Fast Predictive Control of Networked Energy Systems

by

Frank Fu-Han Chuang

A dissertation submitted in partial satisfaction of the requirements for the degree of
Doctor of Philosophy

in

Engineering - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Francesco Borrelli, Chair
Professor J. Karl Hedrick
Professor Laurent El Ghaoui

Fall 2015
Fast Predictive Control of Networked Energy Systems

Copyright 2015
by
Frank Fu-Han Chuang
Abstract

Fast Predictive Control of Networked Energy Systems

by

Frank Fu-Han Chuang

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Francesco Borrelli, Chair

In this thesis we study the optimal control of networked energy systems. Networked energy systems consist of a collection of energy storage nodes and a network of links and inputs which allow energy to be exchanged, injected, or removed from the nodes. The nodes may exchange energy between each other autonomously or via controlled flows between the nodes. Examples of networked systems include building heating, ventilation, and air conditioning (HVAC) systems and networked battery systems. In the building system example, the nodes of the system are rooms which store thermal energy in the air and other elements which have thermal capacity. The rooms transfer energy autonomously through thermal conduction, convection, and radiation. Thermal energy can be injected into or removed from the rooms via conditioned air or slabs. In the case of a networked battery system, the batteries store electrical energy in their chemical cells. The batteries may be electrically linked so that a controller can move electrical charge from one battery to another. Networked energy systems are typically large-scale (contain many states and inputs), affected by uncertain forecasts and disturbances, and require fast computation on cheap embedded platforms.

In this thesis, the optimal control technique we study is model predictive control for networked energy systems. Model predictive or receding horizon control is a time-domain optimization-based control technique which uses predictive models of a system to forecast its behavior and minimize a performance cost subject to system constraints [18]. In this thesis we address two primary issues concerning model predictive control for networked energy systems: robustness to uncertainty in forecasts and reducing the complexity of the large-scale optimization problem for use in embedded platforms. The first half of the thesis deals primarily with the efficient computation of robust controllers for dealing with random and adversarial uncertainties in the forecasts. We show that the exact control policies can be found efficiently for certain types of robust predictive control problems. The second half of the thesis deals with improving computation speed and memory usage through model reduction and exploitation of symmetries in the predictive control problem. We present a model reduction technique tailored for networked systems which preserves sparsity and thus greatly improves computation speed. We also show how to apply symmetry methods
for efficient computation of explicit model predictive controllers to systems which are not perfectly symmetric.
To my parents, Pei-Wen and Jiin-Haur, sisters, Jody and Helen, and grandma.
To my labmates and colleagues in the department of mechanical engineering. Your friendship and support over the years have been invaluable towards both my personal and academic development.
To my friends from UC Berkeley and Monta Vista. We spent the most formative years of our lives together and you have given me the motivation and drive to pursue my doctorate degree.
Contents

List of Figures iv
List of Tables vi

1 Introduction 1
   1.1 Mathematical Preliminaries and Notation ......................... 1
   1.2 Networked Energy Systems .................................... 2
   1.3 Optimization ............................................... 6
   1.4 Model predictive control .................................... 7
   1.5 Summary of contributions and relevant publications ............. 8

2 Certainty equivalent control 11
   2.1 Introduction ............................................... 11
   2.2 Problem Definition ........................................ 13
   2.3 Optimality of Certainty Equivalence ............................ 15
   2.4 Certainty Equivalence for Problems with Symmetries ............ 23
   2.5 Disturbances with Unbounded Support ........................... 31
   2.6 Numerical Examples ......................................... 33
   2.7 Conclusion ................................................ 38

3 Robust control against adversarial disturbances 40
   3.1 Introduction ............................................... 40
   3.2 Problem Definition ........................................ 41
   3.3 Min-Max Finite-Time Optimal Control ........................... 43
   3.4 Sufficient conditions ....................................... 44
   3.5 Verifying the propositions’ conditions .......................... 46
   3.6 Numerical example ........................................... 47
   3.7 Conclusion ................................................ 52

4 Model reduction 53
   4.1 Introduction ............................................... 53
4.2 Problem Definition .............................................................. 54
4.3 State and Input Aggregation .................................................. 55
4.4 Method for state and input aggregation .................................... 60
4.5 Numerical Examples ............................................................. 65
4.6 Conclusion ................................................................. 68

5 Robust Symmetric Model Predictive Control 69
5.1 Introduction ................................................................. 69
5.2 System setup ................................................................. 70
5.3 Robust MPC Design .......................................................... 70
5.4 System symmetry ............................................................. 71
5.5 Approximate symmetry ....................................................... 72
5.6 Symmetric controller design .................................................. 75
5.7 Real time evaluation of sub-optimality ...................................... 78
5.8 Numerical examples ........................................................... 80
5.9 Conclusion ................................................................. 82

6 Radiant slabs: A case study 83
6.1 Model Validation ............................................................... 83
6.2 Problem Formulation .......................................................... 84
6.3 Implementation ............................................................... 84
6.4 Comparison with existing controller ......................................... 85
6.5 Conclusion ................................................................. 87

Bibliography 90
# List of Figures

1.1 Networked nodes .......................................................... 2

2.1 (a) Set $\mathcal{P}_0$ of initial states $x_0 \in \mathcal{X}_0$ for which the certainty equivalence controller $\tilde{\pi}^*_0$ is optimal $\pi^*_0(x_0) = \tilde{\pi}^*_0(x_0)$ for $\beta = 0.2$. (b) $\beta = 0.5$ (c) $\beta = 0.8$ .......................... 20

2.2 (a) Set $\mathcal{P}_0$ of initial states $x_0 \in \mathcal{X}_0$ for which the certainty equivalence controller $\tilde{\pi}^*_0$ is optimal $\pi^*_0(x_0) = \tilde{\pi}^*_0(x_0)$ for $\alpha = 0.4$. (b) $\alpha = 0.6$ (c) $\alpha = 0.8$ ......................... 21

2.3 (a) Set $\mathcal{P}_0$ of initial states $x_0 \in \mathcal{X}_0$ for which the certainty equivalence controller $\tilde{\pi}^*_0$ is optimal $\pi^*_0(x_0) = \tilde{\pi}^*_0(x_0)$. (b) Set of representative $\mathcal{P}_0/\mathcal{G}$ initial states $x_0 \in \mathcal{X}_0$ for which the certainty equivalence controller $\tilde{\pi}^*_0$ is optimal $\pi^*_0(x_0) = \tilde{\pi}^*_0(x_0)$ .......................... 34

2.4 Set of representative $\mathcal{P}_0/\mathcal{G}$ initial states $x_0 \in \mathcal{X}_0$ for which the certainty equivalence controller $\tilde{\pi}^*_0$ is optimal $\pi^*_0(x_0) = \tilde{\pi}^*_0(x_0)$ .......................... 35

2.5 Battery ring network .......................................................... 36

2.6 Outside air temperature .................................................. 38

2.7 $x_0$ for which CE is exact .................................................. 39

3.1 Outside air temperature .................................................. 49

3.2 Set of valid states .......................................................... 51

3.3 Control action and system outputs ..................................... 52

4.1 Battery Network ............................................................. 66

5.1 Networked nodes .......................................................... 80

5.2 Closed-loop simulation of symmetry controller ..................... 81

6.1 East zone MPC model open-loop validation: (A) cooling mode; and (B) coasting mode .................................................. 84

6.2 Comparison of thermal comfort performance of MPC and heuristic control method based on EN 15251 Categories for a typical summer season in Sacramento,CA (June-August) .................................................. 86

6.3 Comparison of energy consumptions between MPC and heuristic methods (June-August) .................................................. 86

6.4 Comparison of Heuristic and MPC methods in control of zone operativetemperature (A) and radiant loop valve (B): East zone on July-09th. ................. 88
6.5 Comparison of Heuristic and MPC methods in control of zone operative temperature (A) and radiant loop valve (B): South zone on July-26th.
List of Tables

1.1 Model variables ................................................................. 4
2.1 Battery network without symmetry ...................................... 36
2.2 Battery network with symmetry .......................................... 36
3.1 Solution cost as function of initial state, $N = 7$ ................... 48
3.2 Computation time as function of horizon length, $x_0 = 0$ .......... 48
4.1 Comparison of Suboptimality for Constrained LQR ................ 66
4.2 Comparison of Average Computation Time and Memory Usage for Constrained LQR .................................................. 66
4.3 Comparison of Suboptimality for Soft-Constrained Problem ....... 68
4.4 Comparison of Suboptimality for Soft-Constrained Problem ....... 68
4.5 Comparison of Average Computation Time and Memory Usage for Soft-Constrained Problem ........................................ 68
Acknowledgments

My utmost gratitude to my advisor Francesco Borrelli for insisting on upholding us to the highest standards. This thesis would not have been possible without his guidance.

I’d like to thank Jingjuan (Dove) Feng and Fred Bauman for their support and collaboration on the David Brower Center project. Portions of this thesis include written work from a journal paper co-authored with the above mentioned individuals.

I’d also like to acknowledge my colleague Claus Danielson for his collaboration on the certainty equivalence paper. His work is also the inspiration for the chapter on robust symmetric MPC. I’d also like to thank Miroslav Baric for his initial suggestion to investigate robust symmetric controllers.

I’d like to thank my colleagues Yudong Ma, Tony Kelman, Sarah Koehler, Sergey Vichik, and Yadranko Matusko. Their input over the years have shaped portions of my research which you find in this thesis.
Chapter 1

Introduction

1.1 Mathematical Preliminaries and Notation

Let $H \in \mathbb{R}^{m \times n}$ and $k \in \mathbb{R}^m$. A polyhedron $P \subset \mathbb{R}^n$ is a set $P = \{x \in \mathbb{R}^n | Hx \leq k\}$, where the inequality is to be interpreted elementwise. A polytope $P \subset \mathbb{R}^n$ is a bounded polyhedron.

Let $P \subset \mathbb{R}^n$ and $Q \subset \mathbb{R}^n$ be two sets. The Minkowski sum of $P$ and $Q$ is defined as $P \oplus Q = \{x \in \mathbb{R}^n | x = p + q \text{ for some } p \in P \text{ and } q \in Q\}$. The Pontryagin difference of $P$ and $Q$ is defined as $P \ominus Q = \{x \in \mathbb{R}^n | x + q \in P \forall q \in Q\}$.

A group $(G, \circ)$ is a set $G$ along with a binary operator $\circ$ such that the operator is associative, the set $G$ is closed under the operation $\circ$, the set includes an identity element, and the inverse of each element is included in the set. In matrix groups the operator $\circ$ is matrix multiplication and the identity element is the identity matrix $I$. In permutation groups the operator $\circ$ is function composition and the identity element is the identity permutation $e$. A group that contains only the identity element is called trivial. For notational simplicity we will drop the $\circ$ and write $gh$ for $g \circ h$. Two groups $G$ and $H$ are isomorphic if there exists a bijective function $f : G \to H$ such that $f(g_1 g_2) = f(g_1) f(g_2) \in H$ for all $g_1, g_2 \in G$.

For a set $S$ of real numbers, we denote by $\min S$ the minimal element of $S$, if it exists, $\max S$ the maximal element of $S$, if it exists, and $\max S$ the maximal element of $S$, if it exists.

Let $g(x) : \mathbb{R}^m \to \mathbb{R}^n$ and let $c \in \mathbb{R}^n$. Then by $g(x) = c$ we mean a system of $n$ equations such that the $i$th element of $g(x)$ is equal to the $i$th element of $c$ for $i = 1, \ldots, n$. By $g(x) \leq c$ we mean a system of $n$ inequalities such that the $i$th element of $g(x)$ is less than or equal to the $i$th element of $c$ for $i = 1, \ldots, n$. Let $d \in \mathbb{R}$. Then by $g(x) = d$ we mean a system of $n$ equations such that the $i$th element of $g(x)$ is equal to $d$ for $i = 1, \ldots, n$. By $g(x) \leq d$ we mean a system of $n$ inequalities such that the $i$th element of $g(x)$ is less than or equal to $d$ for $i = 1, \ldots, n$. 

1.2 Networked Energy Systems

In recent years, predictive control for energy systems have gained wide popularity and interest in both academic and industry research. In this section, we detail the concept of networked energy systems and the relevance of predictive control.

A networked energy system consists of a set of nodes which store energy. The energy states of the nodes are the states of the system. The nodes may pass energy between each other autonomously, such as through thermal conduction. There are also inputs to the nodes which can inject or remove energy from the nodes. In Figure 1.1, a diagram of a networked energy system is shown. This particular system consists of $n$ nodes with the $i$th node having energy $x_i$. The system in Figure 1.1 is connected in a ring network so that adjacent nodes pass energy autonomously to each other due to physical phenomena. Each node $i$ also has an input $u_i$ which allows for the injection or removal of energy from that node. Examples of networked energy systems include building HVAC system, battery networks, or the renewable energy grid.

Figure 1.1: Networked nodes
Predictive control is particularly effective for networked energy systems because of three main reasons which we discuss below.

1. Predictive control takes into account time-domain constraints of the system explicitly in the controller design. Networked energy systems are often subjected to hard constraints on the energy states of the nodes and flow rates of the inputs. A properly designed predictive controller can guarantee persistent satisfaction of these hard constraints for all future times. This is not true of most other control design methodologies, where constraint satisfaction is often solved by meticulous tuning of the control parameters, which is not tractable if one wishes to apply the technology to a wide variety of systems.

2. Predictive control chooses the control action by minimizing a time-domain user-defined cost function. This is especially relevant to the control of networked energy systems since it is usually desirable to choose actions which minimize forecasted energy prices.

3. Predictive control allows for the incorporation of real-time predictions of disturbances. The use of real-time predictions allows the controller to pick actions which are less conservative than controls methodologies which robustifies against all possible disturbance realizations. This is unique to predictive controller because it performs optimizations in real-time.

In this thesis, we consider networked energy systems modeled in discrete-time so that the system dynamics can be written as

\[ x_{t+1} = f(x_t, u_t, d_t), \]  

where \( x_t \in \mathbb{R}^n \) is the system state, \( u_t \in \mathbb{R}^m \) is the controlled input, and \( d_t \in \mathbb{R}^n \) is the uncontrolled disturbance at time \( t \). We assume that \( d_t \in D \), where \( D \) is a polytope. Now we proceed to the discussion of particular examples of networked energy systems which will be the focus of this thesis. We will describe the systems in detail and the role predictive control plays in achieving the control objectives.

**Building heating, ventilation, and air-conditioning (HVAC) systems**

Efficient control of building HVAC systems is essential since commercial buildings consume approximately 40% of the energy in the United States [65]. In recent years, a large literature has been created surrounding predictive control for building systems. Predictive control has been shown to be very effective at saving energy while satisfying the comfort requirements of the occupants [55, 53, 56, 54, 63].

The energy nodes of building HVAC system are thermal zones, which are typically enclosed spaces which house tenants and/or thermally sensitive material. Thermal zones which are geographically adjacent to one another transfer energy through thermal conduction, convection, and/or radiation. Each thermal zone is typically served by an energy input which
provides the ability to change the temperature in that zone. In this section, we will describe
two types of building HVAC systems: forced air systems and radiant slab systems. For each
system, we will describe the physical system setup and also the simplified modeling used for
predictive control.

**Forced-air systems**

Forced-air HVAC systems are the de facto standard in most commercial buildings. Heated or
cooled air from a central air handler unit (AHU) is distributed to the individual zones, where
a variable air volume (VAV) box can reheat the air and limit the air flow with controllable
vanes. Next, we discuss the thermal modeling of the zones for a forced-air system.

Suppose there are \( n \) thermal zones in the building. We index these thermal zones with the
integer set \( \{1, \ldots, n\} \). For zone \( i \), we define the adjacency set, \( A(i) \), to be the set of indices
of the zones which are geographical neighbors of zone \( i \). The thermal zones are modeled
using energy conservation principles \([47]\). The table below summarizes the variables used
in the model equation. The discrete-time dynamics of the system can be derived using

\[
\begin{align*}
\dot{u}_m,i,t & \text{ the mass flow rate of air into zone } i \text{ at time } t \\
u_{s,i,t} & \text{ the supply air temperature to zone } i \text{ at time } t \\
x_{z,i,t} & \text{ the air temperature inside zone } i \text{ at time } t \\
d_{oa,t} & \text{ the outside air temperature at time } t \\
d_{i,t} & \text{ external disturbance heat rate to zone } i \text{ at time } t
\end{align*}
\]

Table 1.1: Model variables

energy-conservation principles and is written compactly as

\[
x_{z,i,t+1} = f(x_{z,i,t}, u_{m,i,t}, u_{s,i,t}, d_{oa,t}, d_{i,t}, \{x_{z,j,t} | j \in A(i)\}). \tag{1.2}
\]

The function \( f \) is usually a bilinear function because of product terms between the mass
flow rate and room and supply air temperatures. However, one can also approximate the
dynamics as linear or piecewise linear by applying linearization techniques around operating
points of the system.

**Radiant slab systems**

Radiant slab systems are distinct from forced-air systems in that the main mode of heat
transfer to the thermal zones is through radiation from thermal slabs. In a typical radiant
slab setup, cement slabs are either mounted in the ceiling or floor of a thermal zone. Water
pipes are laid inside the cement slabs and heated or cooled water is passed through the
pipes to affect the temperature of the slabs. The thermal zones transfer energy with the slab
primarily through radiation and convection. The advantage of radiant systems over forced-air
systems is that water is much more efficient at transferring energy than air, with a potential to
decrease energy usage by 30% across North America [62]. The disadvantages are that radiant systems have much longer time constants than forced-air systems, sometimes on the order of hours, and that cooled slabs are susceptible to condensation in humid climates. These disadvantages make radiant slab systems prime candidates for predictive control since the slow response time can be offset by anticipation and the ability to handle slab temperature constraints will help mitigate condensation effects.

We now proceed to the description of the physical modeling of a radiant slab system. We propose a second-order model with the following two states: zone air temperature, \( x_{z,i,t} \) and the temperature of the slab, \( x_{slab,i,t} \). Other masses in the room, such as walls, are assumed to have temperatures close to the room air temperature [28]. We first define the following parameters:

<table>
<thead>
<tr>
<th>Symbol ( u_{\dot{m},i,t} )</th>
<th>Description: the mass flow rate of air into zone ( i ) at time ( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u_{s,i,t} )</td>
<td>the supply air temperature to zone ( i ) at time ( t )</td>
</tr>
<tr>
<td>( x_{z,i,t} )</td>
<td>the air temperature inside zone ( i ) at time ( t )</td>
</tr>
<tr>
<td>( d_{oa,t} )</td>
<td>the outside air temperature at time ( t )</td>
</tr>
<tr>
<td>( d_{i,t} )</td>
<td>external disturbance heat rate to zone ( i ) at time ( t )</td>
</tr>
<tr>
<td>( x_{slab,i,t} )</td>
<td>the slab temperature inside zone ( i ) at time ( t )</td>
</tr>
<tr>
<td>( x_{lm,i,t} )</td>
<td>the log-mean temperature of the water pipe in zone ( i )</td>
</tr>
</tbody>
</table>

Using basic energy balance concepts, we can derive the zone air discrete-time state equation

\[
x_{z,i,t+1} = f(x_{z,i,t}, x_{slab,i,t}, \{x_{z,j,t} | j \in A(i)\}, u_{\dot{m},i,t}, u_{s,i,t}, d_{oa,t}, d_{i,t}). \tag{1.3}
\]

Likewise using energy balance concepts, we can derive the radiant slab state equation [47]

\[
x_{slab,i,t+1} = f(x_{z,i,t}, x_{slab,i,t}, d_{i,t}, x_{lm,i,t}). \tag{1.4}
\]

**Networked Storage Systems**

Networked storage systems are comprised of a set of storage nodes which primary function is to store and transfer energy. The nodes typically do not interact autonomously with each other. These systems typically have input channels which allow for the controlled movement of energy between nodes or from a node directly. The system dynamics can be written as

\[
x_{t+1} = x_t + Bu_t + d_t, \tag{1.5}
\]

where \( B \in \mathbb{R}^{n \times m} \) is a matrix describing the input topology of the network. Next, we discuss a particular type of networked storage system, the networked battery system, which we will focus on in this thesis.
Networked battery system

Networked battery systems are of particular interest to applications such as renewable energy storage and electric vehicles. In a typical battery system with an active balancer, the batteries are connected by channels with capacitors which allow electric charge to be moved between the batteries [24]. Power draw and input to the batteries are typically modeled as disturbances, $d_t$, since they depend on an uncontrolled demand which can be uncertain in nature.

In a typical battery system, the cells are charged and discharged in unison. For lithium batteries, this presents a problem because overcharging or overdischarging the batteries can physically harm them [52]. Therefore, if the batteries are not properly balanced, the battery system can only be charged until one cell becomes full and discharged until one cell becomes empty. This causes the battery to be underutilized because many cells are not fully charged or discharged.

The solution to is to implement a battery balancer. Battery balancing can be divided into two categories: passive balancers and active balancers. Passive balancers balance the batteries by dissipating power from batteries with the highest states of charge, a process which can waste valuable energy. An active balancer uses power electronics to move charge between the batteries to achieve balance. This thesis will discuss the use of predictive controllers with active balancers [29], particularly the use of system symmetries in battery systems to enable fast embedded implementation of explicit controllers.

1.3 Optimization

Before we formally introduce model predictive control, we first provide a brief introduction to mathematical optimization. Let $g(x) : \mathbb{R}^m \rightarrow \mathbb{R}^n$, $h(x) : \mathbb{R}^m \rightarrow \mathbb{R}^p$, and $f(x) : \mathbb{R}^m \rightarrow \mathbb{R}$. Let $S \subset \mathbb{R}$ be defined as $S = \{ f(x) : x \in \mathbb{R}^m, g(x) \leq 0, h(x) = 0 \}$. Suppose $S$ has a minimal element. By convention, one writes the minimal element of $S$, $\min S$, in the compact form

$$ \min_x f(x) $$

subject to $g(x) \leq 0$

$ h(x) = 0.$

The problem of finding the minimal element of $S$ is called an optimization problem. In general such problems are intractable. Fortunately, most real world optimization problems allow for special assumptions on $g(x)$, $h(x)$, and $f(x)$ so that efficient algorithms exist for their solutions. Complete discussions of optimization algorithms can be found in Boyd and Vandenberghe [21], Bertsekas [10], and Fletcher [39].

In this thesis we are primarily concerned with optimization problems where both $g(x)$ and $h(x)$ are affine functions. We also deal primarily with cost functions $f(x)$ which are affine or quadratic. These types of optimization problems are called linear programs and
quadratic programs, respectively. For a discussion on efficient algorithms for linear and quadratic programs, the reader is referred to Bertsimas and Tsitsiklis [13] and Bertsekas [10].

1.4 Model predictive control

Model predictive or receding horizon control is a time-domain optimization-based control technique which uses predictive models of a system to forecast its behavior and minimize a performance cost subject to system constraints [18].

Consider a fixed control horizon length $N \in \mathbb{Z}^+$. We define

\[
\begin{bmatrix}
  u_t \\
  u_{t+1} \\
  \vdots \\
  u_{t+N-1}
\end{bmatrix}
, \quad
\begin{bmatrix}
  d_t \\
  d_{t+1} \\
  \vdots \\
  d_{t+N-1}
\end{bmatrix}
, \quad \text{and} \quad
\begin{bmatrix}
  x_t \\
  x_{t+1} \\
  \vdots \\
  x_{t+N}
\end{bmatrix}
, \quad (1.6)
\]

We also define

\[
\mathcal{U} = \mathcal{U} \times \cdots \times \mathcal{U} \quad \text{\(N\) times}, \quad
\mathcal{D} = \mathcal{D} \times \cdots \times \mathcal{D} \quad \text{\(N\) times}, \quad \text{and} \quad
\mathcal{X} = \mathcal{X} \times \cdots \times \mathcal{X} \quad \text{\(N\) times}, \quad (1.7)
\]

where $\mathcal{X}$ and $\mathcal{U}$ are polytopes representing constraints for the states and inputs, respectively. Let $g(x_t, u_t, d_t) : \mathbb{R}^{n(N+1)} \times \mathbb{R}^{mN} \times \mathbb{R}^{nN} \to \mathbb{R}$ be a function describing the cost we wish to minimize, such as energy usage or tracking error.

For a given choice of $x_t$ and $u_t$, the cost $g(x_t, u_t, d_t)$ is a set of real numbers if $d_t \in \mathcal{D}$ is uncertain. We define the set $G_{\mathcal{D}}(x_t, u_t) = \{g(x_t, u_t, d_t) | x_{k+1} = f(x_k, u_k, d_k), d_k \in \mathcal{D} \forall k = t, \ldots, t + N - 1\}$. In order to find a $u_t$ which is optimal in some sense, we must first designate a set to number mapping $h : 2^\mathbb{R} \to \mathbb{R}$ which maps $G_{\mathcal{D}}(x_t, u_t)$ to a real number. We discuss some choices for $h$ below.

1. If $d_t$ is only known to reside in the compact polytope $\mathcal{D}$ and $g$ is continuous on $\mathcal{D}$, then a possible choice for $h$ is

\[
h(G_{\mathcal{D}}(x_t, u_t)) = \max_{d_t \in \mathcal{D}} g(x_t, u_t, d_t)
\]

subject to $d_t \in \mathcal{D}$

\[
x_{k+1} = f(x_k, u_k, d_k) \text{ for all } k = t, \ldots, t + N - 1.
\]

Intuitively, $h$ describes an intelligent adversary which chooses the action which most disadvantages the player [69].

2. If $d_t$ is only known to reside in the polytope $\mathcal{D}$, then another simple approach is to simply choose one representative realization of the disturbance from the set. In this case,

\[
h(G_{\mathcal{D}}(x_t, u_t)) = g(x_t, u_t, \hat{d}_t), \quad (1.9)
\]
where \( \hat{d}_t \in \mathcal{D} \). Without further information, \( \hat{d}_t \) is often chosen as the centroid of \( \mathcal{D} \) \cite{70}. If \( d_t \) is a random variable with associated probability distribution with well-defined first moment, then it is common to set \( \hat{d}_t = E(d_t) \) \cite{20}.

3. If \( d_t \) is known to be a random variable and \( g(x_t, u_t, \hat{d}_t) \) and \( f \) are measurable functions then a common choice for \( h \) is

\[
h(G_D(x_t, u_t)) = E(g(x_t, u_t, \hat{d}_t)),
\]

where the system dynamics (1.1) are respected within the expectation \cite{9, 74, 55, 12, 44, 73}.

In a model predictive controller, at each time \( t \), the following finite-time optimization problem is solved

\[
\begin{align*}
\min_{\pi_{t+k}(x_t, \ldots, x_{t+k})} & \quad h(G_D(x_t, u_t)) \\
\text{subject to} & \quad x_t \in \mathcal{X} \forall d_t \in \mathcal{D} \\
& \quad x_{t+N} \in \mathcal{X}_f \\
& \quad x_t = \hat{x}_t,
\end{align*}
\]

where \( x_t \) respect the system dynamics (1.1), \( \mathcal{X}_f \) is a polytopic terminal constraint, \( \hat{x}_t \) is the measured system state, \( \pi_{t+k} : \mathbb{R}^{n(k+1)} \rightarrow \mathcal{U} \) for \( k = 0, \ldots, N - 1 \) are input feedback policies, and \( u_{t+k} = \pi_{t+k}(x_t, \ldots, x_{t+k}) \). The first input \( \pi_t(\hat{x}_t) \) is applied to the system and the process is repeated at time \( t + 1 \).

### 1.5 Summary of contributions and relevant publications

In this section we present a summary of the chapters of this thesis. The summary outlines our contributions to model predictive control for networked energy systems.

