ADVANCING MULTIPARAMETRIC PROGRAMMING FOR MODEL PREDICTIVE CONTROL

A Dissertation

by

JUSTIN KATZ

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Efstratios N. Pistikopoulos
Committee Members,	Costas Kravaris
	Eduardo Gildin
	Joseph Sang-II Kwon
Head of Department,	Arul Jayaraman

May 2020

Major Subject: Chemical Engineering

ABSTRACT

Model predictive control provides the optimal operation for chemical processes by explicitly accounting for the system, constraints, and costs. In an online setting, developing the implicit optimal control action under time consideration is non-trivial. Over a decade ago, it was demonstrated through multiparametric programming that the implicit control law defining the model predictive controller can be determined explicitly, once and offline. The benefit of such an approach is the (i) improved online computational time, (ii) the development of the offline map of solution *a priori*, and (iii) the derivation of the optimal control laws under any state variation. In recent years there has been a significant push for the development of novel algorithms and theoretical advancements for multiparametric model predictive control. These algorithms and theoretical underpinnings have expanded the problem classes that are solvable and improved the computational efficiency. However there is still a need to provide analysis for formulations based on different surrogate models, and to tackle large scale multiparametric model predictive control problems. In this dissertation, the research focus is (i) the inclusion of a new surrogate modeling technique from the machine learning community, (ii) developing a criterion to compare multiparametric model predictive control formulations based on different surrogate models, (iii) the development of an algorithm to solve large scale multiparametric optimization problems, and (iv) improving the online computational performance of online solvers via multiparametric programming. To this end, tools from data science, computational geometry, and the operations research community contributed greatly to the results presented in this work. This research is verified via the optimal operation of chemical engineering processes and the efficacy of the developed algorithms is demonstrated on computational studies.

DEDICATION

To my family and friends.

ACKNOWLEDGMENTS

First, I would like to thank my advisor, Professor Efstratios N. Pistikopoulos. This was a long journey and his continued support and guidance was invaluable. As the old saying goes, he was a bright light in a dark tunnel.

I would also like to thank my dissertation committee members, Professors Costas Kravaris, Eduardo Gildin, and Joseph Sang-Il Kwon for their insightful comments and constructive criticism. The road from start to finish was quite the roller coaster, but things were always kept on a high note with my fellow group members, Dr. Nikolaos Diangelakis, Dr. Styliani Avraamidou, Dr. Gerald Ogumerem, Dr. Melis Onel, Burcu Beykal, Cosar Doga Demirhan, Baris Burnak, William Tso, Yuhe Tian, Iosif Pappas, Marcello Di Martino, Cory Allen, Stefanos Baratsas, Christopher Gordon, and Denis Su-Feher. There is nothing that produces camaraderie like going through a rigorous process together.

In particular I would like to thank my desk mate Baris Burnak, whom I could always rely on for a nice conversation on any topic ranging from research, to social constructs, or even future world leaders. I would like to express further gratitude to Dr. Nikolaos Diangelakis for his crucial guidance during the beginning of this journey. No matter the problem, he was always willing to provide his input. Lastly, I would also like to thank Iosif Pappas for his enthusiasm to discuss research, at any time, day or night.

In addition I would like to express thanks to my family. To my mother and father for always supporting me during my time away from home. My brother and sisters always providing a nice distraction from the difficulties of 'real life'. I would also like to thank my god son, for which his brightness and curiosity is always an inspiration.

Most importantly, I would like to thank my partner Aurora. Without your constant presence this dissertation would have been but a distant fantasy.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professors Efstratios N. Pistikopoulos, Costas Kravaris, and Joseph Sang-Il Kwon of the Artie McFerrin Department of Chemical Engineering and Professor Eduardo Gildin of the Department of the Harold Vance Department of Petroleum Engineering.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Financial support from the NSF SusChEM (Grant No. 1705423), NSF INFEWS (Grant no. 1739977), Texas A&M Energy Institute, and the Rapid Advancement in Process Intensification Deployment (RAPID SYNOPSIS Project - DE-EE0007888-09-04) is gratefully acknowledged.

NOMENCLATURE

ANN	artificial neural network
CR	critical region
CSTR	continuously stirred tank reactor
DSV	decision space volume
MARS	multivariable adaptive regression spline
MILP	mixed-integer linear program
MPC	model predictive control
mpLP	multiparametric linear program
mpMILP	multiparametric mixed integer linear program
mpMIQP	multiparametric mixed integer quadratic program
mpMPC	multiparametric model predictive control
mpNLP	multiparametric nonlinear program
mpQP	multiparametric quadratic program
MPT	multiparametric toolbox
MSE	mean squared error
NC	control horizon
NLP	nonlinear program
NMPC	nonlinear model predictive control
NMSE	normalized mean squared error
OH	output horizon
PAROC	parametric optimization and control
POD	proper orthogonal decomposition

POP	parametric optimization
PRS	psuedo random sequence
PRBS	pseudo random binary sequence
ReLU	rectified linear unit
RMSE	root mean squared error
RTO	real time optimization

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

The advanced process control of a chemical process is a critical component in industry. These strategies ensure the process has the greatest potential to operate at its highest efficiency, while satisfying safety, environmental, operational, and cost considerations. Model predictive control (MPC) is an established advanced process control technology in both academia and industry [3], finding application in many chemical engineering processes [4, 5]. Using a mathematical representation of the process, MPC is an implicit control problem in the form of an optimization formulation. The underlying mathematical model is a vital component in the development of the optimal solution to the model predictive controller, and remains a research focus.

A chemical process is defined by its underlying chemical engineering principles such as mass and energy balances, and thermodynamic relations. Developing highly accurate dynamic mathematical models that represent these complex phenomena results in systems consisting of differential algebraic equations (DAE), or in some instances partial differential algebraic equations (PDAE). Incorporating these complex models directly in an advanced control strategy such as MPC is challenging. In real time applications, where determining the optimal operation of the plant is under time constraints, these detailed descriptions of the plant dynamics requires significant computational cost. Accounting for the need of a mathematical model that captures the process dynamics accurately, and ensuring real time operation, surrogate models are utilized. At the cost of accuracy, or plant-model mismatch, these surrogate models are directly amenable to MPC schemes and real time implementation. Numerous approximation methods for model predictive control currently exist in the literature, such as the well established system identification [6] and steady state linearization. Recent advanced techniques include sparse regression, local dynamic decomposition strategies, and Proper Orthogonal Decomposition (POD) [7, 8, 9, 10]. However, with a desire to control more complex processes, it is becoming necessary to incorporate surrogate models that can capture the nonlinear dynamics associated with a chemical engineering process.

Surrogate models based on data-driven techniques from the machine learning community are

a promising direction to pursue. The models are adept at capturing the associated nonlinearities associated with a process and many make no assumptions on the underlying form of the model, making them applicable to a wide variety of applications. In particular neural networks are a class of nonparametric models with the capacity to accurately represent complex phenomena [11]. Direct utilization of neural networks in MPC formulations is limited because the resulting optimization formulation is nonlinear and nonconvex, requiring specialized strategies to develop the solution under time considerations [12].

In online implementations, the computational performance of determining the optimal solution to the MPC problem can be improved with multiparametric programming. Multiparametric programming is a proven method to alleviate the online computational burden of determining the optimal solution to an MPC problem. It is a methodology that exactly transforms the implicit optimization problem that defines the model predictive controller into an explicit, offline map of solutions. The map of solutions provides a direct relationship between the optimal manipulated actions to the key process parameters, such as disturbances, set points, and plant measurements. The development of the multiparametric solution for multiparametric model predictive control (mpMPC) started with the seminal work by Bemporad et al. [13]. This work demonstrated that the solution of a mpMPC directly relates optimal manipulated actions to process parameters by identifying a unique combination of active constraints. Further research efforts have produced complete algorithms regarding the development of the multiparametric solution for multiparametric linear (mpLP) [14], quadratic (mpQP) [15, 16, 2], mixed-integer linear (mpMILP) [17, 18], and mixed-integer quadratic programming (mpMIQP) problems [19].

Formulating a mpMPC problem to develop the multiparametric solution requires the underlying mathematical model to be linear or piecewise linear. Utilizing a nonlinear process model requires solving a multiparametric nonlinear programming (mpNLP) problem, which no exact solution strategy exists. Because of the restriction of using a linear or piecewise linear model, developing advanced models for multiparametric model predictive control is limited. In addition, when these models are used to approximate the original nonlinear process, analyzing the difference

between the parametric solutions has not been studied.

Developing the multiparametric solution involves identifying all possible optimal active constraint combinations. In large scale problems, the possible combinations become prohibitive because of the potential exponential growth. In addition, if a full solution is developed there are online complications as well. These issues include the (i) point location problem, and (ii) the memory requirement to store the full parametric solution. The point location problem refers to identifying which critical region contains the parameter realization. As the number of possible critical regions grows exponentially, determining the correct optimal solution becomes computationally expensive. Also, because of the possible exponential growth of the solution, storing the resulting critical regions incurs a significant memory requirement. Promising research is being conducted in these fields to (i) obtain the multiparametric solution for large scale problems and (ii) improve its online performance [15, 2, 20, 21, 16]. However, these strategies may not be enough to accommodate the potential exponential growth of the resulting multiparametric solution.

The benefits of multiparametric programming theory for developing the offline, explicit solution have also found relevance in improving online optimization solvers in the form of hot-start strategies [22]. These strategies utilize concepts of the parametric solution to improve online computational performance significantly for model predictive control applications. A critical failure of these hot-start strategies is poor initialization. For instance, under situations of disturbances that have not been modeled or plant-model mismatch, a hot-start strategy may be initialized poorly, resulting in significantly deprecated computational performance.

Multiparametric programming provides a significant contribution to model predictive control formulations, both for developing the offline, explicit solution and as a means to improve online strategies. Based on the previous discussions, the following are key open questions:

Question 1 Can more advanced surrogate models be incorporated into multiparametric model predictive control formulations?

Question 2 Is there a viable method to compare the multiparametric solution derived from different multiparametric model predictive control formulations (based on different surrogate

models)?

Question 3 Can a more efficient algorithm be developed to explore the full multiparametric solution for large scale problems?

Question 4 Can the offline multiparametric solution be used to improve the initialization procedure for hot-start strategies for model predictive control?

These open questions lay the foundation for the research directions in this dissertation.

1.1 Dissertation Contributions

This dissertation consists of further improvements to model predictive control through multiparametric programming. The structure is based on two parts. In part I, the development of a class of advanced surrogate models is presented, along with a novel metric to quantify the impact of differing mpMPC formulations. This class of surrogate models is from the machine learning community and represents the drive to integrate more advanced models that still provide a means to develop the offline explicit solution. In addition, a comparison between different surrogate models based on the optimization formulation itself is necessary to supplement existing closed loop metrics. Part II of this dissertation provides algorithmic advances involving multiparametric programming for the solution of large scale MPC problems. The benefits of multiparametric programming are readily usable for small to medium scale problems, however large scale applications remain challenging. The key contributions of this dissertation are provided as follows.

Key Contribution 1 The incorporation of a more advanced surrogate model in the form of neural networks with rectified linear units (ReLU) into multiparametric model predictive control formulations. Unlike previous strategies, the neural network is exactly recast into an MILP formulation to facilitate determining the optimal solution using standard multiparametric programming mixed-integer techniques.

Key Contribution 2 The development of a metric to compare mpMPC formulations. The metric provides an analysis on the difference between the optimization structures by quantifying the

decision space volume. Standard metrics examine the closed loop performance as a means to ascertain mpMPC effectiveness, but are lacking a quantity to describe how the optimization formulations differ.

Key Contribution 3 The development (i) of an algorithm for the solution of large scale multiparametric optimization problems, and (ii) of an analysis of the multiparametric solution based on the accumulated volume of the identified critical regions. Current algorithms focus on full exploration of the space, and do not provide analysis on developing a partial solution when full exploration is not realistic.

Key Contribution 4 A method to develop a partial multiparametric solution for large scale problems via random walks. Random walks provide a strategy to (i) identify larger critical regions, and (ii) scale to high dimensional problems. Current literature is focused on developing the full multiparametric solution, or approximating the solution directly.

Key Contribution 5 The improvement of hot-start strategies via a partial multiparametric solution. Hot-start strategies are effective at speeding up online MPC strategies and there shortcoming is in their ability to ensure an effective initialization. A partial solution facilitates proper initialization during real time implementation.

2. DEEP LEARNING AND MULTIPARAMETRIC PROGRAMMING

Deep learning is a method to approximate complex systems and tasks by exploiting copious amounts of data to develop rigorous mathematical models. These approximate models are based on neural networks which are popular in the chemical engineering literature [23, 24]. As data-driven modeling techniques are seeing increased value and are being adopted by industry [25], the incorporation of deep learning into multiparametric optimization formulations is paramount.

Surrogate modeling for optimization, both as a tool [26, 27] and as a means for approximate model development [28] is an established field. In the chemical engineering literature, the use of neural networks as surrogate models has found success in a variety of contexts [29], such as (i) modeling [30, 31], (ii) optimization and control [12, 11, 32, 33], (iii) regression [23], and (iv) classification [23]. In all of these applications, the developed artificial neural network (ANN) model is used to represent a complex nonlinear process. However obtaining the global solution for the corresponding optimization problem incorporating a neural network poses a significant computational burden due to the inherent nonconvexity introduced [34]. Hence, incorporating neural networks in real-time applications remains a major challenge.

The ability to use neural network models as surrogates in an optimization formulation is contingent on utilizing a strategy to obtain a quality optimal solution. Current literature demonstrates an interest in obtaining 'good' solutions to optimization formulations incorporating deep learning models. Schweidtmann *et al.* [12] proposed a global optimization strategy based on McCormick relaxations to identify the global minimum to optimization problems involving ANNs. Pfrommer *et al.* [35] utilized a stochastic genetic algorithm to find the minimum for a process involving textile draping where a neural network was utilized as a surrogate model. Nagat *et al.* [36] developed a surrogate neural network model for a fermentation process, and optimal operating conditions were identified using a genetic algorithm.

Deep learning has a distinct advantage compared to other surrogate modeling techniques, such as response surfaces, Kriging models, Bayesian networks, and radial basis functions [37]. Because

of the highly connected structure of deep learning models, they are naturally adept at expressing complex functional relationships. Their ability to approximate a function to an arbitrary level of accuracy is because there is an exponential number of piecewise connected hyperplanes based on the size of the neural network [38, 39]. A similar surrogate modeling approach to neural networks with ReLU activation functions is the multivariate adaptive regression spline (MARS), whereby piecewise linear splines are utilized in the same fashion with ReLU activation functions. A benefit of using MARS models is their interpretability and ease-of-training [40]. However, neural networks display a greater capability of prediction and managing nonlinearities [41].

Given an optimization problem with highly complex and nonlinear components, neural networks with rectified linear activation units (ReLU), that have been shown to have high performance for regression based problems [42], are of interest to be incorporated into optimization formulations as surrogate models. In recent work, it was shown that neural networks utilizing ReLU activation functions can be represented exactly in a mixed-integer linear programming (MILP) formulation [43, 44]. An exact recasting of neural networks with ReLU activation functions as an MILP formulations allows a new avenue for incorporating deep learning models into optimization based formulations. With this capability, the gap between model accuracy and computational performance is reduced.

Previous work has demonstrated integrating neural networks with ReLU activation functions into optimization formulations. However, in online applications where time is a critical factor, determining the optimal solution is challenging because of the inherent nonconvexity of the resulting discrete optimization formulation. In these time critical applications, multiparametric programming is a proven methodology to alleviate the online computational burden by developing the optimal solution offline. By incorporating more advanced surrogate models in multiparametric optimization formulations, the benefits of the developed parametric solution are improved, namely (i) the ability to obtain the optimal solution without having to solve an optimization problem each time the uncertain parameter is identified, (ii) having the explicit map of solution a priori, and (iii) having the explicit functional relationship between optimization variables and uncertain parame-

ters. In addition, the explicit map of solution provides benefits in numerous applications including, multiparametric model predictive control [13, 1, 45], scheduling [46], bilevel programming [47], multi-level decision making [48, 49], portfolio selection [50], and instance-weighted support vector machines [51].

In these multiparametric model formulations, a key detail is their dependence on linear or piecewise linear constraints. Therefore, to incorporate more complex phenomena in parametric formulations surrogate modeling is required. Developing accurate surrogate models to represent nonlinear functional relationships is non-trivial, and deep learning models based on ReLU activation functions bridge this gap. Given that complete theory regarding mpMILPs and algorithmic strategies are available in online solvers [52, 53], the integration of neural networks involving ReLU activation functions and multiparametric programming is a natural step. This chapter is based on submitted work [54].

2.1 Motivation

The focus of this chapter is to approximate nonlinear functions in MPC formulations with neural networks composed of ReLU activation functions. In particular, the key contribution is the integration of these deep learning models and multiparametric model predictive control.

Given is a nonlinear optimization formulation in the form of Eq. (2.1).

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & f(\theta, x) \leq 0 \\ & x \in X \\ & \theta \in \Theta \end{aligned} \tag{2.1}$$

where x is the vector of optimization variables, c is the linear cost coefficient, f is the vector of linear/nonlinear constraints, and θ is the vector of uncertain parameters that are known at the time of solving the nonlinear optimization problem.

Determining the optimal solution at every parameter realization (e.g. model predictive control,

reactive scheduling) for this nonlinear optimization problem is computationally expensive. Multiparametric programming is a technique that transforms an optimization formulation involving bounded, uncertain parameters to an explicit functional relationship between the optimal optimization variables and these parameters. To realize the benefits of multiparametric programming, a surrogate model is developed to replace the nonlinear function in Eq. (2.1). The model accuracy of the surrogate model is ensured by utilizing neural networks with ReLU activation functions. Neural networks are inherently nonconvex, nonlinear functions making them difficult to incorporate into optimization problems, however a recent reformulation technique enables their direct use in an MILP based formulation with no information lost. Following the implementation of the neural network surrogate model, Eq. (2.1) is approximated by Eq. (2.2).

$$\begin{aligned} \min_{x,y} \quad & c^T\omega \\ \text{s.t.} \quad & [A\ E]\omega \leq b + F\theta \\ & \omega = [x^T y^T]^T \\ & x \in X,\ y \in \mathbb{Z}_2^n \\ & \theta \in \Theta \end{aligned} \tag{2.2}$$

Determining the optimal solution of an MILP remains computationally demanding. Unlike nonlinear programming formulations with uncertain parameters, mixed-integer linear program with bounded uncertain parameters are solvable using state-of-the-art multiparametric algorithms. Therefore, the online computational burden of determining the optimal solution to an MILP at every parameter realization is eliminated. The developed explicit solution produces a map of solutions, Fig 2.1, where each region is an affine function that relates the optimal optimization variables to the uncertain parameters.

The procedure to integrate deep learning and multiparametric programming is presented in Fig 2.2. The necessary components include data collection and processing, neural network development, MILP reformulation, and multiparametric programming. The following sections provide necessary information regarding (i) neural networks, (ii) multiparametric programming, and (iii)

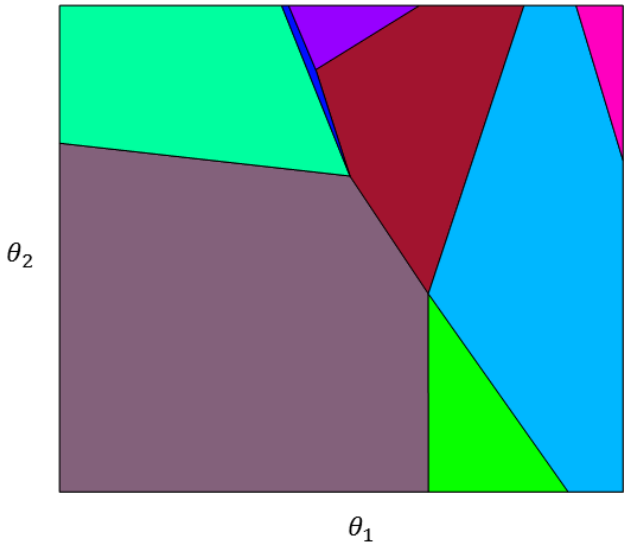


Figure 2.1: The explicit map of solutions to a multiparametric programming problem.

their integration.

2.2 Neural Networks

Neural networks are nonparametric mathematical models that are an effective tool to map input data to output data with a high degree of accuracy. The strength of neural networks is their ability to universally approximate any function [55]. A neural network is defined by its structure, two of which are feedforward and recurrent [56]. In this work, the structure assumed is feedforward because of its proven performance and straightforward structure, shown in Fig. 2.3. The feedforward neural network involves inputs, outputs, an input layer, output layer, hidden layers, weighting parameters, and activation functions. The inputs, outputs, input layer, and output layer are defined by the particular process. For example, Fig. 2.3 has 2 inputs, 1 output, 2 hidden layers, 4 nodes in the first hidden layer, and 2 nodes in the second hidden layer. However, the number of hidden layers and their respective sizes, weighting parameters, and activation functions are all tunable. The number of hidden layers and their respective sizes can be determined through trial and error, and the weighting parameters are typically defined through a local search, such a stochastic gradient descent. Numerous activation functions have shown to be meritous under varying circumstances.

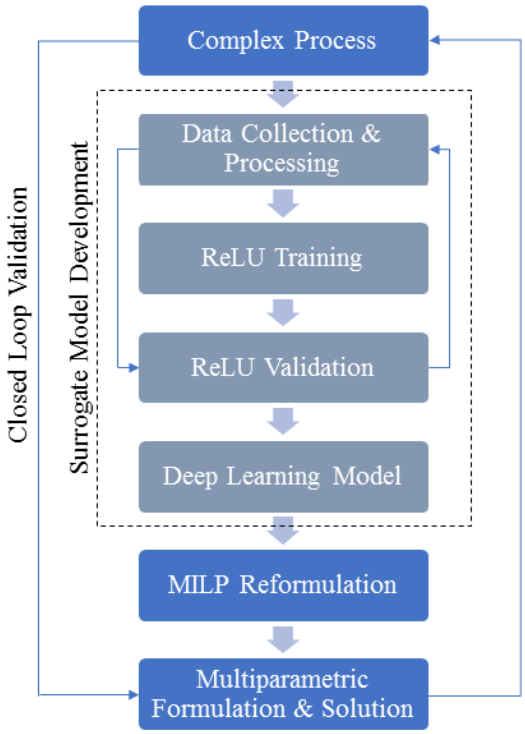


Figure 2.2: The components for integrating deep learning and multiparametric programming.

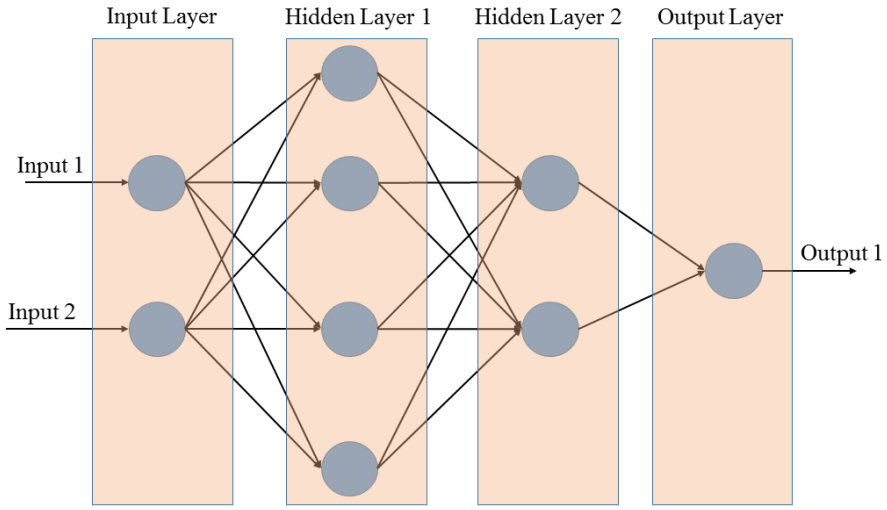


Figure 2.3: The structure of a feedforward neural network with inputs, outputs, and hidden layers.

In this chapter, the ReLU activation function is used throughout the manuscript because it can be exactly reformulated as a MILP.

2.2.1 Training

Training a neural network involves identifying the values for the weights and biases, such that a performance metric is minimized. The training step is critical because it provides the basis for how the neural network will map inputs to outputs. Key factors that must be addressed during training include (i) identifying a performance metric to use, (ii) minimizing the performance metric, and (iii) data selection and processing.

2.2.1.1 Performance Metric

The performance metric is a measure to provide the goodness of fit for the neural network. The metric is based on the deviation between the output of the neural network and the dataset. There are a myriad of performance criteria, such as mean squared error (MSE), normalized mean squared error (NMSE), root mean squared error (RMSE), mean absolute error (MAE), etc. [57]. These metrics are different formulations for quantifying the deviation between the expected output

of the neural network and the output of the dataset. Note that determining which performance criteria should be selected to train the neural network is non-trivial.

2.2.2 Validation

Model validation ensures the trained model is an accurate representation of the dataset. There are many techniques for model validation for data-driven models, such as (i) data partitioning, (ii) cross-validation [58], and (iii) analyzing the residual plot. These techniques provide different approaches for ensuring the regressed model is valid over the dataset.

2.2.3 ReLU Activation Functions

Each node in a neural network's hidden layer is associated with an activation function. Common activation functions are the hyperbolic tangent, sigmoid, and the rectified linear unit. Selecting a suitable activation function is problem dependant, but the rectified linear unit has consistently displayed an ability to perform well on numerous applications, and is the activation function assumed.

