymbol{u}$ is considered constant for predictions beyond the control horizon.

The control actions within the composite matrix U are calculated using Equation 4.9 which is explained in more detail in Section 4.2. These values are calculated within the DRTO layer allows the optimizer to predict the controller's response to the set-points chosen by the DRTO. For example, in a CL-DRTO with $P = 3$ and $M = 2$, Figure 4.1

shows how the CL-DRTO would be structured to achieve closed-loop optimization with integrated MPC control.

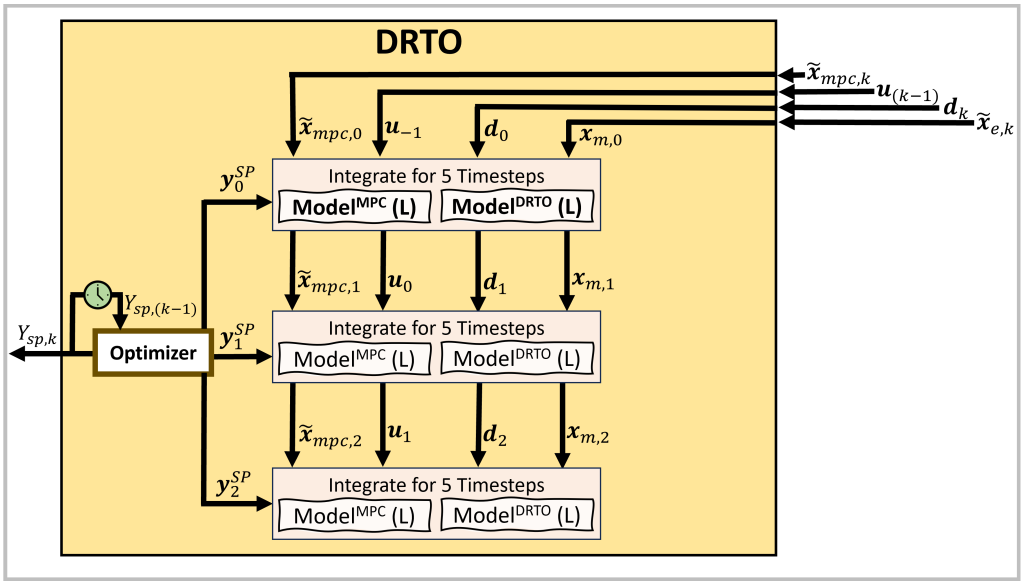

Figure 4.1: Example CL-DRTO internal structure with MPC prediction

In Figure 4.1, the states from the MPC layer ($\tilde{\boldsymbol{x}}_{mpc,k}$), the disturbance between the MPC prediction and the plant measurements ($\boldsymbol{d}_k$), the estimated states from the Kalman filter ($\tilde{\boldsymbol{x}}_{e,k}$), and the previous control action chosen by the MPC ($\boldsymbol{u}_{(k-1)}$) are all used to initialize the DRTO layer. They become $\tilde{\boldsymbol{x}}_{mpc,0}$, $\boldsymbol{d}_0$, $\boldsymbol{x}_{m,0}$, and $\boldsymbol{u}_{-1}$, respectively. In the example from Figure 4.1, these states initialize a set of five time steps targeting the fixed set-point $\boldsymbol{y}_0^{SP}$. The DRTO layer uses the equations from all the integration blocks to build an NLP and they are sent to the solver for simultaneous optimization. Within the integration block, the MPC model calculates $\boldsymbol{u}$ and updates $\tilde{\boldsymbol{x}}_{mpc}$, the DRTO model updates $\boldsymbol{x}_m$, and the difference between each model updates $\boldsymbol{d}$. After the fifth time step, the final values are sent to initialize the next integration block using a new set-point from the optimizer. Note here that although the DRTO receives a state vector from the estimator containing the estimated parameter values, these are not updated within the DRTO layer. These are treated as constant values throughout the integration blocks, and as such $\boldsymbol{x}_m$ only contains the estimated states.

Using the internal structure from Figure 4.1, the optimizer given in Equation Set 4.10

can optimize Y_{sp} for a given set of initial inputs. The states within this prediction horizon are stored in the composite matrix $X_{m,(k+1|k)}$. The notation $(k+1|k)$ used in Equation Set 4.10 denotes that the variable contains values at time step $k+1$ and was based on values from time step k. Typically, some states and parameters are unknown when developing control architecture; an estimator can then be used to predict the true value of these unknown components and enable effective control of the system despite unknowns.

4.1.6 Kalman filter Estimation

Kalman filter estimation can calculate the estimated states and unknown parameters of a model, a helpful tool in systems where not all desired states are measurable or where measurements may be inaccurate due to noise or other disturbances. The estimator treats unknown parameters as pseudo-states, adding them as additional rows to the nonlinear equations in Equation Set 4.2. This new, larger state vector ($\tilde{\boldsymbol{x}}_e$) is able to update its estimation of the states and unknown parameters as described in the algorithm of Equation Set 4.14. This augmented setup is given in Equation Set 4.11.

$$\begin{aligned} \dot{\tilde{\boldsymbol{x}}}_e &= A_e\tilde{\boldsymbol{x}}_e(t) + B_e\boldsymbol{u}(t) \\ \tilde{\boldsymbol{y}}_e &= C_e\tilde{\boldsymbol{x}}_e(t) \end{aligned} \tag{4.11}$$

where $\tilde{\boldsymbol{x}}_e$ represents the values of the known states as well as the estimated pseudo-states. The additional rows to these matrices mean the state space model matrices A_e,B_e,C_e are also larger to describe how these changing pseudo-states interact with the states and inputs. As the dynamics of estimated values are tracked using these additional rows, the $D_{\theta_m}\boldsymbol{\theta}(t)$ component found in Equation Set 4.4 is not needed. This new structure is given in more detail in Equation Set 4.12 below.

$$\begin{aligned}
\begin{bmatrix} \dot{\boldsymbol{x}}_m \\ \dot{\boldsymbol{x}}_{ps} \end{bmatrix} &= \begin{bmatrix} A_m & A_{ps_1} \\ A_{ps_2} & A_{ps_3} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_{ps} \end{bmatrix} + \begin{bmatrix} B_m \\ B_{ps} \end{bmatrix} \boldsymbol{u} \\
\begin{bmatrix} \boldsymbol{y}_m \\ \boldsymbol{y}_{ps} \end{bmatrix} &= \begin{bmatrix} C_m & 0 \\ 0 & C_{ps} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_{ps} \end{bmatrix}
\end{aligned} \tag{4.12}$$

Here the model from Equation Set 4.4 is expanded to include the pseudo-states, given a notation of "ps" here. The new matrices and vectors included in Equation Set 4.12 describe how the estimated states and parameters affect the other states. The vector of states used in the model ($\boldsymbol{x}_m$) combine with the pseudo-states estimated by the Kalman filter ($\boldsymbol{x}_{ps}$) to form the new larger vector used in the estimator ($\tilde{\boldsymbol{x}}_e$). After creating a model of estimated states and parameters, Equation Set 4.11 is discretized using Δt_{opt}, just as in the DRTO layer, resulting in the discrete implementation in Equation Set 4.13.

$$\begin{aligned}
\tilde{\boldsymbol{x}}_{e,(k+1)} &= A_{e,d}\tilde{\boldsymbol{x}}_{e,k} + B_{e,d}\boldsymbol{u}_k \\
\tilde{\boldsymbol{y}}_{e,(k+1)} &= C_{e,d}\tilde{\boldsymbol{x}}_{e,k}
\end{aligned} \tag{4.13}$$

This work will also use the Kalman filter algorithm described by Welch and Bishop [5]. This algorithm is given in Equation Set 4.14.

$$\begin{aligned}
&1. \quad \tilde{\boldsymbol{x}}_{e,(k+1|k)} = A_{e,d}\tilde{\boldsymbol{x}}_{e,(k|k)} + B_{e,d}\boldsymbol{u}_k \\
&2. \quad P_{(k+1|k)} = A_{e,d}P_{(k|k)}A_{e,d}^T + V_k \\
&3. \quad K_{f,(k+1)} = P_{(k+1|k)}C_{e,d}^T \left(W_{(k+1)} + C_{e,d}P_{(k+1|k)}C_{e,d}^T\right)^{-1} \\
&4. \quad \tilde{\boldsymbol{x}}_{e,(k+1|k+1)} = \tilde{\boldsymbol{x}}_{e,(k+1|k)} + K_{f,(k+1)}\left(\boldsymbol{y}_{p,(k+1)} - C_{e,d}\tilde{\boldsymbol{x}}_{e,(k+1|k)}\right) \\
&5. \quad P_{(k+1|k+1)} = P_{(k+1|k)} - K_{f,(k+1)}C_{e,d}P_{(k+1|k)}
\end{aligned} \tag{4.14}$$

The algorithm above is solved iteratively, using a time step notation of "k to distinguish between executions of the Kalman filter. However the Kalman filter, like the DRTO, executes less often than the MPC and plant layers, running once every N executions of the MPC and plant layers. All values are assumed constant until the next execution of the DRTO and Kalman filter layers. The algorithm starts in Step 1 by calculating the predicted states of the plant for the next time step $k+1$. One advantage of the Kalman filter in addition to its estimation of unknown values is its ability to filter noise by means of tuning the Kalman filter gain using the process and measurement noise covariance (V and W, respectively). Loosely, if $V >> W$, the Kalman gain is typically higher and the estimates tend to conform to the current measurement ($\boldsymbol{y}_{p,(k+1)}$). On the other hand, if $W >> V$, the filter places more weight on the model predictions ($\tilde{\boldsymbol{x}}_{e,(k+1|k)}$) when updating the estimates. As a consequence, such tuning will smooth out measurement noise but decrease the filter responsiveness.