When systems are subjected to random disturbances, it is often desired to minimize an expected cost over feedback policies. However, the resulting optimization problem is in general intractable. There are many approximation techniques available to render the problem tractable for computation. One particular approximation method, called certainty equivalence, replaces the expected cost by a nominal cost with the expected disturbance profile. In chapter 2, we study the optimality of the certainty equivalence approximation in robust finite-horizon optimization problems with expected cost. We focus on problems with quadratic and piecewise quadratic expected cost, with an emphasis on linear soft-constrained problems. We provide an algorithm for determining the subset of the state-space for which the certainty equivalence technique is optimal. In the second part of the chapter we show how patterns in the problem structure called symmetries can be used to reduce the computational complexity of the proposed algorithm. Finally we demonstrate our technique
through numerical examples, including a networked battery system and radiant slab system. The relevant publication for this chapter is listed below.


When systems are subjected to adversarial disturbances, it is customary to minimize a cost which is maximized over the disturbances. The solution typically requires the enumeration of the vertices of the disturbance set, which results in a combinatorial explosion in the problem complexity as a function of the horizon length. In chapter 3, we consider these finite time min-max optimization problems for linear systems with additive disturbance subject to input constraints and soft state constraints, as is typical in energy systems. We present a set of sufficient conditions which allow for the computation of the feedback policy without using dynamic programming and yet avoiding vertex enumeration. The set of initial states for which the proposed conditions are satisfied is computed by solving a sequence of linear programs. We compare the proposed method to existing techniques in terms of computational complexity and solution cost. We then demonstrate the efficacy of the approach in an implementation of model predictive control for radiant-slab cooling systems. The relevant publication for this chapter is listed below.


When a large-scale system is too big to be optimized with the given computational equipment, it may be necessary to reduce the problem size. One technique to reduce the problem complexity is to apply model reduction to the system dynamics. In chapter 4, we study a model reduction technique for sparse networked energy systems. Our method differs from standard model reduction techniques in that it aims to preserve the sparsity of the system and also preserves the total energy of the system. We apply our technique to constrained and soft-constrained linear optimal control problems. We show through numerical examples that our method is comparable to standard model reduction techniques in terms of sub-optimality but have faster solution times because of its ability to preserve sparsity. The relevant publication for this chapter is listed below.


When a system must be controlled with an embedded platform at high sampling rates, real time optimization may no longer be a viable solution. In this case, multi-parametric optimization techniques may be applied to solve the optimization problem explicitly offline and a lookup table implemented in the embedded platform to retrieve the solutions. However, the technique explodes in computation time and memory usage with respect to the problem
size. This problem can be alleviated if the problem has inherent symmetries, allowing us to avoid redundant computation for regions of the state-space related by the symmetries. However, truly symmetric systems rarely exist, making the application of symmetric explicit controllers difficult. In chapter 5, we study the explicit model predictive control (MPC) design for quasi-symmetric systems. The quasi-symmetric system is modeled as a symmetric one plus an additive disturbance residing in a symmetric set. The explicit MPC controller is computed using symmetry to minimize the number of critical regions stored in memory while being robust to model mismatch and disturbances. We show through numerical examples that approximating quasi-symmetric systems as symmetric ones can drastically reduce memory usage and computation time for explicit controllers while still guaranteeing feasibility and stability. The relevant publication for this chapter is listed below.


In chapter 6, we present a case study of certainty equivalent predictive control of radiant slab systems. The chapter describes the creation of a simplified dynamic model of radiant slab system for implementation in real-time model predictive controller and a simulation study of the controller in closed-loop with an EnergyPlus model of the test building, the David Brower Center. The MPC is compared with the existing rule-based control method for a cooling season in a dry and hot climate. The results indicated that the MPC controller was able to maintain zone operative temperatures at EN 15251 Category II level more than 95% of the occupied hours for all zones, while with the rule-based method, only the core zone was maintained at this thermal comfort level. Compared to the rule-based method, MPC reduced the cooling tower energy consumption by 55% and pumping power consumption by 25%. The relevant publication for this chapter is listed below.

Chapter 2

Certainty equivalent control

2.1 Introduction

In this chapter we examine robust predictive control for systems with bounded stochastic disturbances. In particular, we examine the case when the finite-horizon optimization is over the expected cost

\[ h(G_D(x_t, u_t)) = E(g(x_t, u_t, \hat{d}_t)), \]  

(2.1)

where it is assumed that \( d_t \) is a random variable, \( g(x_t, u_t, \hat{d}_t) \) and \( f \) are measurable functions, and the system dynamics (1.1) are respected within the expectation \([9, 74, 55, 12, 44, 73]\). Optimization problems with expected cost and robust constraints have been widely studied in literature. In general, finding the exact optimal feedback control law to such problems is computationally intractable. However there are several approaches for approximating an optimal feedback law. For simple problems, such as unconstrained linear quadratic control, the exact optimal solution to the expected value problem can be computed via dynamic programming \([9]\). For more complex problems, tractable alternatives to computing exact feedback solutions to the expected-value problem are available, including Monte Carlo sampling, affine disturbance feedback, open-loop input sequences, and certainty equivalence. For general distributions and costs, the problem is often solved approximately using Monte Carlo sampling \([20]\). The effect of finite sampling with respect to the original expected value problem was investigated by Wang and Ahmed \([74]\). For certain distributions, such as Gaussian, affine feedback can be used to approximate the feedback solution and propagate the distribution forward. Goulart, Kerrigan, and Maciejowski \([42]\) detail the use of affine disturbance feedback in the robust control of linear systems with additive disturbance. The solution of the expected value problem using affine feedback subject to probabilistic constraints was addressed by Ma \([55]\) in the context of chance-constrained stochastic MPC. While affine disturbance feedback is computationally efficient, it is conservative because, in general, the optimal feedback policies are non-linear. Bertsimas, Iancu, and Parrilo \([12]\) have proven the optimality of affine disturbance feedback for a specific class of 1-D problems. Meanwhile, Hadjiyiannis, Goulart, and Kuhn \([44]\) and more recently Van Parys, Goulart, and Morari
[73] have characterized the suboptimality of affine disturbance feedback in expected value problems. Alternatively, open-loop input sequences can be used which generally lead to even more conservative solutions. The advantage is the faster computation time over affine feedback.

One popular approach for solving expected value control problems is certainty equivalence in which the stochastic disturbance is replaced in the cost by its expected value.

\[
h(G_D(x_t, u_t)) = g(x_t, u_t, \hat{d}_t),
\]

where \(\hat{d}_t \in \mathcal{D}\). We assume \(d_t\) is a random variable with associated probability distribution with well-defined first moment and that \(\hat{d}_t = E(d_t)\). While based on a potentially bad approximation, certainty equivalence often performs very well when applied to problems in economics [20]. For instance, it is well documented in the literature that the optimal unconstrained finite-horizon linear quadratic expected value controller is equivalent to the finite-horizon LQR controller for disturbances with zero mean. However, in general, the certainty equivalence controller is not optimal for constrained control problems.

This chapter studies the optimality of the certainty equivalence approximation in robust finite-horizon optimization problems with expected cost. We consider finite-time expected value optimal control of linear systems with additive stochastic disturbance subject to robust constraints. The cost is separable in time so that dynamic programming can be applied. We provide an algorithm based on dynamic programming for calculating a region of the state-space in which the certainty equivalence controller is optimal. In the second part of the section, we investigate how symmetry of the model predictive problem can be exploited to decrease computation time and memory usage of the explicit certainty equivalence controller.

Computing the subset of the state-space in which certainty equivalence is optimal can be computationally taxing. Therefore, in the second part of the section, we propose a modification of our algorithm that exploits symmetric structure in the system to reduce computational complexity. Symmetry has been used extensively in numerous fields to reduce computational complexity. In recent years symmetry has been applied to optimization to solve linear-programs [15], semi-definite programs [40], and integer-programs [16]. In [36] and [45] symmetry was studied in control theory to decompose large-scale systems into invariant subsystems. In [27] the authors exploited symmetry to reduce the computational complexity of \(H_2\) and \(H_{\infty}\) controllers. In [31] the authors studied symmetry in linear model predictive control. This section extends these results to dynamic programming to solve the expected value problem with robust constraints.

In the numerical examples section, we apply our technique to a simple integrator system, battery network system, and building HVAC system. For these systems, we identify regions of the state-space for which certainty equivalence provides the optimal feedback solution. Where applicable, we also demonstrate the use of system symmetry to reduce computation time and memory usage.
2.2 Problem Definition

Consider the linear time-invariant, discrete-time system with additive disturbance

\[ x_{t+1} = Ax_t + Bu_t + d_t, \quad t \geq 0 \quad (2.3) \]

where \( x_t \in \mathbb{R}^n \) is the system state, \( u_t \in \mathbb{R}^p \) the controlled input, \( d_t \in \mathbb{R}^n \) the disturbance, \( A \in \mathbb{R}^{n \times n} \), and \( B \in \mathbb{R}^{n \times p} \). The system is subject to constraints on the states and inputs

\[ x_t \in X_t \text{ and } u_t \in U, \quad \forall t \geq 0, \quad (2.4) \]

where \( X_t \subset \mathbb{R}^n \) and \( U \subset \mathbb{R}^p \) are polytopes. The disturbances \( \{d_0, d_1, \ldots\} \) are random variables which are independently distributed. We assume that the disturbances have finite support

\[ d_t \in D_t, \quad \forall t \geq 0, \]

where \( D_t \subset \mathbb{R}^n \) is a polytope. Note that the disturbances are not required to have zero mean, thus our method can be extended to affine systems by simply including the affine term with the disturbance.

We consider the following cost function

\[ J(x, u) = E \left( f_N(x_N, u_N) + \sum_{t=0}^{N-1} f_t(x_t, u_t) \right), \quad (2.5) \]

where \( N \) is a fixed horizon length and the stage costs \( f_t : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R} \) and terminal cost \( f_N : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R} \) are jointly convex in \( x_t \) and \( u_t \) for all \( 0 \leq t \leq N \). We are interested in finding the feedback control policies \( \pi_t \) which minimize the cost (2.5) subject to constraints,

\[ J^*_0(x_0) = \min_{\pi_0, \ldots, \pi_{N-1}} E \left( f_N(x_N, u_N) + \sum_{t=0}^{N-1} f_t(x_t, u_t) \right) \quad (2.6) \]

subject to \( x_{t+1} = Ax_t + B\pi_t(x_t) + d_t \)

\[ x_t \in X_t, \quad \forall d_t \in D_t, \quad \forall t \geq 0 \]

where the control input is \( u_t = \pi_t(x_t) \) and the control law \( \pi_t : X_t \to U \) maps the system state \( x_t \in X_t \) to a feasible input \( u_t \in U \) for \( t = 0, \ldots, N - 1 \).

Exact Controller using Dynamic Programming

Problem (2.6) can be solved using dynamic programming in the following sense. The terminal cost is defined as

\[ J^*_N(x_N) = f_N(x_N, u_N) \]
and for each time \( t = N - 1, \ldots, 0 \) we calculate the cost-to-go by solving the following optimization problem

\[
J^*_t(x_t) = \min_{\pi_t(x_t) \in U} J_t(x_t, u_t)
\]

subject to \( Ax_t + Bu_t + d_t \in X_{t+1}, \forall d_t \in D_t \) \hspace{1cm} (2.7)

for each \( x_t \in X_t \) where

\[
J_t(x_t, u_t) = f_t(x_t, u_t) + E_d(J^*_{t+1}(Ax_t + Bu_t + d_t)). \quad (2.8)
\]

For each time \( t \) the optimal control policy \( u_t = \pi^*_t(x_t) \) is the optimizer \( \pi^*_t : X_t \rightarrow U \) of problem (2.7). Note that in general \( E_d(J^*_{t+1}(Ax_t + Bu_t + d_t)) \) does not have a closed-form solution.

**Certainty Equivalence Controller**

One approach to obtain an approximation to the optimal controller \( \pi^*_t(x_t) \) is to use the certainty equivalence principle. We define the certainty equivalence controller to be the optimal controller to problem (2.6) with the expected value cost replaced by a nominal cost evaluated by replacing \( d_t \) with their expected values \( E(d_t) \). The certainty equivalence controller can be obtained using dynamic programming as follows. The terminal cost is defined as

\[
\tilde{J}_N^*(x_N) = f_N(x_N, u_N)
\]

and for each time \( t = N - 1, \ldots, 0 \) we calculate the cost-to-go by solving the following optimization problem

\[
\tilde{J}^*_t(x_t) = \min_{\tilde{\pi}_t(x_t) \in \tilde{U}} \tilde{J}_t(x_t, u_t)
\]

subject to \( Ax_t + Bu_t + d_t \in X_{t+1}, \forall d_t \in D_t \) \hspace{1cm} (2.9)

for \( x_t \in X_t \) where

\[
\tilde{J}_t(x_t) = f_t(x_t, u_t) + \tilde{J}^*_{t+1}(Ax_t + Bu_t + E(d_t)).
\]

For each time \( t \) the optimal control policy \( u_t = \tilde{\pi}^*_t(x_t) \) is the optimizer \( \tilde{\pi}^*_t : X_t \rightarrow U \) of problem (2.9).

The difference between the exact and certainty equivalence control problems is the cost minimized at each stage. The exact problem includes expected value of the cost-to-go \( E(J^*_{t+1}(Ax + Bu + d)) \). The certainty equivalence problem includes the cost-to-go \( \tilde{J}^*_{t+1}(Ax_t + Bu_t + E(d_t)) \) with the expected disturbance \( E(d_t) \). The certainty equivalence approximation yields the dynamic programming steps that are tractable for problems with quadratic cost and reasonable size.


2.3 Optimality of Certainty Equivalence

In this section we present an algorithm for determining the subset of the state-space \( \mathcal{X}_t \) in which the certainty equivalence controller is optimal. First we review the optimality of certainty equivalence for unconstrained linear quadratic control. We use this result to examine the optimality of certainty equivalence for constrained linear quadratic control with a one-step horizon. Using dynamic programming we establish conditions for the optimality of certainty equivalence for constrained problems with arbitrary horizons. We use this analysis to construct an algorithm that finds the region of the state-space where the certainty equivalence controller is optimal.

Certainty Equivalence for Unconstrained Linear Quadratic Control

In this section we examine the equivalence between the optimal and certainty equivalence control for unconstrained linear systems with quadratic cost. We consider problem (2.6) with no constraints on the states \( (\mathcal{X}_t = \mathbb{R}^n) \) and inputs \( (\mathcal{U}_t = \mathbb{R}^p) \) and quadratic terminal and stage costs

\[
\begin{align*}
    f_N(x_N) &= x_N^T Q_N x_N \\
    f_t(x_t, u_t) &= x_t^T Q_t x_t + u_t^T R_t u_t
\end{align*}
\]

where \( Q_t \succ 0 \) and \( R_t \succ 0 \) are positive definite and the horizon \( N \) is finite. If the disturbance \( d_t \) is zero-mean \( E(d_t) = 0 \) for all \( t \), then the certainty equivalence problem (2.9) reduces to the standard linear quadratic regulator problem. Thus the certainty equivalence controller is the linear state-feedback controller

\[
\pi_t^*(x_t) = - \left( B^T P_{t+1} B + R_t \right)^{-1} B^T P_{t+1} A x_t
\]

where \( P_{t+1} \) is the solution to the discrete Riccati equation

\[
P_{t-1} = A^T P_t A - A^T P_t B (B^T P_t B + R_t)^{-1} B^T P_t A + Q_{t-1}
\]

and \( P_N = Q_N \). Under these assumptions the certainty equivalence controller is also the optimal controller. We briefly explain why this is the case.

The optimal controller \( \pi_t^*(x_t) \) is obtained by solving the dynamic programming problem (2.7). Suppose the cost-to-go \( J_{t+1}^*(x_{t+1}) \) is a quadratic function \( J_{t+1}^*(x_{t+1}) = x_{t+1}^T Q x_{t+1} + q^T x_{t+1} + C \) of the state \( x_{t+1} \), as is the case for \( t = N \). Then by straightforward substitution we have

\[
J_{t+1}^*(Ax_t + Bu_t + d_t) = J_{t+1}^*(Ax_t + Bu_t) + 2 x_t^T A^T Q d_t + 2 u_t^T B^T Q d_t + q^T d_t + d_t^T Q d_t,
\]
where \( J_{t+1}^*(Ax_t + Bu_t) \) is the certainty equivalence cost for \( E(d_t) = 0 \). The expected value of this cost is

\[
E_{d_t}(J_{t+1}^*(Ax_t + Bu_t + d_t)) = J_{t+1}^*(Ax_t + Bu_t) + \text{trace}(E(d_t^T d_t)Q) + 2x_t^T A^TQE(d_t) + 2u_t^T B^TQE(d_t) + q^T E(d_t). \quad (2.12)
\]

If \( E(d_t) = 0 \) then the expression further simplifies to

\[
E_{d_t}(J_{t+1}^*(Ax_t + Bu_t + d_t)) = J_{t+1}^*(Ax_t + Bu_t) + \text{trace}(d_t^T d_t)Q.
\]

Note that the term \( \text{trace}(d_t^T d_t)Q \) is independent of the control input \( u_t \). When we solve the optimization problem (2.7) with the cost function

\[
J_t(x_t, u_t) = f_t(x_t, u_t) + J_{t+1}^*(Ax_t + Bu_t) + \text{trace}(d_t^T d_t)Q
\]

we obtain the linear quadratic regulator since the constant term \( \text{trace}(d_t^T d_t)Q \) does not change the optimizer \( \pi_t^*(x_t) \). Thus the optimal controller is the linear quadratic regulator \( \pi_t^*(x_t) = -Fx_t \) when the cost-to-go \( J_{t+1}^*(x_t) \) is a quadratic function of the state \( x_t \).

Finally we note that the new cost-to-go

\[
J_t^*(x_t) = f_t(x_t, -Kx_t) + J_{t+1}^*(Ax_t - BFx_t) + \text{trace}(d_t^T d_t)Q
\]

is also a quadratic function of the state \( x_t \). Thus certainty equivalence holds for \( t - 1 \).

Unfortunately this inductive argument does not hold for constrained \( \mathcal{X}_t \subset \mathbb{R}^n \) and \( \mathcal{U}_t \subset \mathbb{R}^p \) control problems. In this case, the optimal controller \( \pi_t^*(x_t) \) will generally be non-linear \( \pi_t^*(x_t) = \pi_t(x_t) \). Thus the updated cost-to-go \( J_t^*(x_t) \) will not be a quadratic function of the state \( x_t \). The exception is a one-step horizon \( N = 1 \) optimal control problem since the terminal cost-to-go \( J_N^* = f_N(x_N) \) is quadratic. This is summarized by the following proposition.

**Proposition 1.** Consider problem (2.6) with the horizon \( N = 1 \) and cost (2.10). Then the certainty equivalence controller is optimal \( \pi_0^*(x_0) = \tilde{\pi}_0^*(x_0) \) and the difference between the exact and certainty equivalence cost functions \( J_0^*(x_0) - \tilde{J}_0^*(x_0) \) is a constant.

**Proof.** Note that it is sufficient to show that \( E_{\tilde{d}_0}(J_1^*(Ax_0 + Bu_0 + d_0)) - J_1^*(Ax_0 + Bu_0 + E(d_0)) \) is a constant for all \( x_0 \in \mathbb{R}^n \) and \( u_0 \in \mathcal{U} \). From equation (2.12) it follows that \( E_{\tilde{d}_0}(J_1^*(Ax_0 + Bu_0 + d_0)) - J_1^*(Ax_0 + Bu_0 + E(d_0)) = \text{trace}(E(d_0^T d_0)Q) \), which proves the proposition.

In the rest of this section, we will determine the subset of the state-space where this result can be extended for horizon lengths larger than \( N = 1 \).
Determining where Certainty Equivalence is Optimal

In this section we determine the region of the state-space in which the certainty equivalence controller is optimal. First we review the relevant results for multiparametric quadratic programming [18].

Theorem 1. Consider the following multiparametric program

\[ J^*(x) = \text{minimize } J(z, x) \]  \hspace{1cm} (2.13a)
subject to \( z \in Z(x) \) \hspace{1cm} (2.13b)
\( x \in X \) \hspace{1cm} (2.13c)

where \( z \) are the decision variables, \( x \) are the parameters, \( J(z, x) = \frac{1}{2} z^T H z \) is a strictly-convex quadratic function, \( F(x) = \{ z : C z + G x \leq K \} \) is the feasible region, and \( X \) is a polytope.

Then

1. The optimizer \( z^* : X \rightarrow Z \) is a continuous piecewise affine on polyhedra function

\[ z^*(x) = \begin{cases} 
F_1 x + G_1 & \text{for } x \in \mathcal{R}_1 \\
\vdots \\
F_r x + G_r & \text{for } x \in \mathcal{R}_r 
\end{cases} \]

2. The value function \( J^*(x) \) is a convex piecewise quadratic on polyhedra function

\[ J^*(x) = \begin{cases} 
J^*_1(x) & \text{for } x \in \mathcal{R}_1 \\
\vdots \\
J^*_r(x) & \text{for } x \in \mathcal{R}_r 
\end{cases} \]

where \( J^*_i(x) \) are quadratic functions.

3. The closure of the critical regions \( \mathcal{R}_i \) are polyhedra. The critical region partition is denoted by \( \mathcal{R} = \{ \mathcal{R}_1, \ldots, \mathcal{R}_r \} \).

Let \( \mathcal{P}_{t+1} = \{ \mathcal{R}_{t+1}^1, \ldots, \mathcal{R}_{t+1}^r \} \) be a P-collection of critical regions \( \mathcal{R}_{t+1}^i \subseteq X_{t+1} \) where the certainty equivalence controller is optimal. In other words \( \pi_{t+1}^*(x) = \tilde{\pi}_{t+1}^*(x) \) and \( J_{t+1}^*(x) - J_{t+1}^*(x) \) is a constant for all \( x \in \mathcal{R}_{t+1}^i \) and \( \tilde{\mathcal{P}}_{t+1} \subseteq \mathcal{P}_{t+1} \). Note that at time \( t = N - 1 \), the certainty equivalence P-collection \( \mathcal{P}_{N-1} \) is simply the set of critical regions of \( J_{N-1}^*(x_{N-1}) \) by Proposition 1.

For some critical region \( \mathcal{R}_{t+1}^j \in \mathcal{P}_{t+1} \), consider the problem

\[ \hat{J}_t^*(x_t) = \min_{\pi_t(x_t) \in \tilde{U}} f_t(x_t, u_t) + E_{d_t}(J_{t+1}^*(Ax_t + Bu_t + d_t)) \]
subject to \( Ax_t + Bu_t + d_t \in \mathcal{R}_{t+1}^j \forall d_t \in \mathcal{D}_t \) \hspace{1cm} (2.14)
and its certainty equivalence approximation

\[
\tilde{J}_t^*(x_t) = \min_{\hat{\pi}(x_t) \in \mathcal{D}_t} f_t(x_t, u_t) + \tilde{J}_{t+1}^*(Ax_t + Bu_t + E(d_t))
\]

subject to \(Ax_t + Bu_t + d_t \in \mathcal{R}_{t+1}^j \forall d_t \in \mathcal{D}_t\)

(2.15)

where \(\tilde{J}_{t+1}^*(x_{t+1}) = J_{t+1}^*(x_{t+1}) + C\) for all \(x_{t+1} \in \mathcal{R}_{t+1}^j\) by definition of \(\mathcal{P}_{t+1}\). Since \(J_{t+1}^*(x)\) is quadratic in \(\mathcal{R}_{t+1}^j\), we conclude that \(\hat{\pi}_t^*(x_t) = \hat{\pi}_t^*(x_t)\) for all \(x_t\) in the domain of the optimizers by Proposition 1. Thus for states \(x_t \in \mathcal{X}_t\) where \(\hat{\pi}_t^*(x_t) = \pi_t^*(x_t)\), we have trivially that \(\hat{\pi}_t^*(x_t) = \pi_t^*(x_t)\). Next we will derive conditions for determining the subset of \(\mathcal{X}_t\) for which \(\hat{\pi}_t^*(x_t) = \pi_t^*(x_t)\) for \(x_t\) in the subset.

Note that \(\mathcal{R}_{t+1}^j \cap \mathcal{D}_t \subseteq \mathcal{X}_{t+1} \cap \mathcal{D}_t\) since \(\mathcal{R}_{t+1}^j \subseteq \mathcal{X}_{t+1}\). Thus we can write \(\mathcal{X}_{t+1} \cap \mathcal{D}_t = \{x \in \mathbb{R}^n | P_t^i x \leq P_t^c \text{ for } i = 1, \ldots, p\}\) and \(\mathcal{R}_{t+1}^j \cap \mathcal{D}_t = \{x \in \mathcal{X}_{t+1} \cap \mathcal{D}_t | R_t^j x \leq R_t^c \text{ for } i = 1, \ldots, r\}\) where the constraints \(P_t^i x \leq P_t^c\) and \(R_t^j x \leq R_t^c\) are non-redundant for \(\mathcal{X}_{t+1} \cap \mathcal{D}_t\) and \(\mathcal{R}_{t+1}^j \cap \mathcal{D}_t\) respectively. We augment the matrix \(R^x\) and vector \(R^c\) so that \(\mathcal{R}_{t+1}^j \cap \mathcal{D}_t = \{x \in \mathbb{R}^n | R_t^x x \leq R_t^c \text{ for } i = 1, \ldots, r, r+1, \ldots, r'\}\) is in minimal representation.

**Proposition 2.** \(\hat{\pi}_t^*(x_t) = \pi_t^*(x_t)\) for states \(x_t \in \mathcal{X}_t\) for which the constraints \(R_t^j x \leq R_t^c\) are inactive at optimum for all \(i \in \{1, \ldots, r\}\).

**Proof.** We first show that the cost-to-go at each time step \(J_t^*(x_t)\) is convex for every \(t \in \{0, \ldots, N\}\). We show this recursively. Observe that \(J_N^*(x_N)\) is convex because it is a quadratic cost. Suppose at time \(t, J_t^*(x_t)\) is convex. Since \(x_t = Ax_{t-1} + Bu_{t-1} + d_{t-1}\) is an affine map from \((x_{t-1}, u_{t-1})\) to \(x_t\) for fixed \(d_{t-1}\), the function \(J_t^*(Ax_{t-1} + Bu_{t-1} + d_{t-1})\) is jointly convex in \((x_{t-1}, u_{t-1})\) for fixed \(d_{t-1}\). It was shown in [26] that \(E_{d_{t-1}}(J_t^*(Ax_{t-1} + Bu_{t-1} + d_{t-1}))\) is a convex function in \((x_{t-1}, u_{t-1})\). Let \(\mathcal{U}_t(x) = \{u \in \mathcal{U} : Ax + Bu + d \in \mathcal{X}_{t+1} \forall d \in \mathcal{D}_t\}\). Next, we show that \(\mathcal{U}_t(x)\) is a convex point-to-set map. Let \(x_1, x_2\) be two initial states and \(u_1 \in \mathcal{U}_t(x_1)\) and \(u_2 \in \mathcal{U}_t(x_2)\). We must show that \(\lambda u_1 + (1 - \lambda)u_2 \in \mathcal{U}_t(\lambda x_1 + (1 - \lambda)x_2)\). For any \(d \in \mathcal{D}_t\), we have

\[
A(\lambda x_1 + (1 - \lambda)x_2) + B(\lambda u_1 + (1 - \lambda)u_2) + d = \lambda(Ax_1 + Bu_1 + d) + (1 - \lambda)(Ax_2 + Bu_2 + d).
\]

Since \(\mathcal{X}_{t+1}\) is convex, the above equation shows that \(\lambda x_1 + (1 - \lambda)x_2 \in \mathcal{X}_{t+1}\), which implies \(\lambda u_1 + (1 - \lambda)u_2 \in \mathcal{U}_t(\lambda x_1 + (1 - \lambda)x_2)\). Using the convexity of \(E_{d_{t-1}}(J_t^*(Ax_{t-1} + Bu_{t-1} + d))\) and \(\mathcal{U}_{t-1}(x)\), the authors of [38] showed that \(J_{t-1}^*(x_{t-1})\) is convex.

