Robust Constraint Satisfaction: Invariant Sets and Predictive Control

Eric C. Kerrigan
St John's College

Control Group
Department of Engineering
University of Cambridge

A dissertation submitted for
the degree of Doctor of Philosophy

November 2000

For my family

Abstract

Set invariance plays a fundamental role in the design of control systems for constrained systems since the constraints can be satisfied for all time if and only if the initial state is contained inside an invariant set. This thesis is concerned with robust set invariance theory and its application to guaranteeing feasibility in model predictive control.

In the first part of this thesis, some of the main ideas in set invariance theory are brought together and placed in a general, nonlinear setting. The key ingredients in computing robust controllable and invariant sets are identified and discussed. Following this, linear systems with parametric uncertainty and state disturbances are considered and algorithms for computing the respective robust controllable and invariant sets are described. In addition to discussing linear systems, an algorithm for computing the robust controllable sets for piecewise affine systems with state disturbances is described.

In the second part, the ideas from set invariance are applied to the problem of guaranteeing feasibility and robust constraint satisfaction in Model Predictive Control (MPC). A new sufficient condition is derived for guaranteeing feasibility of a given MPC scheme. The effect of the choice of horizons and constraints on the feasible set of the MPC controller is also investigated. Following this, a necessary and sufficient condition is derived for determining whether a given MPC controller is robustly feasible. The use of a robustness constraint for designing robust MPC controllers is discussed and it is shown how this proposed scheme can be used to guarantee robust constraint satisfaction for linear systems with parametric uncertainty and state disturbances. A new necessary and sufficient condition as well as some new sufficient conditions are derived for guaranteeing that the proposed MPC scheme is robustly feasible.

The third part of this thesis is concerned with recovering from constraint violations. An algorithm is presented for designing soft-constrained MPC controllers which guarantee constraint satisfaction, if possible. Finally, a mixed-integer programming approach is described for finding a solution which minimises the number of violations in a set of prioritised constraints.

Keywords: robust control, constrained systems, invariant sets, controllable sets, piecewise affine systems, predictive control, feasibility, exact penalty functions, multi-objective optimisation

Declaration

As required by the University Statute, I hereby declare that this dissertation is not substantially the same as any that I have submitted for a degree or diploma or other qualification at any other university. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration.

I also declare that the length of this thesis is less than 65,000 words and that the number of figures is less than 150.

Eric C. Kerrigan
St John's College
Cambridge
30 November 2000

Acknowledgements

This thesis is the product of my interaction with a large number of people, with whom I have had the pleasure to discuss a wide range of topics in control, engineering and mathematics.

I would first like to thank my supervisor, Dr Jan Maciejowski, for listening to my sometimes very scattered thoughts and helping me channel them in the right direction. Throughout my studies he took an active interest in my work and encouraged me to keep at it when I couldn't see the light at the end of the tunnel.

I would also like to thank Prof. Manfred Morari for allowing me to visit his group at ETH in Zürich during the previous two summers. These visits have allowed me to establish a great working relationship with the group. In particular, I would like to thank Dr Alberto Bemporad, Domenico Mignone and Fabio Torrisi for the many interesting and productive discussions we have had. Thanks also to Francesco Borrelli and Giancarlo Ferrari-Trecate for sharing their work with me.

My interaction with the members of the COSY (Control of Complex Systems) project has also been highly enjoyable. I would like to thank Dr Roozbeh Izadi-Zamanabadi and Prof. Mogens Blanke for involving me in the various activities of the fault-tolerant control group.

The Control Lab in Cambridge has been my home for the last three years and everybody who has passed through has contributed to my understanding of control systems in some way or another. Tim, thank you for helping me figure out what robust control is really all about.

I would also like to take this opportunity to thank the various bodies that have provided me with generous financial support during the course of my research. These are the Emmanuel Bradlow Foundation, Sir Henry Strakosch Memorial Trust, St John's College Benefactors' Fund, The First National Trust of South Africa, Cambridge University Engineering Department, The Royal Academy of Engineering and the European Science Foundation.

Finally, I would like to thank all the members of the Lady Margaret Boat Club for making my stay in Cambridge unforgettable.

Viva Laeta

Contents

List of Figures

Notation

Ω, Φ and Ψ will be used to denote arbitrary subsets of a given Euclidean space.

Given Ω, Ω° denotes its interior and Ω^c its complement.

The set $\Omega\backslash\Phi$ is the complement of Φ that is contained in Ω, i.e. $\Omega\backslash\Phi \triangleq \{x \in \mathbb{R}^n \mid x \in \Omega, x \notin \Phi\} = \Omega \cap \Phi^c$.

The notation $\Omega \subseteq \Phi$ is used to denote that Ω is a subset of Φ and $\Omega \subset \Phi$ denotes that Ω is a proper subset of Φ.

B_r is an open norm-ball of radius r, i.e. $B_r \triangleq \{x \mid \|x\| < r\}$.

$\mathbb{Z}$ is the set of integers, $\mathbb{N}$ is the set of non-negative integers and $\mathbb{N}_+ \triangleq \mathbb{N}\backslash\{0\}$. Similarly, $\mathbb{R}_+ \triangleq \mathbb{R}\backslash(-\infty, 0]$.

A sequence of states or control inputs is denoted by $\{x_k\}_0^P \triangleq \{x_0, x_1, \ldots, x_P\}$. The notation $\{x_k \in \Omega\}_0^P$ is used to indicate that each element of the sequence $\{x_k\}_0^P$ is an element of Ω.

Upper case will usually be used to denote matrices and lower case to denote vectors. If Q is a matrix, then Q_i is the i'th row of Q. If q is a vector, then q_i is the i'th component of q.

$Q \succeq 0$ if and only if $x'Qx \geq 0, \forall x$. $Q \succ 0$ if and only if $x'Qx > 0, \forall x \neq 0$.

$q \preceq s$ if and only if $q_i \leq s_i, \forall i$.

0_n is a vector of length n containing only zeros and $\mathbf{1}_n$ is a vector of length n containing only ones.

If $f : \mathbb{R}^m \mapsto \mathbb{R}^n$ and $\Omega \subset \mathbb{R}^m$, then $f(\Omega) \triangleq \{y \in \mathbb{R}^n \mid \exists x \in \Omega : y = f(x)\}$.

Chapter 1

Introduction

1.1 Motivation

Engineering, biological and economic systems can often be described in terms of mathematical models. These models help one to understand the behaviour of the system and how one could control the resources or inputs in order to affect the outputs of the system. However, real systems are highly complex and it is not possible to model every detail exactly. There is always some mismatch between the ideal mathematical description and the physical world.

The main aim of control theory is to exploit the phenomenon of feedback to allow for the uncertainty present in the mathematical model. The outputs of the actual system are compared with the predicted outputs of the mathematical system and the difference is fed back to a controller which changes the inputs to the system in an appropriate fashion. An aeroplane auto-pilot is an example of a controller which uses feedback to account for uncertainty. The flaps and elevators are used to compensate for any atmospheric disturbances in order to maintain a level and comfortable flight.

Most physical systems are complex and the requirements on the performance of the controller are usually quite demanding. It is the role of the engineer to design, within budget and available time, a controller which is guaranteed to meet the client's specifications during testing and commercialisation. This motivates the need for an effective, systematic method whereby a designer can use an approximate model of the system to design a controller which is guaranteed to work on the actual physical system.

All physical systems have inputs and outputs which are limited in size due to the presence of safety or physical constraints. Furthermore, an application might also require a certain level of performance, which can be translated into additional constraints on the controlled system.

Omitting these constraints in the controller design phase may lead to a control action that could result in the violation of these constraints. Depending on the criticality of the constraint this violation might result in system failure, which in turn could possibly lead to loss of human life.

Similarly, if the effect of the uncertainty in the model is not taken into account, then the actual and theoretical behaviour of the system will differ. It is possible that a controller which does not take account of uncertainty would drive the system into an unsafe region. A small disturbance or fault could then cause the system to break.

Given this need for designing safe controllers, this thesis concentrates on incorporating the effect of uncertainty on control systems and how to design controllers which will guarantee that the constraints will not be violated.

1.1.1 A Mathematical Framework and Computational Tools for Constrained Systems

The first part of this thesis is concerned with the development of a mathematical framework which incorporates both constraints and uncertainty in controller design. This framework brings together a number of ideas from the last thirty years and attempts to place them in a more general, modern context.

The main concept behind the framework is that before a controller can be designed, one needs to compute the largest ‘safe’ region in which the system should be kept. This region could be smaller than the pre-specified region defined by the safety and performance constraints. The reason for this is that the specified constraints do not necessarily take into account the actual physics of the process.

For example, the national speed limit for cars does not always take into account the conditions of the road and if one encounters a very sharp bend in the road then this limit might not be safe. Factors such as the age and technology used in the car, as well as the driver’s experience place a practical limit on the speed and angle with which the car can approach the bend. A more experienced driver can be thought of as a well-designed controller that incorporates both a knowledge of the physics of the system and an understanding of the effects of disturbances on the system.

A theoretical framework is not very useful unless it can be implemented for practical systems. Various algorithms have been proposed for computing the safe regions for uncertain linear systems subject to disturbances. However, even though linear systems are quite simple, many of the proposed algorithms require large amounts of computational power. This thesis is therefore also concerned with the presentation of some slightly more efficient algorithms which might allow the computation of safe regions for larger and more complex systems, such as hybrid systems.

Many real-life systems are hybrid in nature. The term ‘hybrid’ as used here is meant to describe systems whose inputs and states can take on discrete and continuous values. An example of a system which only takes on discrete values is a light switch, the state being either on or off. A bathtub is an example of a system with a continuous state, where the water level can take on any value between empty and full.

Strictly speaking, though, a bath could be thought of as a hybrid system if one includes the state of the plug in the system description. If the plug has been removed then the water will drain, with the water level dropping continuously until empty. Replacing the plug will stop the bath from draining. The bath can then be topped up at a desired rate by setting the position of the tap anywhere between shut and fully open. Depending on the flow rate and the state of the plug, the bath will either drain, remain at the same level or fill up.

The presence of discrete inputs and states complicates the computation of the safe regions of a controlled system. However, it is possible to extend the algorithms developed for continuous systems to a large class of hybrid systems. In this thesis it is shown how to compute the corresponding safe regions for the class of hybrid systems which can be modelled as piecewise affine systems.

1.1.2 Feasibility in Model Predictive Control

Model Predictive Control (MPC) is one of the most popular advanced control techniques in industry, mainly due to the ease with which constraints can be included in the controller formulation. Though highly successful in practice, a large number of properties of MPC are not well understood. One of the most fundamental problems in MPC is that of guaranteeing constraint satisfaction in the presence of uncertainty.

Furthermore, current industrial implementations of MPC do not explicitly take into account the effect of uncertainty or disturbances on the future evolution of the system. As a result, constraint satisfaction cannot be guaranteed.

The mathematical framework and computational tools developed during the first part of this thesis allow one to develop new theoretical conditions and tools for guaranteeing constraint satisfaction in MPC. The framework allows one to develop design methods for implementing MPC controllers which are robust to a pre-specified level of uncertainty. Constraint satisfaction can be guaranteed by computing a safe region and including it in the design of the MPC controller. The controller then only selects control inputs for which the predicted response will remain within this safe region.

1.1.3 Recovering From Constraint Violations

If the system constraints cannot be satisfied, then a control action has to be computed which ensures that the least damaging course of action is taken. This is further complicated by the fact that it is often possible to prioritise the constraints and objectives. For example, it is more important to satisfy a safety constraint rather than a performance constraint. Consequently, a control action which satisfies the safety constraint is preferred. The last part of this thesis focuses on methods for computing a control action which satisfies as many of the constraints as possible, while taking the priorities into account.

1.2 Organisation and Highlights of this Dissertation

This dissertation is organised as follows:

Chapter 2: Robust Set Invariance Theory

Set invariance is a fundamental concept in the design of controllers for constrained systems. The reason for this is because constraint satisfaction can be guaranteed for all time and for all disturbances if and only if the initial state is contained inside a robust control invariant set. This chapter aims to bring together some of the important ideas from set invariance theory that have been developed during the course of modern control theory. The results in subsequent chapters are based on this set-theoretic framework. The unifying concept in the chapter is that many of the described sets are special cases of the so-called “robust controllable sets” and can be computed using Algorithm 2.1.

Chapter 3: Uncertain Linear Time-Invariant Systems

If the constraints on the system are given by convex polyhedra and the system is linear and time-invariant, then it is possible to compute all of the sets defined in Chapter 2. One can compute the robust controllable sets not only if there are state disturbances, but also if there is parametric uncertainty present in the model. The idea of contractive sets are introduced in Section 3.2 and Theorem 3.1 gives a guarantee that, for uncertain LTI systems, a robust control invariant set can be computed in a finite number of iterations of Algorithm 2.1.

Standard algorithms for computing the robust one-step set such as projection and Minkowski summation are briefly described in Section 3.3. Section 3.4 presents a result that allows one to compute the linear map of a polyhedron in polynomial time and derive an upper bound on the number of faces of the resultant polyhedron.

Chapter 4: Robust Controllable Sets for Hybrid and Piecewise Affine Systems

One of the classes of systems that has recently been receiving a lot of interest in the control literature is hybrid systems. The Mixed Logic Dynamical (MLD) modelling framework of [BM99a] is briefly introduced in Section 4.2. The motivation for the introduction of MLD systems is that hybrid systems which can be modelled using the MLD framework have been shown to be formally equivalent to piecewise affine (PWA) systems [BFM00]. The main aim of this chapter is to describe how one would proceed in computing the robust controllable sets for these PWA systems. The main building block for the proposed algorithm is the development in Section 4.5.1 of a method for computing the Pontryagin difference between a non-convex polygon and a convex polyhedron. Section 4.5.2 shows how the results from Chapters 2 and 3 can be used to complete the computation of the robust controllable set once the Pontryagin difference has been found.