The ReLU activation function is defined in Eq. (2.3). It is a piecewise linear function alternatively represented by Eq. (2.4). The alternative representation gives rise to a formalism for introducing a neural network with ReLU activation functions into a mixed-integer optimization formulation.

$$y = max\{0, x\} \tag{2.3}$$

$$y = \delta x,\ \delta = \begin{cases} 1, & \text{if } 0 < x \\ 0, & \text{otherwise} \end{cases} \tag{2.4}$$

The ReLU activation functions approximates a function by deconstructing the original function into a set of piecewise hyperplanes. The number of piecewise hyperplanes that define the ReLU neural network scales exponentially with the number of hidden layers of the neural network, shown in Eq. (2.5) [38, 39].

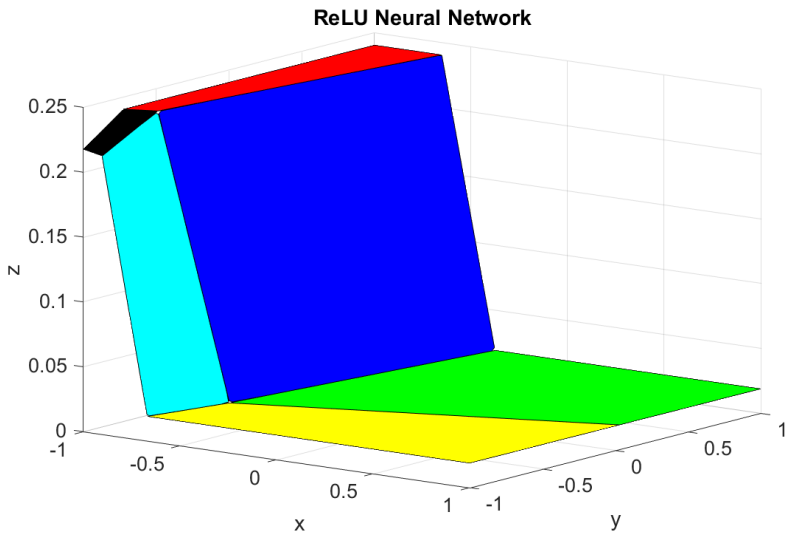


Figure 2.4: A neural network with ReLU activation functions and the representative connected hyperplanes.

$$\Omega\left(\left(\frac{n}{n_0}\right)^{L-1} n^{n_0}\right) \tag{2.5}$$

where L is the number of hidden layers, n_0 is the number of inputs, and n is the number of nodes. The exponential scaling of the number of piecewise hyperplanes is the reason ReLU activation functions in deep learning models are successful. In other words, a deep neural network with ReLU activation functions expresses an arbitrary function with an exponential number of piecewise affine hyperplanes. Fig. 2.4 shows the connected piecewise affine hyperplane structure for an arbitrary neural network, with two inputs, x and y, and one output, z. Leveraging the deep neural networks ability for accurate function approximation makes it an ideal candidate surrogate model for optimization studies.

2.3 Multiparametric Programming

Optimization formulations involving bounded uncertain parameters are defined as a multiparametric programming problem. The multiparametric solution provides a means to derive analytic explicit functional relationships between the optimization variables and these uncertain parame-

ters. A brief review of relevant topics is presented, and further details can be found in the literature [13, 2, 16, 59].

Developing the multiparametric solution has several benefits including (i) determining the optimal solution faster in real time as the uncertainty is revealed, (ii) deriving fundamental knowledge of the optimization formulation via the offline map of solution, and (iii) providing the explicit expressions relating optimization variables to uncertain parameters that can be embedded in multi-level optimization frameworks [49].

In this chapter, we focus on mpMILPs, which are used as the exact reformulation of deep neural networks maintaining ReLU activation functions. Because the solution strategy for mpMILPs is based on continuous mpLPs, a brief discussion on mpLPs is provided in the subsequent sections. A discussion regarding developing the full multiparametric solution for mpMILPs is then presented in Section 2.3.2.

2.3.1 Fundamental Concepts

A generic mpLP has the following form

$$\begin{aligned} \min_{x} \quad & c_x^T x \\ \text{s.t.} \quad & A_i x \leq b_i + F_i \theta, \ \forall i \in \mathbb{I} \\ & A_j x = b_j + F_j \theta, \ \forall j \in \mathbb{J} \\ & \theta \in \Theta := \{\theta \in \mathbb{R}^m \,|\, CR_A \theta \leq CR_b\} \\ & x \in \mathbb{R}^n \end{aligned} \tag{2.6}$$

where the matrices $A_i \in \mathbb{R}^{1\times n}$, $F_i \in \mathbb{R}^{1\times m}$, $A_j \in \mathbb{R}^{1\times n}$, $F_j \in \mathbb{R}^{1\times m}$ and the scalars b_i, b_j correspond to the i^{th} and j^{th} inequality and equality constraints of the sets $\mathbb{I}$ and $\mathbb{J}$ respectively.

The multiparametric solution of Eq. (2.6) returns a list of critical regions, and each critical region defines affine functions relating the bounded uncertain parameters to the optimal continuous decision variables, Eq. (2.7).

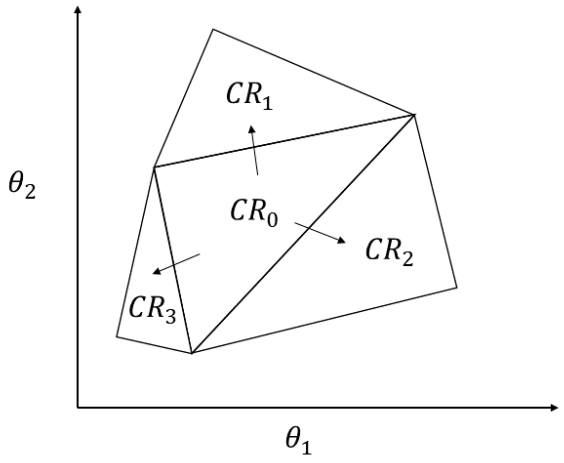


Figure 2.5: Illustrative representation of the exploration of the parameter space using a geometrical algorithm for multiparametric programming.

$$x^* = K_i\theta^* + r_i,\ \theta^* \in CR^i = \{CR^i_A\theta \leq CR^i_b\} \tag{2.7}$$

where x^* is the optimal solution at the parameter realization θ^*, CR^i define the i^{th} critical region, and K_i and r_i define the affine expression for the i^{th} critical region.

The development of the multiparametric solution has been addressed using three types of algorithmic approaches (i) geometric-based strategies [13], (ii) active set-based strategies [15], and (iii) combinations of geometric and active set methodologies [16].

Geometric-based algorithms are founded on the following notion: given a critical region, the neighborhoods area is explored to identify adjacent critical regions, Fig 2.5. These identified critical regions are subsequently utilized to explore their neighboring critical regions, until the full parametric solution is determined.

On the other hand, combinatorial approaches aim to implicitly explore the parameter space through enumeration and efficient pruning criteria of all possible active set which can yield an optimal solution for a feasible parameter realization, Fig 2.6. Because each optimal active set

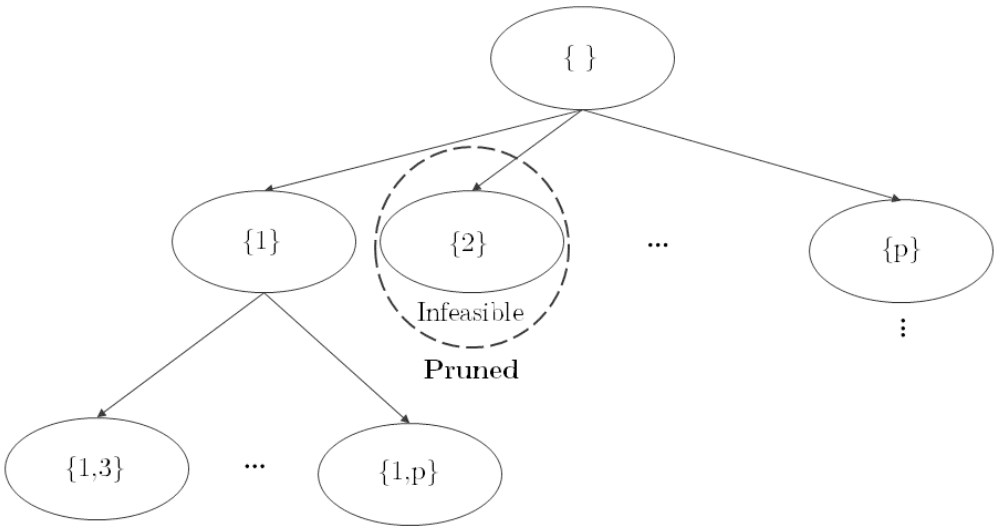


Figure 2.6: Illustrative representation of the exploration of the parameter space using a combinatorial algorithm for multiparametric programming.

yields a critical region, the identification of the complete map of solutions is achieved.

Each of the approaches offers its own advantages and drawbacks in regards to developing the solution of the multiparametric optimization problem, and the selection of the algorithmic strategy depends on the structure of each particular problem [60]. Such algorithms are provided in the state-of-the-art software, Parametric OPtimization (POP) Toolbox [52] and Multiparametric Toolbox (MPT) [53].

2.3.2 Multiparametric Mixed-Integer Linear Programming

A vital element for the integration of multiparametric programming and deep learning using ReLU activation functions are mpMILPs, which are represented by Eq. (2.2). Developing the full multiparametric solution for mpMILPs is significantly more computationally taxing than for mpLPs or mpQPs. However, there exists full theory and algorithms to solve these challenging problems.

An overview of the solution procedure is as follows. Given the mpMILP, a candidate set of critical regions is determined for a particular combination of binary variables. In an iterative

manner, each critical region is examined to identify if an improved solution exists for a different binary combination. The comparison procedure is performed by introducing integer cuts into the original mpMILP within the confines of the critical region under examination. The resulting map of solutions provides the optimal critical regions, the optimization variables as functions of the uncertain parameters, and the optimal integer variables, Eq. (2.8).

$$\left.\begin{aligned} x^* &= K_i\theta^* + r_i, \\ y^* &= y_i, \end{aligned}\right\} \quad \theta^* \in CR^i = \{CR^i_A\theta \leq CR^i_b\} \cdot \tag{2.8}$$

Note that improving mpMILP algorithms is accomplished via strengthening the comparison procedure, with the goal of identifying fewer candidate critical regions.

2.4 Integration

Current theory regarding multiparametric programming indicates the optimization formulation must have linear or piecewise linear constraints with a linear or convex quadratic objective function. Hence, the description of nonlinear constraints or objective functions that naturally appear in real processes cannot be achieved directly in multiparametric programs. Therefore, surrogate models are a necessity to capture such complex relationships. Because of their strong predictive capabilities, deep learning models using ReLU functions are ideal candidates to be incorporated into multiparametric programming formulations.

In the remaining of this section, the strategy for integrating deep learning models and multiparametric programming is presented. First the multiparametric nonlinear programming (mpNLP) problem is formulated. The nonlinear equations are then approximated using a deep neural network with ReLU activation functions. The neural network is then recast as a MILP. The mpNLP is reformulated to incorporate the MILP representation of the neural network, yielding a mpMILP. With the developed multiparametric programming problem formulated, the problem is solved using the POP toolbox. Further details are provided in the subsequent sections.

2.4.1 Optimization Formulation

Eq. (2.9) defines a mpNLP with a linear objective and bounded uncertain parameters, θ. In addition, it is assumed the objective function is linear for clarity purposes only, and is not a restriction of the proposed integration strategy.

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & h(x,\theta) = 0 \\ & g(x,\theta) \leq 0 \\ & x \in X \\ & \theta \in \Theta \end{aligned} \tag{2.9}$$

where c is a cost vector, $x \in \mathbb{R}^n$ is the vector of optimization variables, $\theta \in \mathbb{R}^m$ is the vector of uncertain parameters, and h and g are nonlinear relationships between optimization variables and uncertain parameters. Some examples of the nonlinear constraints defined by h and g are thermodynamic relations, disturbance models, or path constraints. The nonlinear constraints defined by h and g are considered to be complex, nonlinear, and nonconvex ensuring the development of the solution for a given parameter realization of the resulting NLP is challenging. Reducing the complexity of h and g improves the computational performance of determining the optimal solution for a given parameter realization.

2.4.2 Neural Network Model Development

The nonlinear functions h and g in Eq. (2.9) are approximated using deep neural networks with ReLU activation functions. The neural network is developed through data sampling, training, and validation. Note that each nonlinear function can be represented as a deep neural network. However, it may be advantageous to group functions that are related to reduce the total number of neural networks used in the final formulation.

Prior to model development, it is critical to develop a suitable dataset. The input space to the functions h and g are sampled to construct the input/output dataset. The relationship between

the inputs of the functions to the outputs is important to understand prior to invoking a sampling procedure. For instance, if the functions being approximated are probability distributions, the sampling procedure should follow Monte Carlo techniques or Markov Chain Monte Carlo methods [61]. In the case h and g are defined by a dynamic model, system identification techniques are used to properly sample and excite the system to achieve the desired relationship between the inputs and outputs [6]. Following sampling the space of the nonlinear functions, the dataset is normalized to improve training the neural network model.

Once a suitable input/output dataset is constructed, the neural network is defined by specifying the number of hidden layers and the number of nodes in each hidden layer. Determining the optimal number of hidden layers and nodes is not trivial, and in many cases heuristically searching for a suitable size is sufficient.

2.4.3 Reformulation

As demonstrated in the literature [43, 44], a neural network involving ReLU activation functions can be exactly recast as a MILP following a big M formulation. Using a MILP formulation allows the neural network to be embedded into an optimization problem directly without the associated difficulty of a composite function with the nonlinear max operator. The procedure is presented as follows.

For an arbitrary layer with n nodes, the output takes the form of Eq. (2.10), where k is the layer, W^k is the matrix of weights for layer k, $\hat{b}^k$ is the vector of biases for layer k, $x^{k-1} \in \mathbb{R}^n$ is the output of the previous layer, and $x^k \in \mathbb{R}^n$ is the output of the current layer. The max operator is taken element-wise.

$$x^k = max\{0, W^k x^{k-1} + \hat{b}^k\} \tag{2.10}$$

The importance of the ReLU activation function is its piecewise linear nature. Therefore, Eq. (2.10) can be exactly recast in an optimization formulation via the inclusion of binary variables.

Eq. (2.11) is the reformulation of the k^{th} hidden layer in an MILP structure[1].

$$W^k x^{k-1} + \hat{b}^k = x^k - s^k \tag{2.11a}$$

$$x^k \leq M_1 y \tag{2.11b}$$

$$s^k \leq -M_2(1-y) \tag{2.11c}$$

$$x^k \geq 0 \tag{2.11d}$$

$$s^k \geq 0 \tag{2.11e}$$

$$y \in \{0,1\}^n \tag{2.11f}$$

In (2.11), y is a vector of binary variables, $s^k \in \mathbb{R}^n$ is the vector of slack variables, and M_i is a large scalar value. To improve the computational performance of identifying the optimal solution, the big M values, M_1 and M_2, are defined to tightly bound the optimization variables x^k and s^k [62]. If the big M values are well defined, an unnecessary computational burden is avoided. After recasting, the total number of binary variables is equal to the total number of nodes that constitute the hidden layers. Note that the recasted neural network with ReLU activation function is an exact reformulation. The binary variables enable the activation function to output a value of 0 or x, via the constraints (2.11b) and (2.11c). Incorporating the recasted neural network into an optimization formulation provides an effective strategy to maintain high accuracy with a surrogate model, and obtaining the global optimum does not require specialized global optimization techniques. Instead, standard mixed-integer linear programming methods are utilized such as cutting planes and branch and bound techniques which are readily available in existing software.

During training, the number of active nodes (nodes that take a value other than 0) can be reduced using regularization techniques. Minimizing the number of active nodes directly corresponds to a reduction in the number of binary variables needed during the recasting procedure.

[1]The final layer, or the output layer, does not require the modification presented in (2.11) because the associated activation function layer is linear

By reducing the number of binary variables needed to define the neural network, the computational burden of solving the final optimization problem is significantly diminished. After training, postprocessing the neural network model also has the potential to minimize unnecessary binary variables. For instance, a node in a hidden layer that is always positive is represented by a linear activation function. Hence, that node does not require a slack variable and a binary variable. Because of the mixed-integer nature of the optimization formulation, any reduction in the number of binary variables has the potential for impacting the computational performance notably.

2.4.3.1 Variable Aggregation

Following Eq. (2.11), the neural network is transformed to a system of equality constraints, inequality constraints, binary variables, and slack variables. Because the equality constraints are linear, the intermediate optimization variables can be eliminated via variable aggregation. By eliminating intermediate variables, the number of optimization variables and constraints is reduced, which is important in developing the multiparametric solution.

$$x^1 = W^1 x^0 + \hat{b}^1 + s^1 \tag{2.12a}$$

$$x^k = \prod_{i=k}^{1} W^i x^0 + \sum_{i=1}^{k-1} \prod_{j=k}^{i+1} W^j (\hat{b}^i + s^i) + \hat{b}^k + s^k,\ k = 2, \ldots, K-1 \tag{2.12b}$$

$$x^K = \prod_{i=K}^{1} W^i x^0 + \sum_{i=1}^{K-1} \prod_{j=K}^{i+1} W^j (\hat{b}^i + s^i) + \hat{b}^K \tag{2.12c}$$

where W^k and $\hat{b}^k$ define the weights of the k^{th} hidden layer, K is the output layer, s^k is the vector of slack variables, x^0 is the input vector to the neural network, and x^K is the vector of outputs of the neural network. Equations (2.11) and (2.12) are combined to provide a set of constraints in the form of Eq. (2.2).

2.4.3.2 The Importance of Sparsity

Introducing sparsity into the neural network has a profound impact on the resulting optimization formulation. Because sparsity aims to minimize the active nodes in the neural network, the

number of optimization variables and constraints are reduced. Furthermore, by reducing the size of the final optimization problem, the resulting parametric solution requires less offline computational effort, and maintains fewer defining critical regions. The offline computational performance is improved because the complexity of mpMILPs scales with the number of optimization variables and constraints.

2.4.4 Final mpMILP Formulation

The final multiparametric formulation is in the form of an mpMILP, Eq. (2.2). The mpMILP contains the inherent linear components of the optimization problem, such as box constraints, and the embedded neural network. By using a neural network to approximate the nonlinear parts of the mpNLP, the optimization formulation is tractable and provides an optimal solution resembling the true optimal. Another key feature of the resulting mpMILP is that the full parametric solution can be developed, unlike the original mpNLP, using the POP toolbox.

2.5 Results

The effectiveness of the proposed integration of deep neural networks with ReLU activation functions and multiparametric programming is demonstrated on two separate processes. The first example is a reaction and separation process with the aim to maximize product concentration and minimize by-product concentration by manipulating cooling, reactant, and recycle flowrates. The second example is the ACUREX solar field, which is based on the ACUREX solar field in Spain. The field utilizes solar energy to heat a liquid medium, and the control objective is to maintain a temperature setpoint via manipulating the flowrate of liquid into the process.

2.5.1 Reaction and Separation Process

The example chemical process is adapted from Tippett *et al.* [4]. It includes 2 nonisothermal continuously stirred tank reactors (CSTR), 1 flash separator, and 1 recycle stream. The process maintains an exothermic, irreversible series reaction following $A \rightarrow B \rightarrow C$, where B is the product of interest and C is the by-product. The reactions follow an Arrhenius rate law expression. It is also assumed the reactors have constant molar hold up and the species have constant relative

volatility. The mathematical model, provided in Appendix A.2, consists of 11 differential equations defining the mass and energy balances.

Manipulated variables of the process are the flowrates of material into the cooling jacket and reactors, the heat duty into the separator, and the recycle stream. The outputs of interest for the process are the temperatures of the reactors and separators, and the mole fractions of product B and by-poduct C in the separator. The control objective is to meet safety considerations, maximize product concentration, and minimize by-product concentration.

Formulating the control problem involves a (i) linear cost function, (ii) constraints to ensure safety considerations and system requirements, and (iii) the nonlinear set of equations defining the chemical process. Therefore, the control problem is a nonlinear multiparametric optimization problem. To address the nonlinearity and complexity of the chemical process, a neural network with ReLU activation functions is developed as a surrogate model. Reformulating the neural network in a mixed-integer representation, the nonlinear multiparametric controller is transformed to a multiparametric mixed-integer linear programming problem.

2.5.1.1 Training the Neural Network

To train the neural network, input/output data is generated. The input signal developed was based on a pseudorandom sequence (PRS) to properly excite the process. The input/output data used is normalized between $[-1, 1]$ to assist in training the neural network and presented in Fig. 2.7.

The generated neural network has two hidden layers, with 4 and 3 nodes respectively. The neural network is trained using the MATLAB neural network toolbox. The Levenberg-Marquardt backpropagation technique is used to train the neural network model, and a MSE of 1.1×10^{-4} is recorded. To ensure the trained model does not overfit the data, the input/output data set is split between 70% training, 15% validation, and 15% testing.

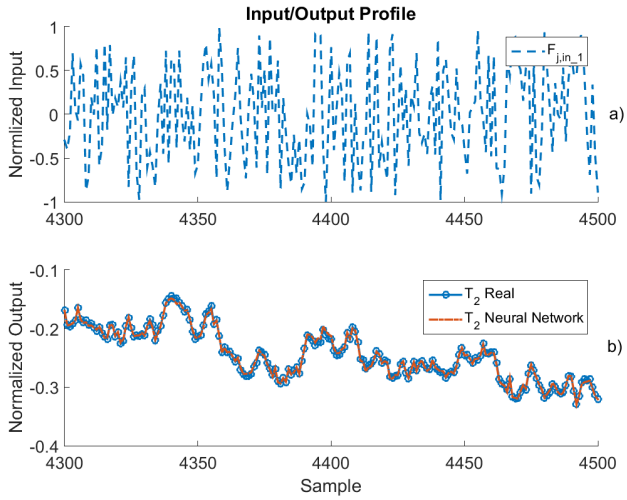


Figure 2.7: a) A snapshot of the input data set used. b) A snapshot of the output of the real process and neural network.

2.5.1.2 Controller Formulation

The controller formulation is presented in (2.13). The objective is to maximize the concentration of the desired product, B, and minimize the concentration of by-product, C, while ensuring the reactor and separator temperatures stay at safe operating conditions.

$$\begin{aligned} \max_{u} \quad & x_{b3} - x_{c3} \\ & x^{+} = f(x_0, u) \\ & \underline{\mathrm{x}}^{+} \leq x^{+} = [x_{b3}, x_{c3}, T_1, T_2, T_3]^T \leq \bar{x}^{+} \\ & \underline{\mathrm{x}}_0 \leq x_0 = [x_{b1,0}, x_{b2,0}, x_{b3,0}, T_{1,0}, T_{2,0}, T_{3,0}]^T \leq \bar{x}_0 \\ & \underline{\mathrm{u}} \leq u = [F_{in\_1}, F_{in\_2}, F_R, F_{j,in\_1}, F_{j,in\_2}, Q]^T \leq \bar{u} \end{aligned} \tag{2.13}$$

where f is the trained neural network that approximates the process, the inputs to the neural network are the mole fraction of product in the reactors ($x_{b1,0}$ and $x_{b2,0}$), the mole fraction of product in the separator ($x_{b3,0}$), the temperatures of the reactors and separators ($T_{1,0}$, $T_{2,0}$, and $T_{3,0}$), the flowrates of material into the reactors (F_{in_1}, F_{in_2}, and F_R), the flowrate of cooling material into the cooling jacket (F_{j,in_1} and F_{j,in_2}), and the heat duty to the separator (Q). The

variable x^+ defines the outputs at the next time step (x_{b3}, x_{c3}, T_1, T_2, and T_3). The controller formulation is recast as a multiparametric model predictive controller involving integer decision following the discussion presented in Section 2.4.

2.5.1.3 Multiparametric Model Predictive Controller

The mpMPC has 6 uncertain parameters corresponding to the initial states of the system, 7 binary variables associated with the number of total hidden nodes in the neural network, and 13 continuous optimization variables defining the manipulated actions and slack variables. The POP toolbox was used to solve the mpMILP, and produces 1271 critical regions.

2.5.1.4 Closed Loop Performance

The process is simulated using the explicit model predictive controller. We see from Fig. 2.9 the closed loop response of the plant demonstrates good performance of the combined explicit model predictive controller and deep learning model. The process is simulated for 4 hours, with a sample time of 2 minutes. At each time step, the explicit model predictive controller aims to minimize the by-product concentration while maximizing the product concentration. It is evident from Fig. 2.9 that the concentration of product B is maximized while minimizing by-product C. The controller is able to leverage a centralized view of the plant to manipulate several variables to obtain an optimal closed loop response. The response drives the system to a steady state with minimal oscillations in the output of the process. Furthermore, the explicit controller maintains safety constraints by ensuring the temperature of the reactors and separator remain within their specified bounds.