Let \((\hat{\pi}_t^*(x_t), \hat{\lambda}_t^*)\) be primal and dual optimizers, respectively, for problem (2.14). Let \(\lambda_t\) be the dual variable for problem (2.7). We assume without loss of generality that the \(i\)-th element of \(\hat{\lambda}_t\) corresponds to the \(i\)-th row of \(R^x\) and the \(i\)-th element of \(\lambda_t\) corresponds to the \(i\)-th row of \(P^x\). We will show that \(\hat{\pi}_t^*(x_t)\) is optimal for problem (2.7) by appropriately constructing \(\lambda_t\) and showing that the primal-dual pair, \((\hat{\pi}_t^*(x_t), \lambda_t)\), satisfies the KKT condition for problem (2.7). For each \(i \in \{1, \ldots, p\}\) such that \(\{x | P_t^i x \leq P_t^c\} = \{x | R_t^j x \leq R_t^c\}\) for some \(j \in \{r + 1, \ldots, r'\}\), let the \(i\)-th element in \(\lambda_t\) be set equal to the \(j\)-th element in \(\hat{\lambda}_t\), where \(j\) is such that \(P_t^i = R_t^j\). Set the remaining elements of \(\lambda_t\) equal to 0. Since \((\hat{\pi}_t^*(x_t), \hat{\lambda}_t)\)
satisfies the KKT conditions for problem 2.14, it follows that if the constraints \( R_i^x x \leq R_i^c \) are inactive for all \( i \in \{1, ..., r\} \), then \( \hat{\pi}_t^*(x_t), \lambda_t \) satisfy the KKT conditions for problem 2.7. Since problem (2.7) is convex, we conclude that \( \hat{\pi}_t^*(x_t) \) is an optimal solution for problem (2.7).

Using this proposition we can find the region where the certainty equivalence controller is optimal by solving the multi-parametric quadratic program (2.15) and testing the active constraints for the resulting critical regions. This is summarized in Algorithm 1.

Algorithm 1 maintains a P-collection \( \mathcal{P}_t \) of critical regions where the certainty equivalence solution is optimal. For each time \( t = N - 1, \ldots, 0 \), a multi-parametric quadratic program solver is used to solve problem (2.15) with a critical region \( R_{i+1}^j \in \mathcal{P}_{t+1} \). This produces an array of critical regions \( \mathcal{R} = \{R_1, \ldots, R_r\} \). For each critical region \( R_i \in \mathcal{R} \), Algorithm 1 uses the constraint test from Proposition 2 to determine if certainty equivalence holds inside \( R_i \). At termination this algorithm returns the P-collection \( \mathcal{P}_0 \subseteq 2^{\mathbb{R}_0} \) where the certainty equivalence control is the optimal solution to (2.6).

\[ x_{t+1} = x_t + u_t + d_t \] (2.16)

where \( x_t, u_t, d_t \in \mathbb{R}^2 \). Suppose for a horizon \( N = 3 \) we would like to solve the problem (2.6) with terminal and stage costs

\[ f_N(x_N) = x_N^Tx_N \]
\[ f_t(x_t, u_t) = x_t^Tu_t + u_t^Tu_t \]

**Numerical Example**

For our first example, we consider a 2-D discrete integrator system before building up to more complicated examples. The system dynamics is described by

For our first example, we consider a 2-D discrete integrator system before building up to more complicated examples. The system dynamics is described by

\[ x_{t+1} = x_t + u_t + d_t \] (2.16)

where \( x_t, u_t, d_t \in \mathbb{R}^2 \). Suppose for a horizon \( N = 3 \) we would like to solve the problem (2.6) with terminal and stage costs

\[ f_N(x_N) = x_N^Tx_N \]
\[ f_t(x_t, u_t) = x_t^Tu_t + u_t^Tu_t \]
and constraints $x_t \in [-10, 10]^2$, $u_t \in [-1, 1]^2$, and $d_t \in [-\beta, \beta]^2$, where $\beta \in \mathbb{R}^+$ is a parameter.

Using the method described in section 2.3, we compute the set of states at each time step for which certainty equivalence is optimal. At time step 0, the set of states for which the certainty equivalence approximation is exact is plotted below in Figure 2.1 for $\beta = 0.2, 0.5,$ and 0.8, respectively.

![Figure 2.1](image_url)

Figure 2.1: (a) Set $\mathcal{P}_0$ of initial states $x_0 \in \mathcal{X}_0$ for which the certainty equivalence controller $\tilde{\pi}_0^*$ is optimal $\pi_0^*(x_0) = \tilde{\pi}_0^*(x_0)$ for $\beta = 0.2$. (b) $\beta = 0.5$ (c) $\beta = 0.8$

The results show that smaller disturbance sets yield a larger region where the certainty equivalence controller is optimal.

For our second example, we consider a 2-D system with system dynamics described by equation (2.16) and $A$ is a diagonal matrix with entries $\alpha \in \mathbb{R}$ along the diagonal. Suppose
for a horizon \( N = 3 \) we would like to solve the problem (2.6) with terminal and stage costs

\[
\begin{align*}
    f_N(x_N) &= x_N^T x_N \\
    f_t(x_t, u_t) &= x_t^T x_t + u_t^T u_t
\end{align*}
\]

and constraints \( x_t \in [-10, 10]^2 \), \( u_t \in [-1, 1]^2 \), and \( d_t \in [-0.5, 0.5]^2 \).

Using the method described in section 2.3, we compute the set of states at each time step for which certainty equivalence is optimal. At time step 0, the set of states for which the certainty equivalence approximation is exact is plotted below in Figure 2.2 for \( \alpha = 0.4, 0.6, \) and 0.8, respectively.

![Figure 2.2](image)

Figure 2.2: (a) Set \( \mathcal{P}_0 \) of initial states \( x_0 \in \mathcal{X}_0 \) for which the certainty equivalence controller \( \tilde{\pi}_0^* \) is optimal \( \tilde{\pi}_0^*(x_0) = \tilde{\pi}_0^*(x_0) \) for \( \alpha = 0.4 \). (b) \( \alpha = 0.6 \) (c) \( \alpha = 0.8 \)

The results show that when the dynamics drive the states quickly towards the origin (\( \alpha \) is small), more regions of the state space are optimal for the certainty equivalence controller.
CHAPTER 2. CERTAINTY EQUIVALENT CONTROL

This is because a small $\alpha$ drives the system robustly into a critical region centered at the origin without extra inputs.

Certainty Equivalence for Piecewise Quadratic Problems

Algorithm 1 can be extended to find the region of the state-space where the certainty equivalence controller is optimal for piecewise quadratic cost functions. We consider problem (2.6) with a fixed horizon $N$ and polytopic piece-wise quadratic (PPWQ) cost

$$f_N(x_N) = x_N^T Q_{N,i} x_N \text{ if } x_N \in \mathcal{X}_{N,i}$$

(2.17a)

$$f_t(x_t, u_t) = x_t^T Q_{t,i} x_t \text{ if } x_t \in \mathcal{X}_{t,i}$$

(2.17b)

where $\{\mathcal{X}_{t,i}\}$ is a P-collection such that $\mathcal{X}_t = \bigcup \mathcal{X}_{t,i}$.

We modify Algorithm 1 to tailor it for PPWQ cost, which is reported as Algorithm 2. Algorithm 2 is very similar to Algorithm 1. The main difference is that at time $t$ a MPQP must be solved for each $\mathcal{X}_{t,i}$.

**Algorithm 2 Computing Certainty Equivalence State Subset**

1: for $t = N$ to 0 do
2:   for each $\mathcal{R}_i^t \in \mathcal{P}_t$ do
3:     for each $\mathcal{X}_{t-1,i}$ do
4:       Solve (2.15) at time $t - 1$. Obtain critical region array $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_j\}$
5:     for each $\mathcal{R}_i \in \mathcal{R}$ do
6:       if the constraints $R_i^t x \leq R_i^c$ are inactive for region $\mathcal{R}_i$ then
7:         Add critical regions $\mathcal{R}_i$ to certainty equivalence partition $\mathcal{P}_{t-1}$
8:       end if
9:     end for
10:   end for
11: end for
12: end for

The results above can be extended to problems with expected linear cost. The only difference is that the piecewise quadratic cost are replaced by piecewise linear cost.

Controller Implementation

From Algorithms 1 and 2 we can construct a subset $\bigcup_{\mathcal{R}_i \in \mathcal{P}_0} \mathcal{R}_i \subseteq \mathcal{X}$ of the state-space $\mathcal{X}$ for which the certainty equivalence approximation is optimal. The set $\bigcup_{\mathcal{R}_i \in \mathcal{P}_0} \mathcal{R}_i \subseteq \mathcal{X}$ is called the underlying set of the set array $\mathcal{P}_0$. In addition, the algorithm can also keep track of the optimal affine controllers in each critical region. The controller can then be implemented directly in a receding horizon controller.
The other alternative is to store only the P-collection of critical regions. Whenever the measured state \( x_0 \in P_0 \) is inside the certainty equivalence region \( P_0 \), one can solve the following problem to retrieve the optimal control.

\[
\min_{K_t, c_t} f_N(\bar{x}_N, \bar{u}_N) + \sum_{t=0}^{N} f_t(\bar{x}_t, \bar{u}_t)
\]

subject to \( x_t \in X_t \)

\[
u_t = K_t x_t + c_t \in U, \forall d_t \in D_t \forall t \geq 0,
\]

where \( \bar{x}_{t+1} = A\bar{x}_t + Bu_t + E(d_t), \bar{x}_0 = x_0, \) and \( \bar{u}_t = K_t \bar{x}_t + c_t \). The authors in [42] have detailed a method to solve the above problem using affine disturbance feedback, which transforms the problem into a tractable convex problem.

In the case that \( x_0 \) is not in \( P_0 \), one approach is to continue using the affine disturbance feedback controller. The authors in [44] and [73] detail the implementation of affine controllers in expected value problems and also methods to compute the suboptimality of such controllers.

### 2.4 Certainty Equivalence for Problems with Symmetries

In this section we present a method for reducing the computational complexity of Algorithms 1 and 2. Our modification exploits patterns in the problem structure called symmetries.

#### Symmetry of the Optimal Control Problem

In this section we define symmetry for problem (2.6) and show how symmetry affects the exact and certainty equivalence controllers. A symmetry of problem (2.6) is a state-space transformation \( \Theta \) and input-space transformation \( \Omega \) that preserves the dynamics, constraints, and stage cost. Let \( x_{c,t} \) and \( u_{c,t} \) be the centroids of \( X_t \) and \( U_t \), respectively. Also, let \( d_t \) be distributed according to the density function \( p(d) \).

**Definition 1.** An affine symmetry of Problem (2.6) is a pair of invertible matrices \( \Theta \in \mathbb{R}^{n \times n} \) and \( \Omega \in \mathbb{R}^{p \times p} \) such that for each \( t = 0, \ldots, N \)

\[
\Theta A = A\Theta \\
\Theta B = B\Omega \\
\Theta(X_t - x_{c,t}) = X_t - x_{c,t} \\
\Omega(U_t - u_{c,t}) = U_t - u_{c,t}
\]

\[
f_t(\Theta(x - x_{c,t}) + x_{c,t}, \Omega(u - u_{c,t}) + u_{c,t}) = f_t(x, u)
\]
for all \( x \in X_t \) and \( u \in U_t \), and \( p(d) = p(\Theta(d + Bu_{c,t}) - Bu_{c,t}) \) for all \( d_t \in \Theta(D_t + Bu_{c,t}) - Bu_{c,t} = D_t \).

**Proposition 3.** The set of all symmetries \((\Theta, \Omega)\) that satisfy Definition 1 is a group under matrix multiplication, which we denote \(\text{Aut}(MPC)\).

**Proof.** Associativity follows directly from associativity of matrix multiplication. The identity element \((I, I) \in \text{Aut}(MPC)\) satisfies Definition 1. Each symmetry \(\Theta\) and \(\Omega\) is invertible and the inverses \(\Theta^{-1}\) and \(\Omega^{-1}\) satisfy Definition 1. In particular if we define \( x = \Theta^{-1}(y - x_{c,t}) + x_{c,t} \in X_t \) and \( u = \Omega^{-1}(v - u_{c,t}) + u_{c,t} \in U_t \) then (2.19) gives
\[
f_t(y, v) = f_t(\Theta^{-1}(y - x_{c,t}) + x_{c,t}, \Omega^{-1}(v - u_{c,t}) + u_{c,t})
\]
for any \( y \in X_t \) and \( v \in U_t \).

It remains to show that the set \(\text{Aut}(MPC)\) is closed under matrix multiplication. Let \((\Theta_1, \Omega_1), (\Theta_2, \Omega_2) \in \text{Aut}(MPC)\). We must show that \((\Theta_1\Theta_2, \Omega_1\Omega_2) \in \text{Aut}(MPC)\). We do this by verifying the conditions in Definition 1. Note
\[
\Theta_1\Theta_2 A = \Theta_1 A \Theta_2 = A \Theta_1 \Theta_2
\]
\[
\Theta_1\Theta_2 B = \Theta_1 B \Omega_2 = B \Omega_1 \Omega_2
\]
and
\[
f_t(\Theta_1\Theta_2(x - x_{c,t}) + x_{c,t}, \Omega_1\Omega_2(u - u_{c,t}) + u_{c,t})
\]
\[
= f_t(\Theta_1(\Theta_2(x - x_{c,t}) + x_{c,t} - x_{c,t}) + x_{c,t}, \Omega_1(\Omega_2(u - u_{c,t}) + u_{c,t} - u_{c,t}) + u_{c,t})
\]
\[
= f_t(\Theta_2(x - x_{c,t}) + x_{c,t}, \Omega_2(u - u_{c,t}) + u_{c,t})
\]
\[
= f_t(x, u)
\]
and
\[
\Theta_1\Theta_2(X_t - x_{c,t}) + x_{c,t} = \Theta_1(\Theta_2(X_t - x_{c,t}) + x_{c,t} - x_{c,t}) + x_{c,t}
\]
\[
= \Theta_1(X - x_{c,t}) + x_{c,t}
\]
\[
= X
\]
\[
\Omega_1\Omega_2(U_t - u_{c,t}) + u_{c,t} = \Omega_1(\Omega_2(U_t - u_{c,t}) + u_{c,t} - u_{c,t}) + u_{c,t}
\]
\[
= \Omega_1(U - u_{c,t}) + u_{c,t}
\]
\[
= U
\]

\( \square \)

Symmetries of problem (2.6) affect the exact and certainty equivalence controllers. Proposition 4 shows that symmetries \((\Theta, \Omega) \in \text{Aut}(MPC)\) relate the exact control law \(\pi^*_t\) at different points in the state-space. First we state the following lemma.
**Lemma 1.** Consider the multiparametric program (2.13) where the strictly convex cost $J(x,u)$ and feasible region $\mathcal{F}(x)$ satisfy

\[ J(\Theta(x-x_{c,t}) + x_{c,t}, \Omega(u-u_{c,t}) + u_{c,t}) = J(x,u) \]

\[ \Omega(\mathcal{F}(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) = \mathcal{F}(x) - u_{c,t} \]

\[ \Theta(\lambda' - x_{c,t}) = \lambda' - x_{c,t}. \]

Then the optimal solution satisfies $\Omega(\pi^*(x) - u_{c,t}) + u_{c,t} = \pi^*(\Theta(x-x_{c,t}) + x_{c,t})$ for all $x \in \mathcal{X}$.

**Proof.** First we show that $\Omega(\pi^*(x^-1(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t}$ is a feasible solution to the multiparametric program. Note

\[ \Omega(\pi^*(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t} \]

where $\Theta^{-1}(x-x_{c,t}) + x_{c,t} \in \Theta(\lambda' - x_{c,t}) + x_{c,t} = \lambda'$. Next we show $\Omega(\pi^*(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t}$ is an optimal solution to the multiparametric program. Suppose $\Omega(\pi^*(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t}$ is suboptimal then

\[ J(x, \pi^*(x)) < J(x, \Omega(\pi^*(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t}) \]

which implies

\[ J(y, \Omega^{-1}(\pi^*(\Theta(y-y_{c,t}) + y_{c,t}) - u_{c,t}) + u_{c,t}) < J(y, \pi^*(y)) \]

where $x = \Theta y \in \Theta \lambda' = \lambda'$. However this contradicts the optimality of $\pi^*$ at $y \in \lambda'$. Thus $\pi^*(x)$ and $\Omega(\pi^*(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t}$ are both optimal solutions of the multiparametric program (2.13). Since the cost is strictly convex the solution of the multiparametric program (2.13) is unique. Therefore $\pi^*(x) = \Omega(\pi^*(\Theta^{-1}(x-x_{c,t}) + x_{c,t}) - u_{c,t}) + u_{c,t}$ for all $x \in \lambda'$.

**Proposition 4.** Let $\pi_i^*$ be the solution to (2.7). Then for each $(\Theta, \Omega) \in \text{Aut}(\text{MPC})$ we have $\pi_i^*(\Theta(x-x_{c,t}) + x_{c,t}) = \Omega(\pi_i^*(x) - u_{c,t}) + u_{c,t}$ for all $x \in \lambda_i$.

**Proof.** For each time $t$ the cost function $J_t(x,u)$ in (2.7) is strictly convex. Therefore using Lemma 1 we can show $\pi_i^*(\Theta(x-x_{c,t}) + x_{c,t}) = \Omega(\pi_i^*(x) - u_{c,t}) + u_{c,t}$ for all $x \in \lambda_i$ if the cost-to-go $J_t(x,u)$ and the feasible region $\mathcal{F}_t(x) = \{u : u \in \mathcal{U}, Ax + Bu + d \in \lambda_{t+1}, \forall d \in \mathcal{D}\}$ are symmetric.

First we prove the feasible region $\mathcal{F}_t(x)$ of (2.7) is symmetric. From Definition 1 we have

\[ \mathcal{F}_t(\Theta(x-x_{c,t}) + x_{c,t}) = \{u : u \in \mathcal{U}, A(\Theta(x-x_{c,t}) + x_{c,t}) + Bu + d \in \lambda_{t+1}, \forall d \in \mathcal{D}\} \]

\[ = \{\Omega(u' - u_{c,t}) + u_{c,t} : u' \in \Omega(\mathcal{U} - u_{c,t}) + u_{c,t}, Ax + Bu' + d' \in \lambda_{t+1}, \forall d' \in \Theta(\mathcal{D} + Bu_{c,t}) - Bu_{c,t}\} \]

\[ = \Omega(\mathcal{F}_t(x) - u_{c,t}) + u_{c,t} \]
where \( u' = \Omega^{-1}(u - u_{c,t}) + u_{c,t} \) and \( d' = \Theta^{-1}(d + Bu_{c,t}) - Bu_{c,t} \).

Next we prove by induction that the cost \( J_t(x, u) \) is symmetric. This holds for \( t = N \) by Definition 1. For \( t < N \) assume \( J_{t+1}(x, u) \) and \( \pi_{t+1}^*(x) \) are symmetric. By Definition 1 we have \( f_t(x, u) = f_t(\Theta(x - x_{c,t}) + x_{c,t}, \Omega(u - u_{c,t}) + u_{c,t}) \) for each \((\Theta, \Omega) \in \text{Aut}(MPC)\). We need to show this holds for the second term \( E(J_{t+1}^*(Ax + Bu + d)) \) in the cost function (2.8). Let \( \tilde{x} = \Theta(x - x_{c,t}) + x_{c,t}, \tilde{u} = \Omega(u - u_{c,t}) + u_{c,t} \), \( \hat{x} = Ax + Bu + d' - x_{c,t}, \hat{u} = \pi_{t+1}^*(Ax + Bu + d') - u_{c,t} \), and \( \hat{d} = d' + Bu_{c,t} \). By Definition 1 and the induction hypothesis

\[
E(J_{t+1}^*(Ax + Bu + d)) = \int_{D_t} J_{t+1}(Ax + Bu + d, \pi_{t+1}^*(Ax + Bu + d)) d\mathbf{p}(d)
\]

\[
= \int_{\Theta d' \in D_t} J_{t+1}(\Theta \hat{x} + x_{c,t}, \Omega \hat{u} + u_{c,t}) d\mathbf{p}(\Theta \hat{d} - Bu_{c,t})
\]

\[
= \int_{d' \in D_t} J_{t+1}(\hat{x} + x_{c,t}, \hat{u} + u_{c,t}) d\mathbf{p}(d')
\]

\[
= E(J_{t+1}^*(Ax + Bu + d))
\]

where \( d' = \Theta^{-1}(d + Bu_{c,t}) - Bu_{c,t} \). Therefore by Lemma 1 we conclude \( \pi_t^*(\Theta(x - x_{c,t}) + x_{c,t}) = \Omega(\pi_t^*(x) - u_{c,t}) + u_{c,t} \) for all \( x \in X_t \). \( \square \)

For the certainty equivalence controller \( \pi_t^* \) we have a stronger result: in addition to relating the control law \( \pi_t^* \) at different points in the state-space, symmetries permute the critical regions \( R_t \) of the controller.

**Proposition 5.** Let \( \tilde{\pi}_t^* \) be the solution to (2.9). Then for each \((\Theta, \Omega) \in \text{Aut}(MPC)\) we have \( \tilde{\pi}_t^*(\Theta(x - x_{c,t}) + x_{c,t}) = \Omega(\tilde{\pi}_t^*(x) - u_{c,t}) + u_{c,t} \) for all \( x \in X_t \). Furthermore for any critical region \( R_t^i \in R_t \) there exists \( R_t^j \in R_t \) such that \( R_t^j = \Theta(R_t^i - x_{c,t}) + x_{c,t} \).

**Proof.** For each time \( t \) the cost function \( J_t(x, u) \) in (2.7) is strictly convex. Therefore using Lemma 1 we can show \( \tilde{\pi}_t^*(\Theta(x - x_{c,t}) + x_{c,t}) = \Omega(\tilde{\pi}_t^*(x) - u_{c,t}) + u_{c,t} \) for all \( x \in X_t \) if the cost-to-go \( \tilde{J}_t(x, u) \) and the feasible region \( F_t(x) = \{ u : u \in U, Ax + Bu + d \in X_{t+1}, \forall d \in D \} \) are symmetric.

The symmetry of \( F_t(x) \) has already been proven in the proof for Proposition 4.

Next we prove by induction that the cost \( J_t(x, u) \) is symmetric. This holds for \( t = N \) by Definition 1. For \( t < N \) assume \( \tilde{J}_{t+1}(x, u) \) and \( \tilde{\pi}_{t+1}^*(x) \) are symmetric. By Definition 1 we have \( f_t(x, u) = f_t(\Theta(x - x_{c,t}) + x_{c,t}, \Omega(u - u_{c,t}) + u_{c,t}) \) for each \((\Theta, \Omega) \in \text{Aut}(MPC)\). We need to show this holds for the second term \( \tilde{J}_{t+1}(Ax + Bu + d) \) in the cost function (2.8). Let \( \tilde{x} = \Theta(x - x_{c,t}) + x_{c,t}, \tilde{u} = \Omega(u - u_{c,t}) + u_{c,t} \), and \( \hat{x} = Ax + Bu + E(d_t) \). Since \( E(d_t) = -Bu_{c,t} \), we have

\[
A\tilde{x} + B\tilde{u} + E(d_t) = \Theta(Ax + Bu - Bu_{c,t} - x_{c,t}) + x_{c,t}
\]
By Definition 1 and the induction hypothesis
\[
\tilde{J}_{t+1}(A\hat{x} + Bu + E(d_t)) = \tilde{J}_{t+1}(\Theta(\hat{x} - x_{c,t}) + x_{c,t}, \pi_{t+1}^*(\Theta(\hat{x} - x_{c,t}) + x_{c,t}))
\]
\[
= \tilde{J}_{t+1}(\Theta(\hat{x} - x_{c,t}) + x_{c,t}, \Omega(\pi_{t+1}^*(\hat{x}) - u_{c,t}) + u_{c,t})
\]
\[
= \tilde{J}_{t+1}(\hat{x}, \pi_{t+1}^*(\hat{x}))
\]
\[
= \tilde{J}_{t+1}^*(Ax + Bu + E(d_t)).
\]

Therefore by Lemma 1 we conclude \(\tilde{\pi}_i^*(\Theta(x - x_{c,t}) + x_{c,t}) = \Omega(\tilde{\pi}_i^*(x) - u_{c,t}) + u_{c,t}\) for all \(x \in \mathcal{X}_i\). The fact for any critical region \(\mathcal{R}_i \in \mathcal{R}\) there exists \(\mathcal{R}_j \in \mathcal{R}\) such that \(\mathcal{R}_i^j = \Theta(\mathcal{R}_i - x_{c,t}) + x_{c,t}\) follows from the symmetry of \(\mathcal{X}_i\) and the fact that \(\tilde{\pi}_i^*(\Theta(x - x_{c,t}) + x_{c,t}) = \Omega(\tilde{\pi}_i^*(x) - u_{c,t}) + u_{c,t}\) for all \(x \in \mathcal{X}_i\).

We say two critical regions \(\mathcal{R}_i^1, \mathcal{R}_i^2 \in \mathcal{R}\) are equivalent, if there exists a state-space transformation \(\Theta \in \mathcal{G} = \text{Aut}(\text{MPC})\) such that \(\mathcal{R}_i = \Theta(\mathcal{R}_i - x_{c,t}) + x_{c,t}\). The set of all critical regions equivalent to region \(\mathcal{R}_i\) is called an orbit

\[
\mathcal{G}\mathcal{R}_i = \{\Theta(\mathcal{R}_i - x_{c,t}) + x_{c,t} : \Theta \in \mathcal{G}\} \subseteq \mathcal{R}. \quad (2.20)
\]

The \(P\)-collection \(\mathcal{G}\mathcal{R}_i\) is the set of critical regions \(\mathcal{R}_j = \Theta(\mathcal{R}_i - x_{c,t}) + x_{c,t} \in \mathcal{R}\) that are equivalent to critical region \(\mathcal{R}_i\) under the state-space transformations \(\Theta \in \mathcal{G} = \text{Aut}(\text{MPC})\).

The set of critical region orbits is denoted by \(\mathcal{R}/\mathcal{G} = \{\mathcal{G}\mathcal{R}_1, \ldots, \mathcal{G}\mathcal{R}_r\}\), read as \(\mathcal{R}\) modulo \(\mathcal{G}\), where \(\{\mathcal{R}_1, \ldots, \mathcal{R}_r\}\) is a set that contains one representative critical region \(\mathcal{R}_j\) from each orbit \(\mathcal{G}\mathcal{R}_j\). With abuse of notation we will equate the set of critical region orbits \(\mathcal{R}/\mathcal{G}\) with sets of representative critical regions \(\mathcal{R}/\mathcal{G} = \{\mathcal{R}_1, \ldots, \mathcal{R}_r\}\).

**Symmetric Certainty Equivalence Algorithm**

In this section we use symmetry to reduce the computational complexity of Algorithms 1 and 2.

The following theorem shows that if certainty equivalence holds on a critical region \(\mathcal{R}_i \in \mathcal{R}\) then it holds on the orbit \(\mathcal{G}\mathcal{R}_i \subseteq \mathcal{R}\) of that region \(\mathcal{R}_i\).

**Theorem 2.** If certainty equivalence holds on a critical region \(\mathcal{R}_i \in \mathcal{R}\) then it holds for each critical region \(\mathcal{R}_j = \Theta(\mathcal{R}_i - x_{c,t}) + x_{c,t} \in \mathcal{G}\mathcal{R}_i\) in the orbit \(\mathcal{G}\mathcal{R}_i\).