This also calculates the updated values for estimated states and the parameters treated as pseudo-states in the augmented state vector $\tilde{\boldsymbol{x}}_e$. In Step 2 the filter updates the covariance of the estimates (P) and uses this matrix to update the Kalman filter gain (K_f) in Step 3. These steps also account for the process and measurement noise (V and W, respectively). The new gain is used to improve the estimate of the next iteration's states in Step 4. In Step 5 the new gain gives an improved estimation of the covariance matrix for the next iteration. Over time, P and K_f converge to steady-state values and $\tilde{\mathbf{x}}_e$ approaches the true values, including the estimated states and unknown parameters.

4.1.7 Observability

To infer if the states and parameters can be reliably estimated from the knowledge on the past measurements, we check if the resulting model is observable or not by using the rank of the observability matrix shown in Equation Set 4.15.

$$O_e = \begin{bmatrix} C_e \\ C_e A_e \\ C_e A_e^2 \\ \vdots \\ C_e A_e^{n-1} \end{bmatrix} \tag{4.15}$$

where O_e represents the observability matrix of the system and is used to assess the observability of the estimator setup. Here n represents the combined number of states and pseudo-states. For example, in a system with 3 states, 1 estimated state and 2 estimated parameters, n would be 6. If the rank of O_e equal to n, the system is considered observable using an estimator [6].

4.1.8 Nonlinear Adaptations

DRTO and MPC predictions rely heavily on the accuracy of their model predictions, and many plants display at least some degree of nonlinearity. However it may still be desirable to use simplified linear models in the control structure, which poses an interesting challenge. The nonlinear dynamics of the plant will induce plant-model mismatch in the prediction steps of these control layers, and this work will demonstrate how the control structure handles such an inconsistency. To that end, the linearized control structure will not be altered when applied to a nonlinear plant, and instead the disturbance term of the MPC and the Kalman filter will need to adapt to the mismatch inherent in this control scenario. Ideally this will pose a practical method for implementing this architecture when nonlinear predictions are computationally expensive or otherwise infeasible to use. To apply nonlinear control layers to a nonlinear plant in a DRTO setting, an Extended Kalman filter may be needed, and an example of this can be found in [7].

4.1.9 Input Saturation

When the MPC is attempting to reach a large set-point change, the controller actions chosen may exceed their limits permitted by the plant. In this case the MPC layer will clip the inputs resulting in input saturation. Input clipping can involve boolean logic, and incorporating it into DRTO optimizers will result in mixed-integer optimization formulations. Without accounting for this behavior, the DRTO predicts behavior not possible at the plant level, and plant-model mismatch occurs. Plant-model mismatch in general can perturb and derail the estimator from its true value, resulting in poor control, as illustrated in Figure 4.2.

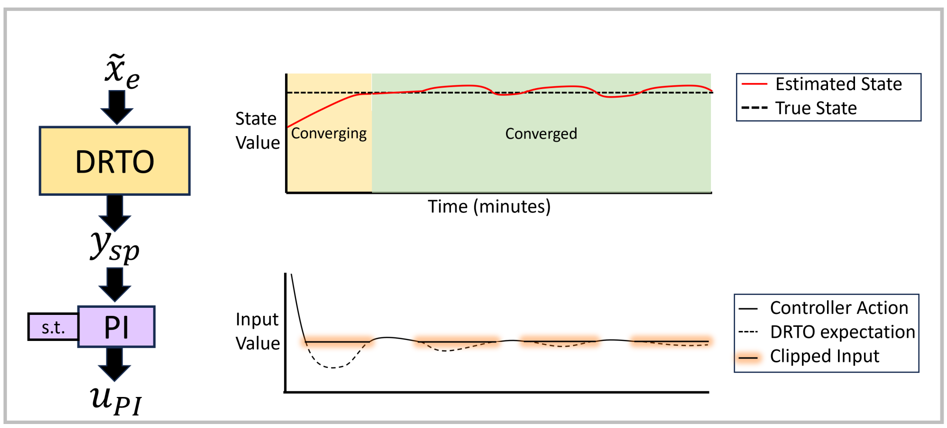

Figure 4.2: Issues with estimators and input saturation

Input saturation is common in practice, and accounting for its effect on system dynamics is critical, especially in highly constrained systems or economic objectives that drive the system to its limits. Several approaches to incorporate this Boolean behavior have been made, such as complementarity constraints in the DRTO setup used by Jamaludin and Swartz [4], however this method is complex and difficult to implement in CL-DRTO. Instead, this chapter uses CL-DRTO to predict the effect of the optimized set-points within the DRTO, and simply constrains at the DRTO level the inputs determined by the unconstrained MPC subproblems. By predicting how the MPC layer responds to a given set-point, the controller response is constrained in the DRTO so only feasible set-points are sent to the MPC layer. This input constrained

DRTO was used throughout the chapter to handle input saturation and its effects on the estimator. It is important to note that any time plant-model mismatch occurs there can be poor control and constraint violation as the optimizer sends poor set-points to the MPC layer. This suggests that while the estimator is still converging to its steady-state value, the resulting plant-model mismatch may cause input clipping. However if the Kalman filter converges relatively quickly, this method may still result in control superior to DRTO without MPC constraint knowledge. Relying on these findings, this work will use input constrained CL-DRTO to minimize the effects of plant-model mismatch due to estimator convergence.

4.2 Proposed Solution

While the individual control elements used here (DRTO, MPC, Kalman filter Estimation, etc.) are not novel on their own, using them together is indeed novel for CL-DRTO. See Section 3.2 for an explanation of the studies done on these control elements elsewhere in literature. Adding state and parameter estimation to DRTO can improve its robustness in practical use where not all variables are measurable. MPC offers a predictive optimizer, offering improved control over PI control layers. The DRTO can make more accurate predictions if the MPC dynamics are predicted within the DRTO layer. The predicted value of the MPC disturbance vector is also updated within the DRTO to improve the accuracy of the DRTO optimization. All these elements together pose a control architecture particularly robust to noise, unknown values, and input saturation.

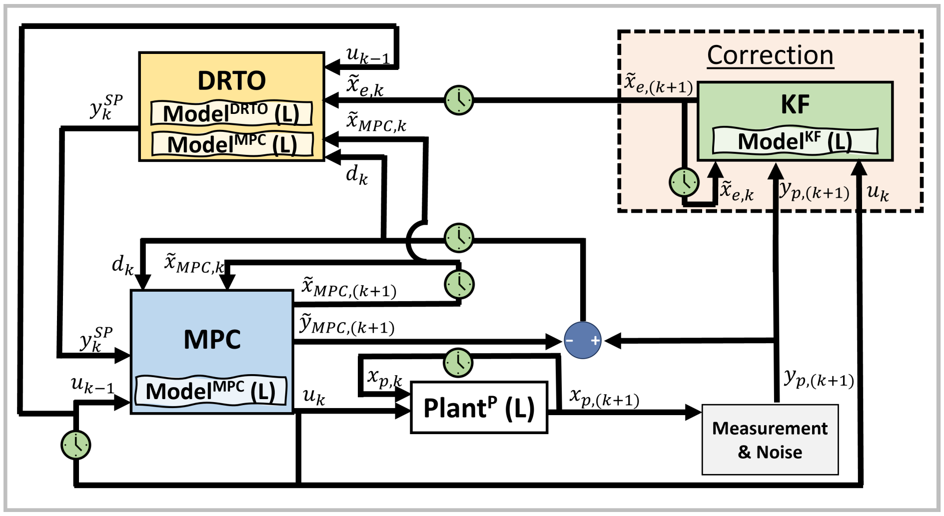

Figure 4.3: Information flow diagram of proposed control architecture using DRTO, MPC, and Kalman filter Estimation

Figure 4.3 above illustrates the flow of information in the proposed control architecture. A clock icon is used to indicate a time step increment in real-time. Note three distinct models of the process. These linear models with the superscript "DRTO", "MPC", and "KF", represent the linearized state space models for the DRTO, MPC and Kalman filter Estimation layers respectively. They are similar but have slight structural differences. For example, the Kalman filter model treats estimated parameters as pseudo-states, and the MPC model treats disturbance terms as pseudo-states. These additional rows cause slightly different structure in the linear models, but they all originate from a linearization of Equation Set 4.1.

A critical component of this solution is the incorporation of the MPC model into the DRTO layer. This allows the DRTO to simulate the expected response of the MPC to its chosen set-points. With access to this predicted response, the DRTO can constrain its set-points to the feasible region of the MPC and avoid input clipping. This is accomplished by using the calculation for $\underline{\boldsymbol{u}}_k$ from Equation Set 4.33 of the MPC algorithm. For this task, the DRTO utilizes only the first time step of the MPC response, so the composite matrices used to calculate $\underline{\boldsymbol{u}}_k$ are modified slightly to only select the data linked to the first control action. By substituting Equation 4.31 into the equation for $\Delta\underline{\boldsymbol{u}}_k$, then substituting the equation for $\Delta\underline{\boldsymbol{u}}_k$ into $\underline{\boldsymbol{u}}_k$, a formula is obtained for expressing the MPC response as a function of the DRTO set-point.

$$\begin{aligned} \boldsymbol{u}_k &= \boldsymbol{u}_{k-1} + SI_L K_1 \left[\underline{\boldsymbol{y}}_k^{SP} - (\underline{B}_1 X_k + \underline{B}_2 \boldsymbol{u}_{k-1}) \right] \\ S &= \begin{bmatrix} I_s & 0 \end{bmatrix} \end{aligned} \tag{4.16}$$

Equation set 4.16 is used to predict the MPC controller action within the CL-DRTO, and is a crucial component of the CL-DRTO structure demonstrated in Figure 4.1. Note $\boldsymbol{u}_k$ and $\boldsymbol{u}_{k-1}$ only contain the control actions of the first step in the control horizon, this differs in structure from the formula for $\boldsymbol{u}_k$ in the MPC algorithm. This is accomplished by the addition of a selection matrix S, which only selects the first n_u rows: the rows corresponding to the first step of the control horizon. S contains an identity matrix I_s of shape n_u x n_u, and the *"0"* matrix is of shape $n_u \times [(M-1) * n_u]$. Now by obtaining the current states, control actions, and disturbance values, the DRTO can estimate the response of the MPC layer to a set-point of its choice, and the resulting $\mathbf{u}_k$ value can be constrained within the DRTO optimization problem. The proposed architecture will be validated by under conditions challenging to estimators and optimizers. Significant input saturation, a nonlinear plant, and linearization mismatch will be used to evaluate this methodology in terms of its robustness and capability of controlling economic and target tracking objectives.