Chapter 5: Nominal Feasibility in Model Predictive Control

Due to the finite horizon nature of Model Predictive Control (MPC), feasibility for all time cannot be guaranteed in general, even if there are no disturbances present. This chapter is concerned only with the nominal case where there are no disturbances and no model mismatch.

The concept of "strong feasibility" is introduced in Section 5.4. An MPC problem is said to be strongly feasible if and only if it is feasible for all time, even if the computed solution is sub-optimal. One way of guaranteeing that an MPC problem is strongly feasible is to add a control invariant terminal constraint to the original problem. Theorem 5.2 gives a new sufficient condition on the feasible set of the MPC problem such that strong feasibility is guaranteed, even if the terminal constraint is not control invariant. The terminal constraint set condition can then be shown to be a special case of this new condition.

Sections 5.7 and 5.8 bring together many of the results on the behaviour of the feasible set and the feasibility of the MPC problem for different choices of horizons and terminal constraint set. In particular, Theorem 5.3 implies that if the terminal constraint is not used and the control and prediction horizons are chosen to be equal to one another, then strong feasibility is possible if and only if there exists a finite control horizon such that the feasible set is control invariant.

Chapter 6: Robust Feasibility in Model Predictive Control

This chapter introduces the notion of "robust strong feasibility" in MPC in order to guarantee a feasible MPC problem for all time, despite the presence of disturbances. A new condition which is both necessary and sufficient for an MPC scheme to be robust strongly feasible is given by Theorem 6.1. If the MPC controller satisfies this condition, then no modifications need to be made to the original scheme of Chapter 5 in order to guarantee strong robust feasibility.

However, sometimes the nominal MPC scheme is not robust strongly feasible for any size of disturbance and it is therefore necessary to modify the original scheme. Section 6.4 suggests the addition of a "robustness constraint" to the nominal MPC problem, as proposed in [CZ99], in order to robustify the original MPC controller against disturbances. Theorem 6.3 gives a new necessary and sufficient condition and Theorem 6.4 contains a number of new sufficient conditions for the proposed scheme to be robust strongly feasible. Section 6.5 shows that the robustness constraint approach can be used to design robust strongly feasible MPC controllers for LTI systems with state disturbances and model uncertainty.

Often the most economic setpoint for a process is on or close to the constraints. It is not always desirable to regulate the system close to the constraints, since a disturbance could result in a violation of a safety constraint. As a result, the setpoint is often chosen to be a safe distance away from the constraints. Section 6.8 discusses how to compute a setpoint which is as close as possible to the desired reference, while being compatible with the constraints and bearing in mind that there are unknown, but bounded disturbances on the state and output.

Chapter 7: Soft Constraints and Exact Penalty Functions

Often a disturbance comes along which makes the violation of the constraints unavoidable, resulting in an infeasible MPC problem. The ability to recover from infeasibility is often implemented via the use of soft constraints. In addition, it is desirable that the solution to the soft-constrained MPC problem be equal to the solution of the original, hard-constrained MPC controller if the latter would have been feasible.

The theory of exact penalty functions allows one to derive a condition on the weight used to penalise the constraint violations in order to guarantee the equality of the solutions to the two problems. The lower bound for this weight is related to the norm of the Lagrange multipliers of the solution to the original, hard-constrained problem. The problem is complicated in MPC by the fact that the Lagrange multipliers are dependent on the current state and the Lagrange multipliers need to be computed for all states in the feasible set of the hard-constrained problem. Section 7.5 gives an algorithm based on the explicit solution of the MPC control law for computing a lower bound on the penalty weight.

Chapter 8: Optimisation Subject to Prioritised Constraints

Often constraints can be prioritised and when constraint violation is inevitable, the control law has to take this into account. A control action which results in the violation of the lower-prioritised constraints is preferred. The recovery from constraint violation can therefore be interpreted as a prioritised, multi-objective optimisation problem. The main result of this chapter is Theorem 8.1 which gives a condition on the cost function of a mixed-integer optimisation problem such that the solution is guaranteed to be a prioritised-optimal solution to a multi-objective optimisation problem.

This result is then applied in Theorem 8.2 which shows how a single mixed-integer program can be set up such that the number of constraint violations are minimised in a prioritised fashion. The same idea is applied in Theorem 8.3 for the computation of a minimum-time, output-prioritised MPC control law for hybrid systems which can be modelled in MLD form.

Chapter 9: Concluding Remarks

This chapter summarises the contributions made by this thesis and outlines directions for future research.

Appendices

Appendix A briefly describes how the ideas from Chapter 2 need to be adapted to compute robust controllable sets for time-varying systems.

Appendix B describes a simple algorithm for the removal of redundant inequalities from the description of a convex polyhedron.

Appendix C outlines the idea behind the process of eliminating variables from a set of inequalities using Fourier elimination. Fourier elimination can be used to compute the projection of a convex polyhedron onto a subspace.

Appendix D describes the principles behind an algorithm for computing the complement of a set given by the union of convex polyhedra.

Appendix E gives a list of the functions in a Matlab toolbox which has been developed for the computation of the various sets described in this thesis.

Part I

Set Invariance Theory for Discrete-time Systems

Chapter 2

Robust Set Invariance Theory

The concept of invariant and robust controllable sets and their role in the control of constrained systems are introduced. Some set-theoretic results are given which will be useful in developing algorithms for computing such sets.

2.1 Introduction

A fundamental control problem is that of determining the subset of the state space which can be steered via an admissible control sequence to any given target set, while guaranteeing that the state constraints will be satisfied for all allowable disturbance sequences. This is a more general interpretation of the classical reachability and controllability problems of linear, unconstrained systems.

The problem of steering a system to a target set in the presence of input constraints and a bounded disturbance was considered relatively early in modern control literature. In [DM69], very general results are given for determining whether it is possible to steer a system to a given target set, despite the presence of disturbances. The target set was said to be "strongly reachable" from a given state if such a control existed.

The problem of steering a time-varying nonlinear system of the form

$$x_{k+1} = f_k(x_k, u_k) + g_k(w_k)$$

with time-varying constraints on the input, state and disturbance to a target set in a finite number of steps is discussed in [BR71]. The problems considered are described as the "reachability of a target set" and the "reachability of a target tube". The results are once again very general and some compactness results are given for linear time-varying systems. The problem of imperfect state observation is also discussed. Similar results as in [BR71] are reported in [GS71], where the results are applied to the synthesis of controllers for linear time-varying systems with time-varying constraints.

One of the most influential recent papers is [Bla94]. The idea of contractive sets is introduced and the case of LTI systems with polytopic uncertainty and bounded disturbances on the state are considered.

A thorough discussion of LTI systems in closed-loop with a linear feedback control law and bounded disturbances on the state is given in [KG98], where the work of [GT91] is extended for deriving practical results on the computation of the “maximal output admissible set”.

A very comprehensive survey of papers on set invariance is given in [Bla99]. This chapter does not attempt to duplicate the discussion in the survey, but aims to consolidate some of the generality of the work of the early researchers with more recent ideas, terminology and notation.

2.1.1 Nonlinear Discrete-Time Systems Subject to State and Input Constraints

The discussion in this chapter assumes the following uncertain, discrete-time dynamic system:

$$x_{k+1} = f(x_k, u_k, w_k) \tag{2.1}$$

where $k \in \mathbb{Z}$, x_k is the system state, u_k is the control input and

$$w_k \in \mathbb{W} \subset \mathbb{R}^d$$

is an unknown disturbance. If the system does not have a control input or there is no disturbance, then with a slight abuse of notation $x_{k+1} = f(x_k, w_k)$ or $x_{k+1} = f(x_k, u_k)$ will be used to denote this.

The system is subject to pointwise-in-time constraints on the control inputs and/or the states:

$$u_k \in \mathbb{U} \subset \mathbb{R}^m \tag{2.2a}$$

$$x_k \in \mathbb{X} \subseteq \mathbb{R}^n \tag{2.2b}$$

The set $\mathbb{U}$ is compact, while $\mathbb{X}$ and $\mathbb{W}$ are closed. It is assumed that the system and constraints are time-invariant[1]. The system $f(x_k, u_k, w_k)$ is uniquely defined over $\mathbb{X} \times \mathbb{U} \times \mathbb{W}$. Exact state measurement is available.

An *admissible control input, sequence* or *law* is one which satisfies the input constraints $\mathbb{U}$. The elements of an *allowable disturbance sequence* are contained in $\mathbb{W}$. From this point on, it is understood that the control law and states are subject to the constraints in (2.2) and that the disturbance sequence is allowable.

2.1.2 Distinguishing Between the Nominal and Robust Sets

If there is a disturbance present and the calculation of the resulting set took this fact into account, a tilde and the word “robust” will be used to indicate this, e.g. $\tilde{\mathcal{C}}_\infty(\mathbb{X})$ is the maximal robust control invariant set. If there is no disturbance or the disturbance is ignored in the calculation of the set,

[1]Appendix A gives a brief discussion on how the algorithms can be modified in order to account for a time-varying system with time-varying constraints.

i.e. $x_{k+1} = f(x_k, u_k)$ was used as the system model, then the use of the tilde and "robust" will be dropped, e.g. $\mathcal{C}_\infty(\mathbb{X})$ is the maximal control invariant set. The latter case with no disturbance will also be referred to as the *nominal* case.

This chapter is largely an extension of the definitions for invariant sets in [KM00a, Sect. 2] to the more general case of including a disturbance in the system description. The notation and results in this chapter are consistent with [KM00a].

2.2 Input and Output Admissible Sets

It is of interest to determine which subset of a given set is compatible with the input and output constraints. This section defines the concept of the input and output admissible sets.

If the system is in closed-loop with the control law[2]

$$u_k = h(x_k)\,,$$

then a superscript will be used to emphasise this fact. The input admissible set is the subset of a given Ω in which the control law satisfies the input constraints.

Definition 2.1 (Input admissible set). Given a control law $u_k = h(x_k)$, the *input admissible* subset of $\Omega \subseteq \mathbb{R}^n$ is given by

$$\Omega^h \triangleq \{x_k \in \Omega \mid h(x_k) \in \mathbb{U}\}\,. \tag{2.3}$$

The closed-loop system is then given by

$$x_{k+1} = f(x_k, h(x_k), w_k)$$

and the constraints on the state can be replaced by

$$x_k \in \mathbb{X}^h \triangleq \{x_k \in \mathbb{X} \mid h(x_k) \in \mathbb{U}\}\,.$$

Statements about systems *without* control inputs will also apply to closed-loop systems, bearing in mind that the state constraints should be replaced by the input admissible subset, where necessary.

If the disturbance acts directly on the input of the system it is usually possible to redefine the system such that the disturbance acts directly on the state of the system. It is therefore assumed that the disturbance does not act on the input.

[2]Note that if the constraints on the input $\mathbb{U}$ are given as a hyper-rectangle and the control law is given by an appropriate saturation function $u_k = \text{sat}(\cdot)$, as is often the case, then $\Omega^h = \Omega$ (provided the control law is defined over Ω). If the system is LTI and $u_k = \text{sat}(Kx_k)$, where $K \in \mathbb{R}^{m\times n}$, then the resulting closed-loop system can be treated as a piecewise affine system, as in Chapter 4.

In a similar fashion as with (2.3), if the output constraints of the system can be given by

$$y_k = \phi(x_k, w_k) \in \mathbb{Y} \subseteq \mathbb{R}^p \,, \tag{2.4}$$

then one can define the output admissible subset[3] of Ω.

Definition 2.2 (Output admissible set). If the output constraints on the system are given by (2.4), then the *output admissible* subset of $\Omega \subseteq \mathbb{R}^n$ is given by

$$\Omega^{\phi} \triangleq \{x_k \in \Omega \mid \phi(x_k, w_k) \in \mathbb{Y}, \forall w_k \in \mathbb{W}\} \,. \tag{2.5}$$

The subset of Ω which is both input- and output admissible is therefore given by

$$\Omega^{h,\phi} \triangleq \Omega^h \cap \Omega^{\phi} \,.$$

Note that the constraints (2.2) can be modified by replacing $\mathbb{X}$ with $\mathbb{X}^{\phi}$ or $\mathbb{X}^{h,\phi}$ in all calculations. The case of a system with constraints on the output therefore reduces to a problem with modified constraints on the state. Output constraints will not be considered as a separate case[4].

2.3 Robust One-step Set and Reach Set

There are two sets which are used throughout the controllability and reachability analysis of systems. The first set that will be introduced is the robust one-step set.

Definition 2.3 (The robust one-step set $\tilde{\mathcal{Q}}(\Omega)$). [Bla94] The set $\tilde{\mathcal{Q}}(\Omega)$ is the set of states in $\mathbb{R}^n$ for which an admissible control input exists which will guarantee that the system will be driven to Ω in one step, for all allowable disturbances, i.e.

$$\tilde{\mathcal{Q}}(\Omega) \triangleq \{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : f(x_k, u_k, w_k) \in \Omega, \forall w_k \in \mathbb{W}\} \,. \tag{2.6}$$

For closed-loop systems, $\tilde{\mathcal{Q}}^h(\Omega)$ is the set of states in $\mathbb{R}^n$ from which the system is guaranteed to evolve to Ω at the next time instant, given any allowable disturbance, i.e.

$$\tilde{\mathcal{Q}}^h(\Omega) \triangleq \{x_k \in \mathbb{R}^n \mid f(x_k, h(x_k), w_k) \in \Omega, \forall w_k \in \mathbb{W}\} \,.$$

An alternative, equivalent definition of the robust one-step set is

$$\tilde{\mathcal{Q}}(\Omega) \triangleq \{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : f(x_k, u_k, \mathbb{W}) \subseteq \Omega\} \,.$$

[3]Note that this definition of the output admissible set is different from the one given in [GT91, KG95, KG98]. The definition of the latter includes the additional condition that the output admissible subset be a robust positively invariant set.