**Proof.** By definition of certainty equivalence on \(\mathcal{R}\) we have

\[
\pi_i^*(x) = \tilde{\pi}_i^*(x)
\]

for all \(x \in \mathcal{R}_i\). By Propositions 4 and 5 we have \(\Omega(\pi_i^*(x) - u_{c,t}) + u_{c,t} = \pi_i^*(\Theta(x - x_{c,t}) + x_{c,t})\) and \(\Omega(\tilde{\pi}_i^*(x) - u_{c,t}) + u_{c,t} = \tilde{\pi}_i^*(\Theta(x - x_{c,t}) + x_{c,t})\). Thus

\[
\tilde{\pi}_i^*(\Theta^{-1}(x - x_{c,t}) + x_{c,t}) = \Omega^{-1}(\pi_i^*(x) - u_{c,t}) + u_{c,t}
\]

\[
= \Omega^{-1}(\pi_i^*(x) - u_{c,t}) + u_{c,t}
\]

\[
= \pi_i^*(\Theta^{-1}(x - x_{c,t}) + x_{c,t})
\]

(2.22)
for all $\Theta(x - x_{c,t}) + x_{c,t} \in \mathcal{R}_1$. In other words $\tilde{\pi}_t^\ast(x) = \pi_t^\ast(x)$ for all $x \in \Theta(\mathcal{R}_i - x_{c,t}) + x_{c,t}$. \hfill \Box$

This theorem can be used to reduce the number of multi-parametric quadratic programs solved in Algorithm 1. Algorithm 3 is a modification of Algorithm 1 that only tests one representative from each orbit for certainty equivalence. Algorithm 3 maintains a P-collection $P_0 \subseteq X_0$.

**Algorithm 3 Compute Certainty Equivalence Region $P_0 \subseteq X_0$**

1: Solve (2.9) at time $t = N - 1$. Obtain P-collection $\mathcal{R}_{N-1}$ of critical regions. Certain equivalence holds on $P_{N-1} = \mathcal{R}_{N-1}$ for each set in this array by Prop 1.
2: Construct P-collection $P_{N-1}/\mathcal{G}$ that contains one representative region $\mathcal{R}_{N-1}^i$ from each orbit $\mathcal{G}\mathcal{R}_{N-1}^i$ for $\mathcal{R}_{N-1}^i \in \mathcal{R}_{N-1}$.
3: for $t = N - 1$ to 0 do
   4:   for each $\mathcal{R}_t^i \in P_t/\mathcal{G}$ do
      5:     Solve (2.14) at time $t - 1$ with $\mathcal{R}_t^i$. Obtain an array of critical regions $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_r\}$
      6:     while critical region array $\mathcal{R}$ is not empty do
         7:        Select $\mathcal{R}_j \in \mathcal{R}$
         8:        Remove orbit $\mathcal{G}\mathcal{R}_j$ of $\mathcal{R}_j$ from $\mathcal{R}$
         9:        if the constraints indexed by $E_t^i$ are inactive for region $\mathcal{R}_j$ then
            10:            Add $\mathcal{R}_j$ to $P_{t-1}/\mathcal{G}$
            11:        end if
      12:     end while
   13: end for
5:  end for
14: Construct full certainty equivalence P-collection $P_0$ by calculating the orbit $\mathcal{G}\mathcal{R}_0^i$ of each element $\mathcal{R}_0^i$ of $P_0/\mathcal{G}$.

$P_t/\mathcal{G}$ of representative regions $\mathcal{R}_t^i$ for each orbit $\mathcal{G}\mathcal{R}_t^i$ where certainty equivalence holds. This array is initialize by solving the terminal multi-parametric quadratic program (2.9) to obtain the initial certainty equivalence P-collection $P_{N-1} = \mathcal{R}$. The representative certainty equivalence P-collection $P_{N-1}/\mathcal{G}$ is constructed storing a single representative $\mathcal{R}_t^i \in \mathcal{P}_{N-1}$ from each of the orbits $\mathcal{G}\mathcal{R}_t^i \subseteq \mathcal{P}_{N-1}$.

In the dynamic programming loop, Algorithm 3 solves the multiparametric program (2.14) for each representative region $\mathcal{R}_t^i \in P_t/\mathcal{G}$ of the certainty equivalence P-collection $P_t$. This produces an array $\mathcal{R}$ of critical regions. For each orbit of critical regions $\mathcal{G}\mathcal{R}_j \subseteq \mathcal{R}$, Algorithm 3 test one representative $\mathcal{R}_j$ for certainty equivalence. If certainty equivalence holds then the region is added to the representative array $P_{t-1}/\mathcal{G}$. Thus Algorithm 3 only adds a single representative $\mathcal{R}_j$ from each critical region orbit $\mathcal{G}\mathcal{R}_j$.

Finally Algorithm 3 uses symmetry to reconstruct the full certain equivalence P-collection $P_0$ from the P-collection of representative regions $P_0/\mathcal{G}$. 
For problems with piecewise quadratic cost functions (2.17), we initialize $\mathcal{R}_N$ with the pieces of the terminal cost $f_N(x_N)$. Algorithm 4 modifies algorithm 2 to take advantage of system symmetries.

**Algorithm 4 Compute Certainty Equivalence Region $\mathcal{P}_0 \subseteq \mathcal{X}_0$**

1: Construct $\mathcal{P}$-collection $\mathcal{P}_N/\mathcal{G}$ that contains one representative region $\mathcal{R}_N^i$ from each orbit $\mathcal{G}\mathcal{R}_N^i$ for $\mathcal{R}_N^i \in \mathcal{R}_N$.
2: for $t = N$ to 0 do
3:     for each $\mathcal{R}_t^i \in \mathcal{P}_t/\mathcal{G}$ do
4:         for each $\mathcal{X}_{t-1,i}$ do
5:             Solve (2.14) at time $t - 1$ with $\mathcal{R}_t^i$. Obtain an array of critical regions $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_r\}$
6:         while critical region array $\mathcal{R}$ is not empty do
7:             Select $\mathcal{R}_j \in \mathcal{R}$
8:             Remove orbit $\mathcal{G}\mathcal{R}_j$ of $\mathcal{R}_j$ from $\mathcal{R}$
9:             if the constraints indexed by $E_i^j$ are inactive for region $\mathcal{R}_j$ then
10:                Add $\mathcal{R}_j$ to $\mathcal{P}_{t-1}/\mathcal{G}$
11:         end if
12:     end for
13: end for
14: end for
15: end for
16: Construct full certainty equivalence $\mathcal{P}$-collection $\mathcal{P}_0$ by calculating the orbit $\mathcal{G}\mathcal{R}_0^i$ of each element $\mathcal{R}_0^i$ of $\mathcal{P}_0/\mathcal{G}$.

Algorithm 3 requires solving $\sum_{t=0}^N |\mathcal{P}_t/\mathcal{G}|$ multiparametric quadratic programs verses the $\sum_{t=0}^N |\mathcal{P}_t|$ multiparametric programs solved in Algorithm 1. Algorithms 3 and 4 include the additional task of calculating the orbit $\mathcal{G}\mathcal{R}$ of polytopes $\mathcal{R}$. However this can be accomplished efficiently using Algorithm 5.

**Algorithm 5 Orbit $\mathcal{G}\mathcal{R}_i$ of region $\mathcal{R}_i$ in $\mathcal{P}$-collection $\mathcal{R}$**

1: Find point $x \in \text{int}(\mathcal{R}_i)$
2: Calculate orbit $\mathcal{G}x$ of $x$ under $\mathcal{G}$
3: if $y \in \text{int}(\mathcal{R}_j)$ for $\mathcal{R}_j \in \mathcal{R}$ and $y \in \mathcal{G}x$ then
4:     Add $\mathcal{R}_j$ to $\mathcal{G}\mathcal{R}_i$
5: end if

**Symmetry of Soft-Constrained Problems**

In this section, we show that linear soft-constraint cost functions have particular symmetry group which can be exploited for computational speed. In particular, we consider problem
(2.6) with a fixed horizon $N$ and a piece-wise linear terminal and stage costs

$$f_N(x_N) = \sum_{i=1}^{n} \max\{x_{N,i} - x_{\max}, 0, x_{\min} - x_{N,i}\}$$

and

$$f_t(x_t, u_t) = \sum_{i=1}^{n} \max\{x_{t,i} - x_{\max}, 0, x_{\min} - x_{t,i}\} + r(u_t)$$

(2.23)

where $x_{\max}, x_{\min} \in \mathbb{R}$ with $x_{\max} > x_{\min}$, are the maximum and minimum desired bounds, respectively, for each component of $x_t$ and $r : \mathbb{R}^p \rightarrow \mathbb{R}$ is a conical combination of $1-$, $2-$, or $-\infty$ norms. These cost functions penalize violations of the constraints $x_{\min} \leq x \leq x_{\max}$ with a cost that grows linearly with the constraint violation.

We characterize the symmetries of the cost function (2.23).

**Proposition 6.** Consider the cost function (2.23). Let $x_{c,t} = \frac{1}{2}(x_{\max} + x_{\min})$ and $u_{c,t} = \frac{1}{2}(u_{\max} + u_{\min})$. Then the sets of symmetries $\Theta$ and $\Omega$ such that

$$f_t(\Theta(x - x_{c,t}) + x_{c,t}, \Omega(u - u_{c,t}) + u_{c,t}) = f_t(x, u)$$

is isomorphic to the hyperoctohedral groups $BC_n$ and $BC_p$, respectively.

Before proceeding with the proof, we first state the following theorem used in our proof.

**Theorem 3.** [33] The hyperoctohedral group $BC_n$ is isomorphic to the group of matrices with exactly 1 positive or negative 1 in each row and column under matrix multiplication, which we name $PL_n$. This group of matrices are linear mappings which permute the vertices of a hypercube.

We now proceed with the proof to Proposition 6 by showing that the groups of matrices $\Theta$ and $\Omega$ are equal to $PL_n$ and $PL_p$, respectively.

**Proof.** We first prove that the set of $\Theta$ matrices is equal to $PL_n$. Observe that the 0—level set of $f_t$ projected onto $x$ is the hypercube $\{x | x_{\min} \leq x_i \leq x_{\max}\}$. Therefore, the set of matrices $\Theta$ is a subset of $PL_n$. It remains to show that $PL_n$ is a subset of the set of matrices $\Theta$.

Let $\Theta \in PL_n$ and let $x = d + \frac{x_{\max} + x_{\min}}{2\hat{x}}, \hat{x} = \Theta(x - x_{c,t}) + x_{c,t}$, and $\hat{x} = \frac{x_{\min} - x_{\max}}{2}$. We have

$$\sum_{i=1}^{n} \max\{\hat{x}_i - x_{\max}, 0, x_{\min} - \hat{x}_i\} = \sum_{i=1}^{n} \max\{(\Theta d)_i + \hat{x}, 0, \hat{x} - (\Theta d)_i\}. $$

Observe that $\Theta$ is a signed permutation of the coordintaes of $d$. Therefore,

$$\sum_{i=1}^{n} \max\{(\Theta d)_i + \hat{x}, 0, \hat{x} - (\Theta d)_i\} = \sum_{i=1}^{n} \max\{\pm d_i + \hat{x}, 0, \hat{x} \mp d_i\} = \sum_{i=1}^{n} \max\{x_i - x_{\max}, 0, x_{\min} - x_i\}$$
Next, we prove that the set of $\Omega$ matrices is equal to $PL_q$. It is known that the $1-$ and $\infty-$ norms have hyperoctahedral symmetry while the $2-$ norm has orthogonal symmetry, which is a supergroup of the hyperoctahedral group \cite{32}. Therefore, we conclude that the set of $\Omega$ matrices is equal to $PL_q$. \hfill \Box

\section{Disturbances with Unbounded Support}

So far we have assumed that the disturbance set $D_t$ is compact for each time $t = 0, \ldots, N$. However, many commonly used probability distributions, such as Gaussian, have unbounded support. This means the constraints in problem (2.6) cannot be satisfied. Instead we can reformulate problem (2.6) with probabilistic constraints that guarantee constraints satisfaction with near-certain probability $1 - \varepsilon$

\[
\min_{\pi_0, \ldots, \pi_{N-1}} \ E\left( \sum_{t=0}^{N-1} f_t(x_t, u_t) + f_N(x_N, u_N) \right) \tag{2.24}
\]

subject to $P(x_t \in X_t, u_t \in U) \geq 1 - \varepsilon \ \forall t \geq 0$

where $0 < \varepsilon \ll 1$ is an upper-bound on the allowable probability of constraint violation and the controller $u_t = \pi_t(x_t)$, $\pi_t : \mathbb{R}^n \rightarrow \mathbb{R}^p$ is a mapping from the system state $x_t \in \mathbb{R}^n$ to the input space $u_t \in \mathbb{U}$ for $t = 0, \ldots, N - 1$.

To apply the method discussed in section 2.3, we must have compact disturbance set. The idea is to find a polytopic subset $\tilde{D}_t \subset D_t$ of the unbounded disturbance set $D_t$ such that the expected disturbance is unchanged $E(d_t|d_t \in \tilde{D}_t) = E(d_t)$ and the probability that the disturbance $d_t$ is inside $d_t \in \tilde{D}_t$ this set is large $P(d_t \in \tilde{D}_t) = (1 - \varepsilon)^\frac{1}{N}$. We then solve the robust control problem

\[
\min_{\pi_0, \ldots, \pi_{N-1}} \ E\left( \sum_{t=0}^{N-1} f_t(x_t, u_t) + f_N(x_N, u_N) \right) \tag{2.25}
\]

subject to $x_t \in X_t, \forall d_t \in \tilde{D}_t \ \forall t \geq 0$

where $u_t = \pi_t(x_t)$, $\pi_t : X_t \rightarrow U$ is a mapping from the system state $x_t \in \mathbb{R}^n$ to the input space $u_t \in \mathbb{U}$ for $t = 0, \ldots, N - 1$. This approach will result in a conservative solution of the probabilistic constraint problem.

In order to construct this set $\tilde{D}_t$, we make the following assumption about the disturbance $d_t$.

\textbf{Assumption 1.} The probability density function $p(d)$ of the disturbance $d \in \mathcal{D}$ satisfies $p(E(d) + d) = p(E(d) - d)$.

Assumption 1 says that the probability density function $p(d)$ is symmetric about the mean $E(d)$. Under this assumption, it is straightforward to show that if $\tilde{D}_t - E(d_t) \subset \mathbb{R}^n$ is a balanced Borel set $\tilde{D}_t - E(d_t) = -(\tilde{D}_t - E(d_t))$, then $E(d_t|d_t \in \tilde{D}_t) = E(d_t)$.
We can guarantee the optimality of the certainty equivalence approximation with probability $1 - \varepsilon$ by constructing Borel sets $\tilde{D}_t$ for each time step $t$ satisfying the following two assumptions.

**Assumption 2.**

1. $\tilde{D}_t - E(d_t) \subset \mathbb{R}^n$ is balanced $\tilde{D}_t - E(d_t) = -(\tilde{D}_t - E(d_t))$.

2. $P(d_t \in \tilde{D}_t) = (1 - \varepsilon)\frac{1}{N}$.

For certain distributions, such as 1-D Gaussian, the sets $\tilde{D}_t$ satisfying Assumptions 2 are easily computed. For distributions where the computation of the set is not straightforward, generalized versions of the Chebyshev inequality can be employed. Olkin and Pratt [64] provides the following bound for a random vector $(X_1, \ldots, X_n) \in \mathbb{R}^n$.

$$P\left(\bigcap_{i=1}^{n} \frac{|X_i - \mu_i|}{\sigma_i} \leq k_i\right) \geq 1 - \frac{\left(\sqrt{u} + \sqrt{n-1} \sqrt{n\Sigma_{i=1}^{n} \frac{1}{k_i^2} - u}\right)^2}{n^2}, \quad (2.26)$$

where

$$u = \sum_{i=1}^{n} \frac{1}{k_i^2} + 2 \sum_{i=1}^{n} \sum_{j<i} \frac{\rho_{ij}}{k_i k_j},$$

$\mu_i$ is the $i$-th mean, $\sigma_i$ is the $i$-th standard deviation, and $\rho_{ij}$ is the correlation between $X_i$ and $X_j$. Observe that the set $\bigcap_{i=1}^{n} \frac{|X_i - \mu_i|}{\sigma_i} \leq k_i$ is a hypercube which is symmetric about the mean. Therefore, to satisfy Assumption 2, we just need to solve for the $k_i$’s such that

$$1 - \frac{\left(\sqrt{u} + \sqrt{n-1} \sqrt{n\Sigma_{i=1}^{n} \frac{1}{k_i^2} - u}\right)^2}{n^2} \geq (1 - \varepsilon)\frac{1}{N}$$

**Numerical Example**

In this numerical example we demonstrate the use of the Olkin and Pratt bound for a 2-dimensional random variable $X = (X_1, X_2) \in \mathbb{R}^2$. We assume that the random variable is distributed so that the covariance matrix $\Lambda_x$ is

$$\Lambda_x = \begin{bmatrix} 0.5 & 0.1 \\ 0.1 & 0.5 \end{bmatrix}.$$  

Suppose we set $\varepsilon = 0.05$ and $N = 3$. We solve for the variable $k$ by solving the nonlinear program

$$\min_k \left| 1 - \frac{\left(\sqrt{u} + \sqrt{n-1} \sqrt{n\Sigma_{i=1}^{n} \frac{1}{k_i^2} - u}\right)^2}{n^2} - (1 - \varepsilon)\frac{1}{N} \right|$$

s.t. $k \geq 0$
to find $k_1$ and $k_2$ such that
\[
1 - \left( \frac{\sqrt{u + \sqrt{n - 1}} \sqrt{n \Sigma_i k_i} - u}{n^2} \right)^2 = (1 - \varepsilon)^{\frac{1}{n}}, \ i = 1, 2.
\]

The nonlinear program was solved using an interior point method and $k$ was found to be $k_1 = k_2 = 10.8$.

### 2.6 Numerical Examples

In this section we present four numerical examples that demonstrate our methodology.

#### Integrator System

For our first example, we return to the 2-D discrete integrator system described by equation (2.16). Suppose for a horizon $N = 3$ we would like to solve the problem (2.6) with terminal and stage costs

\[
f_N(x_N) = x_N^T x_N
\]

\[
f_t(x_t, u_t) = x_t^T x_t + u_t^T u_t
\]

and constraints $x_t \in [-10, 10]^2$, $u_t \in [-1, 1]^2$, and $d_t \in [-0.5, 0.5]^2$.

Using the method described section 2.3, we compute the set of states at each time step for which certainty equivalence is optimal. At time step 0, the set of states for which the certainty equivalence approximation is exact is plotted below in Figure 2.3a.

From Figure 2.3a it is apparent that this problem has symmetries which can be exploited to reduce computation time and memory usage. Since the matrix $A$, $B$, $Q$, and $R$ are identity, the symmetry group is determined by the constraints sets $\mathcal{X}$ and $\mathcal{U}$ which are squares. The symmetry group is the dihedral-4 group which consists of the four rotations by 90 degrees and reflections about the horizontal, vertical, and both diagonal axis. Using algorithm 3, we compute the representative regions shown in Figure 2.3b where certainty equivalence is optimal. To obtain the full set of states, we simply compute the orbit of each representative region. This is done by rotating the regions by 90 degree increments and reflecting them about the horizontal, vertical, and diagonal axis.

Note that the use of symmetry lead to a reduction in memory cost when storing just the representative regions. The biggest benefit, however, is the decreased computation time of solving 7 mpQP’s (7.83 seconds) instead of 19 mpQP’s (14.76 seconds).
Consider again the 2-D discrete integrator system described by equation (2.16). Suppose for a horizon $N = 4$ we would like to solve the problem (2.6) with cost

$$f_t(x_t, u_t) = \sum_{i=1}^{n} \max\{x_{t,i} - x_{\text{max}}, 0, x_{\text{min}} - x_{t,i}\} + \|u_t\|_\infty + \|u_t\|_1$$

$$f_N(x_N) = \sum_{i=1}^{n} \max\{x_{N,i} - x_{\text{max}}, 0, x_{\text{min}} - x_{N,i}\}$$

where $x_{\text{max}} = 5$ and $x_{\text{min}} = -5$. The constraints are $x_t \in [-10, 10]^2$, $u_t \in [-1, 1]^2$, and $d_t \in [-0.1, 0.1]^2$.

The 2-D discrete integrator system presented above has symmetries which can be exploited to reduce computation time and memory usage. Since the matrices $A$ and $B$ are identity, the symmetry group is determined by the constraints sets and the cost which have hyperoctahedral symmetry. In 2-D, the symmetry group is the dihedral-4 group which consists of the four rotations by 90 degrees and reflections about the horizontal, vertical, and both diagonal axis. Using algorithm 3, we compute the representative regions shown in Fig 2.4 where certainty equivalence is exact. To obtain the full set of states, we simply compute the orbit of each representative region.

**Network Battery System**

In this example we consider a network of $n$ batteries connected in a ring as shown below. The states $x_t$ of the system are the amount of charge on each battery. The inputs $u_t$ are the...
current flows across each edge of the network. The system dynamics can be approximated by a linear system update equation of the form

\[ x_{t+1} = x_t + \frac{I_{\text{max}}}{C} B(u_t + d_t), \]

(2.27)

where \( I_{\text{max}} \) is the maximum current on each edge, \( C \) is the charge capacity of each node, \( B \in \mathbb{R}^{n \times p} \) is the incidence matrix of the graph and \( d_t \) is a stochastic disturbance to the edge flows. The constraints on the system are \( x_t \in [0,1]^n \), \( u_t \in [-1,1]^p \), and \( d_t \in [-0.1,0.1]^n \).

We are interested in balancing the charges on the battery while minimizing the amount of charge moved on each edge. The problem (2.6) with cost (2.10) can be directly applied to solve this problem with

\[ Q = I_n - \frac{1}{n} J_n \quad \text{and} \quad R = 10^{-6} I_n, \]

where \( J_n \) is a \( n \times n \) matrix of ones and \( Q \) penalizes the deviation of the states \( x_t \) from the average \( \frac{1}{n} J_n x \). The set of symmetries for this problem is determined by the graph structure shown in Figure 2.5 which has dihedral symmetry \( D_n \). We can exploit this symmetry to reduce computation time and storage requirements.

We solved the problem with \( n = 5 \) battery cells, \( I_{\text{max}} = 5 \), \( C = 3.6 \cdot 10^5 \), and horizon \( N = 2 \). The tables below compares the solution times and number of critical regions with
and without the use of symmetry. We also report the percentage by volume of the state-space for which certainty equivalence is exact in Table 2.1.

<table>
<thead>
<tr>
<th>N=1</th>
<th>Computation time (s)</th>
<th># of regions</th>
<th>% of state-space</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.8</td>
<td>1.0</td>
<td>211</td>
<td>99.7</td>
</tr>
<tr>
<td>N=2</td>
<td>6,580</td>
<td>1998</td>
<td>99.8</td>
</tr>
<tr>
<td>N=3</td>
<td>63,800</td>
<td>8684</td>
<td>99.8</td>
</tr>
</tbody>
</table>

Table 2.1: Battery network without symmetry

<table>
<thead>
<tr>
<th>N=1</th>
<th>Computation time (s)</th>
<th># of regions</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.1</td>
<td>1.0</td>
<td>26</td>
</tr>
<tr>
<td>N=2</td>
<td>647</td>
<td>213</td>
</tr>
<tr>
<td>N=3</td>
<td>16,000</td>
<td>904</td>
</tr>
</tbody>
</table>

Table 2.2: Battery network with symmetry
Radiant Slab System

In this example we apply our methodology to the radiant-slab system implemented at the Brower Center in Berkeley, CA [37]. The system can be model by the state vector \( x_t = [x_{\text{slab},t} \ x_{\text{room},t}]^T \), where \( x_{\text{slab}} \) is the temperature of the radiant slab and \( x_{\text{room}} \) is the temperature of the room. Let \( u_t \) be the temperature of the water supplied to the radiant slab. The radiant slab system can be approximated by a linear system update equation of the form

\[
x_{t+1} = Ax_t + Bu_t + Wd_t,
\]

where

\[
A = \begin{bmatrix} 0.9579 & 0.0406 \\ 0.0093 & 0.9883 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0016 \\ 0 \end{bmatrix}, \quad W = \begin{bmatrix} 0 \\ 0.0025 \end{bmatrix},
\]

and \( d_t \) is the outside air temperature at time \( t \), with time measured in hours. The parameters in equation (2.28) were experimentally identified by performing step-tests on the actual building.

We are interested in controlling the water temperature supplied to the slabs to maintain the room air temperature close to a comfortable temperature of 70\(^\circ\)F. The supply water temperature is constrained to be within 55\(^\circ\)F and 90\(^\circ\)F. We investigate controlling the building temperature on a hot summer day, with a 48-hour outside air temperature prediction, \( d_t \), as shown in Figure (2.6). We assume that the weather prediction has a 5 degree radius uncertainty, which is shown by the dotted bounding lines above and below the nominal temperature profile. Suppose that we wish to maintain the room temperature, \( x_{\text{room},t} \), close to an optimal temperature of 70\(^\circ\)F while minimizing energy usage. Suppose that the water supply temperature, \( u_t \), has a nominal temperature of 70\(^\circ\)F and that changing the water temperature from the nominal temperature will require energy. Suppose for a horizon of \( N \) hours we would like to minimize the cost

\[
E \left( \sum_{t=0}^{N-1} \left[ (x_{\text{room},t} - 70)^2 + \rho (u_t - 70)^2 \right] + (x_{\text{room},N} - 70)^2 \right)
\]

subject to the robust constraint \([55 \ 80] \leq x_t \leq [90 \ 80]\) and \( u_t \in [55, 90] \). In order to write the cost in the form 2.10, we introduce new states \( \tilde{x}_t = x_t - [70 \ 70] \) and \( \tilde{u}_t = u_t - 70 \). By straightforward substitution, the state update equation becomes

\[
\tilde{x}_{t+1} = A\tilde{x}_t + B\tilde{u}_t + A\begin{bmatrix} 0 \\ 70 \end{bmatrix} + 70B - \begin{bmatrix} 0 \\ 70 \end{bmatrix} + Wd_t
\]

and the cost becomes

\[
f_t(\tilde{x}_t, \tilde{u}_t) = \tilde{x}_t^T \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tilde{x}_t + \rho \tilde{u}_t^2, f_N(\tilde{x}_N) = \tilde{x}_N^T \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tilde{x}_N
\]

For a horizon of 24 hours, the figure 2.7 shows the set of initial states \( x_0 \) such that the certainty equivalence approximation can be used to obtain an exact solution to problem 2.6.
The plot shows that for our radiant-slab system, the set of states for which certainty equivalence can be applied covers almost the entire operating regime. This shows that for the system and problem under consideration, there is little value in knowing the distribution of the disturbance beyond the first moment.

2.7 Conclusion

This section considered finite-time expected value optimization problems for linear systems with additive stochastic disturbance subject to robust constraints and piecewise quadratic cost. We considered problems with quadratic cost separable in time so that dynamic programming can be applied. We presented an algorithm to compute regions of the state-space such that the solution over feedback policies that satisfies robust constraints and minimizes the expected cost is the solution obtained by certainty equivalence. We also presented an algorithm which takes advantage of symmetries in the MPC problem to drastically reduce computation time and memory requirements. The algorithm was demonstrated on several numerical problems. We also demonstrated with the integrator, battery network, and radiant slab systems that symmetries can drastically reduce computation time and memory requirements.
Figure 2.7: $x_0$ for which CE is exact
Chapter 3

Robust control against adversarial disturbances

3.1 Introduction

In this chapter we consider a cost maximized over the disturbances

\[ h(G_D(x_t, u_t)) = \max_{d_t} g(x_t, u_t, d_t) \]

subject to \( d_t \in \mathcal{D} \)

\[ x_{k+1} = f(x_k, u_k, d_k) \quad \text{for all } k = t, \ldots, t + N - 1. \]  

(3.1)

Intuitively, \( h \) describes an intelligent adversary which chooses the action which most disadvantages the player.\[69\]. In particular, we consider finite-time min-max optimization problems for linear systems with additive disturbance subject to hard input constraints and soft state constraints.

We consider a cost composed of a weighted sum of 1- and \( \infty \)- norms on the input and constraint violation vectors. The cost considered does not need to be separable. This is useful, for instance, in energy control problems where peak power (the infinity norm of the entire input sequence over the control horizon) has to be minimized.

The objective of this chapter is to efficiently compute the solution over feedback policies that satisfies input constraints and minimizes the worst case cost for all admissible disturbances. The solution is then used for implementation in min-max robust model predictive control (MPC). We do not impose any structure on the feedback policies.