4.3 Case Study

4.3.1 Nonlinear Plant

The case study used here is a multiple-input multiple-output CSTR. The parameters governing the system dynamics were obtained from Li et al. [8]. A schematic of the CSTR used here is given in Figure 4.4 below.

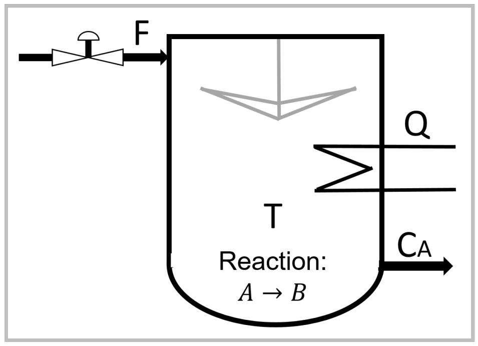

Figure 4.4: A diagram of the CSTR controlled by the proposed architecture

The inlet flowrate (F) and heat to the reactor (Q) are the manipulated variables and the reactant concentration in the CSTR (C_A) and reactor temperature (T) are the controlled variables. To test the estimator, a heater efficiency term (η) has been added to the forumulas from Li et al. [8]. The nonlinear equations governing the CSTR are given in Equation Sets 4.17 and 4.18.

$$\frac{dC_A}{dt} = \frac{F}{V_R}(C_{A_0} - C_A) - k_0 e^{-\frac{E}{RT}} C_A^2 \tag{4.17}$$

$$\frac{dT}{dt} = \frac{F}{V_R}(T_0 - T) - \frac{\Delta H k_0}{\rho_R C_p} e^{-\frac{E}{RT}} C_A^2 + \frac{\eta Q}{\rho_R C_p V_R} \tag{4.18}$$

An open-loop test of the equations above was performed to determine a viable steady-state condition to use for future linearization. These steady state values (set as the initial values of the simulation) along with the parameters from Li et al. [8] and a nominal efficiency value of 0.9 compose Table 4.1 below.

Table 4.1: Parameters used to simulate a CSTR for the case study

Symbol	Description	(Initial) Value	Units
C_A	Conc. of A in CSTR	0.339	kmol/m^3
T	Temperature of CSTR	545	K
F	Inlet Flowrate	5	m^3/h
Q	Heater Power	99,840	kJ/h
t	time	0	h
C_{A0}	Inlet Conc. of A	3.5	kmol/m^3
T_0	Inlet Temperature	300	K
k_0	Pre-exponential rate factor	8.46×10^6	m^3/kmol-h
E	Activation Energy	5×10^4	kJ/kmol
R	Ideal Gas Constant	8.314	kJ/kmol-K
ρ_R	Density of fluid in CSTR	1000	kg/m^3
C_p	Heat capacity of fluid in CSTR	0.231	kJ/kg-K
V_R	Reactor fluid volume	1.0	m^3
ΔH	Heat of reaction	-1.16×10^4	kJ/kmol
η	Heater efficiency	0.9	——

The efficiency value of 0.9 will be used to simulate the true response of the plant, but this value will be considered unknown to the control architecture, and it will be the estimator's task to converge on this true value given an initial guess of 0.85. Nonlinear models tend to describe the true behavior of the plant most accurately; however, control architectures typically utilize linear models for simplicity and computational considerations.

4.3.2 Linearized Plant

The plant model will linearized and expressed in deviation form by subtracting the current value from the nominal/steady state value given in Table 4.1. This linear model for the system is given below.

$$
\begin{aligned}
\boldsymbol{x} &= \begin{bmatrix} C'_A \\ T' \end{bmatrix} = \begin{bmatrix} C_A - C_{A,\text{ss}} \\ T - T_{\text{ss}} \end{bmatrix} \\
\boldsymbol{u} &= \begin{bmatrix} F' \\ Q' \end{bmatrix} = \begin{bmatrix} F - F_{\text{ss}} \\ Q - Q_{\text{ss}} \end{bmatrix} \\
\boldsymbol{\theta} &= [\eta'] = \eta - \eta_{\text{ss}} \\
\frac{d\boldsymbol{x}_p}{dt} &= \begin{bmatrix} a_p C'_A + b_p T' + e_p F' + f_p Q' + p_{1,p}\eta' \\ g_p C'_A + h_p T' + k_p F' + l_p Q' + p_{2,p}\eta' \end{bmatrix}
\end{aligned}
$$

The linearized model in state space form is given by

$$
\begin{bmatrix} \frac{dC'_A}{dt} \\ \frac{dT'}{dt} \end{bmatrix} = \begin{bmatrix} a_p & b_p \\ g_p & h_p \end{bmatrix} \begin{bmatrix} C'_A \\ T' \end{bmatrix} + \begin{bmatrix} e_p \\ f_p \end{bmatrix} \begin{bmatrix} F' \\ Q' \end{bmatrix} + \begin{bmatrix} p_{1,p} \\ p_{2,p} \end{bmatrix} [\eta']
$$

which may be compactly expressed as

$$
\begin{aligned}
\dot{\boldsymbol{x}}_p &= A_p \boldsymbol{x}_p + B_p \boldsymbol{u} + D_{\theta,p} \boldsymbol{\theta} \\
\boldsymbol{y}_p &= C_p \boldsymbol{x}_p
\end{aligned}
\tag{4.19}
$$

In Equation Set 4.19, a_p, b_p, e_p, f_p and $p_{1,p}$ represent the partial derivatives of $\frac{dC'_A}{dt}$ with respect to each variable. Similarly, g_p, h_p, k_p, l_p, and $p_{2,p}$ represent the partial derivatives of $\frac{dT'}{dt}$ with respect to each variable. This linearization is done at the nominal values given in Table 4.1 and using a time interval of $\Delta t_{MPC/plant} = 0.0\bar{3}3$ hrs, or 2 minutes. The resulting A_p, B_p, C_p and $D_{\theta,p}$ will be the state space model matrices used to simulate the actual response of the plant. Subsequent model linearizations will take similar matrix structure and will not be characterized elementwise for brevity. In this case study, the true value of the unmeasured parameter will be assumed constant, but this may not always be the case, and so the plant is structured to accommodate dynamic values as well.

4.3.3 MPC Control Layer

In this case study, the MPC uses the state space setup in Equation Set 4.8, where disturbance terms are added as pseudo-states and incorporated into the state space model matrices. This larger state vector ($\tilde{\boldsymbol{x}}_{MPC}$) is exclusively used for the MPC model, and takes the form below.

$$\begin{aligned}\tilde{\boldsymbol{x}}_{MPC,(k+1)} &= A_{\text{MPC},d}\tilde{\boldsymbol{x}}_{MPC,k} + B_{\text{MPC},d}\boldsymbol{u}_k \\ \tilde{\boldsymbol{y}}_{MPC,(k+1)} &= C_{\text{MPC},d}\tilde{\boldsymbol{x}}_{MPC,k}\end{aligned}$$

Where in this case, the formula for $\tilde{\boldsymbol{x}}_{MPC,(k+1)}$ is

$$\begin{bmatrix} C_{A,(k+1)} \\ T_{(k+1)} \\ d_{C_A,(k+1)} \\ d_{T,(k+1)} \end{bmatrix} = A_{\text{MPC},d} \begin{bmatrix} C_{A,k} \\ T_k \\ d_{C_A,k} \\ d_{T,k} \end{bmatrix} + B_{\text{MPC},d} \begin{bmatrix} F_k \\ Q_k \end{bmatrix} \tag{4.20}$$

These matrices are shaped using the parameters given in Table 4.2. These parameters govern the contruction of the matrices used in thet MPC algorithm. Table 4.2 can be applied to the generic MPC algorithm in the Appendix to produce the matrices used in this case study.

Table 4.2: Parameters governing the shape of the MPC matrices

Symbol	Description	Value
n_u	Number of Inputs	2
n_x	Number of Outputs	2
n_d	Number of Disturbance Terms	2
n_y	Number of Outputs with Set-Points	2
P	Prediction Horizon Length	3
M	Control Horizon Length	2

To see the exact contents of the matrices used in this case study, see the Case Study Specific MPC Algorithm in the Appendix. Once these matrices are initialized, the MPC controls the system using the algorithm in Section 4.1.4. At the end of the algorithm, the MPC will clip any inputs that exceed the plant input bounds. This clipping may result in suboptimal results, as the MPC equations used in this layer are derived from the analytical solution of an unconstrained MPC. These bounds for the target tracking and economic objective functions (denoted by "Tgt" and "Econ", respectively) are given in Table 4.3.

Table 4.3: Bounds set for input clipping

Value	Lower Bound	Upper Bound (Tgt)	Upper Bound (Econ)	Units
F	0	13	2.8	m^3/hr
Q	0	4×10^5	4×10^5	kJ/hr

Here the MPC layer provides an analytical solution for the entire prediction horizon explicitly using the composite matrices proposed by Li and Swartz [2], avoiding the need for optimizer generated solutions at the higher MPC frequency, significantly reducing computational expense.