[4]If the output is of the form $y_k = \phi_x(x_k) + \phi_w(w_k)$, then it is possible to compute the output admissible subset of $\mathbb{X}$ as the Pontryagin difference between $\{x_k \in \mathbb{X} \mid \phi_x(x_k) \in \mathbb{Y}\}$ and $\phi_w(\mathbb{W})$. The Pontryagin difference is discussed in Section 2.10.1.

Proposition 2.1. *For all* Ω_1, Ω_2,

$$\Omega_1 \subseteq \Omega_2 \Rightarrow \tilde{\mathcal{Q}}(\Omega_1) \subseteq \tilde{\mathcal{Q}}(\Omega_2)\,.$$

Proof. This proof is similar to the proof in [VSLS99, Prop. 2], where the operator $\mathrm{Pre}(\Omega) = \tilde{\mathcal{Q}}(\Omega) \cap \Omega$ is defined. The results follows from the fact that $\forall x_k \in \tilde{\mathcal{Q}}(\Omega_1)$, $\exists u_k \in \mathbb{U}$ such that $x_{k+1} \in \Omega_1, \forall w_k \in \mathbb{W}$. Since $\Omega_1 \subseteq \Omega_2$ it follows that the same u_k results in $x_{k+1} \in \Omega_2, \forall w_k \in \mathbb{W}$, therefore $x_k \in \tilde{\mathcal{Q}}(\Omega_2)$. □

The next set to be introduced is the reach set.

Definition 2.4 (The reach set $\tilde{\mathcal{R}}(\Omega)$). The set $\tilde{\mathcal{R}}(\Omega)$ is the set of states in $\mathbb{R}^n$ to which the system will evolve at the next time step given any $x_k \in \Omega$, admissible control input and allowable disturbance, i.e.

$$\tilde{\mathcal{R}}(\Omega) \triangleq \{x_{k+1} \in \mathbb{R}^n \mid \exists x_k \in \Omega, u_k \in \mathbb{U}, w_k \in \mathbb{W} : x_{k+1} = f(x_k, u_k, w_k)\}\,. \tag{2.7a}$$

For closed-loop systems $\tilde{\mathcal{R}}^h(\Omega)$ is the set of states in $\mathbb{R}^n$ to which the system will evolve at the next time step given any $x_k \in \Omega$ and allowable disturbance, i.e.

$$\tilde{\mathcal{R}}^h(\Omega) \triangleq \{x_{k+1} \in \mathbb{R}^n \mid \exists x_k \in \Omega, w_k \in \mathbb{W} : x_{k+1} = f(x_k, h(x_k), w_k)\}\,. \tag{2.7b}$$

An alternative, equivalent definition of the reach set is

$$\tilde{\mathcal{R}}(\Omega) \triangleq \{x_{k+1} \in \mathbb{R}^n \mid x_{k+1} \in f(\Omega, \mathbb{U}, \mathbb{W})\}\,.$$

Remark 2.1. If no disturbance is present in the model, then

$$\mathcal{Q}(\Omega) \triangleq \{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : f(x_k, u_k) \in \Omega\}$$

and

$$\mathcal{R}(\Omega) \triangleq \{x_{k+1} \in \mathbb{R}^n \mid \exists x_k \in \Omega, u_k \in \mathbb{U} : x_{k+1} = f(x_k, u_k)\}\,.$$

The reach set as defined in (2.7a) is not always practically useful. The reach set with no disturbances $\mathcal{R}(\Omega)$ or the reach set of a closed-loop system $\tilde{\mathcal{R}}^h(\Omega)$ are used more often.

Proposition 2.2. *If Ω is given by the union*

$$\Omega \triangleq \bigcup_i \Omega_i\,, \tag{2.8}$$

then

$$\mathcal{Q}(\Omega) = \bigcup_i \mathcal{Q}(\Omega_i)\,. \tag{2.9}$$

Proof. If $x_k \in \mathcal{Q}(\Omega)$, then there exists a $u_k \in \mathbb{U}$ such that $x_{k+1} \in \Omega$. But $x_{k+1} \in \Omega_i$ for some i, therefore, $x_k \in \mathcal{Q}(\Omega_i)$ for some i, hence $x_k \in \bigcup_i \mathcal{Q}(\Omega_i)$. This proves that $\mathcal{Q}(\Omega) \subseteq \bigcup_i \mathcal{Q}(\Omega_i)$.

If $x_k \in \bigcup_i \mathcal{Q}(\Omega_i)$, then there exists a $u_k \in \mathbb{U}$ such that $x_{k+1} \in \Omega_i$ for some i. But $\Omega_i \subseteq \Omega$, therefore $x_{k+1} \in \Omega$ and hence $x_k \in \mathcal{Q}(\Omega)$. This proves that $\mathcal{Q}(\Omega) \supseteq \bigcup_i \mathcal{Q}(\Omega_i)$. □

Remark 2.2. It is important to recognise that the one-step set and the reach set operate in different directions. The one-step set is the set of states *from* which the system can be driven to a given set. The reach set is the set of states *to* which the system can be driven from a given set. No explicit relation exists between the two sets.

2.4 Robust Positively Invariant Sets

Given a set Ω and an initial state $x_0 \in \Omega$, it is of interest to determine whether the evolution of the system will remain inside the set for all time, despite the presence of disturbances.

Definition 2.5 (Robust positively invariant set). [Bla99] The set $\Omega \subset \mathbb{R}^n$ is *robust positively invariant* for the system $x_{k+1} = f(x_k, w_k)$ if and only if $\forall x_0 \in \Omega$ and $\forall w_k \in \mathbb{W}$, the system evolution satisfies $x_k \in \Omega, \forall k \in \mathbb{N}$.

In other words, Ω is robust positively invariant if and only if

$$x_k \in \Omega \Rightarrow x_{k+1} \in \Omega, \forall w_k \in \mathbb{W}\,.$$

The following result follows immediately from the definition.

Proposition 2.3. *The union of two robust positively invariant sets is robust positively invariant.*

Remark 2.3. The same statement cannot be made about the intersection of two robust positively invariant sets, even in the absence of disturbances.

In general, a given set Ω is not robust positively invariant. However, often one would like to determine the largest robust positively invariant set contained in Ω.

Definition 2.6 (Maximal robust positively invariant set). The set $\tilde{\mathcal{O}}_\infty(\Omega)$ is the *maximal robust positively invariant set* contained in Ω for the system $x_{k+1} = f(x_k, w_k)$ if and only if $\tilde{\mathcal{O}}_\infty(\Omega)$ is robust positively invariant and contains all the robust positively invariant sets contained in Ω.

Remark 2.4. It can be shown that the maximal robust positively invariant set is unique.

This definition implies that a set Φ is robust positively invariant only if

$$\Phi \subseteq \tilde{\mathcal{O}}_\infty(\Omega) \subseteq \Omega\,.$$

Remark 2.5. Based on the discussion in Section 2.2, the maximal robust positively invariant set[5] for the *closed-loop* system $x_{k+1} = f(x_k, h(x_k), w_k)$ will be denoted by $\tilde{\mathcal{O}}^h_\infty(\Omega)$ and is defined as the maximal robust positively invariant set contained in the input admissible set Ω^h, i.e. $\tilde{\mathcal{O}}^h_\infty(\Omega) \triangleq \tilde{\mathcal{O}}_\infty(\Omega^h)$ for the system $x_{k+1} = f(x_k, h(x_k), w_k)$.

2.5 Robust Control Invariant Sets

In a similar fashion as with robust positively invariant sets, one would like to determine whether given a set Ω and an initial state $x_0 \in \Omega$, it is possible to choose a control law such that the state evolution remains in Ω for all time, despite the presence of disturbances.

Definition 2.7 (Robust control invariant set). [Bla99] The set $\Omega \subset \mathbb{R}^n$ is a *robust control invariant* set for the system $x_{k+1} = f(x_k, u_k, w_k)$ if and only if there exists a feedback control law $u_k = h(x_k)$ such that Ω is a robust positively invariant set for the closed-loop system $x_{k+1} = f(x_k, h(x_k), w_k)$ *and* $u_k \in \mathbb{U}, \forall x_k \in \Omega$.

In other words, a set Ω is robust control invariant if and only if

$$x_k \in \Omega \Rightarrow \exists u_k \in \mathbb{U} : x_{k+1} \in \Omega, \forall w_k \in \mathbb{W}\,.$$

The following result is a direct consequence of the above definition.

Proposition 2.4. *The union of two robust control invariant sets is robust control invariant.*

Remark 2.6. The same statement cannot be made about the intersection of two robust control invariant sets, even in the absence of disturbances.

In general, a given set Ω is not robust control invariant. However, often one would like to determine the largest robust control invariant set[6] contained in Ω.

Definition 2.8 (Maximal robust control invariant set). [Bla94] The set $\tilde{\mathcal{C}}_\infty(\Omega)$ is the *maximal robust control invariant set* contained in Ω for the system $x_{k+1} = f(x_k, u_k, w_k)$ if and only if $\tilde{\mathcal{C}}_\infty(\Omega)$ is robust control invariant and contains all the robust control invariant sets contained in Ω.

Remark 2.7. As with the maximal robust positively invariant set, it can be shown that the maximal robust control invariant set is unique.

It is obvious that Φ is robust control invariant only if

$$\Phi \subseteq \tilde{\mathcal{C}}_\infty(\Omega) \subseteq \Omega\,.$$

[5]This definition for the maximal robust positively invariant set is analogous to the definition of the "maximal d-invariant set" for LTI systems with no control input as given in [KG95, KG98]. The maximal d-invariant set is the extension of the "maximal output admissible set" of [GT91] to the case with bounded state disturbances.

[6]A conceptual algorithm for computing the maximal robust control invariant set is given by Algorithm 2.3.

The following result follows immediately from the definitions and is the reason why invariant set theory plays a fundamental role in the study of constrained systems.

Proposition 2.5. *Given the uncertain system* (2.1), *there exists an admissible control law such that the state constraints* (2.2b) *can be satisfied for all time $k \in \mathbb{N}$ and for all allowable disturbance sequences if and only if the initial state $x_0 \in \tilde{\mathcal{C}}_\infty(\mathbb{X}) \subseteq \mathbb{X}$.*

Remark 2.8. An equivalent statement regarding closed-loop systems and the corresponding maximal robust positively invariant set $\tilde{\mathcal{O}}^h_\infty(\mathbb{X})$ can be made.

The following is an important, well-known geometric condition for a set to be control invariant and is used throughout the thesis in the derivation of many of the results.

Theorem 2.1 (Geometric condition for invariance). *[DH99] The set $\Omega \subset \mathbb{R}^n$ is a robust control invariant set[7] if and only if $\Omega \subseteq \tilde{\mathcal{Q}}(\Omega)$.*

Proof. Proving the contrapositive for both the necessary and sufficient parts: ($\Rightarrow$) If $\Omega \nsubseteq \tilde{\mathcal{Q}}(\Omega)$ then $\exists x_k \in \Omega$ which is not an element of $\tilde{\mathcal{Q}}(\Omega)$, i.e. $\forall x_k \in \Omega \backslash \tilde{\mathcal{Q}}(\Omega)$, $\nexists u_k \in \mathbb{U}$ such that $x_{k+1} \in \Omega, \forall w_k \in \mathbb{W}$, hence Ω is not a robust control invariant set. ($\Leftarrow$) If Ω is not a robust control invariant set then $\exists x_k \in \Omega$ for which $\nexists u_k \in \mathbb{U}$ such that $x_{k+1} \in \Omega, \forall w_k \in \mathbb{W}$, i.e. $\exists x_k \in \Omega$ which is not an element of $\tilde{\mathcal{Q}}(\Omega)$, hence $\Omega \nsubseteq \tilde{\mathcal{Q}}(\Omega)$. □

It follows immediately that the set Ω is robust control invariant if and only if $\tilde{\mathcal{Q}}(\Omega) \cap \Omega = \Omega$, since

$$\tilde{\mathcal{Q}}(\Omega) \cap \Omega = \Omega \Leftrightarrow \Omega \subseteq \tilde{\mathcal{Q}}(\Omega) .$$

Most algorithms which test whether a given set Ω is robust control invariant is based directly or indirectly on Theorem 2.1. Testing for invariance can be summarised as follows:

1. Compute $\tilde{\mathcal{Q}}(\Omega)$;
2. Test whether $\Omega \subseteq \tilde{\mathcal{Q}}(\Omega)$;
3. If $\Omega \subseteq \tilde{\mathcal{Q}}(\Omega)$, then Ω is robust control invariant. If $\Omega \nsubseteq \tilde{\mathcal{Q}}(\Omega)$, then Ω is not robust control invariant.

2.6 Robust Controllable Sets

The problem of finding a control law such that a target set is reached in a finite number of steps, despite disturbances on the state, is fundamentally linked with the problem of finding the *robust controllable sets*[8].

[7] The same statement holds for robust positively invariant sets.

[8] The concept of a robust controllable set is equivalent to the "reachability of a target tube" of [BR71, GS71]. However, in this paper the word *controllable* is used to define these sets and distinguish them from the *reachable* sets as defined in [Las93]. The use of *controllable* and *reachable* as used in this thesis is more consistent with modern control literature.