Min-max robust MPC for linear systems was first investigated by Witsenhausen \[77\] and Campo and Morari \[23\]. In general, the feedback solution is obtained either by dynamic programming or by solving one optimization problem (batch approach) considering the scenario of all possible disturbance realizations \[60, 57, 5, 59, 77, 11, 69, 51, 49, 50, 3\]. Using dynamic programming, min-max robust control of linear systems was investigated by Lee and Yu \[51\] for uncertain parameters residing in ellipsoids and polyhedrons. Offline computation of state
feedback policies for linear systems with additive and parametric uncertainties was investigated by Bemporad, Borrelli and Morari [4] using multi-parametric programming detailed in [17] for linear programs and [6, 72, 71] for quadratic programs. However, dynamic programming in this chapter is challenged by the non-separability in time of the cost, while the cost is assumed to be separable in the cases discussed above. Robust min-max optimization for linear systems with additive disturbance was investigated by Scokaert and Mayne [69] using a single optimization with vertex enumeration. However, the computation time for vertex enumeration is exponential with respect to the horizon length, rendering the solution method intractable for long prediction horizons.

Tractable alternatives to computing exact feedback solutions to the min-max problem include using open-loop input sequences [67] and affine disturbance feedback [42]. However, in general using open-loop input sequences lead to conservative solutions [60]. Goulart, Kerrigan, and Maciejowski [42] detail the use of affine disturbance feedback in the robust control of linear systems with additive disturbance. The solution of the min-max problem using affine disturbance feedback subject to robust linear constraints was addressed by Oldewurtel [63] in the context of stochastic MPC. While affine disturbance feedback is computationally efficient, it is conservative because, in general, the optimal feedback policies are non-linear.

In this chapter, we present a set of state-dependent sufficient conditions which allow for the computation of the exact solution of the min-max problem over all feedback policies without using dynamic programming and yet avoiding vertex enumeration. The idea of this chapter is to predetermine a set of disturbance realizations which, for some initial states and inputs, worst case performance is attained. Then, we compute the set of initial states for which the worst case performance is attained for all admissible inputs at the precomputed set of disturbance realizations. Our experience with building energy control systems show that this approach may be very effective [56, 54]. The computational complexity of the proposed method is linear with respect to the length of the control horizon.

We compare the proposed method to open-loop, vertex enumeration, and affine disturbance feedback techniques [42, 63, 69] in terms of computational complexity and optimal cost using a simple example. We then demonstrate the efficacy of the approach in an implementation of model predictive control for radiant-slab cooling systems.

### 3.2 Problem Definition

Consider the linear time-invariant discrete-time system

\[
x_{t+1} = Ax_t + Bu_t + d_t,
\]

where \(x_t \in \mathbb{R}^n\) is the system state, \(u_t \in \mathbb{R}^p\) the controlled input, \(d_t \in \mathbb{R}^n\) the disturbance, \(A \in \mathbb{R}^{n \times n}\), and \(B \in \mathbb{R}^{n \times p}\). The system is subject to the constraints

\[
u_t \in U, \forall t \geq 0 \text{ and } d_t \in D, \forall t \geq 0,
\]
where $U \subseteq \mathbb{R}^p$ and $D \subseteq \mathbb{R}^n$ are polytopes for all $t \geq 0$.

Consider a fixed control horizon length $N$. We define

$$A = \begin{bmatrix} I_n \\ A \\ \vdots \\ A^N \end{bmatrix}, \quad B = \begin{bmatrix} 0_{n \times p} & \cdots & \cdots & 0_{n \times p} \\ B & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}B & \cdots & AB & B \end{bmatrix},$$

$$W = \begin{bmatrix} 0_n & \cdots & \cdots & 0_n \\ I_n & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1} & \cdots & A & I_n \end{bmatrix}$$

and

$$u_t = \begin{bmatrix} u_t \\ u_{t+1} \\ \vdots \\ u_{t+N-1} \end{bmatrix}, \quad d_t = \begin{bmatrix} d_t \\ d_{t+1} \\ \vdots \\ d_{t+N-1} \end{bmatrix}, \quad x_t = \begin{bmatrix} x_t \\ x_{t+1} \\ \vdots \\ x_{t+N} \end{bmatrix},$$

so that we can compactly rewrite the evolution of system (3.2) as

$$x_t = Ax_t + Bu_t + Wd_t.$$
3.3 Min-Max Finite-Time Optimal Control

At each time $t \geq 0$, we are given a reference input signal $\tilde{u}_k \in \mathbb{R}^p$ for $k \in \{t, \ldots, t + N - 1\}$. We define

$$\tilde{u}_t = [\tilde{u}_t \hspace{1cm} \tilde{u}_{t+1} \hspace{1cm} \cdots \hspace{1cm} \tilde{u}_{t+N-1}]^T.$$  

We are interested in the following min-max problem for a fixed horizon $N$

$$\min_{\pi, \ldots, \pi_{t+N-1}} \max_{d_t \in \mathcal{D}} \|u_t - \tilde{u}_t\|_p + f(\delta(P, b, x_t))$$  

(3.6)

where $u_k = \pi_k(x_0, u_0, \ldots, x_{k-1}, u_{k-1}, x_k)$, $\pi_k : \mathbb{R}^n \times \mathbb{R}^p \times \cdots \times \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^n \rightarrow U$ is a mapping from known system states and inputs $(x_0, u_0, \ldots, x_{k-1}, u_{k-1}, x_k)$ to the input space $u_k \in U$ for $k \in \{t, \ldots, t + N - 1\}$, and $f : \mathbb{R}^m \rightarrow \mathbb{R}^+$ is continuous and non-decreasing in each argument.

The solution to (3.6) can be obtained by using the full enumeration approach [69] described below.

Let $\mathcal{A}$ be an indexing set for the elements of $\mathcal{D}$. Note that since $\mathcal{D}$ is a polytope, $\mathcal{A}$ is uncountably infinite. For $\alpha \in \mathcal{A}$, $d_{j|t}^\alpha \in \mathbb{R}^p$ is the disturbance realization corresponding to $\alpha$ at time $j \geq t$ and $u_{j|t}^\alpha \in \mathbb{R}^p$ is the control action at time $j \geq t$ associated with $d_{j|t}^\alpha$. We define

$$u_\alpha^t = \begin{bmatrix} u_{0|t}^\alpha \\ u_{1|t}^\alpha \\ \vdots \\ u_{t+N-1|t}^\alpha \end{bmatrix}, d_\alpha^t = \begin{bmatrix} d_{0|t}^\alpha \\ d_{1|t}^\alpha \\ \vdots \\ d_{t+N-1|t}^\alpha \end{bmatrix}, x_\alpha^t = \begin{bmatrix} x_{0|t}^\alpha \\ x_{1|t}^\alpha \\ \vdots \\ x_{t+N-1|t}^\alpha \end{bmatrix},$$

where $x_{j+1|t}^\alpha = Ax_{j|t}^\alpha + Bu_{j|t}^\alpha + d_{j|t}^\alpha$ for $j \in \{t, \ldots, t + N - 1\}$ and $x_{t|t}^\alpha = x_t$. Then, we have

$$x_\alpha^t = Ax_t + Bu_\alpha^t + Wd_\alpha^t.$$  

Consider the min-max control problem formulated below.

$$\min_{u_\alpha^t} \max_{\alpha \in \mathcal{A}} \|u_\alpha^t - \tilde{u}_t\|_p + f(\delta(P, b, x_\alpha^t))$$  

subject to $u_{j|t}^\alpha \in U, j \geq t, \forall \alpha \in \mathcal{A}$

$$u_{j|t}^\alpha = u_{j|t}^{\alpha_1} \text{ if } x_{\tau|t}^{\alpha_2} = x_{\tau|t}^{\alpha_2} \forall t \leq \tau \leq j,$$

$$j \geq t, \forall \alpha_1, \alpha_2 \in \mathcal{A}$$  

(3.7)

The constraint $u_{j|t}^{\alpha_1} = u_{j|t}^{\alpha_2}$ if $x_{j|t}^{\alpha_1} = x_{j|t}^{\alpha_2}$ is referred to as a “causality constraint” in [69] to enforce one control action for each feedback state. Note that the norm on the input sequence is left unspecified. In general, the problem as formulated above is computationally intractable.

Remark. The problem described in (3.6) can be useful for applications where energy is to be minimized while state constraint satisfaction is desired but not critical. One example is in building HVAC control, where we are concerned with minimizing energy usage while keeping the room temperatures inside a comfort band as often as possible.
3.4 Sufficient conditions

If a single disturbance realization causes the largest constraint violation at all constraints, then problem (3.7) can be solved by considering only that disturbance realization. This is formalized in the proposition below.

**Proposition 7.** Suppose $\exists \alpha^* \in \mathcal{A}$ that is a solution to

$$\arg \max_{\alpha \in \mathcal{A}} P_i \mathbf{W} d^\alpha_t$$

for all $i \in \{1, ..., m\}$. Then problem (3.7) is equivalent to

$$\min_{\mathbf{u}^\alpha_t} \| \mathbf{u}^\alpha_t - \tilde{\mathbf{u}}_t \|_p + f(\delta(P, b, x_t^\alpha))$$

subject to $u^\alpha_j \in U, j \geq t$ (3.8)

We omit the proof to Proposition 1 since we will prove a generalization of it in the next proposition. If a single disturbance realization causes the largest constraint violation at a subset of the constraints, and all other constraints are never violated, then we can still solve problem (3.7) considering only that disturbance realization. This is stated in the proposition below.

**Proposition 8.** Suppose there is a subset $S$ of the set $\{1, ..., m\}$ such that

1. $\exists \alpha^* \in \mathcal{A}$ that is a solution to

$$\arg \max_{\alpha \in \mathcal{A}} P_i \mathbf{W} d^\alpha_t$$

for all $i \in S$.

2. $P_i A x_t + P_i B u_t^\alpha + P_i \mathbf{W} d^\alpha_t - b_i \leq 0 \forall \alpha \in \mathcal{A}, u^\alpha_{jt} \in U, j \geq t, i \in \{1, ..., m\} \setminus S$.

Then, problem (3.7) is equivalent to problem (3.8)

**Proof.** Let the optimal cost to problem (3.7) be $C_1$ and the optimal cost to problem (3.8) be $C_2$. It is easy to see that $C_1 \geq C_2$. Since $P_i \mathbf{W} d^\alpha_t \leq P_i \mathbf{W} d^\alpha^\ast \forall \alpha \in \mathcal{A}, i \in S$, we have

$$P_i A x_t + P_i B u_t^\alpha + P_i \mathbf{W} d^\alpha_t - b_i \leq P_i A x_t + P_i B u_t^\alpha + P_i \mathbf{W} d^\alpha_t - b_i \forall \alpha \in \mathcal{A}, i \in S.$$ 

Also since $P_i A x_t + P_i B u_t^\alpha + P_i \mathbf{W} d^\alpha_t - b_i \leq 0$ for all $\alpha \in \mathcal{A}, u^\alpha_{jt} \in U, j \geq t, i \notin S$, it follows that if we let $u^\alpha_t = u^\alpha^\ast_t$ then $\delta_i(P, b, x_t^\alpha) \leq \delta_i(P, b, x_t^\alpha) \Rightarrow f(\delta(P, b, x_t^\alpha)) \leq f(\delta(P, b, x_t^\alpha))$ for all $\alpha \in \mathcal{A}, i \in \{1, ..., m\}$. Therefore, every feasible input sequence of problem (3.8) can be used to construct a feasible feedback policy for problem (3.7) achieving the same cost. This implies that $C_1 \leq C_2$. However, we know that $C_1 \geq C_2$ and so $C_1 = C_2$. 

\[\square\]
In the case that the conditions of the first two propositions are not satisfied, we present a third proposition which will guarantee that the disturbance index set $A$ can be reduced to a subset of two elements. This is proved for a special class of problems which satisfy the assumption below.

**Assumption 1:** $D$ is a hypercube and $p = n$. We write

$$D = [d_{min}, d_{max}] \times \cdots \times [d_{min}, d_{max}].$$

Define $d_{max}, d_{min} \in \mathbb{R}^n$ such that $d_{max,i} = d_{max}$ and $d_{min,i} = d_{min}$ for all $i \in \{1, ..., Nn\}$. Let $\alpha_1 \in A$ index $d_{max}$ and $\alpha_2 \in A$ index $d_{min}$. We are ready to prove the following proposition.

**Proposition 9.** Suppose the system (3.2) satisfies Assumptions 1-3. Then, problem (3.7) is equivalent to

$$\begin{align*}
\min_{u_{i}^{\alpha_k}} \max_{k \in \{1,2\}} \|u_{i}^{\alpha_k} - \tilde{u}_i\|_p + f(\delta(P, b, x_i^{\alpha_k})) \\
\text{subject to } u_{j}^{\alpha_k} \in U, j \geq t, \forall k \in \{1,2\} \\
\quad \quad u_{j}^{\alpha_1} = u_{j}^{\alpha_2} \text{ if } x_{\tau|t}^{\alpha_1} = x_{\tau|t}^{\alpha_2} \forall t \leq \tau \leq j, \\
\quad \quad j \geq t \tag{3.9}
\end{align*}$$

if

1. $\|u_{i}^{\alpha_k} - \tilde{u}_i\|_p = \|u_{i}^{\alpha_k} - \tilde{u}_i\|_\infty + \rho \|u_{i}^{\alpha_k} - \tilde{u}_i\|_1$

2. $(PB)_{i,(j-t)}n+k u_{j|t,k}^{\alpha_1} + (PW)_{i,(j-t-1)n+k}d_{max} \geq (PB)_{i,(j-t)}n+k u_{j|t,k}^{\alpha_2} + (PW)_{i,(j-t-1)n+k}d_{min}$ for all $j \geq t+1, i \in \{1, ..., m\}$, $k \in \{1, ..., n\}$ and $(PW)_{i,(N-n)n+k}d_{max} \geq (PW)_{i,(N-n)n+k}d_{min}$ for all $i \in \{1, ..., m\}$, $k \in \{1, ..., n\}$.

3. $|u_{i|t}^{\alpha_1}| \geq |u_{i|t}^{\alpha_2}|$ or $|u_{i|t}^{\alpha_1}| \leq |u_{i|t}^{\alpha_2}|$ for all $i \in \{1, ..., Np\}$,

where $u_{i}^{\alpha_1}$ and $u_{i}^{\alpha_2}$ are the optimizers found for problem (3.9) corresponding to $\alpha_1$ and $\alpha_2$, respectively.

**Proof.** Let the optimal cost to problem (3.7) be $C_1$ and the optimal cost to problem (3.9) be $C_2$. It is clear that $C_1 \geq C_2$.

Using Assumption 1, let $\alpha \in A$ index the disturbance realization $d_{j|t}^{\alpha} = [d_{max} + \lambda_{j,1}(d_{min} - d_{max}), ..., d_{max} + \lambda_{j,n}(d_{min} - d_{max})]$, where $0 \leq \lambda_{j,k} \leq 1$ for $j \geq 1$, $k \in \{1, ..., n\}$. Let $u_{j|t}^{\alpha} = [u_{j|t,1}^{\alpha_1} + \lambda_{j-1,1}(u_{j|t,1}^{\alpha_2} - u_{j|t,1}^{\alpha_1}), ..., u_{j|t,n}^{\alpha_1} + \lambda_{j-1,n}(u_{j|t,n}^{\alpha_2} - u_{j|t,n}^{\alpha_1})]$ for $j \geq t+1$. By condition 2, we have

$$\begin{align*}
P_B u_{i}^{\alpha_1} + P_W d_{max} & \geq P_B u_{i}^{\alpha} + P_W d_{i}^{\alpha} \\
-P_B u_{i}^{\alpha_2} - P_W d_{min} & \geq -P_B u_{i}^{\alpha} - P_W d_{i}^{\alpha}.
\end{align*}$$
So with $u^{a_1} = u^a_i$ and $u^{a_2} = u^a_i$, we have
\[
\begin{align*}
    f(\delta(P, b, x^{a_1}_t)) &\geq f(\delta(P, b, x^{a_2}_t)) \\
    f(\delta(P, b, x^{a_2}_t)) &\geq f(\delta(P, b, x^{a}_t))
\end{align*}
\]

Also, by conditions 1 and 3, we have
\[
\|u^{a_1}_t - \bar{u}_t\|_p \geq \|u^a_t - \bar{u}_t\|_p \text{ or } \|u^{a_2}_t - \bar{u}_t\|_p \geq \|u^a_t - \bar{u}_t\|_p
\]

Therefore, the optimizers for problem (3.9) can be used to construct a feasible feedback policy for problem (3.7) achieving the same cost. This implies that $C_1 \leq C_2$. Since $C_1 \geq C_2$, we have $C_1 = C_2$. \qed

### 3.5 Verifying the propositions’ conditions

In this section, we describe a general procedure to verify the conditions of Propositions 7 and 8.

1. For each $i \in \{1, ... m\}$, solve the linear program
\[
\arg \max_{\alpha \in A} \ P_i W d^a_t
\]
and store the solution $\alpha^{*i}$ and the cost $D^{*i}$.

2. Group the set $\{1, ..., m\}$ into partitions $S_1, ..., S_r$, $r \geq 1$, such that $\exists s_k \in S_k$ so that
\[
P_i W d^{a^{*s_k}}_t = D^{*i}
\]
for all $i \in S_k$, $k \in \{1, ..., r\}$.

3. If $r = 1$ (i.e. there is only one partition), then the condition of proposition (7) is satisfied and the procedure terminates.

4. Suppose there are $r > 1$ partitions. For each $i \in \{1, ..., m\}$, solve the linear program
\[
\arg \max_{u \in U} \ P_i B u
\]
and store the cost $C^{*,i}$.

5. For $k \in \{1, ..., r\}$, compute the polytope of initial states, $\mathcal{X}^k$, defined by the set
\[
\mathcal{X}^k = \{x_t \in \mathbb{R}^n | P_i A x_t + C^{*,i} + D^{*,i} - b_i \leq 0, i \in \{1, ..., m\} \setminus S_k\}
\]
Then, for each $k \in \{1, ..., r\}$, $\alpha^{*,s_k}$, along with the states $x_t \in \mathcal{X}^k$, satisfy the conditions of proposition (8). Therefore, $\bigcup_{k=1}^r \mathcal{X}^k$ are the set of initial states for which proposition (8) can be applied.
CHAPTER 3. ROBUST CONTROL AGAINST ADVERSARIAL DISTURBANCES

Note that if $\mathcal{D}$ and $\mathcal{U}$ are hypercubes, then the linear programs in steps (1) and (4) are easily solved. To illustrate this, consider the linear program

$$\max_{y \in \mathcal{Y}} c^T y,$$

where $\mathcal{Y} \subset \mathbb{R}^n$ is a hypercube defined by $\mathcal{Y} = [y_{min,1}, y_{max,1}] \times \cdots \times [y_{min,n}, y_{max,n}]$. We write $c^T = [c_1 \cdots c_n]$. Then, the optimizer for problem (3.10) is

$$y^*_i = \begin{cases} y_{max,i} & \text{if } c_i \geq 0 \\ y_{min,i} & \text{if } c_i < 0 \end{cases} \quad \forall i \in \{1, ..., n\}.$$

3.6 Numerical example

Efficient batch closed-loop vs. alternative methods

Consider the 1-state, 1-input system described by

$$x_{t+1} = x_t + u_t + d_t$$

subject to the constraints $u_t \in [-1, 1]$ and $d_t \in [-3, 3]$ Let

$$P = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ -1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 13 \\ 13 \\ \vdots \\ 13 \end{bmatrix},$$

where $P \in \mathbb{R}^{2(N+1) \times N+1}$ and $b \in \mathbb{R}^{2(N+1)}$ corresponds to desired box constraints on the system state.

Over a fixed control horizon $N$, consider solving the problem (3.7) at $t = 0$ with norm on the input satisfying condition 1 of Proposition 9 with $\rho = 1$ and $f(\delta) = \|\delta\|_1 + \|\delta\|_\infty$. This problem satisfies A1 and conditions 1-3 of Proposition 9. Therefore, we may apply Proposition 9.

We compare our proposed method to the vertex enumeration method, affined disturbance feedback, and open-loop prediction for the system described above. We tabulate the solution cost for different values of $x_0$ and for $N = 7$ in table 3.1.

The proposed method and vertex enumeration methods achieved the same costs because $d_{max}$ and $d_{min}$ are vertices of the disturbance set. Open-loop predictions and affine disturbance feedback returned higher costs than the two for most of the initial states investigated.
The difference between vertex enumeration and the other methods is apparent when we investigate the mean computation time for the methods. To see this, we fixed $x_0 = 0$ and recorded the computation time in seconds for $N = 2, 3, 4, 5, 6$ in table 3.2.

Vertex enumeration is the slowest of the four methods, which is expected since its complexity is exponential with respect to the horizon length. Affine disturbance feedback, open-loop predictions, and the proposed method had comparable computation times, with affine disturbance feedback only slightly slower than the other two.

We compare the computational times of open-loop prediction, affine disturbance feedback, and our proposed method for a horizon length of $N = 24$ for $x_0 = 0$, for which the vertex enumeration method is computationally intractable. For our proposed method, the computation time was 0.1952 seconds. For open-loop predictions, the computation time was 0.1938 seconds. Using affine disturbance feedback, the computation time was 1.1850 seconds. Our proposed method and open-loop predictions are noticeably faster than affine disturbance feedback, which requires more variables than either of the other two methods. Our proposed method again achieved the lowest cost of the three methods.

**Radiant slab HVAC systems**

We apply our results to the energy efficient control of the radiant slab system at the Brower Center in Berkeley, CA. For this example, we consider the MPC of a single radiant slab zone. We define the thermal model of the system as follows. $T_t = [T_{\text{slab},t} \ T_{\text{room},t}]^T$ represents the states of the system, where $T_{\text{slab}}$ is the temperature of the radiant slab and $T_{\text{room}}$ is the temperature of the room. $u_t$ is the temperature of the water supplied to the radiant slab. The radiant slab system can be approximated by a linear system update equation of the form

$$T_{t+1} = AT_t + Bu_t + Wd_t,$$

(3.13)
where
\[
A = \begin{bmatrix} 0.9579 & 0.0406 \\ 0.0093 & 0.9883 \end{bmatrix}, 
B = \begin{bmatrix} 0.0016 \\ 0 \end{bmatrix}, 
W = \begin{bmatrix} 0 \\ 0.0025 \end{bmatrix},
\]
and \(d_t\) is the outside air temperature at time \(t\), with time measured in hours. The parameters in equation (3.13) were identified by performing step-tests on the actual building.

We are interested in controlling the water temperature supplied to the slabs to maintain the room air temperature within acceptable bounds. Consider a control horizon \(N = 24\) hours and suppose we wish to keep the room air temperature between 65°F and 75°F at all times within the horizon. We define

\[
\begin{bmatrix} 
0 & 1 & \cdots & \cdots & \cdots & 0 \\
0 & -1 & \cdots & \cdots & \cdots & 0 \\
0 & 0 & 0 & 1 & \cdots & 0 \\
0 & 0 & 0 & -1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & \cdots & 0 & 1 \\
0 & \cdots & \cdots & 0 & -1 
\end{bmatrix}, 
\begin{bmatrix} 
75 \\
-65 \\
75 \\
-65 
\end{bmatrix},
\]

where \(P \in \mathbb{R}^{50 \times 50}\) and \(b \in \mathbb{R}^{50}\).

For the system considered, the supply water temperature is constrained to be within 55°F and 90°F. We investigate controlling the building temperature on a hot summer day, with a 48-hour outside air temperature prediction, \(OAT_t\), as shown in Figure (3.1).

![Figure 3.1: Outside air temperature](image_url)

We assume that the weather prediction has a 5 degree radius uncertainty, which is shown by the dotted bounding lines above and below the nominal temperature profile. For this
example, we consider RHC of the radiant slab system over a 24-hour period using a fixed control horizon of $N = 24$ hours. Note that with this horizon length, vertex enumeration is computationally intractable because of the exponential explosion of the number of variables and constraints. We use Proposition (8) to solve for the controller at each time step $t \in \{1, \ldots, 24\}$. From the information given above, at some time $t$, the set of possible input realizations, $\mathcal{U} \subset \mathbb{R}^{24}$, and the set of possible disturbance realizations, $\mathcal{D} \subset \mathbb{R}^{24}$, are given by

$$\mathcal{U} = \prod_{i=1}^{24} [55, 90]$$

and

$$\mathcal{D} = \prod_{i=0}^{N-1} d_{t+i},$$

where

$$d_k = \begin{bmatrix} 0 \\ 0.0025 r_k \end{bmatrix}, r_k \in [OAT_k - 5, OAT_k + 5]$$

for $k \in \{t, \ldots, t + N - 1\}$. The nominal input sequence is when the water temperature is pumped at the normal temperature of $70^\circ F$. Therefore, for all $t$, we have

$$\tilde{u}_t = \begin{bmatrix} 70 \\ \cdots \\ 70 \end{bmatrix}^T$$

Let $S_1$ be the set of positive odd integers less than 50. It can be shown that for all times $t$

$$d_{\text{max},t} = \begin{bmatrix} 0 \\ 0.0025d_{\text{max},t} \end{bmatrix} \times \cdots \times \begin{bmatrix} 0 \\ 0.0025d_{\text{max},t+N-1} \end{bmatrix},$$

where $d_{\text{max},k} = OAT_k + 5$ for $k \in \{t, \ldots, t + N - 1\}$, along with $S_1$ satisfy condition (1) of the proposition. Next, for each time $t = 1, \ldots, 24$, we compute the set of states, $\mathcal{X}_t$, such that condition (2) of Proposition (8) is satisfied. The sets, $\mathcal{X}_t$, are plotted in Figure (3.2) for $t = 0, 7, 15, \text{and } 23$. We consider states lying inside the box $[60, 80] \times [60, 80]$.

We now proceed with the RHC of the radiant slab system. Suppose that we start at the initial state $x_0 = [70\ 66]^T$. At each time $t = 0, \ldots, 23$, we check that the state, $x_t$, is in $\mathcal{X}_t$ and then solve problem (3.8) in order to compute the control sequence $u^*_t$. We then apply the first control input $u^*_t$ and re-solve problem (3.8) at time $t + 1$. We ran the RHC simulation for three different test cases: the nominal disturbance profile, the maximal (hottest) disturbance profile, and the minimal (coldest) disturbance profile. In closed-loop, the state $T_t \in \mathcal{X}_t$ for all $t \in \{1, \ldots, 24\}$. We report the plots of the room temperature for these three test cases in Figure 3.3.

The results for the closed-loop room air temperature are as expected. The plot suggests that as the outside air temperature is increased, the closed-loop room air temperature increases as well. Figure (3.3) shows the closed-loop input water temperature for the three cases. The plot shows that when the maximal disturbance profile is applied, the input water temperature is always saturated at $55^\circ F$. However, if the nominal disturbance profile is applied, the input water temperature switches to the nominal zero-cost value of $70^\circ F$ at certain times of the day. When the minimal disturbance profile is applied, the input water temperature switches to $70^\circ F$ for an even larger period of the day.
Figure 3.2: Set of valid states
3.7 Conclusion

In this chapter, we presented a set of state-dependent sufficient conditions which allow for the computation of the exact solution of the min-max problem with soft-constraint costs over all feedback policies without using dynamic programming and yet avoiding vertex enumeration. The computational complexity of the proposed methods are linear with respect to the length of the control horizon. A general procedure was outlined to compute the set of initial states for which the proposed conditions are satisfied. This can be done offline prior to real-time implementation to ensure fast implementation times.