2.5.2 ACUREX Process

The ACUREX solar field uses parabolic mirrors to focus the sun's rays on a pipe carrying an energy conversion fluid, typically oil, as seen in Fig. 2.10. The process makes use of renewable solar energy to maintain the temperature of the energy conversion fluid at a predefined setpoint. The fluid, at the proper temperature, is then utilized further downstream to either (i) produce steam that is used to generate power in a turbine, (ii) or as a heat exchange medium in a desalination

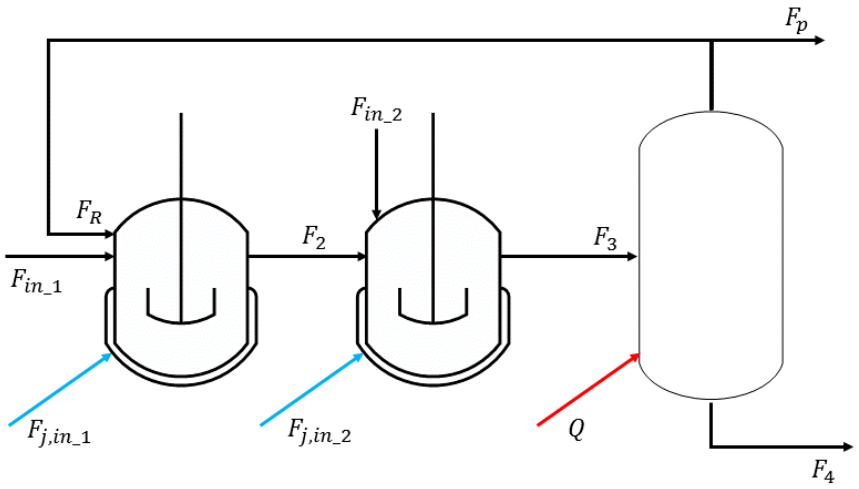


Figure 2.8: A schematic of the chemical process involving reaction and separation.

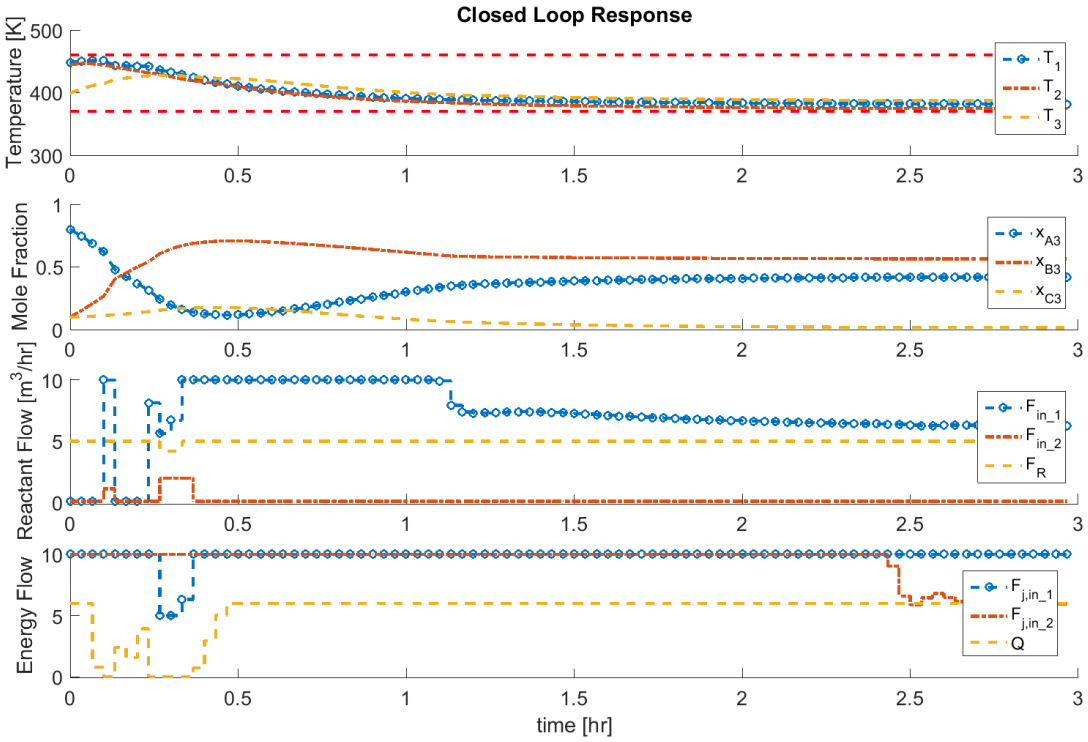


Figure 2.9: The closed loop response of the chemical process with the derived explicit model predictive controller: a) The temperature profile for the reactors and separator. b) The mole fractions of material A, B, and C. c) The flowrates of material into the process. d) The cooling flowrates and energy into the process.

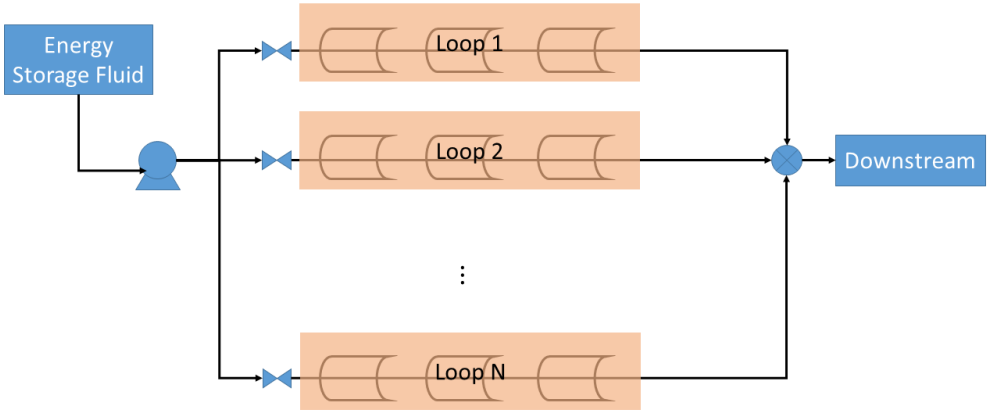


Figure 2.10: A schematic diagram of the ACUREX process.

process [63, 64]. The fluid in the pipeline is assumed to be Therminol 55 [65] because of its workable temperatures of $-28°C$ to $290°C$.

The ACUREX process is modeled by a system of partial differential equations. Following the assumptions by [63] and the additional assumption of a single loop of parametric mirrors, the system of partial differential equations is reduced to a single first order partial differential equation, (2.14).

$$\rho_f C_f A_f \frac{\partial T_f}{\partial t} = \eta_0 G I - h_l (T_f - T_a) - \rho_f C_f u(t) \frac{\partial T_f}{\partial x} \tag{2.14}$$

where T_f is the fluid temperature, x is the spatial coordinate, t is time, T_a is the ambient temperature, ρ_f is the density of the fluid, C_f is the heat capacity of the fluid, u is the fluid flowrate, η_0 is the mirror efficiency, G is the mirror aperture, I is the solar irradiance, and h_l is the coefficient of global heat losses. The partial differential system is converted to a system of ordinary differential equations that depend on time by discretizing the spacial domain via the method of lines [66]. It was found that 57 discretization points sufficed to develop a proper open loop profile.

The empirical relationship between the temperature of the fluid in the pipe and the density and

heat capacity is defined by (2.15).

$$\rho_f = 903 - 0.672T_f \tag{2.15a}$$

$$C_f = 1820 + 3.478T_f \tag{2.15b}$$

Given the high fidelity model of the process, the control objective is to maintain the temperature of the oil at $280\ ^\circ C$. The objective function is defined by the L_1 norm between the temperature of the oil at the exit of the pipe and the setpoint. To meet the control objective, the volumetric flow rate of oil into the system is the manipulated action. There are box constraints on the manipulated action that define the limits of operation, and there is an upper bound on the temperature of the fluid in the pipe for safety and material considerations.

2.5.2.1 Training the Neural Network

Given the dynamic model representing the ACUREX solar field, it is challenging to directly implement the high fidelity model in a model predictive control scheme. Therefore, a neural network with ReLU activation function is utilized as a surrogate model to capture the nonlinear dynamics, while minimizing model complexity when embedded in the explicit model predictive controller.

The ACUREX plant is perturbed in open loop to properly excite the system over a range of inputs. The perturbed inputs are the solar irradiance and the fluid flow to the pipe. The measured output of the system is the temperature at the end of the pipe. Data collection accounts for $10,000$ samples and are used to train the neural network. The collected samples are split into training, validation, and testing sets following a $70\%, 15\%$, and 15% division. Through trial and error, the size of the neural net that was found to accurately fit the open loop data has three hidden layers with sizes 8, 7, and 7 respectively. The fit neural network has a mean squared error of $4 \cdot 10^{-4}$ for the validation set, and the open loop response for the entire data set is presented in Fig. 2.11.

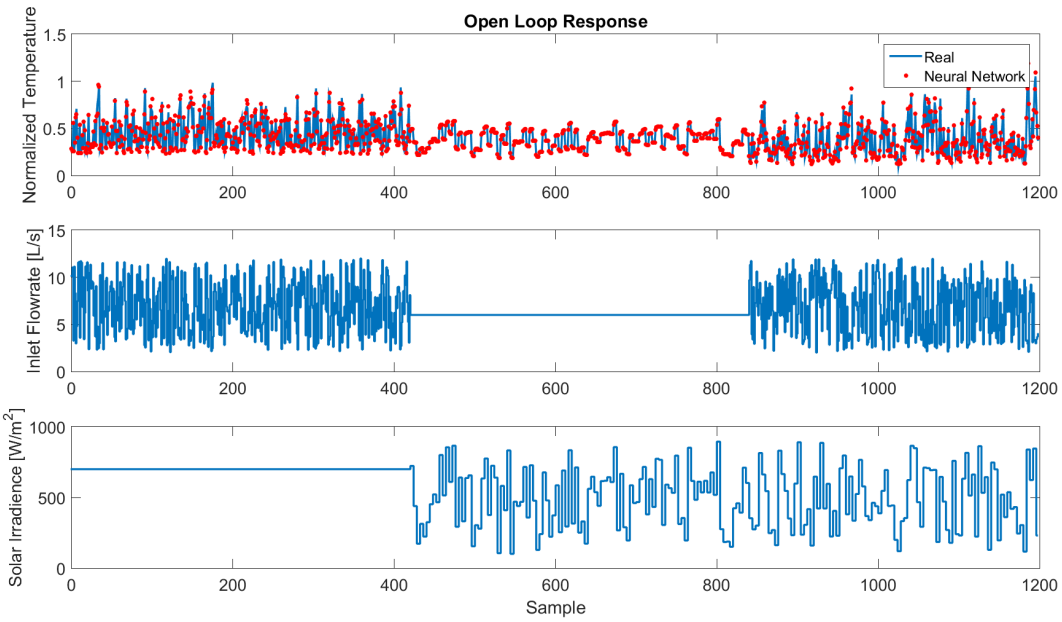


Figure 2.11: Open loop response of the ACUREX solar field.

2.5.2.2 *Controller Formulation*

$$\begin{aligned} \min_{Q} \quad & |T_{end}^{+} - T_{ref}|_{\infty} \\ & T_{end}^{+} = f(T_0, Q, I) \\ & \underline{\mathbf{T}} \leq T_{end}^{+} \leq \bar{T} \\ & \underline{\mathbf{Q}} \leq Q \leq \bar{Q} \\ & \underline{\mathbf{I}} \leq I \leq \bar{I} \\ & \underline{\mathbf{T}} \leq T_0 \leq \bar{T} \end{aligned} \tag{2.16}$$

where T_{end}^{+} is the predicted temperature at the end of the pipe, Q is the volumetric flowrate into the pipe, and I is the solar irradiance. The objective is to maintain the outlet temperature at the setpoint T_{ref} while ensuring the temperature remains within appropriate operating bounds under the time varying disturbance of solar irradiance.

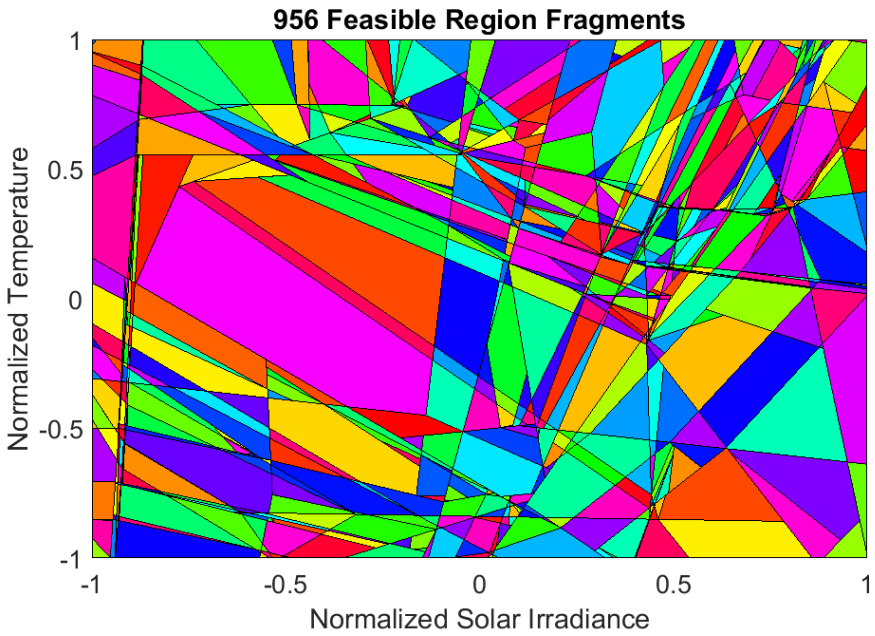


Figure 2.12: The critical regions for the multiparametric model predictive controller.

2.5.2.3 Multiparametric Model Predictive Controller

The developed neural network with ReLU activation function is implemented in an mpMPC. The explicit controller has an output and control horizon of one. Increasing the horizon length is possible by (i) feeding the output of the neural network back into itself, or (ii) during training incorporate future outputs into the neural network directly. For the presented process, it was found a horizon of one was sufficient for a suitable closed loop response.

The mpMPC is represented by 24 continuous optimization variables, 22 binary variables, 2 uncertain parameters, and 94 constraints. The multiparametric solution is constructed using the POP toolbox with the graph algorithm. The developed multiparametric solution yields 956 critical regions and is presented in Fig. 2.12.

2.5.2.4 Closed Loop Performance

The closed loop response of the ACUREX process with an explicit model predictive controller based on a neural network with ReLU activation function is presented in Fig. 2.13. It is evident

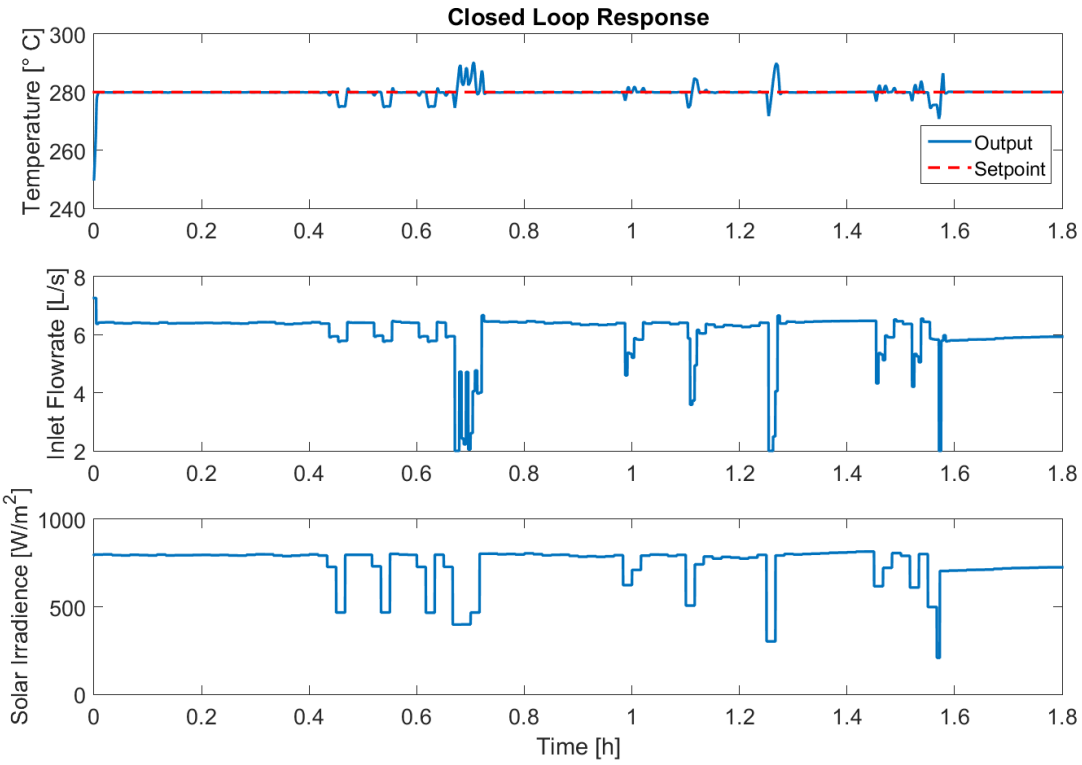


Figure 2.13: The closed loop response of the ACUREX process.

the explicit controller is able to maintain a set point, and recover the setpoint after a significant disturbance is passed to the system [67]. Given the ability of the controller to maintain and manage significant deviations from the setpoint, the closed loop response is considered acceptable.

2.6 Conclusion

In online applications where time is a critical factor, determining the optimal solution with these embedded deep learning models can be a challenge. Multiparametric programming alleviates the computational burden by developing the offline, explicit solution to the parametric optimization problem. With available online solvers, developing the full parametric solution to the mpMILP is manageable and permits its application for the integration of deep learning models and multiparametric programming.

This chapter presented the integration of deep learning models (i.e. feedforward neural networks) and multiparametric programming. Rectified linear units defined the activation function of the neural network due to their piecewise linear nature. Therefore, neural networks with ReLU activation functions are suitable candidates to approximate nonlinear functions because of their predictive capabilities and ability to be embedded as a MILP in a general optimization formulation. The ability to integrate neural networks with ReLU activation functions into parametric optimization was demonstrated by the optimal operation of a reaction separation process and the ACUREX process.

2.6.1 Future Work

Integrating accurate surrogate models in optimization formulations is a critical step in many fields. Because of the importance of surrogate models, there are numerous avenues for future work.

2.6.2 Chance Constrained Programming

In robust optimization, the problem is not to find the optimal solution, but to find an optimal solution under uncertainty. While similar to multiparametric programming, the major difference is what the solution provides. The optimal solution of a robust optimization problem is a single value that satisfies all constraints for any disturbance realization. The optimal solution intrinsically accounts for all parameter realization to ensure constraint satisfaction under all possible uncertainties. In many instances, the solution to a robust optimization problem is overly conservative. One

strategy provides an improved optimal solution from the point of view of the objective function, at the price of having potential constraint violation. This strategy is chance constrained optimization [68].

Assuming the uncertainty follows a discrete probability distribution, Eq. (2.17) is a representative robust optimization formulation.

$$\begin{aligned} \min_{x} \quad & g(x) \\ \text{s.t.} \quad & f(x,\theta) \leq 0 \\ & \theta \in \Theta \end{aligned} \tag{2.17}$$

where θ is unknown and belongs to the discrete probability distribution Θ. An optimal solution to this problem ensures that no constraint violation will occur for any realization of the uncertainty, θ. However, this can be overly conservative, or no feasible solution may exist in the worst case. To alleviate this burden, a relaxed problem is formulated given by Eq. (2.18).

$$\begin{aligned} \min_{x} \quad & g(x) \\ \text{s.t.} \quad & Pr[f(x,\theta) \leq 0] \geq 1-\epsilon \\ & \theta \in \Theta \end{aligned} \tag{2.18}$$

where Pr is the probability of an event, and ϵ is the risk tolerance level, or the percent of failure that is acceptable. This formulation is the basis for chance constrained optimization, where a less conservative solution is determined at the cost of an acceptable risk tolerance. Eq. (2.18) can be reformulated to remove the Pr function using mixed-integer programming techniques, Eq. (2.19).

$$\begin{aligned} \min_{x} \quad & g(x) \\ \text{s.t.} \quad & f(x,\theta) \leq M(1-y_i),\ \forall i \\ & \sum_{k} P_k y_k \geq 1-\epsilon \end{aligned} \tag{2.19}$$

where M is a large scalar quantity, P_k is the probability of the kth event, and y is a binary

variable associated with an event occurring. Determining the optimal solution to Eq. (2.19) is challenging because of the nonlinearity posed by f and g, and the introduction of mixed-integer variables. The nonlinear models can be replaced by neural networks with ReLU activation functions to capture the nonlinearities at the cost of additional mixed-integer variables, but with the benefit of transforming the original problem into an MILP. Thus the incorporation of neural network surrogate models in chance constrained programming for model predictive control applications is proposed as a future direction. In addition, the risk tolerance level can be incorporated as an uncertain parameter when developing the multiparametric solution as a means to alter the risk tolerance in real time without incurring an additional computational penalty.

2.6.3 Other Machine Learning Models

This chapter demonstrated that a machine learning model, amenable to a mixed-integer formulation, is readily implementable in a multiparametric model predictive control formulation. However, in the literature there are numerous machine learning models that are readily recastable as mixed-integer programs similar to neural networks with ReLU activation functions. The following are the current machine learning models under consideration, with the objective of comparing these approaches in a mpMPC context.

1. Recurrent Neural Networks - Similar to the feedforward neural network, except it reuses information from previous time points, the recurrent neural network has the potential to be recast into an MILP formulation. These models are of interest because they utilize previous information to predict into the future, which is a common occurrence in chemical engineering applications.

2. Multivariate Adaptive Regression Splines - MARS models are similar to neural network models and it is also possible to directly incorporate them into MILP formulations [69]. Unlike neural networks, MARS models enables interpretable models, which is important in many industrial settings, and are fast to train without requiring as much data.

3. Decision Trees - These models are capable of many modeling tasks and are easy to train and

interpret. Another benefit is their ability to train on data with minimal preprocessing.

4. Random Forests - A single decision tree provides an interpretable model, but a random forest utilizes an agglomeration of decision trees to make accurate predictions that balances the bias vs. variance trade-off.

3. A NOVEL METRIC FOR MULTIPARAMETRIC MODEL PREDICTIVE CONTROL *

Model Predictive Control is a growing field in the academic and industrial communities for more than three decades now [70, 71, 72]. In recent years, more advanced models are being incorporated into these model predictive control frameworks [73].

While direct formulations preserve the dynamics of the high fidelity model, they may result in an intractable, large scale, non-convex optimization problem depending on the complexity of the original model. Problems of this nature can be reduced to tractable forms via model approximation techniques [74] that are then incorporated into a model predictive control formulation. Multiparametric programming enables improved online computational effort by providing an offline, explicit solution to the MPC optimization formulation [13, 52].

The development of approximate models for model predictive control is pivotal when controlling a complex system. Numerous approximation methods for model predictive control currently exist in the literature, such as system identification [6] and linearization around a steady state. Recently, advanced techniques using sparse regression, local dynamic decomposition strategies, and Proper Orthogonal Decomposition have been introduced into the literature [7, 8, 9, 10]. More advanced techniques to construct approximate models for MPC applications include Hammerstein-Wiener models [75], feedback linearization [76, 77], emperical grammians [78], trajectory linearization [79, 80], piecewise linearization [81], neural networks [23], and closed loop re-identification [82].

In tandem with model approximation development, there has been development in model validation via quantitative and qualitative metrics. Balaguer et al. [83] has utilized frequency analysis to perform model validation, and Alvarado et al. [84] has extended this approach by incorporating (i) Integral Time Absolute Error (ITAE), (ii) frequency analysis, and (iii) closed loop performance.

*Parts of this section are reprinted with permission from "The impact of model approximation in multiparametric model predictive control" by Justin Katz, Baris Burnak, and Efstratios N. Pistikopoulos, 2018. Chemical Engineering Research and Design, Volume 139, Pages 211-223.

The aim of these metrics is to produce quantitative measures to compare developed MPC strategies against each other as a means to rule out certain strategies from implementation. Or in other words, these metrics provide confidence in the approximate models developed to be used in control applications. However, there is a lack of metrics that examine the optimization formulation defining the MPC structure.

To provide analysis of different mpMPC formulations that arise from various surrogate models, this chapter provides a novel metric based on the decision space volume. Also, to further emphasize the critical importance of the developed surrogate model, its direct effect on the multiparametric solution is explored. This chapter is based on published work [1].

3.1 The Influence of Approximate Models in Multiparametric Model Predictive Control

The relationship between the approximate model and a high fidelity model in the context of multiparametric programming is conceptualized as follows. We start with a high fidelity model that is single input with one state, as seen in Eq. (3.1). The high fidelity model is represented by an approximate model of an affine form, as seen by Eq. (3.2).

$$\dot{x} = f(x, u),\ x(0) = x_0 \tag{3.1}$$

where $x \in \mathbb{R}$ is the state of the system, $u \in U$ is the manipulated action of the system, and f represents the state evolution.

$$x_{t+1} = ax_t + bu_t + c,\ x(0) = x_0 \tag{3.2}$$

where x_t is the state of the approximate model at a given time instance, and u is the same as in Eq. (3.1). The a and b coefficients together with c define this affine approximate model. For clarity, in the following steps c is taken to be zero. The initial state of this system x_0, a, and b are treated as bounded uncertain parameters.

