Graph-based Computation of Control Invariant Sets: Algorithms, Analysis and Applications

by

Benjamin Decardi-Nelson

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Process Control

Department of Chemical and Materials Engineering

University of Alberta

Abstract

Increasingly faced with sustainability and profitability objectives, chemical process plants have become very complex with many operating constraints. To achieve these objectives, a high level of automation is required. Unfortunately, the ability of the automated controllers to ensure that the control objectives are met at all future times is complicated by constraints on the available control energy and uncertainties present in the control system. Robust control invariant set (RCIS) and control invariant set (CIS) are fundamental tools used in the analysis of constrained, controlled dynamical systems. However, determining these sets is very difficult, especially in a nonlinear setting. This thesis tackled the problem of computing invariant sets in two directions.

In the first part of this thesis, we developed several graph-based invariant set (GIS) tools for computing invariant sets of constrained, controlled dynamical systems and proved the convergence of the set to the largest RCIS. Specifically, we first presented algorithms that inner and outer approximate the largest RCIS (and by extension CIS) contained in the state constraint. Thereafter, we identified several bottlenecks in the GIS algorithm and proposed remedial strategies including adaptive subdivision, parallelization, and system decomposition. The improved GIS has a much improved scalability compared to the standard GIS algorithm. We demonstrated the efficacy of the proposed algorithms and remedial strategies using several numerical examples, including a six dimensional continuous stirred tank reactor (CSTR).

In the second part of this thesis, we used the algorithms developed in the first part to

develop a robust economic model predictive control with zone tracking (RZEMPC) framework for nonlinear systems. Because the zone to be tracked is not necessarily robust control invariant, we proposed to obtain an RCIS subset of the zone to be tracked and introduced the concept of risk factor in the controller design. This not only ensured the stability of the closed-loop system but also ensured guaranteed economic performance in the presence of uncertainties. We conducted rigorous stability analysis and demonstrated the efficacy of the RZEMPC framework using a nonlinear CSTR example. Thereafter, we further improved the applicability of the RZEMPC framework to higher dimensional systems by tracking an ellipsoidal control invariant set instead of a polytopic control invariant set. We demonstrated the suitability of the proposed RZEMPC algorithm to avoid solvent flooding and over-circulation in the absorption column of a post-combustion CO_2 capture plant.

Preface

The materials presented in this thesis are part of the research project under the supervision of Dr. Jinfeng Liu, and is funded by the Natural Sciences and Engineering Research Council of Canada (NSERC). This research was also enabled in part by the support provided by Westgrid (https://www.westgrid.ca) and Compute Canada (http://www.computecanada.ca).

Chapter 2 of this thesis is a revised version of Benjamin Decardi-Nelson and Jinfeng Liu, Computing robust control invariant set of constrained nonlinear systems: a graph algorithm approach. *Computers & Chemical Engineering*, 145:107177, 2021.

Chapter 3 of this thesis is a revised version of Benjamin Decardi-Nelson and Jinfeng Liu, An efficient implementation of graph-based invariant set algorithm for constrained nonlinear dynamical systems which is currently under review in the Computers & Chemical Engineering journal.

Chapter 4 of this thesis is under preparation to be submitted as Benjamin Decardi-Nelson and Jinfeng Liu, Control invariant set computation for nonlinear systems: a distributed approach. A short version has been accepted for oral presentation in the 2022 American Control Conference in Atlanta, Georgia, USA.

Chapter 5 of this thesis is a revised version of Benjamin Decardi-Nelson and Jinfeng Liu, Robust economic model predictive control with zone tracking. *Chemical Engineering Research & Design*, 177:502-512, 2022.

Chapter 6 of this thesis is a revised version of Benjamin Decardi-Nelson and Jinfeng Liu. Robust economic MPC of the absorption column in post-combustion CO_2 capture through

zone tracking. *Energies*, 15:1140, 2022.

To my family

Acknowledgements

I cannot begin to express my deepest thanks to my advisor Dr. Jinfeng Liu, who provided the necessary advice and a stimulating environment for this research to be conducted. He allowed the research to be my own while steering me in the right direction whenever I needed it.

I would also like to express my thanks to Dr. Zukui Li, Dr. Stevan Dubljevic, Dr. Mahdi Tavakoli Afshari and Dr. Yang Shi for agreeing to serve on my examination committee, and for their valuable comments and suggestions which helped improve my thesis.

I would also like to extend my sincere thanks to Dr. Su Liu for the numerous discussions we had early on in the program which led to this research direction in control invariance and zone model predictive control.

I would also like to thank Dr. Xunyuan Yin who provided me with numerous collaboration opportunities through which I learnt to conduct research in systems and control.

I would also like to acknowledge the help and friendship of every member of the PSACE lab, especially Bernard Agyeman who was always available to listen and critique my ideas. Without them, my stay at the university of Alberta would not have been as pleasant as I experienced.

Finally, I would like to acknowledge my parents and numerous friends, who either actively or passively, helped me achieve the goals of this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Modern chemical process plants are becoming increasingly more complex with many operating constraints as a result of increased requirements for sustainability and profitability [3]. To ensure that these objectives are achieved in an optimal way, high level of automation is necessary. Automated controllers are deployed in chemical process plants in an attempt to drive the plants to operate in a desired way. Unfortunately, the design of controllers is complicated by potential system instability which may cause the plant to operate outside the safe region – usually in the form of hard state/output constraints – prescribed by the plant designers. Operating outside the prescribed safe region often lead to improper plant behaviour, potentially causing damage to equipment, loss of human lives, violation of environmental regulations and degraded plant profitability. A properly designed controller ensures that the system states stay within the prescribed safe sets. Therefore, it is important that an automated controller is designed to guarantee stability and constraint satisfaction, even in the presence of uncertainties. Figure 1.1 shows different sources of operating constraints and uncertainties in a typical feedback control loop. In an ideal situation where there are no limitations on the available control energy, controllability imply that the system

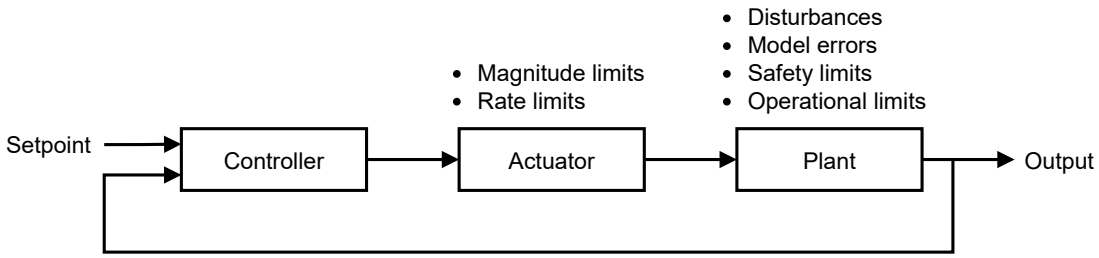


Figure 1.1: Feedback control loop with various constraints, limitations and disturbances.

can be controlled everywhere within the state constraint. However, in a real world, there are limitations on the available control energy for process control. Take for example a valve. It can open between a range of 0 – 100 % and nothing outside this range. The presence of limitations on the actuator, limit the ability of the control system from ensuring that the system states stay within the safe region at all future times [4]. As a demonstration, consider the linear discrete-time system in Example 1.1.

Example 1.1. *Consider the discrete-time linear system*

$$x^+ = 2x + u$$

where x and u are the system state and manipulated input respectively, and x^+ is the system state at the next time step.

In Example 1.1, the goal is to transfer the system from an initial state of $x = 1$ to a final state of $x = 0$ in finite time. This is possible in the absence of input constraints because the system is controllable. However, this may not be possible when a limitation is put on the input. The evolution of the system from an initial state of $x = 1$ under an unconstrained input and an input constrained to be within -0.5 and 0.5 is presented in Figure 1.2. It can be seen in the figure that without input constraints, the controller is able to drive the system state to the origin whereas the presence of the input constraint makes the controller unable to drive the state of the system to the origin or even stabilize the system. This shows that the presence of input constraints limit the ability of the controller to stabilize the system at

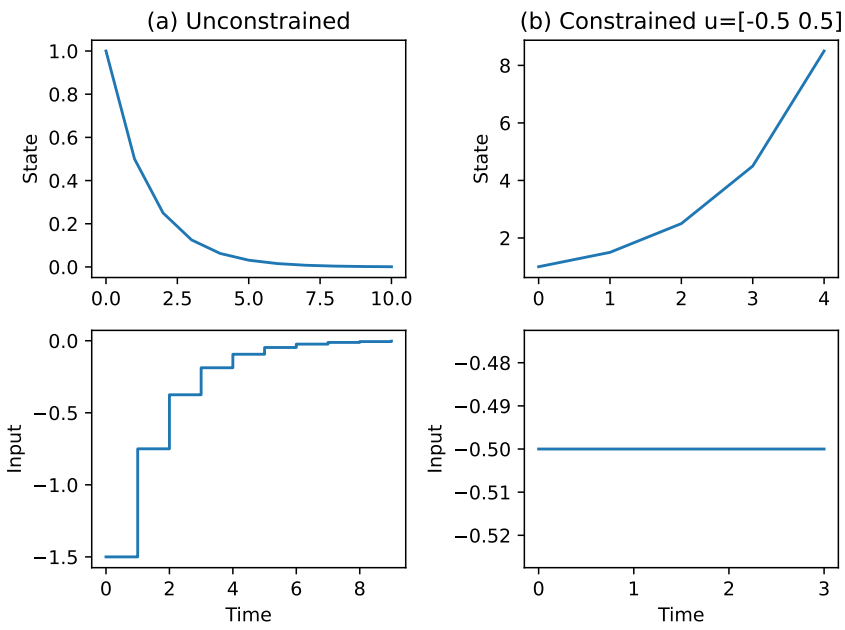


Figure 1.2: The effect of actuator limitations (hard input constraints) on the control system. The initial state of the system is $x = 1$.

the origin and eventually the states of the plants will violate the state constraints. In short, failure to account for input constraints, disturbances, etc. in the controller design could lead to poor performance and/or instability [5].

It therefore is necessary that the set of states for which a control input exist to keep the state of system within the state constraint at all times is known for controller design purposes. This set of states is known as a control invariant set (CIS). In addition, if disturbances are considered in the determination of this set, then the set is known as a robust control invariant set (RCIS). The concept of control invariant set is closely related to reachable sets [6], null controllable sets [7] and viability kernels [8]. The importance of control invariant sets in controller design and assessment is evident in the numerous attention it has received in the control literature (see [9] for an extensive survey on the subject). For example, it is used as a terminal region constraint in the design of Model Predictive Control (MPC) strategies to guarantee stability, recursive feasibility and constraint satisfaction [10, 11]. Also, advances in economic MPC with zone tracking [12, 13] further motivate the need to determine control

invariant sets. A Lyapunov function is a widely used tool for stability analysis of a system [14, 15, 16]. It also well known that sublevel sets of Lyapunov functions are invariant [9]. Therefore, finding the CIS or RCIS of a given system imply that a Lyapunov function can be constructed [9, 8, 17, 18]. More recently, control invariant set has been found to be useful in ensuring stability and constraint satisfaction in reinforcement learning [19, 20, 21, 22]. The above considerations motivate the need to find the largest (robust) control invariant sets contained in the state constraint. In this thesis, we will focus on the computation of CIS and RCIS of constrained controlled nonlinear systems and its applications to zone MPC.

1.2 Background

Despite the well known importance of control invariant sets, accurate and efficient determination of control invariant sets is still an open issue, even for linear systems. This is because determining a control invariant set for a given system involves considering all possible trajectories of the system. For several decades, the challenges of estimating control invariant sets for discrete-time systems have been widely studied in the systems and control literature [23]. Linear systems, in particular, have gotten a lot of attention [1, 24, 25, 26, 27]. Most of the algorithms for computing control invariant sets of linear systems are based on the dynamic programming technique [28], which provide convergence to the largest (with respect to inclusion) CIS. For general nonlinear systems however, only a few results exit [2, 29, 30]. Although Lyapunov functions are important tools for set invariance analysis, it is difficult to obtain such functions for nonlinear systems in general. Moreover, most algorithms for nonlinear systems do not provide convergence to the largest CIS. This is because of the inherent difficulty in computing control invariant sets of general nonlinear systems. Homer and Mhaskar presented an algorithm for estimating null controllable regions of nonlinear systems by enlarging an initial estimate of a control invariant set [5, 31]. However, the algorithm makes use of an invariance test function that require guessing an appropriate

input sequence for a specified prediction horizon which may not be easy especially the case of systems which are not input affine. This was further extended to an approach that solves a reverse time-optimal control problem in [7]. In these cases however, the presence of disturbances were not considered. Fiacchini and coworkers also presented an algorithm based on difference of two convex (DC) functions to estimate convex robust control invariant sets of nonlinear systems [2]. However, the algorithm requires contractivity and convergence to the largest RCIS was not provided. In the level-set algorithm proposed by Mitchell et al. [6], a grid-based algorithm which solves a time-dependent Hamilton-Jacobi formulation was used to estimate reachable sets of continuous systems with uncertainties.

Another major challenge with determining control invariant set is that many of the current state-of-the-art algorithms have inadequate scalability when it comes to the dimension of the system states. For example, the methods for linear time invariant (LTI) systems are based on vertex enumeration of the set's full-dimensional polytopic representation [1]. The number of vertices in the polytope grows exponentially as the system's dimension increase, resulting in intractable computation. For nonlinear systems, only a few results exist with majority of the algorithms either being grid-based [5, 6, 31] or set-based [32]. Similar issues which are considerably more challenging exist for nonlinear systems because of the state space discretization. A trade-off between set complexity and computational complexity is frequently made to overcome scalability concerns in determining control invariant sets. The methods presented in [33, 34] have been successful in computing control invariant set of high dimensional systems but assume a control invariant set of prescribed complexity or shape which results in faster computation but overly conservative sets. A decomposition and reconstruction based method for exact computation of backward reachable sets of self-contained nonlinear subsystems using the Halmilton-Jacobi (HJ) formulations was developed in [35]. In [36, 37], decentralized and distributed algorithms were developed to obtain a family of practical positively invariant sets of linear systems. In [38], a decomposition based method was proposed to compute the backward reachable sets of nonlinear systems. In these cases,

however, the missing interconnection information were treated as disturbances which results in conservative results.

Motivated by the above considerations, this thesis presents new algorithms for computing approximations of the largest robust control invariant set of general constrained time-invariant discrete-time uncertain nonlinear systems.

1.3 Contributions and thesis outline

In Chapter 2, we present a graph-based invariant set (GIS) algorithm for computing the largest robust control invariant sets (RCISs) of constrained nonlinear systems. The proposed approach is based on casting the search for the invariant set as a graph theoretical problem. Specifically, a general class of discrete-time time-invariant nonlinear systems is considered. First, the dynamics of a nonlinear system is approximated with a directed graph. Subsequently, the condition for robust control invariance is derived and an algorithm for computing the robust control invariant set is presented. The algorithm combines the iterative subdivision technique with the robust control invariance condition to produce outer approximations of the largest robust control invariant set at each iteration. Following this, we prove convergence of the algorithm to the largest RCIS as the iterations proceed to infinity. Based on the developed algorithm, an algorithm to compute inner approximations of the RCIS is also presented. A special case of input affine and disturbance affine systems is also considered. Finally, two numerical examples are presented to demonstrate the efficacy of the proposed method.

In Chapter 3, we present an improved and efficient implementation of the GIS algorithm for general discrete-time controlled nonlinear systems. We first identify the bottlenecks of the GIS algorithm through extensive analysis, and then provide remedial procedures to improve the implementation of the GIS algorithm. Specifically, we developed an adaptive subdivision scheme using a supervised machine learning-based algorithm to reduce the cell growth rate

and parallelize the graph construction step. We extensively demonstrate the performance of the improved GIS algorithm using a numerical example and compare the result to that of the standard GIS algorithm. The results show that the adaptive subdivision and the parallelization improved the speed of the algorithm by about 8x and 3x respectively, that of the standard GIS algorithm.

In Chapter 4, we present a distributed framework based on the graph algorithm for computing control invariant set for nonlinear cascade systems. The proposed algorithm exploits the structure of the interconnections within a process network. First, the overall system is decomposed into several subsystems with overlapping states. Second, the control invariant set for the subsystems are computed in a distributed manner. Finally, an approximation of the control invariant set for the overall system is reconstructed from the subsystem solutions and validated. We demonstrate the efficacy and convergence of the proposed method to the centralized graph-based algorithm using several numerical examples including a six dimensional continuous stirred tank reactor system.

In Chapter 5, we present a robust economic model predictive control (EMPC) formulation with zone tracking for discrete-time uncertain nonlinear systems. The proposed design ensures that the zone tracking objective is achieved in finite steps and at the same time optimizes the economic performance. In the proposed design, instead of tracking the original target zone, a robust control invariant set within the target zone is determined and is used as the actual zone tracked in the proposed EMPC. This approach ensures that the zone tracking objective is achieved within finite steps and once the zone tracking objective is achieved (the system state enters the robust control invariant set), the system state does not come out of the target zone anymore. To optimize the economic performance within the zone in the presence of disturbances, we introduce the notion of risk factor in the controller design. An algorithm to determine the economic zone to be tracked is provided. The risk factor determines the conservativeness of the controller and provides a way to tune the EMPC for better economic performance. A nonlinear chemical example is presented to demonstrate

the performance of the proposed formulation.

In Chapter 6, we present an EMPC with zone tracking algorithm as an effective means to ensure optimal operation of the absorption column of a post-combustion CO_2 capture plant. The proposed control algorithm incorporates a zone tracking objective and an economic objective to form a multi-objective optimal control problem. To ensure that the zone tracking objective is achieved in the presence of model uncertainties and time-varying flue gas flow rate, we propose a method to modify the original target zone with a control invariant set. The zone modification method combines both ellipsoidal control invariant set techniques and a back-off strategy. The use of ellipsoidal control invariant sets ensure that the method is applicable to large scale systems such as the absorption column. We present several simulation case studies that demonstrate the effectiveness and applicability of the proposed control algorithm to the absorption column in a post-combustion CO_2 capture plant.

Chapter 7 summarizes the main contributions of this thesis and discusses possible research directions in graph-based control invariant set algorithms.

Chapter 2

Computing robust control invariant sets of constrained nonlinear systems: A graph algorithm approach

2.1 Introduction

As mentioned in Chapter 1, algorithms for computing RCIS of general constrained nonlinear systems has received less attention in the systems and control literature compared to the linear counterpart. In addition to the methods for nonlinear systems mentioned earlier, a common approach for handling nonlinearity is to linearize the nonlinear system about an equilibrium point and then use a well-known linear algorithm to compute the RCIS [11]. This approach usually leads to overly conservative results due to the introduction of additional uncertainties into the system model. Another approach is to assume that the system has polynomial dynamics and then use the methods in [39, 40]. A major limitation is that most models of chemical processes have complex dynamics which is far from such an assumption.

Graph theoretical methods have been successfully used in the analysis of nonlinear systems [41, 42, 43, 44]. The idea is to approximate the trajectories of the dynamical system

using directed graphs and then analyze the constructed graph using graph theoretical methods. This method has been used in identifying or computing periodic orbits, invariant sets, recurrent sets, Lyapunov exponents, etc (see [45] for more applications). However, all the studies focused on autonomous dynamical systems with the exception of [44] which used graph theoretical method to determine control sets. Control sets are maximal subsets of the state space where complete controllability holds [46]. They are subsets of the largest control invariant sets. Moreover, only outer approximations of the invariant sets were considered in these studies which are not very useful in control.

Motivated by the success of graph theory in the analysis of autonomous nonlinear systems, in this chapter, we present new algorithms for computing approximations of the largest robust control invariant set of general constrained time-invariant discrete-time uncertain nonlinear systems. The first algorithm computes an outer approximation and the second algorithm computes an inner approximation of the largest RCIS. In contrast to some existing methods, we do not assume polynomial dynamics, nor require contractivity nor require prior knowledge of the structure of the set. More importantly, the proposed algorithms yield inner and outer approximations of the largest robust control invariant set for a sufficiently high precision. By combining our derived robust control invariant condition with the subdivision technique, our algorithm shows higher efficiency as compared to the grid-based methods. Furthermore, we propose an approach similar to feedback linearization of nonlinear systems to further reduce the computational load.

2.2 Preliminaries

2.2.1 Notation

Throughout this chapter, $\mathbb{Z}$ denotes the set of integers $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$. $\mathbb{Z}_+$ denotes the set of non-negative integers $\{0, 1, 2, \ldots\}$. $\{z_k\}_{k\in\mathbb{Z}_+}$ denotes an ordered set of numbers according to $k \in \mathbb{Z}_+$ $\{z_0, z_1, z_2, \ldots\}$. $\mathbb{B}$ denotes the unit ball in $\mathbb{R}^n$ with respect to the infinity

norm. The operator $|\cdot|$ denotes the Euclidean norm of a vector. The Minkowski set addition of two sets $P, Q \subseteq \mathbb{R}^n$ is defined as $Q + P = \{p + q \in \mathbb{R}^n | q \in Q,\ p \in P\}$. We slightly abuse notation and use $x + Q$ instead of $\{x\} + Q$ to denote the Minkowski set addition of a point x and a set Q. A directed graph is denoted as $G = (V, E)$ with V denoting the set of vertices of the graph and E denoting the set of ordered pairs of vertices known as edges. A function $f : X \to X$ is said to be homeomorphic in X if it is continuous with continuous inverse in X.

2.2.2 System description and problem formulation

In this chapter, we consider a class of discrete-time nonlinear systems that can be described by the following model:

$$x^+ = f(x, u, w) \tag{2.1}$$

where $x^+ \in \mathbb{R}^n$ denotes the state at the next sampling time, $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ represents the control input and $w \in \mathbb{R}^n$ denotes the unknown disturbance input. We consider that the state, control and disturbance are subject to the following constraints:

$$x \in X \subseteq \mathbb{R}^n,\ u \in U \subseteq \mathbb{R}^m \text{ and } w \in W \subseteq \mathbb{R}^n \tag{2.2}$$

Throughout our discussion, we make the following assumptions:

Assumption 2.1 (Compactness of constraints)**.** *The sets X, U and W are compact metric spaces. In addition, we assume that U and W have the origin in their interiors.*

Assumption 2.2 (Smoothness of system)**.** *The function $f : X \times U \times W \to X$ is a sufficiently smooth vector field in X. In addition, we assume that for each $x \in X$, $u \in U$ and $w \in W$, $f(x, u, w)$ is uniquely defined.*

We also recall the following definitions on forward invariant set and robust control invariant set:

Definition 2.1 (Forward invariant set [9]). *A set $R \subseteq X$ is said to be a forward or positively invariant set of the system $x^+ = f(x)$ if for every $x \in R$, $f(x) \in R$.*

Definition 2.2 (Robust control invariant set [9]). *A set $R \subseteq X$ is said to be a robust control invariant set (RCIS) for system (2.1) and constraint set (2.2) if for every $x \in R$, there exist a feedback control law $u = \mu(x) \in U$ such that R is forward invariant for the closed-loop system $f(x, u, w)$ for all $w \in W$.*

In the absence of disturbances, the robust control invariant set R, falls back to the usual notion of control invariant set. In this work, we are interested in determining the largest robust control invariant set contained in X. This set is also known as the maximal RCIS, strongly reachable set [28] or discriminating kernel [47]. We denote by $R(X)$ the largest RCIS of system (2.1) with constraints (2.2).

2.2.3 Graph construction

In this section, we briefly present the notion of symbolic image of a dynamical system and its construction for autonomous systems. For a more detailed discussion, the reader may refer to Chapters 2 and 5 of [45].

Consider a discrete-time autonomous system as follows:

$$x^+ = \hat{f}(x) \tag{2.3}$$

where $\hat{f} : X \to X$ is a homeomorphism on a compact domain $X \subseteq \mathbb{R}^n$.

A symbolic image of (2.3) is a finite approximation of the dynamics of the system using a directed graph. To construct a symbolic image, the state space X is quantized with the help of a finite covering, $\mathcal{C} = \{B_1, \ldots, B_l\}$, of the state space X. The finite covering $\mathcal{C}$ is a

collection of closed sets known as cells $B_i, i = 1, \dots l$, such that

$$X \subseteq \cup_{B_i \in \mathcal{C}} B_i \tag{2.4a}$$

$$B_i \cap B_j = \emptyset, \ \forall B_i, B_j \in \mathcal{C} \text{ with } i \neq j \tag{2.4b}$$

The diameter of the covering $\mathcal{C}$ is given by

$$\text{diam}(\mathcal{C}) := \max_{B_i \in \mathcal{C}} \text{diam}(B_i)$$

where $\text{diam}(B_i) = \sup\{|x-y|\colon x, y \in B_i\}$. Since X is compact, it is always possible to obtain a finite covering. We now introduce the symbolic image of $\hat{f}$ with respect to the covering $\mathcal{C}$.

Definition 2.3 (Symbolic image [45]). *Let G be a directed graph with l vertices where each vertex is a cell or box B_i in a finite covering $\mathcal{C}$ of the domain X of system (2.3). The vertices B_i and B_j are connected by a directed edge $B_i \to B_j$ if*

$$B_j \cap \hat{f}(B_i) \neq \emptyset$$

where $\hat{f}(B_i) := \{y | y = \hat{f}(x), x \in B_i\}$. The graph G is called a symbolic image of (2.3) with respect to the covering $\mathcal{C}$.

Definition 2.4 (Admissible path [45]). *A sequence $\{z_k\}_{k \in \mathbb{Z}_+}$ with each element z_k taking a value from the set of vertices of G is called an admissible path if for each $k \in \mathbb{Z}_+$, the graph G contains the edge $z_k \to z_{k+1}$.*

To understand the relationship between an admissible path on the symbolic image and the trajectories of system (2.3), we recall the notion of ε-orbit.

Definition 2.5 (ε-orbit [48]). *For a given $\varepsilon > 0$, a sequence of points $\{x_k\}_{k \in \mathbb{Z}_+}$ in X is called an ε-orbit of system (2.3) if for any $k \in \mathbb{Z}_+$*

$$|\hat{f}(x_k) - x_{k+1}| < \varepsilon$$

Due to round off errors in numerically computed trajectory of system (2.3), its real trajectory is rarely known in practice. Thus, a numerically computed trajectory of system (2.3) is usually no more than an ε-orbit for sufficiently small positive ε. There is therefore a natural correspondence between admissible paths on the symbolic image and the ε-orbits. That is, an admissible path on the graph G represents an ε-orbit of system (2.3) and vice versa. Specifically, if the sequence $\{z_k\}_{k\in\mathbb{Z}_+}$ is an admissible path on the symbolic image G, then there exist a sequence $\{x_k, x_k \in z_k\}_{k\in\mathbb{Z}_+}$ that is an ε-orbit of system (2.3) such that the following inequality hold

$$|\hat{f}(x_k) - x_{k+1}| \leq \operatorname{diam}(z_{k+1}) < \varepsilon$$

It is obvious that the finer the covering, the more precise the approximation of the system trajectories.

Definition 2.6 (Out-degree of a vertex)**.** *The out-degree of a vertex in a directed graph is the number of edges going out of the vertex.*

If a vertex (B_i) of the symbolic image of system (2.3) has zero out-degree, then its image $\hat{f}(B_i)$ has no intersection with any other vertex on the symbolic image. i.e. $\hat{f}(B_i) \cap X = \emptyset$. Therefore its image $\hat{f}(B_i)$ lies outside X. This implies that any trajectory starting from the cell will exit the state constraint X in finite time. An admissible path on the symbolic image G may either be finite or infinite. An admissible path on the symbolic image is finite if it ends with a vertex that has zero out-degree. Otherwise, it is infinite.

The construction of the symbolic image is depicted in Example 2.1.

Example 2.1 (Construction of the symbolic image)**.** *Let* $X = \{x \in \mathbb{R}^2 : \|x\|_\infty \leq 2\}$ *and consider the autonomous two dimensional system defined by*

$$x^+ = \hat{f}(x) = \begin{bmatrix} x_1 + 1.5 \\ x_2 + 1.5 \end{bmatrix}$$

To construct the symbolic image of the system in Example 2.1, the state constraint X

is first quantized. Quantization of X is not unique. One of such quantization is to use 16 cells of unit length to obtain the finite covering $\mathcal{C} = \{B_1, \ldots, B_{16}\}$ of X. Since X satisfies Assumption 2.1, this is possible. Thereafter, the image of each cell is obtained and used to construct the symbolic image. The quantized X is shown in Figure 2.1(a). The shaded cells $(B_6, B_7, B_{10}, B_{11})$ in Figure 2.1(a) are the image of cell B_{13} i.e. they have an intersection with $\hat{f}(B_{13})$. The resulting graph is shown in Figure 2.1(b). The existence of a directed edge between cells B_{13} and B_6 implies there exist an admissible path between the two cells. Cells whose image have no intersection with any other cells will have no outgoing edges in the symbolic image i.e. out-degree will be zero. The symbolic image is constructed by repeating this procedure for all other cells. The union of the resulting graphs form the symbolic image.

What remains is how the image of a cell can be approximated. One approach to approximate the image of a cell is to finitely sample points in the cell and then find the image of the points. The image of the cell is the union of the image of each sampled point in the cell. Different sampling methods that can be used are shown in Figure 2.2. Another approach is the use of interval arithmetic [49]. This involves performing numerical computations on intervals rather than numbers. Interval arithmetic was used in [30] to compute one-step reachable sets.

2.2.4 Set invariance condition for autonomous systems

In the previous section, we described the construction of the symbolic image G of system (2.3) which is an approximation of its dynamics using directed graphs. In this section, we describe how the resulting directed graph can be investigated using graph theory to obtain an outer approximation of the largest forward invariant set for autonomous dynamical systems. We recall the terminology in graph theory:

Definition 2.7 (Strongly connected graph)**.** *A directed graph $G = (V, E)$ is said to be strongly connected if there is an admissible path in both directions between each pair of vertices of the graph.*

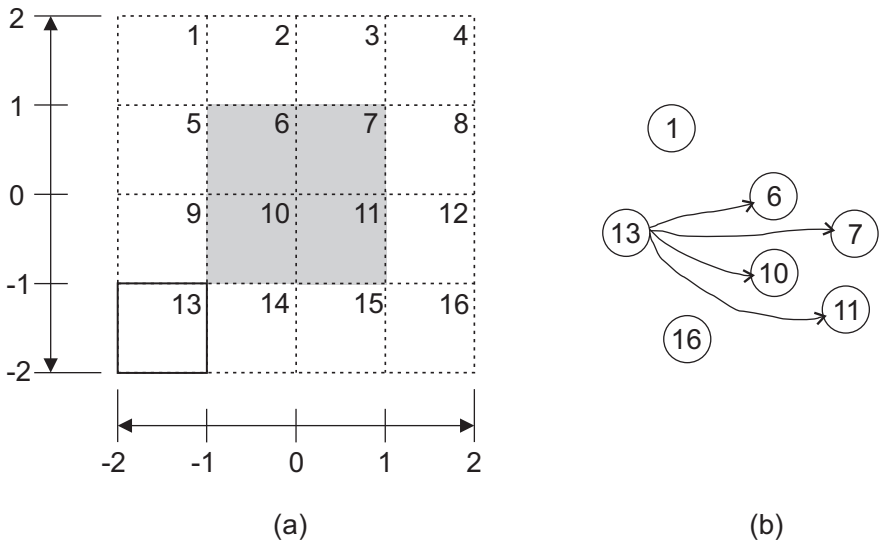


Figure 2.1: Construction of the symbolic image. (a) Image of B_{13} intersects with the shaded cells B_6, B_7, B_{10}, B_{11}. (b) Directed graph depicting the image of B_{13}.

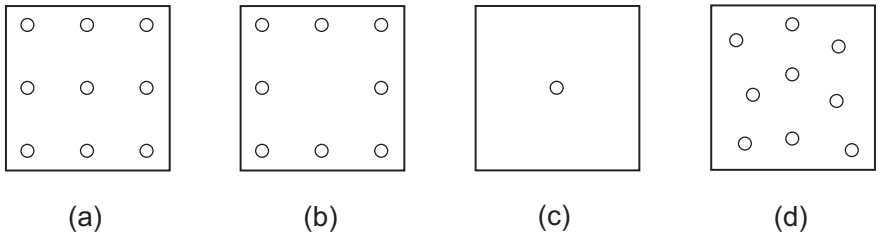


Figure 2.2: Different sampling types (a) Uniform sampling (b) Boundary sampling (c) Center sampling (d) Random sampling.

For a graph G that is not strongly connected, it may contain subgraphs that are strongly connected. These subgraphs are known as the strongly connected component subgraphs of G.

Definition 2.8 (Non-leaving cells)**.** *The set of vertices of the directed graph G with infinite admissible paths passing through them, denoted as $I^+(G)$, is the union of vertices of the largest strongly connected component subgraph of the directed graph G and any vertex of G that is not in the largest strongly connected components but has a path to a vertex in the largest strongly connected component subgraph.*

The following theorem summarizes how we may obtain an outer approximation of the largest forward invariant set of an autonomous system based on its symbolic image [45].

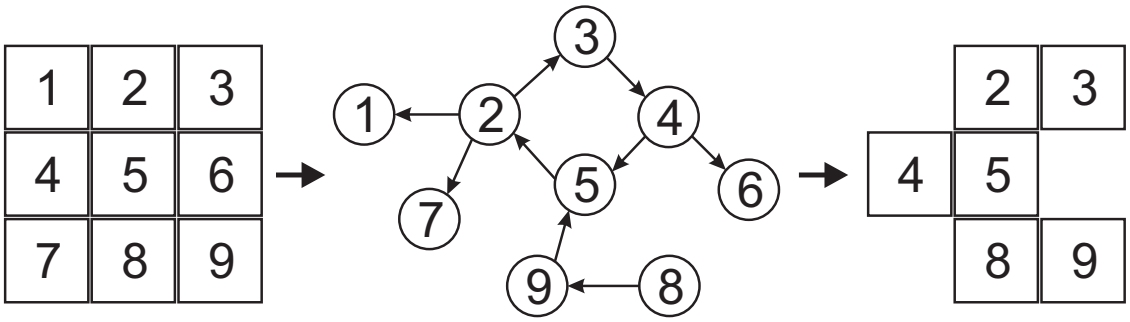


Figure 2.3: Example graph construction for a fictitious autonomous system and resulting invariant set. Left: Region in state space under study; Center: Approximation of the flow of the system using directed graph. Right: Selected cells which is an outer approximation of the forward invariant set.

Theorem 2.1. *Let $G = (V, E)$ having a set of vertices V and a set of ordered pairs of vertices E be a symbolic image of the mapping $\hat{f}$ in (2.3) with respect to a finite covering $\mathcal{C}$ of X. Then*

i. the vertices of the largest strongly connected component subgraph $G_s = (V_s, E_s)$ of G have infinite admissible paths passing through them.

ii. any element of V but not V_s with a path to at least one vertex of G_s also has an infinite admissible path passing through it.

iii. the union of the elements of (i) and (ii), $I^+(G)$, is a closed neighbourhood of the largest forward invariant set R of (2.3) in X. i.e.

$$R \subseteq I^+(G) \tag{2.5}$$

Fundamentally, Theorem 2.1 characterizes cells that have infinite admissible paths passing through them on the symbolic image. The central idea is illustrated in Figure 2.3 for a fictitious autonomous system. In the figure, cells B_2, B_3, B_4 and B_5 form the strongly connected components subgraph of the symbolic image since there exist an admissible path in both directions between each pair of vertices on the symbolic image (Theorem 2.1: i). Also, since there exist an admissible path from cells B_8 and B_9 to an element (Cell B_5) of

the the strongly connected components subgraph, they also have infinite admissible paths passing through them (Theorem 2.1: ii). Notice that Cell B_2 also has admissible path to Cells B_1 and B_7 which have zero outdegree. Similarly, there exist an admissible path from Cell B_4 to Cell B_6. Thus, both finite and infinite admissible paths pass through Cells B_2 and B_4. Therefore, the union of Cells B_2, B_3, B_4, B_5, B_8 and B_9 form a closed neighbourhood of the largest forward invariant set in the region of the state space under study (Theorem 2.1: iii). Since Cells B_2 and B_4 also have finite admissible paths passing through them, the set is not actually forward invariant and is merely an outer approximation of the largest forward invariant set contained in X. Finally, since Cells B_1, B_6 and B_7 have no outgoing edges (zero outdegree), any trajectory starting from them will exit the region of state space understudy in finite time and therefore do not form part of the approximation of forward invariant set.

We now proceed to present the main results of this work.

2.3 Main results

In the previous section, we demonstrated how forward invariant sets can be outer approximated for autonomous dynamical systems based on graph theory. In this section, we extend this result for constrained dynamical systems with controls and disturbances, and then use this result to develop efficient algorithms for computing robust control invariant sets. Recall that the dynamics of a system needs to be transcribed in a directed graph before analysis using graph algorithms. While the graph construction is straight forward to do for autonomous dynamical systems, it is not the case for dynamical systems with controls and disturbances (2.1). We show how the directed graph can be constructed for system (2.1) for our intended purposes as well as the robust control invariance condition based on the constructed graph.