4.3.4 DRTO Control Layer

The DRTO layer will need to predict the behavior of the plant as it chooses optimal set-points to send to the MPC layer. The model used in the DRTO to predict the plant is shown in Equation Set 4.21 and follows the generic structure of Equation Set 4.6

$$\boldsymbol{x}_{m,(k+1)} = A_{m,d}\boldsymbol{x}_{m,k} + B_{m,d}\boldsymbol{u}_k + D_{\theta,m,d}\boldsymbol{\theta}_k$$

where in this case, the equation for $\boldsymbol{x}_{m,(k+1)}$ is

$$\begin{bmatrix} C'_{A,(k+1)} \\ T'_{(k+1)} \end{bmatrix} = A_{m,d} \begin{bmatrix} C'_{A,k} \\ T'_k \end{bmatrix} + B_{m,d} \begin{bmatrix} F'_k \\ Q'_k \end{bmatrix} + D_{\theta,m,d} \begin{bmatrix} \eta'_k \end{bmatrix} \tag{4.21}$$

For this Case Study, the DRTO model will be derived by linearizing the same equations as the linearized plant model. The DRTO model will take the same form as the plant model with two key differences. First, the DRTO operates at a lower frequency than the plant, and thus the time interval to discretize the DRTO model will be larger than that of the plant, with $\Delta t_{\text{opt}} = 0.1\bar{6}6$ hrs, or 10 minutes. The second difference will be the effect of the unknown parameter vector $\boldsymbol{\theta}_k$. Without access to the true value of the parameters, the DRTO will have to simulate the plant using estimated values. This will cause plant-model mismatch while the estimator converges and any time the estimation deviates from the true value during the simulation. In addition to predicting the plant dynamics with a model, this architecture also includes an MPC model within the DRTO. The DRTO uses the same model as Equation Set 4.20 to increment the simulated MPC response, but to calculate the control action chosen by the MPC the DRTO layer will use Equation 4.16. Here a selection matrix S selects the first two rows of Equation 4.16, corresponding to the F and Q values of the first step in the control horizon. Solving explicitly in this way bypasses the need for optimization to simulate the MPC, and instead the DRTO optimization problem becomes quite simple. The optimization setup of Equation Set 4.10 is applied here, and the DRTO is tested under an economic and a target tracking objective function. A copy of the optimization problem used in this case study is included as Equation Set 4.22 for reference.

$$
\begin{aligned}
&\min_{Y_{\text{sp}}} \phi(Y_{\text{sp}}) \\
\text{s.t. } & X_{\text{m,min}} \leq X_{m,(k+1|k)} \leq X_{\text{m,max}} \\
& Y_{\text{sp,min}} \leq Y_{\text{sp}} \leq Y_{\text{sp,max}} \\
& U_{\text{min}} \leq U \leq U_{\text{max}} \\
& U = \begin{bmatrix} \boldsymbol{u}_k^T & \boldsymbol{u}_{(k+1)}^T & \dots & \boldsymbol{u}_{(k+M-1)}^T \end{bmatrix}^T \\
& Y_{\text{sp}} = \begin{bmatrix} \boldsymbol{y}_{\text{sp},k}^T & \boldsymbol{y}_{\text{sp},(k+1)}^T & \dots & \boldsymbol{y}_{\text{sp},(k+M-1)}^T \end{bmatrix}^T \\
& X_{m,(k+1|k)} = \begin{bmatrix} \boldsymbol{x}_{m,(k+1|k)}^T & \boldsymbol{x}_{m,(k+2|k)}^T & \dots & \boldsymbol{x}_{m,(k+P|k)}^T \end{bmatrix}^T
\end{aligned}
\tag{4.22}
$$

The DRTO layer is comprised of Equation Set 4.22 and and subject to the model in Equation Set 4.21. The optimization problem above can either track an economic or a target tracking objective. Constraints for the case study are provided in the next section. The target tracking objective is provided as Equation 4.23.

$$
\varphi_{Tgt} = \alpha_{1,Tgt}(C_A - C_{A_{Tgt}})^2 + \alpha_{2,Tgt}(T - T_{Tgt})^2 \tag{4.23}
$$

where minimizing the objective value φ_{Tgt} will drive the system towards the predetermined targets for concentration and temperature ($C_{A_{Tgt}}$ and T_{Tgt}, respectively). This objective function is used iteratively by the DRTO to choose a set of optimal set-points to send to the MPC layer. The function is also integrated across the entire simulation to determine the performance of the control setup as a whole. In most cases, the ultimate purpose of optimization is to achieve efficient and profitable performance from the process. This control structure will test its economic performance using an objective function proposed by Li et al. [8] to approximate profitability.

$$\varphi_{Econ} = -\alpha_{1,Econ}F(C_{A0} - C_A) + \alpha_{2,Econ}Q^2 \tag{4.24}$$

In Equation 4.24, the amount of reactant A converted to product B is given a weighting parameter $\alpha_{1,Econ}$ to indicate the revenue of the process. A second weighting parameter $\alpha_{2,Econ}$ was applied to the heat input to approximate the cost of the system. By minimizing φ_{Tgt} or φ_{Econ}, the DRTO can obtain different objectives. The weighting parameters α_1 and α_2 were adjusted for both functions to achieve balanced emphasis between both components.

Table 4.4: Objective function weights used in this Case Study

Weights	**Values**
$\alpha_{1,\text{Tgt}}, \alpha_{2,\text{Tgt}}$	$10^3, 1$
$\alpha_{1,\text{Econ}}, \alpha_{2,\text{Econ}}$	$10^5, 10^{-7}$

The DRTO is subject to several constraints, including constraints on the predicted response from the MPC layer. These constraints are given in Table 4.5.

Table 4.5: Constraints applied to the DRTO

State	Upper Bound	Lower Bound	Units
C_A	3.5	0.1	kmol/m^3
T	700	400	K
F_{Tgt}	13	0	m^3/h
F_{Econ}	2.8	0	m^3/h
Q	400000	0	kJ/h
$C_{A,sp}$	3.5	0	kmol/m^3
T_{sp}	700	400	K
$\Delta C_{A,sp}$	0.1	-0.1	kmol/m^3
ΔT_{sp}	30	-20	K

Note in Table 4.5 two different constraints for the inlet flowrate F depending on the objective function. While input saturation could be induced using an infeasible target in target tracking, the same could not be done for an economic objective. To test the performance of the control structure under input saturation, the constraints were instead severely tightened. This will cause the inlet flowrate to saturate for the entire simulation, thoroughly testing how the estimator handles this condition. By predicting MPC dynamics, the DRTO can estimate the F and Q chosen by the MPC as a result of the set-point. This is how the DRTO can constrain these manipulated variables.

Special care must be taken when implementing the economic objective function during input saturation. While the inputs are saturated, the poor control caused by plant-model mismatch can cause constraint violation as the DRTO incorrectly predicts the effects of its set-points. During this time, C_A could be driven below its lower bound of 0.1 kmol/m^3, making it seem as if the system is performing better than it truly is. In reality, violating constraints is associated with a cost, such as separation problems downstream. To approximate this reality, the controller will be penalized for violating this lower bound. The penalty was set to be 1% of revenue for every .01 kmol/m^3 below the lower bound, for a maximum penalty of 10% revenue. In practice these bounds may be required to meet regulatory or product specification targets. The lower bound on the concentration is applied as a constraint within the DRTO, however the penalty for violating this constraint is only applied when evaluating the objective functions over the entire time horizon and not within the optimization problem, as it only takes effect when the DRTO is unable to accurately predict the states of the plant.

The performance of the control architecture is affected by the value of the efficiency parameter used in the model, which is not measurable in this case. The Kalman Filter will make this possible.

4.3.5 Kalman filter Estimation Layer

The Kalman filter will attempt to estimate the true efficiency value of 0.9 given an initial guess of 0.85. Several perturbations of this initial guess were tested and had negligible impact on results. The states space model for the Kalman filter is given in Equation Set 4.25.

$$\begin{aligned}\tilde{\boldsymbol{x}}_{e,(k+1)} &= A_{e,d}\tilde{\boldsymbol{x}}_{e,k} + B_{e,d}\boldsymbol{u}_k \\ \tilde{\boldsymbol{y}}_{e,(k+1)} &= C_{e,d}\tilde{\boldsymbol{x}}_{e,(k+1)}\end{aligned}$$

Where in this case, the formula for $\tilde{\boldsymbol{x}}_{e,(k+1)}$ is

$$\begin{bmatrix} C'_{A,(k+1)} \\ T'_{(k+1)} \\ \eta'_{(k+1)} \end{bmatrix} = A_{e,d}\begin{bmatrix} C'_{A,k} \\ T'_k \\ \eta'_k \end{bmatrix} + B_{e,d}\begin{bmatrix} F'_k \\ Q'_k \end{bmatrix} \tag{4.25}$$

Equation Set 4.25 treats the estimated efficiency parameter η as a pseudo state, and updates its value using the Kalman filter algorithm given Equation Set 4.14. The Kalman filter will provide the estimated values of these states and pseudo-states to the DRTO layer, and use its own states in the next iteration. The extended Kalman filter can be used to better approximate nonlinearity, but for this case study Equation Set 4.25 will be used to estimate both linear and nonlinear setups.

The aforementioned setup will be tested in several environments that are classically challenging for estimators. For economic and target tracking objectives, the proposed solution will be tested under input saturation in three scenarios: a base case, linearization mismatch, and nonlinear mismatch.

4.3.6 Scenerio 1: Base Case

In the base case scenario, the proposed solution will be implemented on a linear plant without any additional changes made. The system will be tested during input saturation in both the economic and target tracking objectives. In this setup, there are no additional sources of disturbance. This will serve as a baseline for determining the effectiveness of the control structure in the more challenging scenarios to follow.