To formulate the optimization problem utilizing the approximate model, Eq. (3.3) is developed.

$$
\begin{aligned}
\min_{u_0} \quad & \sum_{i=1}^{P} x_i^T Q x_i \\
\text{s.t.} \quad & x_{t+1} = a x_t + b u_0, \ \forall t \\
& x(0) = x_0 \\
& \underline{x} \leq x_0 \leq \bar{x} \\
& \underline{a} \leq a \leq \bar{a} \\
& \underline{b} \leq b \leq \bar{b} \\
& x_t \in X, \ \forall t \\
& u_0 \in U \\
& t \in [0, 1, ..., P]
\end{aligned}
\tag{3.3}
$$

where upper and lower bars indicate upper and lower bounds. P is the horizon length, t is the discrete time point, Q is a penalty matrix, and the states and input of the system belong to the set X and U respectively.

Eq. (3.3) is transformed into a multiparametric programming problem as seen in Eq. (3.4). The solution to this multiparametric programming problem returns expressions of u as functions of the uncertain parameters, x_0, a, and b. Note, the treatment of a and b as uncertain parameters yields left hand side uncertainty manifesting in the $A(\theta)$ term in Eq. (3.4). In the general case, the solution to this problem class is still an open question and standard multiparametric techniques are not readily available to solve these problem structures [85, 86]. In the proposed multiparametric problem, Eq. (3.4), the small scale and low complexity allows us to determine the exact multiparametric solution.

$$
\begin{aligned}
u(\theta) = \underset{u}{\text{argmin}} \quad & (Q_{mp}u + H\theta)^T u \\
\text{s.t.} \quad & A(\theta)u \leq b + F\theta \\
& \theta = [x_0 \; a \; b]^T \in \Theta
\end{aligned}
\tag{3.4}
$$

where the representative matrices Q_{mp}, H, A, b, and F are of appropriate size and are developed by expanding Eq. (3.2) to the horizon length P and substituting recursively. The bounded uncertain parameters, θ, belong to the set Θ. The solution to this multiparametric programming problem is represented by a collection of critical regions, given in Eq. (3.5). Inside each critical region belongs a functional representation of the optimization variable as a function of the uncertain parameters. In this case, the uncertain parameters are the initial condition of the system, x_0, and the coefficients of the approximate model developed, a and b.

The optimal action associated with each critical region is defined in Eq. (3.5), where Critical Region 1 (CR_1), Critical Region 2 (CR_2), and Critical Region 3 (CR_3) are defined in Appendix A.1.

$$
u = \begin{cases} -\frac{a^2(ab+b)+ab}{(ab+b)^2+b^2}x_0, & \text{if} \quad \theta \in CR_1 \\ U_{max}, & \text{if} \quad \theta \in CR_2 \\ U_{min}, & \text{if} \quad \theta \in CR_3 \end{cases}
\tag{3.5}
$$

The graphical representation of the critical regions for this motivating example is seen in Figure 3.1. In the figure, each colored region represents a particular Critical Region with an associated functional relationship between the optimal action, u, and the uncertain coefficients. For instance, the optimal action in Eq. (3.5) defined by Critical Region 1 is a nonlinear function of the uncertain parameters a and b. To understand this nonlinear effect, consider two instances of a and b realizations in Critical Region 1, namely, (i) $a = 0.5$ and $b = -0.5$, and (ii) $a = 0.2$ and $b = -0.1$. This realization yields to distinct optimal control laws that are affine function of the initial state, as seen by Eq. (3.6).

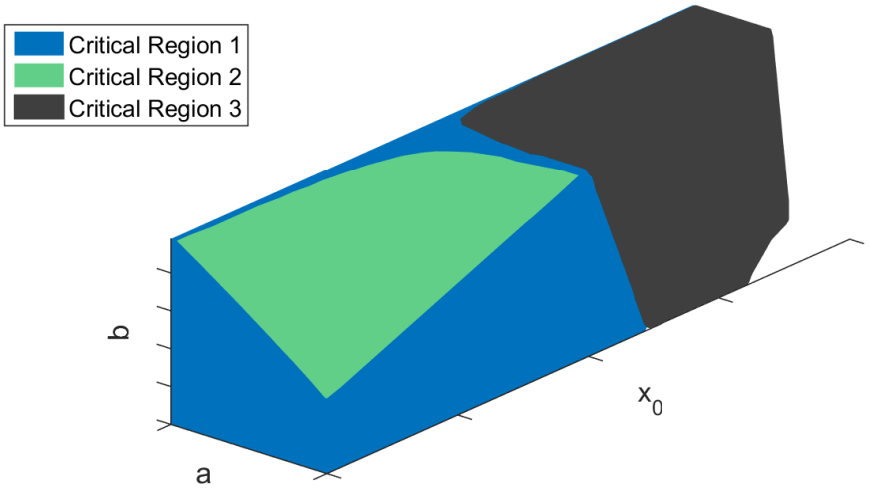


Figure 3.1: The map of solutions for the conceptual example. Reprinted with permission from [1].

$$u(x_0, a = 0.5, b = -0.5) = 0.54x_0 \tag{3.6a}$$

$$u(x_0, a = 0.2, b = -0.1) = 1.02x_0 \tag{3.6b}$$

It is evident from Eq. (3.6) that for the same realization of x_0 in Critical Region 1, the optimal control action would be different.

The benefit of the parametric solution developed is the resulting closed form solution, e.g. Eq. (3.5), which can be integrated into the high fidelity model, e.g. Eq. (3.1). The possibility for the explicit parametric solution to be embedded in the high fidelity model is a key characteristic of mpMPC, and the affine form of the approximate model used is a necessary component in deriving the multiparametric solution.

In this conceptual example, the approximate model was left uncertain, yielding a multiparametric programming problem with left hand side uncertainty. Currently, no general solution strategy exists for solving multiparametric programming problems of this type. Therefore, the approximate model used for the mpMPC must be fixed prior to developing the multiparametric solution.

Fixing the surrogate model is equivalent to taking a slice of nonlinear multiparametric solution, demonstrated by Eq. (3.6). Different surrogate model realizations amount to changing the location of the slice performed. Therefore, a natural comparison is between the size of the slices taken from the different developed surrogate model realizations. One technique to perform this comparison is through a volume calculation and is the basis for the metric used in this chapter.

3.2 Background and Methodology

A brief review of the major topics are presented. In particular, discussions of (i) model predictive control, (ii) system identification, and (iii) the novel metric of decision space volume.

This section presents model approximation techniques in conjunction with metrics to assess multiparametric model predictive controllers.

3.2.1 Model Based Control

Consider a control policy based on the continuous time high fidelity model, given in Eq. (3.7). Another control policy, based on a discrete time formulation, customarily described as the conventional model predictive control formulation[71], is given by Eq. (3.8).

$$\begin{aligned} \min_{u} \quad & \int_0^{\tau} \left((y - y_{sp})^T QR (y - y_{sp}) + u^T R u \right) dt \\ \text{s.t.} \quad & \dot{x} = f(x, u),\ x_0 = x(0) \\ & y = h(x, u) \\ & x \in X \\ & y \in Y \\ & u \in U \end{aligned} \tag{3.7}$$

where QR and R are cost matrices, y_{sp}, y, and x are the set point, output, and states of the high fidelity system respectively. f is a function defining the evolution of the states over time, h defines the relationship between the states of the system and inputs, and u is the input to the high fidelity system. The states, outputs, and inputs belong to the set X, Y, and U respectively. Note, a terminal cost on the final states can be incorporated in the objective function but is not included in

this formulation for brevity.

$$\begin{aligned}
\min_{u_0,\ldots,u_{NC}} \quad & \sum_{i=1}^{OH}(y_i - y_{sp})^T QR(y_i - y_{sp}) + \sum_{i=0}^{NC} u_i^T R u_i \\
\text{s.t.} \quad & x_{t+1} = Ax_t + Bu_t,\ \forall t \\
& y_t = Cx_t + Du_t,\ \forall t > 0 \\
& u_t = u_{NC},\ \forall t > NC \\
& x_0 = x(0) \\
& y_t \in Y,\ \forall t > 0 \\
& u_t \in U,\ \forall t \\
& t \in [0, 1, ..., OH]
\end{aligned} \tag{3.8}$$

where A, B, C, and D are the associated state space matrices. The states of the approximate system are x, the estimated outputs are y, OH is the output horizon of the system and NC is the control horizon of the system where $NC \leq OH$. The outputs of the system, y, belong to the set Y and are predictions of the high fidelity output. All other terms are the same as in Eq. (3.7).

Due to the approximation of the high fidelity model, the output of the approximate model in Eq. (3.8) is not guaranteed to follow the real output of the system in Eq. (3.7). To account for this mismatch, a mismatch term is added to the MPC formulation in the form of Eq. (3.9) [87].

$$e = y_{real} - y_0 \tag{3.9}$$

where y_{real} is the real system output, y_0 is the approximate output, and e is the associated error.

Eq. (3.8) is transformed to a multiparametric MPC which can be solved explicitly offline to determine the optimal control action as an affine function of the uncertain parameters, namely the initial state of the system and the set point [13, 88, 89, 90].

3.2.2 Model Approximation

In this work, the development of an affine approximate model used in the multiparametric formulation is mandatory. To this end, the model approximation technique used is system identification. This technique is a data-driven strategy and requires input/output measurements of the process. The model is developed using the system identification toolbox from MATLAB. Other model approximation techniques that are readily implementable can be found in Katz *et al.* [1].

To generate an input signal, a Pseudo Random Binary Sequence (PRBS) is used. The choice of the input signal used is to ensure proper excitation of the underlying high fidelity system. Following generation of the PRBS input signal, the output data is generated by passing the input signal to the high fidelity model. The developed approximate models from the input/output data are used to approximate the f and h functions in Eq. (3.7). A brief review of system identification is discussed in the following subsection.

3.2.2.1 System Identification

System identification is a technique based on input/output data which results in a discrete time state space model [6]. The form of the equation is presented in Eq. (3.10).

$$x_{t+1} = Ax_t + Bu_t \tag{3.10a}$$

$$y_t = Cx_t + Du_t \tag{3.10b}$$

where the matrices A, B, C, and D are the state space matrices that define the evolution of the states. The states of the approximate model are x, u is the manipulated input to the system, and y is the predicted output of the system. It is important to note that the states developed from system identification are pseudostates and do not hold physical meaning.

3.2.3 Novel Metric

Established approaches to model validation assess the approximate model in the context of open loop or closed loop performance. In this section, a novel metric focusing on the optimization formulation developed via the approximate model by analyzing the decision space volume is presented.

3.2.4 Decision Space Volume

The proposed metric calculates the feasible space volume projected on its decision space. The decision space volume is calculated for each mpMPC developed from the various approximate models as a quantitative test for the degree of freedom for each controller. Smaller decision space volumes indicate an mpMPC that is more restricted in the selection of the optimal control action. The procedure for calculating the decision space volume is described as follows. A representative feasible space for an mpMPC is defined by Eq. (3.11).

$$Au \leq b + F\theta \tag{3.11}$$

In Eq. (3.11), u is the control action, θ is the uncertain parameter, and A, b, and F are the associated matrices of the feasible space. The vertices of the polytope associated with Eq. (3.11) are determined to calculate the decision space volume. The dimensionality of the decision space volume depends on the control horizon and number of manipulated actions because they directly affect the number of optimization (decision) variables. This is apparent from the length of the vector u, which is equal to the number of manipulated actions multiplied by the control horizon. Thus an increase in either the manipulated actions or the control horizon increases the dimensionality of the decision space volume.

The procedure for the decision space volume calculation is conceptualized as follows. Consider a feasible region defined by Eq. (3.12).

Table 3.1: The vertices defining the polytope in Eq. (3.12). Reprinted with permission from [1].

Vertex ID	u	θ
1	0.1	0.2
2	0.9	0.2
3	0.5	0.8

$$0 \leq u \leq 1 \tag{3.12a}$$

$$0 \leq \theta \leq 1 \tag{3.12b}$$

$$0.6\theta - 0.8u \leq 0.03 \tag{3.12c}$$

$$-\theta \leq -0.2 \tag{3.12d}$$

$$0.8u + 0.6\theta \leq 0.9 \tag{3.12e}$$

The vertices defining the polytope in Eq. (3.12) are shown in Table 3.1. The bounds of u are bounded between 0 and 1, however, from Table 3.1 it is seen that the maximum and minimum attainable u values are 0.9 and 0.1, respectively. Therefore, the decision space volume is the difference between these maximum and minimum attainable bounds for u and is 0.8.

3.3 Example

The example chemical process is adapted from Tippett *et al.* [4] and is similar to the example presented in Section 2.5.1. It includes 1 nonisothermal continuously stirred tank reactors (CSTR), 1 flash separator, and 1 recycle stream. The process maintains an exothermic, irreversible series reaction following $A \rightarrow B \rightarrow C$, where B is the product of interest. The reactions follow an Arrhenius rate law expression. It is also assumed the reactors have constant molar hold up and the species have constant relative volatility. The mathematical model consists of 7 differential equations defining the mass and energy balances. In Appendix A.2, the equations in the section Reactor 1 and Separator are used to define the high fidelity model.

Manipulated variables of the process are the flowrates of material into the cooling jacket and reactor, the heat duty into the separator, and the recycle stream. The outputs of interest for the process is the temperature of the reactor and separator, and the mole fraction of product B in the separator. The control objective is to meet safety considerations while tracking a predetermined setpoint for the mole fraction.

3.3.1 Model Predictive Control Formulation

The control formulation is based on tracking the defined setpoint of 0.6 while ensuring the operation of both the reactor and separator remain within the specified temperature bounds of $370K$ and $480K$.

$$\begin{aligned} \min_{u} \quad & \sum_{i=1}^{5} (y_i - y_{sp})^T QR (y_i - y_{sp}) + u^T R u \\ \text{s.t.} \quad & x_{t+1} = Ax_t + Bu,\ x_0 = x(0) \\ & y_t = Cx_t + e \\ & e = y - y_{meas} \\ & y_t \in Y \\ & u \in U \end{aligned} \tag{3.13}$$

where the matrices A, B, and C are determined during the development of the surrogate model. The uncertain parameter e enables the MPC formulation to incorporate plant-model mismatch. The weight matrices QR and R are defined to be $1e3$ and a $4x4$ diagonal matrix of entries $1e-3$ respectively. The output horizon was selected to be 5 and the control horizon was selected to be 1. These tuning parameters were chosen because of their suitable closed loop performance. The defined mpMPC is solved once and offline using the POP toolbox to develop the multiparametric solution.

For this example, two different control formulations are developed based on two different surrogate models. The first surrogate model, Model 1, is based on a system identified model with 2 pseudostates. The second surrogate model, Model 2, is based on a system identified model with 4

pseudostates. The resulting multiparametric solution to the mpMPC formulation returns a solution with 287 and 2257 critical regions respectively.

3.3.2 Closed Loop Performance

The closed loop simulation is provided by Figs. 3.3 and 3.4, and relevant metrics are provided in Table 3.2. The figures are based on the closed loop performance of an mpMPC strategy with a system identified model of 2 and 4 states respectively. Each strategy is able to track the setpoint effectively without constraint violation, or excessive over/under shoots. However, the mpMPC strategy that is based on a surrogate model with 2 pseudostates is 0.1% away from the setpoint at the end of the 3 hour operation, while the mpMPC strategy with 4 pseudostates is 1.8% away from the setpoint at the end of the operation. From this point of view the system identified model with 2 pseudostates would be preferred.

The metric presented in this chapter, the decision space volume, returns a value of 1.32 for the mpMPC strategy based on Model 1, and 5.12 for the mpMPC strategy based on Model 2. Because the process has 4 manipulated variables, and the mpMPC strategy has a control horizon of 1, the largest the decision space volume could be is 16. Therefore, Model 1 occupies approximately 8% of the decision space volume while Model 2 occupies approximately 32%. Coupled with the number of critical regions developed, the number of critical regions per unit volume (i.e. density) for mpMPC scheme based on Model 1 and Model 2 is approximately $217\frac{CR}{Vol}$ and $441\frac{CR}{Vol}$ respectively. The larger density implies there are significantly many more possible control laws that can be utilized during the online implementation as seen by the more diverse input profile in Fig 3.4 compared to Fig 3.3. At a cursory glance, this could be expected to provide an improved operating performance. However, with the potential for more possible manipulated actions, it is possible some actions are unnecessary and may even contribute to a poorer operation, demonstrated by the difference in output profiles during closed loop operation.

Table 3.2: The performance metrics for the example problem.

Surrogate Model	Setpoint Deviation	Decision Space Volume
Model 1	0.1%	1.32
Model 2	1.8%	5.12

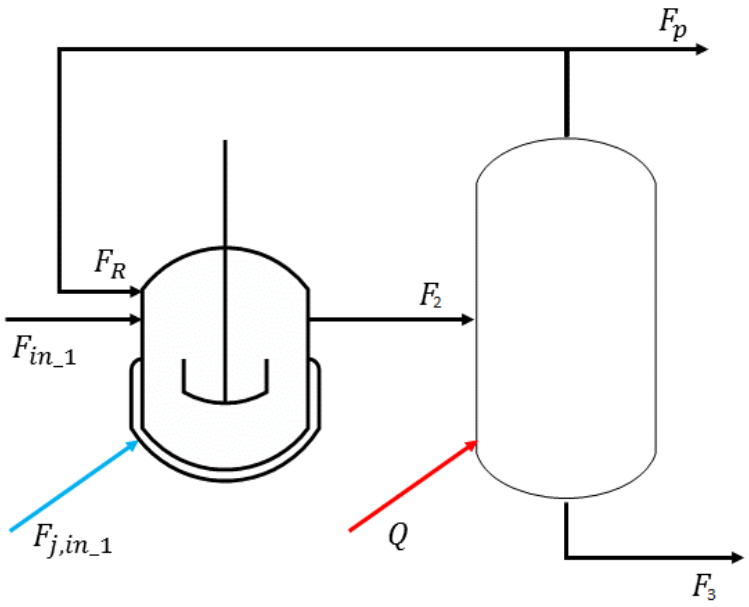


Figure 3.2: A schematic representation of the example problem.

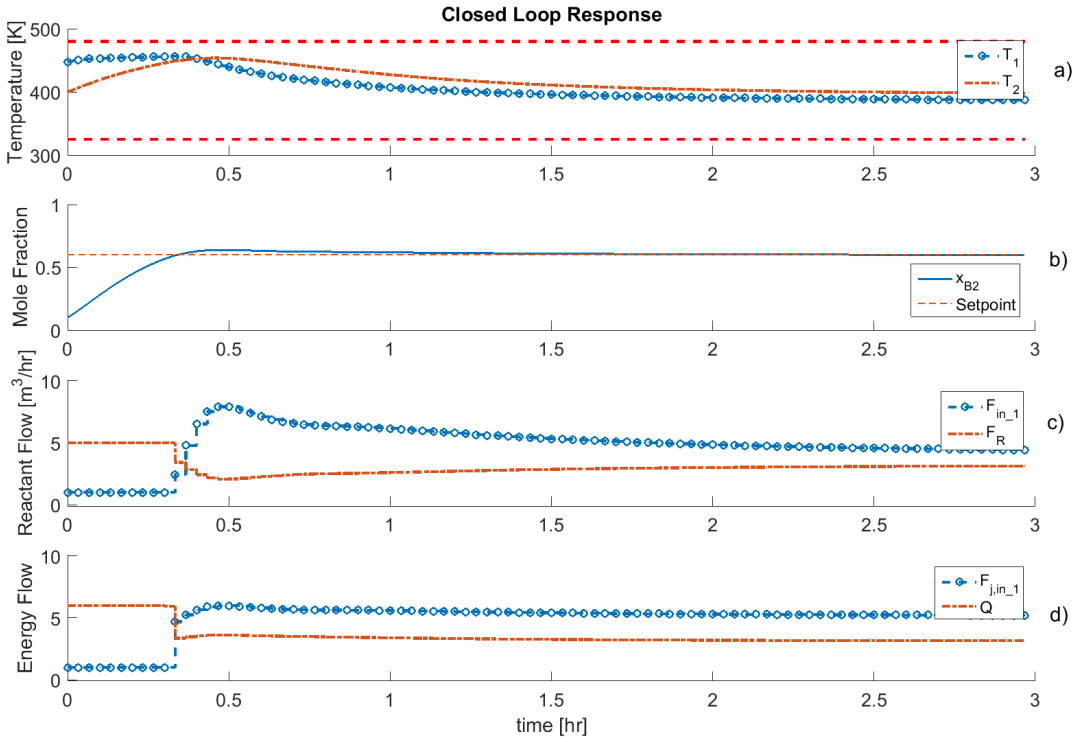


Figure 3.3: The closed loop simulation with surrogate model 1: a) The temperature profile for the reactor and separator. b) The mole fraction of material B. c) The flowrates of material into the process. d) The cooling flowrate and energy into the process.

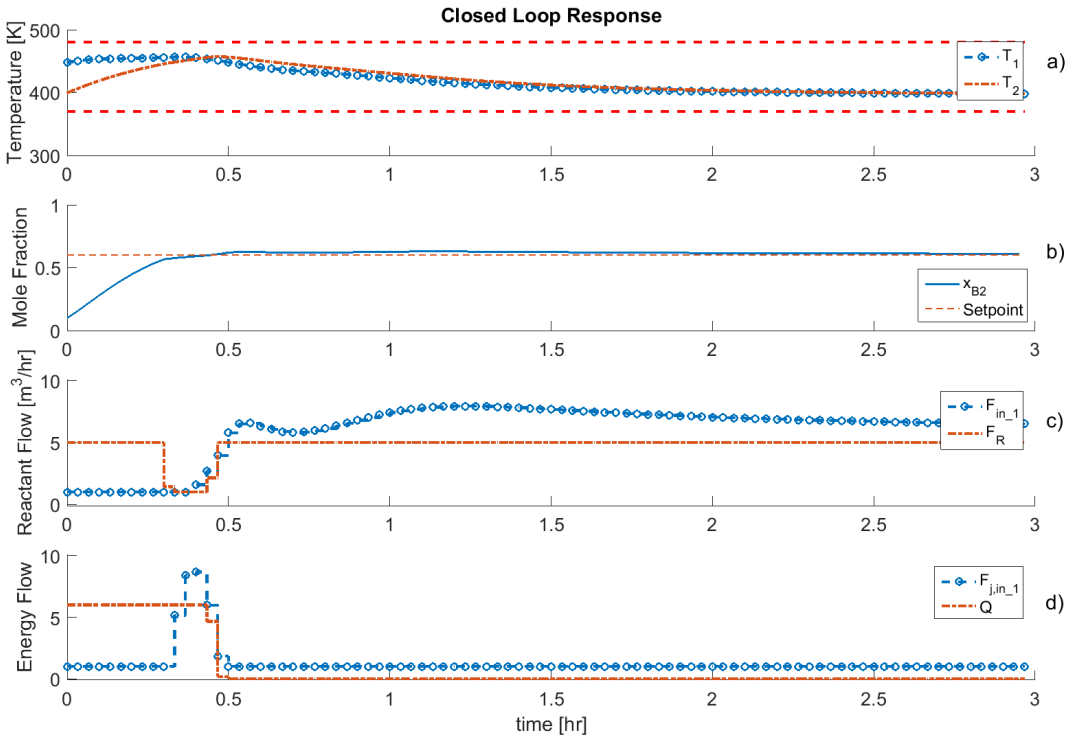


Figure 3.4: The closed loop simulation with surrogate model 2: a) The temperature profile for the reactor and separator. b) The mole fraction of material B . c) The flowrates of material into the process. d) The cooling flowrate and energy into the process.

3.4 Conclusion

A key component in a multiparametric model predictive control formulation is the underlying surrogate model. It was shown that the surrogate model has a nonlinear effect on the developed parametric solution. This implies that a particular surrogate model with slightly different parameters that define its mathematical representation can return significantly different profiles. Therefore, it is critical to develop a surrogate model that has the capacity to represent the process effectively during closed loop operation. However, because closed loop operation is based on the developed multiparametric solution, it is important to differentiate between multiparametric solutions that are based on different surrogate models. To compare the effect of different surrogate models, a decision space volume calculation is performed. This calculation provides a measure for how the surrogate model has impacted the size of the optimization variable space. In addition, coupled with the number of critical regions determined, an analysis can be performed that aims to compare the developed explicit control law (i.e. the multiparametric solution) prior to any closed loop operation.

3.4.1 Future Work

In this chapter, it was shown that different surrogate models produce different decision space volumes under the same mpMPC formulation. Understanding different features of the decision space of two different mpMPC formulations is a critical factor. For instance, comparing the location of the decision space with different mpMPC solutions can provide further insight. By analysing the location of the decision space volume, it can be ascertained where the optimal control actions will be performed, or even if one decision space of a solution based on one surrogate model lies entirely in another decision space of a different surrogate model, as demonstrated by Fig. 3.5.

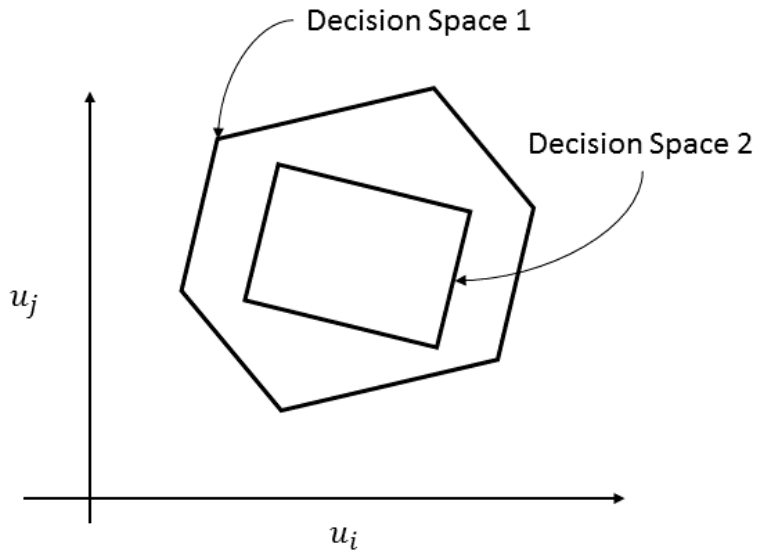


Figure 3.5: A comparison of the location of two different decision spaces.