We begin by presenting the directed graph construction and the robust control invariant set condition for constrained controlled systems with disturbances. Then, we present the

algorithm for computing the robust control invariant set. Finally, we present a way to reduce the computational load for input affine systems which are a special case of system (2.1).

2.3.1 Robust control invariance condition

Let us consider the set-valued map, also called parameterized map,

$$F(x, w) := f(x, U, w) = \{f(x, u, w)\}_{\cup_{u \in U}} \tag{2.6}$$

The map F associates with each state x and disturbance w the subset $F(x, w)$ of feasible next states. Therefore, system (2.1) defined by the family of parameterized difference equations is actually governed by the difference inclusion

$$x^+ \in F(x, w) \tag{2.7}$$

With this difference inclusion, all feasible trajectories of system (2.1) under every initial state and disturbance can be obtained. However, care must be taken when constructing the graph. If one naively constructs the symbolic image of system (2.1) by considering the union of all feasible trajectories of (2.7) for every initial state and disturbance, the graph obtained will not be suitable for our purposes. This is because the so-called "best-case" will be included. The "best-case" are instances where the disturbances aid the control inputs. They must be avoided in the graph construction as it does not provide guaranteed existence of a control law that will keep the system in the constraint set under every disturbance realization. Thus, to ensure guaranteed existence of a control law to keep the states within constraint set (2.2) at all times, not all trajectories must be allowed on the symbolic image which will be investigated. A pessimistic or worst-case view must therefore be adopted in the construction of the symbolic image. In short, the control seeks to enlarge the robust control invariant set while the disturbance seeks to make it smaller.

To address this, we construct individual graphs for each $w \in W$ using the difference

inclusion (2.7) and analyze them using Theorem 2.1. By taking an intersection of the resulting sets for every $w \in W$, we ensure that the disturbance always act against the control inputs. The resulting set obtained from the intersections, outer approximates the largest robust control invariant set. Let us define $G_w = (V_w, E_w)$ as the symbolic image of $F(X, w)$ with respect to the finite covering $\mathcal{C}$ of X and the constraint sets (2.2). Also, let

$$K := \cap_{w \in W} I^+(G_w) \tag{2.8}$$

The set K represents the cells with infinite admissible paths passing through them irrespective of the disturbance realization. The following theorem characterizes an outer approximation of the robust control invariant set of system (2.1) with constraint set (2.2).

Theorem 2.2. *Let K be defined as in Equation 2.8 above. If Assumptions 2.1 and 2.2 hold, then the set K is a closed neighbourhood of the largest robust control invariant set $R(X)$ of system (2.1) with constraint sets (2.2) i.e.*

$$R(X) \subseteq K \tag{2.9}$$

Proof. First we prove by contradiction that infinite admissible paths pass through the cells in K. Assume there exists a state $x \in R(X)$ but $x \notin K$. Then by definition of K, it implies that there exist Cells $B_i \subseteq X$ containing x without any infinite admissible path passing through it and $B_j \subseteq X$ with no outgoing edge (zero outdegree) such that there exist a disturbance sequence $\{w_0, \ldots, w_k\}$ for every input sequence $\{u_0, \ldots, u_k\}$ such that the finite sequence $\{z_0, \ldots, z_k\}$ with $z_0 = B_i$ and $z_k = B_j$ exist. This implies that the trajectory of x will eventually escape from X. This however, contradicts the assumption that $x \in R(X)$. Now we show that the set K is a closed neighbourhood of $R(X)$. Since an admissible path on the symbolic image is an ε-orbit and every cell in K is closed, the set K is a closed neighbourhood of $R(X)$. □

Theorem 2.2 characterizes the cells whose union forms an outer approximation of the robust control invariant set of system (2.1) and constraint sets (2.2) given a quantized state constraint set. This is the central idea we use in designing an algorithm for computing the largest robust control invariant set contained in X.

2.3.2 Computation of robust control invariant set

In this section, we present the algorithm for computing the robust control invariant set of system (2.1) subject to constraint set (2.2) and prove its convergence to the maximal RCIS. The algorithm is combined with the subdivision process (see [50] for more discussion) to improve the computational efficiency. This is realized using a family of finite coverings of the RCIS starting from the state constraint. In each step, a set of cells are selected according to Theorem 2.2 and then subdivided while the remainder are discarded. The algorithm is achieved in three main steps namely subdivision, graph construction and selection. Considering the k-th iteration in the proposed algorithm, the operations are outlined below.

- In the subdivision step, a finer covering of the RCIS is generated by dividing the current cells along one of the dimensions. If $\hat{\mathcal{C}}_{d_k}$ and $\mathcal{C}_{d_{k-1}}$ are coverings of the RCIS where d_k and d_{k-1} denote their respective diameters, then $d_k > d_{k-1}$ and

$$\cup_{B \in \hat{\mathcal{C}}_{d_k}} B = \cup_{B \in \mathcal{C}_{d_{k-1}}} B$$

  The set that is subdivided does not change other than have cells with smaller diameter. In each iteration of the algorithm, the dimension along which the cells are divided is cycled. The function *subdivide*() is used to compute the subdivision.

- Following the subdivision step, the graph construction step is conducted. This is achieved by creating a collection of graphs $G_k = \{G_w^k = (V_w^k, E_w^k)\ \forall w \in W\}$ with

Algorithm 1: Computing maximal robust control invariant set

Input: System (2.1), constraint sets (2.2) and maximum number of iterations N
Output: largest robust control invariant set

$\mathcal{C}_{d_0} \leftarrow X$ `// Initialization`
for $k \leftarrow 1, 2, 3, \cdots, N$ **do**
  if $\mathcal{C}_{d_k} = \emptyset$ **then**
    $\mathcal{C}_{d_N} \leftarrow \mathcal{C}_{d_k}$
    break
  if $\mathcal{C}_{d_k} = \mathcal{C}_{d_{k-1}}$ **then**
    $\mathcal{C}_{d_N} \leftarrow \mathcal{C}_{d_k}$
    break
  $\hat{\mathcal{C}}_{d_k} \leftarrow subdivide(\mathcal{C}_{d_k})$
  $G_k \leftarrow graph(\hat{\mathcal{C}}_{d_k})$
  $\mathcal{C}_{d_{k+1}} \leftarrow select(G_k)$
return $\mathcal{C}_{d_N}$

respect to the covering obtained from the subdivision step $\mathcal{C}_{d_k}$ where

$$V_w^k = \hat{\mathcal{C}}_{d_k} \; and$$
$$E_w^k = \{(B_i, B_j) \in \hat{\mathcal{C}}_{d_k} \times \hat{\mathcal{C}}_{d_k} \mid F(B_i, w) \cap B_j \neq \emptyset\}$$

This is realized in the algorithm as the function $graph()$.

- Finally, the selection step involves the selection of the set of cells that have infinite paths passing through them irrespective of $w \in W$ using the robust control invariant set condition in Theorem 2.2. i.e.

$$\mathcal{C}_{d_k} = \{B \in \hat{\mathcal{C}}_{d_k} \mid B \in \cap_{G \in G_k} I^+(G)\}$$

The cells that are not selected are discarded while the selected ones goes on to the next iteration. This is represented in the algorithm as the function $select()$.

The complete algorithm is summarized in Algorithm 1 below:

Algorithm 1 is initialized using the state constraint X. This not only ensures that the

domain understudy is restricted to X, but also ensures that the state constraints are enforced. Also, since Equation (2.7) is used during the graph construction step, the input and the disturbance constraint sets need to be finitely sampled for numerical implementation of the algorithm. An alternative is to treat the input and disturbance sets as intervals and use interval arithmetic for the computations. Notice that the computational load of the algorithm depends heavily on the number of cells generated at each iteration. This grows exponentially as the algorithm progresses. Similar to other algorithms for numerically computing invariant sets, there is a trade off between computational load and accuracy.

In what follows, we prove the convergence of the algorithm to the largest robust control invariant set.

2.3.3 Convergence of algorithm

We now prove that Algorithm 1 always converge to the largest RCIS $R(X)$ provided k goes to infinity. Let us denote by R_k the collection of closed sets after the k-th iteration of Algorithm 1. i.e.

$$R_k = \cup_{B \in \mathcal{C}_{d_k}} B$$

A quick observation is that Algorithm 1 generates a nested sequence $\{R_k\}$ of compact sets with $R_k \subseteq R_{k-1}$ due to the continuity of system (2.1). We can therefore observe that the N-th output from the algorithm is given by

$$R_N = \cap_{k=0}^{N} R_k \tag{2.10}$$

and we may write

$$R_\infty = \cap_{k=0}^{\infty} R_k \tag{2.11}$$

as the limit set of the algorithm. Our goal is to show that the robust control invariant set $R(X)$ is a subset of R_∞ and vice versa.

We first begin by showing that the sets R_k contain the robust control invariant set.

Lemma 2.1. *Consider system (2.1) with constraint sets (2.2). If Assumptions 2.1 and 2.2 hold, then the sets R_k obtained at the k-th iteration of Algorithm 1 contain the largest robust control invariant set. i.e.*

$$R(X) \subseteq R_k, \ \forall k \in \mathbb{Z}_+$$

Proof. Obviously, we know that $R(X) \subseteq X = R_0$. It also follows from Theorem 2.2 that $R(X) \subseteq R_k \subseteq R_0$. Therefore $R(X) \subseteq R_k \ \forall k \in \mathbb{Z}_+$. □

We now show that the limit set of Algorithm 1 is robust control invariant.

Lemma 2.2. *Consider system (2.1) with constraint sets (2.2). If Assumptions 2.1 and 2.2 hold, then the limit set R_∞ obtained from Algorithm 1 is robust control invariant.*

Proof. Recall that an admissible path on a symbolic image represents an ε-orbit of system (2.1). From the construction of Algorithm 1, we have that $d_k \to 0$ as $k \to \infty$. As a consequence of the weak shadowing property of an admissible path on the symbolic image (see [45], Theorem 14, 2) and the continuity of system (2.1), $\varepsilon \to 0$. Following similar arguments in [45] (Theorem 41, 2), we have that as $\varepsilon \to 0$ every admissible path approach the true path. Now we prove by contradiction. Suppose there exists an $x \in R_\infty$ such that there exist $w \in W$ for every $u \in U$ such that $f(x, u, w) \notin R_\infty$. This implies that only finite admissible paths pass through x. But this is impossible by the construction of Algorithm 1 since infinite admissible paths pass through every $x \in R_\infty$, and we have obtained the desired contradiction. □

Finally by combining Lemmas 2.1 and 2.2, we obtain the desired convergence result for Algorithm 1.

Theorem 2.3. *Let $R(X)$ be the largest robust control invariant set of system (2.1) with constraint sets (2.2). Consider the sequence $\{R_k\}_{k \in \mathbb{Z}_+}$ generated by Algorithm 1. If Assumptions*

2.1 and 2.2 hold, then

$$R(X) = R_\infty$$

Proof. To prove the above assertion, we need to show that the largest robust control invariant set $R(X)$ is a subset of R_∞ and at the same time a superset of R_∞. Now we proceed with the proof.

First we show that $R(X)$ is a subset of R_∞. It immediately follows from Lemma 2.1 that $R(X)$ is contained in every R_k and therefore is also contained in R_∞. i.e.

$$R(X) \subseteq R_\infty$$

Next we show that R_∞ is a subset of $R(X)$. It follows from Lemma 2.2 that the compact set R_∞ is robust control invariant and therefore must be contained in $R(X)$. i.e.

$$R_\infty \subseteq R(X)$$

Since combining the two Lemmas gives

$$R_\infty \subseteq R(X) \subseteq R_\infty$$

This further implies that

$$R(X) = R_\infty,$$

which completes the proof. □

2.3.4 Inner approximation

Notice that by the construction of Algorithm 1, the sets R_k are outer approximations of the largest robust control invariant set contained in X (see Lemma 2.1). Therefore, though the convergence results show that $R(X)$ can be computed, in practice it is impossible to

infinitely go on with the construction of an arbitrary fine covering of X. Thus, we have that

$$R_k \subseteq R(X) + \varepsilon\mathbb{B} \tag{2.12}$$

at the k-th iteration of Algorithm 1. While the set obtained gives an idea of the location and structure of the robust control invariant set of system (2.1) contained X, in the context of control theory, the sets R_k obtained from Algorithm 1 are not very useful since they are not robust control invariant. Therefore, an inner approximation is desired for controller design purposes.

One approach often used in the dynamic programming type algorithm is to initialize the algorithm from a robust control invariant set and gradually enlarge it (see [23, 2]). An alternative approach is the use of contractive sets. The former is not applicable to our algorithm since the feasible trajectories of the system (2.1) is approximated within the state constraint set and the latter assumes that X must contain a convex λ-contractive set and therefore restrictive. Recall that we do not assume contractivity in our analysis.

We take an approach similar to the stopping criterion of [1] for obtaining the inner approximation. Thus to obtain an inner approximation using Algorithm 1, we modify system (2.1) to

$$x^+ = f(x, u, w) + \varepsilon\mathbb{B} \tag{2.13}$$

and show that there exist $k \in \mathbb{Z}_+$ such that

$$R_k^\varepsilon \subseteq R_{k+1}^\varepsilon + \varepsilon\mathbb{B} \tag{2.14}$$

where R_k^ε denotes the k-th set generated by Algorithm 1 with the graphs constructed using the modified system (2.13).

The algorithm for inner approximation is presented in Algorithm 2. The termination criterion ensures that an inner approximation is obtained.

Algorithm 2: Computing inner approximation of maximal robust control invariant set

Input: system (2.13), constraint sets (2.2) and ε
Output: Inner approximation of largest robust control invariant set

$\mathcal{C}_{d_0} \leftarrow X$ `// Initialization`
for $k \leftarrow 1, 2, 3, \cdots$ **do**
  if $\mathcal{C}_{d_k} = \emptyset$ **then**
    $\mathcal{C}_{d_N} \leftarrow \mathcal{C}_{d_k}$
    break
  if $\mathcal{C}_{d_{k-1}} \subseteq \mathcal{C}_{d_k} + \varepsilon\mathbb{B}$ **then**
    $\mathcal{C}_{d_N} \leftarrow \mathcal{C}_{d_k}$
    break
  $\hat{\mathcal{C}}_{d_k} \leftarrow subdivide(\mathcal{C}_{d_k})$
  $G_k \leftarrow graph(\hat{\mathcal{C}}_{d_k})$
  $\mathcal{C}_{d_{k+1}} \leftarrow select(G_k)$
return $\mathcal{C}_{d_N}$

In what follows, we show that the output of Algorithm 2 is an inner approximation of the largest robust control invariant set.

Lemma 2.3. *Consider system (2.13) and constraint sets (2.2). Let* $\{R_k^\varepsilon, k \in \mathbb{Z}_+\}$ *be a sequence obtained from Algorithm 2. If Assumptions 2.1 and 2.2 hold, then for any* $\varepsilon > 0$*, there exist* $k \in \mathbb{Z}_+$ *so that (2.14) holds.*

Proof. From Lemma 2.1 we know that the sets $\{R_k^\varepsilon\}$ obtained from Algorithm 1 form a closed neighbourhood of $R(X)$ and therefore there exist $k' \in \mathbb{Z}_+$ such that for all $k \geq k'$, we have that $R_k^\varepsilon \subseteq R(X) + \varepsilon\mathbb{B}$. It then follows that $R_k^\varepsilon \subseteq R_{k+1}^\varepsilon + \varepsilon\mathbb{B}$. □

We now show that if the stopping criterion is met in Algorithm 2, then the output of the algorithm is an inner approximation of the robust control invariant set. Note that we do not consider the case where the largest robust control invariant is empty since this is trivial.

Theorem 2.4. *Consider system (2.13) and constraint sets (2.2) and let* $k' \in \mathbb{Z}_+$ *be the smallest index so that (2.14) hold for some* $\varepsilon > 0$*. If Assumptions 2.1 and 2.2 hold, then for any* $k \geq k'$*, the set* R_{k+1}^ε *is robust control invariant.*

Proof. Let $x \in R^{\varepsilon}_{k+1}$ such that $k \geq k'$. Then for all $w \in W$ there exist $u \in U$ such that

$$f(x, u, w) + \varepsilon\mathbb{B} \subseteq R^{\varepsilon}_{k} \subseteq R^{\varepsilon}_{k+1} + \varepsilon\mathbb{B}$$

This implies that $f(x, u, w) \subseteq R^{\varepsilon}_{k+1}$ and therefore R^{ε}_{k+1} is robust control invariant. □

Notice that if the disturbance is affine in w, then the modification in (2.13) is equivalent to increasing the disturbance set W by ε i.e. $W_{\varepsilon} \subseteq W + \varepsilon\mathbb{B}$.

2.3.5 Algorithm complexity

As stated earlier, computing invariant sets even for linear systems is not easy. In this section, we briefly discuss on the complexity of the proposed algorithms. In particular, we focus on how the complexity of the algorithm increases with the dimensions of the system state n, input m and disturbance n. At each iteration of the proposed algorithms, three main operations are conducted namely: subdivision, graph construction and cell selection. Let n_c be the number of cells at iteration k, n_u and n_w be the number of discrete points selected in each dimension of the input and disturbance respectively, and n_s be the number of sampling points per cell. The complexity of the subdivision step is $\mathcal{O}(n_c)$ since each cell is visited once and subdivided. The number of cells then becomes $2n_c$. After the subdivision step, the directed graph is constructed for each disturbance point and then investigated to determine the cells that approximate the robust control invariant set. The construction step require that one step prediction be computed. This leads to a complexity of $\mathcal{O}(2n_c n_s n_u^m n_w^n)$. The selection step require that the cells be separated into cells with infinite admissible paths passing through them and cells without infinite admissible paths passing through them. This can be determined using a combination of strongly connected component (SCC) algorithm and finding the cells with a path to the SCC. This leads to a linear time complexity i.e. $\mathcal{O}(|V|+|E|)$ where $|V|$ is the number of vertices of the directed graph and $|E|$ is the number of edges. Since the directed graph analysis is computed n_w^n times, the complexity of the

selection step is $\mathcal{O}(n_w^n(|V|+|E|))$. As can be seen from the above analysis, the slowest step of the algorithm is determined by the graph construction. If interval arithmetic is used in the computation instead of finite sampling of the input set and cells, the complexity of the graph construction algorithm becomes $\mathcal{O}(2n_c n_w^n)$. Furthermore, if only additive disturbance is present in the system model, then the disturbance discretization can be limited to the vertices of the disturbance set and the origin. This leads to a complexity of $\mathcal{O}(2n_c(2^n+1))$. As mentioned earlier, the number of cells n_c has a tendency to grow quickly, especially as the dimension of the state increases. Adaptive subdivision is one way to prevent the number of cells from growing quickly. To further speed up the computation, the subdivision and graph construction steps of the algorithm can be parallelized.

2.3.6 Special case: Input and disturbance affine systems

The algorithm as presented require the construction of several graphs of system (2.1) and constraint sets (2.2) under different disturbance realization. The computational burden for constructing the symbolic images may be excessive. Therefore in this section, we provide a simple method to reduce the computational load. The key idea is to cancel some or all of the disturbances by employing concepts from feedback linearization of nonlinear systems. We do not assume that the disturbances are known or measured other than the system equation must have a particular structure which can be exploited.

To achieve this, we require that the discrete-time uncertain control system (2.1) to have the following structure

$$x^+ = f(x) + g(x)u + h(x)w \tag{2.15}$$

where $f(\cdot) \in \mathbb{R}^n$, $g(\cdot) \in \mathbb{R}^n \times \mathbb{R}^m$ and $h(\cdot) \in \mathbb{R}^n \times \mathbb{R}^n$. This is possible if system (2.1) is affine in the input and the disturbance. If the state equation takes the form (2.15), then we can cancel out some or all disturbances via the equation

$$u = -g(x)^{-1}h(x)w + v \tag{2.16}$$

to obtain the transformed equation

$$x^+ = f(x) + g(x)v \tag{2.17}$$

The caveat to using this approach is that the state dependent matrix $g(\cdot)$ must be non-singular for every $x \in X$. If $g(\cdot)$ is non-singular, then the bounds on v can be obtained from Equation (2.16). Notice that if $g(\cdot)$ is independent of the state i.e. constant, then the bounds on v remain constant and therefore can be determined offline prior to the start of the algorithm. However, if $g(\cdot)$ or $h(\cdot)$ is state dependent, then the bounds on v may vary for every $x \in X$ and hence has to be determined online. This ensures that the use of this technique does not lead to conservative results. Unfortunately matrix inversion may not be cheap.

In cases where the $g(\cdot)$ matrix is non-square, it is possible to make it square by introducing arbitrary inputs.

Proposition 2.1. *The following constrained systems are equivalent:*

(i) $x^+ = f_1(x) + g_1(x)u + h_1(x)w$ *with* $x \in X$, $u \in U$ *and* $w \in W$

(ii) $x^+ = f_1(x) + g_1(x)u + g_2(x)u_a + h_1(x)w$ *with* $x \in X$, $u \in U$, $u_a \in \{0\}$ *and* $w \in W$

Proof. It is obvious that if $u_a = 0$ for all $t \in \mathbb{Z}_+$, then $g_2(\cdot)$ vanishes and has no effect on the dynamics of (ii). Hence, the two systems are equivalent. □

Proposition 2.1 shows that arbitrary inputs can be added to a difference equation without affecting the dynamics so long as it is constrained to the origin. This makes it possible to alter the structure of $g(\cdot)$ to a square matrix if it is not already a square. This is demonstrated using a linear system in Section 2.4.

2.4 Illustrative examples

In this section we conclude our study by demonstrating the efficacy of our algorithm on two problems. Both algorithms were implemented in the numerical computation language Julia [51]. In the first example, we use a simple two-dimensional linear model to show how feedback linearization-like concepts can be used to reduce the computational load. To demonstrate the correctness of our algorithms, we also compare the inner and outer approximation of the invariant sets generated by our algorithm for the linear system with the algorithm presented in [1]. Our choice for the algorithm in [1] stems from the fact that it is one of the well-known algorithms for computing the largest robust control invariant sets of constrained linear systems. In the second example, we apply our algorithm to the two-dimensional nonlinear model used in the work of [2]. This is to demonstrate the efficacy of our algorithms to that of [2]. The method in [2] is one of the recent results on the approximation of robust control invariant sets of constrained nonlinear systems.

In both examples, 10 points close to the edges of each cell were uniformly sampled and 5 points were also uniformly sampled in U. Also the vertices of the disturbance set were selected since both systems are affine in the disturbance. The maximum number of iterations N in Algorithm 1 and ε in Algorithm 2 are set at 16 and 0.001 respectively.

2.4.1 Example 1

Consider the linear system

$$x^+ = Ax + Bu + Gw$$

where A, B, and G are given by

$$A = \begin{bmatrix} 0.0 & 1.0 \\ 1.0 & 1.0 \end{bmatrix}, B = \begin{bmatrix} 0.0 \\ 1.0 \end{bmatrix} \text{ and } G = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix}.$$

The constraints on the states and input are $X = \{x \in \mathbb{R}^2 : \|x\|_\infty \leq 5\}$ and $U = \{u \in \mathbb{R} : \|u\|_\infty \leq 2\}$ respectively. The disturbance on the other hand is restricted to the set $W = \{w \in \mathbb{R}^2 : \|w\|_\infty \leq 0.3\}$.

As can be observed, B is a vector and therefore not invertible. However from Proposition 2.1, its structure can be altered without changing the dynamics by introducing arbitrary inputs. In this case the modified equation becomes

$$x^+ = Ax + \begin{bmatrix} 0.0 & 1.0 \\ 1.0 & 0.0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_a \end{bmatrix} + Gw \tag{2.18}$$

With this structural change, the system can be transformed to that of Equation 2.17 such that

$$v = \begin{bmatrix} u_1 \\ u_a \end{bmatrix} + \begin{bmatrix} 0.0 & 1.0 \\ 1.0 & 0.0 \end{bmatrix}^{-1} \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} w \tag{2.19}$$

Since $g(\cdot) = B$ is invertible for all $x(t) \in X$ and independent of the current state, the bounds on $v(t)$ can be obtained.

$$v = \begin{bmatrix} u_1 + w_2 \\ u_a + w_1 \end{bmatrix} \tag{2.20}$$

From the above equation, $|v_1| \leq 1.7$ and $|v_2| \leq 0.3$. Notice that v_2 is still a disturbance while v_1 is an input. With the transformed system having 1 disturbance, the graph construction reduces to 2 instead of 4. This cuts down the computation time by half.

We compare the results outer and inner approximations of the largest robust control invariant set obtained by Algorithms 1 and 2 respectively, to that of [1]. We allowed the latter to run for a sufficiently long time to ensure its as close as possible to the actual invariant set. As can be observed in Figure 2.4, our proposed algorithms are able to provide inner and outer approximations of the largest robust control invariant set. Also, as can be observed in Figure 2.5, Algorithms 1 and 2 give accurate outer and inner approximations the largest control invariant set in the absence of disturbances.

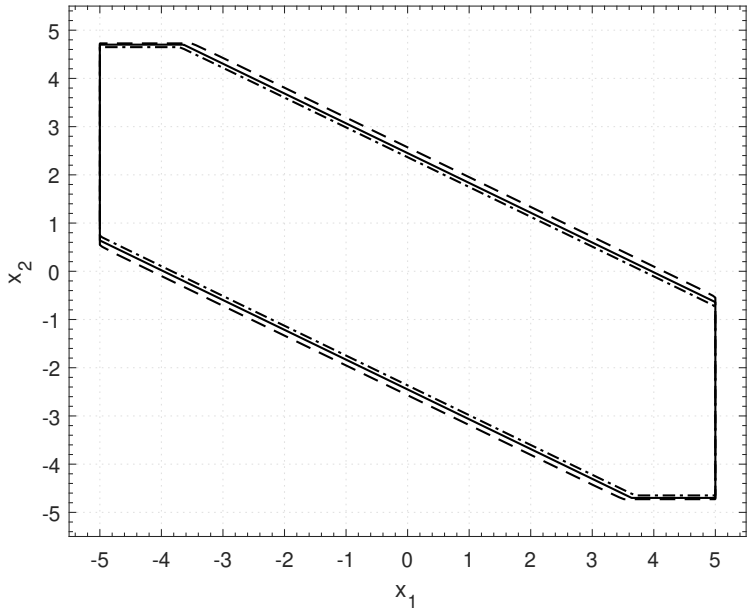


Figure 2.4: Comparison of inner (dashed dot) and outer approximations (dashes) obtained from Algorithm 1 and that of [1] (solid) for a two dimensional linear system. The invariant sets in the figure are obtained by finding the convex hull of the cells obtained from the Algorithms.

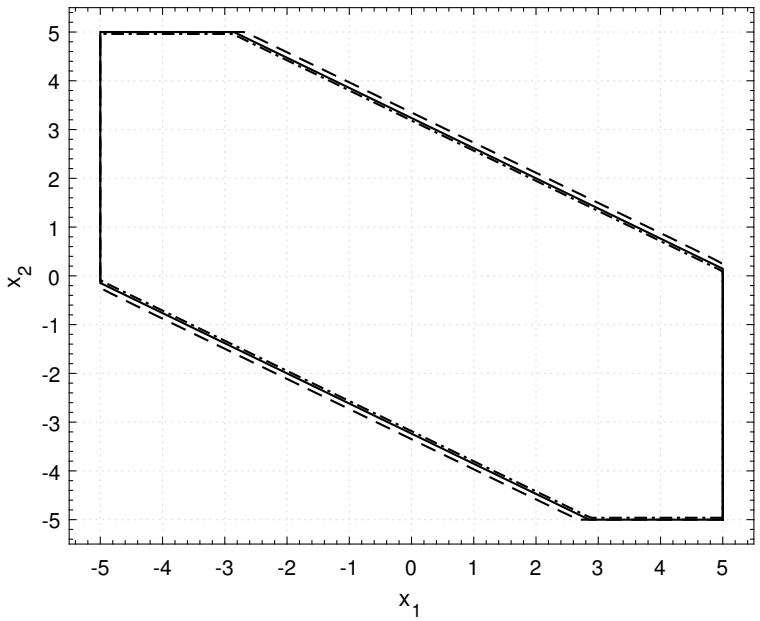


Figure 2.5: Comparison of inner (dashed dot) and outer approximations (dashes) obtained from Algorithm 1 and that of [1] (solid) for a two dimensional linear system without disturbances. The invariant sets in the figure are obtained by finding the convex hull of the cells obtained from the Algorithms.

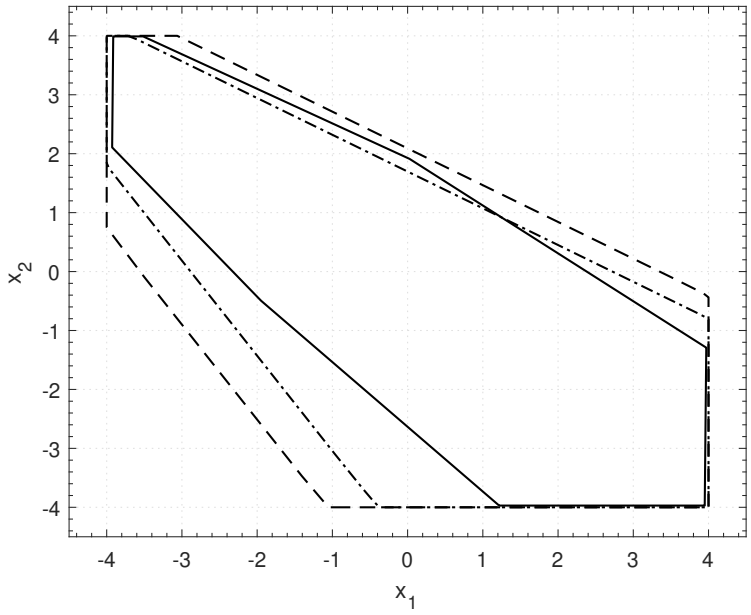


Figure 2.6: Comparison of inner approximations of control invariant set (dashes), robust control invariant set (dashed dot) obtained from Algorithm 2 and control invariant set obtained from [2] (solid) for a two dimensional nonlinear system. The invariant sets in the figure are obtained by finding the convex hull of the cells obtained in Algorithm 2.

2.4.2 Example 2

Consider the nonlinear system

$$\begin{bmatrix} x_1^+ \\ x_2^+ \end{bmatrix} = \begin{bmatrix} 1.0 & T \\ T & 1.0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + T \left\{ \mu \begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix} + (1-\mu) \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & -4.0 \end{bmatrix} x \right\} u + T \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

where $T = 0.01$, $\mu = 0.9$. The constraints on the states and input are $X = \{x \in \mathbb{R}^2 : \|x\|_\infty \leq 4\}$ and $U = \{u \in \mathbb{R} : \|u\|_\infty \leq 2\}$ respectively. The disturbance on the other hand is restricted to the set $W = \{w \in \mathbb{R}^2 : \|w\|_\infty \leq 0.4\}$. From the system equations, it can be seen that $g(x)$ is state dependent. Thus, applying the feedback-linearizing technique to cancel out some of the disturbances will require computing matrix inversion for each sampling point in a cell. This may be computationally excessive. To avoid this, the technique was not used in this example. Figure 2.6 shows the comparison between the inner approximation of the control invariant set obtained after the 16th subdivision from Algorithm 2 and that of the [2]. It

also shows the robust control invariant set obtained from Algorithm 2.

As can be seen in Figure 2.6, our Algorithm was able to compute a much larger control invariant set compared to that of [2].

2.5 Concluding remarks

Given a discrete-time time-invariant uncertain control system with bounded disturbances subject to state and input constraints, the methods presented in this chapter obtains inner and outer approximations of the largest robust control invariant set contained in the state constraint. For this purpose, we presented an algorithm which approximates the dynamics of the uncertain control system as a directed graph allowing for analysis of the system using graph theory. The results of this chapter, proving convergence to the largest robust control invariant set, are important in that they show the theoretical soundness of this approach. Simulations using linear and nonlinear systems demonstrate the effectiveness of the proposed method.

Chapter 3

An efficient implementation of graph-based invariant set algorithm for constrained nonlinear dynamical systems

3.1 Introduction

In Chapter 2, we successfully used the graph-based invariant set (GIS) algorithm to determine the largest control invariant sets of complex nonlinear dynamical systems [32]. More importantly, convergence to the largest CIS was also provided. In the GIS algorithm, the dynamics of the system is approximated with a directed graph and then analyzed to obtain an approximation of the largest CIS. However, the GIS algorithm, like other control invariant set algorithms for nonlinear systems, may require high computing resources. This limits the applicability of the algorithm to high dimensional systems.

In this chapter, we present details of an improved and efficient implementation of the GIS algorithm for computing control invariant sets of constrained dynamical systems. Obviously,

only the boundary of the CIS is of interest during the computation. This is the central idea we employ in the improved GIS algorithm. Our approach involves an adaptive subdivision technique to slow down the cell growth rate, and parallelization of the graph construction with multicore processing and graphics processing units (GPU). The adaptive subdivision technique makes use of a supervised machine learning technique to select the cells for subdivision. We compare the results obtained by the improved GIS algorithm to that of the standard GIS algorithm using a nonlinear example.

3.2 Preliminaries

3.2.1 System description and problem formulation

In this chapter, we are concerned with discrete-time nonlinear systems of the form

$$x^+ = f(x, u) \tag{3.1}$$

where $x^+ \in X \subseteq \mathbb{R}^n$ denotes the state at the next sampling time, $x \in X \subseteq \mathbb{R}^n$ is the state, and $u \in U \subseteq \mathbb{R}^m$ represents the control input. The sets X and U denote the state and input constraints respectively. We assume that the sets X and U are compact, and the function $f : X \times U \to X$ is a sufficiently smooth vector field in X.

The goal is the compute the largest (with respect to inclusion) control invariant set R_X for system (3.1) and the associated constraint sets using the GIS algorithm described in Chapter 1. However, the algorithm in its present form may require high computational resources which eventually, slows down the computation speed. Thus, the goal of this chapter is to identify the bottlenecks of the GIS algorithm and present remedial strategies to address them.

3.2.2 Computational requirements of standard GIS algorithm

In this section, we briefly analyze the computational requirements for each step of the algorithm. When the standard GIS algorithm is used to compute the largest CIS, the computational requirements are directly proportional to the number of cells used to approximate the set. This implies that as the number of cells increase at each iteration, the algorithm gets slower while the memory needed goes higher. Thus, the algorithm is greatly influenced by the number of cells used to approximate R_X. Let n_c be the number of cells at iteration k of the algorithm. The subdivision step involves iterating through all the cells to divide each cell into two. This results in a linear time complexity, that is $O(n_c)$. The graph construction step involves (1) finding one step forward mappings of each cell, (2) finding which cells intersect the one step forward mapping and (3) constructing an edge list of the graph. Without going into the details of how the one step mapping is achieved for each cell, the time complexity is $O(n_c)$ since the one step forward mapping need to be created for each cell. Finding the cells that have an intersection section with the one step forward mapping involves using an R*-tree. The average case (because we do not have data overlaps) time complexity of querying the R*-tree for each cell is $O(\log\, n_c)$. Hence, the time complexity for finding the intersection of the one step forward mapping of all the cells is $O(n_c \log\, n_c)$. Finally, the complexity of creating the edge list of the digraph is $O(|E|)$ where $|E|$ is the number of edges on the digraph. The overall time complexity of the graph construction step is $O(n_c \log\, n_c + |E|)$. Analyzing the digraph involves finding the non-leaving cells. This has a time complexity of $O(|V|+|E|)$ where $|V|$ is the number of vertices which is equivalent to n_c. The time complexities of the major parts of the GIS algorithm as well as their significance are presented in Table 3.1.

Table 3.1: Major parts of the GIS algorithm and their computational requirements

Part	Time complexity	Significance
Cell subdivision	$O(n_c)$	Not significant
Graph construction	$O(n_c \log\, n_c + \|E\|)$	Significant
Graph analysis	$O(\|V\|+\|E\|)$	Not significant

3.3 The improved and efficient GIS algorithm

As mentioned earlier, the standard GIS algorithm approximates the largest control invariant set by iteratively refining the cells that cover R_X. In doing so, it may over refine cells which may be dynamically irrelevant to the computation. This is because in the subdivision step of the standard algorithm, each cell is divided. While this works in principle, it may generate high number of cells as the algorithm proceeds. This significantly slows down the algorithm, and lead to high memory storage and computational requirements as demonstrated in the preceding section. In this section, we propose a method to improve the computational efficiency of the algorithm by adaptively selecting a subset of the cells to be subdivided instead of subdividing all the cells. This reduces the cell growth rate and ultimately the overall computational requirements of the algorithm. In addition, we describe a parallel implementation of the graph construction step to speed up the standard algorithm using graphics processing unit (GPU).

We begin this section by first describing the adaptive subdivision method. Thereafter, we describe the parallelization of the graph construction step using GPU. Finally, we briefly discuss the implications of the modifications on the convergence of the GIS algorithm to R_X.