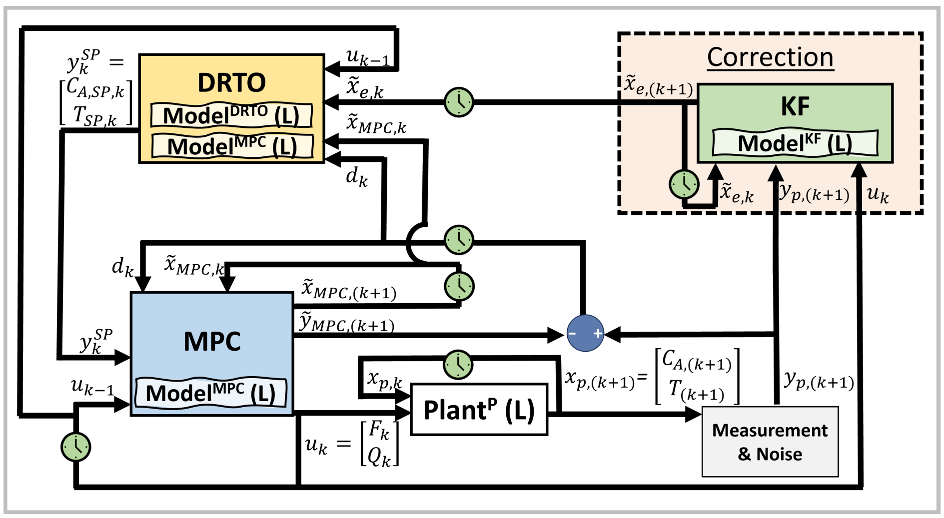

Figure 4.5: Information flow diagram of the base case for the case study

Figure 4.5 shows the control architecture along with the states, inputs, and set-points involved. However, the diagram does not show the pseudo-states found in the MPC and estimator matrices. The state vector used in the Kalman filter Estimation ($\tilde{x}_e$) will also include one pseudo-state of the estimated efficiency value (η).

The rank of the observability matrix O_e for the estimator was 3, which matches the dimensionality of the $\tilde{y}_e$ vector, indicating that the efficiency can be estimated by measuring the concentration and temperature.

This scenario tests the control structure when no plant-model mismatch is present, however this mismatch is a significant weak point of estimators, so it is critical to test the structure's robustness to this type of obstacle.

4.3.7 Scenario 2: Linearization Mismatch

In the base case, all the state space model matrices for the plant, MPC, Estimator, and DRTO models were linearized at the same steady state conditions. This creates a level of model agreement that may be overly ideal for practical use. In practice, updates may be performed on some layers and not others, such as only updating models in lower-risk control layers. Disagreement between the models used in control layers may occur, and testing how the proposed solution handles this situation will be important.

In this scenario, the MPC will be linearized at a different steady state condition than the plant, estimator, and DRTO models. Each layer will need to adapt to accommodate the linearization mismatch resulting from linearizations at different nominal conditions. The MPC will need to use its disturbance term to correct for the mismatch between the plant and the MPC model. The DRTO also simulates the MPC response, using an internal disturbance term to measure the difference between the DRTO's model of the plant and the DRTO's model of the MPC layer. This internal disturbance term will also need to correct for the mismatch created in this scenario. The estimator must also overcome the mismatch as it converges on the true value of the efficiency, and this scenario will test if plant-model mismatch makes this more difficult.

Another open-loop simulation of the nonlinear system in Equation Set 4.1 was performed to select an additional steady-state condition to linearize around. The steady state values are given in Table 4.6.

Table 4.6: Steady State values used to linearize the MPC model in Scenario 1

Variable	Value	Units
C_A	0.373	kmol/m^3
T	516	K
F	3.27	m^3/hr
Q	44,697	kJ/hr

The MPC state space model matrices are linearized around this new steady state condition and discretized following the same procedure outlined in Section 4.1.4. This creates three new state space model matrices for the MPC layer: $A_{MPC2,d}$, $B_{MPC2,d}$, and $C_{MPC2,d}$, with the subscript "MPC2" indicating identical structure but a new set of steady state values. These changes required an experimental retuning of the nominal values used in the Kalman filter to better estimate the new system. As this is a test of the proposed solution, no additional changes will be made to how control is achieved, and Figure 4.5 serves as an accurate reference for how this scenario will be set up.

4.3.8 Scenario 3: Nonlinear Mismatch

Ideally when controlling a nonlinear plant, a nonlinear model would be used in the DRTO, MPC, and Estimator layers. This often does not occur for several reasons; nonlinear models are more computationally expensive and complex than linear models, and if the control layers can handle the mismatch, this provides a feasible approach to controlling a nonlinear process.

Alternatively, the nonlinearity of the plant may not be fully understood. In more complex processes, an accurate nonlinear model may not be available to fully describe the plant: in this case linear models must be used despite the inaccuracy.

This scenario poses a different type of mismatch than Scenario 2. By using a nonlinear plant there will be plant-model mismatch in every layer, and in this scenario even the estimator will have inaccuracies in its predictions.

Plant-model mismatch within the DRTO and estimator layers is a particular vulnerability for estimator based architectures [9]. One issue with plant-model mismatch in estimator based architectures is the ability to ensure plant optimality. Even when the estimates have converged to their stationary values, the DRTO solution using the model updated with these values may still be unable to achieve plant optimality if plant-model mismatch exists [10]. As these layers are unable to accurately predict the response of the plant, infeasible set-points may be sent to the MPC layer and input saturation and constraint violation may occur. This in turn creates dynamics the DRTO and estimator cannot predict, and the suboptimal performance of the optimizer can entirely derail control over the system.

However, this system constrains MPC control action within the DRTO, suggesting that even while the disturbance terms and estimator parameters converge, the DRTO will be constrained against providing infeasible set-points to the MPC, improving the performance of the optimizer. This scenario will also demonstrate a user-friendly approach to applying complex control architecture in practice: using linear models on a nonlinear plant and handling the difference with the built-in disturbance vectors.

This scenario is a modification of the base case, and the only change made for this scenario will be changing the plant response from the linear model used in Equation Set 4.19 to nonlinear equations 4.17 and 4.18. The MPC and DRTO layers will continue using linear models linearized around the same conditions, as in the base case. States will be predicted using the integrator described in Section 3.3 and use the same time interval of $\Delta t_{\text{mpc/plant}} = 0.0\bar{3}3$, or 2 minutes. Here again Figure 4.5 provides an accurate reference of how this scenario will be controlled.

4.4 Results and Discussion

4.4.1 MPC Dynamics in the DRTO Layer

Input saturation is problematic for optimizers as the boolean logic results in a more difficult optimization problem, and the resulting mismatch from input saturation makes estimation difficult as well. The proposed solution suggests dealing with this issue by predicting MPC behavior in the DRTO layer. The DRTO can then prevent input saturation by adding the input bound constraints to the DRTO layer's optimization problem.

Before proceeding with testing the performance of the proposed solution in general, this input constrained CL-DRTO was verified against CL-DRTO without input constraints from the MPC layer. Typically, dealing with input saturation requires an accounting of this behavior in the optimizer, such as complementarity constraints, but proving integrated MPC predictions against alternative strategies falls outside the scope of this chapter. This work seeks only to demonstrate that a CL-DRTO can avoid the obstacles presented by input saturation.

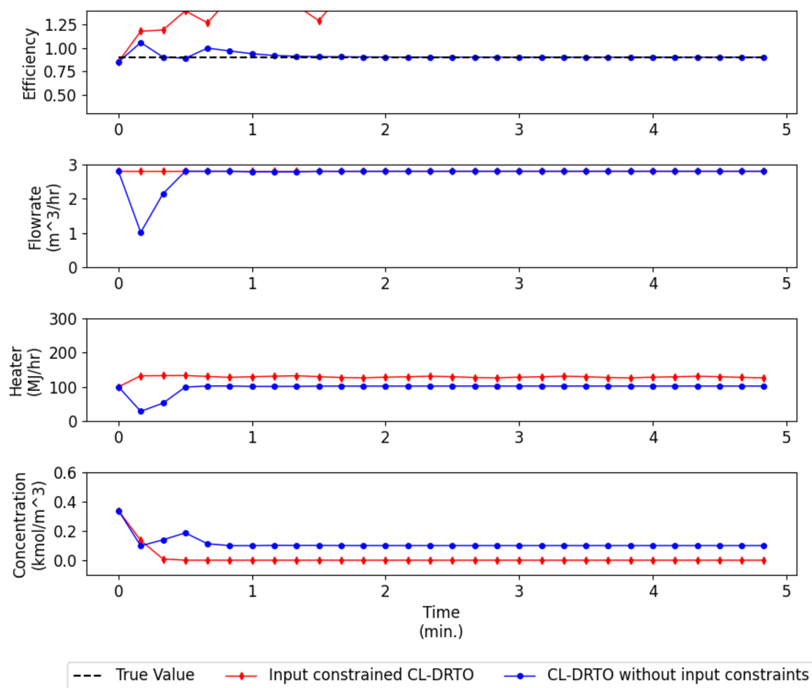

Figure 4.6: Efficiency estimation and input trajectories using input constrained CL-DRTO under an economic objective

Figure 4.6 shows the variables (F, Q, C_A) used in the objective function from Equation 4.24 and the performance of the estimator on its estimated parameter (η). The flowrate F remained saturated for nearly the entire simulation. Despite constant input saturation, the CL-DRTO system used here converged on the true efficiency value of 0.9, and obeyed the C_A lower bound constraint of 0.1 kmol/m^3. Without input bounds in the DRTO, there was mismatch between the DRTO model and the plant response. As a result, the constraint was violated, and the estimator was unable to converge on its estimated values. The profitability for both strategies were calculated using Equation 4.24 and are reported in Table 4.7.