4. A NOVEL GEOMETRIC ALGORITHM FOR MULTIPARAMETRIC PROGRAMMING

The solution strategy of multiparametric programming problems can be observed as early as 1972 with the seminal work by Nedoma et al. [91]. Complete theories and solution strategies were proposed in the literature for multiparametric linear programming (mpLP), quadratic programming (mpQP) [91, 13, 92, 59, 15, 2, 16], and mixed-integer linear programming (mpMILP) [93]. A key difference in existing approaches is their procedure to explore the parametric space to completion. Algorithms proposed by Bemporad *et al.* (2002) [13], Tøndel *et al.* (2003) [92], and Spjøtvold *et al.* (2006) [59] rely on geometrical strategies, where the parametric solution is determined by direct exploration of the parameter space. Strategies for multiparametric programming proposed by Gupta *et al.* (2011) [15] and Ahmadi-Moshkenani *et al.* (2018) [2] develop the parametric solution by enumerating possible active set combinations with a branch and bound style approach. These active set strategies are inherently different from geometrical approaches because they do not rely on the parametric space to identify the optimal explicit expressions that partition the parameter space. Algorithms that incorporate both geometric and active set strategies, by Gal *et al.* (1972) and Oberdieck *et al.* (2017) [91, 16], rely on representing the parametric solution as a connected graph where each node represents an optimal partition that is defined over the parametric space.

Although these approaches theoretically guarantee developing the complete solution over the parameter space, practical implementation becomes more challenging as the number of optimization variables, constraints, and parameters grow because of the exponential increase in optimal partitions. Managing the memory requirements of an exponential solution space has been approached by Drgoňa *et al.* via the so-called regionless explicit model predictive control [21]. The regionless explicit model predictive control strategy saves memory by maintaining factored matrices and active set combinations, instead of the optimal partitions that define the parameter space. However, with a solution space that grows combinatorially with the problem size, developing the full parametric solution becomes impractical, and using the complete explicit solution in offline applications becomes intractable. For instance, in multi-level optimization formulations, the solu-

tion space of the follower (lower level) problems increase rapidly in the number of variables and constraints, necessitating a strategy to account for the potential explosion of optimal partitions that define the multiparametric solution. Current theory and strategies in the literature do not attempt to address the potential explosion in the number of optimal partitions, hence the use of explicit solutions in large scale offline applications is rather limited. Therefore, the exploration of a meaningful partial solution to these large scale problems is necessary. In other words, the question that must be addressed is "What is a good criterion that provides meaningful insight to the multiparametric solution, and how can an efficient strategy be implemented to exploit this criterion?".

In this chapter, a novel geometric algorithm is presented for mpLPs and mpQPs to return a partial solution to large scale multiparametric programming problems of these types. Practical implementation of a partial solution can be found in offline applications such as multi-level optimization formulations where determining a feasible solution can be challenging, and in online applications as a method to warm-start a solution procedure. The solution is analyzed with a new perspective, namely the volume of the developed multiparametric solution. This chapter is based on submitted work [94].

4.1 Methodology of Proposed Algorithm

The proposed algorithm makes use of two key properties of the mpLP and mpQP solution, namely a critical region is (i) uniquely identified by an active set combination, and (ii) is a convex polytope defined by the intersection of a finite number of halfspaces [13]. To exploit these properties, the algorithm postulates j vertices of a convex shape in the parameter space. Each vertex defining the convex shape in the parameter space is defined by a fixed value of the uncertain parameter vector θ_j^*, and corresponds to an active set combination. If all θ_j^* share the same active set combination then the postulated vertices must exist within a critical region (a direct consequence of critical regions being convex), otherwise there exists multiple critical regions within the postulated convex shape. If multiple critical regions exist, then the convex shape is decomposed into further convex shapes, termed child convex shapes. The same procedure is then performed on the children convex shapes in a recursive manner, presented in Fig. 4.1.

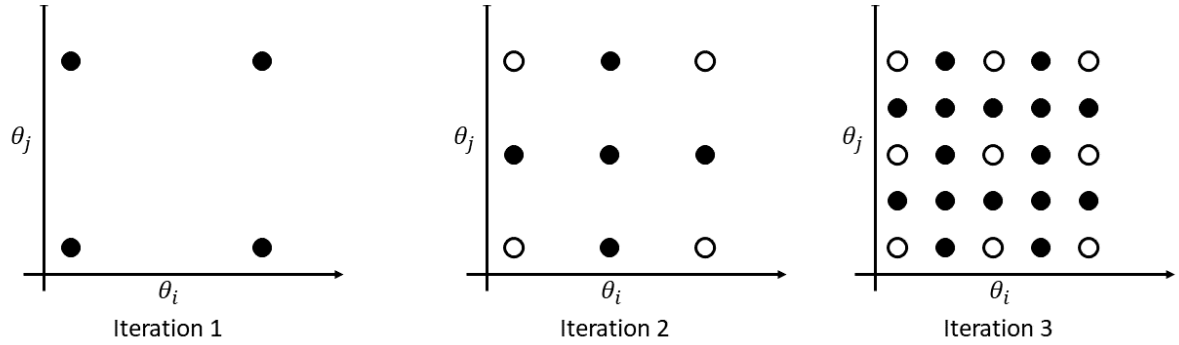


Figure 4.1: Starting with the bounds of the parameter space defined by bounding constraints, and assuming the convex shape is a hyperbox, iteration 1 identifies all active sets of the vertices. If the active sets are non unique, further exploration is performed in a recursive manner, demonstrated by iteration 2 and 3. The filled vertices are the vertices with unknown active set combinations.

Fig. 4.2 is a schematic representing a postulated convex space and the active sets of the associated vertices. The postulated convex space provides a means of where further exploration is needed if it cannot be represented by a unique active set combination. The pseudocode for the proposed algorithm is given in Algorithm 1.

The function named DIVIDECONVEXSPACE in Algorithm 1 is critical because it provides the convex shape used to explore the parameter space. The number of vertices of the shape used to divide the parameter space yields the number of deterministic optimization problems solved for the postulated convex space.

4.1.1 Polytope Decomposition

To minimize the number of deterministic optimization problems during algorithmic implementation, the best convex shape to use is a simplex. For a problem with dimensionality n, a hyperbox requires 2^n vertices (Fig. 4.1), while a simplex requires $n + 1$ vertices. By minimizing the number of deterministic optimization problems solved, the total runtime of the algorithm is reduced. Further details regarding the polytope decomposition using simplices is provided in the full length article [94].

Algorithm 1 Proposed Algorithm

Require: t_f ▷ Time to run algorithm
Require: $\Theta := \{\theta | CR_A\theta \leq CR_b\}$ ▷ Initialize parameter space
$P_i = \mathcal{V}(\Theta)$ ▷ Store vertex representation
$AS \leftarrow \varnothing$
$i \leftarrow 1$ ▷ Initialize halfspace count
$t \leftarrow 0$ ▷ Start clock
while $t < t_f$ **do**
    $i' \leftarrow i$
    $i \leftarrow 0$
    $AS_{temp} \leftarrow \varnothing$
    $P' \leftarrow \varnothing$
    for $j = 1 \rightarrow i'$ **do** ▷ For all halfspaces
        for $k = 1 \rightarrow \text{CARD}(P)$ **do** ▷ For all vertices
            $AS_{aux} = \text{SOLVE}(P_j(k))$ ▷ Get active set from deterministic optimization
            $AS_{temp} = [AS_{temp}, AS_{aux}]$
        end for
        $AS_{temp} = \text{UNIQUE}(AS_{temp})$ ▷ Remove redundant active sets
        $AS = [AS, AS_{temp}]$ ▷ Store active sets in global list
        if $\text{CARD}(AS_{temp}) > 1$ **then** ▷ If more than one active set
            $P_{aux} = \text{DIVIDECONVEXSPACE}(P_j)$ ▷ Strategy to divide convex space
            $P' = \{P', P_{aux}\}$
            $i = i + \text{CARD}(P_{aux})$ ▷ $\text{CARD}(P_{aux})$ convex spaces added
        end if
    end for
    $P \leftarrow \varnothing$ ▷ Reset P
    $P = P'$
    $t = t + \Delta t$ ▷ increment time
end while

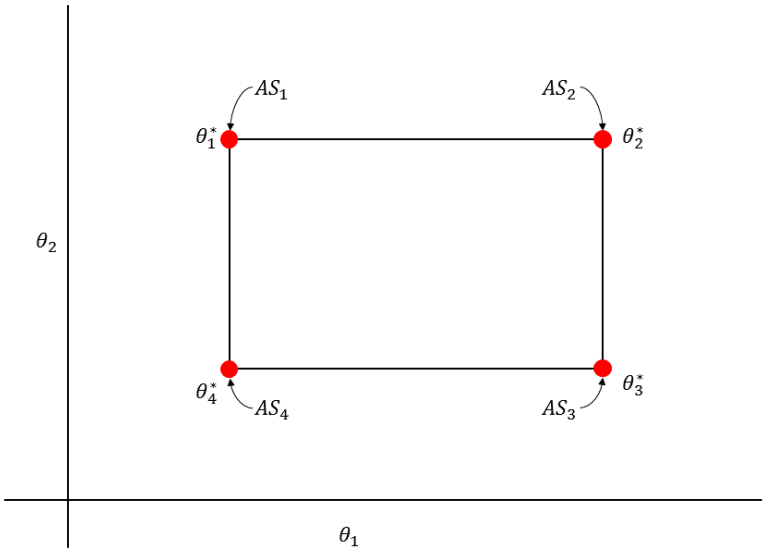


Figure 4.2: A conceptual representation of the main idea behind the proposed algorithm.

4.1.2 Methodology for Analyzing the Solution

The standard strategy to analyze the multiparametric solution is to identify the number of critical regions explored. The approach being that the more critical regions explored, the better the performance of the proposed algorithm. However, this can be a misleading result because the number of critical regions provides no 'physical' meaning with regards to the parametric solution developed. To this end, a volume comparison procedure is developed to analyze the solution of the proposed algorithm with respect to the full solution, and against existing algorithms.

The multiparametric solution can be defined by the sum of volume of all of the critical regions. To calculate the volume, tools from computational geometry can be utilized, such as converting from halfspace representation of a polytope to vertex representation followed by triangulation techniques. Each critical region corresponds to a portion of the total volume, and the developed solution from the algorithm can be represented as the sum of critical region volume determined.

4.1.3 Limitations

While the proposed algorithm is effective in solving multiparametric problems with a large number of optimization variables and constraints, it suffers handling a large number of parameters. Because the parameter space of most multiparametric problems is defined by box constraints, identifying the active set of each vertex scales unfavorably (i.e. 2^n where n is the number of parameters), particularly in the first iteration.

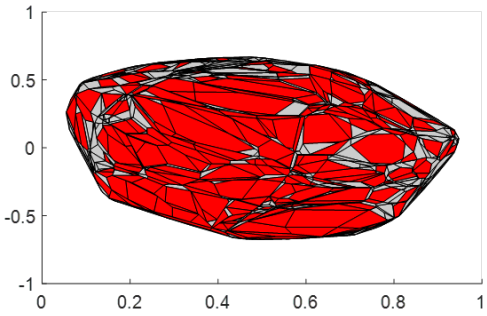
4.2 Computational Results

The proposed algorithm is compared against the solvers offered from the POP toolbox, namely the geometric and graph algorithms, and the solver offered from the Multiparametric Toolbox (MPT) [53]. In these examples, the proposed approach uses simplices as the convex shape. In other words, the function DIVIDECONVEXSPACE is based on a triangulation scheme to generate equally sized simplices.

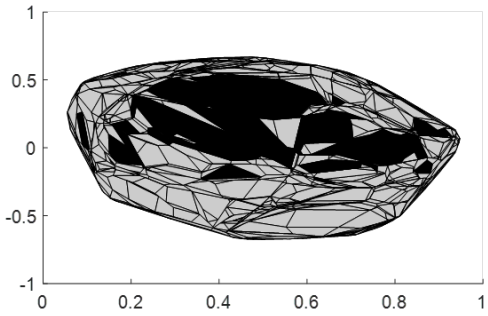
To motivate how each algorithm explores a given multiparametric problem a randomly generated problem is utilized, named Ex 1, Ex 2, Ex 3, and Ex 4. Each algorithm is demonstrated in Figs. 4.3-4.5. The total map of solutions is provided in gray, and the critical regions identified by each algorithm is filled in with a representative color. The map of solutions provided by each algorithm is provided at various breakpoints in time.

To showcase the performance of the proposed algorithm against standard strategies, several randomly generated computational examples of varying complexity are presented. Comparisons are made between each algorithm with respect to the number of critical regions explored and the total volume occupied by the explored critical regions at the end of the allotted time. Table 4.1 provides the relevant details regarding the problem type and size with respect to the number of optimization variables, constraints, and uncertain parameters. Figs. 4.6-4.9 provided details regarding the total volume fraction occupied by each algorithm and the number of critical regions identified.

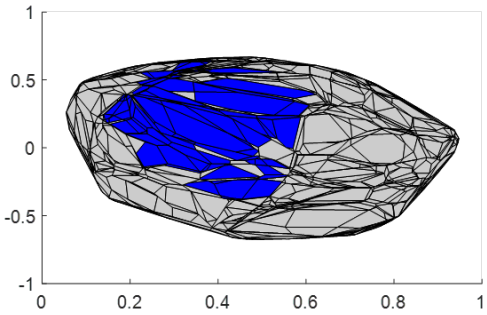
From Figs. 4.6-4.8, it is clear that the proposed algorithm provides an advantage for identifying



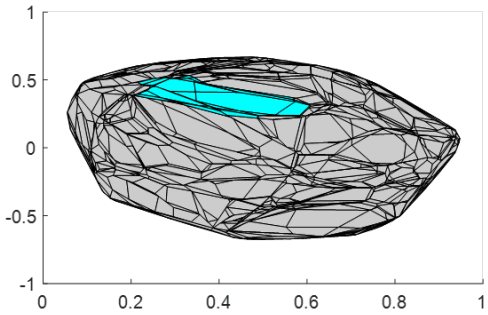
(a) Proposed algorithm



(b) POP - Geometrical

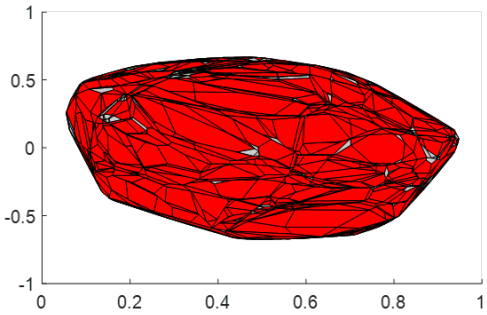


(c) POP - Graph

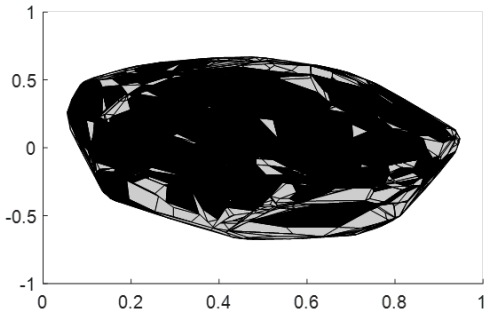


(d) MPT

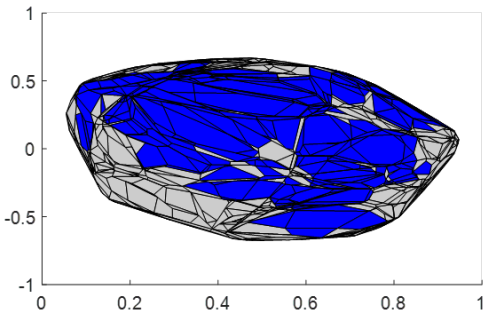
Figure 4.3: Map of solution for each algorithm after 10 seconds.



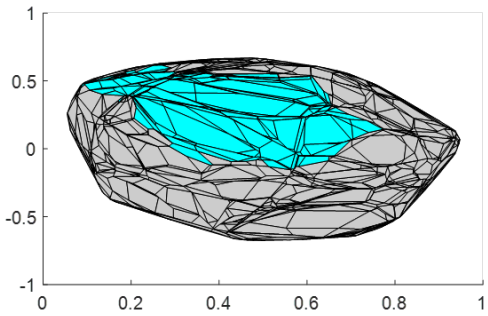
(a) Proposed algorithm



(b) POP - Geometrical

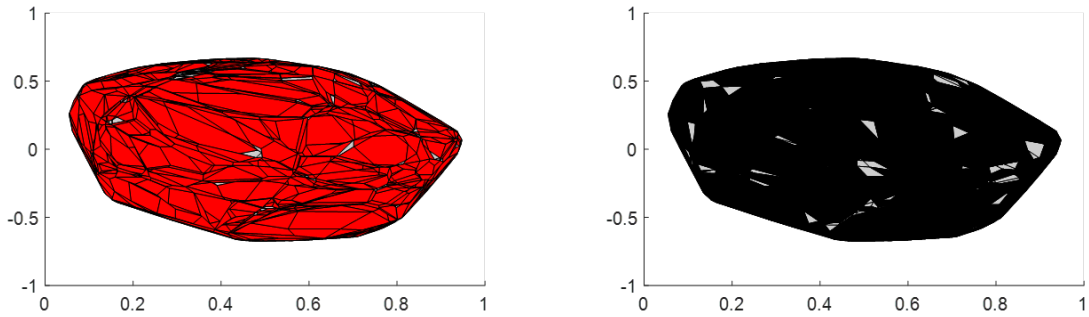


(c) POP - Graph



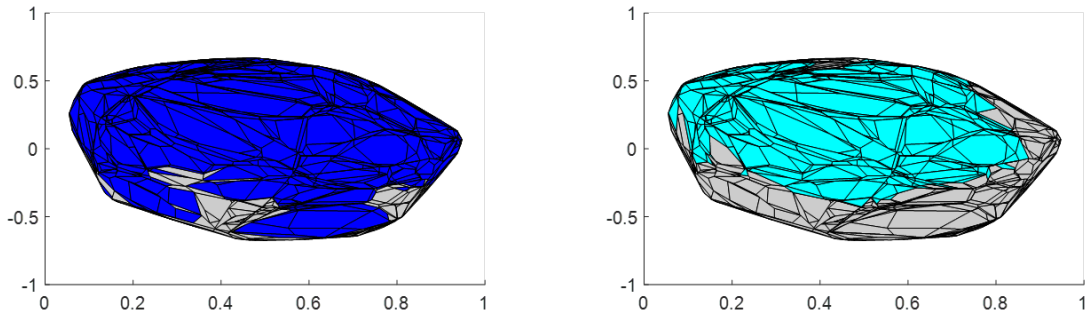
(d) MPT

Figure 4.4: Map of solution for each algorithm after 60 seconds.



(a) Proposed algorithm

(b) POP - Geometrical



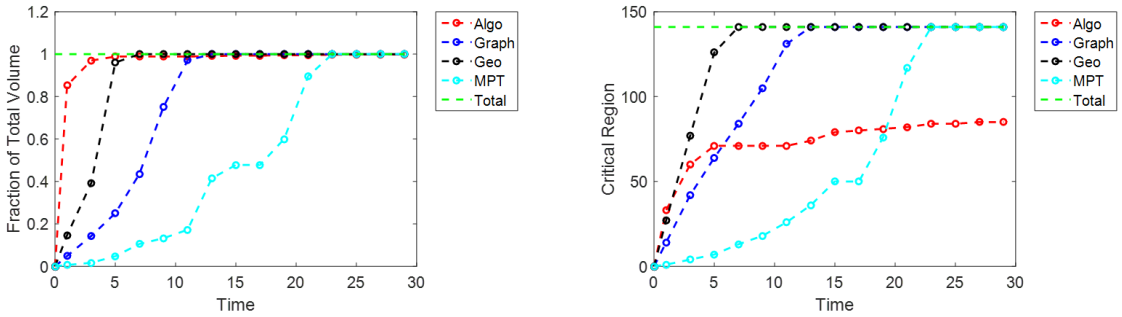
(c) POP - Graph

(d) MPT

Figure 4.5: Map of solution for each algorithm after 180 seconds.

Table 4.1: Details regarding generated example problems.

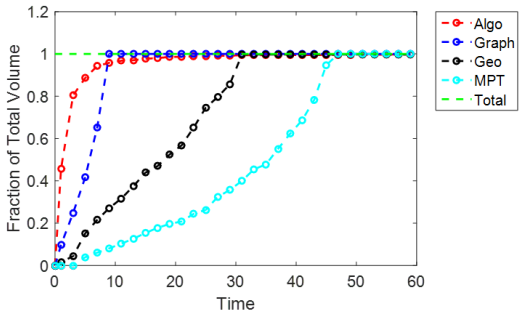
	Type	Constraints	Optimization Variables	Uncertain Parameters
Ex 1	mpLP	40	10	5
Ex 2	mpQP	70	15	2
Ex 3	mpQP	100	20	2
Ex 4	mpQP	800	50	2



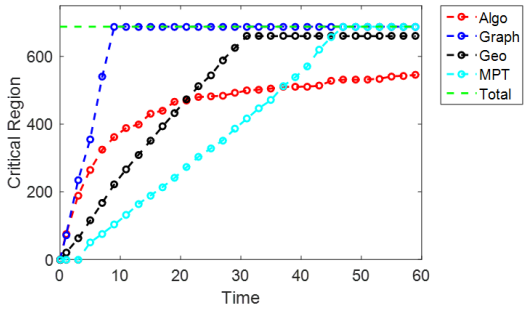
(a) Fraction of total volume determined.

(b) Number of critical regions explored.

Figure 4.6: The fraction of volume explored and the number of critical regions identified for Ex 1.

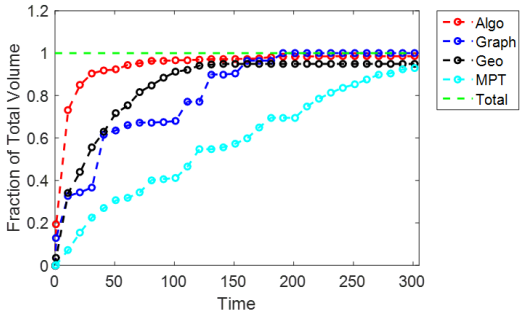


(a) Fraction of total volume determined.

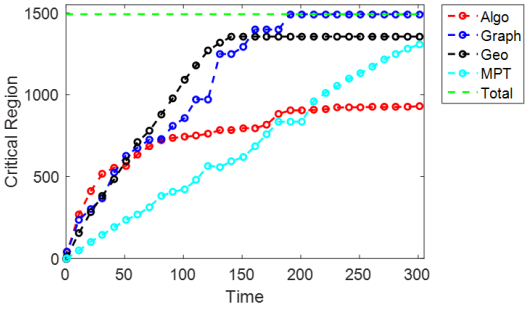


(b) Number of critical regions explored.

Figure 4.7: The fraction of volume explored and the number of critical regions identified for Ex 2.

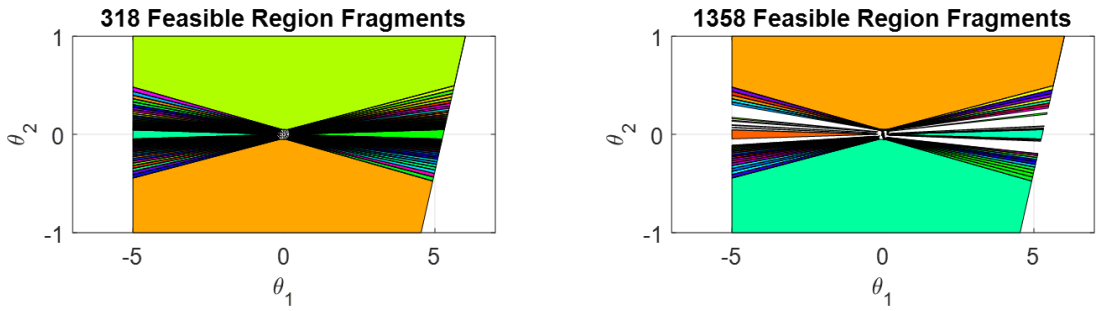


(a) Fraction of total volume determined.



(b) Number of critical regions explored.

Figure 4.8: The fraction of volume explored and the number of critical regions identified for Ex 3.



(a) Proposed algorithm

(b) POP - Geometrical

(c) POP - Graph

(d) MPT

Figure 4.9: Map of solutions after solving for 5 minutes.

the larger critical regions first. This is clear from the figures where in every generated problem, the proposed algorithm can capture the larger volumetric space associated with the explored critical regions compared to the alternative algorithms.

Fig. 4.9 showcases the performance of each of the algorithms after 5 minutes. It is clear that while POP-Graph, POP-Geometrical, and MPT identify significantly more critical regions, the presented algorithm identifies the volumetrically large critical regions. This example illustrates the importance of a method to analyze the partial multiparametric solution that does not rely on the number of critical regions explored, but a meaningful quantity such as volume of critical regions explored.