Definition 2.9 (Robust controllable set). The *i-step robust controllable set* $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$ is the largest set of states in Ω for which there exists an admissible *time-varying* state feedback control law such that an arbitrary terminal set $\mathbb{T} \subset \mathbb{R}^n$ is reached in *exactly i* steps, while keeping the evolution of the state inside Ω for the first $i - 1$ steps, for all allowable disturbance sequences, i.e.

$$\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \triangleq \{x_0 \in \mathbb{R}^n \mid \exists \{u_k = h_k(x_k) \in \mathbb{U}\}_0^{i-1} : \{x_k \in \Omega\}_0^{i-1}, x_i \in \mathbb{T}, \forall \{w_k \in \mathbb{W}\}_0^{i-1}\}. \tag{2.10a}$$

Given a suitable topology such as the Hausdorff topology, the limit, if it exists, defines the *infinite-time robust controllable set*:

$$\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T}) \triangleq \lim_{i\to\infty} \tilde{\mathcal{K}}_i(\Omega, \mathbb{T}). \tag{2.10b}$$

Remark 2.9. The definition here is very subtle and should not be misinterpreted. One is interested in finding the largest set of initial states for which there exists a *time-varying feedback law* which will ensure that the states of the closed-loop system reach the target set for all allowable disturbance sequences. This definition includes the more conservative problem of finding the set of states for which the same *open-loop sequence* will drive the system to the target set irrespective of which disturbance sequence occurs. The latter problem would have the definition

$$\tilde{\mathcal{K}}_i^{ol}(\Omega, \mathbb{T}) \triangleq \left\{x_0 \in \mathbb{R}^n \mid \exists \{u_k \in \mathbb{U}\}_0^{i-1} : \{x_k \in \Omega\}_0^{i-1}, x_i \in \mathbb{T}, \forall \{w_k \in \mathbb{W}\}_0^{i-1}\right\}.$$

By including the constraint that the control input be dependent on the state as well as time, a fundamentally different set results. A better understanding of this problem can be gained in studying the difference between “open-loop” and “feedback” robust MPC, as discussed in Section 6.3 and [MRRS00, Sect. 4].

Remark 2.10. It is interesting to observe that if there are no disturbances present, then

$$\mathcal{K}_i(\Omega, \mathbb{T}) = \mathcal{K}_i^{ol}(\Omega, \mathbb{T}).$$

Remark 2.11. It can be shown that if $\mathbb{T}$ is robust control invariant, then a *time-invariant* feedback control law will also ensure that the state is in $\mathbb{T}$ after exactly i steps. This follows from the fact that by definition a time-invariant control law can be chosen such that $\mathbb{T}$ is robust positively invariant for the resulting closed-loop system. A time-invariant control law can then be chosen such that the system enters $\mathbb{T}$ in the minimum amount of time.

By noting that

$$\tilde{\mathcal{K}}_1(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}(\mathbb{T}) \cap \Omega$$

one can proceed to develop a conceptual algorithm for computing robust controllable sets.

Algorithm 2.1 (Robust controllable sets). *[BR71] The robust controllable sets of a system can be computed via the following iterative procedure:*

$$\tilde{\mathcal{K}}_0(\Omega, \mathbb{T}) = \mathbb{T} \tag{2.11a}$$

$$\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}\left(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})\right) \cap \Omega. \tag{2.11b}$$

If $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$*, then terminate the algorithm and set* $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$.

The main procedures required to implement Algorithm 2.1 are:

1. Computation of the robust one-step set $\tilde{\mathcal{Q}}(\cdot)$;
2. Computation of the intersection $\tilde{\mathcal{Q}}(\cdot) \cap \Omega$;
3. Testing for the set equality $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$.

These three operations are easily implemented for LTI systems subject to linear inequality constraints [Bla94, DH99, GT91, KG87, KG98] and will be discussed in more detail in Chapter 3. In [VSLS99] quantifier elimination is proposed to compute the robust one-step set. Quantifier elimination can also be used, for example, when the constraints are defined by polynomials.

Though the conceptual algorithm presented here is difficult to implement for general nonlinear systems, there exist some classes of nonlinear systems for which the building blocks already are in place, such as piecewise affine systems and some classes of hybrid systems [BTM00a]. Some work on developing algorithms for computing robust control invariant sets for hybrid systems has also been carried out by the authors of [VSLS99]. A routine for computing the robust controllable sets for piecewise affine systems is described in Chapter 4.

The following definition is adapted from [GT91] and is the basis of the termination criterion in Algorithm 2.1.

Definition 2.10 (Finitely determined set). The set $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T})$ is *finitely determined* if and only if $\exists i \in \mathbb{N}$ such that $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$. The smallest element $i^* \in \mathbb{N}$ such that $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_{i^*}(\Omega, \mathbb{T})$ is called the *determinedness index*.

This definition will play an important role in Chapter 5 in obtaining results on the size and invariance properties of the feasible set of an MPC controller.

In general, $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T})$ is not finitely determined. However, a sufficient condition for the finite-determinedness of the infinite-time robust controllable set is:

Lemma 2.1. *If* $\exists i \in \mathbb{N}$ *such that* $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})$ *then* $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T})$ *is finitely determined. Furthermore,* $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T})$ *is robust control invariant.*

Proof. If $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})$, then by construction $\tilde{\mathcal{K}}_{i+2}(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})) \cap \Omega = \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) \cap \Omega$. However, $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) \cap \Omega$, hence $\tilde{\mathcal{K}}_{i+2}(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$. This continues *ad infinitum*, hence $\tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$.

The robust control invariant property follows by noting that $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) \cap \Omega \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}))$. If $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})$, then $\tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) = \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}))$. These two facts combine to give $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}))$. $\square$

It follows that if $\tilde{\mathcal{K}}_{i^*}(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_{i^*+1}(\Omega, \mathbb{T})$, then

$$\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_\infty(\Omega, \mathbb{T}), \forall i \geq i^* .$$

Some properties of robust controllable sets are as follows:

Proposition 2.6.

1. *For $i > 0$, if $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$ is robust control invariant, then so is $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})$. In general, the reverse statement does not hold.*

2. *If $x_k \in \tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) \backslash \tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \neq \emptyset$, then there exists an admissible control input which will ensure that for all allowable disturbances the state at the next time instant is in $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$;*

3. *If $x_k \in \tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \backslash \tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) \neq \emptyset$, then there does not exist an admissible control input which will ensure that for all allowable disturbances the state at the next time instant is in $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$;*

4. *There does not exist an admissible control law which will ensure that the system reaches $\mathbb{T}$ in i steps or less for all allowable disturbance sequences if and only if*

$$x_k \notin \bigcup_{j=0}^{i} \tilde{\mathcal{K}}_j(\Omega, \mathbb{T}) .$$

Proof. The proof of the first property is given here. The other properties are a consequence of the definition of robust controllable sets.

$\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$ is robust control invariant if and only if $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}))$. In addition, if $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$ is robust control invariant, then $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \subseteq \Omega$. This implies that $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) \cap \Omega$.

By construction, $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) \cap \Omega$. As a result, $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})$ and by Proposition 2.1, $\tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})) \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}))$.

Combining this with $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}))$, it follows that $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{Q}}(\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}))$ and hence $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T})$ is robust control invariant.

Example 5.1 includes a counter-example for the reverse statement. □

2.7 Robust Stabilisable Sets

If the target set is a robust control invariant set, then the robust controllable sets take on special geometric properties. To emphasise this special case, the following definition is given[9].

[9] The use of a robust control invariant target set in the calculation of robust controllable sets for LTI systems is also described in [MS97].

Definition 2.11 (Robust stabilisable set). The set $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ is the *i-step robust stabilisable set* contained in Ω for the system $x_{k+1} = f(x_k, u_k, w_k)$ if and only if $\mathbb{T}$ is a *robust control invariant* subset of Ω and $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ contains all states in Ω for which there exists an admissible time-varying feedback law which will drive the state of the system to $\mathbb{T}$ in i steps or less, while keeping the evolution of the state inside Ω for all allowable disturbance sequences, i.e.

$$\tilde{\mathcal{S}}_i(\Omega, \mathbb{T}) \triangleq \{x_0 \in \mathbb{R}^n \mid \exists \{u_k = h_k(x_k) \in \mathbb{U}\}_0^{i-1}, N \leq i : \{x_k \in \Omega\}_0^{N-1}, \\ \{x_i \in \mathbb{T} \subseteq \Omega\}_N^i, \mathbb{T} \subseteq \tilde{\mathcal{Q}}(\mathbb{T})\}. \quad (2.12)$$

Remark 2.12. If the notation $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ is used, $\mathbb{T}$ is a robust control invariant subset of Ω. If $\tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$ is used, $\mathbb{T}$ can be any arbitrary subset of $\mathbb{R}^n$.

The reason for the choice of the word *stabilisable* to distinguish it from *controllable* sets is because in most practical applications the target set is either a bounded robust control invariant set or a robust positively invariant set for a Lyapunov-stable closed-loop system. If the initial state is contained inside a robust stabilisable set, then one can design a control law which guarantees that the target set will be reached in a finite number of steps. Once inside the target set one can switch to the Lyapunov-stable controller[10]. This results in "ultimately bounding" the states of the closed-loop system.

In light of this discussion, the largest possible region of attraction to the target set is equal to the maximal robust stabilisable set.

Definition 2.12 (Maximal robust stabilisable set). The set $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ is the *maximal robust stabilisable set* contained in Ω for the system $x_{k+1} = f(x_k, u_k, w_k)$ if and only if $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ is the union of all i-step robust stabilisable sets contained in Ω.

In general, the maximal robust stabilisable set $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ is not equal to the maximal robust control invariant set $\tilde{\mathcal{C}}_\infty(\Omega)$, even for linear systems. $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T}) \subseteq \tilde{\mathcal{C}}_\infty(\Omega)$ for all robust control invariant $\mathbb{T}$. The set $\tilde{\mathcal{C}}_\infty(\Omega) \backslash \tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ includes all initial states from which it is not possible to robustly steer the system to the stabilisable region $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ (and hence to $\mathbb{T}$). It might only be possible to bound the norm of the states $\|x_k\|$ as in the case of a limit cycle or to drive the system to an alternative stable equilibrium.

If $\mathbb{T}_1 \neq \mathbb{T}_2$ are two robust control invariant sets, then $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T}_1)$ and $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T}_2)$ are not necessarily equal. Similarly, $\tilde{\mathcal{S}}_\infty(\Omega, \{0\})$ is not necessarily equal to $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ if $0 \in \mathbb{T}$, since it is not always possible to drive some systems to the origin[11].

Some properties of robust stabilisable sets are:

[10]For the reader familiar with model predictive control, this is the same idea as used in dual-mode MPC [MM93].

[11]The region $\mathcal{S}_\infty(\mathbb{R}^n, \{0\})$ can be seen to be the generalisation to nonlinear systems of the ANCBI (asymptotically null-controllable with bounded inputs) region for controllable LTI systems with no state constraints and no disturbances [Las93]. The maximal stabilisable set $\mathcal{S}_\infty(\mathbb{X}, \{0\})$ is a generalisation to nonlinear systems of the "maximal admissible set" defined in [KG87] and the feasible region of the predictive control scheme defined in [PN00a].

Proposition 2.7. *[MS97, Thm. 2]*

1. *Each set* $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ *is robust control invariant;*

2. *Each set contains all previous sets:*

$$\tilde{\mathcal{S}}_{i+1}(\Omega, \mathbb{T}) \supseteq \tilde{\mathcal{S}}_i(\Omega, \mathbb{T})\,;$$

3. *For each set*

$$\tilde{\mathcal{S}}_i(\Omega, \mathbb{T}) = \bigcup_{j=0}^{i} \tilde{\mathcal{S}}_j(\Omega, \mathbb{T})\,;$$

4. *If* $x_k \in \tilde{\mathcal{S}}_{i+1}(\Omega, \mathbb{T})$*, then there exists a control input which will drive the state to* $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ *at the next time instant for all allowable disturbances;*

5. *There does not exist a control law which ensures that the system reaches* $\mathbb{T}$ *in* i *steps or less for all allowable disturbance sequences if and only if* $x_k \notin \tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$.

Since robust stabilisable sets are special cases of controllable sets, the same procedure as in Algorithm 2.1 can be followed to compute the respective sets.

Algorithm 2.2 (Robust stabilisable sets). *Algorithm 2.1 can be used to compute the* i*-step robust stabilisable sets* $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ *contained in* Ω *by noting that*

$$\tilde{\mathcal{S}}_i(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})\,. \tag{2.13}$$

If $\tilde{\mathcal{S}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$*, then terminate and set* $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T}) = \tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$.

The notion of a finitely determined maximal stabilisable set once again carries through as with the infinite-time robust controllable set. However, in this case, the condition is both necessary and sufficient.

Theorem 2.2. *The set* $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T})$ *is finitely determined if and only if* $\exists i \in \mathbb{N}$ *such that* $\tilde{\mathcal{S}}_i(\Omega, \mathbb{T}) = \tilde{\mathcal{S}}_{i+1}(\Omega, \mathbb{T})$.

Proof. (Only if) By Proposition 2.7, $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T}) \supseteq \ldots \supseteq \tilde{\mathcal{S}}_{i+1}(\Omega, \mathbb{T}) \supseteq \tilde{\mathcal{S}}_i(\Omega, \mathbb{T}) \supseteq \ldots \supseteq \mathbb{T}$. If $\tilde{\mathcal{S}}_\infty(\Omega, \mathbb{T}) = \tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$, then $\tilde{\mathcal{S}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{S}}_i(\Omega, \mathbb{T})$ must follow, otherwise there would be a contradiction. The reverse follows from Lemma 2.1. $\square$

2.8 Robust Admissible Sets and the Computation of the Maximal Robust Control Invariant Set

Given a set, it is also of interest to determine the set of states for which one can find a control law to keep one inside the set for a specified number of steps. The resulting sets are a special case of the robust controllable sets with the target set equal to the given set Ω and the following definition is given to distinguish this from the general robust controllable sets.