We compared the proposed method to open-loop, affine disturbance feedback, and vertex enumeration techniques in terms of computational complexity and solution cost using a simple discrete integrator example. It was found that the proposed method is less conservative than either affine disturbance feedback or open-loop predictions while providing faster computational times than affine disturbance feedback and vertex enumeration. We then demonstrated the efficacy of the approach in an implementation of model predictive control for radiant-slab cooling systems. The proposed method found solutions which agree with intuition and successfully kept the room temperature within tolerable bounds.
Chapter 4
Model reduction

4.1 Introduction

In recent years, model predictive control (MPC) for large-scale energy systems has become a reality due to advances in numerical solvers and computing platforms. Some applications include energy efficient buildings [46, 54, 55] and battery balancing [31, 30]. In this chapter, we are interested in optimization problems for large scale systems which are still prohibitively large to be solved online due to time or memory constraints. We consider a cost of the form

\[ h(G_D(x_t, u_t)) = g(x_t, u_t, \hat{d}_t), \]  

(4.1)

where \( \hat{d}_t \in D \). Without further information, \( \hat{d}_t \) is often chosen as the centroid of \( D \) [70]. If \( d_t \) is a random variable with associated probability distribution with well-defined first moment, then it is common to set \( d_t = E(d_t) \) [20].

There has been considerable research on fast optimization algorithms for MPC by taking advantage of the sparsity structure induced by the system dynamics [75, 34, 78]. However, if the problem is very large, even efficient solvers may not be able to cope with the time and memory constraints. Recent advances in symmetric MPC [31] allows for the computation of the explicit solution for large symmetric systems. However, the system must have special symmetry properties which can be hard to satisfy especially with varying external disturbances. Another approach to decreasing computation speed is to exploit computation on distributed platforms [8, 19, 76]. The approach however is limited by the communication times required between the platforms. Therefore if solving the original optimization problem becomes too expensive, one may have to reduce the problem size to decrease computation time and memory. One approach is to reduce the prediction horizon or to use the same input over multiple steps in the horizon, a method termed move blocking [22].

In this chapter, we are concerned with model reduction methods which reduces the number of states and inputs of the system. Standard model reduction techniques [41, 68, 80, 2] provide us with optimal ways of reducing the number of states according to some criteria. Unfortunately, from the perspective of optimization, standard model reduction can be
detrimental to speed and memory. For large-scale sparse systems, optimization algorithms can take advantage of the sparsity of the system to increase computation speed by exploiting sparse linear algebra [35, 1]. However, standard model reduction techniques have the tendency to destroy these sparsity patterns, possibly increasing rather than decreasing the solver speed and memory usage. For larger systems, the process of model reduction itself can also be quite expensive.

In this chapter we present a novel approach to model reduction for large-scale sparse networked energy systems. Such systems can be represented by a collection of strongly connected graphs where each node represents a system state and an edge represents an energy flow between the connected states. The discrete-time system matrix $A$ is non-negative, has row sum 1, and is a direct sum of irreducible matrices. The matrix $A$ has a positive left eigenvector $\alpha^T$ with eigenvalue 1 so that $\alpha^T x(t)$ is the conserved total energy of the system. The approach is similar to the model reduction technique presented in [61] using Petrov-Galerkin projections. Whereas the authors of [61] investigated multiagent leader-follower systems, we focus on networked energy systems and propose reduction heuristics for specific problem classes. The method hinges on the idea that in many networked energy systems, nodes which are located close to each other geographically often produce trajectories which have similar energies. The nodes which have trajectories with similar energies are grouped together and their average energy is propagated. Thus, the approach preserves the total energy of the system. In addition, the method also preserves the sparsity of the optimization problem.

We present our method for two classes of common problems: constrained LQR and soft-constrained problems with application to networked energy systems. We also briefly discuss performance bounds for our method. We show that our method reduces memory usage and improves solver time in both cases. We also show that while standard model reduction can reduce memory usage as well, the process typically destroys the sparsity of the system and has increased solver times and memory usage as compared to the proposed method.

4.2 Problem Definition

Consider the linear time-invariant discrete-time system

$$x(t+1) = Ax(t) + Bu(t) + d(t)$$
$$x(0) = x_0,$$  \hspace{1cm} (4.2)

where $t \in \mathbb{N}^+$, $x(t) \in \mathbb{R}^n$ is the system state, $u(t) \in \mathbb{R}^p$ the controlled input, $d(t) \in \mathbb{R}^n$ the disturbance, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times p}$, and $x_0 \in \mathbb{R}^n$ is a given initial state.
Consider a fixed control horizon length $N$ and the finite-time optimization problem

$$\min f(x(0), ..., x(N), u(0), ..., u(N - 1))$$

subject to $x(t + 1) = Ax(t) + Bu(t) + d(t)$

$$P_x x(t) \leq h_x \forall t \in \{0, ..., N\}$$

$$P_u u(t) \leq h_u \forall t \in \{0, ..., N - 1\}$$

$$x(0) = x_0,$$

where $f$ is a convex cost function. For the remainder of the chapter we assume that the polytopic constraints on the inputs and states are box constraints so that $u_{\min}(t) \leq u(t) \leq u_{\max}(t)$ for some $u_{\min}(t), u_{\max}(t) \in \mathbb{R}^p$ and $x_{\min}(t) \leq x(t) \leq x_{\max}(t)$ for some $x_{\min}(t), x_{\max}(t) \in \mathbb{R}^n$. This can be written in matrix-vector form by setting

$$P_x = \begin{bmatrix} I_n & -I_n \end{bmatrix}, h_x = \begin{bmatrix} x_{\max} \\ -x_{\min} \end{bmatrix}$$

and

$$P_u = \begin{bmatrix} I_p & -I_p \end{bmatrix}, h_u = \begin{bmatrix} u_{\max} \\ -u_{\min} \end{bmatrix}. \quad (4.4)$$

The disturbances $d(t)$ are assumed to be constant vectors in $\mathbb{R}^n$.

### 4.3 State and Input Aggregation

In this section we introduce the concept of state and input aggregation. While the technique is similar to standard model reduction techniques in that the number states are reduced, the approach aims to preserve sparsity and conserves the energy of the system. For the remainder of the chapter, we assume that the system matrix $A$ satisfies the following assumption.

**Assumption 3.** The system matrix $A$ has the following properties.

1. $A$ has a positive left eigenvector $\alpha^T$ with eigenvalue 1

**Remark.** Assumption 3 is satisfied for example by the system matrix of a discrete-time energy network where the network is represented by a disjoint union of strongly connected graphs. The total energy of the system is conserved unless outside input is given to the system. We can interpret $\alpha^T x(t)$ as the total energy of the system since $\alpha^T x(t + 1) = \alpha^T Ax(t) = \alpha^T x(t)$ is a conserved quantity over time if no external inputs are applied to the system.

In this chapter, we investigate constrained and soft-constrained linear optimal control problems. For the soft-constrained problem class, we make the additional assumption that the matrix $B$ is diagonal. That is, each node of the system has an independent input.
Petrov-Galerkin Projection

Suppose we would like to reduce the number of states of system (4.2) to \( k < n \). Let \( Z \in \mathbb{R}^{k \times n} \) and \( T \in \mathbb{R}^{n \times k} \) be such that \( ZT = I_k \). Then, the reduced order model using the Petrov-Galerkin projection \([61]\) is obtained as

\[
\begin{align*}
    x_m(t + 1) &= ZATx_m(t) + ZBu(t) + Zd(t) \\
    x_m(0) &= Zx_0,
\end{align*}
\]

where \( x_m \in \mathbb{R}^k \) is the state of the reduced system. In this chapter, we construct the matrices \( Z \) and \( T \) using the eigenvector \( \alpha \) in Remark 4.3. The particular construction will yield a reduction method which preserves the total energy of the system and sparse system matrices for the reduced system.

State aggregation

Before we describe the construction of the \( Z \) and \( T \) matrices, we first provide some intuition with a description of the state reduction process. We begin by partitioning the network into \( k \) groups of nodes. The reduced order system dynamics are then defined such that the average energies of the \( k \) groups are propagated. To describe the process rigorously, let \( S_x = \{S_{x,1}, ..., S_{x,k}\} \) be a partition of the set \( \{1, ..., n\} \). Intuitively, each \( \{S_{x,i}\} \) represents a grouping of states. We define the state averaging matrix \( Z \) such that \( Z_{ij} = \alpha_j \) if \( j \in S_{x,i} \) and \( Z_{ij} = 0 \) otherwise. The \( Z \) matrix is the linear mapping from the original states to their averaged energies. We now define the matrix \( T \) such that \( ZT = I_k \). In particular, \( T \) is such that \( T_{ij} = 1/(\alpha_i |S_{x,j}|) \) if \( i \in S_{x,j} \) and \( T_{ij} = 0 \) otherwise. The matrix \( T \) interpolates the average energies back to the original states so that \( x(t) \approx Tx_m(t) \).

With this particular construction of the \( Z \) and \( T \) matrices, the matrix \( ZAT \) has some special properties which we prove below.

**Proposition 10.** The matrix \( ZAT \) satisfies the following properties:

1. \( ZAT \) is non-negative
2. \( ZAT \) has column sum equal to 1

**Proof.** The fact that \( ZAT \) is non-negative follows directly from the fact that \( Z, A, \) and \( T \) are all non-negative matrices.

We proceed to the proof of the second property. For ease of notation, we let \( T_i \) be the \( i \)th column of \( T \). We have

\[
ZAT = [ZAT_1 \cdots ZAT_k]
\]

From the definition of \( Z \) we have \( 1_k^TZ = \alpha_i^T \). Therefore,

\[
1_k^T[ZAT_1 \cdots ZAT_k] = [\alpha_i^TAT_1 \cdots \alpha_i^TAT_k]
\]

\[
= [\alpha^T T_1 \cdots \alpha^T T_k]
\]
From the definition of $T$, we have $\alpha^T T_i = 1$ for every $T_i$.

Observe that since the column sum of $ZAT$ is 1, $1_k^T$ is a left eigenvector of $ZAT$. Therefore, $1_k^T x_m(t)$ can be interpreted as the total energy of the aggregated system. Using Proposition 10, we can show that the total energy of the original and aggregated systems are the same for all time. We state this rigorously below.

**Corollary 1.** $\alpha^T x(t) = 1_k^T x_m(t) \ \forall t$

*Proof.* We prove this by induction. Observe that $1_k^T x_m(0) = 1_k^T Z x(0) = \alpha^T x(0)$, thus proving the base case.

Assume that $1_k^T x_m(t) = \alpha^T x(t)$. Observe that

$$1_k^T x_m(t + 1) = 1_k^T (Z AT)x_m(t) + 1_k^T Z B u(t) + 1_k^T Z d(t)$$

$$= 1_k^T x_m(t) + \alpha^T B u(t) + \alpha^T d(t)$$

$$= \alpha^T x(t) + \alpha^T B u(t) + \alpha^T d(t)$$

$$= \alpha^T x(t + 1)$$


**Input aggregation**

One can also reduce the number of input variables by grouping the set of inputs into $q < p$ clusters. The inputs inside each cluster become a constant gain of an average input. To describe this rigorously, let $S_u = \{ S_{u,1}, ..., S_{u,q} \}$ be a partition of the set $\{1, ..., p\}$ so that the elements of $S_{u,i}$ correspond to the inputs grouped into cluster $i$. Let $u_m(t) \in \mathbb{R}^q$ be the condensed set of inputs and $H \in \mathbb{R}^{p \times q}$ be the gain matrix so that $u(t) = Hu_m(t)$ and $u_i(t) = H_{ij} u_{m,j}(t)$ if $i \in S_{u,j}$. The details of constructing $H$ will be discussed in Section 4.4. For ease of notation, we define $A_m = Z AT$ and $B_m = Z BH$.

Before we discuss how to construct the partitions $S_u$ and $S_x$, we will show how to construct constraints on $x_m(t)$ and $u_m(t)$ so that the original constraints on $x(t)$ and $u(t)$ are satisfied.

**Aggregated input and state constraint satisfaction**

In this section, we discuss how to reduce the number of constraints of the problem by imposing constraints on $u_m(t)$ and $x_m(t)$ in system (4.6) so that the original constraints on $u(t)$ and $x(t)$ in system (4.2) are satisfied. We first discuss constraints for $u_m(t)$. Algorithm 6 outlines the procedure to impose constraints on $u_m(t)$ such that the original box constraints of $u(t)$ in problem (4.3) are satisfied.

Algorithm 6 computes conservative constraints for $u_m(t)$ so that its satisfaction for system (4.6) will guarantee satisfaction of the original box constraints on $u(t)$ for system (4.2).
Algorithm 6  INPUT CONSTRAINTS

1: for $S_{u,j} \in S_u$ do
2:    for $t = 0, \ldots, N - 1$ do
3:        for $i \in S_{u,j}$ do
4:            if $H_{i,j} > 0$ then
5:                Set
6:                $\min_i(t) = u_{\min,i}(t)/H_{i,j}$
7:                $\max_i(t) = u_{\max,i}(t)/H_{i,j}$.
8:            else if $H_{i,j} < 0$ then
9:                Set
10:               $\min_i(t) = u_{\max,i}(t)/H_{i,j}$
11:               $\max_i(t) = u_{\min,i}(t)/H_{i,j}$.
12:            else
13:                Set
14:                $\min_i(t) = u_{\min,i}(t)$
15:                $\max_i(t) = u_{\max,i}(t)$.
16:        end if
17:        Set $u_{m,j}(t)$ to lie in the box constraint $\max_i\{\min_i(t)\} \leq u_{m,j}(t) \leq \min_i\{\max_i(t)\}$.
18:    end for
19: end for

Next, we discuss how to impose constraints on the averaged states $x_m(t)$ so that the original state constraints on $x(t)$ in problem (4.3) are satisfied. We can write $x(t) - Tx_m(t)$ as a linear function of $x(0), u(0), \ldots, u(t - 1), d(0), \ldots, d(t - 1)$. For convenience, we write

$$x(t) - Tx_m(t) = g_t(x(0), u(0), \ldots, u(t - 1), d(0), \ldots, d(t - 1)).$$

(4.7)

We present Algorithm 7 to construct the constraints of $x_m(t)$ so that the box constraints of each $x_i(t)$ are satisfied.

The linear programs in Algorithm 7 are solved as follows. Write $g_{t,i} = \sum_{k=0}^{t-1} \sum_{i=0}^{n} \alpha_{k,i} u_i(k) + c(x(0), d(0), \ldots, d(t-1))$, where $c(x(0), d(0), \ldots, d(t-1))$ is a linear function of only $x(0), d(0), \ldots, d(t-1)$. Then

$$\arg \min_u g_{t,i}$$

subject to $P_u u(t) \leq h_u \forall t \in \{0, \ldots, N - 1\}$

is determined by setting $u_i(t) = u_{\min,i}(t)$ if $\alpha_{t,i} \geq 0$ and $u_i(t) = u_{\max,i}(t)$ if $\alpha_{t,i} < 0$. The
Algorithm 7 AVERAGE STATE CONSTRAINTS

1: for \( i = 1, \ldots, n \) do
2:     for \( t = 1, \ldots, N \) do
3:         Solve the linear programs
4:             \[
5:             \min_{x, u} g_{t, i}(t) = \min_{x, u} g_{t, i} \quad \text{s.t.} \quad P_u u(t) \leq h_u \forall t \in \{0, \ldots, N - 1\}
6:             \]
7:             \[
8:             \max_{u} g_{t, i}(t) = \max_{u} g_{t, i} \quad \text{s.t.} \quad P_u u(t) \leq h_u \forall t \in \{0, \ldots, N - 1\},
9:             \]
10: where \( g_t \) is as defined in (4.7).
11: end for
12: for \( S_{x,j} \in S_x \) do
13:     Set the box constraint of \( x_{m}(t) \) to be \( \max_{i \in S_{x,j}} \{x_{min,i}(t) - \min_i \} \leq x_{m}(t) \leq \min_{i \in S_{x,j}} \{x_{max,i}(t) - \max_i \} \).
14: end for
15: end for

process is similar to find

\[
\arg \max_u g_{t, i}
\]

subject to \( P_u u(t) \leq h_u \forall t \in \{0, \ldots, N - 1\} \).

It is straightforward to show that by setting the box constraints for \( x_{m}(t) \) using the algorithm above, the original box constraints on \( x(t) \) will be satisfied. Similar to (4.4) and (4.5), we can express the box constraints in matrix vector form and write

\[
P_{x, x_m}(t) \leq \tilde{h}_x \quad \text{and} \quad P_{u, u_m}(t) \leq \tilde{h}_u.
\]

After implementing the reduction process we solve a reduced version of problem (4.3)

\[
\min_{x_{m,u_m}} f(Tx_m(0), \ldots, Tx_m(N), \quad H u_m(0), \ldots, H u_m(N - 1))
\]

s.t. \( x_m(t + 1) = A_{x_m} x_m(t) + B_{u_m} u_m(t) + Zd(t) \)

\[
P_{x,x_m}(t) \leq \tilde{h}_x \forall t \in \{0, \ldots, N\}
\]

(4.8)

\[
P_{u,u_m}(t) \leq \tilde{h}_u \forall t \in \{0, \ldots, N - 1\}
\]

\( x_m(0) = Zx_0 \).
The reduced optimization problem (4.8), if feasible, will return feasible inputs $H u_m(t)$ which can be used directly. In the following section, we discuss methods to perform aggregation for constrained LQR and soft-constrained problems. We also provide a means to evaluate the suboptimality of the reduced optimization problems.

### 4.4 Method for state and input aggregation

In this section we propose methods to perform state and input aggregation for constrained LQR and soft-constrained problems. The process of aggregation is as follows. We first compute approximate state and input trajectories by using suboptimal inputs for the LQR and soft-constrained problems. We then group the inputs and states which have similar energy trajectories over the control horizon.

#### Constrained LQR aggregation

In this subsection we are concerned with problems of the type

$$
\begin{align*}
\min_{x,u} & \quad x^T(N)Px(N) + \sum_{k=0}^{N-1} x^T(k)Qx(k) + u^T(k)Ru(k) \\
\text{subject to} & \quad x(t+1) = Ax(t) + Bu(t) + d(t) \\
& \quad P_x x(t) \leq h_x \forall t \in \{0,\ldots,N\} \\
& \quad P_u u(t) \leq h_u \forall t \in \{0,\ldots,N-1\} \\
& \quad x(0) = x_0.
\end{align*}
$$

(4.9)

For constrained LQR problems, we propose Algorithm 8 to compute the approximate trajectories.

**Algorithm 8 Compute approximate inputs and state trajectories**

1: Compute offline the finite time unconstrained LQR solution for problem (4.9)
2: for $t = 1,\ldots,N$ do
3:   Apply the unconstrained LQR input $u(t-1)$
4:   Project $u(t-1)$ onto the box constraint
5:   Compute $x(t)$ with $u(t-1)$
6: end for

We use the approximate trajectories to aggregate the states and inputs. Suppose that the user has specified $|S_x|$, the size of the reduced model. We propose the Algorithm 9 to aggregate the states.

Intuitively, Algorithm 9 first computes the total energies of each state trajectory and stores it in the vector $E$. The algorithm then sorts the trajectories according to their energies and groups the states which trajectories are closest to one another.
Algorithm 9  **State aggregation**

1: **for** $i = 1, ..., n$  **do**
2: \hspace{1em} Compute $E_i = \sum_{k=0}^{N} \alpha_i x_i(k)$
3: **end for**
4: Let $\hat{E}$ be $E$ sorted in ascending (or descending) order. Let $\sigma : \{1, ..., n\} \rightarrow \{1, ..., n\}$ be the mapping from the indices in $E$ to the corresponding indices in $\hat{E}$. Let $I = (1, 2, \cdots, n)$
5: **if** $n \mod |S_x| > 0$  **then**
6: \hspace{1em} **for** $i = 1, ..., n \mod |S_x|$  **do**
7: \hspace{2em} Construct $S_{x,i} \subset \{1, ..., n\}$ by adding the top $\lceil n \mod |S_x| \rceil$ elements of $I$ mapped through $\sigma$ to the $S_{x,i}$.
8: \hspace{1em} Set $I = I \setminus S_{x,i}$.
9: **end for**
10: **end if**
11: **for** $i = n \mod |S_x| + 1, ..., |S_x|$  **do**
12: \hspace{1em} Construct $S_{x,i} \subset \{1, ..., n\}$ by adding the top $\lfloor n \mod |S_x| \rfloor$ elements of $I$ mapped through $\sigma$ to the set.
13: \hspace{1em} Set $I = I \setminus S_{x,i}$.
14: **end for**

Suppose the user has also specified $|S_u|$, the size of the reduced number of inputs. We propose Algorithm 10 to aggregate the inputs using the approximate inputs computed in Algorithm 8.

The intuition behind Algorithm 10 is similar. We group the inputs together which are closely separated when averaged over the control horizon.

**Performance bounds for constrained LQR problems**

To evaluate the suboptimality of the aggregation method, we find upper and lower bounds for the original problem cost using the optimizers found with the reduced optimization. The amount of suboptimality is then bounded by the difference between the upper and lower bounds.

Note that upper bounding the cost of the original problem is straightforward. Since the inputs $u_m(t)$ were constrained so that $Hu_m(t)$ are feasible inputs for the original problem, we can simply set $u(t) = Hu_m(t)$ and evaluate the original cost.

To lower bound the performance of our method, we use the dual problem and compute approximately optimal dual variables to arrive at the lower bound.

Before we proceed, we briefly review the dual problem for a quadratic program. For a
Algorithm 10 Input Aggregation

1: for $i = 1, \ldots, p$ do
2:     Compute $U_i = \sum_{k=0}^{N} u_i(k)$
3: end for
4: Let $\hat{U}$ be $U$ sorted in ascending (or descending) order. Let $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ be the mapping from the indices in $U$ to the corresponding indices in $\hat{U}$. Let $I = (1, 2, \cdots, n)$
5: if $p \mod |S_u| > 0$ then
6:     for $i = 1, \ldots, p \mod |S_u|$ do
7:         Construct $S_{u,i} \subset \{1, \ldots, p\}$ by adding the top $\lfloor p \mod |S_u| \rfloor$ elements of $I$ mapped through $\sigma$ to $S_{u,i}$.
8:     end for
9: end if
10: for $i = p \mod |S_u| + 1, \ldots, |S_u|$ do
11:     Construct $S_{u,i} \subset \{1, \ldots, p\}$ by adding the top $\lfloor p \mod |S_u| \rfloor$ elements of $I$ mapped through $\sigma$ to $S_{u,i}$.
12: Set $I = I \setminus S_{u,i}$.
13: end for

QP of the form
\[
\inf_{z} \quad \frac{1}{2} z^T H z + c^T z
\]
subject to $Gz \leq W$
$Cz = H,

the dual problem is
\[
-\inf_{\lambda} \quad \frac{1}{2} \lambda^T (GH^{-1}G^T) \lambda + \lambda^T (W + GH^{-1}c)
+ \frac{1}{2} \nu^T (CH^{-1}C^T) \nu - \nu^T (H + CH^{-1}c)
+ \nu^T CH^{-1}G^T \lambda + \frac{1}{2} c^T H^{-1} c
\]
subject to $\lambda \geq 0$.

Any feasible dual cost is a lower bound to the primal problem.

We can set $z = [x(0), \ldots, x(N), u(0), \ldots, u(N-1)]^T$ and define the matrices $H, c, G, W, C$, and $H$ appropriately to write problem (4.9) in the form of problem (4.10). Algorithm 11 computes suboptimal dual variables which can be used to lower bound the original problem. It is based on techniques described in [14] for initializing dual variables for the primal-dual interior point method. Algorithm 11 first computes the suboptimal primal variables then uses a least squares approach to find the suboptimal dual variables which approximately satisfy the KKT conditions.
Algorithm 11 LQR dual variable approximation

1: Set $u(t) = H u_m(t)$ then compute $x(t+1) = A x(t) + B u(t) + d(t)$ for $t = 0, ..., N - 1$
2: Evaluate $G z - W$ and construct the diagonal matrix $D$ such that $D_{ii} = (G z - W)_i$.
3: Solve the unconstrained least squares problem $\| H z + G^T \lambda + C^T \nu \|_D$ for $\lambda, \nu$.
4: Project each element of $\lambda$ onto $\mathbb{R}^+$
5: Evaluate the dual cost for the lower bound

Remark. The performance bounds allow the user to decide whether or not the current model reduction size is acceptable for their application. The bounds can be used in an adaptation algorithm where the reduced model sizes $k$ and $q$ are increased or decreased depending on the size of the bound. This will be explored in a future work as an improvement to the current method.

Soft-constrained problem aggregation

In this section we are concerned with problems of the type

\[
\min_{x,u} \sum_{k=0}^{N} \max_{i=1}^{n} \{x_i(k) - x_{\max,i}, \ 0, \ x_{\min,i} - x_i(k)\} + \sum_{k=0}^{N-1} p(u(k))
\]

s.t. \( x(t+1) = A x(t) + B u(t) + d(t) \)
\[
P_u u(t) \leq h_u \ \forall t \in \{0, ..., N - 1\}
\]
\[
x(0) = x_0,
\]

where $p(u(t))$ is a function which can be a convex combination of 1- or $\infty$- norms.

Problems of this type are especially relevant to building energy systems, where constraint satisfaction of the zones are not critical but desired. We propose Algorithm 12 to compute the approximate state and input trajectories to be used for aggregation.

We can use then use the approximate state and input trajectories to aggregate the states and inputs using Algorithms 9 and 10.

Performance bounds for soft-constrained problems

We now consider lower bounding the soft-constrained problem (4.11). We briefly review the dual problem for a linear program. For a LP of the form

\[
\inf_x \ c^T z
\]

subject to \( G z \leq W \)
\[
C z = H
\]
Algorithm 12 Compute approximate inputs and state trajectories

1: for $t = 1, \ldots, N$ do
2: Compute $x(t)$ with $u(t - 1) = 0$
3: if $x_i(t) - T_{\text{max}} > 0$ then
4: Set $u_i(t - 1) = \max\left\{\frac{T_{\text{max}} - x_i(t)}{B_{ii}}, u_{\text{min}}\right\}$
5: else if $x_i(t) - T_{\text{min}} < 0$ then
6: Set $u_i(t - 1) = \min\left\{\frac{T_{\text{min}} - x_i(t)}{B_{ii}}, u_{\text{max}}\right\}$
7: else
8: Set $u_i(t - 1) = 0$
9: end if
10: Recompute $x(t)$ with new $u(t - 1)$
11: end for

the dual problem is

$$- \inf_{\lambda} W^T \lambda + H^T \nu$$

subject to

$$c + G^T \lambda + C^T \nu = 0$$

$$\lambda \geq 0.$$ 

It can be shown that any feasible dual cost is a lower bound to the original optimization problem.

For the soft-constrained problem (4.11), slack variables are introduced to rewrite the problem as a linear program. We write $p(u(t)) = \rho_1 \|u(0)\|_1 + \rho_2 \|u(0)\|_\infty$, where $\rho_1, \rho_2 > 0$.

We can similarly define $z, c, G, W, C,$ and $H$ appropriately to write problem (4.13) in the form of problem (4.12). Algorithm 13 computes suboptimal dual variables which can be used to lower bound the original problem.

$$\min_{x} \sum_{k=0}^{N} \sum_{i=1}^{n} \varepsilon_i(k) + \sum_{k=0}^{N-1} \rho_1 s_i(k) + \rho_2 h$$

subject to

$$x(t + 1) = Ax(t) + Bu(t) + d(t)$$

$$P_u u(t) \leq h_u \forall t \in \{0, ..., N - 1\}$$

$$\varepsilon(t) \geq x(t) - x_{\text{max}} \forall t \in \{0, ..., N\}$$

$$\varepsilon(t) \geq 0 \forall t \in \{0, ..., N\}$$

$$\varepsilon(t) \geq x_{\text{min}} - x(t) \forall t \in \{0, ..., N\}$$

$$s(t) \geq u(t) \forall t \in \{0, ..., N - 1\}$$

$$s(t) \geq -u(t) \forall t \in \{0, ..., N - 1\}$$

$$\mathbb{1}_p \otimes h \geq u(t) \forall t \in \{0, ..., N - 1\}$$

$$\mathbb{1}_p \otimes h \geq -u(t) \forall t \in \{0, ..., N - 1\}$$

$x(0) = x_0$. 