Table 4.7: Economic results of using input constrained CL-DRTO

Strategy	Result
Input constrained CL-DRTO	\$26,625
CL-DRTO without input constraints	\$22,963

The true objective values for Table 4.7 were negative and minimized by the DRTO, but as they represent profitability they are reported as positive dollar values corresponding to profit maximization. In dealing with input saturation, adding input bounds as constraints to the DRTO layer improved profitability by 15.9%. Similar results were observed using the target tracking objective, reported in Table 4.8.

Table 4.8: Target tracking results of using input constrained CL-DRTO

Strategy	Result
Input Constrained CL-DRTO	11,959
DRTO without input constraints	12,887

Using input constrained CL-DRTO improved target tracking performance by 7.2% compared to CL-DRTO without input constraints and was able to converge on the estimated parameter despite input saturation. These results confirm that integrating the MPC behavior into the DRTO layer adequately mitigated the effects of input saturation in this case.

4.4.2 Economic Performance

Improving economic performance is a major goal of advanced control objectives, so the three scenarios were tested in their ability to meet an economic objective. The base case will demonstrate the proposed solution's ideal function on a linear plant and minimal disturbance. The robustness will then be tested by replacing the linear plant with a nonlinear one and keeping the linear models. The proposed solution will also be tested by linearizing the MPC around a different steady-state than the rest of the system.

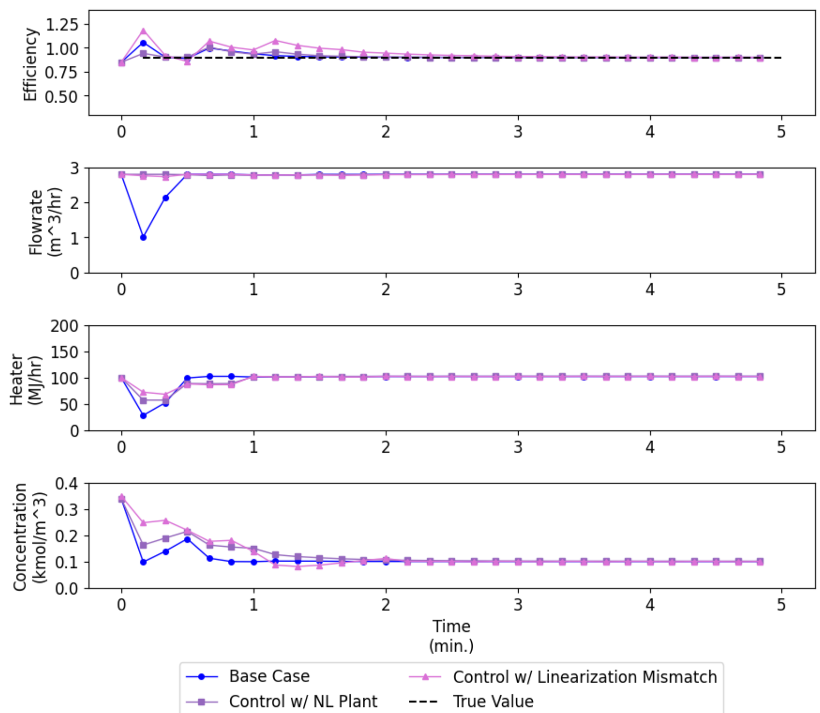

Figure 4.7: Efficiency estimation and input trajectories of the proposed solution under various types of disturbance under an economic objective

The performance displayed in Figure 4.7 indicates several interesting findings. First, all three strategies eventually converge to the correct economic optimum given sufficient time. This time is relatively fast ($< 3\,\text{min.}$) for the given case study and suggests that even given significant levels of disturbance this proposed solution should be able to control systems with frequent set-point changes. Second, the use of the disturbance term appears to slow down the convergence of the estimator. This effect may be due to an interaction between two different layers attempting to converge on true values simultaneously. As the Kalman filter gain K_f converges on a steady state value, the disturbance terms within the MPC and DRTO layers are also converging to their appropriate values, introducing interactions that may affect convergence time. Third, constraint violation still occurred briefly during the base case. The C_A value decreased below the lower bound briefly at $t \approx 0.2$ min. This aligns with the time the parameter estimate of the base case was furthest from the true value, and is an expected side effect of estimators. During convergence, constraint violation is briefly possible, however as shown in these results, this effect is minimal, and eventually the estimator converges and this no longer becomes a risk.

Table 4.9: Economic results of the proposed solution under various types of disturbance

Scenario	Result
Base Case	\$26,625
Control w/ Nonlinear Plant	\$25,947
Control w/ Linearization Mismatch	\$26,101

The proposed solution was able to handle both types of disturbance very well (the disturbance induced by linearization mismatch and the disturbance induced by controlling a nonlinear plant with linear models), and achieved the same steady-state optimum as if these disturbances were not present. Therefore differences in performance will be most noticeable during the transition periods of a plant, and while these transitions here were brief (< 3 min.), in systems with frequent changes these differences in dynamics can become incredibly important. Both disturbance scenarios performed similarly, the difference reported in Table 4.9 being less than 1%, and they each only affected the objective function by about 2.5%. This suggests that the proposed solution can be implemented in scenarios with significant disturbance and can be expected to perform nearly as well as the base case with minimal disturbance.

While these results are significant, the time to achieve steady state is relatively short, and testing the system under multiple changes in steady state would be valuable.

4.4.3 Target Tracking Performance

Using a target tracking objective, the proposed solution is tested over a longer duration and for multiple changes in steady state conditions.

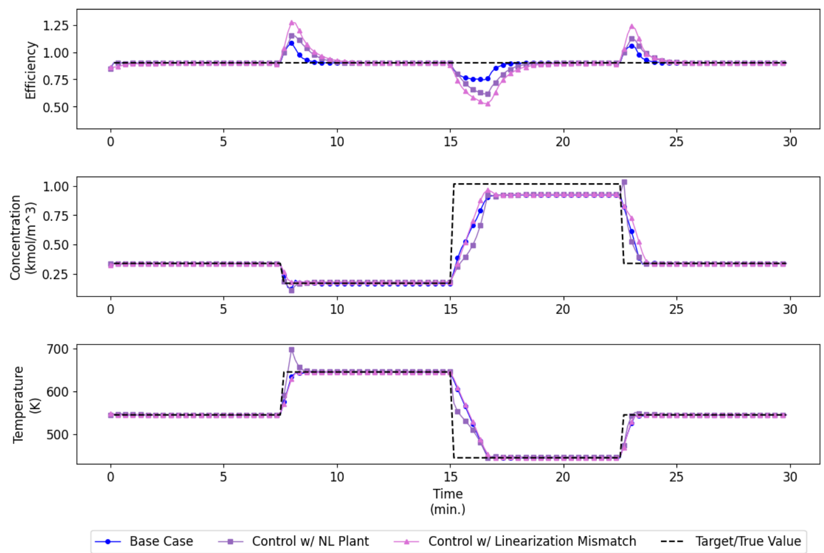

Figure 4.8: Efficiency estimation and target trajectories of the proposed solution under various types of disturbance under an target tracking objective

In Figure 4.8, the temperature target is stepped above the nominal value, below the nominal value, and then are returned to nominal value. The concentration targets are stepped in the opposite direction, and the second step change of the concentration target is intentionally infeasible to induce input saturation. This simulation is also run over a longer period of time than the economic simulation as there are several transient periods to observe in the target tracking simulation.

The behavior shown in Figure 4.8 shows that set-point changes perturb the estimator temporarily, though the estimator recovers from these perturbations quickly. As with the economic objective, all three scenarios can be expected to converge to the same value given sufficient time, however there are variations in how these scenarios reach these values. The estimator takes longer to converge as the disturbance term becomes more active. This results in a slightly longer time to reach steady state values and nonideal behavior during transitions between steady states.

Table 4.10: Target tracking of the proposed solution under various types of disturbance

Scenario	Result
Base case	11,959
Control w/ Nonlinear Plant	13,037
Control w/ Linearization Mismatch	12,412

With a longer simulation length, the differences between the two scenarios became more apparent. The performance of the linear achitecture dropped when using a nonlinear plant by 9.0% but only by 3.8% when handling linearization mismatch. This can be observed in Figure 4.8 where the nonlinear plant deviated significantly from the ideal path, especially immediately after a target change. Interestingly, the estimator converged better under the nonlinear scenario than the linearization mismatch scenerio, but the effect of poor control may be more severe when the plant does not respond as expected.

Despite these findings, the performance observed may be within acceptable deviations given the advantage of using linear models on a nonlinear process, and the results of this experiment confirm the proposed solution can adequately handle even significant sources of disturbance.

4.4.4 Kalman filter Performance

Across economic and target tracking objectives, the Kalman filter was successfully able to converge on the true efficiency value despite various types of model mismatch and significant input saturation. Overcoming these weaknesses without the need for complex complementarity constraints or overly conservative control also made the Kalman filter easier to implement in this study, and explicitly calculating the MPC response in the DRTO layer avoided the nested optimization that makes control architecture so taxing to use. The rapid convergence time of the Kalman filter also suggests that this implementation would also perform well had the system had highly dynamics behavior, and the adequate control during convergence suggests the proposed solution may even be useful during start-up operations that occur entirely out of steady-state conditions. Kalman filter estimation has demonstrated its ability to handle multiple sources of disturbance as well as input saturation while obtaining states and parameters that would have been unavailable to alternative methods such as bias updating.

4.4.5 Controlling Nonlinearity

Although this chapter doesn't explore a rigorous implementation of nonlinearity (such as nonlinear models and an extended Kalman filter), this work showed that good overall control of a nonlinear plant can be achieved by proper parameterization of the linear control architecture. Controlling a nonlinear plant with linear control architecture offers significant advantages, particularly for scalability. The case study chosen in this chapter was simple compared to industrial applications. Scaling up the proposed solution would be problematic if the models were nonlinear and became increasingly complex. To control a large-scale process without burdening the computer, linearization of the nonlinear behavior becomes crucial, provided the control architecture can handle the resulting plant-model mismatch.