Definition 2.13 (Robust admissible set). The *i-step robust admissible set* $\tilde{\mathcal{C}}_i(\Omega)$ contained in Ω is the set of states for which an admissible time-varying feedback control law exists such that the evolution of the state remains inside Ω for i steps, for all allowable disturbance sequences, i.e.

$$\tilde{\mathcal{C}}_i(\Omega) \triangleq \left\{x_0 \in \mathbb{R}^n \mid \exists \{u_k = h_k(x_k) \in \mathbb{U}\}_0^{i-1} : \{x_k \in \Omega\}_0^i, \forall \{w_k \in \mathbb{W}\}_0^{i-1}\right\} . \tag{2.14}$$

From the definition of robust admissible sets, it is easy to show the following:

Proposition 2.8. *[Ber72]*

1. *The $(i+1)$-step robust admissible set is contained in all previous sets:*
$$\tilde{\mathcal{C}}_{i+1}(\Omega) \subseteq \tilde{\mathcal{C}}_i(\Omega) ;$$
2. *For each set*
$$\tilde{\mathcal{C}}_i(\Omega) = \bigcap_{j=0}^{i} \tilde{\mathcal{C}}_j(\Omega) ;$$
3. *There does not exist an admissible control law which will ensure that the state evolution remains within Ω for i steps for all allowable disturbance sequences if and only if the state $x_k \notin \tilde{\mathcal{C}}_i(\Omega)$;*
4. *If the state $x_k \in \tilde{\mathcal{C}}_i(\Omega) \backslash \tilde{\mathcal{C}}_\infty(\Omega) \neq \emptyset$ then there does not exist an admissible control input which will ensure that the state at the next time instant is in $\tilde{\mathcal{C}}_i(\Omega)$ for all allowable disturbances.*

The following result is used in developing an algorithm for computing the maximal robust control invariant set. It is an immediate consequence of the definitions of the robust admissible sets and the maximal robust control invariant set.

Proposition 2.9. *If there exists an $i \in \mathbb{N}$ such that $\tilde{\mathcal{C}}_{i+1}(\Omega) = \tilde{\mathcal{C}}_i(\Omega)$, then $\tilde{\mathcal{C}}_\infty(\Omega) = \tilde{\mathcal{C}}_i(\Omega)$.*

Proof. See the proof of [VSLS99, Thm. 2]. □

As with robust controllable sets, the problem of determining the robust admissible sets is equivalent to the "reachability of a target tube" of [BR71], with the target set $\mathbb{T} = \Omega$:

Algorithm 2.3 (Robust admissible sets). *The i-step robust admissible set $\tilde{\mathcal{C}}_i(\Omega)$ and maximal robust admissible set $\tilde{\mathcal{C}}_\infty(\Omega)$ can be computed using Algorithm 2.1 by noting that*

$$\tilde{\mathcal{C}}_i(\Omega) = \tilde{\mathcal{K}}_i(\Omega, \Omega) . \tag{2.15}$$

If $\tilde{\mathcal{C}}_i(\Omega) = \emptyset$, then terminate and set $\tilde{\mathcal{C}}_\infty(\Omega) = \emptyset$.

If $\tilde{\mathcal{C}}_{i+1}(\Omega) = \tilde{\mathcal{C}}_i(\Omega)$, then terminate and set $\tilde{\mathcal{C}}_\infty(\Omega) = \tilde{\mathcal{C}}_i(\Omega)$.

This method for computing the maximal robust control invariant set was first described in [Ber72]. [Ber72] applies the ideas of [BR71] to the problem of computing a “strongly reachable” subset of a given set. Convergence questions are addressed and it is shown that certain compactness and continuity conditions are sufficient in order to guarantee convergence of the sequence of robust admissible sets to the maximal robust control invariant set. Relatively weak assumptions on the system and constraints guarantee convergence of the sequence $\tilde{\mathcal{C}}_i(\Omega)$ to the maximal robust control invariant set.

Proposition 2.10 (Convergence to the maximal robust control invariant set). *[Ber72]*
Assuming the system is of the form $x_{k+1} = f_{xu}(x_k, u_k) + w_k$ and that $\tilde{\mathcal{C}}_\infty(\Omega)$ is non-empty. If Ω and $\mathbb{U}$ are compact and the function $f_{xu}(x_k, u_k)$ is continuous, then given any bounded open set Φ such that $\tilde{\mathcal{C}}_\infty(\Omega) \subset \Phi$, there exists a positive integer $i < \infty$ such that $\tilde{\mathcal{C}}_\infty(\Omega) \subseteq \tilde{\mathcal{C}}_i(\Omega) \subset \Phi$.

A necessary and sufficient condition for the finite-determinedness of the maximal robust control invariant set can be derived.

Theorem 2.3. *$\tilde{\mathcal{C}}_\infty(\Omega)$ is finitely determined if and only if $\exists i \in \mathbb{N}$ such that $\tilde{\mathcal{C}}_i(\Omega) = \tilde{\mathcal{C}}_{i+1}(\Omega)$.*

Proof. (Only if) By Proposition 2.8, $\tilde{\mathcal{C}}_\infty(\Omega) \subseteq \ldots \subseteq \tilde{\mathcal{C}}_{i+1}(\Omega) \subseteq \tilde{\mathcal{C}}_i(\Omega) \subseteq \ldots \subseteq \Omega$. If $\tilde{\mathcal{C}}_\infty(\Omega) = \tilde{\mathcal{C}}_i(\Omega)$, then $\tilde{\mathcal{C}}_{i+1}(\Omega) = \tilde{\mathcal{C}}_i(\Omega)$ must follow, or else there would be a contradiction. The reverse follows from Lemma 2.1. □

In general, the maximal robust control invariant set is not finitely determined. However, for LTI systems it is possible to guarantee finite determinedness for some very simple cases where the control is unbounded [VSLS99].

An interesting class of systems for which finite determinedness is guaranteed, is the class of finite state machines with bounded constraints[12]. Since the number of possible states are finite, termination of Algorithm 2.3 is guaranteed.

2.9 Sets for Closed-loop Systems and Systems without Control Inputs

All the definitions and properties regarding robust controllable, stabilisable and admissible sets also apply to closed-loop systems, but with “robust control invariant” substituted with “robust positively

[12]The framework in this chapter needs to be extended slightly to deal with the class of hybrid systems, where some of the state variables can only take on values from a countable set [VSLS99].

invariant". As mentioned in Section 2.2 care has to be taken in calculating the sets by replacing the original state constraints with the input admissible set. To further distinguish the fact that the system is in closed-loop with a control law $u_k = h(x_k)$ or does not have a control input, the notation $\tilde{\mathcal{O}}^h$ and $\tilde{\mathcal{O}}$ will be used, respectively.

Obviously, the use of the word *controllable* does not make sense for systems with no available input. However, the following two definitions are given.

Definition 2.14 (The set $\tilde{\mathcal{KO}}_i^h(\Omega, \mathbb{T})$). The set $\tilde{\mathcal{KO}}_i^h(\Omega, \mathbb{T})$ for the system $x_{k+1} = f(x_k, u_k, w_k)$ in closed-loop with the control law $u_k = h(x_k)$ is defined as

$$\tilde{\mathcal{KO}}_i^h(\Omega, \mathbb{T}) \triangleq \tilde{\mathcal{K}}_i(\Omega^h, \mathbb{T})$$

for the system $x_{k+1} = f(x_k, h(x_k), w_k)$.

Remark 2.13. If $\mathbb{T}$ is robust positively invariant for the closed-loop system, then $\tilde{\mathcal{KO}}_i^h(\Omega, \mathbb{T})$ is robust positively invariant as well.

Note that the input admissible subset of the target set is not included in the above definition. This is to allow one to develop general results without introducing too much additional notation, as will be seen in Chapter 5.

If the target set is equal to the input admissible subset of Ω, then the following definition applies.

Definition 2.15 (The robust admissible set $\tilde{\mathcal{O}}_i^h(\Omega)$). The i-step robust admissible set for the system $x_{k+1} = f(x_k, u_k, w_k)$ in closed-loop with the control law $u_k = h(x_k)$ is defined as

$$\tilde{\mathcal{O}}_i^h(\Omega) \triangleq \tilde{\mathcal{C}}_i(\Omega^h)$$

for the system $x_{k+1} = f(x_k, h(x_k), w_k)$.

Remark 2.14. Note that

$$\tilde{\mathcal{O}}_i^h(\Omega) = \tilde{\mathcal{K}}_i(\Omega^h, \Omega^h) = \tilde{\mathcal{KO}}_i^h(\Omega, \Omega^h) .$$

The sets introduced in this section can be computed using Algorithm 2.1.

Though most of the sets defined in this chapter is not guaranteed to be finitely determined, it is possible to obtain a determinedness result for autonomous LTI systems where Ω is given by linear inequalities.

Proposition 2.11. *[KG98] Assume the system is given by $x_{k+1} = Ax_k + Ew_k$, $y_k = \phi(x_k) = Cx_k$, $\mathbb{Y}$ and $\mathbb{W}$ are convex, compact polyhedra containing the origin and $0 \in \tilde{\mathcal{O}}_\infty(\mathbb{X}^\phi) \neq \emptyset$. If the eigenvalues of A are all contained inside the unit disk and (C, A) is observable, then $\tilde{\mathcal{O}}_\infty(\mathbb{X}^\phi)$ is finitely determined.*

This result allows one to guarantee that if the output constraints for an observable LTI system are bounded and one has designed an asymptotically stabilising state feedback controller, then the maximal robust positively invariant set contained inside the input-output admissible set is finitely determined, assuming it exists.

2.10 Some Set-theoretic Concepts

For practical implementation, it is necessary to develop a procedure for computing the robust one-step set. Two set-theoretic concepts which will be useful in developing such an algorithm are the Pontryagin difference and the Minkowski sum. The support function is another tool which helps one to develop a number of algorithms for working with sets. The application of these ideas to specific classes of systems will be illustrated in Chapters 3 and 4.

2.10.1 The Pontryagin Difference

On close investigation of the literature on robust invariant set theory, it will be noted that before the robust one-step set $\tilde{\mathcal{Q}}(\Omega)$ can be computed, an intermediate set has to be computed if there is an additive state disturbance present. This set is the Pontryagin difference.

Definition 2.16 (The Pontryagin Difference). Given the sets $\Omega \subset \mathbb{R}^n$ and $\Phi \subset \mathbb{R}^n$, the Pontryagin difference between Ω and Φ is defined as

$$\Omega \sim \Phi \triangleq \left\{\omega \in \mathbb{R}^n \mid \omega + \psi \in \Omega, \forall \psi \in \Phi\right\} . \tag{2.16}$$

The Pontryagin difference, sometimes referred to as the Minkowski difference [MS97], is useful in various aspects of geometry and control theory. A detailed discussion of the properties of the Pontryagin difference is given in [KG98].

Remark 2.15. Note that

$$0_n \in \Phi \Rightarrow \Omega \sim \Phi \subseteq \Omega .$$

A result which allows one to compute the Pontryagin difference if Ω is a convex polyhedron, is given in Section 3.3.3.

Disturbance Acting on the State and the System Structure

Often the system (2.1) can be written as

$$x_{k+1} = f_\Delta(x_k, u_k, w_k^\Delta) + f_s(w_k^s) , \tag{2.17}$$

where the disturbance consists of a component w_k^Δ which acts on the system structure and a component w_k^s which acts additively on the state:

$$w_k = (w_k^\Delta, w_k^s) \in \mathbb{W}^\Delta \times \mathbb{W}^s . \tag{2.18}$$

If one defines

$$\mathbb{D} \triangleq f_s(\mathbb{W}^s) \tag{2.19}$$

and provided $\Omega \sim \mathbb{D} \neq \emptyset$,

$$\mathcal{Q}_\Delta(\Omega \sim \mathbb{D}) \triangleq \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : f_\Delta(x_k, u_k, w_k^\Delta) \in \Omega \sim \mathbb{D}, \forall w_k^\Delta \in \mathbb{W}^\Delta\right\} \tag{2.20}$$

is equal to the robust one-step set that one is interested in computing, i.e.

$$\tilde{\mathcal{Q}}(\Omega) = \mathcal{Q}_\Delta(\Omega \sim \mathbb{D}) \; . \tag{2.21}$$

Algorithm 2.1 is now modified by substituting (2.11) with

$$\tilde{\mathcal{K}}_0(\Omega, \mathbb{T}) = \mathbb{T} \tag{2.22a}$$

$$\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \mathcal{Q}_\Delta(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \sim \mathbb{D}) \cap \Omega \, . \tag{2.22b}$$

A procedure for computing $\mathcal{Q}_\Delta(\Omega \sim \mathbb{D})$ for LTI systems with parametric uncertainty is briefly described in Section 3.3.3.

Disturbance Acting on the State

Often the system dynamics (2.1) can be split into two parts, with the disturbance acting only on the state:

$$x_{k+1} = f_{xu}(x_k, u_k) + f_w(w_k) \, . \tag{2.23}$$

If this is the case, then the computation of the one-step set can be done as before, by first calculating the intermediate set $\Omega \sim \mathbb{D}$, where

$$\mathbb{D} \triangleq f_w(\mathbb{W}) \, .$$

Once the Pontryagin difference $\Omega \sim \mathbb{D}$ has been computed, the robust one-step set can be found by calculating the *nominal* one-step set $\mathcal{Q}(\Omega \sim \mathbb{D})$:

$$\tilde{\mathcal{Q}}(\Omega) = \mathcal{Q}(\Omega \sim \mathbb{D}) \triangleq \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}, x_{k+1} \in \Omega \sim \mathbb{D} : x_{k+1} = f_{xu}(x_k, u_k)\right\} \; . \tag{2.24}$$

Algorithm 2.1 can now be modified by substituting (2.11) with

$$\tilde{\mathcal{K}}_0(\Omega, \mathbb{T}) = \mathbb{T} \tag{2.25a}$$

$$\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \mathcal{Q}(\tilde{\mathcal{K}}_i(\Omega, \mathbb{T}) \sim \mathbb{D}) \cap \Omega \tag{2.25b}$$

To complete the computation of the robust one-step set, one needs to develop an algorithm which eliminates the existential quantifier in (2.24). One way of achieving this is by noting that $\tilde{\mathcal{Q}}(\Omega)$ is the orthogonal projection of the set

$$\Psi \triangleq \left\{[x_k' \; u_k']' \in \mathbb{R}^{n+m} \mid f_{xu}(x_k, u_k) \in \Omega \sim \mathbb{D}, u_k \in \mathbb{U}\right\} \tag{2.26}$$

onto the subspace spanned by the first n coordinates, i.e.

$$\tilde{\mathcal{Q}}(\Omega) = \Pi_{\mathbb{R}^n}\Psi \,,$$

where $\Pi_{\mathbb{R}^n}$ is the projection operator.