3.3.1 Adaptive cell subdivision

Theoretically, the part of the control invariant set of interest to us is its boundary. For linear systems with convex state and input constraints, the largest control invariant set is convex. This explains why for linear systems with convex constraints, it is sufficient to test for control

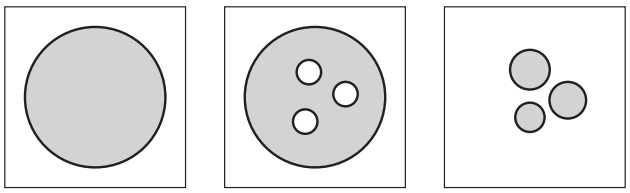


Figure 3.1: Types of control invariant set boundaries. The thick black lines represent the boundary of the set and the shaded portion represent the interior of the control invariant set. The box represent the region of the state space of interest X. Left: Control invariant set with continuous boundary; Middle: Control invariant set with pocket of holes creating a discontinuous boundary; Right: Multiple control invariant set in the search region.

invariance only at the vertices (boundary) as the iteration progresses. For nonlinear systems however, the largest control invariant set may not be convex. The set can be in any form or shape depending on the system dynamics. Moreover, there could be more than one invariant set in the region of interest. Hence, the search for the control invariant set involves searching everywhere within the state constraint. Figure 3.1 shows some examples of the shape of control invariant set that may be encountered in nonlinear systems. If the boundary area of the control invariant set contained in X is roughly known, then it suffices to just refine around the boundary area and use cells with bigger diameter for the interior. As an example, Figure 3.2 shows how the same set can be represented by different number of cells. One with uniform cell diameter and the other with non-uniform cell diameter. It can be seen that both sets have the same shape and size, albeit the number of cells. This is the central idea we use in the proposed adaptive algorithm.

The proposed algorithm seeks to alleviate the cell growth draw back by subdividing only a subset of the cells obtained after the selection step. The question remains what criteria to use to select the cells to be subdivided. Indeed, an earlier work on set-oriented methods for analysis of autonomous dynamical system used approximations of the Sinai-Bowen-Ruelle (SBR) measures as the criterion to adaptively select the cells for subdivision [52]. We however take a different approach since it could be equally challenging to compute the SBR measures.

Let $\mathcal{B}(\mathcal{C}_{d_k})$, $\mathcal{N}(\mathcal{B}(\mathcal{C}_{d_k}))$ and $\mathcal{I}(\mathcal{C}_{d_k})$ be the boundary, neighborhood of the boundary and interior cells respectively, at iteration k of the standard GIS algorithm. In line with the

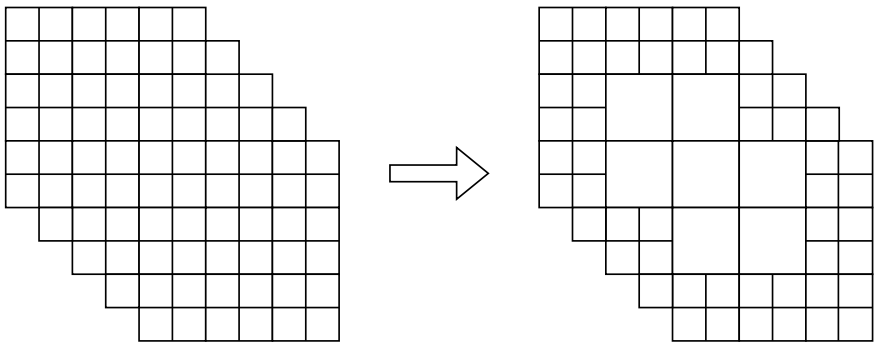


Figure 3.2: Representation of the same set with different number of cells. Left: Uniform cell subdivision as used in the standard algorithm. Right: Adaptive subdivision where the boundary is refined and the interior is not

earlier explanation, the proposed algorithm selects the boundary cells $\mathcal{B}(\mathcal{C}_{d_k})$ for subdivision. However, since the boundary of the largest control invariant set R_X or the region where it lies is not precisely known, the algorithm includes additional cells within the neighborhood of the boundary cells $\mathcal{N}(\mathcal{B}(\mathcal{C}_{d_k}))$. The selection of the boundary cells $\mathcal{N}(\mathcal{B}(\mathcal{C}_{d_k}))$ is based on the N-nearest neighbor (N-NN) supervised machine learning algorithm, and is controlled by a parameter N. This will be described later in this section. The adaptive subdivision step at each iteration of the algorithm involves three main steps. The first step involves selecting the boundary cells $\mathcal{B}(\mathcal{C}_{d_k})$. The second step involves selecting a neighborhood of the boundary cells $\mathcal{N}(\mathcal{B}(\mathcal{C}_{d_k}))$. Finally, both group of cells are subdivided.

We begin this section by first describing the procedure for selecting the boundary. Thereafter, we described how the neighborhood of the boundary cells are selected.

3.3.1.1 Boundary selection

The goal of the boundary selection step is to find $\mathcal{B}(\mathcal{C}_{d_k})$. Consider the cell $B_i \in \mathcal{C}_{d_k}$ at iteration k of the algorithm. The Cell B_i is first enlarged by a small factor δ. Afterwards, the vertices of the enlarged cell are selected. The idea is that, each vertex of the enlarged cell must intersect a neighboring cell if it is an interior cell $\mathcal{I}(\mathcal{C}_{d_k})$, otherwise it is a boundary

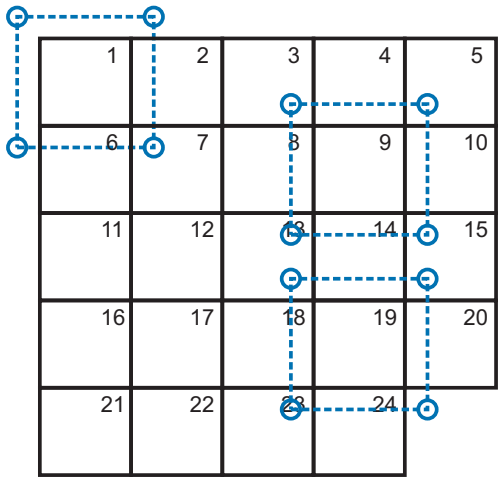


Figure 3.3: Graphical illustration of the process for selecting the boundary. The dashed blue lines represents the cell enlargement and the circles represent the vertices of the enlarged cells.

cell $\mathcal{B}(\mathcal{C}_{d_k})$. This is illustrated in Figure 3.3.

In Figure 3.3, Cells B_1 to B_{24} are the cells that constitute the cells to be subdivided $\mathcal{C}_{d_k}$. This implies that Cell B_{25} has been removed from the previous iteration of the algorithm. For illustration purposes, the cell enlargement is shown for Cells B_1, B_9 and B_{19} as the blue dashed rectangles. The vertices of the enlarged cells are selected after the enlargement. This is shown as the blue small circles. The expectation is that if a cell in an interior cell, then all four selected vertices must intersect a neighboring cell. For Cell B_1, it can be seen that only one vertex intersects a neighboring cell, that is Cell B_7. The other vertices do not intersect any neighboring cell. This makes Cell B_1 a boundary cell. Similarly for Cell B_{19}, three vertices of the enlarged cell intersect neighboring cells with one vertex not intersecting any cell. This implies that Cell B_{19} is also a boundary cell. Finally, following the same procedure for Cell B_9, it can be seen that all the vertices intersect its neighboring cells. This makes it an interior cell. A summary of the boundary selection algorithm is summarized in Algorithm 3.

Remark 3.1. *It is possible that a cell which is supposed to be a boundary cell is not selected. This is an edge case. While edge cases in the boundary selection process are not expected,*

Algorithm 3: Selection of boundary cells

Input: Cells to be subdivided $\mathcal{C}_{d_k}$, boundary cells $\mathcal{B}(\mathcal{C}_{d_k})$
Output: Boundary cells $\mathcal{B}(\mathcal{C}_{d_k})$

```
B(C_{d_k}) ← ∅ // Initialization
for B_i ∈ C_{d_k} do
    Enlarge the cell B_i
    Select the vertices of the enlarged cell
    if all the vertices do not intersect cells in C_{d_k} then
        Add B_i to the collection B(C_{d_k}) // B_i is a boundary cell
return B(C_{d_k})
```

they can occur. For example, in Figure 3.3, if Cell 4 is not present, then Cell 9 is supposed to be a boundary cell. However, Cell 9 will not be selected as a boundary cell because all the vertices of the enlarged cell satisfy the criterion for it to be an interior cell. In this case, the procedure can be modified such that points along the edges of the enlarged cell are also included. This may however impact the computational speed.

Moreover, the selection of the neighborhood of the boundary will automatically resolve such edge cases. This will be described shortly in the next subsection.

3.3.1.2 Selection of neighborhood of boundary cells

As mentioned earlier, since the location of the boundary of the largest control invariant set R_X is unknown in advance, a neighborhood of the boundary cells is also selected for subdivision. The implications of selecting or not selecting the neighborhood of the boundary cells will be demonstrated in the results section. Therefore, following the boundary cell selection, the neighborhood cells of each boundary cell are also selected for subdivision. This is achieved using the N-nearest neighbors (N-NN) of a point algorithm. N is a parameter which determines the number of neighboring cells to be selected. This ultimately determines how far from the boundary cells we want to move into the interior of the set.

N-NN is a supervised machine learning technique which is used to solve classification and regression problems. Given a point p, it selects the N neighboring points of p using the

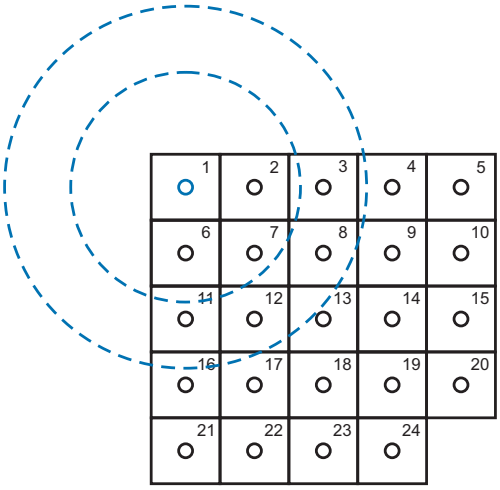


Figure 3.4: Graphical illustration of the process for selecting the neighborhood of the boundary cells. The center of the cells are indicated with the solid circles. The two dashed circles show the N-nearest neighbors of Cell B_1 for different N. The smaller dashed circle corresponds to an N of 3 while the larger dashed circle corresponds to an N of 7. Thus, for the smaller dashed circle, the three cells namely, B_2, B_6 and B_7 are selected. The Cells B_2, B_6, B_7, B_3, B_8, B_{12} and B_{11} are selected for the larger dashed circle.

distances from the point. This is implemented in the adaptive GIS as follows. First, the center of the cells are selected. Then for each boundary cell, the N-nearest neighboring points are selected. Figure 3.4 shows how the N-NN algorithm is used to select the N-neighborhood of the Cell B_1.

Remark 3.2. *The presence of dataset imbalance and outliers can significantly affect the N-NN algorithm. However, in this work we do not expect these situations to happen since the outliers are absent and there is no dataset imbalance.*

3.3.2 Efficient parallelization with GPU

Because of the complicated nature of the GIS algorithm, parallelization with GPU is not trivial. For example, if the parallelization is not done properly, it can lead to computational inefficiencies due to excessive communications between the processors and GPU. Several issues need to be addressed for efficient parallelization:

- Load-balancing: Parallelization cannot be achieved if the tasks to be completed are

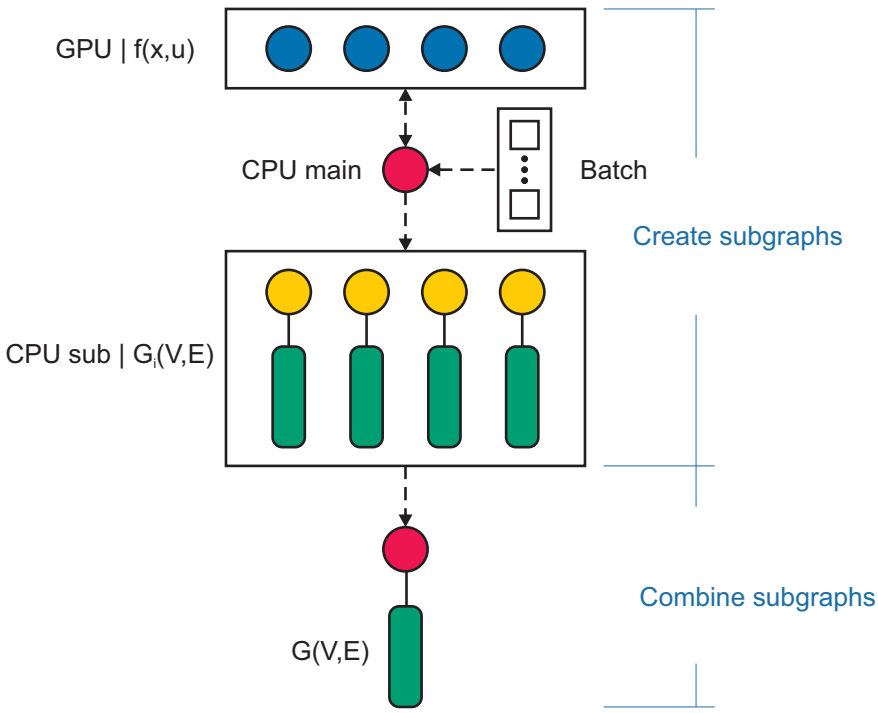


Figure 3.5: Coordination of CPU and GPU, and data flow in the parallelized graph construction step of the improved GIS algorithm

not distributed fairly among the compute cores. We use load-balancing to improve the parallelization in the graph creation step.

- Batching: The number of cells in the GIS algorithm can grow quickly. Loading and unloading data to and from the GPU can significantly degrade the benefits of using the GPU. We use batching to load a number of cells onto the GPU at time to maximize the use of the GPU.

Using appropriate batching and load-balancing schemes can significantly reduce the data traffic between the CPU cores and the GPU. In this section, we outline the details of the parallelization of the GIS algorithm.

In the GIS algorithm, one step forward mappings need to be computed for each cell. This can be done for each cell independently. Thus, the same instruction is used for each cell. This type of parallelization is known as data parallelization. GPUs are particularly suited for this kind of operation than CPUs. This is because GPUs have a highly parallel structure

which make them more efficient for algorithms that process large chunks of data in parallel. A typical GPU usage sequence involves

1. loading the data from the CPU to the GPU
2. performing the computation on the data
3. offloading the data from the GPU to the CPU

The main speed up when using the GPU is from the second step. Steps (1) and (3) are the main bottlenecks when using a GPU. Thus, frequent loading and offloading of data to and from the GPU can significantly overshadow the gains made in Step (2). To address this problem, we load the cells in batches. This is in contrast to the sequential GIS algorithm where the one step forward mapping is computed one cell at a time. By loading a number of cells at a time (batch), the communication frequency between the CPU and GPU is reduced significantly. The number of cells to load from the CPU to the GPU depends on the available memory on the GPU.

Let B_i^+ be the cells that have an intersection with $F(B_i)$, that is, $B_i^+ =: \{B_j | B_j \cap F(B_i) \neq \emptyset, B_j \in \mathcal{C}_{d_k}\}$. In the parallelized GIS algorithm, the main CPU passes a batch of n cells $B_i \dots B_{i+n}$ to the GPU which then computes and returns the images $F(B_i) \dots F(B_{i+n})$. Once the main CPU receives the data from the GPU, it creates and distributes the data across a number of subprocesses. Each subprocess finds the corresponding B_i^+ using the $F(B_i)$ information. To avoid race condition when each subprocess writes the edge data into the same graph, a subgraph $G_i(V, E)$ is created for each subprocess i. This continues until all the cells in $\mathcal{C}_{d_k}$ are exhausted. Then in the second step, the graphs are merged into a single graph $G(V, E)$. This is then passed to the graph analysis step for processing. Figure 3.5 shows the data flow and how the main CPU coordinates with the GPU and subprocesses.

3.3.3 Convergence issues

In this section, we briefly discuss the implications of the modifications to the GIS algorithm on the convergence of the sets to the largest control invariant set R_X. We note that parallelization of the graph construction step does not affect convergence to R_X in anyway. This is because other than speeding up the construction of the graph, the graph is not modified in any way. We therefore focus on the implication of the adaptive subdivision on the convergence to the algorithm to R_X.

In the adaptive subdivision technique, only a subset of the cells are subdivided. This is in contrast to the standard GIS algorithm where all the cells are subdivided at each iteration. However, we note that this modification does not affect the convergence of the algorithm other than the speed of convergence. As an illustration, let us assume the worst case scenario where the boundary of the largest control invariant set lies somewhere deep in the interior of the state constraint. If only the boundary cells are selected and subdivided, then the algorithm will spend majority of the time refining cells which do not contain the boundary of R_X. The algorithm will keep refining the irrelevant cells until they are sufficiently small, only to remove those cells before moving to the next boundary cells which are much coarser since they have not been subdivided. This continues until the boundary cells which contain the boundary of R_X are eventually located. While the cell growth is significantly reduces, the rate of convergence also reduces. This is certainly different from the standard GIS algorithm where all the cells are subdivided and therefore the boundary of R_X can be found much faster. The addition of the neighborhood of the boundary cells helps to balance the trade off between faster convergence and cell growth rate. This will be demonstrated in the results section.

3.4 Results

In this section, we test the effectiveness of the modifications to the standard graph-based control invariant set computation algorithm. First, we consider the effects of the adaptive subdivision modification on the standard algorithm. Then, we consider the impact of the parallelization. Both tests were performed using a nonlinear continuously stirred tank reactor example. In all these cases, the computations were run on workstation with the following configuration: a quadcore Intel i7-4720HQ CPU with frequency of 2.6 GHz, 16 GB of random access memory (RAM), and Nvidia GeForce GTX 960M GPU with 2 GB video RAM.

3.4.1 Process description

We consider a well-mixed continuously stirred tank reactor (CSTR) in which a first-order reaction of the form $A \rightarrow B$ takes place. Because the reaction is exothermic, a cooling jacket is used to remove excess heat from the reactor. Equation (3.2) describes the dynamics of the CSTR

$$\frac{dc_A}{dt} = \frac{q}{V}(c_{Af} - c_A) - k_0 \exp(-\frac{E}{RT})c_A \tag{3.2a}$$

$$\frac{dT}{dt} = \frac{q}{V}(T_f - T) + \frac{-\Delta H}{\rho c_p} k_0 \exp(-\frac{E}{RT})c_A + \frac{UA}{V\rho c_p}(T_c - T) \tag{3.2b}$$

where c_A and T are the reactant concentration and temperature of the reaction mixture in mol/L and K respectively, T_c is the temperature of the coolant stream in K, q denotes the volumetric flow rate of the inlet and outlet streams of the reactor in L/min, c_{Af} is the concentration of reactant A in the feed stream, V is the volume of the reaction mixture, k_0 denotes the reaction rate pre-exponential factor, E denotes the activation energy, R is the universal gas constant, ρ denotes the density of the reaction mixture, T_f denotes the temperature of the feed stream, c_p is the specific heat capacity of the reaction mixture, ΔH is the heat of reaction and UA is the heat transfer coefficient between the cooling jacket and the reactor. The parameters used in the simulations are presented in Table 3.2. The

Table 3.2: Table of parameter values

Parameter	Unit	Value
q	L/min	100.0
V	L	100.0
c_{Af}	mol/L	1.0
T_f	K	350.0
E/R	K	8750.0
k_0	min^{-1}	7.2×10^{10}
$-\Delta H$	J/mol	5.0×10^4
UA	$J/min \cdot K$	5.0×10^4
c_p	$J/g \cdot K$	0.239
ρ	g/L	1000.0

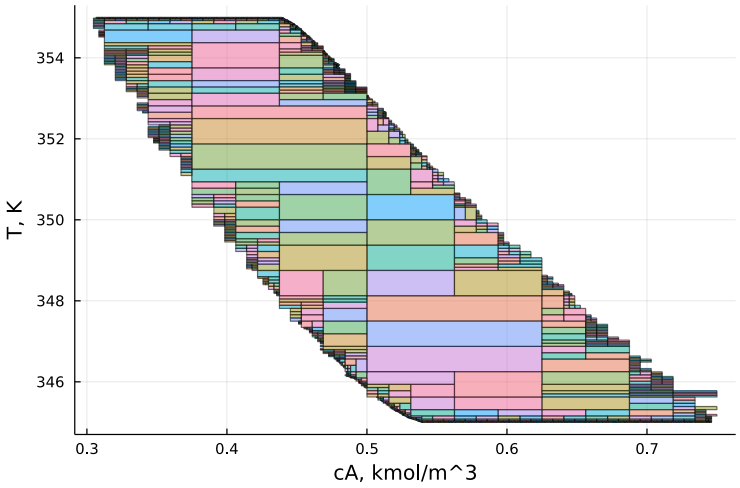


Figure 3.6: Sample plot of the cells after 20 iterations of the adaptive algorithm with $N = 0$

nonlinear model of (3.2) is discretized with a step size $h = 0.1\ min$ to obtain a discrete-time nonlinear state space model in the form of system (3.1). In this case $x = [c_A\ T]^T$ is the state vector and $u = T_c$ is the input. The state and input are constrained to be in the following sets: $0.0 \leq x_1 \leq 1.0$, $345.0 \leq x_2 \leq 355.0$, $285.0 \leq u \leq 315.0$.

3.4.2 Adaptive subdivision results

In this section we present the results of the adaptive subdivision without the parallelization. Figure 3.6 shows a sample output of the algorithm after 20 iterations with $N = 0$. This

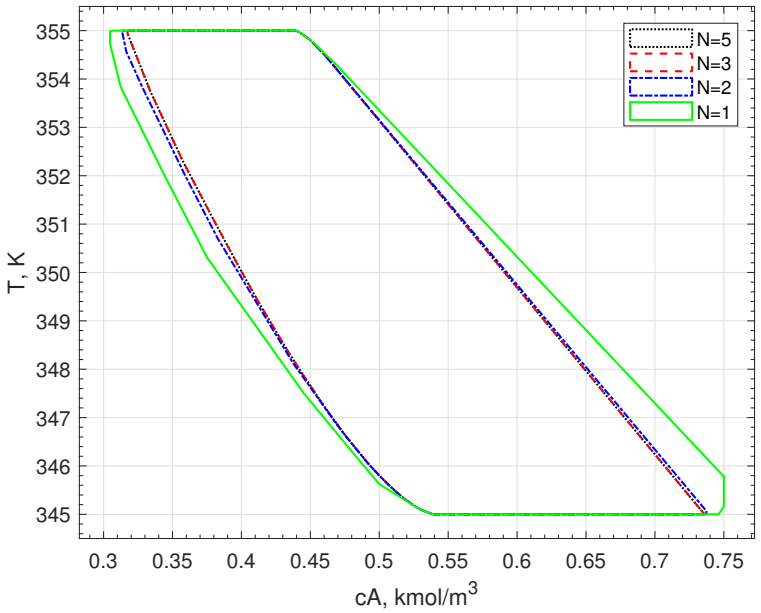


Figure 3.7: Sets for different N after 20 iterations. The invariant sets in the figure were obtained by finding the convex hull of the final cells after the algorithm

means that only the boundary cells are selected for subdivision. This value of N was chosen to ensure that the number of cells is not too large to slow down the plotting of the figure. As expected, it can be seen that the cells are finer at the boundaries and coarser at the interior of the set.

In the next set of computations, we varied the parameter N and recorded the number of cells generated at each iteration as well as the computation times. Figures 3.7, 3.8 and 3.9 show the convex hull of the sets generated, the number of cells generated and the computation times after 20 iterations respectively for different values of N. It can be seen in Figure 3.7 that the parameter N affects the speed of convergence of the algorithm to the largest control invariant set R_X. For the same number of iterations, that is 20, the computation with a higher value of N converges faster to R_X compared that with a smaller value. This implies that a higher number of iterations is needed for the set to converge to R_X when for example, $N = 1$. As explained in Section 3.3, when a small value of N is used, the computation focuses on the boundary cells while searching for the boundary of R_X. Thus, more time is spent refining the cells in areas which do not contain the boundary of R_X. However, as

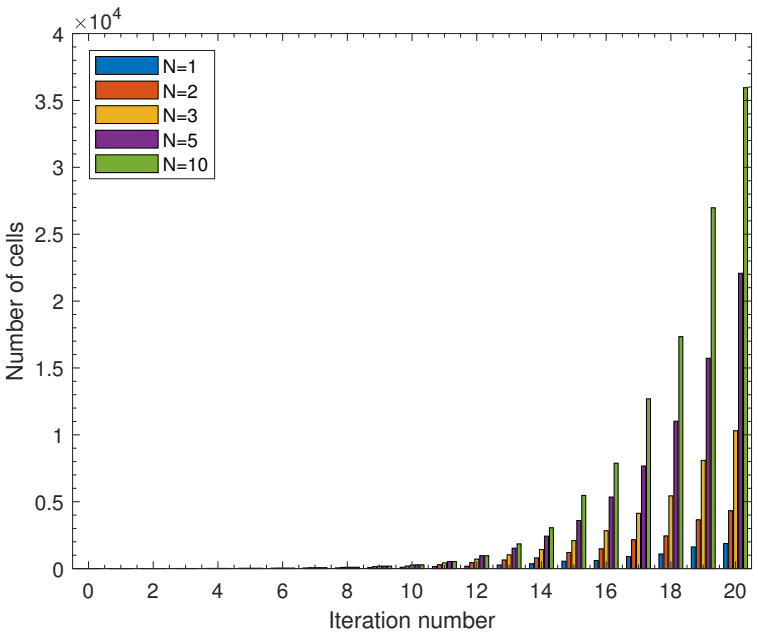


Figure 3.8: Number of cells generated at each iteration of the algorithm with different N.

the value of N increases, more cells are selected. This means that larger areas within the domain of interest is explored in the search for the largest control invariant set. Hence, the selection of the parameter N is not a trivial task. The choice of the value of N depends on the properties of the system under study.

In this example, it can be seen from Figure 3.7 that the optimal value of N is 3 since there's no difference between the set when $N = 3$ and $N = 5$. Furthermore, taking a closer look at the number of cells generated and the computation times (in Figures 3.8 and 3.9 respectively), the number of cells is significantly reduced at $N = 3$. Also, at $N = 3$, the computational savings is 8 times that of $N = 1000$ which represents the case when all the cells are subdivided at each iteration.

3.4.3 Parallelization results

In this section, we compare the computational speed improvements for the parallelization of the graph construction step of the algorithm. In this set of simulations, all the cells were subdivided without using the adaptive cell subdivision. This is to ensure that only the

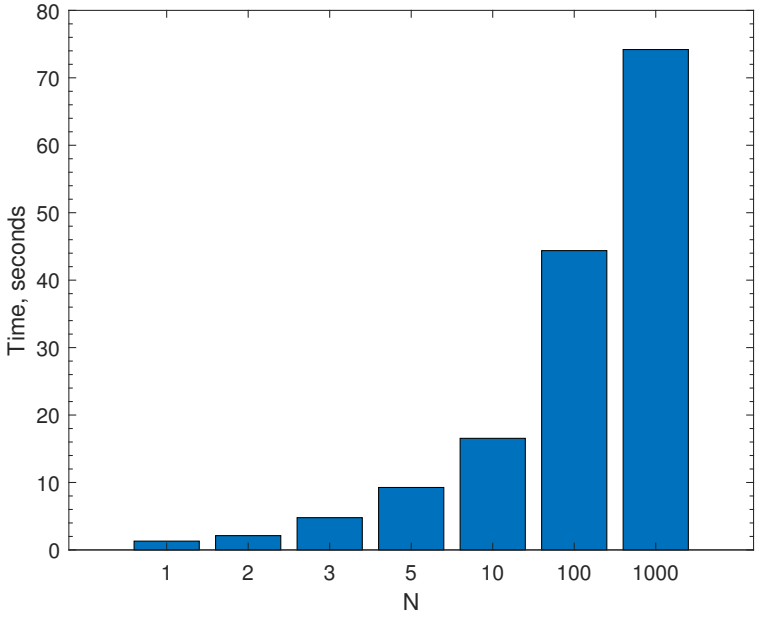


Figure 3.9: Computation times for different N.

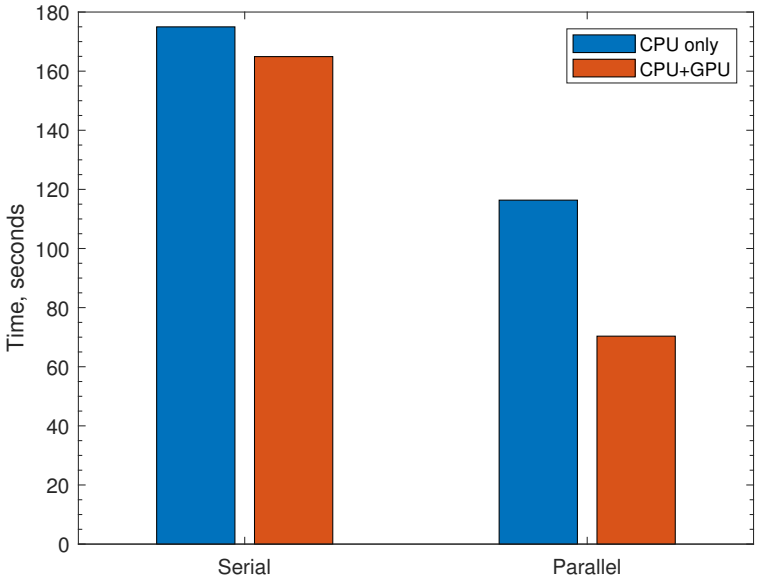


Figure 3.10: Comparison of computation speed for both serial and parallel computation, with and without GPU usage.

effect of the parallelization observed. We also considered the effect of using the GPU on the computation speed. Figure 3.10 summarizes the results of the computation of the largest control invariant set using the parallelized algorithm. It can be seen that in both cases, the parallelization sped up the computation with a much improved speed, in the GPU case. The number of cells in a batch was selected as 1024 in the case where GPU is used.

3.5 Concluding remarks

In this chapter, we have presented an improved and efficient graph-based invariant set algorithm for computing approximations of the largest control invariant set of constrained controlled nonlinear systems. We first critically analyzed the computational complexity of the standard GIS algorithm. It was observed that the graph construction and the subdivision steps have significant impacts on the overall time complexity of the algorithm. Thus, we proposed two methods to improve the algorithm namely, adaptive subdivision and parallelization of the graph construction step. We demonstrated the efficacy of the improved algorithm using a nonlinear continuously stirred tank reactor. It was observed that the adaptive subdivision method only affects the speed of convergence to the largest control invariant set and not the convergence itself. Furthermore, the adaptive subdivision improved the speed of the algorithm by about 8x that of standard algorithm. Also, the parallelization of the graph construction step improved the computation speed by about 3x that of the standard algorithm.

Chapter 4

A distributed control invariant set computing algorithm for constrained nonlinear cascade systems

4.1 Introduction

In the previous chapter, we modified the GIS algorithm to improve its computational efficiency with respect to the cell subdivision and graph construction steps. While the modifications were able to improve the algorithm, it does not necessarily solve the problem pertaining to the scalability of the algorithm with respect to the state dimension. In this chapter, we focus on cascade systems and present a system decomposition method and a distributed approach for computing control invariant sets. The proposed algorithm exploits the structure of the interconnections within a process network and decomposes the entire process network into smaller subsystems. Following the decomposition, a distributed approach is developed to compute the control invariant set of the entire system. The proposed approach adopts graph-based algorithms in computing the control invariant sets [32]. In contrast to other works on distributed computation of control invariant sets, our proposed approach produces

sets that approximates the largest control invariant set since the missing interconnection information is not treated as disturbances. We demonstrated the convergence of the results from the decomposition-based graph algorithm to that of the standard centralized algorithm using several numerical examples including a six dimensional continuous stirred tank reactor.

4.2 Problem formulation and background

4.2.1 Notation

Throughout this chapter, the operator $\text{proj}_i(x)$ denotes the projection of the set or point x onto the subspace of subsystem i. The operator / denotes set subtraction such that $\mathbb{A}/\mathbb{B} = \{x : x \in \mathbb{A}, x \notin \mathbb{B}\}$. $G = (V, E)$ represents a directed graph with V denoting the set of vertices of the graph and E denoting the set of ordered pairs of vertices known as edges. The operator $|\cdot|$ denotes the Euclidean norm of a vector.

4.2.2 Problem formulation

We are concerned with a class of nonlinear systems composed of N subsystems coupled together in a cascade manner. The dynamics of the overall system is described by

$$x^+ = f(x, u) \tag{4.1}$$

where $x \in \mathbb{R}^n$ is the current state of the system, $u \in \mathbb{R}^m$ is the current control input, and $x^+ \in \mathbb{R}^n$ denotes the state of the system at the next sampling time. We assume that the state and the control input of the system are restricted to be in the compact constraint sets $X \subset \mathbb{R}^n$ and $U \subset \mathbb{R}^m$ respectively. Without loss of generality, we also assume that the vector field $f : X \times U \to X$ is a sufficiently smooth vector field in X. The coupling between the subsystems is such that the first subsystem is independent of the other subsystems. Also, any subsystem i, other than the first subsystem, is directly affected by the upstream

subsystem $i-1$ but not the downstream subsystem $i+1$. The structure of the cascade system considered in this chapter is represented by

$$x_1^+ = f_1(x_1, u_1) \tag{4.2a}$$

$$x_2^+ = f_2(x_2, u_2) + g_2(x_1) \tag{4.2b}$$

$$\vdots$$

$$x_N^+ = f_N(x_N, u_N) + g_N(x_{N-1}) \tag{4.2c}$$

where f_i denotes the local dynamics of state x_i, and g_i denotes the dynamics of the coupling between the states x_i and x_{i-1}. The subscript $i = 1, 2, \cdots, N$ represents the ith subsystem. The state and input constraints for subsystem i is given by $X_i = \text{proj}_i(X)$ and $U_i = \text{proj}_i(U)$ respectively.

Before we begin our discussion, let us introduce the following definitions which are central and referred to throughout this chapter.

Definition 4.1 (Forward invariant set [9])**.** *A set $R \subseteq X$ is said to be a forward or positively invariant set of the system $x^+ = f(x)$ if for every $x \in R$, $f(x) \in R$.*

Definition 4.2 (Control invariant set [9])**.** *A set $R \subseteq X$ is said to be a control invariant set (CIS) of system (4.1) if for every $x \in R$, there exist a feedback control law $u = \mu(x) \in U$ such that R is forward invariant for the closed-loop system $f(x, u)$.*

Definition 4.3 (Largest control invariant set [25])**.** *A set $R_X \subseteq X$ is said to be the largest (with respect to inclusion) control invariant set of system (4.1) if R_X is control invariant and contains all other control invariant sets contained in X.*

In general, the state constraint X for a given system is not control invariant. However, one may wish to find the largest control invariant set R_X contained in X for controller design and assessment purposes. The goal of this chapter is to present a framework for computing an approximation of the largest control invariant set R_X when the state dimension n is too

large to make the standard GIS algorithm tractable [32]. At present, this happens when $n > 4$. This is because the number of cells generated in the standard GIS algorithm grows exponentially in the state dimension.

Our solution to computing an approximation of R_X is to decompose system (4.2) with N subsystems into M subsystems with overlapping states.

4.3 System decomposition and set invariance

In trying to alleviate the exponential cell growth in the GIS algorithm, we propose to decompose the overall system into smaller subsystems. This makes it computationally tractable for control invariant set approximation using the GIS algorithm. System decomposition is a typical way of addressing the computational challenges associated with control and state estimation of large scale dynamical systems. This is also the case for control invariant set calculation. The goal is to divide the original system into many small subsystems thus making the computations tractable. Usually, the decomposition is achieved by dividing the system in such a way that the subsystems have weak to no coupling or interconnection.

In this section, we investigate the system structures suitable for decomposition and control invariant set computation. In particular, we consider simple parallel and series/cascade system structures. Throughout this section, we restrict the discussion to two and three dimensional system structures namely, series and parallel system structures to make the ideas presented here easier to follow. The ideas presented here can easily be generalized to much higher dimensional systems.

We begin this section by analyzing two types of system structures, their decomposition and the ability to reconstruct the solution from the solution of the subsystems. Thereafter we described the method of overlapping decomposition, which is a precursor for the proposed distributed algorithm.