(4.13)
Algorithm 13 SOFT-CONSTRAINED DUAL VARIABLE APPROXIMATION

1: for $t = 0, \ldots, N - 1$ do
2:   Set $u(t) = Hu_m(t)$ and compute $x(t + 1)$.
3:   Set $\epsilon_i(t) = \begin{cases} x_i(t) - x_{\text{max}} & \text{if } x_i(t) - x_{\text{max}} > 0 \\ x_{\text{min}} - x_i(t) & \text{if } x_{\text{min}} - x_i(t) > 0 \\ 0 & \text{else} \end{cases}$
4:   Set $s(t) = |u(t)|$
5:   Set $h(t) = \max\{|u_i(t)|\}$
6: end for
7: Set $\epsilon_i(N) = \begin{cases} x_i(N) - x_{\text{max}} & \text{if } x_i(N) - x_{\text{max}} > 0 \\ x_{\text{min}} - x_i(N) & \text{if } x_{\text{min}} - x_i(N) > 0 \\ 0 & \text{else} \end{cases}$
8: Evaluate $Gz - W$ and the diagonal matrix $D$ such that $D_{ii} = (Gz - W)_i$ and 0 elsewhere.
9: Solve the unconstrained least squares problem $\|c + G^T\lambda + C^T\nu\|_{DA}$ for $\lambda, \nu$.
10: Project $\lambda, \nu$ on to the feasible set $c + G^T\lambda + C^T\nu = 0$ and $\lambda > 0$.
11: Evaluate $-W^T\lambda - H^T\nu$ for the lower bound

Algorithms 11 and 13 are conceptually very similar except that the latter is tailored for LP’s.

4.5 Numerical Examples

Constrained LQR

As a first example we look at a constrained LQR problem (4.9) for balancing a sparsely connected battery system. The system matrix $A$ is an identity matrix. We assume that the batteries are connected in a ring configuration as shown in Figure 4.1 so that charge can be moved in either direction between the connected batteries. This is represented mathematically by the input matrix $B \in \mathbb{R}^{n \times n}$ with 1’s on the diagonal, −1’s on the subdiagonal, $B_{1,n} = -1$, and zero elsewhere. The inputs are constrained to lie in the box constraint $[-0.5, 0.5]^n$.

The disturbance profile is generated from a normal distribution so that $d_i(t) \sim \mathcal{N}(0, 0.1^2)$. The initial state $x(0)$ is generated by sorting a uniformly sampled vector in $[0, \beta]^n$, where $\beta$ is a parameter to be varied but fixed for each simulation run. This way nodes which are closely separated have similar starting states.

The cost matrices are set to $Q = R = I_n$.

We solve the problem for $n = 750$ and horizon $N = 10$ using Gurobi [43]. We use our proposed method and balanced model reduction [68] to reduce the state size to $n = 50$. We compare the results in the table below.
CHAPTER 4. MODEL REDUCTION

Figure 4.1: Battery Network

<table>
<thead>
<tr>
<th>$\beta$</th>
<th>Suboptimality of Proposed Method (%)</th>
<th>Suboptimality of Balanced Method (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>1.1679</td>
<td>1.2673</td>
</tr>
<tr>
<td>1.22</td>
<td>3.5615</td>
<td>1.126</td>
</tr>
<tr>
<td>1.44</td>
<td>2.0277</td>
<td>0.9248</td>
</tr>
<tr>
<td>1.66</td>
<td>1.2853</td>
<td>0.89818</td>
</tr>
<tr>
<td>1.88</td>
<td>1.0156</td>
<td>0.83812</td>
</tr>
<tr>
<td>2.11</td>
<td>0.9308</td>
<td>0.8514</td>
</tr>
<tr>
<td>2.33</td>
<td>0.6518</td>
<td>0.7669</td>
</tr>
<tr>
<td>2.55</td>
<td>0.5483</td>
<td>0.7164</td>
</tr>
<tr>
<td>2.77</td>
<td>0.3168</td>
<td>0.6633</td>
</tr>
<tr>
<td>3.00</td>
<td>0.3759</td>
<td>0.6474</td>
</tr>
</tbody>
</table>

Table 4.1: Comparison of Suboptimality for Constrained LQR

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Proposed</th>
<th>Balanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation Time (s)</td>
<td>0.2166</td>
<td>0.0171</td>
<td>0.1496</td>
</tr>
<tr>
<td>Memory Usage (Mb)</td>
<td>10</td>
<td>&lt; 1</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4.2: Comparison of Average Computation Time and Memory Usage for Constrained LQR
We discuss the results in Table 4.1 and 4.2. We observe that the proposed method has similar performance as compared to the balanced reduction method in terms of suboptimality. While our proposed method uses a heuristic to group the states and inputs, its performance in optimization problems is acceptable. Next, we discuss the solver time of the methods. Our proposed method, because of its sparsity preserving properties, has the lowest solution time at 0.0171s. The balanced method, while reducing the problem to the same size, destroyed the sparsity pattern of the original problem and thus has a solution time which is almost an order of magnitude longer. In terms of memory performance, the original problem had a memory requirement which is an order of magnitude higher than the reduced problems, since the problem was reduced by more than 10x. Note that the balanced reduced problem has a memory requirement of more than twice that of the proposed method again because of its failure to preserve the sparsity of the system matrices.

**Soft-constrained problem**

For our second example we look at a soft-constrained problem (4.11) for a sparsely connected network system. The system matrix $A$ is a diagonally-dominant band matrix with a bandwidth of 3. The entries of the matrix are randomly generated from normal distributions. The rows of $A$ are normalized so that all rows sum to 1. We set $B = I_n$. The system matrices represent a sparsely connected network with independent inputs to each node. The inputs are constrained to lie in the box constraint $[-0.25, 0.25]^n$.

The initial state $x(0)$ is generated by sorting a uniformly sampled vector in $[-0.5, 0.5]^n$. This way nodes which are closely separated have similar starting states. The disturbance profile is also uniformly generated so that $d_i(t) \sim \mathcal{N}(0, 0.1x_i(0), 0.1^2)$.

We desire the states to reside in the set $\left[-\beta, \beta\right]$, where $\beta$ is a parameter to be varied but fixed for each simulation. The weights on the $1-$ and $\infty-$ norms of $u(t)$ are set to $\rho_1 = 1$ and $\rho_2 = 1$.

We solve the problem for $n = 1000$ and horizon $N = 10$. We use our proposed method and balanced model reduction to reduce the state size to $n = 20$. We compare the results in the table below.

We discuss the results obtained in Table 4.4 and 4.5. Observe that our proposed method had significantly better performance than the balanced reduction method in terms of suboptimality. The heuristic input sequence used to aggregate the states appear to work quite well for problems of this type. Our proposed method, because of its sparsity preserving properties, has the lowest solution time at 0.9036s. The balanced method, while reducing the problem to the same size, destroyed the sparsity pattern of the original problem and thus has a solution time which is almost an order of magnitude longer. In terms of memory performance, the balanced reduced problem has a memory requirement of more than twice that of the proposed method again because of its failure to preserve the sparsity of the system matrices.
CHAPTER 4. MODEL REDUCTION

\[ \beta \]

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>Suboptimality of Proposed Method (%)</th>
<th>Suboptimality of Balanced Method (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10</td>
<td>12.0985</td>
<td>188.7337</td>
</tr>
<tr>
<td>0.14</td>
<td>12.0469</td>
<td>171.3466</td>
</tr>
<tr>
<td>0.18</td>
<td>11.4854</td>
<td>160.875</td>
</tr>
<tr>
<td>0.23</td>
<td>15.0892</td>
<td>176.7073</td>
</tr>
<tr>
<td>0.27</td>
<td>16.5631</td>
<td>183.5553</td>
</tr>
<tr>
<td>0.32</td>
<td>15.4625</td>
<td>178.9404</td>
</tr>
<tr>
<td>0.36</td>
<td>12.5367</td>
<td>173.1094</td>
</tr>
<tr>
<td>0.41</td>
<td>12.0747</td>
<td>176.9672</td>
</tr>
<tr>
<td>0.45</td>
<td>11.788</td>
<td>181.4068</td>
</tr>
<tr>
<td>0.5</td>
<td>16.2288</td>
<td>177.1942</td>
</tr>
</tbody>
</table>

Table 4.3: Comparison of Suboptimality for Soft-Constrained Problem

<table>
<thead>
<tr>
<th></th>
<th>Suboptimality of Proposed Method (%)</th>
<th>Suboptimality of Balanced Method (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12.0985</td>
<td>188.7337</td>
</tr>
</tbody>
</table>

Table 4.4: Comparison of Suboptimality for Soft-Constrained Problem

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Proposed</th>
<th>Balanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation Time (s)</td>
<td>5.3766</td>
<td>0.9036</td>
<td>9.44</td>
</tr>
<tr>
<td>Memory Usage (Mb)</td>
<td>50</td>
<td>8</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 4.5: Comparison of Average Computation Time and Memory Usage for Soft-Constrained Problem

4.6 Conclusion

We presented an efficient state and input aggregation technique for sparse networked energy systems, focusing on constrained LQR and soft-constrained problems. We presented algorithms to perform the aggregation and also discussed methods to bound the performance of the aggregated optimization problem. We showed through numerical examples that our method is comparable to standard model reduction techniques in terms of sub-optimality but have faster solution times because of its ability to preserve sparsity.
Chapter 5

Robust Symmetric Model Predictive Control

5.1 Introduction

In this chapter, we are interested in the use of explicit symmetric MPC for the fast control of systems for which the computation of the full explicit and real-time controllers are prohibitively expensive. We consider the cost

$$h(G_D(x_t, u_t)) = g(x_t, u_t, 0). \quad (5.1)$$

There has been considerable research on fast optimization algorithms for MPC by taking advantage of the sparsity structure induced by the system dynamics [75, 34, 78]. An approach to decreasing computation speed is to exploit computation on distributed platforms [8, 19, 76]. The approach however is limited by the communication times between the platforms. A different approach to decreasing the computation time is to reduce the prediction horizon or to use the same input over multiple steps in the horizon, a method termed move blocking [22]. However, this approach may render the solution suboptimal or even infeasible. Recent advances in symmetric MPC [31] allows for the computation of the explicit solution for large symmetric systems. However, the system must have special symmetry properties which can be hard to satisfy especially with varying external disturbances. In this chapter, we are concerned with developing a methodology to use symmetric controllers for systems which can be closely approximated as symmetric systems. We shall call such systems quasi-symmetric.

Symmetry has been used extensively in numerous fields to reduce computational complexity. In recent years, symmetry has been applied to optimization to solve linear-programs [15], semi-definite programs [40], and integer-programs [16]. In [36] and [45], symmetry was studied in control theory to decompose large-scale systems into invariant subsystems. In [27], the authors exploited symmetry to reduce the computational complexity of $H_2$ and $H_{\infty}$ controllers. In [31], the authors studied symmetry in linear model predictive control. All of these results rely on the fact that the systems in question have predetermined symmetric properties. In the real world, systems rarely satisfy these requirements because of
Chapter 5. Robust Symmetric Model Predictive Control

Parameter variations or disturbances. However, if the system is sufficiently close to an idealized symmetric system, the symmetric controller might still be used given proper design modifications.

This chapter develops a methodology to construct a persistently feasible and input-to-state stable symmetric controller for an almost symmetric system. The first section of the chapter describes how to identify a symmetric approximation of the original system such that the true system state resides in the sum of the symmetric state and a symmetric disturbance set. The second section of the chapter describes how to construct a symmetric controller for the symmetric system such that the persistent feasibility and input-to-state stability is guaranteed. We do so by using the work of Mayne, et al. [58]. Finally, we demonstrate our methodology through numerical examples, demonstrating the robustness and memory reduction benefits of symmetry controllers.

5.2 System setup

Consider the linear time-invariant discrete-time system with additive disturbance

$$x_{t+1} = Ax_t + Bu_t + d_t, \quad t \geq 0 \quad (5.2)$$

where $x_t \in \mathbb{R}^n$ is the system state, $u_t \in \mathbb{R}^p$ the controlled input, $A \in \mathbb{R}^{n \times n}$, and $B \in \mathbb{R}^{n \times p}$. The system is subject to the constraint

$$x_t \in \mathcal{X}, u_t \in \mathcal{U}, \quad \text{and} \quad d_t \in \mathcal{D} \quad \forall t \geq 0, \quad (5.3)$$

where $\mathcal{X} \subset \mathbb{R}^n$, $\mathcal{U} \subset \mathbb{R}^p$, and $\mathcal{D} \subset \mathbb{R}^n$ are polytopes.

5.3 Robust MPC Design

In this section we briefly review the design of a robust MPC for linear systems with additive disturbance proposed by Mayne, et al. [58]. We assume that a matrix $K \in \mathbb{R}^{n \times p}$ has been designed such that $A + BK$ is stable. Let $Z \subset \mathbb{R}^n$ be such that $(A + BK)Z \oplus D \subset Z$ (i.e. $Z$ is a robust positive invariant set). We design $Z$ so that it is as small as possible. It was shown that such a minimal set is $\bigoplus_{i=0}^{\infty} (A + BK)^i D$ [48], which is typically intractable to compute. A polytopic outer approximation of the set can be computed by truncating the series and appropriately scaling the resultant set [66].
Consider the problem

\[
\begin{align*}
\min_{u_0, \ldots, u_{N-1}, \delta x_0} & \quad \sum_{t=0}^{N} x_t^T Q x_t + u_t^T R u_t \\
\text{subject to} & \quad x_t \in \mathcal{X} \ominus Z, \\
& \quad u_t \in \mathcal{U} \ominus KZ \forall t \geq 0, \\
& \quad x_{t+1} = A x_t + B u_t \forall t \geq 1, \\
& \quad x_1 = A(x_0 - \delta x_0) + B u_t, \\
& \quad \delta x_0 \in Z, \\
& \quad x_N \in \mathcal{X}_f \ominus Z,
\end{align*}
\]

(5.4)

where \( Q, R, P \succ 0 \), \( x_0 \) is the initial state, and \( \mathcal{X}_f \) is assumed to satisfy the following two assumptions.

**Assumption 4.** \((A + BK) \mathcal{X}_f \subset \mathcal{X}_f, \mathcal{X}_f \subset \mathcal{X} \ominus Z, K \mathcal{X}_f \subset \mathcal{U} \ominus KZ.\)

**Assumption 5.** \(((A + BK)x)^T P((A + BK)x) + x^T Q x + (Kx)^T R (Kx) \leq x^T P x \forall x \in \mathcal{X}_f.\)

Let \( u_0^*(x_0) \) and \( \delta x_0^*(x_0) \) be the optimizers for the variables \( u_0 \) and \( \delta x_0 \), respectively, in problem (5.4) as a function of \( x_0 \). The authors in [58] proved the following theorem.

**Theorem 4.** The set \( Z \) is robustly exponentially stable for the controlled uncertain system \( x_{t+1} = Ax_t + Bu_t + K(x_t - \xi^*(x_t)) + d_t \) satisfying Assumptions 4 and 5, where \( d_t \in \mathcal{D} \). The region of attraction is the domain of problem (5.4).

We will use the techniques discussed in this section to design a robust symmetric controller for our quasi-symmetric system. In the following section, we introduce the definition of symmetry and discuss its implications for explicit controllers.

### 5.4 System symmetry

A symmetry of Problem (5.4) is a state-space transformation \( \Theta \) and input-space transformation \( \Omega \) that preserves the dynamics, constraints, and stage cost [31].

**Definition 2.** A linear symmetry of Problem (5.4) is a pair of invertible matrices \((\Theta, \Omega)\) such that for \( t = 0, \ldots, N \)

\[
\begin{align*}
\Theta A &= A \Theta \\
\Theta B &= B \Omega \\
\Theta^T P \Theta &= P \\
\Theta^T Q \Theta &= Q \\
\Omega^T R \Omega &= R
\end{align*}
\]

(5.5a) (5.5b) (5.5c) (5.5d) (5.5e)
\[ \Theta(\mathcal{X} \oplus Z) = \mathcal{X} \oplus Z \quad (5.5f) \]
\[ \Theta(\mathcal{X}_f \oplus Z) = \mathcal{X}_f \oplus Z \quad (5.5g) \]
\[ \Theta Z = Z \quad (5.5h) \]
\[ \Omega U = U \quad (5.5i) \]

Following the convention of [31], we write problem (5.4) as a parametric program.

\[
\begin{align*}
\min_{U, \delta x} & \quad J(U, \delta x_0, x_0) \\
\text{subject to} & \quad U \in \mathcal{T}(x_0, \delta x_0) \\
& \quad x_0 \in \mathcal{X} \\
& \quad \delta x_0 \in \mathcal{Z},
\end{align*}
\]

where \( U = [u(0), \ldots, u(N - 1)]^T \), \( J(U, \delta x_0, x_0) \) is the quadratic cost, and \( \mathcal{T}(x_0, \delta x_0) \) is the set of feasible input trajectories \( U \) that satisfy the state, input, and dynamic constraints.

The system symmetry yields an important proposition which allows for the reduction of memory usage and computation time for explicit controllers.

**Proposition 11.** Let \((U^\star, \delta x_0^\star)\) be the optimizer for problem 5.4 for the initial state \( x_0 \). Then \(((I \otimes \Theta)U^\star, \Theta \delta x_0^\star)\) is the optimizer for the initial state \( \Theta x_0 \).

**Proof.** The proof follows from a modification of Proposition 2 in [31] by substituting \( x_0 - \delta x_0 \) for \( x \). \( \square \)

The proposition allows us to remove redundant controller pieces that are related through symmetry. The amount of memory and computation time reduction can be significant for large problems and is discussed in [31].

### 5.5 Approximate symmetry

When dealing with real world systems, we seldom find systems which are symmetric due to variations in system parameters or limitations in design. However, we can find systems behave closely to a symmetric system. We can approximate such systems with a symmetric one and profit from the advantages of a symmetric controller. In this section we detail the procedure to find a symmetric system which approximates the original system.

In the rest of the chapter, we assume that the nominal constraints \( \mathcal{X} \) and \( \mathcal{U} \) satisfy \( \Theta \mathcal{X} = \mathcal{X} \) and \( \Omega \mathcal{U} = \mathcal{U} \) for \((\Theta, \Omega) \in \mathcal{G}\), where \( \mathcal{G} \) is a finitely-generated group of linear symmetries.

**Identifying system matrices**

Let \( \mathcal{S} \subset \mathcal{G} \) be a subset of the group of symmetries of the constraint polytopes. Let \( \mathcal{T} \) be the generating set of \( \mathcal{S} \). It is straightforward to show that two matrices \( \hat{A} \) and \( \hat{B} \) satisfy
\( \Theta \hat{A} = \hat{A} \Theta \) and \( \Theta \hat{B} = \hat{B} \Omega \) for all \((\Theta, \Omega) \in S\) if and only if \( \Theta \hat{A} = \hat{A} \Theta \) and \( \Theta \hat{B} = \hat{B} \Omega \) for all \((\Theta, \Omega) \in T\). Therefore, the symmetry requirements for the \( \hat{A} \) and \( \hat{B} \) matrices amount to a finite set of linear constraints.

Given two system matrices \( A \) and \( B \) which do not satisfy the symmetry conditions, we wish to identify symmetric matrices \( \hat{A} \) and \( \hat{B} \) which are close in the sense of matrix norms to the original matrices \( A \) and \( B \). We can write the problem as two convex optimizations.

\[
\hat{A} = \arg \min_{\hat{A}} \| \hat{A} - A \| \\
\text{subject to } \Theta \hat{A} = \hat{A} \Theta \forall (\Theta, \Omega) \in T
\]

\[ (5.7) \]

\[
\hat{B} = \arg \min_{\hat{B}} \| \hat{B} - B \| \\
\text{subject to } \Theta \hat{B} = \hat{B} \Omega \forall (\Theta, \Omega) \in T
\]

\[ (5.8) \]

If the Frobenius-norm is used, the optimization programs are quadratic programs. If the \( 1- \) or \( \infty- \) norms are used, then the optimization programs become linear programs.

**Reformulation as symmetric system**

In order to apply the results from symmetric MPC, we must first reformulate system (5.2) as a symmetric system. This is done by reformulating system (5.2) using the matrices identified in section 5.5 plus an additive disturbance which captures the differences between the matrices. Define \( \delta A = A - \hat{A} \) and \( \delta B = B - \hat{B} \). We have

\[
x_{t+1} = Ax_t + Bu_t + d_t \\
= \hat{A}x_t + \hat{B}u_t + \delta Ax_t + \delta Bu_t + d_t
\]

We will treat \( \hat{d}_t = \delta Ax_t + \delta Bu_t + d_t \) as a disturbance to the symmetric system so that the system equation is written as

\[ x_{t+1} = \hat{A}x_t + \hat{B}u_t + \hat{d}_t \]

(5.9)

We will show how to identify disturbance sets \( \hat{D} \) such that \( \hat{d}_t \in \hat{D} \) so as to preserve the overall symmetry of the system. Before we continue, we first present the following propositions.

**Proposition 12.** Suppose two sets \( P, Q \subset \mathbb{R}^n \) satisfy \( \Theta P = P \) and \( \Theta Q = Q \) for some nonsingular \( \Theta \in \mathbb{R}^{n \times n} \), then \( \Theta(P \ominus Q) = P \ominus Q \) and \( \Theta(P \oplus Q) = P \oplus Q \).

**Proof.** We first show that \( \Theta(P \ominus Q) = P \ominus Q \).

(\( \subset \)) Let \( r \in \Theta(P \ominus Q) \). Let \( r' \in P \ominus Q \) be such that \( \Theta r' = r \). By definition, we have \( r' + q \in P \forall q \in Q \). Since \( P = \Theta P \), we have \( \Theta r' + \Theta q \in P \forall q \in Q \). Since \( \Theta \) is nonsingular and \( \Theta r' = r \), we have \( r \in P \ominus Q \).

(\( \supset \)) Let \( r \in P \ominus Q \). Define \( r' = \Theta^{-1} r \). We show that \( r' \in P \ominus Q \). By definition, we have \( r + q \in P \forall q \in Q \). Since \( P = \Theta P \) implies \( P = \Theta^{-1} P \), we have \( \Theta^{-1} r + \Theta^{-1} q \in P \forall q \in Q \). Since \( \Theta^{-1} r = r' \), we have \( r' \in P \ominus Q \). By definition, \( \Theta r' = r \in \Theta(P \ominus Q) \).
Now, we show that $\Theta(P \oplus Q) = P \oplus Q$.

(c) Let $r \in \Theta(P \oplus Q)$. Let $r' \in P \oplus Q$ be such that $\Theta r' = r$. By definition, $\exists p \in P$ and $q \in Q$ such that $r' = p + q$. Therefore, we have $r = \Theta r' = \Theta p + \Theta q$. Since $\Theta p \in P$ and $\Theta q \in Q$, it follows that $r \in P \oplus Q$.

(2) Let $r \in P \oplus Q$. Define $r' = \Theta^{-1} r$. We show that $r' \in P \oplus Q$. By definition, $\exists p \in P$ and $q \in Q$ such that $r = p + q$. Therefore, we have $r' = \Theta^{-1} r = \Theta^{-1} p + \Theta^{-1} q$. Since $\Theta^{-1} p \in P$ and $\Theta^{-1} q \in Q$, it follows that $r' \in P \oplus Q$. By definition, $\Theta r' = r \in \Theta(P \oplus Q)$. $\square$

Let $M \in \mathbb{R}^{m \times n}$. Let $P \subset \mathbb{R}^m$ and $Q \subset \mathbb{R}^n$. Then $P \circ M = \{x \in \mathbb{R}^n | Mx \in P\}$ and

**Proposition 13.** Let $X \subset \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$, $K \in \mathbb{R}^{m \times n}$ be such that $X = \Theta X$, $A \Theta = A \Theta$, and $\Omega K = K \Omega$ for some $\Theta \in \mathbb{R}^{n \times n}$ and $\Omega \in \mathbb{R}^{m \times m}$. Then $X \circ A = \Theta(X \circ A)$ and $K \circ X = \Omega(K \circ X)$

**Proof.** We first prove $X \circ A = \Theta(X \circ A)$.

(c) Let $x \in X \circ A$. By definition we have $Ax \in X$. Since $X = \Theta^{-1} X$, we have $\Theta^{-1}(Ax) = A(\Theta^{-1} x) \in X$. Therefore, $\Theta^{-1} x \in X \circ A$ which implies $x \in \Theta(X \circ A)$.

(2) Let $x \in \Theta(X \circ A)$. Therefore, $\Theta^{-1} x \in X \circ A$. By definition, we have $A(\Theta^{-1} x) = \Theta^{-1}(Ax) \in X$. Since $X = \Theta X$, $Ax \in X$ and so $x \in X \circ A$.

We now prove $K \circ X = \Omega(K \circ X)$.

(c) Let $y \in K \circ X$. Let $x \in X$ be such that $Kx = y$. Since $X = \Theta^{-1} X$, we have $\Theta^{-1} x \in X$. Therefore, $K \Theta^{-1} x = \Omega^{-1} Kx \in K \circ X$. This implies that $Kx \in \Omega(K \circ X)$.

(2) Let $y \in \Omega(K \circ X)$. Therefore, $\Omega^{-1} y \in K \circ X$. Let $x \in X$ be such that $Kx = \Omega^{-1} y$. Since $X = \Theta X$, $\Theta x \in X$. Therefore $K \Theta x = \Omega Kx = y \in K \circ X$ $\square$

Propositions 12 and 13 suggest that we should identify $\hat{D}$ that have the same symmetries as $X$, since computation of robust invariant sets consist of Minkowski sums, Pontryagin differences, and set composition with affine maps. If $X$ has symmetry which is a subgroup of hyperoctahedral symmetry, one can bound $D$ using the 1- or $\infty-$ norms since their sublevel sets have hyperoctahedral symmetry. This is done by solving the LP

$$
\min_{x_t, u_t} \| \delta Ax_t + \delta Bu_t + d_t \|_{1, \infty}
$$

subject to $x_t \in X$,

$$
\begin{align*}
& u_t \in U, \\
& d_t \in D \forall t \geq 0.
\end{align*}
$$

(5.10)

If $X$ has more complicated symmetry, we propose Algorithm 14 to identify $\hat{D}$ with the same symmetry group.

We prove below that Algorithm 14 constructs a set $\bar{D}$ which has the same symmetry group as $X$ and $\bar{D} \in B(0, \varepsilon) \subset \hat{D}$, where $B(0, \varepsilon)$ is the 2- norm ball with radius $\varepsilon$ centered at the origin.

**Proof.** We first prove that $\bar{D}$ and $X$ have the same symmetry group. Let a nonsingular $\Theta \in \mathbb{R}^{n \times n}$ be such that $\Theta X = X$. We show that $\Theta c X = c X$ for any $c \in \mathbb{R}$.
Algorithm 14 Identify \( \hat{\mathcal{D}} \)

1: Solve a LP to find the radius \( r \) of a 2-norm ball which inscribes the set \( \mathcal{X}_t \).
2: Bound \( \| \delta A x_t + \delta B u_t + d_t \|_2 \leq \varepsilon \) by solving LP (5.10) with the 1-norm.
3: Set \( \hat{\mathcal{D}} = \frac{\varepsilon}{r} \mathcal{X}_t \)

\( \subset \) Let \( x \in \Theta c \mathcal{X} \). Let \( x' \in \mathcal{X} \) be such that \( x = \Theta c x' \). Since \( \Theta \mathcal{X} = \mathcal{X} \), we have \( \Theta x' \in \mathcal{X} \). By definition, we have \( c \Theta x' = \Theta c x' \in c \mathcal{X} \).

\( \supset \) Let \( x \in c \mathcal{X} \). Let \( x' \in \mathcal{X} \) be such that \( c x' = x \). Since \( \Theta \mathcal{X} = \mathcal{X} \), we have \( x' \in \Theta \mathcal{X} \).

Therefore, \( \Theta \mathcal{D} = \Theta \frac{\varepsilon}{r} \mathcal{X} = \frac{\varepsilon}{r} \mathcal{X} = \hat{\mathcal{D}} \).