An alternative way of computing the robust one-step set, provided the system has a special structure, is as the inverse map of a Minkowski sum.

2.10.2 The Minkowski Sum

In many cases (2.23) can be split further into three distinct parts:

$$x_{k+1} = f_x(x_k) + f_u(u_k) + f_w(w_k)\,. \tag{2.27}$$

The Minkowski sum can then be used to get an alternative expression for the robust one-step set.

Definition 2.17 (Minkowski sum). [GS93] Given two sets $\Omega \subset \mathbb{R}^n$ and $\Phi \subset \mathbb{R}^n$, the Minkowski sum (vector sum) of Ω and Φ is defined as

$$\Omega \oplus \Phi \triangleq \{x \in \mathbb{R}^n \mid \exists \omega \in \Omega, \phi \in \Phi : x = \omega + \phi\}\,. \tag{2.28}$$

Remark 2.16. Note that if $0 \in \Phi$, then the set inclusion

$$(\Omega \sim \Phi) \oplus \Phi \subseteq \Omega$$

always holds, but

$$0 \in \mathbb{D} \not\Rightarrow (\Omega \sim \Phi) \oplus \Phi = \Omega\,.$$

By defining

$$\mathbb{V} \triangleq -f_u(\mathbb{U}),$$

from (2.24) it follows that

$$\begin{aligned}
\mathcal{Q}(\Omega \sim \mathbb{D}) &= \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}, x_{k+1} \in \Omega \sim \mathbb{D} : x_{k+1} = f_x(x_k) + f_u(u_k)\right\} \\
&= \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}, x_{k+1} \in \Omega \sim \mathbb{D} : f_x(x_k) = x_{k+1} - f_u(u_k)\right\} \\
&= \left\{x_k \in \mathbb{R}^n \mid f_x(x_k) \in (\Omega \sim \mathbb{D}) \oplus (-f_u(\mathbb{U}))\right\} \\
&= \left\{x_k \in \mathbb{R}^n \mid f_x(x_k) \in (\Omega \sim \mathbb{D}) \oplus \mathbb{V}\right\}
\end{aligned}$$

and hence

$$\tilde{\mathcal{Q}}(\Omega) = \left\{x_k \in \mathbb{R}^n \mid f_x(x_k) \in (\Omega \sim \mathbb{D}) \oplus \mathbb{V}\right\}\,. \tag{2.29}$$

In other words, the robust one-step set can be computed as the inverse map $f_x^{-1}(x_k)$ of the Minkowski sum of $\Omega \sim \mathbb{D}$ and $\mathbb{V}$.

2.10.3 The Support Function

The *support function* [Grü67] is another concept which has proven itself to be useful when using a set-theoretic framework in control and information theory [Sch73, Wit80, KG98]. The support function will be used in Chapter 3 in developing algorithms for computing the Pontryagin difference and Minkowski sum of two convex polyhedra.

Definition 2.18 (Support function). The *support function* of the set Ω, evaluated at $\eta \in \mathbb{R}^n$, is defined as

$$h_\Omega(\eta) \triangleq \sup_{\omega \in \Omega} \eta' \omega\,. \tag{2.30}$$

The domain on which the support function is defined is all η for which $\eta'\omega$ is bounded from above on Ω. If Ω is bounded, then the domain is $\mathbb{R}^n$.

From this point on, it is assumed that the support function is always defined and that the supremum is a maximum.

Geometrically, if $\eta'\eta = 1$, then $h_\Omega(\eta)$ is the distance from the origin to a support hyperplane of Ω with a normal in the direction η.

Remark 2.17. Note that if Ω is a closed, convex polyhedron, then the support function can be computed by noting that the optimisation in (2.30) is a linear program.

It can be shown [Hny69] that an equivalent expression $\breve{\Omega}$ for the polyhedron

$$\Omega \triangleq \{\omega \in \mathbb{R}^n \mid Q\omega \preceq q\}$$

in terms of its support function is given by

$$\breve{\Omega} \triangleq \{\omega \in \mathbb{R}^n \mid Q\omega \preceq H(Q, \Omega)\}\,, \tag{2.31}$$

where the i'th component of $H(Q, \Omega)$ is given by the value of the support function of Ω, evaluated at Q_i', the transpose of the i'th row of Q:

$$H_i(Q, \Omega) \triangleq h_\Omega(Q_i')\,. \tag{2.32}$$

Note also that

$$H(Q, \breve{\Omega}) = H(Q, \Omega) \preceq q\,.$$

It follows immediately that if the polyhedron is *irredundant*[13], then

$$q = H(Q, \Omega)\,.$$

For a more detailed discussion on how the support function can be used to obtain an equivalent expression of a polyhedron, see [Grü67, Hny69] and [Sch73, App. G].

[13] An inequality representation of a polyhedron is irredundant if and only if none of the inequalities describing the polyhedron are redundant [Grü67, GS93].

2.11 Summary

This section brought together a number of ideas, definitions and results from set invariance theory. The concept of invariant sets were introduced and it was shown that a set is control invariant if and only if it is contained inside its robust one-step set. This condition forms the basis of a test for set invariance and is often used in the derivation of results on invariance.

Robust stabilisable and admissible sets were introduced and these were shown to be special cases of the robust controllable sets with different target sets. These sets and the maximal robust control invariant and maximal robust stabilisable sets can be computed using the iterative procedure of Algorithm 2.1. The key ingredients for implementing this algorithm are procedures for computing

- the robust one-step set $\tilde{\mathcal{Q}}(\cdot)$,
- the intersection $\tilde{\mathcal{Q}}(\cdot) \cap \Omega$ and
- whether the equality $\tilde{\mathcal{K}}_{i+1}(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i(\Omega, \mathbb{T})$ holds.

Some set-theoretic concepts were introduced which will be useful in implementing the above algorithm. It was shown that the Pontryagin difference could be used to compute an intermediate set if there are state disturbances present. If the disturbance does not act on the system structure then the robust one-step set can be computed by computing the nominal one-step set to the computed Pontryagin difference. If the system is of the form (2.27), then the Minkowski sum can be used to complete the computation of the nominal one-step set to the Pontryagin difference. Finally, the support function was introduced and it was shown that a polyhedron has an equivalent expression in terms of its support function.

Chapter 3

Uncertain Linear Time-Invariant Systems

This chapter deals with linear, time-invariant systems subject to linear inequality constraints. Parametric uncertainty and state disturbances are assumed. Results are given which allow the development of algorithms for computing invariant sets for such systems.

3.1 Introduction

Consider the uncertain, discrete-time, linear time-invariant (LTI) system:

$$x_{k+1} = A\left(w_k^{\Delta}\right)x_k + B\left(w_k^{\Delta}\right)u_k + Ew_k^s \tag{3.1}$$

with $k \in \mathbb{Z}$, x_k is the system state, u_k is the control input and $w_k = (w_k^{\Delta}, w_k^s) \in \mathbb{W}^{\Delta} \times \mathbb{W}^s$ is an unknown disturbance with w_k^s acting linearly on the state via the matrix $E \in \mathbb{R}^{n \times q_s}$, similar to the discussion in Section 2.10.1. The state is assumed to be measured. If there is no disturbance, then $x_{k+1} = Ax_k + Bu_k$. If there is no control input, then $x_{k+1} = A(w_k^{\Delta})x_k + Ew_k^s$.

The system is subject to linear inequality constraints on the control inputs and/or the states over the whole time horizon $k \in \mathbb{N}$:

$$\mathbb{U} \triangleq \{u \in \mathbb{R}^m \mid Gu \preceq g\} \tag{3.2a}$$

$$\mathbb{X} \triangleq \{x \in \mathbb{R}^n \mid Hx \preceq h\} \tag{3.2b}$$

where $h \in \mathbb{R}_+^{n_h}$ and $g \in \mathbb{R}_+^{n_g}$ define the constraints, with n_h and n_g denoting the number of state and input constraints respectively; $H \in \mathbb{R}^{n_h \times n}$ and $G \in \mathbb{R}^{n_g \times m}$ are the state and input constraint distribution matrices. Since the sets in (3.2) are given as the intersection of a finite number of half-spaces and h and g are positive, $\mathbb{U}$ and $\mathbb{X}$ are closed, convex polyhedra[1] containing the origin in their interior. Additionally, it is assumed that $\mathbb{U}$ is compact.

For the case of the disturbances that act linearly on the state, it is assumed that $\mathbb{W}^s$ is a compact set given by linear inequalities, as with (3.2), with the origin contained in the interior[2]. No further assumptions regarding the state disturbances are made.

It is assumed that there are parametric uncertainties [Bla94, De 94, De 97, KBM96] in the mathematical model of the system. More specifically, it is assumed that the actual system matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are contained in the convex hull of a set of p matrix pairs, i.e.

$$(A, B) \in \Delta \,, \tag{3.3}$$

where

$$\Delta \triangleq \operatorname{conv}\left\{(A_1, B_1), \ldots, \left(A_p, B_p\right)\right\} \,. \tag{3.4}$$

This means that $A\left(w_k^\Delta\right)$, $B\left(w_k^\Delta\right)$ and w_k^Δ satisfy

$$\begin{aligned} \left(A\left(w_k^\Delta\right), B\left(w_k^\Delta\right)\right) &= \sum_{j=1}^{p} \left(w_k^\Delta\right)_j \left(A_j, B_j\right) \,, \\ \sum_{j=1}^{p} \left(w_k^\Delta\right)_j = 1, &\quad \left(w_k^\Delta\right)_j \geq 0 \,. \end{aligned} \tag{3.5}$$

This relation defines the set $\mathbb{W}^\Delta$. Note that if $p = 1$, then there is no uncertainty in A and B.

3.2 Contractive Sets

Contractive sets are related to robust control invariant sets. The main idea behind contractive sets is that one is interested in computing a set for which an admissible control exists which will guarantee that the state at the next point in time is inside a subset of the original set.

One of the more useful results from contractive set theory and the reason for including the discussion on contractive sets in this chapter, is given by Theorem 3.1. The result allows one to compute an arbitrarily close inner approximation of the robust control invariant set if the latter is not finitely determined.

Definition 3.1 (C-set). [Bla94] A C-Set is a convex and compact set containing the origin.

Given a C-set, one would like to know whether there exists a control that will allow one to drive the system to a specified subset of the given set:

[1]In order to distinguish between bounded and unbounded constraints, a polytope is defined to be a bounded polyhedron, while a polyhedron can be bounded or unbounded.

[2]It is not assumed that $\mathbb{W}^\Delta$ contains the origin in its interior.

Definition 3.2 (Contractive set). [Bla94, Bla99] A *C-set* $\Omega \subset \mathbb{R}^n$ is *contractive* for a discrete-time system of the form (3.1) if and only if there exists a *nonlinear* feedback law $u_k = h(x_k)$ and a positive $\lambda \leq 1$ such that if $x_k \in \Omega$, then $x_{k+1} = A\left(w_k^\Delta\right) x_k + B\left(w_k^\Delta\right) h(x_k) + E w_k^s \in \lambda\Omega$ for all allowable disturbances $(w_k^\Delta, w_k^s) \in \mathbb{W}^\Delta \times \mathbb{W}^s$.

The fact that a given C-set is contractive, allows one to construct a controller with a given rate of convergence. See [Bla94, FG97] for more details on how to construct such a controller.

Remark 3.1. It is important to note that if a set is contractive, then it is also robust control/positively invariant. Hence, from this point on statements regarding robust control/positively invariant sets also apply to contractive sets. The properties of the robust "admissible", "stabilisable" and "control invariant" sets can also be applied to contractive sets. A superscript λ will be used to denote that a given set is contractive.

One can define the λ-contractive controllable sets $\tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$ as in Chapter 2. The following algorithm can be used to construct these sets.

Algorithm 3.1. *For a given λ, the maximal λ-contractive controllable set $\tilde{\mathcal{K}}_\infty^\lambda(\Omega, \mathbb{T})$ contained in Ω can be computed via the iteration:*

$$\tilde{\mathcal{K}}_0^\lambda(\Omega, \mathbb{T}) = \mathbb{T} \tag{3.6a}$$

$$\tilde{\mathcal{K}}_{i+1}^\lambda(\Omega, \mathbb{T}) = \tilde{\mathcal{Q}}\left(\lambda \tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})\right) \cap \Omega\,. \tag{3.6b}$$

If $\tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T}) = \emptyset$, then terminate and set $\tilde{\mathcal{K}}_\infty^\lambda(\Omega, \mathbb{T}) = \emptyset$.

If $0 \notin \tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$, then terminate and set $\tilde{\mathcal{K}}_\infty^\lambda(\Omega, \mathbb{T}) = \emptyset$.

If $\tilde{\mathcal{K}}_{i+1}^\lambda(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$, then terminate and set $\tilde{\mathcal{K}}_\infty^\lambda(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$.

As can be seen above, the only difference with respect to Algorithm 2.1 is that the set $\lambda\tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$ is used instead of $\tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$ in the calculation of the new set.