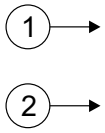


Figure 4.1: Parallel or independent system structure

4.3.1 System structures and invariance

Consider the disjoint system structure

$$x^+ = \hat{f}(x) = \begin{bmatrix} \hat{f}_1(x_1) \\ \hat{f}_2(x_2) \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} x \tag{4.3}$$

where $x = [x_1 \; x_2]^T$ subject to the state constraint X. A graphical depiction of the system is shown in Figure 4.1. Since the evolution of the system states are independent of each other, the system equations can be intuitively decomposed into the following subsystems

$$S_1 : x_1^+ = \hat{f}_1(x_1) = A_{11}x_1 \text{ and } S_2 : x_2^+ = \hat{f}_2(x_2) = A_{22}x_2 \tag{4.4}$$

with subspaces $X_1 = \text{proj}_1(X)$ and $X_2 = \text{proj}_2(X)$ respectively. To construct a graph representation of the dynamics of each subsystem, the subspaces X_1 and X_2 need to be quantized into $\mathcal{C}_1$ and $\mathcal{C}_2$. In what follows, we show how the control invariant set can be computed for the full system from the subsystem information.

Definition 4.4 (Graph Cartesian product [53])**.** *The Cartesian product of the directed graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, denoted $G_1 \times G_2$, is a graph $G = (V, E)$ such that $V = V_1 \times V_2$ and for any two points $u = (u_1, u_2)$ and $v = (v_1, v_2)$ in V, the directed edge $(u, v) \in E$ whenever $u_1 = v_1$ and $(u_2, v_2) \in E_2$, or $u_2 = v_2$ and $(u_1, v_1) \in E_1$.*

An illustration of the Cartesian product of two fictitious digraphs G_1 and G_2 is presented in Figure 4.2.

Assumption 4.1 (Subsystem quantization)**.** *The set $\mathcal{C}$, which is a quantization of X, is the cross product of $\mathcal{C}_1$ and $\mathcal{C}_2$, that is $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2$.*

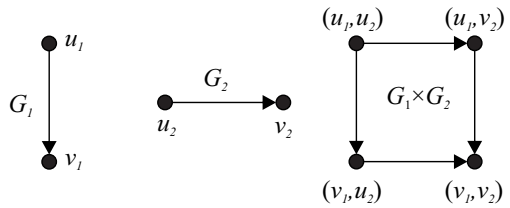


Figure 4.2: The Cartesian product of two graphs

Proposition 4.1. *Consider the system described by Equation* (4.3) *and decomposed in the form of Equation* (4.4). *Let* $G_1 = (V_1, E_1)$ *and* $G_2 = (V_2, E_2)$ *be a directed graph representation of the dynamics of each subsystem* S_1 *and* S_2 *respectively based on the sets* $\mathcal{C}_1$ *and* $\mathcal{C}_2$. *Also, let* $G = (V, E)$ *be the directed graph representation of system* (4.3) *based on* $\mathcal{C}$. *If Assumption 4.1 holds, then the full system solution can be exactly obtained from the subsystem solutions, that is,*

$$I^+(G) = I^+(G_1 \times G_2)$$

Proof. To prove the above assertion, we first need to show that the vertices of the two graphs G and $G_1 \times G_2$ are equal. Thereafter, we need to show that if there is an admissible path between any two cells in G, then there is an equivalent admissible path between those same two cells in $G_1 \times G_2$.

First, we show that the vertices of the two graphs G and $G_1 \times G_2$ are equal. It follows directly from Assumption 4.1 that $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2$. Therefore from Definition 4.4, we have that $V = V_1 \times V_2$.

Now, we show by that if there is a path between any two cells in G, then there exist a path between those same two cells in $G_1 \times G_2$. Let $B_i = (u_1, u_2)$ and $B_j = (v_1, v_2)$ be any two cells in V such that u_1 and v_1 are in V_1, and u_2 and v_2 are cells in V_2. From the graph

construction procedure, we have that

$$\begin{aligned} E &= \{(B_i, B_j) \in \mathcal{C} \times \mathcal{C} : \hat{f}(B_i) \cap B_j \neq \emptyset\} \\ &= \{(B_i, B_j) \in \mathcal{C} \times \mathcal{C} : \hat{f}_1(u_1) \cap v_1 \neq \emptyset \wedge \hat{f}_2(u_2) \cap v_2 \neq \emptyset\} \\ &= \{(B_i, B_j) \in \mathcal{C} \times \mathcal{C} : (u_1, v_1) \in E_1 \wedge (u_2, v_2) \in E_2\} \end{aligned}$$

From the definition of $G_1 \times G_2$ and E, we have that the path $(u_1, u_2) \to (u_1, v_2) \to (v_1, v_2)$ and $(u_1, u_2) \to (v_1, u_2) \to (v_1, v_2)$ exist in $G_1 \times G_2$. This implies that there is an admissible path from (u_1, u_2) to (v_1, v_2) in $G_1 \times G_2$. It then follows that if an infinite admissible path passes through the cell B_i in G, then an infinite admissible path also passes through the cell $B_i = (u_1, u_2)$ in $G_1 \times G_2$. Hence we have that

$$I^+(G) = I^+(G_1 \times G_2),$$

which completes the proof. □

While we have shown that an approximation of R_X can be obtained for the disjoint system by constructing the digraph $G_1 \times G_2$, in most cases this will require a considerable amount of memory to store the graph for higher dimensional systems. The following theorem shows how an approximation of the largest control invariant set can be exactly obtained from the solutions of the subsystems.

Theorem 4.1. *Consider the system described by Equation* (4.3) *and decomposed in the form of Equation* (4.4). *Let* $G_1 = (V_1, E_1)$ *and* $G_2 = (V_2, E_2)$ *be a directed graph representation of the dynamics of each subsystem* S_1 *and* S_2 *respectively based on the sets* $\mathcal{C}_1$ *and* $\mathcal{C}_2$. *Also, let* $G = (V, E)$ *be the directed graph representation of system* (4.3) *based on* $\mathcal{C}$. *If Assumption 4.1 holds, then the full system solution can be exactly obtained from the subsystem solutions, that is,*

$$I^+(G) = I^+(G_1) \times I^+(G_2)$$

Proof. We know from Proposition 4.1 that $I^+(G) = I^+(G_1 \times G_2)$. Let $B_i = (u_1, u_2) \in V$ where $u_1 \in V_1$ and $u_2 \in V_2$. It can be inferred from the definition of $G_1 \times G_2$ that wherever there is an infinite admissible path passing through B_i, there is also infinite admissible paths passing through u_1 in G_1 and u_2 in G_2. This implies that

$$I^+(G) = I^+(G_1) \times I^+(G_2) \qquad \square$$

Theorem 4.1 shows that the graph representation of the subsystems can be analyzed independently. Then the cells that approximate the largest control invariant set for full system can be computed by finding the Cartesian product of the individual subsystem solutions. This is computationally more efficient since a large graph is not constructed for analysis. To illustrate the ideas presented so far, let us consider Example 4.1.

Example 4.1. *Consider the system*

$$x^+ = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u \tag{4.5}$$

where the constraints on the states and input are $X = \{x \in \mathbb{R}^2 : \|x\|_\infty \leq 5\}$ *and* $U = \{u \in \mathbb{R}^2 : \|u\|_\infty \leq 1\}$ *respectively.*

The system in Example 4.1 can be decomposed into two subsystems in a form similar to Equation (4.4). Figure 4.3 shows the largest control invariant sets R_{X_1} and R_{X_2} for each of the subsystems, the largest control invariant set R_X for the full system and the reconstructed solution from the control invariant sets of the subsystems. It can be seen that the full system solution can be exactly obtained from the subsystem solution without incurring any error due to the decomposition. This agrees well with our earlier arguments in Theorem 4.1, that $R_X = R_{X_1} \times R_{X_2}$. Ultimately, this shows that we can freely move between the control invariant sets of the lower dimension subsystems and the control invariant set for the full system without incurring an losses. Geometrically, the structure of the control invariant set

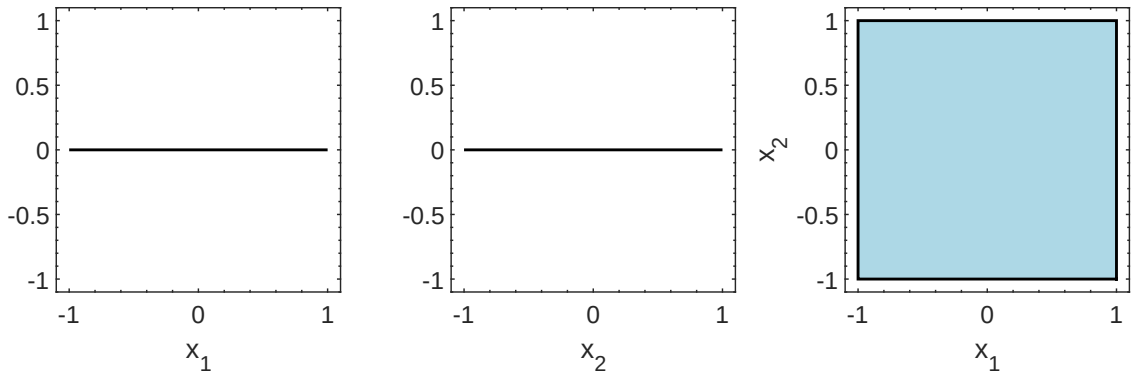


Figure 4.3: Decomposition and full solution reconstruction for system (4.5). Left: Solution of Subsystem 1. Middle: Solution of Subsystem 2. Right: The reconstructed full system solution from the subsystem solutions (black line) and the actual solution of the full system (light blue region).

is always going to be a box.

While an approximation of the largest control invariant set can be obtained from the solutions of the subsystems without incurring any errors due to decomposition for disjoint subsystems, such systems rarely occur in the real world. In most cases, the subsystems may be interconnected. Let us consider one of such cases where is coupling between the system states. Consider the system

$$x^+ = \hat{f}(x) = \begin{bmatrix} A_{11} & 0 \\ A_{21} & A_{22} \end{bmatrix} x \tag{4.6}$$

where $x = [x_1 \ x_2]^T$ subject to the state constraint X. A graphical depiction of the system is shown in Figure 4.4. Similar to the disjoint structure in Equation (4.3), system (4.6) can be decomposed into two subsystems such that

$$S_1 : x_1^+ = \hat{f}_1(x_1) = A_{11}x_1 \text{ and } S_2 : x_2^+ = \hat{f}_2(\tilde{x}_1, x_2) = A_{22}x_2 + A_{21}\tilde{x}_1 \tag{4.7}$$

with subspaces $X_1 = \text{proj}_1(X)$ and $X_2 = \text{proj}_2(X)$ respectively. Notice that while Subsystem 1 is independent of the dynamics of Subsystem 2, Subsystem 2 is dependent on the dynamics of Subsystem 1. The state information about Subsystem 1 required by Subsystem 2

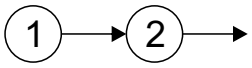


Figure 4.4: Series or connected system structures

is denoted by $\tilde{x}_1$ in Equation (4.7). To work with such a decomposition, the fundamental issues that needs to be addressed in this scenario are how to obtain the values of $\tilde{x}_1$ and how to use this information in Subsystem 2.

It is easy to see that the range of values for $\tilde{x}_1$ can be obtained from the solution of Subsystem 1. For the choice of usage of the $\tilde{x}_1$ information, one approach is to use the $\tilde{x}_1$ information as an input in Subsystem 2. However, this more often than not result in an over approximation of the solution of the full system.

Theorem 4.2. *Consider the system described by Equation* (4.6) *and decomposed in the form of Equation* (4.7)*. Let* $G_1 = (V_1, E_1)$ *and* $G_2 = (V_2, E_2)$ *be a directed graph representation of the dynamics of each subsystem* S_1 *and* S_2 *respectively based on the sets* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. Also, let* $G = (V, E)$ *be the directed graph representation of system* (4.6) *based on* $\mathcal{C}$*. If Assumption 4.1 holds, then*

$$I^+(G) \subseteq I^+(G_1) \times I^+(G_2)$$

Proof. This proof follows along the same lines as Theorem 4.1. We know from Proposition 4.1 that $I^+(G) = I^+(G_1 \times G_2)$. Let $B_i = (u_1, u_2) \in V$ where $u_1 \in V_1$ and $u_2 \in V_2$. It can be inferred from the definition of $G_1 \times G_2$ that wherever there is an infinite admissible path passing through B_i, there is also infinite admissible paths passing through u_1 in G_1 and u_2 in G_2. This implies that

$$I^+(G) \subseteq I^+(G_1) \times I^+(G_2) \qquad \square$$

The reverse of Theorem 4.2, that is $I^+(G_1) \times I^+(G_2) \subseteq I^+(G)$ is necessarily not true. To demonstrate this, let us consider the following example.

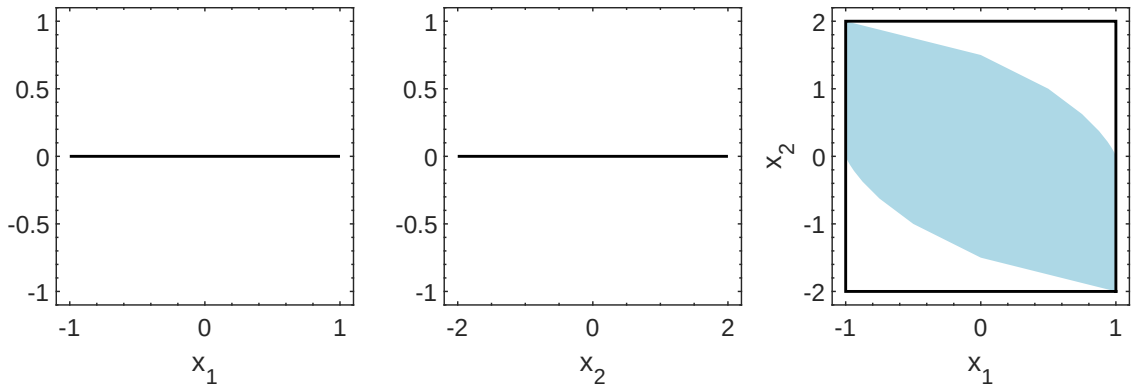


Figure 4.5: Decomposition and full solution reconstruction for system (4.8). Left: Solution of Subsystem 1. Middle: Solution of Subsystem 2 when $\tilde{x}_1$ is treated as an input. Right: The reconstructed full system solution from the subsystem solutions (black line) and the actual solution of the full system (light blue region).

Example 4.2. *Consider the system*

$$x^+ = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u \tag{4.8}$$

where the constraints on the states and input are $X = \{x \in \mathbb{R}^2 : \|x\|_\infty \leq 5\}$ *and* $U = \{u \in \mathbb{R}^2 : \|u\|_\infty \leq 1\}$ *respectively.*

As an illustration of our arguments so far, consider the system in Example 4.2. System (4.8) can be decomposed into two subsystems following Equation (4.7). Figure 4.5 shows the solution of the subsystems R_{X_1} and R_{X_2}, when $\tilde{x}_1$ is treated as an input in Subsystem 2, the reconstructed solution of the full system from the solution of the subsystems, and the solution of the full system R_X. It can be seen that $R_X \subseteq R_{X_1} \times R_{X_2}$. Hence, there exists some states in $R_{X_1} \times R_{X_2}$ which are not in R_X.

To address the issue of over approximation, several studies have proposed to treat $\tilde{x}_1$ as an uncertain input in Subsystem 2 [38, 54]. By treating $\tilde{x}_1$ as an uncertain input, a worse case scenario analysis can be considered during the computation of the solution of Subsystem 2. This results in a robust control invariant set solution for Subsystem 2. While this approach works, there are several downsides to it. First, it is neither trivial nor easy to compute

robust control invariant sets. In fact, more resources are needed to compute robust control invariant sets than to compute control invariant sets. Second, the resulting reconstructed set is often conservative or small compared to the control invariant set for the full system. In some instances, the computation may end up with an empty set for Subsystem 2 which is undesired. Let us consider the system in Example 4.2 again. When $\tilde{x}_1$ is treated as an uncertainty in its worst case, $R_{X_2} = \emptyset$, implying that $R_{X_2} \times R_{X_2} = \emptyset$, which is not desirable. This also shows that $R_{X_1} \times R_{X_2} \subseteq R_X$ when the $\tilde{x}_1$ is treated as an uncertainty, which is a conservative result.

In a nutshell, it can be seen that when the subsystems are connected, the full system solution cannot be easily obtained from the solution of the subsystems. Further scrutinizing the actual control invariant set for system (4.8) in Figure 4.5, it can be seen that $\tilde{x}_1$ actually behaves as an input at some points in x_2 and as an uncertainty in other x_2 locations. This is the central idea on which we develop our distributed algorithm.

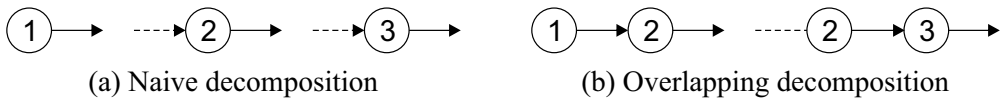
4.3.2 Overlapping system decomposition

In the preceding section, we have discussed the implication of decomposition of set invariance using 2 dimensional systems. But how do we decompose connected systems when the system dimension is greater than 2? In this section, we briefly discuss how to decompose three dimensional cascade systems.

Let us consider the following system

$$x^+ = \hat{f}(x) = \begin{bmatrix} A_{11} & 0 & 0 \\ A_{21} & A_{22} & 0 \\ 0 & A_{32} & A_{33} \end{bmatrix} x \tag{4.9}$$

where $x = [x_1 \ x_2 \ x_3]^T$ subject to the state constraint X. There are different ways to decompose system (4.9) into several subsystems. A simple but naive decomposition is to consider each state in Equation (4.9) separately as shown in Figure 4.6. This way, three



(a) Naive decomposition (b) Overlapping decomposition

Figure 4.6: Different decomposition strategies for the cascade system

subsystems will be obtained with each subsystem focusing on a single state. While this look simple and is similar to the parallel decomposition, it require information about the state in the preceding subsystem. Moreover, the control invariant sets obtained this way based on the subsystems are not very useful in reconstructing the control invariant set of the original system since they do not contain the interconnection information about the neighboring states. The only way to reconstruct the control invariant set from the subsystem solutions is to find the Cartesian product of the resulting control invariant set for the each subsystem. This will ultimately result in a large over approximation of R_X.

A more useful way is to decompose system (4.9) into subsystems such that each subsystem is chained to its neighboring subsystem as shown in Figure 4.6. This way, the interactions between the states can be accounted for. As an example, system (4.9) can be decomposed into two subsystems that share a common part. This can be considered as equivalent to expanding the system to a higher dimension since the overlapping states are repeated in the state equation as shown in Equation (4.10). The overlapping states play an important role in reconstructing the control invariant set from the distributed computing results. The role of the overlapping states will be made clear in the later discussion.

More formally, let $z \in \mathbb{R}^{n_1+n_2}$ denote the states for the expanded system for system (4.6) such that $z = [x_1\ x_2\ x_2\ x_3]^T$ is the expanded system state vector with $z_1 = [x_1\ x_2]^T \in \mathbb{R}^{n_1}$ and $z_2 = [x_2\ x_3]^T \in \mathbb{R}^{n_2}$ being the subsystem states. The two subsystems are shown in Equation (4.11). The corresponding decomposition is depicted in Figure 4.6.

$$z^+ = \left[\begin{array}{cc|cc} A_{11} & 0 & 0 & 0 \\ A_{21} & A_{22} & 0 & 0 \\ \hline A_{21} & 0 & A_{22} & 0 \\ 0 & 0 & A_{32} & A_{33} \end{array}\right] z \tag{4.10}$$

$$S_1 : z_1^+ = f_1(z_1) = \begin{bmatrix} A_{11} & 0 \\ A_{21} & A_{22} \end{bmatrix} z_1 \quad S_2 : z_2^+ = f_2(z_2, \tilde{x}_1) = \begin{bmatrix} A_{22} & 0 \\ A_{32} & A_{33} \end{bmatrix} z_2 + \begin{bmatrix} A_{21} \\ 0 \end{bmatrix} \tilde{x}_1 \tag{4.11}$$

Notice that while Subsystem 1 is independent of any external state, Subsystem 2 requires information about x_1 similar to our earlier discussions. We will refer to the information about x_1 required by Subsystem 2 as missing state information, denoted by $\tilde{x}_1$, in the next section. How the missing state information is obtained will be discussed in the subsequent section. As can be seen in the above equations, the decomposed system has a higher dimension than the original system i.e. $n_1 + n_2 > n$. While the above example used a linear system, this type of decomposition can be readily applied to nonlinear systems with exogenous inputs and serves as a precursor to the proposed algorithm. Unfortunately, there is no general way of reconstructing the control invariant set for the overall system from the solution of the subsystems. This is because of the coupling between the two subsystems and that in Subsystem 2, though x_1 information is used, the dynamics is not considered.

4.4 Computation of the largest control invariant set via system decomposition

In this section, we present an algorithm to compute an approximation of R_X via system decomposition. We focus on the cascade structure presented in Section 4.3. Given a cascade system with its associated overlapping decomposition, the algorithm computes the control

invariant set for each of the subsystems, first in a decentralized manner, followed by a distributed. Thereafter, the control invariant for the overall system is reconstructed and subset of the cells tested for control invariance. The central idea employed here is that the missing state information is treated as both a regular input and an uncertain input. This way, the two aspects of the missing state is utilized. As we will show in this section, the use of graph analysis makes it easier for this to be achieved. Throughout this section, we assume there are only two subsystems S_1 and S_2, that is $M = 2$, as described in Equations (4.11) with subspaces $X_1 = \text{proj}_1(X)$ and $X_2 = \text{proj}_2(X)$ respectively.

We begin this section by first describing procedure for the decentralized and distributed computation. Thereafter, we present the procedure for the reconstruction and validation of the solution. We end this section by conducting a brief analyzes of the complexity of the proposed algorithm.

4.4.1 Decentralized and distributed computation

The decentralized and the distributed computation makes use of the standard GIS algorithm to determine the cells which approximate the largest control invariant sets (R_{X_1} and R_{X_2}) for each of the subsystems. We denote the solutions of the subsystems by R_1 and R_2 respectively. During the computations, the directed graphs G_1 and G_2 for each subsystem are constructed based on the collections $\mathcal{C}_1$ and $\mathcal{C}_2$ of each subsystem respectively. Further analysis on these graphs is vital for the reconstruction of the solution of the overall system.

In the decentralized step, each subsystem is treated as independent of each other. This allows for the computation of R_1 and R_2 in parallel. The goal is to quickly obtain an approximation of the largest control invariant sets and reduce the search space for further computations. To make this possible, the interval for missing state information for x_1 in Subsystem 2 is held constant during the computation. In this case, an approximation which

is obtained by projecting the state constraint onto the dimension of x_1 may be used, that is

$$\tilde{x}_1 = \text{proj}_u X \tag{4.12}$$

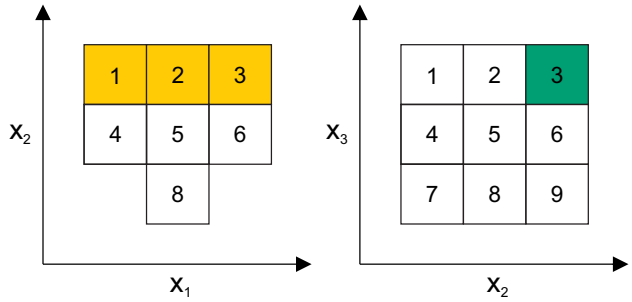
where u denotes the dimension of x_1. Once the computation begin, there is no communication between the solutions of the two subsystems. The algorithm for the decentralized step is summarized in Algorithm 4. The algorithm takes as input the subsystem equations, the quantized subspaces for each subsystem, the input constraint and the number of subsystems. As mentioned earlier, this is fixed since only two subsystems are considered here. The algorithm returns estimates of the control invariant sets for each of the subsystems as well as their associated digraphs.

Algorithm 4: Decentralized computation

Input: Subsystems in Equation (4.11) and the collections $\mathcal{C}_1$ and $\mathcal{C}_2$, U, $M = 2$
Output: G_i, R_i, $i = 1, \cdots, M$

```
G_1 ← Construct the directed graph representation of S_1
R_1 ← I^+(G_1)
for i = 2 ··· M do
    // Can be done in parallel
    x̃_{i-1} ← proj_{i-1} X
    G_i ← Construct the directed graph representation of S_i utilizing x̃_{i-1}
    R_i ← I^+(G_i)
return R_i, G_i, (i = 1, ··· , M)
```

In the distributed step, the control invariant set for each subsystem is computed sequentially. Thus, the solution for the immediately preceding subsystem is used to estimate the missing state information for the current subsystem. While the missing state information for each subsystem is obtained from the state constraint in the decentralized computation, the missing state information is dynamically obtained from the neighboring subsystem solution in the distributed computation. This is the key difference between the decentralized computation and the distributed computation. It is also the reason for computing the solution for each subsystem in a sequential fashion. The procedure for dynamically estimating the



(a) Subsystem 1 (b) Subsystem 2

Figure 4.7: Procedure for dynamically estimating the missing x_1 information for each cell in Subsystem 2. (a) Solution of Subsystem 1 R_1. The yellow cells indicate the cells that correspond to B_3 in the solution of Subsystem 2. Merging and projecting these cells onto the x_1 dimension produces the range of x_1 for B_3. (b) Solution of Subsystem 2 R_2. The green cell indicate the cell whose missing state information is being estimated. This procedure is repeated for all other cells in R_2.

missing state information in the computation of the control invariant set of Subsystem 2 is presented in Figure 4.7. By allowing unidirectional communication through dynamic estimation of the missing state information, the solution of each subsystem is further refined. Furthermore, a more realistic graph representation of the dynamics of each subsystem is obtained. This will be useful during the set reconstruction step of the algorithm. The algorithm for the distributed computation is presented in Algorithm 5. It is similar to the algorithm used for the decentralized computation. As mentioned earlier, the key difference is how the missing state information $\tilde{x}_i$ is obtained.

Algorithm 5: Distributed computation

Input: Subsystems in Equation (4.11) and the collections $\mathcal{C}_1$ and $\mathcal{C}_2$, U, $M = 2$
Output: G_i, R_i, $i = 1, \cdots, M$

$G_1 \leftarrow$ Construct the directed graph representation of S_1
$R_1 \leftarrow I^+(G_1)$
for $i = 2 \cdots M$ **do**
    Estimate the range of $\tilde{x}_{i-1}$ for S_i from R_{i-1}
    $G_i \leftarrow$ Construct the directed graph representation of S_i utilizing $\tilde{x}_{i-1}$
    $R_i \leftarrow I^+(G_i)$
return $R_i, G_i, (i = 1, \cdots, M)$

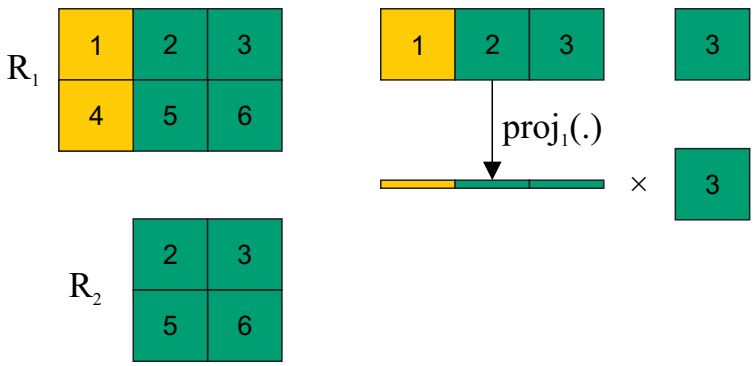


Figure 4.8: Procedure for reconstructing the control invariant set for the overall system from the subsystem solutions R_1 and R_2. The procedure is indicated for Cell B_3 in R_2. Because of the overlap, B_6 in R_2 has a connection with with B_1, B_2 and B_3 in R_1. These cells are then projected onto the x_1 dimension to obtain the range of x_1 for each cell. The Cartesian product of the B_3 with the range of x_1 produces 3 dimensional cells (in this case 3 cells). This is repeated for all other cells in R_2 to obtain the solution R for the overall system.

4.4.2 Set reconstruction and validation

Following the distributed computation step, the solutions from the two subsystems, that is R_1 and R_2, together with the directed graphs G_1 and G_2 are used to reconstruct the solution for the overall system R. Notice that the Cartesian product of the two sets will result in a 4 dimensional object which is not accurate. This is because of the overlapping decomposition. The procedure for reconstructing R therefore involves projecting the cells in R_1 onto the x_1 dimension in a dynamic fashion similar to the procedure for estimating the missing state information in the distributed computation. However, the range of x_1 is not merged in this case since the focus is reconstruction of the set for the overall system. The procedure for reconstructing the set is described in Figure 4.8.

As described earlier, reconstructing the solution of the overall system from the subsystem solutions may often than not result an approximation error due to the decomposition and coupling between the subsystems. However, compared to the naive decomposition, the approximation error for the overlapping decomposition is expected to be smaller. Nonetheless, the approximation error need to be addressed to ensure that the reconstructed solution from the solution of the subsystems converges to solution when the full system model is used.

To address the approximation error due to the decomposition, we propose that a subset of the cells forming the reconstructed set needs to be validated for inclusion in the final reconstructed solution. The validation test involves computing the next feasible states of each cell to be tested using the overall system model. Thereafter, the next feasible states of the cell is checked for intersection with the reconstructed set. A cell B_i fails the validation test if its successive states does not intersect the reconstructed set R, that is

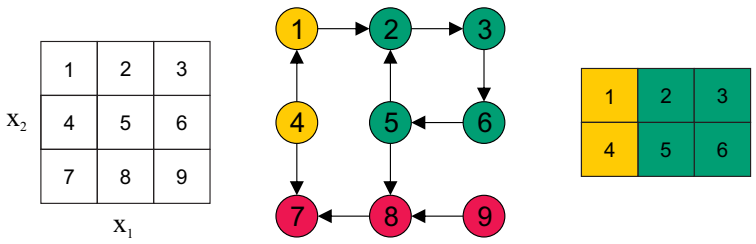
$$F(B_i) \cap R = \emptyset \tag{4.13}$$

This is a direct intuition from the definition of control invariance. Any cell that fails the validation test is removed from the reconstructed set.

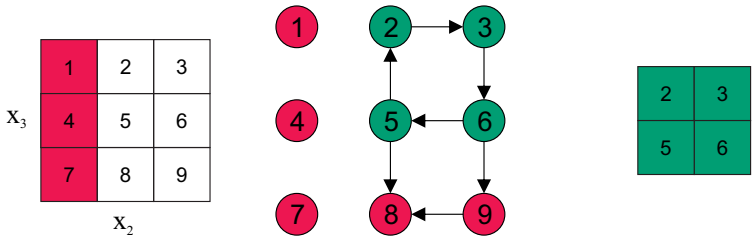
What now remains is how to select the cells from the reconstructed set R for the validation test. In principle, each cell in the reconstructed set need to be tested. However, this may result in having to check for a large number of cells especially when the system dimension is high. An alternative approach is to further investigate the graph G_2 from the distributed computation step to determine the cells that need to be tested. This way, the cells to test will be significantly reduced. The general idea we use to further investigate the graph G_2 is to consider the adversarial aspects of the missing state information used to construct G_2. We know by definition that the set R_2 is made of cells with infinite admissible paths passing through it, that is non-leaving cells. Thus, if there is a direct path from a cell in R_2 to a cell not in R_2, then that path could have been caused by the missing state $\tilde{x}_1$. By doing this, we consider the contribution of the missing state $\tilde{x}_1$ as a regular input and as an uncertain input.

Given the collection $\mathcal{C}_2$ for Subsystem 2, let $\mathcal{C}_2^{lc}$ denote the leaving cells. It is easy to see that $\mathcal{C}_2 = R_2 \cup \mathcal{C}_2^{lc}$.

Definition 4.5 (Incoming neighbors)**.** *Given the directed graph $G = (V, E)$, the incoming or in neighbors of the vertex $u \in V$ are the nodes $v \in V$ such that the edges $(v, u) \in E$ exist.*



(a) Graph construction and analysis for Subsystem 1



(b) Graph construction and analysis for Subsystem 2

Figure 4.9: Procedure for the analysis of the distributed graphs to find the cells to be tested.

The central idea is to find the incoming neighbors of the leaving cells in the graph of Subsystem 2. Then we find those incoming neighbors that have an intersection with the non-leaving cells. Let $\mathcal{C}_2^t$ denote the cells that need to be tested for Subsystem 2 and $\text{in}(G_2, \mathcal{C}_2^{lc})$ be the incoming neighbors of the leaving cells on G_2. Then the following relationship describes the cells that need to be flagged for further testing after the reconstruction

$$\mathcal{C}_2^t = \text{in}(G_2, \mathcal{C}_2^{lc}) \cap R_2 \tag{4.14}$$

By obtaining the cells to be tested according to Equation (4.14), we ensure that any cell that has a direct path to any of the leaving cells are verified. The cells to be tested, that is $\mathcal{C}^t$ for the full dimensional problem can be obtained from $\mathcal{C}_2^t$ by following the set reconstruction procedure described earlier for R_2.

To illustrate how a cell is selected for testing, Figure 4.9 is presented. From the figure, it can be seen that the non-leaving cells of Subsystem 2 R_2 are $\{B_2, \cdots, B_6\}$ while the leaving cells $\mathcal{C}_2^{lc}$ are $\{B_8, B_9\}$. Also, the incoming neighbors of the leaving cells are $\{B_5, B_6\}$. Thus, $\mathcal{C}_{d,2}^t = \{B_5, B_6\}$.

The algorithm for the set validation is presented in Algorithm 6. The algorithm takes as input the full system model, the constraint set, the cells to be tested $\mathcal{C}^t$ and the reconstructed solution R. It returns the final validated solution R^*. A while loop is used in the validation algorithm since the order to conduct the tests for the cells in the testing set in not known in advanced. Hence, we check several times until no cell is removed.

Remark 4.1. *It is worth mentioning that the presence of both finite and infinite path passing through a particular cell may be caused by the effects of the actual system inputs and not the missing state. The effects of this two can be separated. However, in this work we take a conservative approach and treat the presence of both finite and infinite admissible path on a cell as an effect of the missing state.*

Algorithm 6: Set validation

Input: System (4.1), constraint sets X and U, $\mathcal{C}^t$, R
Output: R^*

```
cont = true
while cont do
    tmp ← ∅
    for B_i ∈ C^t do
        if F(B_i) ∩ R = ∅ then
            Add B_i to tmp
    if tmp ≠ ∅ then
        R ← R/tmp // Set difference
        C^t ← C^t/tmp
R* ← R
return R*
```

4.4.3 Computational complexity

In this section, we briefly analyze the computational complexity of the proposed algorithm. In particular, we focus on how the complexity of the algorithm increases in the system state dimension n. Consider that only two subsystems S_1, S_2 with state dimensions $n_1 < n, n_2 < n$ have been created and let n_c denote the number of interval divisions per state dimension.

Based on the results of [32], we know that the overall complexity of the graph-based control invariant set computation algorithm is determined by the graph construction step which is $\mathcal{O}(n_c^n)$. This implies that the algorithm increases exponentially in the state dimension n.

Owing to the decomposition, the computation of the control invariant sets for the each of the subsystems is $\mathcal{O}(n_c^{n_1})$ and $\mathcal{O}(n_c^{n_2})$ respectively which is less than $\mathcal{O}(n_c^n)$. Thus, the complexity of the decentralized and the distributed steps is determined by the subsystem with the largest dimension i.e. $\max(\mathcal{O}(n_c^{n_1}), \mathcal{O}(n_c^{n_2}))$. In both of the steps, the missing states are determined and stored in a dictionary prior to the computation and therefore retrieving it has a complexity of $\mathcal{O}(1)$. The centralized step is probably the most time consuming part of the algorithm since the control invariant set is reconstructed from the subsystem solution and validated. However, since only the cells flagged for testing $\mathcal{C}^t$ are tested, three possibilities can occur depending on the number of cells flagged. Let n_t be the number of cells flagged for testing after the distributed computation step. At worst, all the cells are flagged and therefore all the cells in the reconstruction step need to be validated. At the best case, no cell is flagged and therefore the overall control invariant set can be reconstructed without any validation. On the average case, not all the cells are flagged for validation. The complexity in this stage is therefore $\mathcal{O}(n_t)$ which is on average less than or at worst equal to $\mathcal{O}(n_c^n)$.

4.5 Examples

In this section we present three numerical examples to demonstrate the efficacy of the proposed algorithm. The first two are 3 dimensional linear and nonlinear examples. This is used to demonstrate that the solution converges to the solution of the full system model for both the linear and nonlinear case. This is because, it is computationally difficult to obtain an approximation of R_X for higher dimensional systems using the standard GIS algorithm. Finally, we compute an outer approximation of R_X for a 6 dimensional nonlinear system.

Unless otherwise stated, the number of division in each dimension was fixed at 128.The computation was performed on a laptop computer with Intel i7 CPU at 2.60 GHZ and 16 GB RAM. We refer to the solution from the decomposition as the distributed solution and that from the full system model as the centralized solution.

4.5.1 Linear system example

Consider the linear time-invariant cascade system

$$x^+ = Ax + Bu \tag{4.15}$$

where A and B are given by

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 2 & 0 \\ 0 & 1 & 2 \end{bmatrix} \text{ and } B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

The constraints on the states and input are $X = \{x \in \mathbb{R}^3 : \|x\|_\infty \leq 5\}$ and $U = \{u \in \mathbb{R} : \|u\|_\infty \leq 1\}$ respectively.