4.3 Discussion

The proposed geometric algorithm was demonstrated on example multiparametric programming problems of considerable size and complexity. The range of total number of critical regions

for these problems ranged from hundreds to thousands. A comparison was shown between the proposed approach and established techniques coming from the POP toolbox and MPT. It was shown that the established approaches work well and are able to yield the full multiparametric solution. However, when comparing the solution determined at different time breakpoints, before yielding the full solution, the proposed approach demonstrated a clear advantage. The presented algorithm is able to determine the largest volume of the full solution faster than the other approaches. The partial solution based on the volume occupied is important because it is an indication of the total amount of the parametric solution identified, unlike the number of critical regions explored. The comparison between total solution explored vs the number of critical regions explored is exemplified in Fig. 4.9. For this example problem, while the Graph and MPT solutions yield over 10 times the number of critical regions, it is seen that the volume occupied by these critical regions is quite low, especially compared to the proposed algorithm.

From the figures depicting the number of critical regions, the proposed algorithm has a slower rate of identification compared to the other algorithms. There are two primary reasons for this. The first reason is that the larger critical regions have been identified, and the algorithm is spending time searching for the smaller critical regions. These small critical regions can be difficult to find from a sample based algorithm. The second reason is the lack of a stronger fathoming criteria for the postulated convex spaces, stemming from Step 18 in Algorithm 1. The current fathoming criterion prevents convex spaces from exploring within a given critical region. However, if all neighboring critical regions have been identified, then there is no reason to further explore that particular region. Incorporating a fathoming criteria that can account for explored neighboring critical regions would allow the algorithm to explore more critical regions faster.

4.4 Conclusion

In this chapter, a novel space exploration was presented for mpLP and mpQP problems. In addition, a novel metric for analyzing a partial multiparametric solution that does not rely on the number of critical regions was developed. Due to the nature of the proposed exploration strategy, optimal partitions that occupy a significant portion of volume in the parametric space are prior-

itized. The ability to prioritize larger critical regions is a salient feature that is not exhibited by existing multiparametric algorithms.

Randomly generated numerical examples were presented that are large scale in the number of variables and constraints to showcase the proposed algorithms ability to identify the volumetrically significant critical regions compared to the state-of-the-art solvers, especially in the early phase of the exploration. The promising results determined from this algorithm encourage future research utilizing the underlying concepts of the proposed algorithm.

4.5 Future Work

Two key improvements to the proposed algorithm that will provide significant benefit are (i) improving the termination criterion, and (ii) providing an early termination criterion based on the recursion depth.

4.5.1 Stronger Termination Criterion

In its current form, the proposed approach is run until a final time is reached. An improved termination criterion would make use of multiparametric theory to fathom a postulated convex space not only if all of the vertices share the same active set combination, but if the postulated convex space cannot explore additional critical regions. For example if the multiparametric solution has two critical regions, any postulated convex space that contains the boundary of the critical region will be further decomposed even though there are no more critical regions to be identified. However, utilizing an improved strategy that verifies if any additional critical region can exist, and using this as a means to further fathom additional postulated convex spaces would provide a strong termination criterion that guarantees full exploration of the space.

4.5.2 Bounding the Maximum Critical Region Volume Unexplored

Currently, early termination of the algorithm is performed based on a run time. Analysis can be performed to identify if the partial volume is plateauing and this can be utilized as a early stopping criterion. However, another strategy worth exploring avoids calculating the volume of the identified critical regions, which can be an expensive operation. Demonstrated in Fig. 4.10, the objective

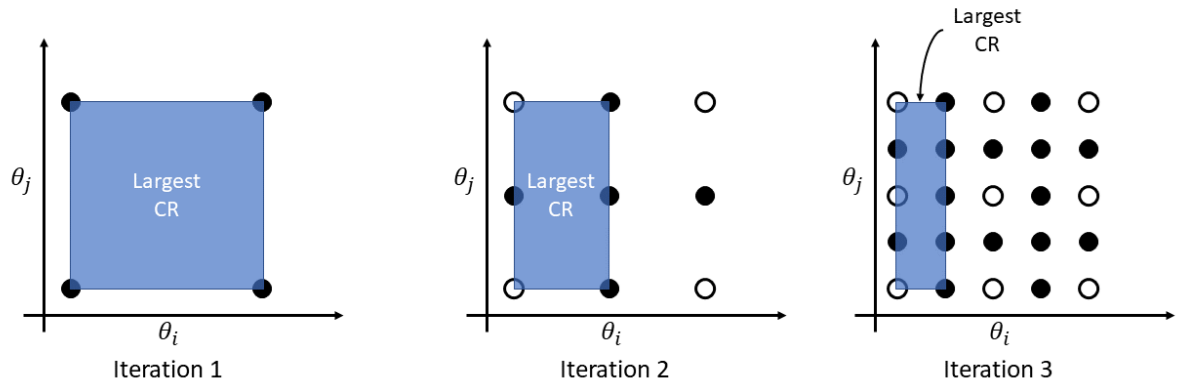


Figure 4.10: The largest critical region that can be explored at various recursion depths if hyper-boxes are used to decompose the space. The filled vertices are the vertices generated at the current recursion depth.

would be to identify *a priori* the volume of the largest critical region that could be unexplored by the proposed strategy as a function of recursion depth. By providing a boundary on the largest critical region volume unexplored, an early termination criterion can be developed such that all unexplored critical regions cannot exceed this threshold. Therefore, the partial solution would have a meaningful representation for the information that was not developed when the algorithm terminates.

5. IMPROVING HOT-START STRATEGIES THROUGH RANDOM WALKS AND A PARTIAL MULTIPARAMETRIC SOLUTION

Recently, starting with the seminal paper by Bemporad et al. [13], the improvements of multiparametric theory and algorithms have been significant [15, 17, 92, 59, 95, 2]. However, there are three inherent issues associated with multiparametric programming regardless of the strength of its theoretical underpinnings, (i) the point location problem, (ii) the memory requirement to store the full parametric solution, and (iii) algorithms to obtain the full parametric solution as the problem size increases. Promising research is being conducted in these fields to further (i) obtaining the multiparametric solution and (ii) improve its online performance [15, 2, 20, 21, 16, 94]. With increasing problem sizes, developing and utilizing the full multiparametric solution for online applications may not be the correct research direction to move in. Instead, an approach that utilizes the fundamental concepts from multiparametric programming to improve online strategies may prove to be more applicable.

It has been shown that active set strategies utilizing concepts from multiparametric programming can significantly improve the computational performance of determining the optimal control actions for MPC applications [22]. This strategy is denoted as a hot-start strategy. The work demonstrated significant online computational savings is achievable by only using the theory of multiparametric programming. However, the hot-start strategy, without adaptation, cannot manage situations in which large perturbations are expected. Large perturbations will cause the hot-start strategy to initialize from a poor starting point thus leading to unfavorable solution times.

In this chapter, to overcome the inherent shortcomings with using and developing the full multiparametric solution, an improved hot-start strategy is presented. The strategy has its fundamental mechanism based on a standard hot-start strategy [22] with the adaptation of incorporating *relevant* critical regions *a priori*. The developed critical regions provide insurance of a suitable initialization in an online setting. Closed loop simulations, involving numerically generated state space models and their corresponding MPC formulations, are performed to validate the proposed strategy.

5.1 Key Contribution

To improve the performance of hot-start strategies involving active set information, *relevant* critical regions are determined offline. These critical regions are then used to provide an active set to initialize a hot-start strategy. The novelty is determining and defining a *relevant* critical region, and integrating this concept into a hot-start strategy.

5.1.1 Relevant Critical Region

In multiparametric programming, a critical region is uniquely identified by an active set combination. Different established multiparametric algorithms aim to explore all existing critical regions for a given problem. However, these algorithms and the corresponding theory of critical regions do not provide indication for why two critical regions are different except by comparing the active sets. In Chapter 4 a novel perspective was presented and the developed critical regions were examined based on their size, or volume. By utilizing the concept of volume, a meaningful quantity can be defined to represent a critical region. In this chapter, the concept of a *relevant* critical region is defined to be a critical region that occupies a significant amount of volume compared to other critical regions. In other words, larger critical regions in a multiparametric solution are considered relevant.

5.2 Random Walks

Chapter 4 demonstrated an algorithmic procedure for identifying large critical regions. However, the limitation of this algorithm was its ability to scale with the number of uncertain parameters associated with the multiparametric problem. In this chapter, because the goal is to simply develop volumetrically significant critical regions and not explore the parameter space to its entirety, a random walk strategy is employed.

Random walks enable a uniform distribution to be developed in an arbitrary bounded space. In particular, hit and run sampling is performed to develop a random sample of points in the parameter space of the associated multiparametric programming problem [96]. The random sample of points is not expected to determine every critical region, and in fact this is the intent. Instead, the random

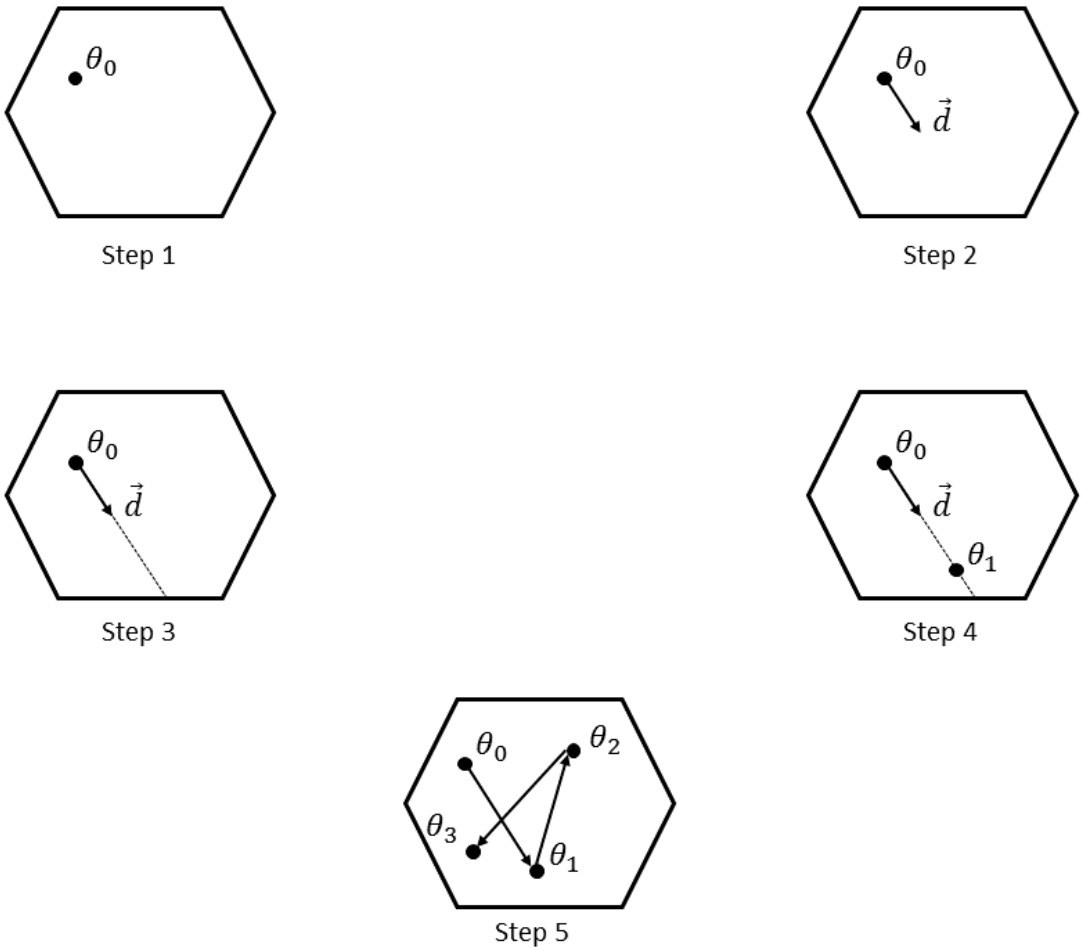


Figure 5.1: Step 1) Start with an interior point. Step 2) Randomly choose a direction. Step 3) The distance from the point to the edge of the space is determined. Step 4) Randomly select a random point between the point and the distance determined along the chosen direction, this is the updated point. Step 5) Repeat until termination.

sampling has a high likelihood of identifying volumetrically significant critical regions.

Many strategies for random walks exist in the literature, however because of its ease of implementation and its effectiveness, the hit and run sampling algorithm is utilized. Core details of hit and run sampling are not incorporated in this chapter, but a schematic representation of the algorithm is presented in Fig. 5.1, and is adapted from [96].

5.2.1 Hit and Run Sampling in Multiparametric Programming

The hit and run sampling strategy is implemented on the convex polytope defined by the feasible region of the multiparametric programming problem, Eq. (5.1).

$$Ax \leq b + F\theta \tag{5.1}$$

where the polytope is defined in the optimization space x, and the uncertain parameter space θ. To perform the hit and run sampling, first a feasible point is chosen $p_0 = [x_0^T|\theta_0^T]^T$. Then, a random direction is chosen, $\vec{d}$. The maximum distance from the initial point p_0 to the boundary of the polytope is determined. In other words, find the smallest positive t satisfying $[A|-F](p_0+\vec{d}t) = b$. The next point is sampled from the polytope by selecting a random number between 0 and the value t, and moving in the direction $\vec{d}$ by this amount. This procedure is repeated until enough sample points have been collected. The amount of sample points needed depends on the dimensionality of the polytope, but scales favorably (i.e. polynomially). From the sampled points, a random selection is utilized as the basis to develop their associated critical region. Because the sampled points from the hit and run procedure produce a random distribution, and a random selection is made from these points, the critical regions identified have a high probability of being volumetrically significant. It is possible to utilize every point from the random walk, however there is a trade off between storing all of these critical regions and the computational improvement during online implementation.

The presented hit and run sampling procedure is the standard approach for sampling in a convex polytope. However, because the objective is to determine a random sampling of points in the parameter space, it is not necessary to consider the full dimensional space of the polytope. By performing the random walk in the parameter space, the number of total points to ensure a random distribution is significantly reduced. To perform the hit and run sampling in the parameter space, a slight adjustment is needed for the original strategy. To identify the largest distance that can be traversed, the following optimization problem is solved, Eq. (5.2), where the initial point is now defined as θ_0.

$$\begin{aligned} \min_{t} \quad & t \\ \text{s.t.} \quad & Ax \leq b + F(\theta_0 + \vec{dt}) \\ & t > 0 \end{aligned} \tag{5.2}$$

5.3 Preliminaries

Details regarding the model predictive control formulation assumed, and relevant multiparametric programming theory are presented.

5.3.1 Model Predictive Control

The MPC formulation assumed in this chapter is given by Problem (5.3).

$$\begin{aligned} \min_{u_0,\ldots,u_{N-1}} \quad & \vec{x}_N^T P \vec{x}_N + \sum_{i=1}^{N-1} \vec{x}_i^T Q_i \vec{x}_i + \sum_{i=0}^{N-1} u_i^T R_i u_i \\ \text{s.t.} \quad & \vec{x}_{t+1} = A_d \vec{x}_t + B_d u_t, \quad \vec{x}_0 = \vec{x}(0) \\ & \underline{\mathrm{x}} \leq \vec{x}_t \leq \bar{x} \\ & \underline{\mathrm{u}} \leq u_t \leq \bar{u} \end{aligned} \tag{5.3}$$

where $\vec{x}_t$ is the state of system at time t, $\vec{u}_t$ is the manipulated action at time t, $Q_i \succ 0$ and $R_i \succ 0$ are weight matrices, P is the terminal weight matrix and is the solution to the discrete time algebraic Ricatti equation, A_d and B_d define the evolution of the state space in discrete time, $\bar{x}$ and $\underline{\mathrm{x}}$ are the upper and lower bounds for the states of the system, $\bar{u}$ and $\underline{\mathrm{u}}$ are the upper and lower bounds for the manipulated actions of the system, and N is the control and output horizon.

5.3.2 Multiparametric Programming

Problem (5.3) can be reformulated to its multiparametric quadratic programming (mpQP) counterpart[13], problem (5.4), where the uncertain parameter θ is defined by the initial state vector $\vec{x}_0$, and the decision variable x is defined by $[u_0^T, ..., u_{N-1}^T]^T$. The variable θ and x will represent the uncertain parameter vector and the optimization vector, respectively, for the rest of

the discussion on multiparametric programming.

$$\begin{aligned} z(\theta) = \min_{x} \quad & (Qx + H\theta + c)^T x \\ \text{s.t.} \quad & Ax \leq b + F\theta \\ & \theta \in \Theta = \{\theta \mid \tilde{A}\theta \leq \tilde{b}\} \end{aligned} \tag{5.4}$$

where $Q \succ 0$, $\tilde{A} \in \mathbb{R}^{p \times q}$ and $\tilde{b} \in \mathbb{R}^{p \times 1}$ define the bounds for the uncertain parameter vector, and $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^{m \times 1}$, and $F \in \mathbb{R}^{m \times q}$ define the resulting system of inequalities.

Relevant properties related to problem (5.4) have been explored in the literature[13, 15, 92, 59, 16] and are summarized below.

Theorem 5.3.1. (Solution of mpQP)[13] Consider the mpQP defined by problem (5.4) with $Q \succ 0$. The space of feasible parameters, $\Theta_f \subseteq \Theta$, is convex, the optimization variables, $x(\theta) : \Theta_f \to \mathbb{R}^n$, are continuous and piecewise affine, and the objective function, $z(\theta) : \Theta_f \to \mathbb{R}$, is continuous and piecewise quadratic. In the case of mpLP, the objective function is continuous and piecewise affine.

Definition 5.3.1. (Critical Region)[59] Given an index set of a possible active set combination $\mathcal{A} \subseteq \{1, ..., m\}$, and its associated lagrange multipliers, $\lambda(\theta)$, the critical region, CR, is defined as the polytopic region where the active set combination, $\mathcal{A}$, defines the solution to problem (5.4) for a range of θ.

$$CR = \{\theta \mid A_{\{1,..,m\}\setminus\mathcal{A}}\theta \leq b_{\{1,..,m\}\setminus\mathcal{A}} \wedge \tilde{A}\theta \leq \tilde{b} \ \wedge \lambda(\theta) \geq 0\}$$

Remark. Given an optimal active set combination, $\mathcal{A}$, and following the matrix notation of problem (5.4), the optimization variables, $x(\theta)$, and lagrange multipliers, $\lambda(\theta)$, are determined as follows.

$$\begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} 2Q & A_{\mathcal{A}}^T \\ A_{\mathcal{A}} & \mathbf{0} \end{bmatrix}^{-1} \left(\begin{bmatrix} -H \\ F_{\mathcal{A}} \end{bmatrix} \theta + \begin{bmatrix} -c \\ b_{\mathcal{A}} \end{bmatrix} \right)$$

Definition 5.3.2. (Adjacent critical regions)[59] Given a full dimensional critical region, CR_i, a critical region CR_j is adjacent iff $int(CR_i) \cap int(CR_j) = \emptyset$ and $CR_i \cap CR_j \neq \emptyset$

Definition 5.3.3. (Facet-to-facet)[59] The facet-to-facet property is said to hold if CR_i is uniquely adjacent to CR_j along their common facet, illustrated by Fig 5.2.

Theorem 5.3.2. (Active sets of adjacent critical regions)[92]

Given an active set combination $\{i_1, ..., i_k\}$ and its corresponding critical region, CR_0, with all redundant constraints removed. Let CR_i be an adjacent critical region and assume the linear inequality constraint qualification (LICQ) holds on their common facet $\mathcal{F} = CR_0 \cap \mathcal{H}$, where $\mathcal{H}$ is the separating hyperplane. Additionally, let $\lambda(\theta)$ be the lagrange multipliers associated with the active set combination of CR_0. Then:

Type I: If $\mathcal{H}$ is defined by $A_{i_{k+1}} x^*(\theta) = b_{i_{k+1}} + F_{i_{k+1}} \theta$, then CR_i is defined by the active set combination $\{i_1, ..., i_k, i_{k+1}\}$

Type II: If $\mathcal{H}$ is defined by $\lambda_{i_k}(\theta) = 0$, then CR_i is defined by the active set combination $\{i_1, ..., i_{k-1}\}$

Remark. Theorem 5.3.2, illustrated by Fig 5.3, also contains (i) the condition of no weakly active constraints in the critical region CR_0, and (ii) the critical region, prior to removing redundant constraints, has no facet with coincident hyperplanes. Having weakly active constraints is a consequence of dual degenerate problems. The problems explored in this work are convex multi-parametric quadratic programming problems, i.e. $Q \succ 0$, therefore dual degeneracy cannot exist. Facets of critical regions with coincident hyperplanes are rare occurrences and the adjacent critical region has a limited subset of possible active set combinations [92].

Remark. The condition of LICQ holding on the common facet is the strictest condition of Theorem 5.3.2. However, if LICQ does not hold on the common facet, and $\mathcal{H}$ is of Type 1, the possible active set combination of the neighboring critical region is reduced to $\{i_{k+1}\}$ and a proper subset of $\{i_1, ..., i_k\}$[92].

Lemma 5.3.1. (Facet-to-facet condition)[59] The facet-to-facet property holds between two adjacent critical regions if the conditions for Theorem 5.3.2 are satisfied.

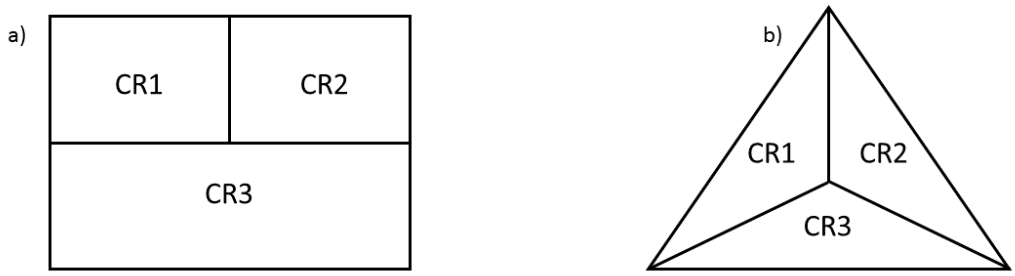


Figure 5.2: Critical regions where the facet-to-facet property (a) is violated and (b) holds.

5.4 Standard Hot-Start Strategy

A hot-start strategy involves utilizing a developed critical region around a previous parameter realization to identify the next optimal critical region. If the facet-to-facet property holds at all critical regions, then exploration of the parameter space to identify the optimal critical region where the current parameter realization lies is readily implementable with standard linear algebra techniques. To ensure the facet-to-facet property holds, and the neighboring critical regions share a hyperplane of Type I or Type II, two assumptions are made. The first assumption is that LICQ is not violated for a given critical region, ensuring the facet-to-facet property holds by Lemma 5.3.1. In the context of model predictive control, this assumption is not restrictive, where a slight perturbation of the weight matrices and constraints can ensure LICQ holding [92]. The second assumption, LICQ holding on the common facet between two adjacent critical regions, is not restrictive because the facet-to-facet property is maintained. However, further analysis is required when determining the neighboring active set combination [92]. Algorithm 2 provides pseudocode for a hot-start strategy and is summarized as follows.

- Initialization -Given an initial measurement, θ^*, the optimal solution at this point requires an initial active set combination. If the mpQP is defined from an MPC formulation that

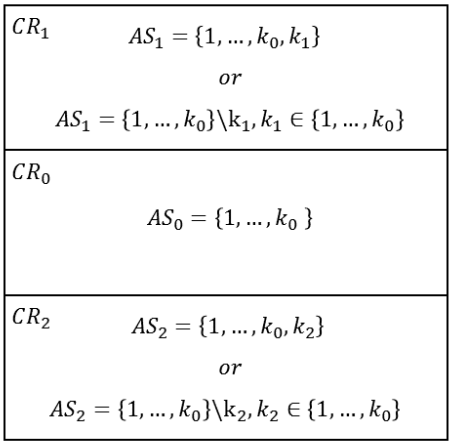


Figure 5.3: The active set combination of adjacent critical in accordance with Theorem 5.3.2[2].

contains the origin, then the critical region associated with no active constraints exists [97]. The measurement θ^* is checked to exist in the unconstrained critical region, otherwise a point is selected in the unconstrained critical region, where the algorithm proceeds to the update step. Another strategy to determine an initial active set combination is to solve a deterministic optimization problem at θ^*. The solution to the deterministic optimization problem provides the active set combination that is used to construct the critical region, CR_0, and control law, i.e. the affine function relating the optimization variables to the uncertain parameters, $x(\theta) = H\theta + G$. The determined control law is utilized until a new parameter realization, θ_i^*, does not belong to the initial critical region, $\theta_i^* \notin CR_0$. If $\theta_i^* \notin CR_0$, proceed to the update step.