Proposition 3.1. *If $\tilde{\mathcal{K}}_\infty^\lambda(\Omega, \mathbb{T})$ is finitely determined, then $\tilde{\mathcal{K}}_\infty^\lambda(\Omega, \mathbb{T})$ is a convex polyhedron.*

As in Chapter 2 with the maximal robust control invariant set, one might also be interested in determining the *maximal λ-contractive* set. It can be computed in a fashion similar to the maximal robust control invariant set, using Algorithm 3.1 and noting that

$$\tilde{\mathcal{C}}_i^\lambda(\Omega) = \tilde{\mathcal{K}}_i^\lambda(\Omega, \Omega)\,.$$

Similarly, one can start with a λ-contractive target set $\mathbb{T}$ and compute the *maximal λ-contractive stabilisable* set $\tilde{\mathcal{S}}_\infty^\lambda(\Omega, \mathbb{T})$ by noting that

$$\tilde{\mathcal{S}}_i^\lambda(\Omega, \mathbb{T}) = \tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})\,.$$

Remark 3.2. Note that in general $\tilde{\mathcal{K}}^{\lambda}_{\infty}(\Omega, \mathbb{T})$, $\tilde{\mathcal{S}}^{\lambda}_{\infty}(\Omega, \mathbb{T})$ and $\tilde{\mathcal{C}}^{\lambda}_{\infty}(\Omega)$ are not polyhedra, nor are they guaranteed to be finitely determined. It is not clear whether the converse of Proposition 3.1 holds, even if $\mathbb{X}$ and $\mathbb{U}$ are both compact[3].

Together with the next result which allows one to compute a contractive set $\mathbb{T}$, by computing the stabilisable sets one can approximate the maximal λ-contractive set arbitrarily closely.

Theorem 3.1. *[Bla94] Assume that Ω is compact, $\tilde{\mathcal{C}}^{\lambda}_{\infty}(\Omega)$ is a C-set and $0 \leq \lambda < 1$. For every λ^* such that $\lambda < \lambda^* \leq 1$, there exists an $i^* < \infty$ such that $\tilde{\mathcal{C}}^{\lambda}_{i}(\Omega)$ is λ^*-contractive for all $i \geq i^*$.*

In general, the maximal robust control invariant and maximal λ-contractive sets are not finitely determined. Theorem 3.1 is useful since it gives a guarantee that the algorithm will terminate after a finite number of iterations. Once a λ^*-contractive set has been found $\mathbb{T} \triangleq \tilde{\mathcal{C}}^{\lambda^*}_{i}(\Omega)$, then one can compute a number of the $\tilde{\mathcal{S}}^{\lambda^*}_{i}(\Omega, \mathbb{T})$ in order to find a larger λ^*-contractive set. Hence one can approximate the corresponding maximal λ^*-contractive set arbitrarily closely.

Theorem 2.1 provides the basis for a test to determine whether a given set is contractive.

Corollary 3.1 (Geometric condition for contractiveness). *The C-set $\Omega \subset \mathbb{R}^n$ is λ-contractive if and only if $\Omega \subseteq \tilde{\mathcal{Q}}(\lambda\Omega)$.*

If a given set is λ-contractive, then the following result also holds:

Corollary 3.2. *If a C-set Ω is λ-contractive, with $\lambda < 1$, then it is also $\tilde{\lambda}$-contractive for all $\tilde{\lambda}$ with $\lambda < \tilde{\lambda} \leq 1$.*

Proof. If $\lambda < \tilde{\lambda}$, then $\lambda\Omega \subseteq \tilde{\lambda}\Omega$. By Proposition 2.1, $\tilde{\mathcal{Q}}(\lambda\Omega) \subseteq \tilde{\mathcal{Q}}(\tilde{\lambda}\Omega)$. Since $\Omega \subseteq \tilde{\mathcal{Q}}(\lambda\Omega)$, it follows that $\Omega \subseteq \tilde{\mathcal{Q}}(\tilde{\lambda}\Omega)$ and by Corollary 3.1, Ω is $\tilde{\lambda}$-contractive. ☐

For uncertain LTI systems, it is possible to say something more specific regarding the topological properties of the robust one-step set $\tilde{\mathcal{Q}}(\Omega)$.

Proposition 3.2.

1. *If Ω is compact, then the set $\tilde{\mathcal{Q}}(\Omega)$ is closed;*
2. *If Ω is convex, then the set $\tilde{\mathcal{Q}}(\Omega)$ is convex;*
3. *If Ω is a polyhedron, then the set $\tilde{\mathcal{Q}}(\Omega)$ is also a polyhedron;*
4. *If $p = 1$, A is non-singular and Ω is compact, then the set $\tilde{\mathcal{Q}}(\Omega)$ is compact.*

Proof. The proofs are standard. See [Bla94] for the first three results. See [BR71, Sect. 4] for the last result. The last result follows because the image of a compact set under the continuous mapping A^{-1} is compact. ☐

[3]The author is not entirely convinced by the argument presented in the proof of [KG87, Thm 4.2(ii)] for $\mathcal{S}_{\infty}(\mathbb{X}, \{0\})$ of controllable systems.

3.3 Computing the Invariant and Contractive Sets

The practical feasibility of computing the various sets and applying the theory described in Chapter 2 and Section 3.2 to controller design is dependent on algorithms existing for the calculation of the set $\tilde{\mathcal{Q}}(\Omega)$, the intersection of two sets and testing for equality or whether a set is a subset of another. The invariance and contractiveness tests can be implemented by recalling the inclusion conditions of Theorem 2.1 and Corollary 3.1.

These procedures are relatively straightforward and routine for uncertain LTI systems subject to polyhedral constraints on the states and control inputs. This section describes some well-known results, while Section 3.4 describes some less well-known results.

The presentation of this chapter is more along the lines of a description of the results that allow one to develop algorithms, rather than a detailed description of the algorithms themselves. In many cases, the algorithmic details follow immediately from the theoretical result and writing out the individual steps does not contribute significantly to the discussion.

A more abstract approach to algorithm development is adopted here and the practical implementation is not discussed. Appendix E contains a brief description of the functions in a Matlab toolbox that has been developed for the computation of the various sets discussed in this chapter.

3.3.1 Intersection of Two Polyhedra

The computation of the intersection of two polyhedral sets which are described by linear inequalities is trivial. Given the two sets $\Omega \triangleq \{x \in \mathbb{R}^n \mid Qx \preceq q\}$ and $\Phi \triangleq \{x \in \mathbb{R}^n \mid Sx \preceq s\}$, the intersection of the sets is found by appending Q and q to S and s, respectively. The intersection is then given by

$$\Omega \cap \Phi = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{bmatrix} Q \\ S \end{bmatrix} x \preceq \begin{bmatrix} q \\ s \end{bmatrix} \right\}. \tag{3.7}$$

Often some of the inequalities in (3.7) are redundant and could be removed, if required.

Proposition 3.3. *The intersection of two convex polyhedra is a convex polyhedron.*

Furthermore, the support function of the intersection of two polyhedra [BR71, App. 1] is given by

$$h_{\Omega\cap\Phi}(\eta) = \min\{h_\Omega(\eta), h_\Phi(\eta)\}\,.$$

3.3.2 Equality and Subset Testing

An equality test is used in all the algorithms to determine whether the iterations should continue. In principle, given polyhedral descriptions of the two sets $\Omega \triangleq \{x \in \mathbb{R}^n \mid Qx \preceq q\}$ and $\Phi \triangleq \{x \in \mathbb{R}^n \mid Sx \preceq s\}$ one can compare the elements of Q and q with S and s. However, the sizes of the matrices and vectors might be different and two differently scaled matrices and vectors can describe the same set. A different approach is therefore needed.

Provided all elements of q and s are non-zero and all redundant constraints have been removed, one can obtain equivalent descriptions of the sets in the normalised form $Q_0 x \preceq \mathbf{1}$ and $S_0 x \preceq \mathbf{1}$. One can test for equality by individually comparing the rows of Q_0 with all the rows of S_0 and removing the matching rows until the sets of rows are empty.

Depending on the numerical robustness of the method used to generate each $\tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$ this approach may work. The author has found that the above two approaches are good enough in many cases. Normalisation is seldom needed, since the matrices and vectors which describe the old and new sets often correspond exactly.

Another equality test can be derived by noting that two sets are equal if and only if each set is a subset of the other, i.e.

$$\Phi = \Omega \Leftrightarrow \Phi \subseteq \Omega \text{ and } \Omega \subseteq \Phi\,. \tag{3.8}$$

Using the support function as defined in Section 2.10.3, testing for inclusion is easy if the sets are convex polyhedra.

Proposition 3.4 (Subset test). *[KG98] If Ω is given by the M linear inequalities*

$$\Omega \triangleq \left\{ x \in \mathbb{R}^n \mid Qx \preceq q \right\}\,,$$

and Φ is any subset of $\mathbb{R}^n$ then

$$\Phi \subseteq \Omega \Leftrightarrow h_\Phi(Q_i') \leq q_i\,,\ \forall i = 1 \ldots M\,,$$

where Q_i is the i'th row of Q and q_i is the i'th component of q.

Remark 3.3. If Φ is also a convex polyhedron, then this condition can be checked by solving a sequence of M linear programming problems, since the support function of Φ can be computed by solving a linear program.

Remark 3.4. Testing whether a polyhedron Φ is a subset of another polyhedron Ω is equivalent to testing whether all the constraints in Ω are redundant with respect to the constraints in Φ.

Remark 3.5. The idea of checking whether all the new constraints are redundant is also used in [GT91, KG98, VSLS99] to test whether the computation of the maximal invariant set should terminate. It follows from Proposition 2.8 that $\tilde{\mathcal{C}}_{i+1}^\lambda(\mathbb{X}) \subseteq \tilde{\mathcal{C}}_i^\lambda(\mathbb{X})$. When calculating the maximal λ-contractive set, one therefore only needs to test whether $\tilde{\mathcal{C}}_i^\lambda(\mathbb{X}) \subseteq \tilde{\mathcal{C}}_{i+1}^\lambda(\mathbb{X})$ to check whether $\tilde{\mathcal{C}}_i^\lambda(\mathbb{X}) = \tilde{\mathcal{C}}_{i+1}^\lambda(\mathbb{X})$.

An elegant method for determining when to terminate the computation of the *maximal stabilisable set* is to fix some tolerance parameter and terminate when the new set is “close” to the true maximal stabilisable set. The authors of [GC87] show, via a compactness argument, that when computing the *maximal stabilisable set* $\mathcal{S}_\infty(\mathbb{X}, \{0\})$ for controllable LTI systems with Ω and $\mathbb{U}$ compact, for any given $\epsilon > 0$ there exists a $\tau = \tau(\epsilon)$ such that for all $i > \tau$,

$$\mathcal{S}_i(\Omega, \{0\}) \subseteq \mathcal{S}_\infty(\Omega, \{0\}) \subseteq (1+\epsilon)\mathcal{S}_i(\Omega, \{0\})\,. \tag{3.9}$$

An algorithm for computing τ is given in [CG86]. Algorithm 2.2 is modified to terminate when i is larger than τ.

It is possible that a similar argument can be made if the terminal set $\mathbb{T}$ is a control invariant set containing the origin and the system is stabilisable. However, it is not clear how to proceed. In addition, $\mathbb{X}$ is not necessarily bounded, thereby violating the assumptions made in [GC87]. It would be interesting to determine whether it is possible to derive a similar result if there are disturbances present.

Approximations for Equality and Subset Testing

Sometimes, due to numerical errors, problems can be experienced when testing for set inclusion. The following definitions can be used to test whether a given subset is a subset of another within a given tolerance:

Definition 3.3. The set $\Phi \subset \mathbb{R}^n$ is a subset of the set $\Omega \subset \mathbb{R}^n$ within a given tolerance $\epsilon > 0$ if and only if $\Phi \subseteq (1+\epsilon)\Omega$.

Based on the approximate subset test, the following definition is given to allow one to determine whether two sets are equal within a given tolerance:

Definition 3.4. The set $\Phi \subset \mathbb{R}^n$ is equal to the set $\Omega \subset \mathbb{R}^n$ within a given tolerance $\epsilon > 0$ if and only if $\Phi \subseteq (1+\epsilon)\Omega$ and $\Omega \subseteq (1+\epsilon)\Phi$.

Remark 3.6. These definitions can be loosely interpreted as a relaxation of the Hausdorff metric defined for two sets. The computation of the Hausdorff metric is computationally difficult to implement, whereas the conditions defined here are very quick and easy to check.

It is arguable as to whether these are good measures for set equality and subset testing and whether some other metric should not be used to define how "close" one set is to another. However, the definition above is intuitive and easy to implement in computing both the maximal robust control invariant set and maximal robust stabilisable sets. An inner approximation of $\tilde{\mathcal{S}}^\lambda_\infty(\Omega, \mathbb{T})$ and an outer approximation of $\tilde{\mathcal{C}}^\lambda_\infty(\Omega)$ will result if these methods are used for termination.

However, it is not desirable to obtain an *outer* approximation of $\tilde{\mathcal{C}}^\lambda_\infty(\Omega)$, since one cannot guarantee that the resulting set is at least 1-contractive. The most elegant, known way to obtain an approximation of $\tilde{\mathcal{C}}^\lambda_\infty(\Omega)$ was discussed earlier and is given by Theorem 3.1. This approach has the benefit of guaranteeing that the algorithm will terminate after a finite number of iterations when a λ^*-contractive *inner* approximation of $\tilde{\mathcal{C}}^{\lambda}_{\infty}(\Omega)$ has been found, where $\lambda < \lambda^* \leq 1$. An arbitrarily close approximation to $\tilde{\mathcal{C}}^{\lambda}_{\infty}(\Omega)$ can be found by choosing $\lambda \neq 1$, but sufficiently close to 1 and terminating when the computed set is λ^*-contractive for any $\lambda < \lambda^* \leq 1$.