Our goal is to compute the control invariant set for the system. To use the proposed algorithm, the system is first expanded to a 4-dimensional system by repeating the state equation for the second state as shown in Equation (4.16).

$$z^+ = f(z,u) = \left[\begin{array}{cc|cc} 2 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ \hline 1 & 0 & 2 & 0 \\ 0 & 0 & 1 & 2 \end{array}\right] z + \left[\begin{array}{c} 1 \\ 0 \\ \hline 0 \\ 0 \end{array}\right] u \tag{4.16}$$

where $z = [x_1\ x_2\ x_2\ x_3]^T \in \mathbb{R}^4$. It can be seen that $n_z = 4 > n_x = 3$. Equation (4.16) is then decomposed to obtain two subsystems namely: subsystem one: $z_1 = (x_1, x_2)$ and subsystem

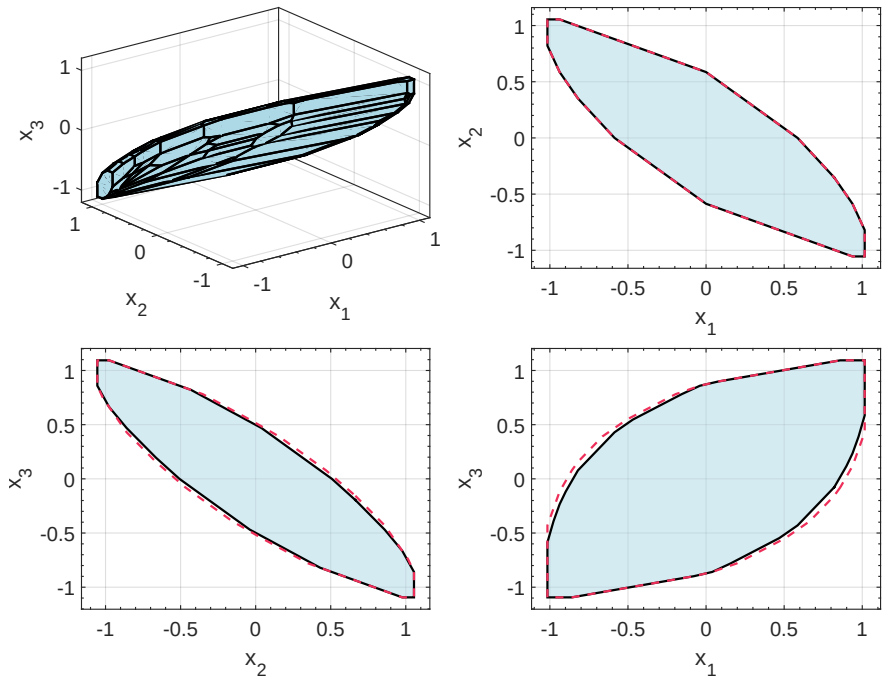


Figure 4.10: Comparison of the convex hull of the cells in the centralized and distributed solutions. Blue shaded area: centralized solution. Dashed red: solution of the sets obtained from the distributed computation. Black: Final solution after reconstructing the solution and validating the cells.

two: $z_2 = (x_2, x_3)$ as shown in Equations (4.17) and (4.18) below.

$$z_1^+ = f_1(z_1, u) = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} z_1 + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u = f_1(x_1, x_2, u) \tag{4.17}$$

$$z_2^+ = f_2(z_2, x_1) = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} z_2 + \begin{bmatrix} 1 \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u = f_2(x_2, x_3, x_1) \tag{4.18}$$

The control invariant sets for the two subsystems and ultimately the full system are computed using the distributed algorithm. The solution from the distributed algorithm is compared to the projections of the control invariant set computed using the centralized model. This is presented in Figure 4.10. It can be seen that the distributed computation converges to the centralized computation.

4.5.2 Nonlinear system example

Consider the nonlinear system

$$x_1^+ = x_1^2 + u \tag{4.19a}$$

$$x_2^+ = x_2^2 + x_1 \tag{4.19b}$$

$$x_3^+ = x_3^2 + x_2 \tag{4.19c}$$

The constraints on the states and input are $X = \{x \in \mathbb{R}^3 : \|x\|_\infty \leq 5\}$ and $U = \{u \in \mathbb{R} : \|u\|_\infty \leq 1\}$ respectively.

Our goal is to compute an approximation of the largest control invariant set for the system. To use the proposed algorithm, the system is first expanded to a 4-dimensional system by repeating the state equation for the second state as shown in Equation (4.20).

$$z^+ = f(z,u) = \left[\begin{array}{cc|cc} x_1^2 & 0 & 0 & 0 \\ x_1 & x_2^2 & 0 & 0 \\ \hline x_1 & 0 & x_2^2 & 0 \\ 0 & 0 & x_2 & x_3^2 \end{array}\right] + \left[\begin{array}{c} 1.0 \\ 0 \\ \hline 0 \\ 0 \end{array}\right] u \tag{4.20}$$

where $z = [x_1\ x_2\ x_2\ x_3]^T \in \mathbb{R}^{n_z}$ with $n_z = 4 > n_x = 3$. Thereafter, the expanded system is decomposed into two subsystems with overlapping states as described in the earlier sections.k

The results from the computations using the full system model and that of the decomposed system model are presented in Figure 4.11 and 4.12. Figure 4.11 shows the convex hull of the sets in the distributed solution and the centralized solution. Again, it can be seen that the solution using the decomposed models converges to that of the full system model after the reconstruction. Furthermore, Figure 4.12 show the scalabilities of the proposed distributed algorithm and the centralized algorithm. It can be seen that the algorithm using the decomposed model scales better than when the centralized model is used.

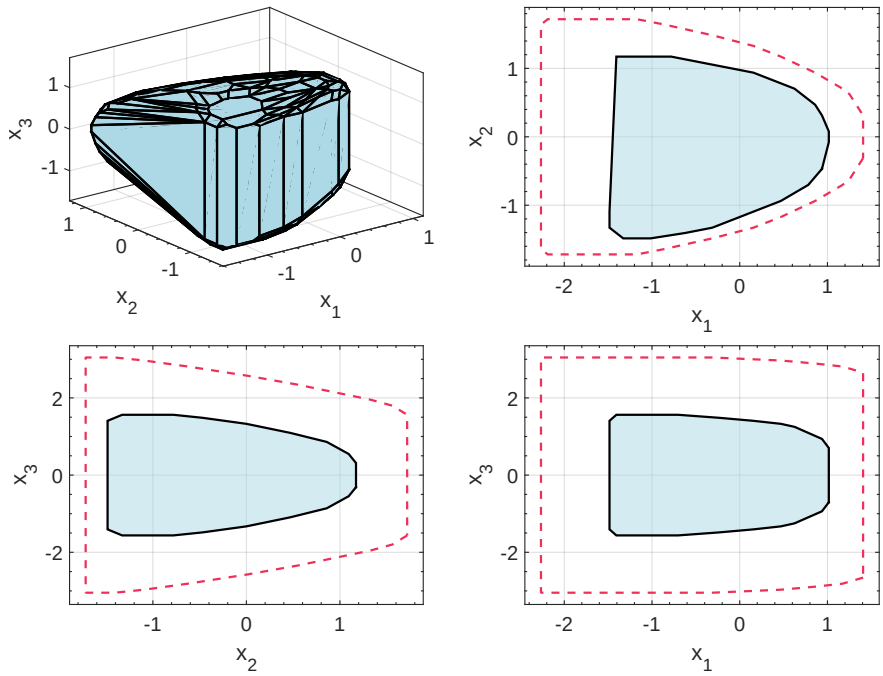


Figure 4.11: Comparison of the convex hull of the cells in the solutions utilizing the decomposed model and the full system model. Blue shaded area: Solution from the computation utilizing the full system model. Dashed red: solution of the sets obtained from the distributed computation. Black: Final solution after reconstructing the solution and validating the cells.

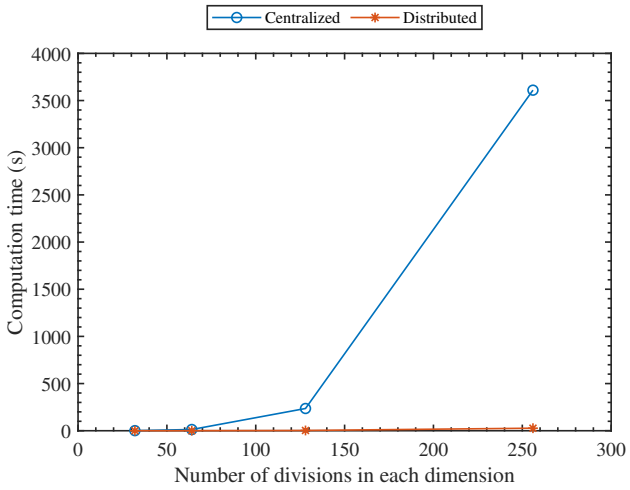


Figure 4.12: The computation vs the number of interval divisions in each dimension. At 256 interval divisions, it took more than an hour for the computation using the centralized model. An hour has been used for better visualization.

4.5.3 Three Continuously stirred tank reactors in series example

Let us consider an isothermal continuous-stirred tank reactor (CSTR) in which the following irreversible chained reactions occurs

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$

The reactor can be described by the following dimensionless modeling equations

$$\frac{dx_1}{dt'} = -x_1 + \mathrm{Da}_1 x_1 + u_1 \tag{4.21a}$$
$$\frac{dx_2}{dt'} = -x_2 + \mathrm{Da}_2 x_2^2 - \mathrm{Da}_1 x_1 \tag{4.21b}$$

where $\mathrm{Da}_1 = 1$ and $\mathrm{Da}_2 = 2$ represent the dimensionless Damkholer number for the reactions 1 and 2, x and u are the dimensionless state and input vectors and t' is dimensionless time. By connecting three of the CSTR model in series, the follow six dimensional model is obtained.

$$\frac{dx_1}{dt'} = -x_1 + \mathrm{Da}_1 x_1 + u_1 \tag{4.22a}$$
$$\frac{dx_2}{dt'} = -x_2 + \mathrm{Da}_2 x_2^2 - \mathrm{Da}_1 x_1 \tag{4.22b}$$
$$\frac{dx_3}{dt'} = -x_3 + \mathrm{Da}_1 x_3 + x_1 \tag{4.22c}$$
$$\frac{dx_4}{dt'} = -x_4 + \mathrm{Da}_2 x_4^2 - \mathrm{Da}_1 x_3 + x_2 \tag{4.22d}$$
$$\frac{dx_5}{dt'} = -x_5 + \mathrm{Da}_1 x_5 + x_3 \tag{4.22e}$$
$$\frac{dx_6}{dt'} = -x_6 + \mathrm{Da}_2 x_6^2 - \mathrm{Da}_1 x_5 + x_4 \tag{4.22f}$$

The continuous-time model is discretized using a step size of 1 before usage in CIS algorithm. The constraints on the states and input are $X = \{x \in \mathbb{R}^6 : 0 \leq x \leq 1\}$ and $U = \{u \in \mathbb{R} : 0 \leq u \leq 1\}$ respectively.

Figures 4.13–4.15 show the results of each reactor. The computations were performed on

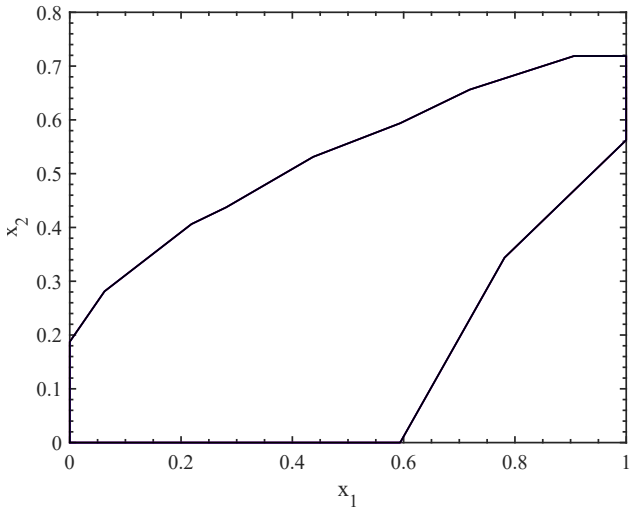


Figure 4.13: The solution for Reactor 1.

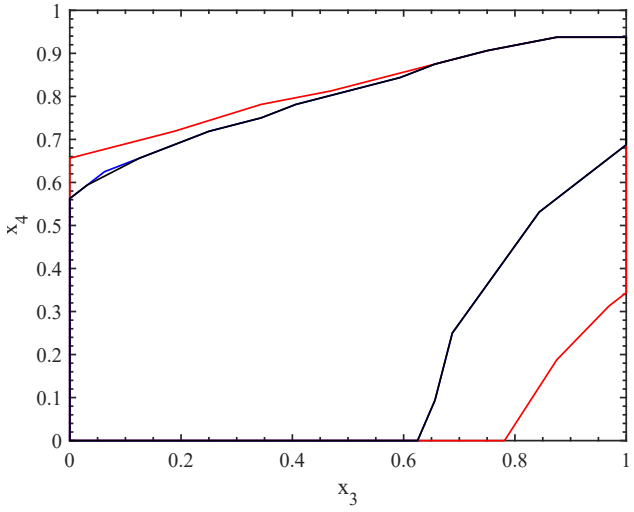


Figure 4.14: The solution for Reactor 2. Red: results from decentralized computation. Blue: results from the distributed computation. Black: Final solution after validation

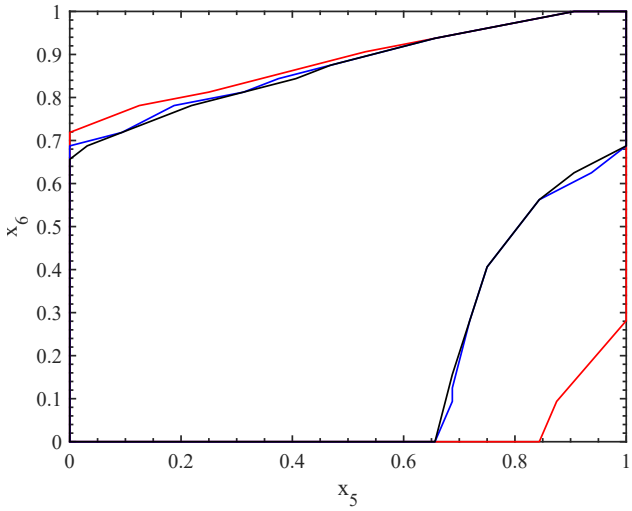


Figure 4.15: The solution for Reactor 3. Red: results from decentralized computation. Blue: results from the distributed computation. Black: Final solution after validation

a computing server with single core Intel Xeon E5 processor with 2.1 GHz frequency and 100 GB of RAM. It took roughly 4 hours to compute the solution using the distributed approach with the reconstruction and validation step taking about 70 % of the total computation time. It is worth mentioning that it is intractable to compute an approximation of the largest control invariant set for the six dimensional system using the GIS algorithm presented in Chapter 2 without system decomposition.

4.6 Concluding remarks

Given a constrained discrete-time time-invariant control system, the method proposed in this chapter obtains outer approximations of the largest control invariant set contained in the state constraint. For this purpose, we presented a graph-based distributed algorithm which approximates the dynamics of the control system as a directed graph allowing for analysis of the system using graph theory. This algorithm paves a promising way to overcome the "curse of dimensionality" encountered in control invariant set computation. Simulations using several numerical examples demonstrate the ability and effectiveness of the proposed method to approximate the largest control invariant set, just like the centralized algorithm.

Chapter 5

Robust economic model predictive control with zone tracking

5.1 Introduction

Nonlinear model predictive control (MPC) with a general objective known as economic MPC (EMPC) has received significant attention in recent years [55, 56, 57]. The objective function in an EMPC generally reflects some economic performance criterion such as profit maximization or heat minimization. This is in contrast with the tracking MPC where the objective is a positive definite quadratic function. The integration of process economics directly in the control layer makes EMPC of interest in many areas especially in the process industry. There has been a significant number of applications of EMPC [58, 59, 60, 61]. To address stability and computational issues of EMPC, different formulations of EMPC has been proposed [62, 56, 57, 63].

Uncertainties arise as a result of imperfect models and/or unmeasured disturbances. The presence of uncertainties in any control system can result in performance degradation and/or loss of feasibility which can lead to loss of stability. Due to the integration of process economics in the control layer, it is not fully understood how the presence of uncertainties

affect the economic performance of EMPC. In the context of tracking MPC, robust MPC is a common approach used to address the robustness of a control system in the presence of uncertainties. See [64] for a recent survey on robust MPC as well as the associated challenges. Robust MPC techniques have also been applied to EMPC in the literature. In [65], an EMPC formulation which is based on robust tracking of a prior nominal trajectory was proposed. In [66], a robust EMPC formulation based on scenario tree approach was presented. In [67], a min-max robust EMPC algorithm was proposed to address transmission delays in networked control systems. Tube-based formulations with and without stochastic information have also been proposed [68, 69, 70]. However, they either use a min-max optimization approach or use the nominal model with tightened invariant constraints. In both cases, the computational demands are very high even for linear systems.

While robust MPC techniques are common in the design of tracking MPCs for handling uncertainty, it was pointed out that simply transferring robust MPC techniques to EMPC could result in poor economic performance [68]. This is because economic optimization and robustness are two objectives and often may conflict with each other. Robust MPC techniques have been designed to reject all disturbances to achieve their desired goal which may not be the case in EMPC as some disturbances can lead to better economic performance. In our previous work, we proposed an EMPC with zone tracking scheme to handle the two objectives in one integrated framework [12]. The use of a target zone allows for flexible handling of multiple objectives in the controller design and at the same time improves the degree of robustness of the controller due to the inherent robustness of zone control. It is worth noting that the concept of zone control is not new. Zone MPC have been reported in several areas such as diabetes treatment [71], control of building heating system [72], control of irrigation systems [73] and coal-fired boiler-turbine generating system [60]. In the context of MPC literature, zone control is often dismissed as a trick to avoid feasibility issues and has received less attention in terms of theoretical analysis. A recent study on the stability analysis of MPC with generalized zone tracking [13] paves the way for further development

of zone control.

In [12], the stability and economic performance of the EMPC with zone tracking framework were studied without considering process uncertainty. In this chapter, we extend [12] to consider constrained nonlinear systems subject to unmeasured but bounded disturbances. Instead of tracking the original target zone, we propose to track a robust control invariant set within the target zone so that once the system state enters the invariant zone, it will not exit the target zone anymore. The proposed design can ensure that the zone tracking objective is achieved in finite steps and at the same time optimizes the economic performance. It is found that in the presence of uncertainty, the economic performance of EMPC not only depends on the optimal steady state but also the size of the tracked zone. To take this into account, we introduce the notion of risk factor in the controller design. The risk factor determines the conservativeness of the controller and provides a way to tune the EMPC for better economic performance. An algorithm to determine the zone integrating the risk factor is also proposed. A nonlinear chemical example is presented to demonstrate the performance of the proposed formulation.

5.2 Preliminaries

5.2.1 Notation

Throughout this chapter, the symbol $\mathbb{I}_{\geq 0}$ denotes the set of nonnegative integers $\{0, 1, 2, \ldots\}$. $\mathbb{I}_M^N$ is the set of integers from M to N: $\mathbb{I}_M^N = \{M, M+1, \ldots N\}$. $|\cdot|$ denotes the Euclidean norm of a scalar or a vector. $\|\cdot\|_n$ denotes the n-norm of a scalar or vector. A continuous function $\alpha : [0, a) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and satisfies $\alpha(0) = 0$. A class $\mathcal{K}$ function α is called a class $\mathcal{K}_\infty$ function if α is unbounded. A continuous function $\sigma : [0, \infty) \to [0, a)$ is said to belong to class $\mathcal{L}$ if it is strictly decreasing and satisfies $\lim_{x\to\infty}\sigma(x) = 0$. A continuous function $\beta : [0, a) \times [0, \infty) \to [0, \infty)$ is said to belong to class $\mathcal{KL}$ if for each fixed r, $\beta(r, s)$ is a class $\mathcal{L}$, and for each fixed s, $\beta(r, s)$ is a $\mathcal{K}$ function. The

operator '/' means set subtraction such that $\mathbb{A}/\mathbb{B} = \{x \in R^{n_x} : x \in \mathbb{A}, x \notin \mathbb{B}\}$.

5.2.2 System description and control problem formulation

In this chapter, we consider discrete-time nonlinear systems described by the following state-space model:

$$x(n+1) = f(x(n), u(n), w(n)) \tag{5.1}$$

where $x(n) \in \mathbb{R}^{n_x}$ is the system state vector at time instant $n \in \mathbb{I}_{\geq 0}$, $u(n) \in \mathbb{R}^{n_u}$ is the control input vector and $w(n) \in \mathbb{R}^{n_w}$ denotes the system disturbance vector. It is assumed that the system state and the control input vectors are restricted to be in the coupled non-empty convex set of the following form:

$$(x(n), u(n)) \in \mathbb{Z} \subseteq \mathbb{X} \times \mathbb{U} \tag{5.2}$$

where $\mathbb{X}$ and $\mathbb{U}$ are the constraints on the state and the input respectively. It is also assumed that the disturbance is unknown and contained in a set $\mathbb{W}$ ($w \in \mathbb{W}$) where

$$\mathbb{W} := \{w \in \mathbb{R}^{n_w} : \|w\|_\infty \leq \theta, \theta > 0\}$$

with θ being a positive real number. Throughout this paper, we make the following assumptions.

Assumption 5.1 (Compact constraints)**.** *The sets $\mathbb{X}$, $\mathbb{U}$ and $\mathbb{W}$ are compact with $\mathbb{W}$ containing the origin in its interior.*

Assumption 5.2 (Continuity)**.** *The function $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ is locally Lipschitz with respect to x and w for all $x \in \mathbb{Z}$, $u \in \mathbb{U}$, $w \in \mathbb{W}$. This implies that there exist positive constants L_x and L_w such that:*

$$|f(x, u, w) - f(z, u, 0)| \leq L_w|w| + L_x|x - z| \tag{5.3}$$

for all $x, z \in \mathbb{X}$, $u \in \mathbb{U}$ *and* $w \in \mathbb{W}$.

The primary control objective of this work is to design a feedback controller such that it can drive the state of system (5.1) to a pre-determined target zone $\mathbb{X}_t \subset \mathbb{X}$ if the initial state of system (5.1) is outside the target zone ($x(0) \in \mathbb{X}/\mathbb{X}_t$) and maintain the state of system (5.1) within the target set $\mathbb{X}_t$ when the zone tracking is achieved. A secondary objective is to minimize the average economic cost over the infinite horizon T characterized as follows:

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{n=0}^{T-1} \ell_e(x(n), u(n)) \tag{5.4}$$

where $\ell_e : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ is a general economic stage cost which is not necessarily quadratic or positive definite but it must be continuous. In order to achieve the above control objectives, we resort to EMPC with zone tracking [12] and takes into account the presence of process disturbance w in the design of the EMPC.

5.3 Robust EMPC with zone tracking

In this section, we present the design of the proposed robust EMPC with zone tracking scheme. The proposed design steers the system state to the target zone and optimizes the economic objective in the process.

Given that there is model uncertainty due to the presence of process disturbances and that the target zone $\mathbb{X}_t$ is not necessarily control invariant, a robust control invariant set $\mathbb{X}_e$ within the target zone ($\mathbb{X}_e \subseteq \mathbb{X}_t$) is determined and is used as the actual tracking zone in the proposed EMPC design [13]. To optimize the economic objective within the robust control invariant set, the set $\mathbb{X}_e$ is optimized also according to the economic objective. In the remainder of this work, we will refer to this robust control invariant set $\mathbb{X}_e$ as the economic zone.

Let us first assume that such an economic zone $\mathbb{X}_e$ has been determined. The procedure to create such an economic zone will be discussed in section 5.3.2. It is also assumed that there

is a steady state (x_s, u_s) with $x_s \in \mathbb{X}_e$, $u_s \in \mathbb{U}$ such that it solves the following steady-state optimization problem:

$$(x_s, u_s) = \arg\min \ \ell_e(x, u) \tag{5.5a}$$

$$s.t. \quad x = f(x, u, 0) \tag{5.5b}$$

$$(x, u) \in \mathbb{X}_e \times \mathbb{U} \tag{5.5c}$$

Without loss of generality, we assume that (x_s, u_s) is the unique solution to the above steady-state optimization problem.

5.3.1 Design of the proposed EMPC with zone tracking

With information about the current state $x(n)$, the proposed EMPC uses the nominal model of system (5.1):

$$z(k+1) = f(z(k), v(k), 0) \tag{5.6}$$

with the initial condition $z(0) = x(n)$ to find a control sequence $\mathbf{v} = \{v(0), \ldots, v(N-1)\}$ and the associated state sequence $\mathbf{z} = \{z(0), \ldots, z(N)\}$ over the entire prediction horizon N to minimize the cost function:

$$V_N(x(n), \mathbf{v}) = \sum_{k=0}^{N-1} \ell(z(k), v(k)) \tag{5.7}$$

In (5.6) and (5.7), $z(n) \in \mathbb{X} \subseteq \mathbb{R}^{n_x}$ and $v(n) \in \mathbb{U} \subseteq \mathbb{R}^{n_u}$ are the nominal state vector and computed control input vector respectively in the proposed EMPC. The stage cost $\ell(\cdot, \cdot)$ is defined as follows:

$$\ell(z, v) = \ell_e(z, v) + \ell_z(z) \tag{5.8}$$

where $\ell_e(\cdot,\cdot)$ is the economic stage cost as introduced in (5.4) and $\ell_z(\cdot)$ is a zone tracking penalty term which is defined as below:

$$\ell_z(z) = \min_{z^z} \quad c_1(\|z - z^z\|_1) + c_2(\|z - z^z\|_2^2) \tag{5.9a}$$

$$s.t. \quad z^z \in \mathbb{X}_e \tag{5.9b}$$

with $c_1 \in \mathbb{R}_{\geq 0}$, $c_2 \in \mathbb{R}_{\geq 0}$ being non-negative weights on the l_1 norm and the squared l_2 norm respectively, z^z is a slack variable and $\mathbb{X}_e$ is the economic zone to be tracked. The zone tracking stage cost reflects the distance of the system states from the economic zone and is positive definite. The weights c_1 and c_2 must be appropriately selected such that the zone tracking cost is given a higher priority than the economic objective. The incorporation of both the l_1 and the squared l_2 norms penalizes the magnitude and duration of the zone tracking violation [12].

At each sampling time, the following dynamic optimization problem $\mathcal{P}_N(x(n))$ is solved:

$$\min_{\mathbf{v}} \quad V_N(x(n), \mathbf{v}) \tag{5.10a}$$

$$s.t. \quad z(k+1) = f(z(k), v(k), 0), \quad k = 0, \ldots, N-1 \tag{5.10b}$$

$$z(0) = x(n) \tag{5.10c}$$

$$z(k) \in \mathbb{X}, \quad k = 0, \ldots, N-1 \tag{5.10d}$$

$$v(k) \in \mathbb{U}, \quad k = 0, \ldots, N-1 \tag{5.10e}$$

$$z(N) = x_s \tag{5.10f}$$

In the optimization problem (5.10) above, Equation (5.10c) is the initial state constraint, Equation (5.10f) is a terminal equality constraint and Equations (5.10d) and (5.10e) are the constraints on the state and inputs respectively. As a result of the cost function employed, the optimization problem in (5.10) is a multi-objective optimization problem which seeks to

minimize the deviation of the system's state from the economic zone $\mathbb{X}_e$ while optimizing the economic objective.

The solution of $\mathcal{P}_N(x(n))$ denoted $\mathbf{v}^*$ gives an optimal value of the cost $V_N^0(x(k))$ and at the same time $u(n) = v^*(0)$ is applied to the actual system (5.1). Notice that the nominal system is used in the optimization and therefore generates mismatch between the prediction in the EMPC optimization and the actual system evolution. We will show in the next section that under some mild conditions, our proposed controller is able to stabilize the plant in the presence of this mismatch. The prediction horizon is shifted forward by one sampling time once information about $x(n+1)$ is known and the optimization problem $\mathcal{P}_N(x(n+1))$ is solved to find $u(n+1)$.

5.3.2 Construction of the economic zone

In the previous section, we have presented the proposed EMPC formulation with zone tracking. In the proposed design, a robust control invariant economic zone $\mathbb{X}_e$ replaces the original target zone and is the zone to be tracked. In this section, we discuss how to construct the economic zone.

5.3.2.1 Risk factor

In determining the economic zone $\mathbb{X}_e$, the idea is to find a robust control invariant set within the original target zone $\mathbb{X}_t$ while taking into account the economic performance of the system within the control invariant set. The use of a robust control invariant set as the actual tracking zone ensures that the system state converges to the zone and will not leave the zone again once enters the invariant zone even in the presence of disturbances. This will be shown in the stability analysis section.

While any robust control invariant set within the original target zone can achieve the zone tracking objective, the size of the robust control invariant set affects the economic performance of the system. Due to the presence of disturbance in the system, the overall

economic performance of the system not only depends on the optimal steady state within the zone $\mathbb{X}_e$ but also depends on the economic performance of the system within the zone. In order to account for this in determining the economic zone, we introduce the concept of risk factor $\delta \in \mathbb{R}$ in the $\mathbb{X}_e$ construction. The risk factor is a positive scalar that can be tuned. It determines the size of the economic zone and ultimately, the conservativeness of the controller. Here, conservative is used to mean the ability of the controller to optimize the economic objective within the target zone. When a higher risk factor is used, the size of the economic zone is larger and the controller is less conservative. Algorithm 8 presented in the next subsection will summarize how the risk factor is used in the computation of the economic zone.

5.3.2.2 Computing the economic zone

The algorithm for determining the economic zone builds on the graph-based robust control invariant set computing algorithm developed in [32]. In the algorithm in [32], the state space $\mathbb{X}$ is quantized into closed sets $B_i, i = 1, \dots l$. The collection of these cells, $\mathcal{C} = \{B_1, \dots, B_l\}$, is called the finite covering of the state space $\mathbb{X}$. The closed sets in the finite covering $\mathcal{C}$ are also known as cells or boxes such that:

$$\mathbb{X} \subseteq \cup_{B_i \in \mathcal{C}} B_i \tag{5.11}$$

The diameter of the covering $\mathcal{C}$ is given by

$$\text{diam}(\mathcal{C}) := \max_{B_i \in \mathcal{C}} \text{diam}(B_i)$$

where $\text{diam}(B_i) = \sup\{|x - y|\colon x, y \in B_i\}$. Since $\mathbb{X}$ is compact, it is always possible to obtain a finite covering. Following the quantization, the system dynamics is approximated using a directed graph G. Graph investigations are then carried out on the directed graph to determine the cells that approximate the largest robust control invariant set while the ones

Algorithm 7: Determination of economic zone

Input: f, $\mathbb{X}_t$, $\mathbb{U}$, $\mathbb{W}$, ℓ_e, δ, d
Output: $\mathbb{X}_e$

1 Create a finite covering $\mathcal{C}_t$ of $\mathbb{X}_t$ with cell diameter d
2 Initialize the cells that satisfy the economic criterion $\mathcal{C}_e$ as empty array
3 **for** B_i *in* $\mathcal{C}_t$ **do**
4 **if** $\forall x \in B_i, \exists u \in \mathbb{U} : \ell_e(x + f(x,u,w) - f(x,u,0), u) \leq \delta,\ \forall w \in \mathbb{W}$ **then**
5 Add B_i to $\mathcal{C}_e$
6 **if** $\mathcal{C}_e$ *is empty* **then**
7 **return** $\emptyset$
8 **else**
9 Initialize Algorithm 2 in [32] with $\mathcal{C}_e$
10 Compute an inner approximation of the largest robust control invariant set $\mathcal{C}_r$ contained in $\mathcal{C}_e$
11 $\mathbb{X}_e \leftarrow \cup_{B_i \in \mathcal{C}_r} B_i$
12 **return** $\mathbb{X}_e$

that do not form part of the robust control invariant set are discarded.

We denote by $\mathcal{C}_r$ the cells that approximate the robust control invariant set.

The procedure for determining the economic zone $\mathbb{X}_e$ is summarized in Algorithm 7. The algorithm has a few inputs including the system model f, the risk factor δ, the initial cell diameter d, the target zone $\mathbb{X}_t$, the economic objective ℓ_e as well as the input and the disturbance sets. The algorithm returns the calculated economic zone $\mathbb{X}_e$.

Intuitively, the algorithm seeks to find an economic zone that compensates for the effects of the disturbances on the economics of the closed-loop system while ensuring good stability property. This is achieved by backing-off from the boundaries of the target zone to obtain $\mathbb{X}_e$. The algorithm is in two main stages. In the first stage of the algorithm (Lines 1 – 5), the target zone is quantized with the help of a finite covering $\mathcal{C}_t$. The initial diameter d of the cells should be selected such that it sufficiently captures how the economic objective changes with the state. The set of cells $\mathcal{C}_e$ within the target zone $\mathbb{X}_t$ that satisfy the economic criterion is then determined. Consider a cell $B_i \in \mathcal{C}_t$, if

$$\forall x \in B_i, \exists u \in \mathbb{U} : \ell_e(x + f(x,u,w) - f(x,u,0), u) \leq \delta,\ \forall w \in \mathbb{W} \tag{5.12}$$

then the cell B_i is added to $\mathcal{C}_e$. This is repeated until all the cells are checked. The remainder of the cells in $\mathcal{C}_t$ are then discarded. The choice of the selection criterion in Algorithm 7 stems from the fact that every state within the target zone is a potential initial state as well as a potential end state after one time-step. We focus on the latter since our proposed controller does not consider the effects of the disturbance. The idea is that, for any potential end state given by the nominal system, we know that the disturbance will be applied in the real system. Thus, we are taking into consideration the effects of the disturbance on the economics implicitly. By considering the end state in the selection criterion, we want to guarantee that the economic performance of the closed-loop system is bounded above by the risk factor δ irrespective of the disturbance w.

It is worth mentioning that the set formed by the union of the cells in $\mathcal{C}_e$ is not necessarily robust control invariant. The second stage (Lines 5 – 8) in Algorithm 7, therefore seeks to address this by finding the cells in $\mathcal{C}_e$ that inner approximate the largest robust control invariant set. This is achieved by initializing Algorithm 2 in [32] with $\mathcal{C}_e$ and then using the algorithm to find the cells that inner approximate the largest robust control invariant set $\mathcal{C}_r$ contained in the set formed by $\mathcal{C}_e$. The economic zone $\mathbb{X}_e$ is obtained after the two stages of Algorithm 7 are carried out.

Remark 5.1. *$\mathbb{X}_e$ is not guaranteed to exist. It depends on the chosen risk factor δ as well as the properties of the system. Thus, it is possible for Algorithm 7 to return an empty set. An obvious possibility is if the target zone $\mathbb{X}_t$ does not contain a robust control invariant set. This ultimately implies that it is impossible to keep the states of the system within the target zone and can therefore not be tracked. To understand why it is so, the reader may refer to the stability analysis in [13]. Another possibility is if a very small risk factor δ is selected such that it is impossible to satisfy condition (12). This will lead to $\mathcal{C}_e$ in Algorithm 1 being empty and subsequently the economic zone. Hence, the risk factor must be appropriately selected to ensure the existence of the economic zone $\mathbb{X}_e$.*

Remark 5.2. *The cells in $\mathcal{C}_r$ needs to be combined and represented in a way that makes the optimization problem presented in (5.10) easier to solve. One of such representations is to find an inner approximation convex hull of the cells in $\mathcal{C}_r$ if the cells form a convex set. Another approach is to use a more general set representation such as alpha hull. However, this may lead to the use of non-convex sets in (5.10) which can increase the complexity of the optimization problem.*

Remark 5.3. *As a result of the presence of the disturbances, it is in general difficult to determine the optimal control inputs. One approach to determine the optimal feedback control law is to solve a min-max optimization problem [64]. However, it suffers from high computational demand which makes it challenging to implement. In this paper, we propose an EMPC scheme based on only the nominal model and zone tracking. The zone to be tracked can be considered as an economic trust region. This makes our proposed approach similar to other trust-region based approaches such as the Lyapunov-based EMPC [74] and that presented in [61]. However, in the proposed formulation, we do not make use of any additional constraints such as Lyapunov constraints in the formulation. Moreover, our formulation introduces economic risk factor in the controller design thus implicitly considers an upper bound on the asymptotic average performance of the closed-loop system.*

5.4 Stability analysis

In this section, we address the stability of the proposed control algorithm. To proceed with the discussion, we first introduce a few relevant definitions and assumptions.

First, we define the N-step reachable set of the optimal steady-state x_s based on the nominal model. The N-step reachable set will be used to construct a set for the initial state of the system to ensure the feasibility of the proposed EMPC.