This chapter shows that the DRTO, MPC and Kalman filter layers can all use linear models on a nonlinear process without significant performance issues. Attempting to control a nonlinear plant with these linear models demonstrated excellent performance despite the plant-model mismatch. While plant-model mismatch tends to derail Kalman filter estimators, these results indicate the estimator can handle the mismatch relatively easily, recovering in only a few minutes, with only minor deviations from ideality during the convergence.

4.5 Vulnerabilities

While the proposed control solution offers many advantages, there are also significant vulnerabilities to the approach. The structure relies heavily on knowledge of plant dynamics. While the structure was able to handle a nonlinear plant using linear models, these two models still have relatively similar dynamics and gains near steady state conditions. If the chosen model poorly represented actual plant behavior, the disturbance terms may not be sufficient in compensating for the plant-model mismatch. This case study's missing variables are also all observable by the plant measurements. This may not always be true, and for plants with unobservable hidden states, the estimator may not be sufficient to obtain critical values. Results also indicated that target changes perturb the estimator and cause nonideal dynamics while the estimator recovers. In plant that takes frequent or large steps away from steady state conditions, these issues may become more serious. In a system with rapid dynamics the Kalman filter may not have a chance to converge, causing nonideality throughout plant operation. The Kalman filter also needed to be retuned when a new steady state value was chosen for linearization. If this is a frequent occurrence (such as in refineries with multiple operating modes) this may increase the complexity of use.

4.6 Chapter Summary

This chapter explores the performance of the Kalman filter estimator using CL-DRTO and MPC control layers. In this study, both the MPC and Kalman filter use pseudostates in augmented state vectors. The proposed solution is studied under input saturation for economic and target tracking objectives, and is placed under several sources of disturbance. By linearizing the MPC at a different state than the remaining control layers and plant, mismatch is induced between the MPC model and the remainder of the system. In another scenario, the base case is altered by replacing the linear plant with a nonlinear plant, inducing mismatch between the linear models and the nonlinear process. Despite these sources of disturbance, the Kalman filter overcame all perturbations from the steady state value of the the unknown parameter. However, if the MPC input bounds were not added to the DRTO optimization problem, the Kalman filter could not converge on the steady-state parameter values, and the system demonstrated poor control. The MPC algorithm involves several composite matrices, and a generic guide for sizing these matrices for a given system is given in this chapter as well.

4.7 Appendix

4.7.1 Generic MPC Algorithm

The matrices employed in the model predictive control (MPC) algorithm are designed to incorporate the effects of disturbances on the system states throughout the entire prediction horizon. This is achieved by utilizing composite matrices, as detailed in Section 4.3.3, to compute the predictions for the full extent of the horizon. Here underbar notation (e.g. $\underline{X}$) denotes a composite matrix composed of component matrices that describe how these component matrices interact across the prediction/control horizon. Time will still be described with the notation k, but the horizon time step will be given a notation i. To structure the matrices used here, the control action and disturbance terms $\boldsymbol{u}_0$ and $\boldsymbol{d}$ are initialized with values from the previous iteration of the MPC. The MPC will weight the target tracking and input move suppression using matrices Q and R respectively. Q is of shape $n_y \times n_y$ where n_y represents the number of states with targets being tracked. R is of shape $n_u \times n_u$. The structure of these matrices is given below.

$$Q = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & q_{n_y} \end{bmatrix} \quad R = \begin{bmatrix} r_1 & 0 & \cdots & 0 \\ 0 & r_2 & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & r_{n_u} \end{bmatrix} \tag{4.26}$$

where q_1 represents the weight corresponding to the first output and r_1 represents the movement penalty on the first input. These weight matrices Q and R are then used to construct composite matrices $\underline{Q}$, $\underline{R}$, and $\underline{I}_L$, as shown in Equation Set 4.27. The shapes of $\underline{Q}$, $\underline{R}$ and $\underline{I}_L$ are $(P * n_y) \times (P * n_y)$, $(M * n_u) \times (M * n_u)$, and $(M * n_u) \times (M * n_u)$, respectively. As mentioned earlier, P and M represent the prediction and control horizon lengths, respectively.

$$\underline{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 \\ 0 & Q & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & Q \end{bmatrix} \quad \underline{R} = \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & R \end{bmatrix} \quad \underline{I}_L = \begin{bmatrix} I & 0 & \cdots & 0 \\ 0 & I & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & I \end{bmatrix} \tag{4.27}$$

The $\underline{A}$ composite matrix is then calculated from the model matrices. In this context, A, B, and C are the $A_{MPC,d}$, $B_{MPC,d}$, and $C_{MPC,d}$ matrices. These state space model matrices become the new composite matrix $\underline{A}$ using Equation 4.28.

$$\underline{A} = \begin{bmatrix} CB & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{M-1} CA^iB & \cdots & CB \\ \sum_{i=0}^{M} CA^iB & \cdots & CAB + CB \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{P-1} CA^iB & \cdots & \sum_{i=0}^{P-M} CA^iB \end{bmatrix} \tag{4.28}$$

There is a row of matrices for each step in the prediction horizon, and a column for each step in the control horizon giving $\underline{A}$ a shape of $(P * n_y) \times (M * n_u)$. Finally, the matrices above can be used to calculate the K_1 matrix used for the MPC algorithm using Equation 4.29.

$$K_1 = (A^T Q A + R)^{-1}(A^T Q) \tag{4.29}$$

The K_1 matrix will have a final shape of $(M * n_u) \times (P * n_x)$. The MPC layer uses a composite vector $\underline{b}_k$ to update the control action, and two matrices used to construct $\underline{b}_k$ can also be constructed outside the control loop. These matrices are given in Equation Set 4.30.

$$\underline{B}_1 = \begin{bmatrix} CA \\ \vdots \\ CA^m \\ CA^{m+1} \\ \vdots \\ CA^p \end{bmatrix} \underline{B}_2 = \begin{bmatrix} CB \\ \vdots \\ \sum_{i=0}^{m-1} CA^i B \\ \sum_{i=0}^{m} CA^i B \\ \vdots \\ \sum_{i=0}^{p-1} CA^i B \end{bmatrix} \tag{4.30}$$

The following components must be updated within the control loop, but they can reference the constant values of Equations 4.27-4.30 to reduce computational expense. When the MPC runs, it first constructs a composite matrix ($\underline{b}_k$) used for calculating controller action.

$$\underline{b}_k = \underline{B}_1 X_k + \underline{B}_2 \boldsymbol{u}_{k-1} \tag{4.31}$$

Equation 4.31 computes the value of $\underline{b}_k$ based on the value of the composite state vector (X_k) and the previous control action ($\boldsymbol{u}_{k-1}$). In the application of Li and Swartz [2], the disturbance terms are applied additively to $\underline{b}_k$. However, as the dynamics are built into the state space model matrices within $\underline{B}_1$ and $\underline{B}_2$, this is not necessary.

At this stage of the process, the MPC algorithm will receive a set-point from the DRTO layer. This set-point is fixed across the MPC's prediction horizon as it operates at a higher frequency than the DRTO. The new composite vector for set-points is given in Equation 4.32.

$$\boldsymbol{y}_k^{SP} = \begin{bmatrix} \boldsymbol{y}_{i,k}^{SP} \\ \boldsymbol{y}_{i+1,k}^{SP} \\ \vdots \\ \boldsymbol{y}_{P,k}^{SP} \end{bmatrix}$$
$$\boldsymbol{y}_{i,k}^{SP} = \boldsymbol{y}_{i+1,k}^{SP} = \cdots = \boldsymbol{y}_{P,k}^{SP} \tag{4.32}$$

Here $\boldsymbol{y}_k^{SP}$ has a shape of $(M * n_y) \times 1$. The change in controller values ($\Delta\underline{\boldsymbol{u}}_k$) can now be calculated and used to predict the next set of states ($\underline{\boldsymbol{y}}_{(k+1)}$) and suggested control actions ($\underline{\boldsymbol{u}}_k$) as shown in Equation Set 4.33.

$$\begin{aligned} \Delta\underline{\boldsymbol{u}}_k &= K_1(\underline{\boldsymbol{y}}_k^{SP} - \underline{b}_k) \\ \underline{\boldsymbol{y}}_{(k+1)} &= \underline{A}\Delta\underline{\boldsymbol{u}}_k + \underline{b}_k \\ \underline{\boldsymbol{u}}_k &= \underline{\boldsymbol{u}}_{k-1} + I_L\Delta\underline{\boldsymbol{u}}_k \end{aligned} \tag{4.33}$$

This derivation and further explanation of its components can be found in Li and Swartz [2]. The first n_y rows of $\boldsymbol{y}_{(k+1)}$ are the predicted plant response from that control action and are labeled as $\boldsymbol{y}_{MPC,(k+1)}$. The first n_u rows of $\underline{\boldsymbol{u}}_k$ are the control actions of the first time step, and these are clipped as needed and sent to the plant for implementation, labeled as $\boldsymbol{u}_k$. The actual response of the plant to $\boldsymbol{u}_k$ is measured, and the difference between the measured plant response ($\boldsymbol{y}_{p,(k+1)}$) and the predicted plant response ($\boldsymbol{y}_{MPC,(k+1)}$) is used to calculate a disturbance term as in Equation 4.34.

$$\boldsymbol{d}_k = \boldsymbol{y}_{p,(k+1)} - \boldsymbol{y}_{MPC,(k+1)} \tag{4.34}$$

Using the equations of the appendix given so far, the MPC algorithm can be executed as shown in Li and Swartz [2] and given below: *Init.* Calculate constant components

($\underline{Q}$, $\underline{R}$, $\underline{I}_L$, $\underline{A}$, $\underline{K}_1$) once outside of the control loop.