- Update -The update corresponds to determining the new critical region, CR_i, and associated control law where the current parameter realization exists, $\theta_i^* \in CR_i$. First, a direction vector, $\vec{d}$, that goes from the previous parameter realization, $\theta_i^* \in CR_0$, to the new parameter realization, θ_i^*, is determined. Then, given the current critical region, CR_0, the previous pa-

rameter realization, θ^*, and the direction vector, $\vec{d}$, it is possible to identify which constraint that defines the critical region will be violated first, seen by Eq.(5.5).

$$t = (\tilde{b} - \tilde{A}\theta^*) \oslash \tilde{A}\vec{d} \tag{5.5a}$$

$$\mathcal{I} = \{i \mid i = \text{argmin}(t \geq 0)\} \tag{5.5b}$$

where $\oslash$ indicates Hadamard division and $\mathcal{I}$ defines the constraint violated of the critical region. The hyperplane defined by $[\tilde{A}_{\mathcal{I}}, \tilde{b}_{\mathcal{I}}]$, is identified as either a Type I or Type II constraint defined in Theorem 5.3.2. A Type I constraint indicates a new constraint is added to the active set combination, $[\mathcal{A}, \mathcal{I}]$, and a Type II constraint indicates a constraint is removed from the active set combination $[\mathcal{A} \setminus \mathcal{I}]$. Given the new active set combination, the critical region, CR_i, is identified. If $\theta_i^* \in CR_i$ the optimal action is identified, otherwise a new point p that lies in CR_i is identified that lies along the direction vector $\vec{d}$. The point, p, critical region, CR_i, and direction vector, $\vec{d}$, are used to identify the next neighboring critical region and associated active set combination. This procedure continues until θ_i^* exists in the newly explored critical region.

Fig. 5.4 provides a visualization of the proposed algorithm in the parameter space. The white arrows indicate the direction vector, the numbers indicate the parameter realizations and the order in which they were revealed, and the white 'x' indicates where the direction vector passed over a critical region. The (uncertain) parameter vector is defined as $\theta = [x_{0,1}, x_{0,2}]^T$.

The functions in Algorithm 2 are defined as follows.

GETCR(AS)

- Given an optimal active set combination, the corresponding critical region can be determined.

GETDIRECTION(θ, p)

- Given the previous parameter realization, and the new parameter realization, a direction vector is determined between these points.

Algorithm 2 Hot-Start Strategy

Require: θ ▷ Parameter realization
Require: CR ▷ Critical region
Require: p ▷ Point inside CR
if $\theta \notin CR$ **then**
  $direction \leftarrow$ GETDIRECTION(θ, p) ▷ Direction vector for p to θ
  while $\theta \notin CR$ **do**
    $whichConstraint \leftarrow$ GETCONSTRAINT$(p, direction, CR)$ ▷ CR facet
    if THEOREM5.3.2CHECK(AS,whichConstraint) **then**
      $AS =$ GETAS$(AS, whichConstraint)$ ▷ Identify neighboring CR
    else
      $AS =$ USEQP$(AS, whichConstraint)$
    end if
    $CR =$ GETCR(AS)
  end while
end if

GETCONSTRAINT$(p, direction, CR)$

- The critical region constraint that is violated next is determined from the direction vector and the previous parameter realization.

THEOREM5.3.2CHECK$(AS, whichConstraint)$

- Verifies LICQ conditions hold on the facet of the critical region, defined by $whichConstraint$.

GETAS$(AS, whichConstraint)$

- Given the active set combination and which constraint is violated next, the neighboring critical region's active set is identified.

USEQP$(AS, whichConstraint)$

- Solves a deterministic QP with a reduced constraint list consisting of the possible active set for the given critical region.

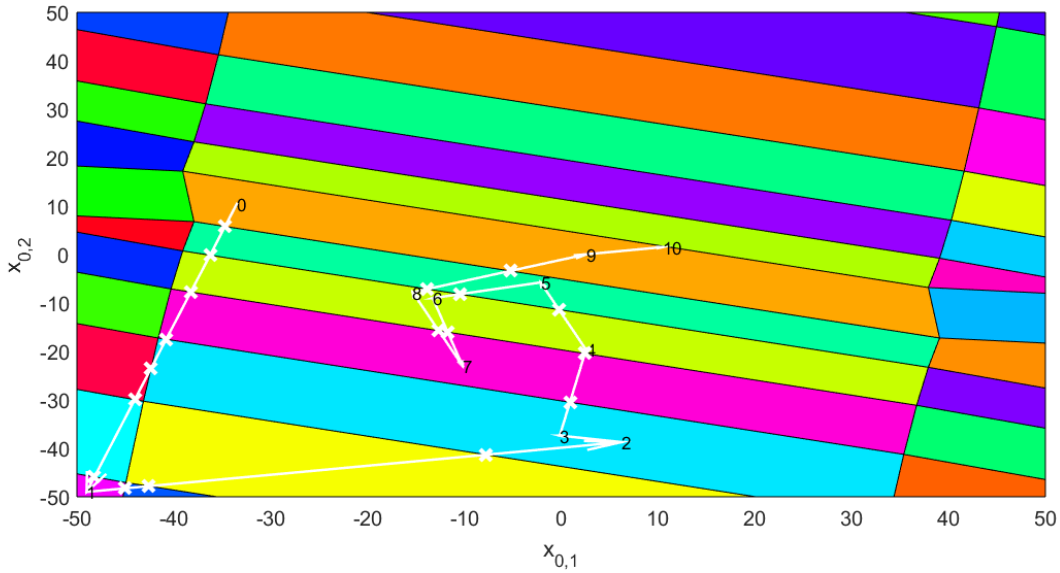


Figure 5.4: Visualization of a hot-start strategy on the associated multiparametric solution.

5.5 Methodology

The process for improving the hot-start strategy involves two phases. The first phase is an offline phase involving hit and run sampling on the feasible domain of the parameter space for the multiparametric problem. A select number of points are then used to develop the associated critical regions. The more sample points used to develop the critical regions leads to a larger number of critical regions that must be stored for online purposes. Therefore a balance is needed between selecting all sample points and selecting a single sample point. In addition, a feasible point is determined for each critical region (i.e. the Chebyshev center), and is stored for online use.

The second phase of the procedure is during online implementation. When a parameter realization is made (e.g. a measurement is made), the distance from the parameter realization to each of the stored feasible points corresponding to each critical region is calculated. The feasible point that is closest to the parameter realization is chosen, and its corresponding critical region is selected. The selected critical region is then the starting point for the hot-start strategy presented in Section 5.4 and Algorithm 2. The pseudocode for the determination of which critical region to initialize with is provided in Algorithm 3.

Algorithm 3 Determining Critical Region

Require: θ ▷ Parameter realization
Require: CR_{List} ▷ Volumetrically significant critical regions
Require: P ▷ Feasible point for each $CR \in CR_{List}$
1: **for** $i = 1 \rightarrow |CR_{List}|$ **do**
2: $D(i) = \text{GETDISTANCE}(\theta, P(i))$
3: **end for**
4: $index = argmin(D)$
5: $CR = CR_{List}(index)$

5.6 Computational Results

The efficacy of the proposed approach is validated using a host of randomly generated MPC problems. The generated MPC problems across all problem sizes maintain an output and control horizon of 45, and 50 states (i.e. uncertain parameters). The results are obtained from closed loop simulation, where the randomly generated state space model makes use of the solution from the developed MPC from the proposed approach and a standard hot-start strategy. Table 5.1 provides a summary of the problem sizes, and results.

The problem sizes tested for the computational study are all of large size where developing the full multiparametric solution is not possible, and determining the optimal solution online using an out of the box QP solver is not practical under time constraints. For each size problem, the percent improvement is based on the average improvement over 100 runs. The time required to determine the solution for each strategy is on the order of seconds and ten times faster compared to developing the optimal solution using CPLEX. From Table 5.1, it is evident the proposed strategy returns significant computational improvement compared to a standard hot-start strategy. It is important to note that the closed loop simulation performed to develop the results does not incorporate a major pitfall of standard hot-start strategies, namely large perturbations that require significant traversal of the parameter space.

Table 5.1: Comparison of a standard hot-start strategy against the proposed approach.

Optimization Formulation (Variables, Constraints)	Manipulation Actions	Percent Improvement
900, 6300	20	17%
1350, 7200	30	21%
1800, 8100	40	39%

5.7 Conclusion

The solution of large scale multiparametric problems with current algorithms is intractable. However, the theory of multiparametric programming has the potential to provide significant computational improvements in online applications demonstrated by hot-start strategies. This chapter proposed a further improvement to standard hot-start strategies through the direct incorporation of multiparametric programming in the form of a partial solution. By utilizing random walk concepts, the partial multiparametric solution was developed offline and incorporated in an online hot-start strategy to improve the online computational performance. A computational study incorporating large randomly generated model predictive control formulation was used as the basis for comparison.

5.8 Future Work

The theory presented for a standard hot-start strategy and for its improvement is rigorously defined for mpLP and mpQP. However, many of the key properties extend to nonlinear convex multiparametric optimization problems. It was demonstrated that for online applications, utilizing multiparametric concepts has the potential for significant computational speed up for quadratic programs, where theory is well established. Therefore, by exploring nonlinear optimization problems, then there is an expectation of improved online performance as well.

To demonstrate the applicability of a hot-start strategy to the nonlinear multiparametric case (such as in real time optimization formulations), we present the motivating example given by Eq. (5.6). In this example, the objective is nonlinear and convex, and the constraints are linear. There are two uncertain parameters, and for different realization cause the optimal solution to change. Fig. 5.5 provides the multiparametric solution and corresponding active sets for this motivating problem. Note the nonlinear and nonconvex critical regions. In the figure, the active set combination corresponds to whether the constraint is active or not. For example $[001]$ means the third constraint is active, $-x_1 - x_2 \leq \frac{1}{2}$, at the optimal solution.

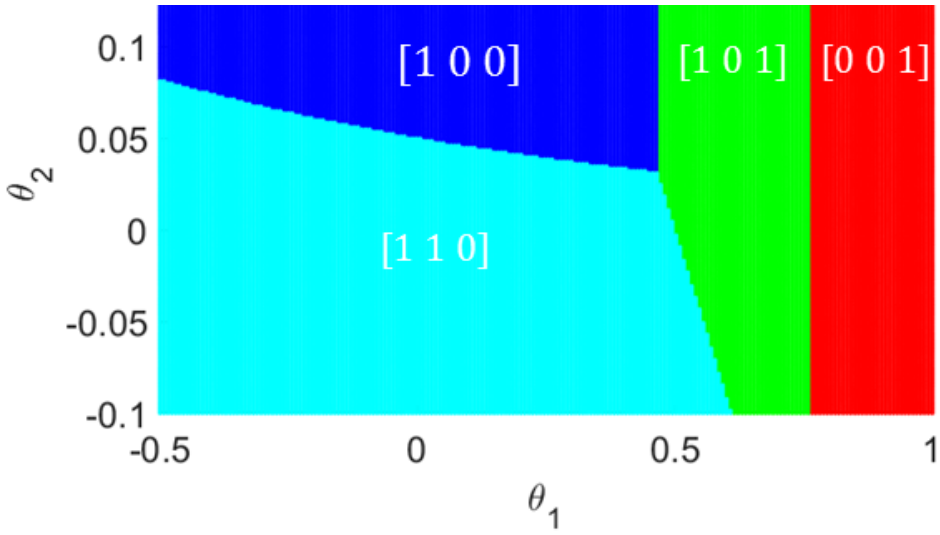


Figure 5.5: The solution to the nonlinear multiparametric programming problem defined by Eq. (5.6). The white text presents the active set combination for the corresponding critical region.

$$
\begin{aligned}
\min_{x} \quad & e^{x_1} + x_2^2 \\
\text{s.t.} \quad & -x_1 + \frac{x_2}{10} \leq \theta_1 \\
& -x_2 \leq \theta_2 \\
& -x_1 - x_2 \leq \frac{1}{2} \\
& -\frac{1}{2} \leq \theta_1 \leq 1 \\
& -\frac{1}{10} \leq \theta_2 \leq 0.15
\end{aligned}
\tag{5.6}
$$

Give the realization $\theta_{-1} = [0\ 0.1]^T$, the optimal solution is at $x^* = [-0.005\ -0.0498]^T$, where the first constraint $-x_1 + \frac{x_2}{10} \leq \theta_1$ is active. To identify the optimal solution at $\theta_0 = [0.9\ 0.1]^T$ the procedure for a hot-start strategy is followed, and is presented in Fig. 5.6, yielding the solution of $x^* = [-0.7388\ 0.2388]$.

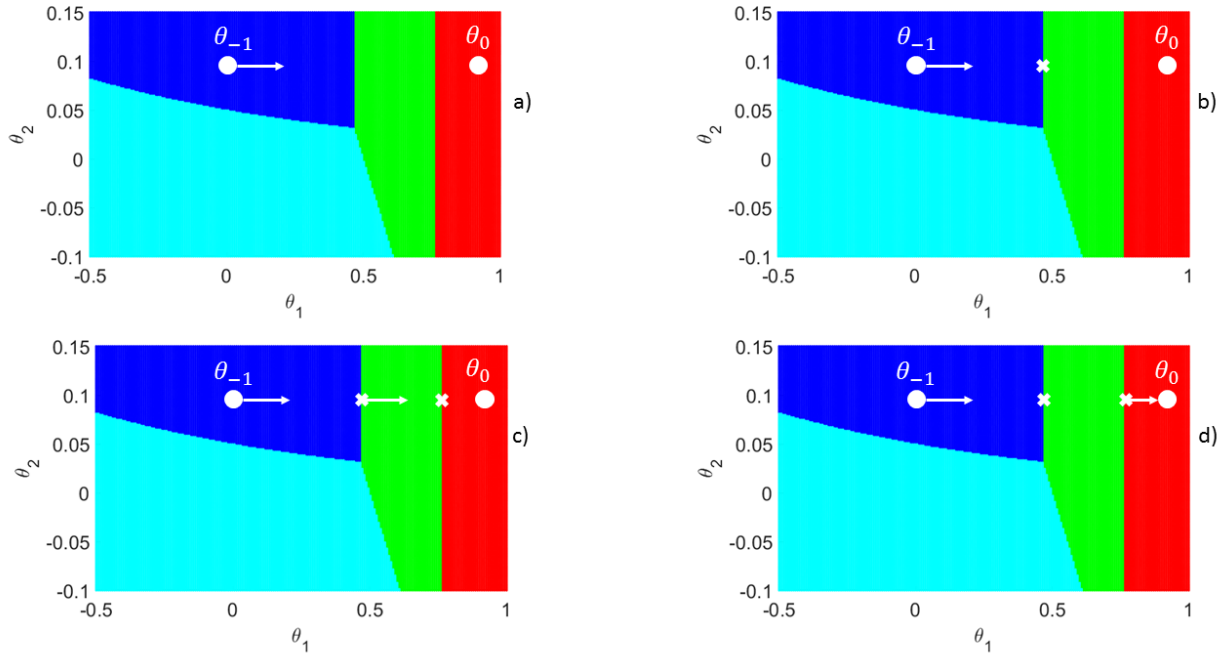


Figure 5.6: The implementation of the hot start strategy for the nonlinear multiparametric programming problem. a) Define the direction vector between the previous parameter realization and the current realization. b) Identify the facet of the critical region that will be crossed. c) Identify the facet of the new critical region that will be crossed. d) Verify the new parameter realization exists in the current critical region and determine the optimal solution.

6. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

6.1 Conclusions

Model predictive control has the capability of providing optimal operation to a chemical engineering process. By utilizing multiparametric programming, MPC strategies are improved during online implementation. These improvements include enhancing existing online strategies and developing the offline, explicit optimal solution.

In this dissertation, it was shown that (i) deep learning can be incorporated into multiparamtric programming, (ii) the decision space volume of the multiparametric solution provides a metric to compare different mpMPC formulations, (iii) examining the accumulated volume of a partial multiparametric solution provides meaningful insight in the developed critical regions, and (iv) the partial multiparametric solution can improve determining the online optimal solution.

In Chapter 2, it was shown that incorporating deep learning into multiparametric programming provides improved surrogate model formulations to model complex processes and phenomena. Because deep learning models can be exactly recast in a mixed-integer linear formulation, they are naturally embedded in an optimization formulation. To alleviate the burden of solving a mixed-integer optimization problem online, multiparametric programming is a proven methodology to develop the offline, explicit solution allowing the use of these advanced surrogate models to be utilized in real time settings.

Developing surrogate models for complex processes to be used in multiparametric MPC formulations yields an inherent discrepancy between model complexity and the true optimal solution. Quantifying this difference provides rigour to compare different mpMPC formulations directly. Therefore, described in Chapter 3, a novel metric based on the volume of the decision space for the developed optimization problem is utilized. This metric aims to quantify how much of the decision space volume remains feasible following the incorporation of a surrogate model.

To solve multiparametric optimization problems involving a large number of optimization vari-

ables and constraints, a novel geometric algorithm was proposed in Chapter 4. It was shown that the proposed algorithm identifies a large portion of the parameter space quickly compared to existing algorithms. Additionally, a new perspective was presented to analyze a partial multiparametric solution that is based on the accumulated volume of the identified critical regions. This perspective provides meaning to an identified set of critical regions that previous algorithms do not provide.

In Chapter 5, it was shown that the development of a partial multiparametric solution utilizing random walks provides an improved initialization procedure for hot-start strategies. The partial solution is utilized online to determine which active set combination, or critical region, is closest to the parameter realization. By utilizing this technique, poor initialization is avoided. enabling a standard hot-start strategy to have improved computational performance.

In summary, the key contributions of this dissertation are:

Chapter 2 The incorporation of deep learning models based on artificial neural networks with ReLU activation functions and multiparametric model predictive control formulations.

Chapter 3 A metric to compare different multiparametric model predictive control explicit solutions by comparing their decision space volume.

Chapter 4 A novel geometric algorithm to solve large scale multiparametric programming problems that prioritizes identifying volumetrically significant critical regions.

Chapter 4 A new perspective to analyze a partial multiparametric solution by calculating the accumulated volume of the explored critical regions.

Chapter 5 Utilizing random walks to develop a partial multiparametric solution to large scale multiparametric optimization problems.

Chapter 5 Improving hot-start strategies via a partial multiparametric solution in the context of model predictive control (and real time optimization).

6.2 Future Research

The contributions presented in this dissertation provided improvements to multiparametric programming for model predictive control. Many of the presented ideas and concepts are readily extendable to new research directions.

6.2.1 Machine Learning Models

Chapter 2 demonstrated the applicability of neural networks with ReLU activation functions into multiparametric model predictive control formulations. However, other machine learning models are readily implementable in a similar methodology. These machine learning models include:

1. Recurrent neural networks with ReLU activation functions
2. Multivariate adaptive regression splines
3. Decision trees
4. Random Forests

These machine learning models provide various benefits and it is critical to demonstrate the efficacy of incorporating these models into multiparametric programming.

6.2.2 Chance Constrained Programming

Another avenue that is worth exploring is in the area of robust model predictive control. In particular, formulating the robust optimization problem using chance constraints. Reformulating the potentially overly conservative robust formulation with chance constraints reduces the conservativeness of the developed optimal solution. The developed chance constrained model predictive controller, i.e. Eq. (6.1), can then be solved explicitly using multiparametric programming.

$$\begin{aligned} \min_{x} \quad & \sum_{i=0}^{OH} x^T Q x \\ \text{s.t.} \quad & Pr[A(\theta)x + B(\theta)u \in X] \geq 1 - \epsilon \\ & \theta \in \Theta \end{aligned} \tag{6.1}$$

In addition, the risk tolerance level, ϵ can be treated as an uncertain parameter. The benefit is apparent for two reasons. The first reason is the explicit, offline solution can be analyzed for its sensitivity to the risk tolerance level. The second reason is in multi-level optimization formulations, the optimal risk tolerance level can be identified.

6.2.3 Real Time Optimization

This dissertation demonstrated a partial multiparametric solution coupled with a hot-start strategy has the capability of improving the online computational performance of large scale quadratic programs. The concepts utilized in both (i) developing the partial multiparametric solution, and (ii) the hot-start strategy are readily transferable to real time optimization formulations, such as economic model predictive control.

By utilizing the same concepts that were shown to improve large scale quadratic programs, the online implementation of large scale real time optimization formulations will exhibit a similar benefit. For this strategy to effective, it is imperative to identify which active/inactive constraint will be violated when entering a neighboring critical region. Because of the associated nonlinearities, the necessary steps for this procedure require advanced strategies, even though the fundamental concept remains the same.

REFERENCES

[1] J. Katz, B. Burnak, and E. N. Pistikopoulos, "The impact of model approximation in multi-parametric model predictive control," *Chemical Engineering Research and Design*, vol. 139, pp. 211 – 223, 2018.

[2] P. Ahmadi-Moshkenani, T. A. Johansen, and S. Olaru, "Combinatorial approach toward multiparametric quadratic programming based on characterizing adjacent critical regions," *IEEE Transactions on Automatic Control*, vol. 63, no. 10, pp. 3221–3231, 2018.

[3] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733 – 764, 2003.

[4] M. J. Tippett and J. Bao, "Distributed model predictive control based on dissipativity," *AIChE Journal*, vol. 59, no. 3, pp. 787–804, 2013.

[5] L. S. Maxeiner and S. Engell, "Hierarchical mpc of batch reactors with shared resources," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12041 – 12046, 2017.

[6] L. Ljung, *System Identification Theory for the User*. EngleWood Cliffs, NJ: Prentice-Hall, 1999.

[7] H. S. Sidhu, A. Narasingam, P. Siddhamshetty, and J. S.-I. Kwon, "Model order reduction of nonlinear parabolic pde systems with moving boundaries using sparse proper orthogonal decomposition: Application to hydraulic fracturing," *Computers & Chemical Engineering*, vol. 112, pp. 92–100, 2018.

[8] A. Narasingam and J. S.-I. Kwon, "Data-driven identification of interpretable reduced-order models using sparse regression," *Computers & Chemical Engineering*, vol. 119, pp. 101–111, 2018.

[9] A. Narasingam, P. Siddhamshetty, and J. S.-I. Kwon, "Handling spatial heterogeneity in reservoir parameters using proper orthogonal decomposition based ensemble kalman filter for

model-based feedback control of hydraulic fracturing," *Industrial & Engineering Chemistry Research*, vol. 57, no. 11, pp. 3977–3989, 2018.

[10] A. Narasingam and J. S.-I. Kwon, "Development of local dynamic mode decomposition with control: Application to model predictive control of hydraulic fracturing," *Computers & Chemical Engineering*, vol. 106, pp. 501–511, 2017.

[11] D. M. Himmelblau, "Accounts of experiences in the application of artificial neural networks in chemical engineering," *Industrial & Engineering Chemistry Research*, vol. 47, no. 16, pp. 5782–5796, 2008.

[12] A. M. Schweidtmann, W. R. Huster, J. T. Lüthje, and A. Mitsos, "Deterministic global process optimization: Accurate (single-species) properties via artificial neural networks," *Computers & Chemical Engineering*, vol. 121, pp. 67 – 74, 2019.

[13] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, pp. 3–20, 2002.

[14] C. N. Jones and M. Morrari, "Multiparametric linear complementarity problems," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 5687–5692, 2006.

[15] A. Gupta, S. Bhartiya, and P. Nataraj, "A novel approach to multiparametric quadratic programming," *Automatica*, vol. 47, no. 9, pp. 2112 – 2117, 2011.

[16] R. Oberdieck, N. A. Diangelakis, and E. N. Pistikopoulos, "Explicit model predictive control: A connected-graph approach," *Automatica*, vol. 76, pp. 103 – 112, 2017.

[17] M. Wittmann-Hohlbein and E. N. Pistikopoulos, "On the global solution of multi-parametric mixed integer linear programming problems," *Journal of Global Optimization*, vol. 57, no. 1, pp. 51–73, 2013.

[18] V. M. Charitopoulos, L. G. Papageorgiou, and V. Dua, "Multi-parametric mixed integer linear programming under global uncertainty," *Computers & Chemical Engineering*, vol. 116, pp. 279–295, 2018.

[19] R. Oberdieck and E. N. Pistikopoulos, "Explicit hybrid model-predictive control: The exact solution," *Automatica*, vol. 58, pp. 152 – 159, 2015.

[20] X. Xiu and J. Zhang, "Grid k-d tree approach for point location in polyhedral data sets application to explicit mpc," *International Journal of Control*, pp. 1–9, 2018.

[21] M. Kvasnica, B. Takács, J. Holaza, and S. D. Cairano, "On region-free explicit model predictive control," *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 3669–3674, 2015.

[22] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit mpc," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 816–830, 2008.

[23] D. M. Himmelblau, "Applications of artificial neural networks in chemical engineering," *Korean Journal of Chemical Engineering*, vol. 17, no. 4, pp. 373–392, 2000.

[24] C. Shang and F. You, "Data analytics and machine learning for smart process manufacturing: Recent advances and perspectives in the big data era," *Engineering*, 2019.

[25] A. Tran, M. Pont, A. Aguirre, H. Durand, M. Crose, and P. D. Christofides, "Bayesian model averaging for estimating the spatial temperature distribution in a steam methane reforming furnace," *Chemical Engineering Research and Design*, vol. 131, pp. 465–487, 2018.

[26] B. Beykal, F. Boukouvala, C. A. Floudas, and E. N. Pistikopoulos, "Optimal design of energy systems using constrained grey-box multi-objective optimization," *Computers & Chemical Engineering*, vol. 116, pp. 488 – 502, 2018.

[27] S. H. Kim and F. Boukouvala, "Machine learning-based surrogate modeling for data-driven optimization: a comparison of subset selection for regression techniques," *Optimization Letters*, pp. 1–22, 2019.

[28] W. W. Tso, C. D. Demirhan, C. A. Floudas, and E. N. Pistikopoulos, "Multi-scale energy systems engineering for optimal natural gas utilization," *Catalysis Today*, 2019.

[29] L. Chiang, B. Lu, and I. Castillo, "Big data analytics in chemical engineering," *Annual Review of Chemical and Biomolecular Engineering*, vol. 8, no. 1, pp. 63–85, 2017.