Definition 5.1 (N-step reachable set)**.** *Consider the nominal system of system (5.1) (i.e., $w \equiv 0$ for all time). A set $\mathbb{X}_N$ is called the N-step reachable set with respect to the steady-*

state x_s if it contains all the states that can be steered to x_s in N steps while satisfying the state and input constraints. That is,

$$\mathbb{X}_N = \{x(0) \in \mathbb{X} | \exists (x(n), u(n)) \in \mathbb{Z}, n \in \mathbb{I}_0^{N-1}, \textit{such that } x(N) = x_s\} \tag{5.13}$$

Assumption 5.3. *The N-step reachable set $\mathbb{X}_N$ is a compact set with x_s in the interior of set.*

Next, we introduce the definition of dissipative systems and the relevant assumptions. These definition and assumptions will be used to establish the stability of the proposed EMPC.

Definition 5.2 (Strictly dissipative systems)**.** *The nominal system $\tilde{x}(n+1) = f(\tilde{x}(n), u(n), 0)$ is strictly dissipative with respect to the supply rate $s : \mathbb{X} \times \mathbb{U} \to \mathbb{R}$ if there exists a continuous storage function $\lambda(\cdot) : \mathbb{X} \to \mathbb{R}$ and a $\mathcal{K}_\infty$ function $\alpha(\cdot)$ such that the following hold for all $\tilde{x} \in \mathbb{X}$ and $u \in \mathbb{U}$:*

$$\lambda(f(\tilde{x}, u)) - \lambda(\tilde{x}) \leq s(\tilde{x}, u) - \alpha(|\tilde{x} - x_s|) \tag{5.14}$$

Assumption 5.4 (Strict disspativity)**.** *The nominal system $\tilde{x}(n+1) = f(\tilde{x}(n), u(n), 0)$ is strictly dissipative with respect to the supply rate*

$$s(\tilde{x}, u) = \ell_e(\tilde{x}, u) - \ell_e(x_s, u_s)$$

Assumption 5.5 (Weak controllability)**.** *There exists a $\mathcal{K}_\infty$ function $\gamma(\cdot)$ such that for all $x \in \mathbb{X}_N$, there exists a feasible solution to (5.10) such that $\sum_{k=0}^{N-1} |v(k) - u_s| \leq \gamma(|x - x_s|)$.*

The following proposition provides an upper bound on the deviation of the nominal system state trajectory from the uncertain system state trajectory when the same input sequence is applied.

Proposition 5.1. *Consider the following system*

$$x(n+1) = f(x(n), u(n), w(n)) \tag{5.15}$$

and the corresponding nominal system

$$\tilde{x}(n+1) = f(\tilde{x}(n), u(n), 0) \tag{5.16}$$

with the initial condition $x(n) = \tilde{x}(n) \in \mathbb{X}$. *The deviation of the nominal system state* $\tilde{x}$ *from the state* x *over one sampling time is bounded as follows:*

$$|x(n+1) - \tilde{x}(n+1)| \leq \sqrt{n_x} L_w \theta \tag{5.17}$$

for all $x(n), \tilde{x}(n) \in \mathbb{X}$ *and all* $w(n) \in \mathbb{W}$.

Proof. Let us define the deviation of $\tilde{x}$ from x as e such that $e = x - \tilde{x}$. Therefore, $e(n+1) = x(n+1) - \tilde{x}(n+1)$, which can further be written as:

$$e(n+1) = f(x(n), u(n), w(n)) - f(\tilde{x}(n), u(n), 0) \tag{5.18}$$

Taking the Euclidean norm of the error e and applying (5.3), the following inequality is obtained

$$|e(n+1)| \leq L_w |w(n)| + L_x |x(n) - \tilde{x}(n)| = L_w |w(n)| + L_x |e(n)| \tag{5.19}$$

for all $x(n), \tilde{x}(n) \in \mathbb{X}$ and $w(n) \in \mathbb{W}$. Since the initial state for both the nominal and the uncertain system are the same i.e. $x(n) = \tilde{x}(n)$, we have that the initial deviation is 0, i.e. $e(n) = 0$. Given that $\|w\|_\infty \leq \theta$, $|w| \leq \sqrt{n_x}\theta$. This leads to (5.17) and proves Proposition 5.1. □

We now state the main results of this section. Theorem 5.1 considers the nominal system of system (5.1) and finds a Lyapunov function of the system with respect to the steady

state x_s. Theorem 5.2 will use this Lyapunov function to study the uncertain system to establish the feasible region, finite step convergence, and ultimate stability and robustness of the proposed EMPC.

Theorem 5.1. *Consider the nominal system of system (5.1) under the control of EMPC (5.10). Suppose that Assumption 5.4 holds and $\lambda(\cdot)$, $\alpha(\cdot)$, $s(\tilde{x},u)=\ell_e(\tilde{x},u)-\ell_e(x_s,u_s)$ are the associated functions that satisfy the condition (5.14) for all $\tilde{x}\in\mathbb{X}$ and $u\in\mathbb{U}$. Define the rotated cost as follows:*

$$\tilde{\ell}_e(\tilde{x},u)=\ell_e(\tilde{x},u)-\ell_e(x_s,u_s)+\lambda(\tilde{x})-\lambda(f(\tilde{x},u,0)). \tag{5.20}$$

Then, the following dynamical optimization problem is equivalent to the proposed EMPC (5.10):

$$\min_{\mathbf{v}} \quad \tilde{V}_N(\tilde{x}(n),\boldsymbol{v})=\sum_{k=0}^{N-1}\Big(\tilde{\ell}_e(z(k),v(k))+\ell_z(z(k))\Big) \tag{5.21a}$$

$$s.t. \quad (5.10b)-(5.10f) \tag{5.21b}$$

If Assumption 5.5 also holds, then the value function of (5.21) denoted as $\tilde{V}_N^0(\cdot)$ is a Lyapunov function of the closed-loop system under the control of EMPC (5.10) with respect to the optimal steady-state x_s

Proof. In this proof, we use $\tilde{x}$ to denote the state of the nominal system under the control of the proposed EMPC. Based on the definition of the rotated cost as in (5.20) and the condition (5.14), it can be concluded that the rotated cost is bounded from below for all $(\tilde{x},u)\in\mathbb{Z}$ as follows:

$$\tilde{\ell}_e(\tilde{x},u)\geq\alpha(|\tilde{x}-x_s|) \tag{5.22}$$

Based on the definition of $\ell(\cdot,\cdot)$ in (5.8) and the rotated cost in (5.20), the stage cost $\ell(z,u)$

can be equivalently expressed as follows:

$$\ell(z,v) = \tilde{\ell}_e(z,v) + \ell_z(z) + \ell_e(x_s,u_s) - \lambda(z) + \lambda(f(z,v,0)) \tag{5.23}$$

Based on the above expression of $\ell(\cdot,\cdot)$, the cost function $V_N(\cdot)$ in the optimization problem (5.10) at time n can be equivalently expressed as follows:

$$V_N(\tilde{x}(n),\mathbf{v}) = \sum_{k=0}^{N-1}\Big(\tilde{\ell}_e(z(k),v(k)) + \ell_z(z(k))\Big) - \lambda(z(0)) + \lambda(z(N)) + N\ell_e(x_s,u_s) \tag{5.24}$$

Taking into account the constraint (5.10f) in EMPC (5.10), the last three terms in the above expression of $V_N(\cdot,\cdot)$ are constants. This implies that if we replace the cost function $V_N(\cdot,\cdot)$ in the EMPC optimization problem (5.10) with the new cost function as in (5.21a), the solution of the EMPC optimization problem remains the same. That is, the original EMPC (5.10) is equivalent to the new EMPC (5.21). Let us denote the optimal value of the cost function (the value function) of the new EMPC (5.21) as $\tilde{V}_N^0(\tilde{x}(n))$. Taking into account (5.22) and the expression of $\tilde{V}_N(\tilde{x}(n),\mathbf{v})$, and noticing that $z(0) = \tilde{x}(n)$ in the EMPC optimization problem, it can be obtained that:

$$\tilde{V}_N^0(\tilde{x}(n)) \geq \tilde{\ell}_e(z(0),u(0)) + \ell_z(z(0)) \geq \tilde{\ell}_e(z(0),u(0)) \geq \alpha(|\tilde{x}(n) - x_s|) \tag{5.25}$$

From Assumption 5.5, there exists a $\beta(\cdot) \in \mathcal{K}_\infty$ such that for all $\tilde{x}(n) \in \mathbb{X}_N$ (see Appendix of [75]):

$$\tilde{V}_N^0(\tilde{x}(n)) \leq \beta(|\tilde{x}(n) - x_s|) \tag{5.26}$$

For the nominal system, it can be shown that the value function $\tilde{V}_N^0(\cdot)$ is non-increasing and satisfies the following condition:

$$\tilde{V}_N^0(\tilde{x}(n+1)) - \tilde{V}_N^0(\tilde{x}(n)) \leq -\tilde{\ell}_e(\tilde{x}(n),u(n)) - \ell_z(\tilde{x}(n),u(n)) \leq -\alpha(|\tilde{x}(n) - x_s|) \tag{5.27}$$

This makes the value function $\tilde{V}_N^0(\cdot)$ a Lyapunov function of the closed-loop system under the control of EMPC (5.10) with respect to the optimal steady state x_s. This proves Theorem 5.1.

□

Before presenting Theorem 5.2, we introduce the set Ω_ρ defined based on the level set of the Lyapunov function $\tilde{V}_N^0(\cdot)$:

$$\Omega_\rho = \{x \in \mathbb{X} : \tilde{V}_N^0(x) \leq \rho\}. \tag{5.28}$$

Let us assume that the value function $\tilde{V}_N^0(x)$ is differentiable with respect to x. Based on the above definition, we also define $\Omega_{\rho_{\min}}$ as follows:

$$\Omega_{\rho_{\min}} := \max\{\tilde{V}_N^0(x(n+1)) : |x(n) - x_s| \leq \alpha^{-1}(K_V L_w \sqrt{n_x}\theta + H n_x L_w^2 \theta^2)\} \tag{5.29}$$

where K_V is a positive constant that bounds the the magnitude of the partial derivative $\frac{\partial \tilde{V}_N^0(x)}{\partial x}$ such that $|\frac{\partial \tilde{V}_N^0(x)}{\partial x}| \leq K_V$ for all $x \in \mathbb{X}_N$, and H is the constant associated with the Taylor expansion of $\tilde{V}_N^0(x)$ (which will be made clearer in the proof of Theorem 5.2). The size of $\Omega_{\rho_{min}}$ is determined by the properties of the system and the disturbance set. Further, we denote the maximum level set within $\mathbb{X}_N$ as $\Omega_{\rho_{\max}}$.

Theorem 5.2. *Consider system (5.1) in closed-loop with EMPC (5.10). Let the target zone and the economic zone satisfy:* $\Omega_{\rho_{\min}} \subset \mathbb{X}_e \subset \mathbb{X}_t \subset \Omega_{\rho_{\max}} \subset \mathbb{X}$. *If Assumptions 5.2 – 5.5 hold, the magnitude of the partial derivative* $\frac{\partial \tilde{V}_N^0(x)}{\partial x}$ *is upper bounded such that* $|\frac{\partial \tilde{V}_N^0(x)}{\partial x}| \leq K_V$ *for all* $x \in \mathbb{X}$, *and if there exist* $\epsilon_s > 0$, $\rho_s > 0$ *such that:*

$$-\alpha(\rho_s) + K_V L_w \sqrt{n_x}\theta + H n_x L_w^2 \theta^2 \leq -\epsilon_s \tag{5.30}$$

where $\alpha(\cdot)$ *is a class* $\mathcal{K}_\infty$ *function associated with Assumption 5.4 and as defined in (5.14), and* H *is the constant associated with the Taylor expansion of* $\tilde{V}_N^0(x)$, *then the closed-loop system state* x *converges to the economic zone* $\mathbb{X}_e$ *in finite steps and then maintains in* $\mathbb{X}_e$

all the time for any initial condition $x(0) \in \Omega_{\rho_{\max}}$.

Proof. In this proof, we consider applying EMPC (5.10) which is designed based on the nominal system to the actual system with disturbance w. At time instant n, the EMPC optimization problem is solved with the actual system state $x(n)$ as the initial condition and only the first input value in the optimal input trajectory is applied to the system. Applying Proposition 5.1, from n to $n+1$, the deviation of the actual system state $x(n+1)$ from the nominal system state $\tilde{x}(n+1)$ is bounded as following:

$$|x(n+1) - \tilde{x}(n+1)| \leq \sqrt{n_x} L_w \theta \tag{5.31}$$

Using Taylor expansion, we can obtain the following relation:

$$\tilde{V}_N^0(x(n+1)) = \tilde{V}_N^0(\tilde{x}(n+1)) + \left.\frac{\partial \tilde{V}_N^0(x)}{\partial x}\right|_{\tilde{x}(n+1)} (x(n+1) - \tilde{x}(n+1)) + H.O.T \tag{5.32}$$

where $H.O.T$ includes the high order terms in the above Taylor expansion. For $x \in \mathbb{X}$, a positive constant H can be found such that the high order terms satisfy the following constraint:

$$H.O.T \leq H|x(n+1) - \tilde{x}(n+1)|^2 \tag{5.33}$$

Taking into account that the initial condition ($\tilde{x}(n) = x(n)$) when solving the EMPC optimization, (5.27), (5.31)–(5.33), it can be derived the following inequality:

$$\tilde{V}_N^0(x(n+1)) - \tilde{V}_N^0(x(n)) \leq -\alpha(|x(n) - x_s|) + K_V L_w \sqrt{n_x}\theta + H n_x L_w^2 \theta^2 \tag{5.34}$$

If condition (5.30) is satisfied, from (5.34), it can be seen that

$$\tilde{V}_N^0(x(n+1)) - \tilde{V}_N^0(x(n)) \leq -\epsilon_s \tag{5.35}$$

for all $x(n) \in \Omega_{\rho_{\max}}$ and $|x(n) - x_s| \geq \rho_s$. This implies that as long as $|x - x_s| \geq \rho_s$, the

Lyapunov function keeps decreasing. By applying (5.35) recursively, it is proved that the system state enters a region such that $|x - x_s| \geq \rho_s$ in finite steps. Given the definition of $\Omega_{\rho_{\min}}$, it further implies that once the state satisfies $|x - x_s| \geq \rho_s$, the state will remain in $\Omega_{\rho_{\min}}$ all the time. Then, the actual system under the control of the proposed EMPC will eventually converge to Ω_ρ. Given that $\Omega_{\rho_{\min}} \subset \mathbb{X}_e$, this proves that the system state enters the economic zone in finite steps and then remains within the economic zone. This proves Theorem 5.2. □

Remark 5.4. *The use of general economic objective in economic MPC may drive the system states to operate close to the operating constraints. This is no different in the zone economic MPC formulation. It is therefore possible that the optimal steady state within the desired economic zone is on the boundary. Since Theorem 5.2 require that the optimal operating point be in the interior of the desired economic zone, this needs to be resolved. One way to achieve this is to construct a smaller economic zone and then use that in the controller design. This way, the desired economic zone will be tracked once the smaller economic zone is tracked. Another approach is to construct an economic zone with a bigger risk factor and then track this new economic zone while ensuring that system's states go to the optimal operating point of the desired economic zone. To achieve this however, the cost function may need to be regularized to ensure that the steady-state point is tracked by the controller once the states are within the economic zone.*

5.5 Illustrative example

In this section, we demonstrate the efficacy of our proposed controller using a chemical process. We first describe the chemical process example used in our analysis. Subsequently, we consider the impact of the risk factor on the asymptotic average economic performance of our proposed controller and then finally compare the performance of our proposed controller to that of the conventional economic MPC.

5.5.1 Process description

Consider a well-mixed continuously stirred tank reactor (CSTR) where a single first-order irreversible reaction of the form $A \rightarrow B$ takes place. Since the reaction is exothermic, thermal energy is removed from the reactor through a cooling jacket. Assuming constant volume reaction mixture, the following nonlinear differential equations are obtained based on energy balance and component balance for reactant A:

$$\frac{dC_A}{dt} = \frac{q}{V}(C_{Af} - C_A) - k_0 \exp(-\frac{E}{RT})C_A \tag{5.36a}$$

$$\frac{dT}{dt} = \frac{q}{V}(T_f - T) + \frac{-\Delta H}{\rho C_p} k_0 \exp(-\frac{E}{RT})C_A + \frac{UA}{V\rho C_p}(T_c - T) \tag{5.36b}$$

where C_A and T denote the reactant concentration and temperature of the reaction mixture in mol/L and K respectively, T_c denotes the temperature of the coolant stream in K, q denotes the volumetric flow rate of the inlet and outlet streams of the reactor in L/min, C_{Af} denotes the concentration of reactant A in the feed stream, V denotes the volume of the reaction mixture, k_0 denotes the reaction rate pre-exponential factor, E denotes the activation energy, R is the universal gas constant, ρ is the density of the reaction mixture, T_f is the temperature of the feed stream, C_p is the specific heat capacity of the reaction mixture, ΔH is the heat of reaction and UA is the heat transfer coefficient between the cooling jacket and the reactor. The values of the parameters used in the simulations are listed in Table 5.1. A linear version of this model was used in [69] in the context of robust tube-based economic MPC.

The nonlinear model of (5.36) is discretized using a step-size $h = 0.1\ min$ to obtain a discrete-time nonlinear state space model of the following form:

$$x(n+1) = f(x(n), u(n), w(n)) \tag{5.37}$$

where $x = [C_A\ T]^T$ is the state vector, $u = T_c$ is the input and $w = [C_{Af}\ T_f]^T$ is the

Table 5.1: Table of parameter values

Parameter	Unit	Value
q	L/min	100.0
V	L	100.0
c_{Af}	mol/L	1.0
T_f	K	350.0
E/R	K	8750.0
k_0	min^{-1}	7.2×10^{10}
$-\Delta H$	J/mol	5.0×10^4
UA	$J/min \cdot K$	5.0×10^4
c_p	$J/g \cdot K$	0.239
ρ	g/L	1000.0

disturbance vector. The state, input and disturbance are assumed to be subject to the following hard constraints: $0.0 \leq x_1 \leq 1.0$, $345.0 \leq x_2 \leq 355.0$, $285.0 \leq u \leq 315.0$, $0.9 \leq w_1 \leq 1.1$ and $348.0 \leq w_2 \leq 352.0$. The disturbances are assumed to be uniformly distributed in the constraints with their nominal values being 1.0 and 350.0 as shown in Table 5.1.

The economic objective ℓ_e is to minimize the concentration of reactant A (i.e. maximize the concentration of reactant B) in the reactor such that

$$\ell_e(x, u) = c_A \tag{5.38}$$

To ensure that the economic cost is optimized within a reasonable temperature range, a zone tracking objective ℓ_z is incorporated into the control objective where

$$\ell_z(x) = \begin{cases} 10.0 \times (348.0 - T)^2 & \text{for } T < 348.0 \\ 0 & \text{for } 348.0 \leq T \leq 352.0 \\ 10.0 \times (352.0 - T)^2 & \text{for } T > 352.0 \end{cases} \tag{5.39}$$

The zone tracking objective is a quadratic function that penalizes the deviation of the system

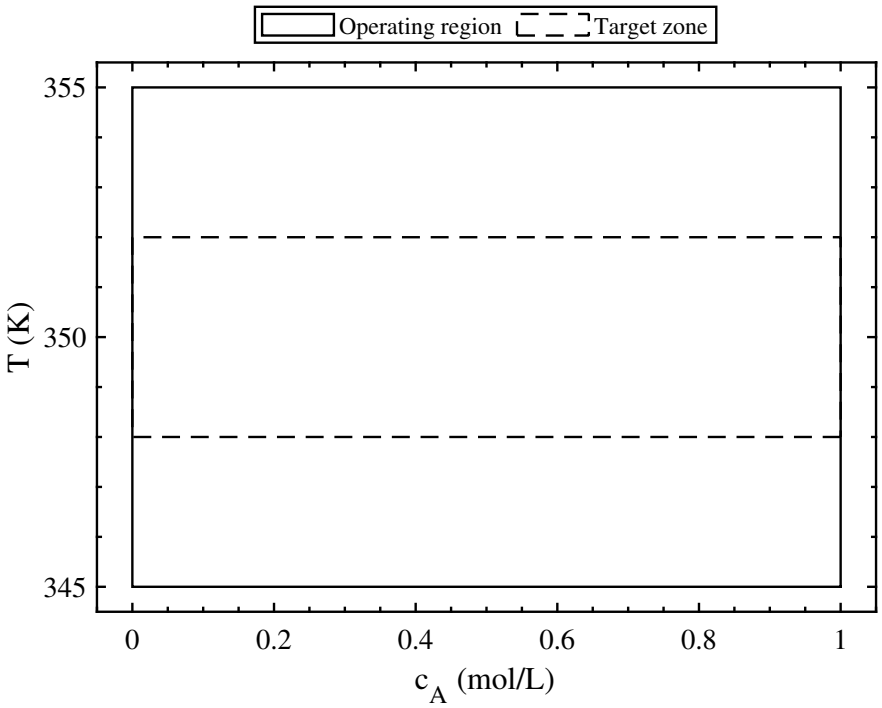


Figure 5.1: The sets used in the controller design. The operating region (solid line) is the hard constraint on the states where the process must be operated within without any violation. The target zone (dashed line) is a soft constraint on the states which ensures that the economic cost is optimized within reasonable temperature bounds.

states from the target zone $\mathbb{X}_t$. The overall control objective therefore becomes

$$\ell(x,u) := \ell_e(x,u) + \ell_z(x) = c_A + \begin{cases} 10.0 \times (348.0 - T)^2 & \text{for } T < 348.0 \\ 0 & \text{for } 348.0 \leq T \leq 352.0 \\ 10.0 \times (352.0 - T)^2 & \text{for } T > 352.0 \end{cases} \tag{5.40}$$

This control objective is multi-objective and can be achieved by manipulating the temperature of the coolant T_c in the cooling jacket. Notice that in this example, only the temperature has a zone requirement. The zone on the concentration therefore spans the entire constraint. The safe operating region (hard constraints) as well as the target zone $\mathbb{X}_t$ for this example is presented in Figure 5.1.

In the simulations, unless otherwise stated, the control and prediction horizons of all con-

trollers are $N = 20$ respectively. The l_1 norm weight c_1 and l_2 norm weight c_2 in (5.10) for our proposed controllers were chosen as 0 and 10 respectively. The selected weight ensures that the zone tracking cost is given a higher priority than the economic cost whenever the states of the system are outside the target zone. We assume that all the system states are available to the controller. The proposed robust economic MPC scheme and the traditional economic MPC scheme were numerically transcribed using the direct multiple shooting method and solved using IPOPT [76]. The optimization problems were implemented in the modeling language JuMP [77]. Each dynamic simulation was run for 1000 time steps and the asymptotic average performance, computed thereafter.

In the subsequent analysis, the optimal steady-state economic cost ℓ_e^* was obtained by solving the steady-state optimization problem in (5.5) to obtain the optimal steady state x_s together with the corresponding steady-state input u_s. The optimal steady-state cost ℓ_e^* was then obtained by computing the value of the economic cost in (5.38) using the steady-state values. The asymptotic average performance ℓ_{avg} on the other hand was obtained by simulating the closed-loop disturbed system for T time steps and finding the average of the overall control objective in (5.40) using the equation

$$\ell_{avg} = \frac{1}{T}\sum_{n=0}^{T-1} \ell(x(n), u(n)) \tag{5.41}$$

where $T = 1000$ time steps. ℓ_{avg} considers the effects of the target zone violations and is used to assess the performance of the controllers in the analysis.

5.5.2 Effect of risk factor δ

We first investigate the effect of the design parameter δ on the optimal steady-state economic cost ℓ_e^* and the asymptotic average performance ℓ_{avg} of the closed-loop system with the proposed economic zone MPC algorithm. This was achieved by varying the risk factor δ and determining ℓ_e^* in the associated economic zone as well as ℓ_{avg}. As can be seen from the

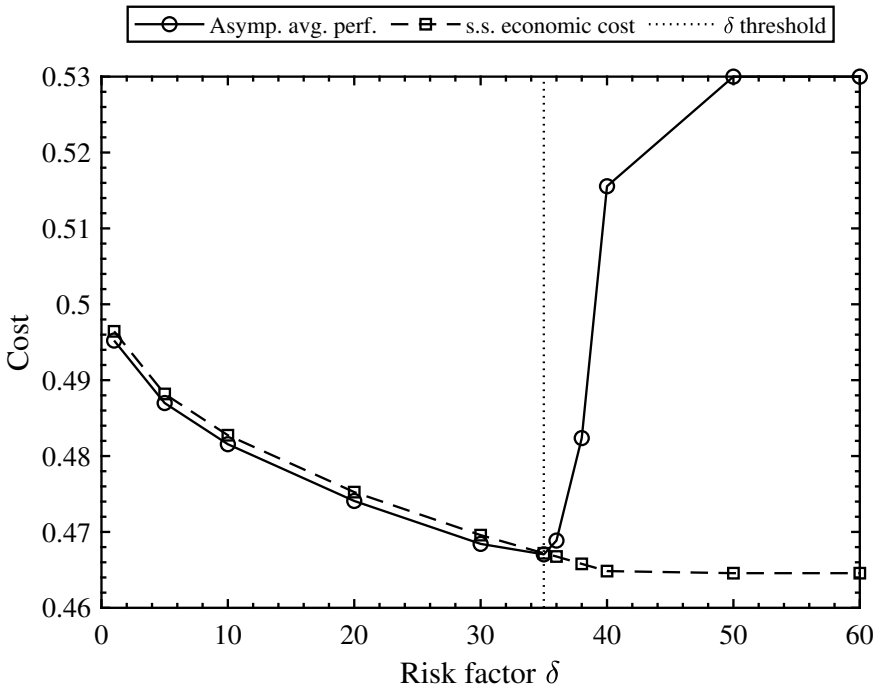


Figure 5.2: Effect of risk factor on the best steady-state cost in the economic zone and the closed-loop asymptotic average performance. The dotted lines show the threshold value of the risk factor after which the closed-loop asymptotic average performance begins to deteriorate implying a violation of the target zone. (Solid line with circle markers: Asymptotic average performance, Dashed line with square markers: Optimal steady-state cost, Dotted line: Risk factor threshold)

results in Figure 5.2, the value of both performance measures generally decrease as the risk factor increases until at $\delta = 35$ where ℓ_{avg} begins to increase.

As mentioned earlier and as shown in Figure 5.3, the size of the economic zone increases as the risk factor increases. This implies that a controller designed with a larger risk factor has a larger operating room to optimize the process economics compared to a controller designed with a smaller risk factor. To explain the reason for the difference in the plots of optimal steady-state economic cost ℓ_e^* and the asymptotic average performance ℓ_{avg}, we look at how the values were obtained. The optimal steady-state economic cost was obtained by solving the static optimization problem of (5.5). Since the effects of the disturbances are not explicitly considered in the steady-state optimization problem, ℓ_e^* represents the potentially achievable economic cost. ℓ_{avg} on the other hand represents the actual cost achieved in the

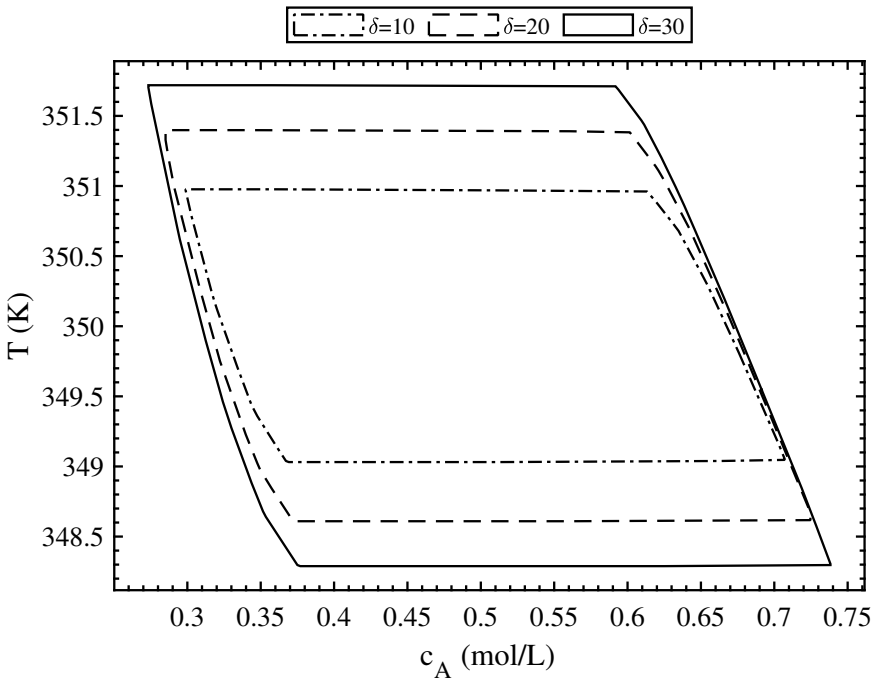


Figure 5.3: Effect of risk factor on the economic zone. As the δ increased, the size of the economic zone also increased and vice versa. The magnitude of the risk factor δ therefore determines the size of the economic zone and ultimately the conservativeness of the controller. (Solid line: $\delta = 30$, Dashed line: $\delta = 20$, Dash-dotted line: $\delta = 10$)

closed-loop system affected by the disturbance. A large economic zone therefore allowed the process to operate close to the target zone. The presence of the disturbance caused the process to violate the target zone and this resulted in a poor economic performance (on average). A violation of the target zone means that the conditions in Theorem 5.2 were not satisfied.

The analysis in Figure 5.2 implies that the risk factor should not be arbitrarily chosen. It should be chosen such that the conditions in Theorem 5.2 are satisfied to ensure that the states of the system converge to the target zone $\mathbb{X}_t$ in finite time and stays in it thereafter even in the presence of disturbances. For this illustrative example, any δ value above the threshold value of 35 resulted in a poor closed-loop asymptotic average performance ℓ_{avg}. Intuitively, the risk factor is a design parameter which offers a trade-off between a conservative controller or a more risk-taking one to maximize the economic objective. This implies that the risk factor δ in the proposed controller needs to be carefully tuned to get a good trade-off.

5.5.3 Comparison with an EMPC tracking the target zone

Following the analysis of the effects of the risk factor on the controller performance, we compare the closed-loop performance of our proposed controller (ZEMPC tracking the economic zone) to that of conventional EMPC (ZEMPC tracking the target zone). The economic zone $\mathbb{X}_e$ for our proposed controller was determined using a risk factor of 30. However, as mentioned in Remark 5.4, the optimal steady state for this process lies on the boundary of $\mathbb{X}_e$. To ensure that the conditions in the Theorems are satisfied, a smaller economic zone with $\delta = 10$ was computed and the optimal steady state within the smaller economic zone determined. Figure 5.4 shows the sets used in the proposed control algorithm with the optimal steady state within its interior. The conventional EMPC on the other hand was designed to track the original target zone without any modification. In both cases, c_1 and c_2 were selected to be 0 and 10 respectively. The steady state values with and without the computed economic zone are $(x_s, u_s) = ([0.483 \;\; 350.970]^T, 299.709)$ and $(x_s, u_s) = ([0.465 \;\; 352.000]^T, 299.413)$

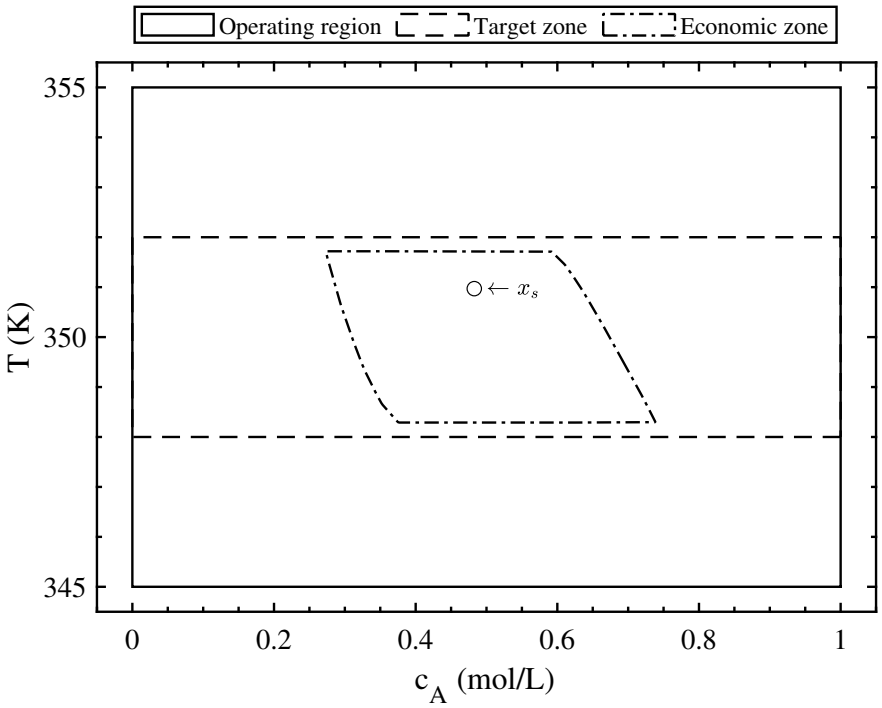


Figure 5.4: The sets used in the controller design. (Solid line: Hard constraint, Dashed line: Original zone, Dash-dotted line: Economic zone)

Table 5.2: Asymptotic average performance for the controllers

Controller	ℓ_{avg}
Conventional EMPC	0.530
Proposed EMPC	0.482

respectively.

The results of the comparison is shown in Table 5.2. As can be seen in the table, our proposed controller gave (on average) a lower asymptotic average performance compared to the conventional EMPC in the presence of the disturbance. To understand why this is so, Figure 5.5 has been provided. Figure 5.5 shows the state, input and average performance trajectories of the closed-loop system under the two controllers in the presence of disturbance. It can be observed that our proposed EMPC forces the system to operate at a temperature below the $352.0K$ thus allowing room for the disturbances to occur without any significant effects on the cost. This results in a fairly stable overall cost ℓ. The conventional EMPC

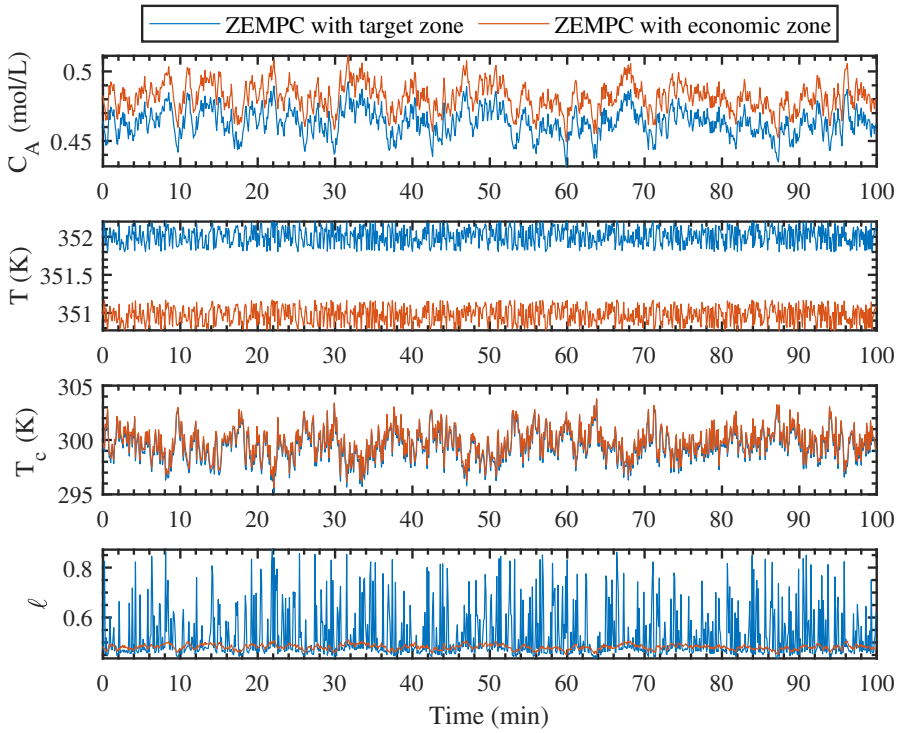


Figure 5.5: State, input and economic cost trajectories of the CSTR process under conventional zone EMPC (black) and our proposed zone EMPC (red)

on the other hand operated close to $352.0K$. Thus, the effects of the disturbances caused the system to operate in an expensive zone which results in a much higher cost. It is worth mentioning that our proposed ZEMPC with economic zone sacrifices some economic cost ℓ_e as can be observed from the trajectories of c_A in Figure 5.5. This ensured that the control objective ℓ is achieved always even in the presence of the disturbances.

5.6 Concluding remarks

In this chapter, we presented a robust EMPC framework with zone tracking for general nonlinear systems. The proposed design ensures that the zone tracking objective can be achieved in finite steps and the economic performance in the operation is optimized. A robust control invariant set within the original target zone is determined and is used as the actual zone tracked. To optimize the economic performance within the zone in the presence of disturbances, the notion of risk factor in the controller design was adopted. An algorithm to determine the economic zone to be tracked was provided. The risk factor determines the conservativeness of the controller and provides a way to tune the EMPC for better economic performance. A nonlinear chemical example was presented to demonstrate the performance of the proposed formulation.