Next, we show that \( B(0, \varepsilon) \subset \hat{\mathcal{D}} \). By construction, we know that \( \hat{d}_t \in B(0, \varepsilon) \), where \( B(0, \varepsilon) \) is the 2-norm ball of radius \( \varepsilon \) centered at the origin. By construction, we have \( B(0, r) \subset \mathcal{X} \). Therefore, we have \( \hat{d}_t \in B(0, \varepsilon) = \frac{\varepsilon}{r} B(0, r) \subset \frac{\varepsilon}{r} \mathcal{X} = \hat{\mathcal{D}} \).

\square

5.6 Symmetric controller design

In this section we describe the design of a robust symmetric controller such that persistent feasibility and input-to-state stability is guaranteed for system (5.2). We assume that the method in section 5.5 was used to construct a symmetric system

\[
 x_{t+1} = \hat{A} x_t + \hat{B} u_t + \hat{d}_t, \tag{5.11}
\]

where \( \hat{d}_t \in \hat{\mathcal{D}} \) so that the group of linear symmetries \( \mathcal{S} \subset \mathcal{G} \) satisfies \( \Theta \hat{A} = \hat{A} \Theta, \Theta \hat{B} = \hat{B} \Theta \), and \( \Theta \hat{d}_t = \hat{d}_t \) for all \( (\Theta, \Omega) \in \mathcal{S} \).

We first discuss the computation of the set \( Z \) in problem (5.4). Before we proceed, we state and prove a few propositions. We assume that we are given \( Q, R \succ 0 \) satisfying \( \Theta^T Q \Theta = Q \) and \( \Omega^T R \Omega = R \) for all \( (\Theta, \Omega) \in \mathcal{S} \).

**Proposition 14.** The solution to the discrete algebraic Riccati equation (DARE) \( P = Q + \hat{A}^T (P - \hat{P} (R + B^T \hat{P} B)^{-1} B^T P) \hat{A} \) for system (5.11) satisfies \( \Theta^T P \Theta = P \) for all \( (\Theta, \Omega) \in \mathcal{S} \).

**Proof.** We first prove \( \Omega (R + B^T P B)^{-1} \Omega^T = (R + B^T P B)^{-1} \). Observe that

\[
 \Omega^T (R + B^T P B) \Omega = \Omega^T R \Omega + \Omega^T B^T P B \Omega
 = R + B^T \Theta^T P \Theta B
 = R + B^T P B.
\]

Rearranging the matrices, we obtain \( \Omega (R + B^T P B)^{-1} \Omega^T = (R + B^T P B)^{-1} \). Next, we show \( \Theta B (R + B^T P B)^{-1} B^T \Theta = B (R + B^T P B)^{-1} B^T \).

\[
 \Theta B (R + B^T P B)^{-1} B^T \Theta = B \Omega (R + B^T P B)^{-1} \Omega^T B^T
 = B (R + B^T P B)^{-1} B^T
\]
We now show that \( \Theta^T P \Theta \) also satisfies the DARE.

\[
\Theta^T P \Theta = \Theta^T Q \Theta + \Theta^T \hat{A}^T (P - PB(R + B^T PB)^{-1}B^T P) \hat{A} \\
= Q + \hat{A}^T (\Theta^T P \Theta - \Theta^T P B (R + B^T PB)^{-1} B^T P \Theta) \hat{A} \\
= Q + \hat{A}^T (\Theta^T P \Theta - \Theta^T P \hat{\Theta} B (R + B^T PB)^{-1} B^T \Theta^T P \Theta) \hat{A}
\]

Therefore, \( \Theta^T P \Theta \) also satisfies the DARE. By uniqueness, \( P = \Theta^T P \Theta \).

**Proposition 15.** The infinite-horizon LQR gain matrix for system (5.11), \( K = -(R + B^T PB)^{-1} B^T P \hat{A} \), satisfies \( \Omega K = K \Theta \) for all \( (\Theta, \Omega) \in S \).

**Proof.** Therefore,

\[
\Omega K = -(R + B^T PB)^{-1} B^T P \hat{A} \\
= -(R + B^T PB)^{-1} (\Omega^T)^{-1} B^T P \hat{A} \Theta \\
= -(R + B^T PB)^{-1} B^T (\Theta^T)^{-1} P \hat{A} \Theta \\
= -(R + B^T PB)^{-1} B^T P \Theta \hat{A} \Theta \\
= -(R + B^T PB)^{-1} B^T P \hat{A} \Theta = K \Theta
\]

**Proposition 16.** The closed-loop LQR system matrix \( A_{cl} = \hat{A} + BK \) satisfies \( \Theta A_{cl} = A_{cl} \Theta \) for all \( (\Theta, \Omega) \in S \).

**Proof.**

\[
\Theta A_{cl} = \Theta \hat{A} + \Theta BK \\
= \hat{A} \Theta + B \Omega K \\
= \hat{A} \Theta + BK \Theta = A_{cl} \Theta
\]

We will use the closed-loop infinite-horizon LQR system to identify the set \( Z \). The authors in [66] proved the following theorem.

**Theorem 5.** Let \( F_\infty \) be the minimal robust invariant set associated with the autonomous system \( x_{t+1} = A_{cl} x + d \), where \( d \in D \). If the integer \( s \in \mathbb{N}^+ \) and scalar \( \alpha \in [0, 1) \) satisfy

\[
A_{cl}^s D \subset \alpha D,
\]

then

\[
F(\alpha, s) = (1 - \alpha)^{-1} \bigoplus_{i=0}^{s-1} A_{cl}^i D
\]

is a polytopic robust invariant set containing \( F_\infty \).
Propositions 12 and 13 imply that if $\Theta \hat{D} = \hat{D}$, then $\Theta Z = Z$.

For larger problems, the method discussed in Theorem 5 can become intractable. In this case, we can compute the maximal robust invariant set in $\alpha \mathcal{X}$, where $0 < \alpha < 1$. The minimal number $\alpha$ can be found iteratively, e.g. with the bisection method. Propositions 12 and 13 imply that if $\mathcal{X}$ satisfies $\Theta \mathcal{X} = \mathcal{X}$, then the maximal robust invariant set in $\alpha \mathcal{X}$, $Z$, satisfies $\Theta Z = Z$.

Next, we discuss the computation of the robust invariant set $\mathcal{X}_f$. In this chapter, we use the infinite-horizon LQR solution and set $\mathcal{X}_f$ as the maximal robust invariant set of the closed-loop LQR system satisfying state and input constraints [18]. This satisfies Assumption 4. Propositions 12 and 13 imply that $\Theta \mathcal{X}_f = \mathcal{X}_f$. Using the DARE solution $P$ as the terminal cost matrix guarantees satisfaction of Assumption 5 [18].

Finally, note that Propositions 12 and 13 imply that $\Omega(U \ominus KZ) = U \ominus KZ$, $\Theta (\mathcal{X} \ominus Z) = \mathcal{X} \ominus Z$, and $\Theta (\mathcal{X}_f \ominus Z) = \mathcal{X}_f \ominus Z$. This shows that problem (5.4) solved with system (5.11) satisfies the symmetry definition (2) with linear symmetries in the group $S$. The entire process of constructing the symmetry controller is summarized below in Algorithm 15.

**Algorithm 15 Computation of symmetry controller**

1: Identify symmetric matrices $\hat{A}$ and $\hat{B}$ which approximate the system matrices $A$ and $B$, respectively, using optimizations (5.7) and (5.8).
2: Identify the symmetric disturbance set $\hat{D}$ by bounding with $1-$ or $\infty-$ norm using (5.10) or with Algorithm 14.
3: Compute the infinite-horizon LQR controller gain $K$ and the associated DARE solution $P$ for the symmetric system.
4: Compute the robust invariant set $Z$ either with Theorem 5 or by computing the maximal robust invariant set in $\alpha \mathcal{X}$ using the infinite-horizon controller.
5: Compute the maximal robust invariant set $X_f$ using the infinite-horizon LQR controller.
6: Solve problem (5.4) using symmetric methods [31].

The authors in [31] discuss methods to compute symmetric controllers. The orbit controller synthesis method first computes the full explicit controller and then reduces the number of controller regions using symmetry. This method can become intractable for large problem sizes even if the symmetric controller has a small number of regions because the full explicit controller is computed first. A second method which takes advantage of symmetries during the initial computation is the fundamental domain controller. The method first computes a subset of the initial state space with which the full initial state space can be recovered using symmetry. Then, the explicit controller is computed only on that subset of the initial state space, reducing both computation time and memory usage during the initial computation.
5.7 Real time evaluation of sub-optimality

It was shown in [31] that the group of system symmetries \( S \) is isomorphic to a group of permutation matrices \( P \) which permute the half-spaces of \( X \) and \( U \), respectively. Let \( \sigma : S \rightarrow P \) be the isomorphism between \( S \) and \( P \). If we write \( X \) and \( U \) in normalized form

\[
X = \{ x \in \mathbb{R}^n | H_x x \leq 1 \} \\
U = \{ u \in \mathbb{R}^p | H_u u \leq 1 \},
\]

then \( \forall (\Theta, \Omega) \in S, (P_x, P_u) = \sigma((\Theta, \Omega)) \in P \) satisfies \( H_x \Theta = P_x H_x \) and \( H_u \Omega = P_u H_u \).

Consider the dual problem to problem (5.6)

\[
\max_{\lambda_x, \lambda_u, \lambda_{\delta x_0}, \nu} \mathcal{D}(\lambda_x, \lambda_u, \lambda_{\delta x_0}, \nu) \\
\lambda_x \geq 0 \\
\lambda_u \geq 0 \\
\lambda_{\delta x_0} \geq 0,
\]

where \( \mathcal{D}(\lambda_x, \lambda_u, \lambda_{\delta x_0}, \nu) \) is the dual function derived from the optimization problem

\[
\min_{U, \delta x_0} \mathcal{L}(U, \delta x_0, x, \lambda_x, \lambda_u, \lambda_{\delta x_0}, \nu).
\]

The function \( \mathcal{L}(U, \delta x_0, x, \lambda_x, \lambda_u, \lambda_{\delta x_0}, \nu) \) is the Lagrangian of problem 5.4 and is written as

\[
\mathcal{L}(U, \delta x_0, x, \lambda_x, \lambda_u, \nu) = J(U, x) + \sum_t \lambda_x^T (H_x x_t - 1) + \sum_t \lambda_u^T (H_u u_t - 1) \\
+ \sum_t \lambda_{\delta x_0}^T (H_{\delta x_0} \delta x_0 - 1) + \sum_t \nu^T (x_{t+1} - Ax_t - Bu_t).
\]

We now present a proposition which allows us to exploit symmetry for the explicit dual solution to problem (5.6).

**Proposition 17.** Let \( (\lambda_x^*, \lambda_u^*, \lambda_{\delta x_0}^*, \nu^*) \) be the optimizer for the dual problem (5.14) for the initial state \( x_0 \). Let \( (\Theta, \Omega) \in S \) and \( (P_x, P_u) = \sigma((\Theta, \Omega)) \in P \), then \( (P_x \lambda_x^*, P_u \lambda_u^*, P_{\delta x_0} \lambda_{\delta x_0}^*, (\Theta^T)^{-1} \nu^*) \) is the optimizer for problem (5.14) for the initial state \( \Theta x_0 \).

**Proof.** Before continuing with the proof, observe that since \( P_x \) and \( P_u \) are permutation matrices, we have \( P_x^T \mathbf{1} = \mathbf{1} \) and \( P_u^T \mathbf{1} = \mathbf{1} \). We now proceed with the proof.

Since \( J(U, x) = J((I_N \otimes \Omega) U, \Theta x) \) and strong duality holds, it is sufficient to show that \( \mathcal{L}(U, \delta x_0, x, \lambda_x, \lambda_u, \nu) = \mathcal{L}((I_N \otimes \Omega) U, \Theta \delta x_0, \Theta x, P_x \lambda_x, P_u \lambda_u, (\Theta^T)^{-1} \nu) \). The Lagrangian consists of the primal cost and three other components corresponding to \( \lambda_x, \lambda_u, \) and \( \nu \). We shall deal with each of these components separately.

Before we proceed, note that substituting \((I_N \otimes \Omega) U \) for \( U, \Theta \delta x_0 \) for \( \delta x_0 \), and \( \Theta x_0 \) for \( x_0 \), results in \( x_t \) being mapped to \( \Theta x_t \) by the equality constraints. This can be shown by induction.
In the following, we substitute \((I_N \otimes \Omega)\) for \(U\), \(\Theta x_0\) for \(x_0\) and \(P_x \lambda_x\) for \(\lambda_x\) to obtain
\[
\sum_{t} (P_x \lambda_{\delta z_0})^T (H_x \Theta \delta x_0 - 1) = \sum_{t} \lambda_{\delta z_0}^T P_x^T (P_x H_x \delta x_0 - 1)
\]
\[
= \sum_{t} \lambda_{\delta z_0}^T (H_x \delta x_0 - 1)
\]
Substituting \(P_x \lambda_{\delta z_0}\) for \(\lambda_{\delta z_0}\), we get
\[
\sum_{t} (P_x \lambda_x)^T (H_x \Theta x_t - 1) = \sum_{t} \lambda_x^T P_x^T (P_x H_x x_t - 1)
\]
\[
= \sum_{t} \lambda_x^T (H_x \Theta x_t - 1)
\]
Substituting \(P_u \lambda_u\) for \(\lambda_u\), we get
\[
\sum_{t} (P_u \lambda_u)^T (H_u \Omega u_t - 1) = \sum_{t} \lambda_u^T P_u^T (P_u H_u u_t - 1)
\]
\[
= \sum_{t} \lambda_u^T (H_u \Omega u_t - 1)
\]
Finally we substitute \((\Theta^T)^{-1} \nu\) for \(\nu\) and obtain
\[
\sum_{t} ((\Theta^T)^{-1} \nu)^T (\Theta x_{t+1} - A \Theta x_t - B \Omega u_t) = \sum_{t} \nu^T \Theta^{-1}(\Theta x_{t+1} - \Theta A x_t - \Theta B u_t)
\]
\[
= \sum_{t} \nu^T (x_{t+1} - A x_t - B u_t)
\]
Therefore, \(L(U, x, \lambda_x, \lambda_u, \nu) = L((I_N \otimes \Omega)U, \Theta x, P_x \lambda_x, P_u \lambda_u, (\Theta^T)^{-1} \nu)\) and the proposition follows.

The above proposition shows that the explicit dual solution can also be stored and computed efficiently using symmetry. A real-time evaluation of the suboptimality of the symmetric controller can be implemented by evaluating the difference between the suboptimal primal and dual costs using the symmetric primal and dual solutions, respectively.

Another possible approach to using the approximate symmetry solution is to use it as a warm starting algorithm to the real-time robust MPC algorithm developed by Zeilinger, et al. [79]. The algorithm presented in [79] is designed to be used for the robust control of general systems for which the explicit solution is too large to compute or store. However, if an approximate symmetric solution can be computed and stored, the symmetric solution can be used as a warm starting solution for the optimization solver.
5.8 Numerical examples

In our first example, we examine a network of \( n \) coupled nodes connected in a ring network. The dynamics of the nodes are described as follows

\[
\begin{align*}
    x_{i,t+1} &= x_{i,t} + a_{i,i-1}(x_{i-1,t} - x_{i,t}) + a_{i,i+1}(x_{i+1,t} - x_{i,t}) \\
    & \quad \text{for } i \in \{2, ..., n-1\} \\
    x_{1,t+1} &= x_{1,t} + a_{1,n}(x_{n,t} - x_{1,t}) + a_{1,2}(x_{2,t} - x_{1,t}) \\
    x_{n,t+1} &= x_{n,t} + a_{n,n-1}(x_{n-1,t} - x_{n,t}) + a_{n,1}(x_{1,t} - x_{n,t})
\end{align*}
\]

where \( a_{i,j} = a_{j,i} \) for all \( i, j \in \{1, ..., n\}, i \neq j \). This results in a banded \( A \) matrix of width 3 with additional non-zero entries at (1, \( n \)) and (\( n, 1 \)). Each \( a_{i,j} \) is generated by sampling the distribution \( \mathcal{N}(0.1, 0.01^2) \). It is assumed that each node is served by an independent input so that \( B = I_{n \times n} \). The system is illustrated in Figure 5.1. The system is also subjected to a uniformly distributed additive disturbance sampled from the distribution \( U(-0.05, 0.05) \). The states and inputs are subject to box constraints so that each element is constrained to lie between \(-1\) and \(1\).

We first solve problem (5.4) explicitly for the original system with \( Q = 10I_{n \times n}, R = I_{n \times n}, n = 8, \) and \( N = 2 \). The terminal constraint and cost matrix are found using the infinite-horizon LQR solution. The explicit solution returned 6557 regions with guaranteed

\[\text{Figure 5.1: Networked nodes}\]
exponential convergence to a set $Z$ of Chebychev radius 0.06. The computation time for the full explicit solution was 538 seconds.

Next, we search for a symmetric approximation of the system with rotational and reflective symmetries. The generator set of the symmetry group consists of two elements, one for rotation and one for reflection. The symmetric $\hat{A}$ matrix was identified using optimization (5.7) with the Frobenius norm. The symmetric additive disturbance set, $\hat{D}$, was identified using optimization (5.10) with the 1-norm.

We then solve problem (5.4) explicitly for the symmetric system using the fundamental domain controller technique with $Q = 10I_{n\times n}$, $R = I_{n\times n}$, $n = 8$, and $N = 2$. The terminal constraint and cost matrix are found using the infinite-horizon LQR solution. The symmetry solution has 237 regions with guaranteed exponential convergence to a set $Z$ of Chebychev radius 0.11. The computation time for the fundamental domain controller was 14 seconds. This is a 96.4% reduction in memory usage and 97.4% reduction in computation time with the penalty of an 83.3% increase in the size of the convergence set $Z$. A closed-loop simulation of the system is shown in Figure 5.2 with all states initialized at 0.8.

![Figure 5.2: Closed-loop simulation of symmetry controller](image)

Next, we consider problem (5.4) for the system with $Q = 10I_{n\times n}$, $R = I_{n\times n}$, $n = 15$, and $N = 2$. For this number of states, the explicit solution for the original problem is intractable. Therefore, we proceed directly to the symmetric approximation. The terminal constraint and
cost matrix are found using the infinite-horizon LQR solution. The symmetry solution has 5405 regions with guaranteed exponential convergence to a set $Z$ of Chebychev radius 0.11.

5.9 Conclusion

In this chapter we study the use of symmetric explicit model predictive control for quasi-symmetric systems. A method is developed to construct symmetric approximations of the original system. The difference between the original and symmetric approximation is modeled as a symmetric additive disturbance to the symmetric system. A symmetric robust explicit MPC is then constructed using existing methods to guarantee input-to-state stability of the original system. We also show how suboptimality can be evaluated in real-time using the explicit solution to the approximate symmetric dual problem. Finally, we conclude with a numerical example demonstrating our technique on a ring network of nodes showing ISS stability and memory reduction using the symmetric approximation.
Chapter 6

Radiant slabs: A case study

In this chapter we present a real-world application of the certainty equivalent principle to the switched control of the radiant slab system at the David Brower Center (DBC) in Berkeley, CA. In this example, the water valves provide binary input to the radiant slabs. Either the water valve is full on or full off. We propose to use a switched discrete linear model to model the dynamic equations (1.3) and (1.4). Let $x_{i,t} = \begin{bmatrix} x_{z,i,t} & x_{slab,i,t} \end{bmatrix}$ be the state vector of zone $i$ and let $h_{i,t}$ and $c_{i,t}$ be the indicator variables for the hot and cold water valves, respectively. Then,

$$
    x_{i,t+1} = \begin{cases}
    A_{cool} x_{i,t} + d_t & \text{if } c_{i,t} = 1, h_{i,t} = 0 \\
    A_{heat} x_{i,t} + d_t & \text{if } c_{i,t} = 0, h_{i,t} = 1 \\
    A_{coast} x_{i,t} + d_t & \text{if } c_{i,t} = 0, h_{i,t} = 0
    \end{cases}
$$

(6.1)

Note that the cooling models were averaged over a range of supply water temperatures. This simplification can reduce model complexity and was tested as valid, because the responses of the slab and the space were slow to changes in water temperature between 15 and 20°C. In cooling tests, we set the cold water supply temperature to the outdoor wetbulb temperature plus 3°C. This was found in preliminary tests to be close to optimal. The heating water temperature was set as a constant at 32°C. Here, $d_t$ is the average net effect of external disturbances to the system, which includes solar loads, internal loads, and outside air temperature.

6.1 Model Validation

In order to validate the correctness of the model, we have identified the model parameters for the calibrated EnergyPlus model of DBC by performing step test simulations. Then, we run the model against a different data set to verify the correctness of the model. Fig 6.1 shows a weeklong open-loop validation of the linear cooling and coasting model for the east zone of the Brower Center. The coasting model for this particular zone had a maximum temperature error of 0.968°C and an average temperature error of 0.304°C over the weeklong period. The
cooling model for this particular zone had a maximum temperature error of 1.124°C and an average temperature error of 0.849°C over the weeklong period.

### 6.2 Problem Formulation

The goal of MPC is to choose the water valve position so that a weighted combination of comfort violation and energy usage is minimized over a prediction horizon $N$. The controls decision is formulated as an optimization problem with constraints. Let $x_{\max,t}$ and $x_{\min,t}$ be the maximum and minimum desired air temperatures at time $t$, respectively. The finite-horizon optimization problem we are solving is

$$
\min_{c_{i,k}, h_{i,k}} \sum_{k=t}^{t+N} \rho \max\{x_{z,i,k} - x_{\max,t}, 0, x_{\min,t} - x_{z,i,k}\} + c_{i,k} + h_{i,k} 
$$

subject to $x_{i,k+1} = \begin{cases} A_{\text{cool}}x_{i,k} + d_k & \text{if } c_{i,k} = 1, h_{i,k} = 0 \\ A_{\text{heat}}x_{i,k} + d_k & \text{if } c_{i,k} = 0, h_{i,k} = 1 \\ A_{\text{coast}}x_{i,k} + d_k & \text{if } c_{i,k} = 0, h_{i,k} = 0 \end{cases} \forall k \in \{t, ..., t + N - 1\}$,

where $\rho$ is a weight to adjust between energy savings and comfort satisfaction. In our experimental runs, $\rho = 1000$, $x_{\max,t} = 26^\circ C$ and $x_{\min,t} = 22^\circ C$ for all $t$.

### 6.3 Implementation

The problem as formulated above is a mixed-integer program. While the problem is in general difficult to solve (there is a combinatorial explosion in computation time), there are efficient solvers available to solve most problems in reasonable time. We used IBM ILOG CPLEX to solve the mixed-integer program in Matlab. In order to interface in closed-loop
with the EnergyPlus model of DBC, we use MLE+ [7]. MLE+ is a Matlab-based toolbox for EnergyPlus/Matlab co-simulation.

### 6.4 Comparison with existing controller

For assessment of the effectiveness of control methods we chose not to run the tests using the weather file of the building site because the mild climatic conditions of the sites provides limited opportunity for the radiant cooling system to operate at all. Instead, Sacramento, CA, representing more severe climate conditions, was selected for the tests. Simulation results from a summer season (June-August) were analyzed here.

**Thermal Comfort**

Thermal comfort can be assessed through thermal comfort categories introduced by the EN 15251 [25] standard. This method of representing the results describes the percentage of occupied hours when the operative temperature exceeds the specified range. During summer operation, for clothing level at 0.55, air speed at 0.12 m/s, metabolic rate at 1.2 and humidity level at 50%, the operative temperature range to achieve Category II (Predicted Percentage Dissatisfied (PPD) ≤ 10%) is 23-26°C and for Category III (PPD ≤ 15%) the range is 22-27°C. For long term performance, according to EN15251 [25] Appendix G, the recommended criteria for acceptable deviation is that the percentage of exceedance be less than 5% of occupied hours of a day, week, month, and year.

Fig 6.2 compares the thermal comfort level for each zone using MPC and heuristic methods. For simplification, the percentages are labeled only for Categories II and III. Overall, the MPC controller was able to maintain zone operative temperatures at Category II thermal comfort level more than 95% of the occupied hours for all zones. With the heuristic method, only the core zone operative temperatures were maintained at Category II level for more than 95% of the occupied hours; for the east zone, only 88.3% of the time. In addition, the MPC controlled zones reached Category IV only 1.3% of the time (the highest zone operative temperature was 27.6°C and the minimum was 20.6°C, and both incidents happened in the west zone), while the heuristic method controlled zones reached Category IV 2.5% of the time (the highest zone operative temperature was 27.5°C in the west zone and the minimum was 21.7°C in the core zone).

**Energy Consumption**

The itemized HVAC energy consumptions are presented in Fig 6.3. Compared to the heuristic control method, MPC reduced total energy consumption by 14.4%. For cooling tower energy consumption, it is a 55% reduction, and for pumps, it is 26%. 
Figure 6.2: Comparison of thermal comfort performance of MPC and heuristic control method based on EN 15251 Categories for a typical summer season in Sacramento, CA (June-August).

Figure 6.3: Comparison of energy consumptions between MPC and heuristic methods (June-August).
Examples of MPC and heuristic control actions

Figs 6.4 and 6.5 present zone operative temperatures and valve operating conditions for two example days from the test. The first example (Fig 6.4) shows the east zone conditions on July 09th, which features a range of outdoor air temperature from 18.3°C at 4:00 am to 39.0°C at 6:00 pm. The wetbulb temperature ranges from 14 to 20°C, and the cooling tower was able to generate cold water at temperatures from 19 to 25°C. With the morning sun hitting the window and later triggering the interior blinds to come down, there was a bump in zone operative temperature in the morning. With the heuristic control, as the maximum outdoor air temperature of the previous day exceeded 28°C, precooling was kicked on from 10:00 pm on the previous day to 6:00 am. The system then stayed off until about 10:00am when zone operative temperature rose to 24°C. At 3:00 pm, outdoor wetbulb temperature is too high and the cooling tower was no longer able to generate water with temperature cool enough and the valve shut off. Zone operative temperature swung from around 23°C early in the morning to a peak of 26.5°C at 3:00 pm. While with the MPC controller, with the predictive knowledge of high cooling demand throughout the day, radiant cooling continues until 3:00 pm and zone operative temperature was maintained well below 26°C, which was set as the upper boundary for thermal comfort in the controller.

The second example (Fig 6.5) shows the south zone conditions on July 26th, which features a range of outdoor air temperature from 13.0°C at 3:00 am to 34.0°C at 5:00 pm. The wetbulb temperature ranges from 12.5 to 20.1°C, and the cooling tower was able to generate cold water at temperatures from 18.9 to 24.7°C. With the heuristic control, precooling was kicked on until 6:00 am according to the rule. Zone operative temperature swung from around 22.5°C early in the morning to a peak of 25.1°C at 5:00 pm. With the MPC controller, cooling was considered not necessary for the whole day and zone operative temperature was maintained within a 23.25°C range throughout the day.

Based on these two examples, MPC was able to make wise decisions on when to turn on/off zone level radiant systems to conserve energy and maintain thermal comfort, with the capability to use predictions of the cooling demand and the thermal response of individual zones.

6.5 Conclusion

In this chapter, we studied the control of the heavyweight radiant slab system for a typical office building. Model predictive control (MPC) was tested against a fine-tuned rule based heuristic control method for this complex control problem. A first-order dynamical model was developed for implementation in the model predictive controller and it was shown to be able to predict system performance reasonably well. The test was conducted for a summer season in a dry and hot climate and the MPC controller using the first-order system model was able to maintain zone operative temperatures at EN 15251 Category II thermal comfort level more than 95% of the occupied hours for all zones. With the heuristic method, only
the core zone operative temperatures were maintained at Category II level for more than 95% of the occupied hours; for the east zone, the number was only 88.3%. Compared to the heuristic method, MPC reduced the cooling tower energy consumption by 55% and pumping power consumption by 25%.
Figure 6.5: Comparison of Heuristic and MPC methods in control of zone operative temperature (A) and radiant loop valve (B): South zone on July-26th.
Bibliography