[30] A. Shokry, P. Vicente, G. Escudero, M. Pérez-Moya, M. Graells, and A. Espuña, "Data-driven soft-sensors for online monitoring of batch processes with different initial conditions," *Computers & Chemical Engineering*, vol. 118, pp. 159 – 179, 2018.

[31] B. R. Hough, D. A. Beck, D. T. Schwartz, and J. Pfaendtner, "Application of machine learning to pyrolysis reaction networks: Reducing model solution time to enable process optimization," *Computers & Chemical Engineering*, vol. 104, pp. 56 – 63, 2017.

[32] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine learning-based predictive control of nonlinear processes. part i: Theory," *AIChE Journal*, vol. 65, no. 11, p. e16734, 2019.

[33] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine-learning-based predictive control of nonlinear processes. part ii: Computational implementation," *AIChE Journal*, vol. 65, no. 11, p. e16734, 2019.

[34] B. Rister and D. L. Rubin, "Piecewise convexity of artificial neural networks," *Neural Networks*, vol. 94, pp. 34 – 45, 2017.

[35] J. Pfrommer, C. Zimmerling, J. Liu, L. Kärger, F. Henning, and J. Beyerer, "Optimisation of manufacturing process parameters using deep neural networks as surrogate models," *Procedia CIRP*, vol. 72, pp. 426 – 431, 2018.

[36] Y. Nagata and K. H. Chu, "Optimization of a fermentation medium using neural networks and genetic algorithms," *Biotechnology Letters*, vol. 25, no. 21, pp. 1837–1842, 2003.

[37] M. J. Asher, B. F. W. Croke, A. J. Jakeman, and L. J. M. Peeters, "A review of surrogate models and their application to groundwater modeling," *Water Resources Research*, vol. 51, no. 8, pp. 5957–5973, 2015.

[38] S. Chen, K. Saulnier, N. Atanasov, D. D. Lee, V. Kumar, G. J. Pappas, and M. Morari, "Approximating explicit model predictive control using constrained neural networks," in *2018 Annual American Control Conference (ACC)*, pp. 1520–1527, June 2018.

[39] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *Advances in neural information processing systems*, pp. 2924–2932, 2014.

[40] J. H. Friedman, "Multivariate adaptive regression splines," *The Annals of Statistics*, vol. 19, no. 1, pp. 1–67, 1991.

[41] L. A. Francis, "Martian chronicles . " is mars better than neural networks ?," 2003.

[42] K. Eckle and J. Schmidt-Hieber, "A comparison of deep networks with relu activation function and linear spline-type methods," *Neural Networks*, vol. 110, pp. 232 – 242, 2019.

[43] M. Fischetti and J. Jo, "Deep neural networks and mixed integer linear optimization," *Constraints*, vol. 23, no. 3, pp. 296–309, 2018.

[44] B. Grimstad and H. Andersson, "Relu networks as surrogate models in mixed-integer linear programs," *arXiv preprint arXiv:1907.03140*, 2019.

[45] J. Lee and H. J. Chang, "Explicit model predictive control for linear time-variant systems with application to double-lane-change maneuver," *PLoS ONE*, vol. 13, no. 12, 2018.

[46] G. M. Kopanos and E. N. Pistikopoulos, "Reactive scheduling by a multiparametric programming rolling horizon framework: A case of a network of combined heat and power units," *Industrial & Engineering Chemistry Research*, vol. 53, no. 11, pp. 4366–4386, 2014.

[47] S. Avraamidou and E. N. Pistikopoulos, "A multi-parametric optimization approach for bilevel mixed-integer linear and quadratic programming problems," *Computers & Chemical Engineering*, vol. 125, pp. 98 – 113, 2019.

[48] B. Burnak, J. Katz, N. A. Diangelakis, and E. N. Pistikopoulos, "Simultaneous process scheduling and control: A multiparametric programming-based approach," *Industrial & Engineering Chemistry Research*, vol. 57, no. 11, pp. 3963–3976, 2018.

[49] B. Burnak, N. A. Diangelakis, J. Katz, and E. N. Pistikopoulos, "Integrated process design, scheduling, and control using multiparametric programming," *Computers & Chemical Engineering*, vol. 125, pp. 164 – 184, 2019.

[50] R. E. Steuer, Y. Qi, and M. Hirschberger, "Portfolio optimization: New capabilities and future methods," *Zeitschrift für Betriebswirtschaft*, vol. 76, no. 2, pp. 199–220, 2006.

[51] M. Karasuyama, N. Harada, M. Sugiyama, and I. Takeuchi, "Multi-parametric solution-path algorithm for instance-weighted support vector machines," *Machine Learning*, vol. 88, no. 3, pp. 297–330, 2012.

[52] R. Oberdieck, N. A. Diangelakis, M. M. Papathanasiou, I. Nascu, and E. N. Pistikopoulos, "POP - Parametric Optimization Toolbox," *Industrial & Engineering Chemistry Research*, vol. 55, pp. 8979–8991, 2016.

[53] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, (Zürich, Switzerland), pp. 502–510, July 17–19 2013. `http://control.ee.ethz.ch/~mpt`.

[54] J. Katz, I. Pappas, S. Avraamidou, and E. N. Pistikopoulos, "Integrating deep learning models and multiparametric programming," *Computers & Chemical Engineering*, 2019.

[55] A. Pinkus, "Approximation theory of the mlp model in neural networks," *Acta Numerica*, vol. 8, p. 143195, 1999.

[56] C. Ning and F. You, "Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming," *Computers & Chemical Engineering*, vol. 125, pp. 434–448, 2019.

[57] A. Afram, F. Janabi-Sharifi, A. S. Fung, and K. Raahemifar, "Artificial neural network (ann) based model predictive control (mpc) and optimization of hvac systems: A state of the art review and case study of a residential hvac system," *Energy and Buildings*, vol. 141, pp. 96 – 113, 2017.

[58] D. L. Poole and A. K. Mackworth, *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, 2010.

[59] J. Spjøtvold, E. C. Kerrigan, C. N. Jones, P. Tøndel, and T. A. Johansen, "On the facet-to-facet property of solutions to convex parametric quadratic programs," *Automatica*, vol. 42, no. 12, pp. 2209 – 2214, 2006.

[60] R. Oberdieck, N. A. Diangelakis, I. Nascu, M. M. Papathanasiou, M. Sun, S. Avraamidou, and E. N. Pistikopoulos, "On multi-parametric programming and its applications in process systems engineering," *Chemical Engineering Research and Design*, vol. 116, pp. 61–82, 2016.

[61] D. van Ravenzwaaij, P. Cassey, and S. D. Brown, "A simple introduction to markov chain monte–carlo sampling," *Psychonomic Bulletin & Review*, vol. 25, pp. 143–154, Feb 2018.

[62] B. Grimstad and H. Andersson, "Relu networks as surrogate models in mixed-integer linear programs," *Computers & Chemical Engineering*, p. 106580, 2019.

[63] E. Camacho, F. Rubio, M. Berenguel, and L. Valenzuela, "A survey on control schemes for distributed solar collector fields. part i: Modeling and basic control approaches," *Solar Energy*, vol. 81, no. 10, pp. 1240 – 1251, 2007.

[64] D. Limon, I. Alvarado, T. Alamo, M. Ruiz, and E. Camacho, "Robust control of the distributed solar collector field acurex using mpc for tracking," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 958 – 963, 2008.

[65] "Therminol 55: Heat transfer fluid," tech. rep., Eastman Chemical Company, 2015. Accessed on 04.17.2019.

[66] M. N. O. Sadiku and C. N. Obiozor, "A simple introduction to the method of lines," *International Journal of Electrical Engineering Education*, vol. 37, no. 3, pp. 282–296, 2000.

[67] C. Maxey and A. Andreas, "Rotating shadowband radiometer," Tech. Rep. NREL DA-5500-56512, Oak Ridge National Laboratory (ORNL), 2007.

[68] B. Zeng, Y. An, and L. Kuznia, "Chance constrained mixed integer program: Bilinear and linear formulations, and benders decomposition," *arXiv preprint arXiv:1403.7875*, 2014.

[69] N. Martinez, H. Anahideh, J. Rosenberger, D. Martinez, V. Chen, and B. Wang, "Global optimization of non-convex piecewise linear regression splines," *Journal of Global Optimization*, 2017.

[70] E. F. Camacho and C. Bordons, *Nonlinear Model Predictive Control: An Introductory Review*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[71] M. Morari and J. Lee, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, no. 4, pp. 667 – 682, 1999.

[72] D. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967 – 2986, 2014.

[73] L. Santos, P. Afonso, J. Castro, N. Oliveira, and L. Biegler, "On-line implementation of nonlinear mpc: An experimental case study," *Control Engineering Practice*, vol. 9, no. 8, pp. 847–857, 2001.

[74] N. A. Diangelakis, B. Burnak, J. Katz, and E. N. Pistikopoulos, "Process design and control optimization: A simultaneous approach by multi-parametric programming," *AIChE Journal*, vol. 63, no. 11, pp. 4827–4846, 2017.

[75] K. Fruzzetti, A. Palazoğlu, and K. McDonald, "Nolinear model predictive control using hammerstein models," *Journal of Process Control*, vol. 7, no. 1, pp. 31 – 41, 1997.

[76] J. A. Primbs and V. Nevistic, "Mpc extensions to feedback linearizable systems," in *Proceedings of the American Control Conference*, pp. 2073–2077, 1997.

[77] D. Simon, J. Lofberg, and T. Glad, "Nonlinear model predictive control using feedback linearization and local inner convex constraint approximations," pp. 2056–2061, 2013.

[78] J. Hahn and T. Edgar, "An improved method for nonlinear model reduction using balancing of empirical grammians," *Computers & Chemical Engineering*, vol. 26, pp. 1379–1397, 2002.

[79] M. Rewieński and J. White, "A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 2, pp. 155–170, 2003.

[80] W. Xie, Y. Bonis, and C. Theodoropoulos, "Off-line model reduction for on-line linear mpc of nonlinear large-scale distributed systems," *Computers & Chemical Engineering*, vol. 35, pp. 750–757, 2011.

[81] L. Ozkan, M. Kothare, and C. Georgakis, "Model predictive control of nonlinear systems using piecewise linear models," *Computers & Chemical Engineering*, vol. 24, pp. 793–799, 2000.

[82] M. Kheradmandi and P. Mhaskar, "Model predictive control with closed-loop re-identification," *Computers & Chemical Engineering*, vol. 109, pp. 249–260, 2018.

[83] P. Balaguer and R. Vilanova, "Frequency-dependent approach to model validation for iterative identification and control schemes," *IET Control Theory and Applications*, vol. 3, no. 1, pp. 98–109, 2009.

[84] C. Alvarado and C. Garcia, "Comparison of statistical metrics and a new fuzzy method for validating linear models used in model predictive control controllers," *Industrial & Engineering Chemistry Research*, vol. 57, no. 10, pp. 3666–3677, 2018.

[85] K. Khalilpour and I. Karimi, "Parametric optimization with uncertainty on the left hand side of linear programs," *Computers & Chemical Engineering*, vol. 60, pp. 31–40, 2014.

[86] V. Charitopoulos, L. Papageorgiou, and V. Dua, "Multi-parametric linear programming under global uncertainty," *AIChE Journal*, vol. 63, no. 9, pp. 3871–3895, 2017.

[87] C. Ziogou, E. N. Pistikopoulos, M. C. Georgiadis, S. Voutetakis, and S. Papadopoulou, "Empowering the performance of advanced nmpc by multiparametric programming–an application to a pem fuel cell system," *Industrial & Engineering Chemistry Research*, vol. 52, no. 13, pp. 4863–4873, 2013.

[88] E. N. Pistikopoulos, V. Dua, N. A. Bozinis, A. Bemporad, and M. Morari, "On-line optimization via off-line parametric optimization tools," *Computers & Chemical Engineering*, vol. 26, no. 2, pp. 175 – 185, 2002.

[89] K. Kouramas, N. Faísca, C. Panos, and E. N. Pistikopoulos, "Explicit/multi-parametric model predictive control (mpc) of linear discrete-time systems by dynamic and multi-parametric programming," *Automatica*, vol. 47, no. 8, pp. 1638 – 1645, 2011.

[90] G. Takács and B. Rohal'-Ilkiv, *Basic MPC Formulation*, pp. 207–251. London: Springer London, 2012.

[91] T. Gal and J. Nedoma, "Multiparametric linear programming," *Management Science*, vol. 18, no. 7, pp. 406–422, 1972.

[92] P. Tøndel, T. A. Johansen, and A. Bemporad, "An algorithm for multi-parametric quadratic programming and explicit mpc solutions," *Automatica*, vol. 39, no. 3, pp. 489–497, 2003.

[93] R. Oberdieck, M. Wittmann-Hohlbein, and E. N. Pistikopoulos, "A branch and bound method for the solution of multiparametric mixed integer linear programming problems," *Journal of Global Optimization*, vol. 59, pp. 527–543, Jul 2014.

[94] B. Burnak*, J. Katz*, and E. N. Pistikopoulos, "A space exploration algorithm for multiparametric programming via delaunay triangulation," *Optimization and Engineering*, 2019.

[95] V. Sakizlis, J. Perkin, and E. N. Pistikopoulos, "Explicit solutions to optimal control problems for constrained continuous-time linear systems," *IEE Proceedings: Control Theory and Applications*, 2005.

[96] A. Yao and D. Kane, "walkr: Mcmc sampling from non-negative convex polytopes," *The Journal of Open Source Software*, vol. 2, 03 2017.

[97] A. Bemporad and C. Filippi, "Suboptimal explicit receding horizon control via approximate multiparametric quadratic programming," *Journal of Optimization Theory and Applications*, vol. 117, no. 1, pp. 9–38, 2003.

APPENDIX A

ADDITIONAL PROBLEM DETAILS

A.1 Critical regions defining a multiparametric problem with left hand side uncertainty

The regions CR_1, CR_2, and CR_3 are defined as follows.

$$CR_1 = \begin{cases} -\frac{2X_{max}+2X_{max}a-ax_0+X_{max}a^2-a^2x_0}{a^2+2a+2} \leq 0 \\ \frac{2X_{min}+2X_{min}a-ax_0+X_{min}a^2-a^2x_0}{a^2+2a+2} \leq 0 \\ -\frac{2X_{max}+2X_{max}a+ax_0+X_{max}a^2}{a^2+2a+2} \leq 0 \\ \frac{2X_{min}+2X_{min}a+ax_0+X_{min}a^2}{a^2+2a+2} \leq 0 \\ -U_{max} - \frac{2a^2(b+ab)+2ab}{2((b+ab)^2+b^2)}x_0 \leq 0 \\ U_{min} + \frac{2a^2(b+ab)+2ab}{2((b+ab)^2+b^2)}x_0 \leq 0 \end{cases}$$

$$CR_2 = \begin{cases} 2Qx_0a^3b + 2QU_{max}a^2b^2 + 2Qx_0a^2b + 4QU_{max}ab^2 + 2Qx_0ab + 4QU_{max}b^2 \leq 0 \\ U_{max}b - X_{max} + ax_0 \leq 0 \\ X_{min} - U_{max}b - ax_0 \leq 0 \\ a^2x_0 - X_{max} + U_{max}(b+ab) \leq 0 \\ X_{min} - a^2x_0 - U_{max}(b+ab) \leq 0 \end{cases}$$

$$CR_3 = \begin{cases} -(2Qx_0a^3b + 2QU_{min}a^2b^2 + 2Qx_0a^2b + 4QU_{min}ab^2 + 2Qx_0ab + 4QU_{min}b^2) \leq 0 \\ U_{min}b - X_{max} + ax_0 \leq 0 \\ X_{min} - U_{min}b - ax_0 \leq 0 \\ a^2x_0 - X_{max} + U_{min}(b + ab) \leq 0 \\ X_{min} - a^2x_0 - U_{min}(b + ab) \leq 0 \end{cases}$$

A.2 Defining Equations for the Reaction and Separation process

Reactor 1

$$\dot{x}_{A1} = \frac{F_{in\_1}}{V_1}(x_{A_{F1}} - x_{A1}) + \frac{F_R}{V_1}(x_{Ar} - x_{A1}) - k_1 e^{\frac{-E_1}{RT_1}} x_{A1} \tag{A.1}$$

$$\dot{x}_{B1} = \frac{F_{in\_1}}{V_1}(x_{B_{F1}} - x_{B1}) + \frac{F_R}{V_1}(x_{Br} - x_{B1}) + k_1 e^{\frac{-E_1}{RT_1}} x_{A1} - k_2 e^{\frac{-E_2}{RT_1}} x_{B1} \tag{A.2}$$

$$\dot{T}_1 = \frac{F_{in\_1}}{V_1}(T_{F_1} - T_1) + \frac{F_R}{V_1}(T_3 - T_1) - \frac{\Delta H_1}{C_p} k_1 e^{\frac{-E_1}{RT_1}} x_{A1} - \frac{\Delta H_2}{C_p} k_2 e^{\frac{-E_2}{RT_1}} x_{B1} + \frac{UA_1(T_{J1} - T_1)}{\rho C_P V_1} \tag{A.3}$$

$$\dot{T}_{J1} = \frac{F_{j,in\_1}}{V_{J1}}(T_{J1_{in}} - T_{J1}) + \frac{UA_1(T_1 - T_{J1}}{\rho_J C_{pJ} V_{J1}} \tag{A.4}$$

Reactor 2

$$\dot{x}_{A2} = \frac{F_2}{V_2}(x_{A1} - x_{A2}) + \frac{F_{in\_2}}{V_2}(x_{A_{F2}} - x_{A2}) - k_1 e^{\frac{-E_1}{RT_2}} x_{A2} \tag{A.5}$$

$$\dot{x}_{B2} = \frac{F_2}{V_2}(x_{B1} - x_{B2}) + \frac{F_{in\_2}}{V_2}(x_{B_{F2}} - x_{B2}) + k_1 e^{\frac{-E_1}{RT_2}} x_{A2} - k_2 e^{\frac{-E_2}{RT_2}} x_{B2} \tag{A.6}$$

$$\dot{T}_2 = \frac{F_2}{V_2}(T_1 - T_2) + \frac{F_{in\_2}}{V_2}(T_{F2} - T_2) - \frac{\Delta H_1}{C_p} k_1 e^{\frac{-E_1}{RT_2}} x_{A2} - \frac{\Delta H_2}{C_p} k_2 e^{\frac{-E_2}{RT_2}} x_{B2} + \frac{UA_2(T_{J2} - T_2)}{\rho C_P V_2} \tag{A.7}$$

$$\dot{T}_{J2} = \frac{F_{j,in\_2}}{V_{J2}}(T_{J2_{in}} - T_{J2}) + \frac{UA_2(T_2 - T_{J2})}{\rho_J C_{pJ} V_{J2}} \tag{A.8}$$

Separator

$$\dot{x}_{A3} = \frac{F_3}{V_3}(x_{A2} - x_{A3}) - \frac{F_R + F_p}{V_3}(x_{Ar} - x_{A3}) \tag{A.9}$$

$$\dot{x}_{B3} = \frac{F_3}{V_3}(x_{B2} - x_{B3}) - \frac{F_R + F_p}{V_3}(x_{Br} - x_{B3}) \tag{A.10}$$

$$\dot{T}_3 = \frac{F_3}{V_3}(T_2 - T_3) + \frac{Q_3}{\rho C_p V_3} \tag{A.11}$$

Phase Equilibrium

$$x_{Ar} = \frac{\alpha_A x_{A3}}{\alpha_A x_{A3} + \alpha_B x_{B3} + \alpha_C x_{C3}} \tag{A.12}$$

$$x_{Br} = \frac{\alpha_B x_{B3}}{\alpha_A x_{A3} + \alpha_B x_{B3} + \alpha_C x_{C3}} \tag{A.13}$$

$$x_{Cr} = \frac{\alpha_C x_{C3}}{\alpha_A x_{A3} + \alpha_B x_{B3} + \alpha_C x_{C3}} \tag{A.14}$$

APPENDIX B

LIST OF PUBLICATIONS AND PRESENTATIONS

- **J. Katz**, I. Pappas, S. Avraamidou, E. N. Pistikopoulos, "Integrating Deep Learning Models and Multiparametric Programming". *Under Review*. 2019.
- B. Burnak*, **J. Katz***, E. N. Pistikopoulos, "A Space Exploration Algorithm for Multiparametric Programming via Delaunay Triangulation," *Under Review*. 2019.
- **J. Katz**, B. Burnak, E. N. Pistikopoulos, "The Impact of Model Approximation in Multiparametric Model Predictive Control," *Chemical Engineering Research and Design* 2018, 139, 211-223.
- M. M. Papathanasiou, B. Burnak, **J. Katz**, N. Shah, E. N. Pistikopoulos, "Assisting continuous biomanufacturing through advanced control in downstream purification," *Computers & Chemical Engineering*, Special Issue 2019, 125, 232-248.
- B. Burnak, N. A. Diangelakis, **J. Katz**, E. N. Pistikopoulos, "Integrated process design, scheduling, and control using multiparametric programming," *Computers & Chemical Engineering*, Special Issue 2019, 125, 164-184.
- Y. Tian, I. Pappas, B. Burnak, **J. Katz**, E. N. Pistikopoulos, "A Systematic Framework for the Synthesis of Operable Process Intensification Systems - Reactive Separation Systems," Computers & Chemical Engineering 2019, Accepted Manuscript.
- B. Burnak, **J. Katz**, N. A. Diangelakis, E. N. Pistikopoulos, "Simultaneous Process Scheduling and Control: A Multiparametric Programming Based Approach," *Industrial & Engineering Chemistry Research* 2018, 57 (11), 3963-3976.
- N. A. Diangelakis, B. Burnak, **J. Katz**, E. N. Pistikopoulos, "Process Design and Control optimization: A simultaneous approach by multi-parametric programming," *AIChE Journal*

2017, 63 (11), 4827-4846.

Conference Proceedings

- **J. Katz**, I. Pappas, S. Avraamidou, E. N. Pistikopoulos, "Integrating Deep Learning and Explicit MPC for Advanced Control," American Control Conference (ACC); *Under Review*. 2019.

- **J. Katz**, I. Pappas, S. Avraamidou, E. N. Pistikopoulos, "The Integration of Explicit MPC and ReLU based Neural Networks," 21st IFAC World Congress; *Under Review*. 2019.

- **J. Katz**, N. A. Diangelakis, E. N. Pistikopoulos, "Model Approximation in Multiparametric Optimization and Control - A Computational Study," 13th International Symposium on Process Systems Engineering (PSE 2018); Elsevier, 2018; pp 655-660.

- B. Burnak, **J. Katz**, N. A. Diangelakis, E. N. Pistikopoulos, "Integration of Design, Scheduling, and Control of Combined Heat and Power Systems: A Multiparametric Programming Based Approach," 13th International Symposium on Process Systems Engineering (PSE 2018); Elsevier, 2018; pp 2203-2208.

- I. Pappas, S. Avraamidou, **J. Katz**, N. A. Diangelakis, E. N. Pistikopoulos, "A Multiparametric Programming Based Approach for Multiobjective Model Predictive Control - Application to a Bioreactor System," 30th European Symposium on Computer-Aided Process Engineering (ESCAPE-30); 2019.

- Y. Tian, I. S. Pappas, **J. Katz**, B. Burnak, S. Avraamidou, N. A. Diangelakis, E. N. Pistikopoulos, "Towards a systematic framework for the synthesis of operational process intensification systems - Application to reactive distillation systems," 29th European Symposium on Computer-Aided Process Engineering (ESCAPE-29); 2019; pp 73-78.

- M. M. Papathanasiou, B. Burnak, **J. Katz**, E. N. Pistikopoulos, "Control of Small-Scale Chromatographic Systems under Disturbances," Foundations of Computer-Aided Process Design (FOCAPD 2019); 2019; p Accepted manuscript.

- M. M. Papathanasiou, B. Burnak, **J. Katz**, E. N. Pistikopoulos, “Control of a dual mode separation process via multi-parametric Model Predictive Control,” 12th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems (DYCOPS 2019); 2019; p Accepted manuscript.

Conference Presentations

- **J. Katz**, E. N. Pistikopoulos, "Enhancing Real Time Optimization through Multiparametric Programming". AICHE Fall 2019.
- **J. Katz**, B. Burnak, E. N. Pistikopoulos, "A Novel Geometric Based Algorithm to Solve Multiparametric Programming Problems". AICHE Fall 2018.
- **J. Katz**, B. Burnak, E. N. Pistikopoulos. "A Novel Geometric Based Algorithm to Solve Multiparametric Programming Problems: Application to Energy Systems". ERS 2018.
- **J. Katz**, N. A. Diangelakis, E. N. Pistikopoulos. "Model Approximation in Multiparametric Optimization and Control A Computational Study". ChEGSA 2018.
- **J. Katz**, N. A. Diangelakis, E. N. Pistikopoulos, "Model Approximation in Multiparametric Optimization and Control - A Computational Study". PSE 2018.
- **J. Katz**, B. Burnak, N. A. Diangelakis, E. N. Pistikopoulos, "Model Reduction and Approximation for Simultaneous Design, Control, and Scheduling". AICHE Fall 2017.
- **J. Katz**, B. Burnak, N. A. Diangelakis, E. N. Pistikopoulos, "Model Reduction and Approximation for Simultaneous Design, Control, and Scheduling". ERS 2017.