Chapter 6

Robust economic MPC of the absorption column in post-combustion carbon capture through zone tracking

6.1 Introduction

Climate change is a pressing global issue that needs immediate solution. A major contributing factor to climate change is the presence of large quantities of anthropogenic greenhouse gases especially carbon dioxide (CO_2) in the atmosphere. A major contributor to the increase in anthropogenic CO_2 in the atmosphere is due to the combustion of fossil fuels such as coal for electricity generation [78]. Several studies have shown that switching to low carbon energy sources such as renewable energy sources can help address the climate change issues [79]. However, the pursuit for relatively cheaper and reliable energy sources in response to the ever increasing demand for energy is making it difficult to switch to these lower carbon energy sources. In 2020, fossil fuels contributed to about 61% of the global electricity generation sources with coal taking up roughly 35% of the share [80]. Clearly, it is impractical to completely eliminate fossil fuels from the energy generation sources at a go. Therefore,

effective means of reducing the emissions from fossil fuel power plants is the best approach to reduce CO_2 emissions while allowing the low carbon electricity generation technologies to reach maturity. Several approaches have been proposed to reduce CO_2 emissions from large point sources such as power plants. These approaches include pre-combustion [81], oxy-fuel combustion [82] and post-combustion [83]. However, amine-based post-combustion CO_2 capture (PCC) is the most mature and viable technology available today.

In amine-based PCC, the flue gas produced after combusting the fossil fuel is sent to a gas processing unit (PCC plant) where the amount of CO_2 in the gas is reduced using an amine solvent before being released into the atmosphere. This makes it easier to retrofit it into existing power plants. However, amine-based PCC is not without downsides. It has been shown that attaching a PCC plant to a power plant reduces the efficiency of the power plant by about 10% for state-of-the-art monoethanolamine (MEA) solvent [84]. This is because of the high energy requirements to regenerate the amine in the desorption unit. Sakwattanapong and coworkers [85] demonstrated that maintaining the CO_2 concentration in the amine solvent within an optimal range during absorption is essential for efficient operation of the regeneration unit. It is therefore critical that advanced model-based process control techniques such as model predictive control (MPC) are employed in the control of the PCC plant.

Model predictive control is an advanced model-based optimal control method that has gained popularity within the chemical process industry. This is because of its ability to handle complex multivariable systems and constraints. Within the context of process control of PCC plants, several control schemes have been developed using MPC. Panahi and Skogestad [86] investigated different control schemes using a linear MPC. He et al. [87] also used a combined scheduling and MPC scheme to achieve the desired carbon dioxide absorption efficiency in the absorption column as well as the CO_2 purity in the gas outlet of the desorption unit. Bankole and coworkers [88] investigated the flexibility of operating a PCC plant attached to a load following power plant using MPC. To address the presence

of uncertainties in the control system, Patron and Ricardez-Sandoval implemented a robust MPC algorithm for the absorption column [89] as well as an integrated control and state estimation scheme for the PCC plant [90]. More recently, a variant of model predictive control (MPC) with a general objective known as economic MPC (EMPC) has received significant attention [55, 56]. The objective function in an EMPC scheme generally reflects some economic performance criterion such as profit maximization or waste minimization. This is in contrast to the standard MPC where the objective is a positive definite quadratic function. The integration of process economics directly in the control layer makes EMPC of interest in many areas especially in the process industry. Decardi-Nelson, Liu and Liu [59] demonstrated the superiority of EMPC scheme over the standard MPC scheme in a full cycle PCC plant.

While EMPC is a promising control algorithm for the PCC process, the general economic objective such as maximizing the absorption efficiency of the absorption column may drive the system states to the constraints. This may lead to column flooding and/or solvent overcirculation in the PCC plant. Column flooding can compromise the safety of both the absorption column and the personnel that manage it while solvent overcirculation may lead to high energy requirements for regeneration of the solvent in the desorption column. In another direction, it is not well understood how the presence of uncertainties affect the economic performance of EMPC in general. This is because of the integration of process economics in the control layer. Uncertainties are unavoidable in the real world. They are caused by the use of imperfect process models in the model-based control algorithms and/or unmeasured disturbances. A typical disturbance in a load following PCC plant attached to a power plant is large fluctuations in the flue gas flow rate [91]. The presence of uncertainty in a control system can result in significant performance degradation and/or loss of stability. This can ultimately compromise safety during operation. A common technique to address uncertainties in a control system is through robust MPC. However, as observed in a recent study by Patron and Ricardez-Sandoval [89], the online computational requirements of robust

MPC techniques such as the multi-scenario approach can be demanding as the number of uncertainties and scenarios increase. Moreover, for EMPC, it was pointed out in the study by Bayer and coworkers [68] that simply transferring robust MPC techniques to EMPC could result in poor performance. This is because economic optimization and robustness are two objectives and often may conflict with each other. Robust MPC techniques have been designed to reject all disturbances to achieve their desired goal which may not be the case in EMPC as some disturbances can lead to better economic performance. It is therefore important to develop robust EMPC algorithms which does not involve complex online computations.

In this chapter, we present an economic MPC with zone tracking algorithm for the control of the absorption column of the PCC process under additive state uncertainties and time-varying flue gas flow rate. The proposed control algorithm only makes use of the nominal process model without explicitly accounting for the uncertainties in the process. This makes the online computations less demanding compared to the scenario-based approach. The integration of zone tracking in EMPC allows for concurrent handling of two objectives while enhancing the degree of robustness of the controller [13]. Zone MPC has been reported in several process control application areas such as diabetes management [71], control of building heating system [72], control of irrigation systems [73] and control of coal-fired boiler-turbine generating system [60]. This work builds on our previous EMPC with zone tracking formulation – with [92] and without [12] uncertainty consideration. In line with the work by Decardi-Nelson and Liu [92] described in Chapter 5, we propose to track a control invariant subset of the target zone contrary to tracking the target zone. However, because of the large number of states in the process model of the absorption column, the zone modification algorithm developed by Decardi-Nelson and Liu [92] is computationally intractable. We therefore propose a target zone modification algorithm using ellipsoidal control invariant set computation techniques and a back-off strategy. This has a potential to extend the applicability of the robust EMPC with zone tracking algorithm to much wider range of

systems.

6.2 Preliminaries

6.2.1 Notation

Throughout this chapter, the symbol $\|\cdot\|_n$ denotes the n-norm of a scalar or a vector, the operator '\' means set subtraction such that $\mathbb{A}\backslash\mathbb{C} = \{x : x \in \mathbb{A}, x \notin \mathbb{C}\}$, $\mathbb{R}_+$ denotes the set of all real numbers greater than or equal to zero, the set $\mathbb{B}$ represents the unit ball.

6.2.2 Process description

An absorption column in an amine-based post-combustion CO_2 capture process is a multi-stage gas processing unit in which an amine solvent selectively removes CO_2 from the flue gas. The amine solvent with low amount of CO_2 (lean solvent) is introduced at the top of the column while the flue gas enters the column from the bottom in a counter-current manner as shown in Figure 6.1. The absorption column is usually filled with packing materials to increase the contact area for mass transfer between the liquid and the gas phases. Following the transfer of CO_2 from the gas phase to the liquid phase, the liquid with increased amount of CO_2 (rich solvent) exits the column at the bottom while the treated gas exits the column at the top.

Owing to the reactive nature of the mass transfer process occurring in the absorption column, the rate-based approach is used to model the process. The rate-based model has been found to be superior to the equilibrium-based approach to modeling reactive mass transfer processes [93]. The following assumptions were used in modeling the absorption column:

- The liquid and the gas phases are well mixed with no spatial variations in properties.

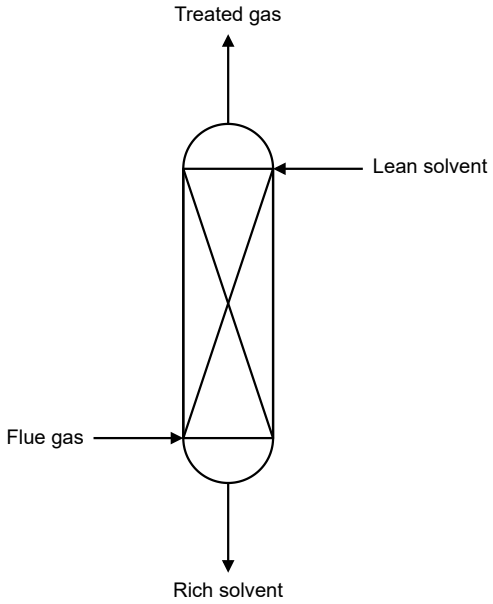


Figure 6.1: A schematic diagram of a packed absorption column

- The reactions are described using enhancement factor and occur only in the liquid phase.
- The heat and mass transfer occurring at the gas-liquid interface is described by the two film theory.
- The pressure drop along the axial direction of the column is linear.
- The velocities of the liquid and gas phases in the column remain constant.
- The absorption column is well insulated.

The simultaneous heat and mass transfer process occurring in the column is described by the partial differential equations in Equations (6.1) – (6.4).

$$\frac{\partial c_{Li}}{\partial t} = \frac{4F_L}{\pi D_c^2}\frac{\partial c_{Li}}{\partial z} + N_i a^I \tag{6.1}$$

$$\frac{\partial c_{Gi}}{\partial t} = -\frac{4F_G}{\pi D_c^2}\frac{\partial c_{Gi}}{\partial z} - N_i a^I \tag{6.2}$$

$$\frac{\partial T_L}{\partial t} = \frac{4F_L}{\pi D_c^2}\frac{\partial T_L}{\partial z} + \frac{Q_L a^I}{\sum_{i=1}^{n} c_{Li} c_{pi}} \tag{6.3}$$

$$\frac{\partial T_G}{\partial t} = -\frac{4F_G}{\pi D_c^2}\frac{\partial T_G}{\partial z} + \frac{Q_G a^I}{\sum_{i=1}^{n} c_{Gi} c_{pi}} \tag{6.4}$$

In Equations (6.1) – (6.4), c_i represents the phase concentration of component i in $kmol/m^3$, F is the phase volumetric flow rate in m^3/s, D_c is the diameter of the column in m, N_i is the mass transfer rate of component i in $kmol/m^2s$, z is the height of the column in m, T denotes the phase temperature in K, Q denotes the heat transfer rate in kJ/m^2s, c_p is the heat capacity in $kJ/kmol$, a^I is the interfacial area in m^2/m^3. Also, subscripts L and G represent the liquid and gas phase respectively, and subscript i denotes the components in the system namely CO_2, N_2, H_2O and MEA.

In the mathematical model above, the enhancement factor approach together with the Chilton-Colburn analogy [94] is used to determine the influence of the reactions on the rate of heat and mass transfer of CO_2 from the gas phase to the liquid phase. Details of the physical and chemical properties of the components in the system can be found in the work by Decardi-Nelson et al. [59].

6.2.3 Model discretization and state space representation

To avoid having to formulate and solve infinite dimensional optimal control problems online, the partial differential equations are converted to ordinary differential equations using the method of lines (MOL). The method of lines involves discretizing the partial derivatives with respect to the length of the column to obtain only differential equations with respect to time. In this work, the derivatives with respect to the length of the column were discretized into five stages to obtain 50 ordinary differential equations as shown in Equations (6.5) – (6.8).

$$\frac{dc_{Li}^j}{dt} = \frac{4F_L}{\pi D_c^2}\frac{c_{Li}^j - c_{Li}^{j-1}}{z^j - z^{j-1}} + N_i^j a^{I,j} \tag{6.5}$$

Table 6.1: Definition of the state variables at the jth discrete point ($j = 1, 2, \cdots, 5$). N_2 = 1, CO_2 = 2, MEA = 3, H_2O = 4

State variable	Definition
x_{1-5}	$c^j_{L,1}$
x_{6-10}	$c^j_{L,2}$
x_{11-15}	$c^j_{L,3}$
x_{16-20}	$c^j_{L,4}$
x_{21-25}	T^j_L
x_{26-30}	$c^j_{G,1}$
x_{31-35}	$c^j_{G,2}$
x_{36-40}	$c^j_{G,3}$
x_{41-45}	$c^j_{G,4}$
x_{46-50}	T^j_G

$$\frac{dc^j_{Gi}}{dt} = -\frac{4F_G}{\pi D_c^2}\frac{c^j_{Gi} - c^{j-1}_{Gi}}{z^j - z^{j-1}} - N^j_i a^{I,j} \tag{6.6}$$

$$\frac{dT^j_L}{dt} = \frac{4F_L}{\pi D_c^2}\frac{T^j_L - T^{j-1}_L}{z^j - z^{j-1}} + \frac{Q^j_L a^{I,j}}{\sum_{i=1}^n c^j_{Li} c^j_{pi}} \tag{6.7}$$

$$\frac{dT^j_G}{dt} = -\frac{4F_G}{\pi D_c^2}\frac{T^j_G - T^{j-1}_G}{z^j - z^{j-1}} + \frac{Q^j_G a^{I,j}}{\sum_{i=1}^n c^j_{Gi} c^j_{pi}} \tag{6.8}$$

where $j = 1, 2, \cdots, 5$ denotes the index of the respective variable at height z^j of the column. The variables at indices $j = 0$ and $j = 5$ represent the inlet and outlet boundary conditions of respectively.

Considering the presence of uncertainties in the system, the dynamics of the CO_2 absorption column can be written in a nonlinear state space model of the form:

$$\dot{x}(t) = f(x(t), u(t)) + w(t) \tag{6.9}$$

where $\dot{x} \in \mathbb{R}^{50}$ is the time derivative of the state, $x \in \mathbb{R}^{50}$ is the state of the system at time $t \in \mathbb{R}_+$, $u = F_L \in \mathbb{R}$ is the manipulated input, and $w \in \mathbb{R}^{50}$ represents the additive uncertainties that may be present in the system. The definition of the state variable x is shown in Table 6.1. The controlled output y of the system is the CO_2 absorption efficiency

and is given by

$$y(t) = h(x(t)) = \frac{\text{Molar flow rate of } CO_2 \text{ in} - \text{Molar flow rate of } CO_2 \text{ out}}{\text{Molar flow rate of } CO_2 \text{ in}} \times 100\% \quad (6.10)$$

For practical reasons, we assume that the system state, input, output and uncertainty are restricted to be in the compact sets $\mathbb{X}$, $\mathbb{U}$, $\mathbb{Y}$ and $\mathbb{W}$ respectively.

6.2.4 Control problem formulation

In a post combustion carbon dioxide capture plant, the primary objective of the absorption column is to reduce the amount of carbon dioxide in the flue gas emanating from the power plant. This can be achieved by controlling the efficiency y of the absorption column. The absorption efficiency can be controlled by manipulating the lean solvent flow rate F_L and the concentration of CO_2 in the lean solvent entering the top of the column. In this work, the concentration of CO_2 in the lean solvent is kept constant since the desorption column is not considered. Therefore, only the lean solvent flow rate is used as the manipulated variable. Under these conditions, a very high CO_2 absorption efficiency which translates to high removal of CO_2 from the inlet flue gas may be achieved by using a high lean solvent flow rate. However, using a high amount of solvent to reduce the amount of CO_2 in the flue gas can have negative effects on the operation of the absorption column and the PCC plant as a whole. First, a high amount of solvent may cause flooding in the column. Column flooding is usually followed by a dramatic increase in column pressure and prevent the flue gas from flowing out of the column. This may result in inefficient operation of the column and/or equipment damage. Second, a high solvent flow rate may often than not lead to overcirculation of the solvent in the PCC plant. This usually results in the solvent leaving the absorption column (rich solvent) having a low CO_2 concentration. This makes it difficult to operate the desorption column efficiently. Therefore the desire is usually to keep

the absorption efficiency y within a target zone $\mathbb{Y}_t$ which ensures a balance between high absorption efficiency, column flooding and overcirculation.

The primary control objective of this work is therefore to design a feedback controller which drives the system state to a predetermined target zone $\mathbb{Y}_t$ if the initial absorption efficiency of the system is outside the target zone and subsequently maintain the state of the system within the target zone $\mathbb{Y}_t$ thereafter. A secondary objective is to minimize the average economic cost ℓ_e over an infinite horizon T which is given by

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \ell_e(y(t)) \tag{6.11}$$

where

$$\ell_e(y) = -y \tag{6.12}$$

denotes the economic objective to be minimized. To achieve the above control objectives, we resort to EMPC with zone tracking [12, 92] and implicitly take into account the presence of process disturbance w in the design of the EMPC.

6.3 Economic model predictive control with zone tracking

In this section, we present the economic model predictive control with zone tracking (ZEMPC) algorithm. Specifically, two variations of the ZEMPC algorithm are presented. The first control algorithm denoted as nominal ZEMPC (NZEMPC) is the economic model predictive control with target zone tracking. In this formulation, the disturbances are not considered in the design. This serves as a basis to compare the second variation of ZEMPC. In the second control algorithm denoted as robust ZEMPC (RZEMPC), the original target zone $\mathbb{Y}_t$ is modified to implicitly consider the effects of the uncertainty in the process system.

We begin this section by presenting the NZEMPC formulation. Thereafter, an algorithm to modify the original target zone $\mathbb{Y}_t$ is presented. Finally, we present the RZEMPC formulation.

6.3.1 Economic MPC with target zone tracking

Given information about the current state $x(t_k)$ at sampling time t_k, the ZEMPC uses the nominal model of system (6.9):

$$\dot{\tilde{x}}(t) = f(\tilde{x}(t), v(t)) \tag{6.13}$$

with the initial condition $\tilde{x}(t_k) = x(t_k)$ to find a control sequence $\mathbf{v} = \{v(t_k), \ldots, v(t_k+N\Delta)\}$ and the associated state sequence $\tilde{\mathbf{x}} = \{\tilde{x}(t_k), \ldots, \tilde{x}(t_k + \Delta N)\}$ over the entire prediction horizon N at a sampling time Δ to minimize the cost functional:

$$V_N(x(t_k), \mathbf{v}) = \int_{t_k}^{t_k+N\Delta} \ell(y(t))dt \tag{6.14}$$

In Equations (6.13) and (6.14), $\tilde{x}(t) \in \mathbb{X} \subseteq \mathbb{R}^{50}$ and $v(t) \in \mathbb{U} \subseteq \mathbb{R}$ are the nominal state vector and computed control input vector respectively. The stage cost $\ell(\cdot)$ is defined as follows:

$$\ell(y) = \ell_e(y) + \ell_z(y) \tag{6.15}$$

where $\ell_e(\cdot)$ is the economic stage cost as introduced in (6.12) and $\ell_z(\cdot)$ is a zone tracking penalty term which is defined as below:

$$\ell_z(y) = \min_{y_z} \quad c_1(\|y - y_z\|_2^2) \tag{6.16a}$$

$$s.t. \quad y_z \in \mathbb{Y}_t \tag{6.16b}$$

with $c_1 \in \mathbb{R}_+$ being a non-negative weight on the zone tracking term and y_z is a slack variable. The zone tracking stage cost reflects the distance of the system states from the target zone and is positive definite. The weights c_1 must be appropriately selected such that the zone tracking cost is given a higher priority than the economic objective.

At each sampling time, the following dynamic optimization problem $\mathcal{P}_N(x(t_k))$ is solved:

$$\min_{\mathbf{v},\mathbf{y_z}} \quad \int_{t_k}^{t_k+N\Delta} -y(t) + c_1(\|y(t) - y_z(t)\|_2^2)dt \tag{6.17a}$$

$$s.t. \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), v(t)) \tag{6.17b}$$

$$y(t) = h(\tilde{x}(t)) \tag{6.17c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{6.17d}$$

$$\tilde{x}(t) \in \mathbb{X} \tag{6.17e}$$

$$v(t) \in \mathbb{U} \tag{6.17f}$$

$$y(t) \in \mathbb{Y} \tag{6.17g}$$

$$y_z(t) \in \mathbb{Y}_t \tag{6.17h}$$

In the Optimization problem (6.17) above, Equation (6.17b) is the model constraint, Equation (6.17c) represents the output relationship, Equation (6.17d) is the initial state constraint, Equation (6.17e) – (6.17g) are the constraints on the state, input and output respectively, and Equation (6.17h) is the zone constraint. As a result of the cost function employed, the Optimization problem (6.17) is a multi-objective optimization problem which seeks to minimize the deviation of the absorption efficiency from the target zone $\mathbb{Y}_t$ while at the same time maximizing the the efficiency within the target zone.

The solution of $\mathcal{P}_N(x(t_k))$ denoted $\mathbf{v}^*$ gives an optimal value of the cost $V_N^0(x(t_k))$ and at the same time $u(t_k) = v^*(t_k)$ is applied to the actual system (6.9).

6.3.2 Modification of the target zone

While the NZEMPC described in the earlier section ensures that the zone tracking objective is achieved for the absorption column without any uncertainty, the zone tracking objective may not be achieved in general for systems with uncertainty. This may be due to two reasons. First, the target zone may not necessarily be forward invariant for the closed-loop system. This means that it is possible for the uncertainty to drive the system's states to a region outside the forward invariant set but within the target zone. Once the state is outside the forward invariant set, the target zone cannot be tracked anymore and will ultimately result in system instability. Second, the NZEMPC algorithm may cause the system output to operate very close to the boundary of the target zone due to the secondary economic objective employed; that is maximization of the absorption efficiency. While this may not be an issue in the nominal case, the presence of the uncertainty may drive the system states outside the target zone making it difficult to track the zone.

Therefore to achieve the zone tracking objective in the presence of uncertainty, we modify the target zone used in the formulation of the EMPC with zone tracking algorithm. It is worth mentioning that modification of the target zone is not trivial. For example, merely tracking the center of the target zone may not be the best choice. This will be demonstrated in the simulation section. The idea is to find a robust control invariant set within the target zone which ensures that the target zone can still be tracked in the presence of the uncertainty. A robust control invariant set R is a set of initial states in the target zone for which there exists a control action such that the trajectory of the system stays in R for all future times irrespective of the disturbances. Ideally, the largest robust control invariant set within the target zone is desired [92]. However, finding the largest robust control invariant sets for large scale systems is very difficult and the method proposed by Decardi-Nelson and coworkers [92] cannot be applied to the absorption column due to the number of states involved. We therefore resort to using simpler ellipsoidal control invariant set techniques. Since the ellipsoidal control invariant set does not consider the presence of uncertainty in the control

system, we use a back-off approach not only to account for the presence of disturbances but to also avoid operating close to the boundary of the target zone $\mathbb{Y}_t$.

Before we present the zone modification algorithm, let us first define the steady-state (SS) optimization problem with respect to the target zone as

$$(\ell_e^*, x_s, u_s) = \arg\min \ \ell_e(y) \tag{6.18a}$$

$$s.t. \quad 0 = f(x, u) \tag{6.18b}$$

$$y = h(x) \tag{6.18c}$$

$$x \in \mathbb{X} \tag{6.18d}$$

$$u \in \mathbb{U} \tag{6.18e}$$

$$y \in \mathbb{Y}_t \tag{6.18f}$$

In Optimization problem (6.18), Equation (6.18b) is the system model defined in system (6.9) without any uncertainty, and Equation (6.18c) – (6.18f) are the same as defined previously. The Optimization problem (6.18) returns the optimal economic cost ℓ_e^* within the target zone $\mathbb{Y}_t$ as well as the steady-state state x_s and input u_s. The ℓ_e^* serves as a lower bound on the economic cost that can be achieved in the target zone $\mathbb{Y}_t$. In the proposed zone modification algorithm, this value is relaxed by multiplying it with the relaxation rate r to obtain the relaxed optimal economic cost ℓ_e^r. Here, relaxation of the optimal steady-state economic cost within the target zone means increasing the value of the economic cost above ℓ_e^* such that

$$\ell_e^* < \ell_e^r \tag{6.19}$$

Relaxing the best economic cost within the target zone sacrifices some economic performance to implicitly account for the effects of the uncertainty in the control system. To achieve the

relaxation, the following cost-relaxed steady-state (CRSS) optimization problem is solved

$$(x_s, u_s) = \arg\min \ 0 \tag{6.20a}$$

$$s.t. \quad 0 = f(x, u) \tag{6.20b}$$

$$x \in \mathbb{X} \tag{6.20c}$$

$$u \in \mathbb{U} \tag{6.20d}$$

$$y \in \mathbb{Y}_t \tag{6.20e}$$

$$\ell_e(y) = \ell_e^r \tag{6.20f}$$

In Optimization problem (6.20) above, Equation (6.20f) is the economic cost relaxation constraint which needs to be achieved. It can be seen that Optimization problem (6.20) is a feasibility problem since it does not seek to minimize any cost function. The CRSS problem returns the steady-state operating point (x_s,u_s) at the relaxed economic cost function value. This is used in the subsequent parts of the zone modification algorithm to obtain the ellipsoidal control invariant set.

To compute the ellipsoidal control invariant set, the nonlinear system is first linearized about the steady-state operating point (x_s,u_s) from Optimization problem (6.20) to obtain a linear system of differential equations of the form

$$\dot{\bar{x}}(t) = A\bar{x}(t) + B\bar{u}(t) \tag{6.21}$$

where $A = \frac{\partial f(x,u)}{\partial x}|_{x_s,u_s}$ and $B = \frac{\partial f(x,u)}{\partial u}|_{x_s,u_s}$ are matrices of appropriate dimensions, and $\bar{x}$ and $\bar{u}$ denote the system states and input in deviation form i.e. $\bar{x} = x - x_s$ and $\bar{u} = u - u_s$. Following the linearization, a semi-definite program (SDP) is formulated according to the work by Polyak and Shcherbakov [95]. The SDP to be solved is presented in Optimization problem (6.22).

$$\max_{P,Y} \quad \text{trace}(P) \tag{6.22a}$$

$$s.t. \quad AP + PA^T + BY + Y^T B^T \prec 0 \tag{6.22b}$$

$$\begin{bmatrix} P & Y^T \\ Y & \bar{u}_{\max}^2 I \end{bmatrix} \succeq 0 \tag{6.22c}$$

where P and Y are matrices of appropriate dimension and $\bar{u}_{\max}$ is the bound on the input i.e. $\|\bar{u}\| \leq \bar{u}_{\max}$. By maximizing the trace of P, the maximal ellipsoidal control invariant set under the constrained input can be obtained from the optimal solution of (6.22) as

$$\mathbb{X}_m = \{\bar{x} \in \mathbb{R}^{50} : \bar{x}^T P^{-1} \bar{x} \leq 1\}, \qquad P \succ 0 \tag{6.23}$$

It is worth mentioning that computing an ellipsoidal control invariant set for the case where system (6.21) is stable (i.e. real part of the eigen values of A are negative) is trivial. This is because the stability region spans the entire state space and the input is not necessary for stabilization. Thus, Optimization problem (6.22) will not return a solution since the maximal ellipsoidal control invariant set is unbounded. In such a situation, the Lyapunov equation $AP + PA^T + Q = 0$ is solved with Q being an identity matrix of appropriate dimension. In this case the ellipsoidal control invariant set is obtained as $\mathbb{X}_m = \{\bar{x} \in \mathbb{R}^{50} : \bar{x}^T P \bar{x} \leq \alpha\}$ where $\alpha \geq 0$ is a scalar parameter which determines the size of the invariant set.

Remark 6.1. *A linear system obtained by linearizing a nonlinear system might only be accurate within a very small region of the linearization point (origin). To avoid obtaining an ellipsoidal control invariant set that spans areas in the state space for which the linear model is inaccurate, the bound on the input u_{max} should be reduced. The magnitude of the reduction ultimately depends on the properties of the system.*

Once $\mathbb{X}_m$ is obtained, it is projected into the output space using the output equation to

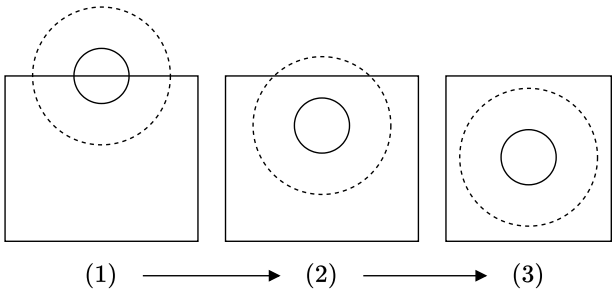


Figure 6.2: An illustration of three iterations of the zone modification algorithm in a fictitious two dimensional space. The rectangle represents the original target zone, the circle with solid line represent the ellipsoidal invariant set and the circle with dashed lines represent the $\varepsilon\mathbb{B}$ enlargement of the ellipsoidal invariant set. The algorithm terminates in the third step when the $\varepsilon\mathbb{B}$-enlarged ellipsoidal invariant set does not intersect with the set outside the target zone.

obtain the modified target zone $\mathbb{Y}_m$ such that

$$\mathbb{Y}_m = h(\mathbb{X}_m) \tag{6.24}$$

Since the output equation in Equation (6.10) depends on only the concentration of CO_2 in the gas exiting the top of the absorption column (inlet CO_2 concentration in the gas phase is fixed), the minimum and the maximum state in $\mathbb{X}_m$ can be used to obtain $\mathbb{Y}_m$. For more general cases, a finite sample of states in $\mathbb{X}_m$ may be required. The modified output zone $\mathbb{Y}_m$ is then enlarged by an ε-ball. This is used as a stopping criterion by checking if the enlarged modified output target zone $(\mathbb{Y}_m + \varepsilon\mathbb{B})$ does not intersect with any part of the output space outside the output target zone, that is

$$\mathbb{Y}_m + \varepsilon\mathbb{B} \cap \mathbb{Y}_t \backslash \mathbb{Y} = \emptyset \tag{6.25}$$

The procedure is run in while loop until the stopping criterion in Equation (6.25) is met. The entire zone modification algorithm is summarized in Algorithm 1. A visual depiction of the algorithm is shown in Figure 6.2. The algorithm takes as inputs the nominal system model f, the output equation h, the state $\mathbb{X}$ and input $\mathbb{U}$ constraints, the optimal economic

Algorithm 8: Modification of target zone

Input: f, h, ℓ_e, $\mathbb{X}$, $\mathbb{U}$, $\mathbb{Y}_t$, $u_{\max}$, ε, r, $N_{\max}$, ℓ_e^*

Output: $\mathbb{X}_m$, $\mathbb{Y}_m$

$\mathbb{Y}_0 \leftarrow \mathbb{Y}_t$
$\mathbb{X}_0 \leftarrow \emptyset$
$\ell_e^r \leftarrow \ell_e^*$
$i \leftarrow 1$
while $\mathbb{Y}_{i-1} + \varepsilon\mathbb{B} \cap \mathbb{Y}_t \backslash \mathbb{Y}$ **do**
    $\ell_e^r \leftarrow (1+r)\ell_e^r$
    Solve the optimization problem in (6.20) to obtain (x_s, u_s)
    Linearize the system (6.9) at (x_s, u_s) to obtain A and B
    if *A is stable* **then**
        Solve the Lyapunov equation to obtain P
    else
        Solve (6.22) to obtain P
    Find the ellipsoidal control invariant set $\mathbb{X}_i$ using P
    $\mathbb{Y}_i \leftarrow h(\mathbb{X}_i)$
    if $i = N_{max}$ **then**
        $\mathbb{X}_i \leftarrow \emptyset$
        $\mathbb{Y}_i \leftarrow \emptyset$
        break
    $i \leftarrow i + 1$
$\mathbb{X}_m \leftarrow \mathbb{X}_i$
$\mathbb{Y}_m \leftarrow \mathbb{Y}_i$
return $\mathbb{X}_m$, $\mathbb{Y}_m$

cost within the target zone ℓ_e^* as well as the parameters ε, the cost relaxation rate r and the maximum number of iterations $N_{\max}$. $N_{\max}$ is introduced in the algorithm to ensure that the while loop does not run indefinitely if a poor choice of the parameters are selected. The algorithm terminates with an empty set if the parameters are poorly chosen.

While the modified output zone $\mathbb{Y}_m$ is in a form which can be used in the NZEMPC optimization (6.17), this may not the best approach. This is because there is no guarantee that the set $\mathbb{Y}_m$ obtained from the projection of the ellipsoidal control invariant set $\mathbb{X}_m$ is also control invariant. Therefore, the ellipsoidal control invariant set $\mathbb{X}_m$ needs to be used the MPC algorithm. However, replacing the zone slack constraint (6.17h) in Optimization problem (6.17) will lead to having the number of slack variables equal to the dimension of the state of the system. Since in most control systems the dimension of the output is smaller than or equal to the dimension of the system states, the increased number of slack variables y_z will eventually increase the size of the optimization problem to be solved online. Thus to mitigate this problem, the zone constraint is modified to Equation (6.26h) and an additional positivity constraint (6.26i) is added. This results in only one slack variable with is independent of the input or the output dimension. Full details of the robust economic Model Predictive Control with zone tracking (RZEMPC) algorithm is presented in Optimization problem (6.26).

$$\min_{\mathbf{v},\mathbf{y_z}} \quad \int_{t_k}^{t_k+N\Delta} -y(t) + c_1(\|y_z(t)\|_2^2)dt \tag{6.26a}$$

$$s.t. \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t), v(t)) \tag{6.26b}$$

$$y(t) = h(\tilde{x}(t)) \tag{6.26c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{6.26d}$$

$$\tilde{x}(t) \in \mathbb{X} \tag{6.26e}$$

$$v(t) \in \mathbb{U} \tag{6.26f}$$

$$y(t) \in \mathbb{Y} \tag{6.26g}$$

$$(\tilde{x}(t) - x_s)^T P(\tilde{x}(t) - x_s) \leq 1 + y_z(t) \tag{6.26h}$$

$$y_z(t) \geq 0 \tag{6.26i}$$

The constraints in Optimization problem (6.26) are the same as previously defined.

6.4 Simulation results

In this section, we present different set of simulations to demonstrate the effectiveness and applicability of the proposed EMPC algorithm. We compare the results of the control algorithm with the modified zone to that of the controller with the original target zone. Specifically, we present the performance of the controllers under additive state uncertainties which represent various kinds of uncertainties in the process model. We also compare the performance of the controllers under time-varying inlet flue gas flow rate.

We begin this section with the simulation settings and model parameters used in this work. Then, we investigate the effects of additive state uncertainty that may be present in the process. Finally, we analyze and compare the performance of the controllers when the inlet flue gas flow rate vary.

Table 6.2: CO_2 gas absorption column configuration

Property	Value
Column internal diameter D_c (m)	0.43
Packing height (m)	6.1
Packing type	IMTP #40
Nominal packing size (m)	0.038
Specific packing area (m^2)	143.9

Table 6.3: Nominal flue gas condition

Property	Value
Temperature (K)	319.70
Volumetric Flow rate F_G (m^3/s)	0.0832
CO_2 mole fraction	0.1500
N_2 mole fraction	0.8000
MEA mole fraction	0.0000
H_2O mole fraction	0.0500

Table 6.4: Nominal inlet amine solvent condition

Property	Value
Temperature (K)	314.0
CO_2 mole fraction	0.0266
N_2 mole fraction	0.0000
MEA mole fraction	0.1104
H_2O mole fraction	0.8630

6.4.1 Simulation settings

The plant configuration for the process model described in Section 6.2.2 is determined according to Decardi-Nelson et al. [59] and is presented in Table 6.2.

The properties of the inlet flue gas entering the column from the bottom and the inlet solvent entering the column from the top are also shown in Tables 6.3 and 6.4 respectively. In this work, we assume that the properties of the solvent entering the absorption column from the desorption unit is fixed with the exception of the flow rate which is manipulated. The flue gas flow rate to the column may vary but this is unknown to the controller.

In this work, we assume that all the states are measured and available to the controller at any sampling time $t_{k\geq 0}$ with a sampling interval Δ set at 10 minutes. This is a reasonable assumption since it has been shown that only the temperature measurements, which can be easily obtained, can be used to reconstruct the full states of the absorption column [96]. Unless otherwise stated, the prediction and control horizon N of the controllers is set at 10. The parameters ε, r, α and $N_{\max}$ in Algorithm 1 are fixed at 0.009, -0.005, 5, and 10 respectively. The value of $u_{\max}$ was fixed at 20 % of the available input energy for control and the identity matrix was used as the value of Q. The output zone to be tracked was chosen to be between 0.85 and 0.90 i.e. $\mathbb{Y}_z = [0.85\ 0.90]$ with a target zone tracking weight $c_1 = 10000$. This ensures that a high absorption efficiency is not pursued by the controller due to the economic objective which can cause operational issues such as flooding of the column and solvent over-circulation at high liquid flow rates. This also prevents the controller from allowing large quantities of CO_2 to be released into the atmosphere resulting in higher CO_2 taxes. In the implementation of the control algorithms, the system states were scaled such that

$$\hat{x}(t_k) = x(t)/x_{\text{scale}}, \quad \hat{u}(t_k) = u(t)/u_{\text{scale}} \tag{6.27}$$

where $\hat{x}$ and $\hat{u}$ are the scaled states and inputs respectively, and x_{scale} and u_{scale} are the steady-state values corresponding to the center of the zone i.e. an absorption efficiency of 0.875. The additive state uncertainty w in (6.9) was assumed to be uniformly distributed in $[-0.00001 \times \mathbf{1}\ 0.00001 \times \mathbf{1}]$ where $\mathbf{1} = x_{\text{scale}}/x_{\text{scale}}$ is a vector of ones having the same dimension as the state. The nonlinear process model (6.13) was used in all the simulations.