1. Calculate $\underline{b}_k$, a function of the states and inputs.
2. Form a composite matrix of set-points $\underline{\boldsymbol{y}}_k^{SP}$ from the DRTO set-points received.
3. Calculate the change in controller values, $\Delta\underline{\boldsymbol{u}}_k$.
4. Use these controller actions to generate a predicted plant response, $\underline{\boldsymbol{y}}_{(k+1)}$.
5. Select the first n_y rows of $\underline{\boldsymbol{y}}_{(k+1)}$ to select the first response in the prediction horizon, $\boldsymbol{y}_{\text{mpc},(k+1)}$.
6. Select the first n_u rows of $\Delta\underline{\boldsymbol{u}}_k$ and use it to determine the next set of control actions $\boldsymbol{u}_k$.
7. Send $\boldsymbol{u}_k$ to the plant and compare the measured response $\boldsymbol{y}_{p,(k+1)}$ with the predicted response $\boldsymbol{y}_{\text{mpc},(k+1)}$.
8. Update $\boldsymbol{X}_k$ with the disturbance $\boldsymbol{d}_k$ between the measured and predicted responses.

The disturbance term is then used as an input in the next iteration of the MPC along with the composite state vector (X_k) and the previous control action ($\boldsymbol{u}_{k-1}$). At this point the loop is restarted.

4.7.2 Case Study Specific MPC Algorithm

Several initial values are needed to govern the behavior of the MPC layer. The values used in this case study are given in Table 4.11 for reference.

Table 4.11: Weights and penalties used in constructing MPC matrices

Symbol	Description	Value
q_1	Target tracking weight for C_A	100
q_2	Target tracking weight for T	1
r_1	Input Movement penalty for F	0.01
r_2	Input Movement penalty for Q	0

Using these values, Q and R are constructed, then used to build $\underline{Q}$ and $\underline{R}$ as shown in Equation Set 4.35.

$$Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix} \qquad R = \begin{bmatrix} r_1 & 0 \\ 0 & r_2 \end{bmatrix} \qquad I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$\underline{Q} = \begin{bmatrix} Q & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & Q \end{bmatrix} \qquad \underline{R} = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix} \qquad \underline{I_L} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \tag{4.35}$$

The sizes of these matrices are governed by the nature of the case study, based on the values in Table 4.2 and following the procedure in the General MPC Algorithm section of the Appendix. The $\underline{A}$ matrix is then constructed, shown in Equation 4.36.

$$\underline{A} = \begin{bmatrix} CB & 0 \\ CAB + CB & CB \\ CAAB + CAB + CB & CAB + CB \end{bmatrix} \tag{4.36}$$

The state space model matrices used here are the $A_{MPC,d}$, $B_{MPC,d}$ and $C_{MPC,d}$ matrices declared in the Preliminaries section. The matrices in Equation Sets 4.35 and 4.36 can now be used to calculate the K_1 value using Equation 4.29. The final two time-independent matrices needed for the MPC algorithm are calculated below

$$\underline{B}_1 = \begin{bmatrix} CA \\ CAA \\ CAAA \end{bmatrix} \underline{B}_2 \quad = \begin{bmatrix} CB \\ CAB \\ CAAB \end{bmatrix} \tag{4.37}$$

where the state space model matrices used are the same as those used in Equation 4.36. After the time-independent components of the MPC algorithm are constructed outside the control loop, the loop can begin and several matrices must be recalculated each iteration. These steps are covered in Section 4.3.3.

References

[1] T. Tosukhowong, J. M. Lee, J. H. Lee, and J. Lu. "An introduction to a dynamic plant-wide optimization strategy for an integrated plant". In: *Computers & Chemical Engineering* 29.1 (2004), pp. 199–208.

[2] H. Li and C. L. E. Swartz. "Approximation techniques for dynamic real-time optimization (DRTO) of distributed MPC systems". In: *Computers & Chemical Engineering* 118 (2018), pp. 195–209.

[3] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 2001.

[4] M. Z. Jamaludin and C. L. E. Swartz. "Approximation of closed-loop prediction for dynamic real-time optimization calculations". In: *Computers & Chemical Engineering* 103 (2017), pp. 23–38.

[5] G. Welch and G. Bishop. "An Introduction to the Kalman Filter". In: *University of North Carolina at Chapel Hill Technical Report* (1997).

[6] C. Paige. "Properties of numerical algorithms related to computing controllability". In: *IEEE Transactions on Automatic Control* 26.1 (1981), pp. 130–138.

[7] E. Walter, L. Pronzato, and J. Norton. *Identification of parametric models from experimental data*. Vol. 1. 2. Springer, 1997.

[8] X. Li, L. Zhang, M. Nakaya, and A. Takenaka. "Application of economic MPC to a CSTR process". In: *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*. IEEE. 2016, pp. 685–690.

[9] J. Matias, J. P. Oliveira, G. A. Le Roux, and J. Jäschke. "Steady-state real-time optimization using transient measurements on an experimental rig". In: *Journal of Process Control* 115 (2022), pp. 181–196.

[10] B. Chachuat, A. Marchetti, and D. Bonvin. "Process optimization via constraints adaptation". In: *Journal of Process Control* 18.3-4 (2008), pp. 244–257.

Chapter 5

Conclusion

This study proposes Kalman filter (KF) estimation with CL-DRTO and studies both MPC and PI control layers. The research's significance lies in its ability to handle input saturation and various disturbance sources, particularly under nonlinear conditions and frequent target changes. This work implements input constrained CL-DRTO to address input saturation challenges. Traditional methods of handling input saturation rely on Boolean logic and are complex and difficult to scale. By incorporating input bounds from the controller into the DRTO layer's optimization problem, the proposed solution effectively prevents input saturation, overcoming this obstacle. Chapters 3 and 4 test the proposed Kalman filter with PI and MPC control layers, respectively, and their findings are given here.

5.1 Findings for KF with CL-DRTO and PI Control

In a PI controlled system, controller dynamics can be directly incorporated into the model, giving the Kalman filter additional information in its prediction step. This strategy was compared with the standard Kalman filter strategy (using a model without controller dynamics) as well as the standard for handling plant-model mismatch: bias updating. In the MPC controlled system, the proposed strategy was tested handling under various sources of disturbance. An ideal scenario of full state feedback was also run in parallel for comparison. All these strategies sent their DRTO setpoints directly to the control layer. In the PI controlled system, the proposed Kalman filter strategy demonstrated excellent economic and target tracking performance, slightly outperforming bias updating even during input saturation. The proposed Kalman filter significantly outperformed bias updating when measurement noise was increased. The resulting addition to the Kalman filter strategy makes the framework more robust, more profitable, and more practical. The proposed strategy was also tested on a nonlinear system using an extended Kalman filter, demonstrating excellent performance even under input saturation. While linearization is often desirable for efficiency and simplicity, it may not be feasible in highly nonlinear or transient systems. This approach demonstrated an example of how to control a nonlinear process using nonlinear models and nonlinear estimators, further increasing the scope of the proposed solution. This work has sought to improve on recently proposed Kalman filter estimation techniques in DRTO, specifically in highly constrained systems. Future work on the subject may be to approximate the behavior of input clipping a the DRTO level using smoothed switching functions to approximate Boolean logic in continuous optimization setting. Determining how well these smoothing functions linearize and can be manipulated by an optimizer requires more study. Additionally, there are many applications where PI-control is insufficient, and an MPC layer is needed to control the PI-layer. Expanding this methodology on a MPC-based system would expand its scope for practical use.

5.2 Findings for KF with CL-DRTO and MPC Control

The Kalman filter also performed well in an MPC controlled system, with input constrained CL-DRTO adhering to constraints and accurately converging on the true values of the states and unknown parameter. The Kalman filter was tested under various sources of plant-model mismatch. By linearizing the MPC around a different steady state than the DRTO model and plant were linearized, there was mismatch between the control layer models and between the MPC model and plant. This chapter also used linear architecture to control a nonlinear plant, inherently introducing plant-model mismatch between the plant and all the control layers. On top of these mismatch sources, the architecture was given infeasible targets to induce input saturation, a particular difficulty for estimators. Despite these sources of mismatch and input saturation, the MPC layer and estimator were both able to overcome their obstacles; the control architecture was able to achieve effective control and accurate state and parameter estimation in under all these conditions. The findings did suggest that as plant-model mismatch increases, the estimator is further perturbed from the steady state value and takes longer to converge. However, this effect is shown to be negligible in the case study tested. The integration of the Kalman filter in this architecture is a significant advancement. It successfully converged on true values despite model mismatches and input saturation. The CL-DRTO relies on these estimated states and parameters to accurately predict the behavior of the controller and plant throughout the prediction horizon. Without an accurate estimation of model values, the CL-DRTO would send set-points that drive the controller beyond its bounds. This enhancement is particularly relevant for dynamic systems and start-up operations, where rapid convergence and accurate control during transitions are crucial. By controlling nonlinear plants using linear control architecture, the study demonstrates that the DRTO, MPC, and Kalman filter layers can manage nonlinear behavior with minimal performance degradation, a crucial capability for scalability in larger, more complex systems. Further research is needed to explore the full potential of this approach, particularly in systems undergoing multiple steady-state changes.

The scalability of this approach to larger, more complex systems remains an area for future exploration. This work also focuses mainly on estimating a single unknown parameter. The effect of converging on multiple unknown values and their potential interaction requires further study. Input constrained CL-DRTO outperformed the DRTO without input constraints, but it would be more valuable to compare this methodology to rigorous handling of input saturation, for example through Boolean representations using complementarity constraints.

5.3 Final Insights

These research areas and the strategy proposed here represent promising potential for estimators to become more robust and accessible to industry, where noise, missing measurements and input saturation are a common occurrence. By testing the estimation technique in both PI and MPC controlled systems, this research offers a robust, efficient, and scalable solution for controlling nonlinear plants and dynamic systems, laying a solid foundation for future research and practical applications in various industries.