6.4.2 Results and discussion

In this section, we present the results for the two control algorithms namely EMPC with target zone tracking (NZEMPC) and EMPC with modified zone tracking (RZEMPC). We

Table 6.5: Comparison of the nominal EMPC with target zone tracking and the two EMPCs with modified zone tracking (smaller is better).

Controller	Average cost
EMPC tracking the target zone	−0.72785
EMPC tracking the modified zone	−0.88639
EMPC tracking the center of target zone	−0.86811

also consider the case where the modified zone is at the center of the target zone. This case was added to illustrate the notion that arbitrarily tracking a zone within the center of the target zone, though easy, may not be the best strategy to ensure finite-time zone tracking with good performance. Performance or cost as defined in this section refers to Equation (6.15) which represents both the economic performance and the ability of the controller to ensure that the output is within target zone at all times.

6.4.2.1 Additive state uncertainty

The average performance of the controllers for the case of additive state disturbances are shown in Table 6.5. As can be seen from Table 6.5, our proposed zone EMPC control algorithm with modified target zone outperforms that of the controller that tracks the original target zone. The EMPC with modified target zone at the center of the zone performs better than tracking the original target zone but does not perform better than that with the modified target zone. This implies that the parameters in Algorithm 1 need to be carefully tuned to ensure that both the performance and the zone tracking objectives are achieved. One way to do this is to consider that uncertainty information when selecting ε as this ensures a reasonable back-off from the boundary of the target zone.

To understand why this happens, Figures 6.3 and 6.4 have been provided. As mentioned earlier, the presence of the secondary economic objective can cause the system output to operate close to the boundary of the target zone. This is the case for the operation of the absorption column since the economic objective is to maximize the absorption efficiency.

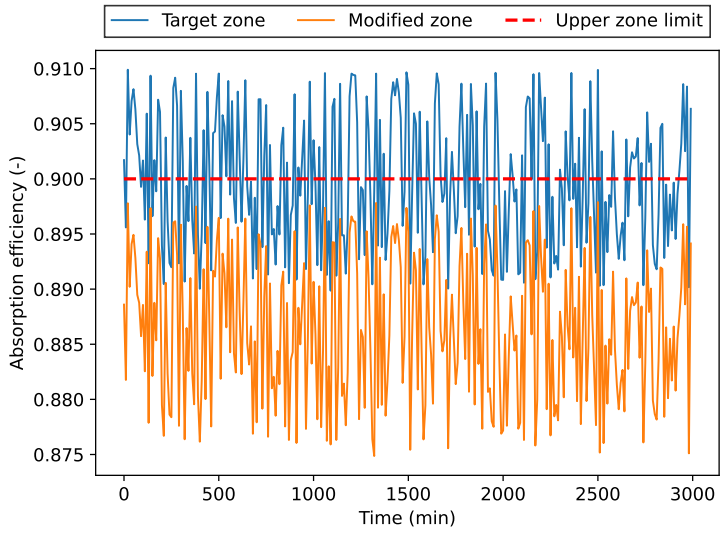


Figure 6.3: Trajectories of the absorption efficiency (y) for the absorption column under the operation of the zone EMPC control algorithm tracking the original target zone (blue) and modified zone (orange). Target zone: EMPC tracking the target zone; Modified zone: EMPC tracking the modified zone; Upper zone limit: upper bound of the target zone.

The presence of the uncertainty causes the system output to move out of the target leading to high cost and inability to track the target zone. By modifying the target zone, our proposed controller ensures that there is room available for the system to operate within the target zone even in the presence of the uncertainty. It is worth mentioning that, by modifying the target zone, some performance is sacrificed in favour of achieving the zone tracking objective. This can be seen in Figure 6.3 where the controller tracking the target zone operates at a higher absorption efficiency compared to the one that tracks the modified target zone. Because of the target zone violation, it can be seen in Figure 6.4 that the cost trajectory of the NZEMPC is erratic compared to that of the RZEMPC.

6.4.2.2 Time-varying flue gas flow rate

The operation of a typical power plant is usually periodic every day and seasonally. This is because of the variation in electricity demand. Electricity demand is usually low in the early morning and very late at night where consumer activity is low. It gradually rises to

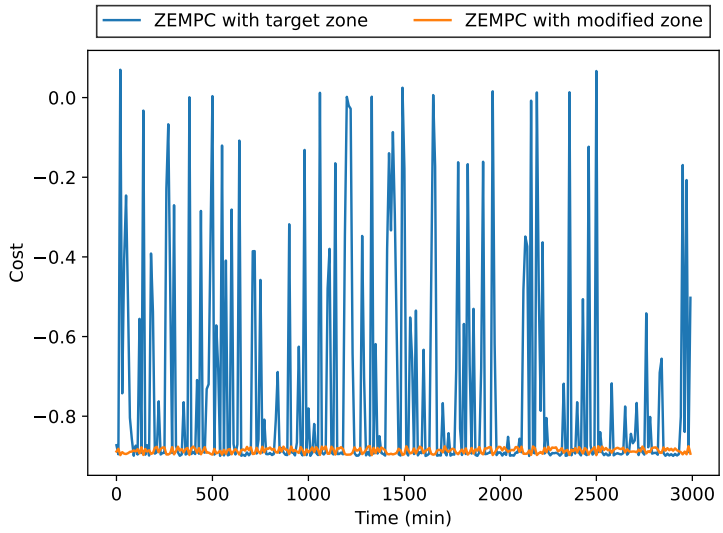


Figure 6.4: Trajectories of the stage cost for the absorption column under the operation of the zone EMPC control algorithm tracking the original target zone (blue) and modified zone (orange)

a peak around noon and stays there for sometime before finally reducing again at night. Furthermore, it has been suggested that renewable energy sources be integrated into the energy generation mix with the renewable energy sources being the main power generation sources and the fossil fuel power plants as backups. However, some renewable energy sources may not be very reliable. For example, the ability of a solar panel to generate electricity depends on the availability of sunlight which may not always be available. This integrated energy mix will therefore further cause more erratic operation of the fossil fuel power plant. A consequence of this changes in the demand and subsequently the output of the power plant is that the flue gas emanating from the power plant to the absorption column will vary frequently. This time-varying behaviour can have significant effects on the performance of the PCC plant attached to the fossil fueled power plant. Therefore, flexible operation of the PCC plant attached to the power plant is inevitable. This has been the subject of several studies in the control of PCC plant attached to a load-following power plant [88, 89, 59, 91].

We compare the performance of the EMPC with target zone tracking (NZEMPC) to that of the EMPC with modified zone tracking (RZEMPC) under a time-varying flue gas flow

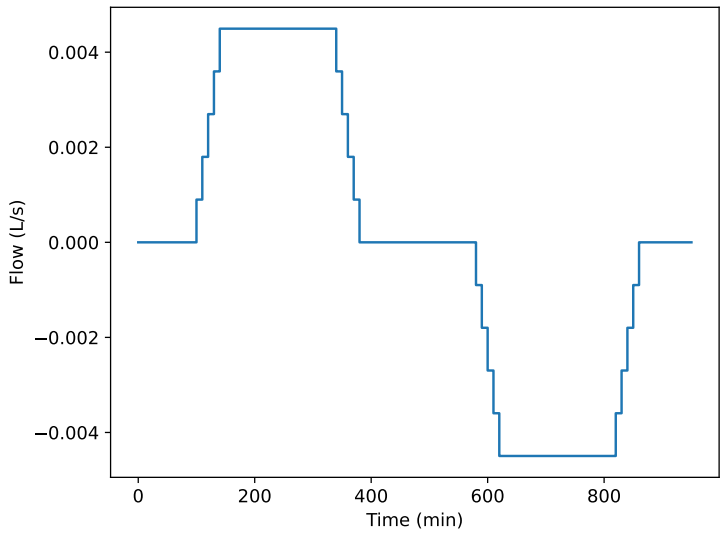


Figure 6.5: Generated trajectories of the disturbance of the flue gas flow rate signifying ramping up and ramping down operations of a power plant

Table 6.6: Comparison of the average cost of NZEMPC and RZEMPC under time-varying flue gas flow rate (smaller is better)

Controller	Ramping up cost	Ramping down cost	Average overall cost
EMPC tracking the target zone	−0.8948	−0.4525	−0.6690
EMPC tracking the modified zone	−0.8813	−0.8916	−0.8866

rate setting. This was achieved by varying the flue gas flow rate using the disturbance shown in Figure 6.5. The disturbance to the flue gas mimics typical ramp up and ramp down behaviour of a power plant. The average costs of the operation of the absorption column under the two controllers is presented in Table 6.6. As can be seen the EMPC with modified zone tracking yields a better overall cost on average than that of the EMPC with target zone tracking. The reason for the poor performance in the NZEMPC is the same as explained in the earlier section. The economic objective drives the absorption efficiency to the boundary of the target zone which leads to a target zone violation once the disturbance is present. The RZEMPC on the other hand causes the system to operate away from the boundary thus making room for the effects of the disturbance. This ensures that the absorption

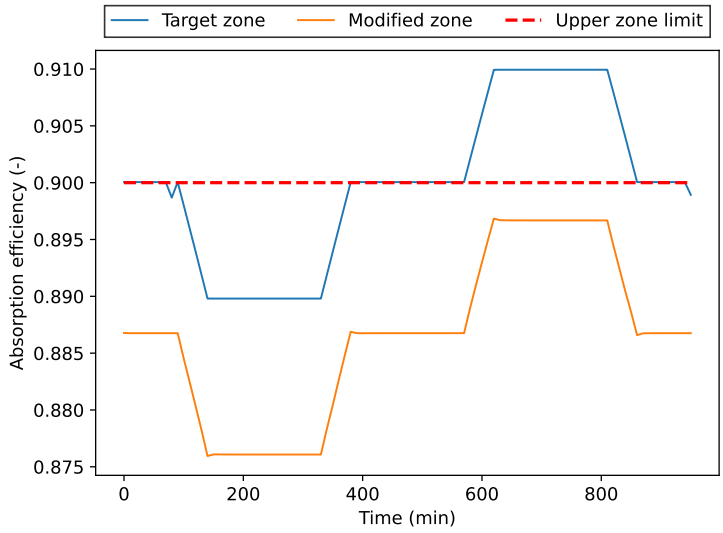


Figure 6.6: Trajectories of the absorption efficiency for the absorption column under the operation of the zone EMPC control algorithm tracking the original target zone (blue) and modified zone (orange) for the time-varying flue gas scenario

efficiency stays within the target zone at all times which leads to a better cost on average. The absorption efficiency trajectory can be seen in Figure 6.6. It can be seen that in both cases, the absorption efficiency decreases during the ramp up and increases during the ramp down. This is because when the flue gas flow rate increase, the amount of CO_2 entering the column also increase. The controllers still try to capture the same amount of CO_2 since the process model used in the controller uses the nominal flue gas flow rate. A careful look at the cost trajectories in Figure 6.7 and as shown in Table 6.6 shows that during the ramping up phase, the NZEMPC yields a better cost than that of the RZEMPC. However, the NZEMPC performs poorly during the ramping down phase. The RZEMPC on the other hand ensures a fairly constant cost throughout the operation. This shows the benefits of modifying the target zone to ensure finite time zone tracking.

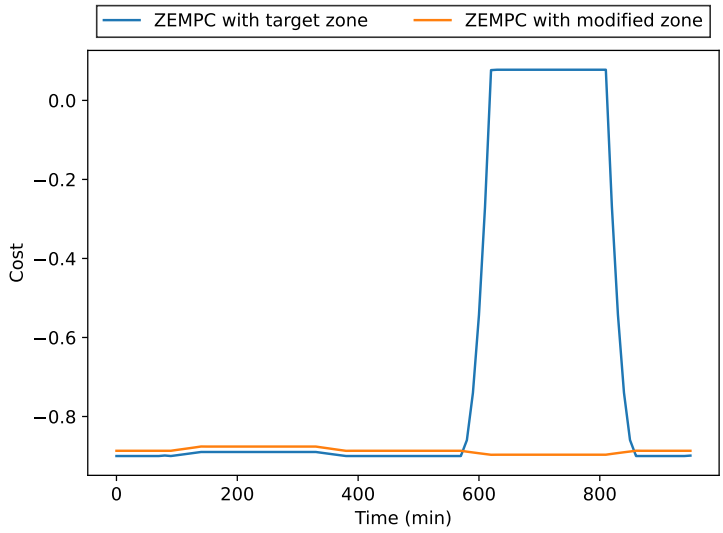


Figure 6.7: Trajectories of the stage cost for the absorption column under the operation of the zone EMPC control algorithm tracking the original target zone (blue) and modified zone (orange) for the time-varying flue gas flow rate

6.5 Concluding remarks

In this chapter, a control problem which typically arises in the operation of the absorption unit in a post-combustion CO_2 capture plant is addressed using an EMPC with zone tracking formulation. This helps to avoid the problems of solvent overcirculation and flooding in the column during operation. To ensure that finite-time zone tracking objective is achieved in the presence of modelling uncertainty, the target zone to be tracked is modified. The proposed zone modification algorithm makes use of ellipsoidal control invariant set and a back-off strategy which is scalable for systems with large number of states such as the absorption column. The use of the control invariant set as the zone ensures that the zone can be tracked since there is no guarantee that the original target zone is control invariant. It was shown that the EMPC with modified zone tracking performed better than the EMPC with target zone tracking in the presence of model uncertainties and exogenous disturbances. Finally, the simulation example demonstrates the efficacy of the proposed EMPC algorithm with zone tracking as a effective control strategy for the absorption column of a typical post-combustion

CO_2 capture plant.

Chapter 7

Concluding remarks and future work

7.1 Introduction

This thesis presented methods to compute the largest (robust) control invariant sets of constrained nonlinear systems using graph theoretical algorithms. In the systems and control literature, computing the control invariant sets for constrained nonlinear systems has received less attention than the linear counterpart. Moreover, in most cases, the nonlinear system are linearized leading to conservative results. Then again, the issue of scalability of the present algorithms for computing control invariant sets for higher dimensional systems is still an open problem.

In this thesis, we tackled the problem of computing control invariant sets from multiple facets. We proposed an algorithm for determining inner and outer approximations of control invariant sets of general nonlinear systems with guarantees of finding the control invariant set (if it exists). We then provided several avenues for improving the computational complexity of the proposed algorithms including adaptive subdivision, parallelization using GPU and system decomposition. Thereafter, we demonstrated the use control invariant sets to improve the robustness and asymptotic performance of zone EMPC using several examples.

This chapter summarizes the overall findings in this thesis and provides discussions on

possible research directions based on the findings in this study.

7.2 Concluding remarks

In Chapter 2, we proposed a general framework for computing the largest (robust) control invariant sets of constrained nonlinear systems based on graph theory. A mathematical analysis of the proposed algorithm showed that the algorithm is able to converge to the largest (robust) control invariant set which demonstrates the theoretical soundness of the proposed algorithm. Owing to the challenges of numerical implementation of the algorithm, algorithms to compute the inner and outer approximations of the control invariant sets were proposed. The efficacy of the proposed algorithms were compared to well-known algorithms in literature using both a linear and a nonlinear example.

Chapters 3 and 4 are extensions of the algorithms proposed in Chapter 2. In Chapter 3, we systematically analyzed the framework proposed in Chapter 2 and determined the bottlenecks which include graph construction and cell subdivision steps. Thereafter, we proposed ways to improve the algorithm using an adaptive subdivision scheme and parallelization using graphics processing units. An example using a continuous stirred tank reactor was used to demonstrate the effectiveness of the measures proposed to improve the graph-based invariant set algorithm. In Chapter 4, we further tackled the issue of scalability of the GIS algorithm using a decomposition-based approach. By decomposing the overall system into smaller subsystems, computationally manageable problems can be solved. A notable aspect of the proposed decomposition-based GIS algorithm is that the states that couple the subsystems together are not treated as disturbances contrary to other decomposition-based algorithms. We showed that the algorithm based on system decomposition is able to converge to the centralized algorithm results. We demonstrated the suitability of the decomposition-based algorithm using several cascade system examples including a six dimensional CSTR example.

Chapters 5 and 6 are applications of robust control invariant in the design of economic

model predictive control. In Chapter 5, we developed a robust economic model predictive control with zone tracking scheme for control of nonlinear systems. Specifically, we introduced the concept of risk factor in the selection of a subset of the zone to be tracked by the controller. The subset of the zone to be tracked was selected such that it is robust control invariant for stability guarantees. We conducted rigorous stability analysis on the proposed control algorithm and demonstrated the applicability of the algorithm to chemical processes using a CSTR example. In Chapter 6, we extended and applied the robust economic model predictive control with zone tracking scheme presented in Chapter 5 to a large scale chemical process. Specifically, we considered a typical absorption column of a post-combustion CO_2 capture plant. Because of the scale of the system, we proposed an ellipsoidal control invariant set as the subset of the zone to be tracked. This removed the limitation on the applicability of the RZEMPC scheme to large scale systems. We used several scenarios to demonstrate the suitability of the proposed RZEMPC scheme to improve the operation of the absorption column under uncertainty in the the flue gas flow rate and endogenous model uncertainty.

7.3 Future research directions

There are several exciting future research directions in the development of graph-based invariant set algorithms for nonlinear systems. Below, a few discussions on possible research directions are provided.

Interdependent graph networks: In the decomposition-based GIS algorithm, the solution of the overall system need to be reconstructed from the solution of the subsystems and then some cells tested for invariance. This step could be computationally demanding requiring large amount of memory and computation resources. We propose that the reconstruction and validation step may be eliminated by utilizing recent advances in interdependent graph networks to analyze the distributed graphs for control invariance instead. Two of such possi-

ble interdependent network analysis tools are percolation theory and cascading failure theory [97].

Integration of the various techniques into a single framework: In this thesis, we have developed a graph-based invariant set algorithm for nonlinear systems and proposed several ways to improve the computational efficiency including system decomposition, adaptive subdivision and parallelization. As it stands now, the various methods are not integrated into a single framework. It will be worth knowing the overall computational efficiency when the various methods are integrated into a single framework. We note that the integration is not trivial. Several issues such as convergence, missing state estimation, tuning of the parameters and inner approximation need to be addressed in the integrated framework.

Invariant set representation: The GIS algorithm produces sets which may not necessarily be convex. While it is easy to represent convex sets in controller design such as MPC using the convex hull algorithm, it is in general difficult to obtain a representation of non-convex invariant sets. A possible direction is to find the largest convex invariant set for MPC applications. Another possible direction is to use semi-algebraic sets [98] or deep neural networks to represent the non-convex sets.

Extraction of the control law: Currently, the algorithms proposed in this thesis compute the largest CIS or RCIS without determining the appropriate control law that enforces the control invariance. A possible future direction is to use reinforcement learning to determine a general control law that enforces the control invariance for a given CIS or RCIS.

Bibliography

[1] M. Rungger and P. Tabuada, "Computing robust controlled invariant sets of linear systems," *IEEE Transactions on Automatic Control*, vol. 62, pp. 3665–3670, July 2017.

[2] M. Fiacchini, T. Alamo, and E. Camacho, "On the computation of convex robust control invariant sets for nonlinear systems," *Automatica*, vol. 46, pp. 1334–1338, Aug. 2010.

[3] A. Nikolopoulou and M. G. Ierapetritou, "Optimal design of sustainable chemical processes and supply chains: A review," *Computers & Chemical Engineering*, vol. 44, p. 94–103, 2012.

[4] M. Mahmood and P. Mhaskar, "Enhanced stability regions for model predictive control of nonlinear process systems," *AIChE Journal*, vol. 54, no. 6, p. 1487–1498, 2008.

[5] T. Homer and P. Mhaskar, "Constrained control lyapunov function-based control of nonlinear systems," *Systems & Control Letters*, vol. 110, pp. 55–61, 2017.

[6] I. Mitchell, A. Bayen, and C. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on Automatic Control*, vol. 50, pp. 947–957, jul 2005.

[7] T. Homer, M. Mahmood, and P. Mhaskar, "A trajectory-based method for constructing null controllable regions," *International Journal of Robust and Nonlinear Control*, vol. 30, no. 2, pp. 776–786, 2020.

[8] J. P. Aubin, *Viability theory*. Modern Birkhäuser classics, Boston: Birkhäuser, 2009.

[9] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[10] D. Q. Mayne, "Control of constrained dynamic systems," *European Journal of Control*, vol. 7, no. 2-3, pp. 87–99, 2001.

[11] M. Cannon, V. Deshmukh, and B. Kouvaritakis, "Nonlinear model predictive control with polytopic invariant sets," *Automatica*, vol. 39, no. 8, pp. 1487–1494, 2003.

[12] S. Liu and J. Liu, "Economic model predictive control with zone tracking," *Mathematics*, vol. 6, no. 5, p. 65, 2018.

[13] S. Liu, Y. Mao, and J. Liu, "Model predictive control with generalized zone tracking," *IEEE Transactions on Automatic Control*, 2019.

[14] Z. Artstein, "Stabilization with relaxed controls," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 7, p. 1163–1173, Jan 1983.

[15] E. D. Sontag, "A 'universal' construction of artstein's theorem on nonlinear stabilization," *Systems & Control Letters*, vol. 13, p. 117–123, Aug 1989.

[16] H. Khalil, *Nonlinear Systems*. Pearson Education, Prentice Hall, 2002.

[17] S. Munir, M. Hovd, and S. Olaru, "Low complexity constrained control using higher degree lyapunov functions," *Automatica*, vol. 98, p. 215–222, Dec 2018.

[18] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*. SIAM, 1994.

[19] M. Han, Y. Tian, L. Zhang, J. Wang, and W. Pan, "Reinforcement learning control of constrained dynamic systems with uniformly ultimate boundedness stability guarantee," *Automatica*, vol. 129, p. 109689, Jul 2021.

[20] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, vol. 64, p. 2737–2752, Jul 2019.

[21] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration," *2018 IEEE Conference on Decision and Control (CDC)*, Dec 2018.

[22] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Advances in neural information processing systems*, vol. 30, 2017.

[23] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*. Springer International Publishing, 2015.

[24] S. V. Rakovic, E. C. Kerrigan, K. I. Kouramas, and D. Q. Mayne, "Invariant approximations of the minimal robust positively invariant set," *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 406–410, 2005.

[25] E. C. Kerrigan, *Robust constraint satisfaction: Invariant sets and predictive control*. PhD thesis, University of Cambridge, 2001.

[26] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Mathematical problems in engineering*, vol. 4, no. 4, pp. 317–367, 1998.

[27] E. G. Gilbert and K. T. Tan, "Linear systems with state and control constraints: the theory and application of maximal output admissible sets," *IEEE Transactions on Automatic Control*, vol. 36, pp. 1008–1020, Sep. 1991.

[28] D. Bertsekas, "Infinite time reachability of state-space regions by using feedback control," *IEEE Transactions on Automatic Control*, vol. 17, pp. 604–613, October 1972.

[29] T. Alamo, A. Cepeda, M. Fiacchini, and E. F. Camacho, "Convex invariant sets for discrete-time lur'e systems," *Automatica*, vol. 45, no. 4, pp. 1066–1071, 2009.

[30] J. M. Bravo, D. Limón, T. Alamo, and E. F. Camacho, "On the computation of invariant sets for constrained nonlinear systems: An interval arithmetic approach," *Automatica*, vol. 41, no. 9, pp. 1583–1589, 2005.

[31] T. Homer and P. Mhaskar, "Utilizing null controllable regions to stabilize input-constrained nonlinear systems," *Computers & Chemical Engineering*, vol. 108, pp. 24–30, 2018.

[32] B. Decardi-Nelson and J. Liu, "Computing robust control invariant sets of constrained nonlinear systems: A graph algorithm approach," *Computers & Chemical Engineering*, vol. 145, p. 107177, 2021.

[33] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis: internal approximation," *Systems & control letters*, vol. 41, no. 3, pp. 201–211, 2000.

[34] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, "Spaceex: Scalable verification of hybrid systems," in *International Conference on Computer Aided Verification*, pp. 379–395, Springer, 2011.

[35] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin, "Decomposition of reachable sets and tubes for a class of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 11, pp. 3675–3688, 2018.

[36] S. Riverso, K. Kouramas, and G. Ferrari-Trecate, "Decentralized and distributed robust control invariance for constrained linear systems," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 5978–5984, IEEE, 2017.

[37] S. V. Raković, B. Kern, and R. Findeisen, "Practical set invariance for decentralized discrete time systems," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 3283–3288, IEEE, 2010.

[38] A. Li and M. Chen, "Guaranteed-safe approximate reachability via state dependency-based decomposition," in *2020 American Control Conference (ACC)*, pp. 974–980, IEEE, 2020.

[39] M. Korda, D. Henrion, and C. N. Jones, "Convex computation of the maximum controlled invariant set for polynomial control systems," *SIAM Journal on Control and Optimization*, vol. 52, p. 2944–2969, Jan 2014.

[40] M. A. Ben Sassi and A. Girard, "Controller synthesis for robust invariance of polynomial dynamical systems using linear programming," *Systems & Control Letters*, vol. 61, p. 506–512, Apr 2012.

[41] G. S. Osipenko, "On a symbolic image of dynamical system," *Boundary Value Problems*, vol. , pp. Interuniv. Collect, Sci. Works, perm (in Russian) 101–105, 1983.

[42] M. Eidenschink, "Exploring global dynamics: A numerical algorithm based on the conley index theory.," 1997.

[43] K. Mischaikow, "Topological techniques for efficient rigorous computation in dynamics," *Acta Numerica*, vol. 11, pp. 435–477, 2002.

[44] D. Szolnoki, "Set oriented methods for computing reachable sets and control sets," *Discrete and Continuous Dynamical Systems - Series B*, vol. 3, pp. 361–382, May 2003.

[45] G. Osipenko, *Dynamical Systems, Graphs, and Algorithms*. No. 1889 in Lecture Notes in Mathematics, Berlin ; New York: Springer, 2007. OCLC: ocm75927357.

[46] F. Colonius and W. Kliemann, *The dynamics of control*. Springer Science & Business Media, 2012.

[47] P. Cardaliaguet, M. Quincampoix, and P. Saint-Pierre, "Some algorithms for differential games with two players and one target," *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, vol. 28, no. 4, pp. 441–461, 1994.

[48] K. Sakai, "Pseudo-orbit tracing property and strong transversality of diffeomorphisms on closed manifolds," *Osaka Journal of Mathematics*, vol. 31, no. 2, pp. 373–386, 1994.

[49] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to interval analysis*, vol. 110. Siam, 2009.

[50] M. Dellnitz and O. Junge, "Set Oriented Numerical Methods for Dynamical Systems," in *Handbook of Dynamical Systems*, vol. 2, pp. 221–264, Elsevier, 2002.

[51] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[52] M. Dellnitz and O. Junge, "An adaptive subdivision technique for the approximation of attractors and invariant measures," *Computing and Visualization in Science*, vol. 1, no. 2, p. 63–68, 1998.

[53] W. T. Trotter and P. Erdös, "When the cartesian product of directed cycles is hamiltonian," *Journal of Graph Theory*, vol. 2, no. 2, p. 137–142, 1978.

[54] S. Kaynama and M. Oishi, "A modified riccati transformation for decentralized computation of the viability kernel under LTI dynamics," *IEEE Transactions on Automatic Control*, vol. 58, p. 2878–2892, Nov 2013.

[55] J. B. Rawlings, D. Angeli, and C. N. Bates, "Fundamentals of economic model predictive control," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 3851–3861, IEEE, 2012.

[56] S. Liu and J. Liu, "Economic model predictive control with extended horizon," *Automatica*, vol. 73, pp. 180–192, 2016.

[57] M. Ellis, H. Durand, and P. D. Christofides, "A tutorial review of economic model predictive control methods," *Journal of Process Control*, vol. 24, no. 8, pp. 1156–1178, 2014.

[58] S. Liu, J. Zhang, and J. Liu, "Economic MPC with terminal cost and application to an oilsand primary separation vessel," *Chemical Engineering Science*, vol. 136, pp. 27–37, 2015.

[59] B. Decardi-Nelson, S. Liu, and J. Liu, "Improving flexibility and energy efficiency of post-combustion CO_2 capture plants using economic model predictive control," *Processes*, vol. 6, no. 9, p. 135, 2018.

[60] Y. Zhang, B. Decardi-Nelson, J. Liu, J. Shen, and J. Liu, "Zone economic model predictive control of a coal-fired boiler-turbine generating system," *Chemical Engineering Research and Design*, vol. 153, pp. 246–256, 2020.

[61] D. W. Griffith, V. M. Zavala, and L. T. Biegler, "Robustly stable economic nmpc for non-dissipative stage costs," *Journal of Process Control*, vol. 57, pp. 116–126, 2017.

[62] D. Angeli, R. Amrit, and J. B. Rawlings, "On average performance and stability of economic model predictive control," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1615–1626, 2011.

[63] R. Amrit, J. B. Rawlings, and D. Angeli, "Economic optimization using model predictive control with a terminal cost," *Annual Reviews in Control*, vol. 35, no. 2, pp. 178–186, 2011.

[64] D. Mayne, "Robust and stochastic model predictive control: Are we going in the right direction?," *Annual Reviews in Control*, vol. 41, pp. 184–192, 2016.

[65] R. Huang, L. T. Biegler, and E. Harinath, "Robust stability of economically oriented infinite horizon NMPC that include cyclic processes," *Journal of Process Control*, vol. 22, no. 1, pp. 51–59, 2012.

[66] S. Lucia, J. A. Andersson, H. Brandt, M. Diehl, and S. Engell, "Handling uncertainty in economic nonlinear model predictive control: A comparative case study," *Journal of Process Control*, vol. 24, no. 8, pp. 1247–1259, 2014.

[67] Y. Mao, S. Liu, and J. Liu, "Robust economic model predictive control of nonlinear networked control systems with communication delays," *International Journal of Adaptive Control and Signal Processing*, vol. 34, no. 5, pp. 614–637, 2020.

[68] F. A. Bayer, M. A. Müller, and F. Allgöwer, "Tube-based robust economic model predictive control," *Journal of Process Control*, vol. 24, no. 8, pp. 1237–1246, 2014.

[69] F. A. Bayer, M. Lorenzen, M. A. Müller, and F. Allgöwer, "Robust economic model predictive control using stochastic information," *Automatica*, vol. 74, pp. 151–161, 2016.

[70] Z. Dong and D. Angeli, "Tube-based robust economic model predictive control on dissipative systems with generalized optimal regimes of operation," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 4309–4314, IEEE, 2018.

[71] B. Grosman, E. Dassau, H. C. Zisser, L. Jovanovič, and F. J. Doyle III, "Zone model predictive control: a strategy to minimize hyper-and hypoglycemic events," *Journal of Diabetes Science and Technology*, vol. 4, no. 4, pp. 961–975, 2010.

[72] S. Privara, J. Širokỳ, L. Ferkl, and J. Cigler, "Model predictive control of a building heating system: The first experience," *Energy and Buildings*, vol. 43, no. 2-3, pp. 564–572, 2011.

[73] Y. Mao, S. Liu, J. Nahar, J. Liu, and F. Ding, "Soil moisture regulation of agro-hydrological systems using zone model predictive control," *Computers and Electronics in Agriculture*, vol. 154, pp. 239–247, 2018.

[74] M. Heidarinejad, J. Liu, and P. D. Christofides, "Economic model predictive control of nonlinear process systems using lyapunov techniques," *AIChE Journal*, vol. 58, no. 3, pp. 855–870, 2012.

[75] M. Diehl, R. Amrit, and J. B. Rawlings, "A lyapunov function for economic optimizing model predictive control," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 703–707, 2010.

[76] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[77] I. Dunning, J. Huchette, and M. Lubin, "Jump: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[78] M. Höök and X. Tang, "Depletion of fossil fuels and anthropogenic climate change—a review," *Energy Policy*, vol. 52, p. 797–809, Jan 2013.

[79] C. Bataille, H. Waisman, M. Colombier, L. Segafredo, J. Williams, and F. Jotzo, "The need for national deep decarbonization pathways for effective climate policy," *Climate Policy*, vol. 16, no. sup1, p. S7–S26, 2016.

[80] British Petroleum Company, *BP statistical review of world energy.* 70th ed., 2021.

[81] P. Babu, R. Kumar, and P. Linga, "Pre-combustion capture of carbon dioxide in a fixed bed reactor using the clathrate hydrate process," *Energy*, vol. 50, p. 364–373, Feb 2013.

[82] A. Chansomwong, K. Zanganeh, A. Shafeen, P. Douglas, E. Croiset, and L. Ricardez-Sandoval, "Dynamic modelling of a co2 capture and purification unit for an oxy-

coal-fired power plant," *International Journal of Greenhouse Gas Control*, vol. 22, p. 111–122, Mar 2014.

[83] B. Decardi-Nelson, A. Akachuku, P. Osei, W. Srisang, F. Pouryousefi, and R. Idem, "A flexible and robust model for low temperature catalytic desorption of co 2 from co 2 -loaded amines over solid acid catalysts," *Chemical Engineering Science*, vol. 170, p. 518–529, Oct 2017.

[84] J. Davison, L. Mancuso, and N. Ferrari, "Costs of CO_2 capture technologies in coal fired power and hydrogen plants," *Energy Procedia*, vol. 63, p. 7598–7607, 2014.

[85] R. Sakwattanapong, A. Aroonwilas, and A. Veawab, "Behavior of reboiler heat duty for CO_2 capture plants using regenerable single and blended alkanolamines," *Industrial & Engineering Chemistry Research*, vol. 44, no. 12, p. 4465–4473, 2005.

[86] M. Panahi and S. Skogestad, "Economically efficient operation of co_2 capturing process. part ii. design of control layer," *Chemical Engineering and Processing: Process Intensification*, vol. 52, p. 112–124, 2012.

[87] Z. He, M. H. Sahraei, and L. A. Ricardez-Sandoval, "Flexible operation and simultaneous scheduling and control of a co_2 capture plant using model predictive control," *International Journal of Greenhouse Gas Control*, vol. 48, p. 300–311, 2016.

[88] T. Bankole, D. Jones, D. Bhattacharyya, R. Turton, and S. E. Zitney, "Optimal scheduling and its lyapunov stability for advanced load-following energy plants with CO_2 capture," *Computers & Chemical Engineering*, vol. 109, p. 30–47, 2018.

[89] G. D. Patron and L. Ricardez-Sandoval, "A robust nonlinear model predictive controller for a post-combustion CO_2 capture absorber unit," *Fuel*, vol. 265, p. 116932, 2020.

[90] G. D. Patrón and L. Ricardez-Sandoval, "An integrated real-time optimization, control, and estimation scheme for post-combustion CO_2 capture," *Applied Energy*, vol. 308, p. 118302, 2022.

[91] J. Rúa, M. Bui, L. O. Nord, and N. Mac Dowell, "Does ccs reduce power generation flexibility? a dynamic study of combined cycles with post-combustion CO_2 capture," *International Journal of Greenhouse Gas Control*, vol. 95, p. 102984, 2020.

[92] B. Decardi-Nelson and J. Liu, "Robust economic model predictive control with zone tracking," *Chemical Engineering Research and Design*, vol. 177, p. 502–512, 2022.

[93] M. Wang, A. Lawal, P. Stephenson, J. Sidders, and C. Ramshaw, "Post-combustion CO_2 capture with chemical absorption: A state-of-the-art review," *Chemical engineering research and design*, vol. 89, no. 9, pp. 1609–1624, 2011.

[94] T. H. Chilton and A. P. Colburn, "Mass transfer (absorption) coefficients prediction from data on heat transfer and fluid friction," *Industrial & Engineering Chemistry*, vol. 26, no. 11, p. 1183–1187, 1934.

[95] B. Polyak and P. Shcherbakov, "Ellipsoidal approximations to attraction domains of linear systems with bounded control," *2009 American Control Conference*, 2009.

[96] X. Yin, B. Decardi-Nelson, and J. Liu, "Distributed monitoring of the absorption column of a post-combustion CO_2 capture plant," *International Journal of Adaptive Control and Signal Processing*, vol. 34, no. 6, p. 757–776, 2019.

[97] D. Y. Kenett, J. Gao, X. Huang, S. Shao, I. Vodenska, S. V. Buldyrev, G. Paul, H. E. Stanley, and S. Havlin, "Network of interdependent networks: overview of theory and applications," *Networks of Networks: The Last Frontier of Complexity*, pp. 3–36, 2014.

[98] F. Dabbene, D. Henrion, and C. M. Lagoa, "Simple approximations of semialgebraic sets and their applications to control," *Automatica*, vol. 78, p. 110–118, Apr 2017.