3.3.3 The Robust One-step Set

Before proceeding, it is useful to recall the discussion and definitions from Section 2.10. It was shown that if part of the disturbance acts directly on the state, then an intermediate set needs to be calculated in order to obtain $\tilde{\mathcal{Q}}(\lambda\Omega)$, namely the Pontryagin difference. If one defines

$$\mathbb{D} \triangleq \left\{x_k^s \in \mathbb{R}^n \mid \exists w_k^s \in \mathbb{W}^s : x_k^s = E w_k^s\right\}$$

and the modified one-step set as

$$\mathcal{Q}_{\Delta}(\lambda\Omega \sim \mathbb{D}) \triangleq \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : A\left(w_k^{\Delta}\right) x_k + B\left(w_k^{\Delta}\right) u_k \in \lambda\Omega \sim \mathbb{D}, \forall w_k^{\Delta} \in \mathbb{W}^{\Delta}\right\} , \quad (3.10)$$

then

$$\tilde{\mathcal{Q}}(\lambda\Omega) = \mathcal{Q}_{\Delta}(\lambda\Omega \sim \mathbb{D}) . \quad (3.11)$$

The next section describes how one can compute the Pontryagin difference. Given this set, one can then use a projection method or the Minkowski sum method to complete the computation of the robust one-step set.

The Pontryagin Difference

If the two sets under consideration are given by linear inequalities, then the Pontryagin difference can be computed as follows:

Proposition 3.5 (Pontryagin difference). *Given two polyhedra*

$$\Omega \triangleq \{x \in \mathbb{R}^n \mid Qx \preceq q\}$$

and

$$\mathbb{D} \triangleq \{x \in \mathbb{R}^n \mid Sx \preceq s\}$$

with $Q \in \mathbb{R}^{M\times n}$, $q \in \mathbb{R}^M$, $S \in \mathbb{R}^{N\times m}$ and $s \in \mathbb{R}^N$, the Pontryagin difference $\lambda\Omega \sim \mathbb{D}$ is given by

$$\lambda\Omega \sim \mathbb{D} = \left\{x \in \mathbb{R}^n \mid Qx \preceq \lambda q - H(Q, \mathbb{D})\right\} , \quad (3.12)$$

where the i'th element of $H(Q,\mathbb{D})$ is the value of the support function of $\mathbb{D}$, evaluated at the i'th row of Q, i.e.

$$H_i(Q,\mathbb{D}) \triangleq h_{\mathbb{D}}(Q_i') = \max_{x\in\mathbb{D}} Q_i x . \quad (3.13)$$

If $\mathbb{D} = E\mathbb{W}^s$, then

$$H_i(Q, \mathbb{D}) = \max_{w \in \mathbb{W}^s} Q_i E w \,. \tag{3.14}$$

Proof. See [KG98, Thm. 2.3] □

Remark 3.7. Note that the Pontryagin difference requires solving no more than M linear programming problems. It is also not necessary to compute the mapping $E\mathbb{W}$, thereby reducing computation time. Furthermore, some of the constraints in $\lambda\Omega \sim \mathbb{D}$ might be redundant and these can be removed by solving M additional LPs.

The following result follows immediately and states that the complexity of the Pontryagin difference is independent of the number of inequalities describing $\mathbb{D}$:

Corollary 3.3 (Pontryagin difference). *If the polyhedra Ω and $\mathbb{D}$ are given by M and N linear inequalities, respectively, then the Pontryagin difference $\lambda\Omega \sim \mathbb{D}$ is given by (at most) M linear inequalities.*

The next step in the calculation of the robust one-step set $\tilde{\mathcal{Q}}(\lambda\Omega)$, given $\lambda\Omega \sim \mathbb{D}$, is to calculate the modified one-step set $\mathcal{Q}_\Delta(\lambda\Omega \sim \mathbb{D})$, either via a projection operation or, if there is no uncertainty in the plant matrices, via a Minkowski summation.

Remark 3.8. If there is no control input to the system, then the second step of projection or Minkowski summation is not necessary and the algorithm reduces to those described in [GT91, KG98]. One can still have parametric uncertainty in A, which is not considered in [GT91, KG98], but is discussed in [De 97].

Computing the Robust One-Step Set via Projection

By linearity and convexity, if the control input is such that $x_{k+1} = Ax_k + Bu_k \in \lambda\Omega \sim \mathbb{D}$ for all

$$(A, B) \in \left\{(A_1, B_1), \ldots, \left(A_p, B_p\right)\right\}$$

then the same control input guarantees that $x_{k+1} = Ax_k + Bu_k \in \lambda\Omega \sim \mathbb{D}$ for all

$$(A, B) \in \operatorname{conv}\left\{(A_1, B_1), \ldots, \left(A_p, B_p\right)\right\} .$$

By this argument, it follows that

$$\begin{aligned}
\mathcal{Q}_\Delta(\lambda\Omega \sim \mathbb{D}) &\triangleq \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : A\left(w_k^\Delta\right) x_k + B\left(w_k^\Delta\right) u_k \in \lambda\Omega \sim \mathbb{D}, \forall w_k^\Delta \in \mathbb{W}^\Delta\right\} \\
&= \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : Ax_k + Bu_k \in \lambda\Omega \sim \mathbb{D}, \forall (A, B) \in \Delta\right\} \\
&= \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U} : A_i x_k + B_i u_k \in \lambda\Omega \sim \mathbb{D}, \forall i = 1, \ldots, p\right\} .
\end{aligned}$$

Hence, one way of computing $\mathcal{Q}_\Delta(\lambda\Omega \sim \mathbb{D})$, given

$$\lambda\Omega \sim \mathbb{D} = \{x_k \in \mathbb{R}^n \mid Qx_k \preceq \tilde{q}\}$$

where

$$\tilde{q} \triangleq \lambda q - H(Q, \mathbb{D}),$$

is as the orthogonal projection of the polyhedron

$$\begin{aligned}\Psi &\triangleq \left\{ \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathbb{R}^{n+m} \,|u_k \in \mathbb{U}, A_i x_k + B_i u_k \in \lambda\Omega \sim \mathbb{D}, i = 1, \dots, p \right\} \\ &= \left\{ \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathbb{R}^{n+m} \left| \begin{bmatrix} QA_1 & QB_1 \\ \vdots & \vdots \\ QA_p & QB_p \\ 0 & G \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \preceq \begin{bmatrix} \tilde{q} \\ \vdots \\ \tilde{q} \\ g \end{bmatrix} \right. \right\}\end{aligned}$$

onto the subspace spanned by the first n coordinates [Bla94, Sect. VI], i.e.

$$\mathcal{Q}_\Delta(\lambda\Omega \sim \mathbb{D}) = \left\{ x_k \in \mathbb{R}^n \left| \exists [x_k', u_k']' \in \Psi \right. \right\}.$$

There are two popular ways of computing this projection:

- If Ψ is bounded one can compute the vertices of Ψ. The set $\mathcal{Q}_\Delta(\lambda\Omega \sim \mathbb{D})$ is then the convex hull of the projection of the vertices of Ψ. This is the technique implemented in the *Matlab Geometric Bounding Toolbox* [VKV$^+$] function `PROJECT`[4].

- By systematically eliminating u_k from the inequalities in Ψ. A popular method for solving a set of linear inequalities is the *Fourier-Motzkin elimination method* [Chv83, KS90], for which a division-free algorithm is given in [KG87]. The intuitive argument behind Fourier elimination is briefly described in Appendix C.

The vertex-based method can become computationally intractable, since the number of vertices can become quite large with an increase in the dimension and number of plants describing the uncertainty set [Chv83, Cha. 18]. It is also not possible to give a practically useful bound on the geometric complexity of the resulting set. The implementation of efficient, numerically robust algorithms for finding all the vertices and computing the convex hull of the projections is tricky and therefore not always the preferred approach.

[4]This toolbox only works with simple polyhedra, i.e. the number of edges attached to each vertex is no more than the dimension of the subspace in which the polyhedron is contained. Small, random perturbations are added to the polyhedron's components to overcome this limitation. As a result, the problem quickly becomes ill-conditioned.

Because of its simplicity and ease of implementation, Fourier elimination is quite popular. Although Fourier elimination is very simple it could be inefficient, since many redundant inequalities[5] are generated when solving for x_k. These inequalities need to be removed at each step in order to reduce the size of the problem in future iterations. Fourier elimination is therefore always followed by the removal of redundant inequalities from $\tilde{\mathcal{Q}}(\lambda\tilde{\mathcal{C}}_i(\Omega,\mathbb{T}))\cap\Omega$ or $\tilde{\mathcal{Q}}(\lambda\tilde{\mathcal{S}}_i(\Omega,\mathbb{T}))\cap\Omega$ using the algorithm[6] described in Appendix B. However, Fourier elimination does suffer from the fact that the number of redundant constraints could increase exponentially in the worst case [Sch86]. As a result, this method cannot guarantee computation of the robust one-step set in polynomial time or give a good bound on the geometric complexity of the resulting set.

Remark 3.9. Note that projection does not require that any of the matrices A_i be invertible. The vertex-based method will only work if Ψ is bounded. Fourier elimination may or may not work if Ψ is unbounded, but is guaranteed to work if Ψ is bounded.

Computing the Robust One-Step Set via Minkowski Summation

If one assumes that there is no uncertainty in the pair (A, B), then the following approach which does not rely on projection, can be used to compute $\mathcal{Q}(\lambda\Omega\sim\mathbb{D})$ [BR71, GS71]. If A^{-1} is the *inverse map* of A, then recalling the discussion in Section 2.10.2 it follows that

$$\begin{aligned}\tilde{\mathcal{Q}}(\lambda\Omega)=\mathcal{Q}(\lambda\Omega\sim\mathbb{D})&=\left\{x_k\in\mathbb{R}^n \mid \exists u_k\in\mathbb{U}: x_{k+1}=Ax_k+Bu_k\in\lambda\Omega\sim\mathbb{D}\right\}\\&=\left\{x_k\in\mathbb{R}^n \mid \exists u_k\in\mathbb{U}, x_{k+1}\in\lambda\Omega\sim\mathbb{D}: x_k=A^{-1}(x_{k+1}-Bu_k)\right\}\\&=A^{-1}\left((\lambda\Omega\sim\mathbb{D})\oplus(-B\mathbb{U})\right),\end{aligned}$$

where $(\lambda\Omega\sim\mathbb{D})\oplus(-B\mathbb{U})$ is the Minkowski sum of $\lambda\Omega\sim\mathbb{D}$ and $\mathbb{V}\triangleq-B\mathbb{U}$.

The Minkowski sum can be computed by finding the vertices of the corresponding sets and computing the convex hull of their sums [GS93], i.e.

$$\begin{aligned}(\lambda\Omega\sim\mathbb{D})\oplus\mathbb{V}&\triangleq\left\{x_k\in\mathbb{R}^n \,|\exists x_{k+1}\in\lambda\Omega\sim\mathbb{D}, v_k\in\mathbb{V}: x_k=x_{k+1}+v_k\right\}\\&=\operatorname{conv}\left\{x_k\in\mathbb{R}^n \,|x_{k+1}\in\operatorname{vert}(\lambda\Omega\sim\mathbb{D}), v_k\in\operatorname{vert}(\mathbb{V}), x_k=x_{k+1}+v_k\right\},\end{aligned}$$

where $\mathbb{V}$ can either be computed using a projection algorithm or, more efficiently, as suggested by Propositions 3.6 and 3.7.

Assuming A is invertible, then the robust one-step set can be computed in a similar fashion to the computation of the Minkowski sum, i.e.

$$\mathcal{Q}(\lambda\Omega\sim\mathbb{D})=\operatorname{conv}\left\{x_k \mid x_{k+1}\in\operatorname{vert}(\lambda\Omega\sim\mathbb{D}), u_k\in\operatorname{vert}(\mathbb{U}), x_k=A^{-1}x_{k+1}-A^{-1}Bu_k\right\}.$$

[5]In [CS00] an algorithm which is based on finding the minimal generators of a cone [Las86] is used to compute the projection. An *ad hoc* way of avoiding redundancies is described in [DDD89, Sect. 6]. However, it is possible that the number of inequalities could still be exponential in the worst case.

[6]In the later stages of Fourier elimination the linear inequalities encountered arise from the original constraints in a special way. If this is cleverly exploited, the redundant inequalities can be detected and removed with a minimum of computational effort [Chv83, Cha. 16].

This vertex-based method is adopted in [GC87, Alg. 4.4] and [MS97], where the Geometric Bounding Toolbox [VKV$^+$] was used in the latter to perform the vertex and convex hull computations.

Note that the invertibility assumption can be dropped. If the hyperplane representation of the Minkowski sum

$$\{x_k \in \mathbb{R}^n \mid Tx_k \preceq t\} \triangleq (\lambda\Omega \sim \mathbb{D}) \oplus (-B\mathbb{U})$$

has been computed, then

$$\begin{aligned} \tilde{\mathcal{Q}}(\lambda\Omega) = \mathcal{Q}(\lambda\Omega \sim \mathbb{D}) &= \{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}, x_{k+1} \in \lambda\Omega \sim \mathbb{D} : Ax_k = x_{k+1} - Bu_k\} \\ &= \{x_k \in \mathbb{R}^n \mid Ax_k \in (\lambda\Omega \sim \mathbb{D}) \oplus (-B\mathbb{U})\} \\ &= \{x_k \in \mathbb{R}^n \mid TAx_k \preceq t\}\,. \end{aligned}$$

By careful choice of algorithm the conversion from hyperplane- to vertex-representation, and vice-versa, can be achieved in polynomial time [GS93, Sect. 2.3.3]. However, as the dimension of the system n grows the number of vertices increases quite rapidly compared to the number of hyperplanes required to describe the various sets. The vertex-based method can therefore still become impractical for large systems.

The Minkowski sum can also be computed using a projection method such as Fourier elimination. Though Fourier elimination has worst-case exponential complexity this is not always a problem, since the matrix is often quite sparse and many redundant constraints can be removed at each step. More work needs to be done, however, in order to determine the suitability of using Fourier elimination in computing the Minkowski sum. It is possible that a mix of different inequality- and vertex-based methods might be best suited for the job.

3.3.4 Computation of the Reach Set

The reach set, as defined in Section 2.3, can also be computed using the projection method or via Minkowski summation if there is no uncertainty in the pair (A, B).

For example, if Ω is a polyhedron, then recalling that

$$\mathcal{R}(\Omega) \triangleq \left\{x_{k+1} \in \mathbb{R}^n \mid \exists x_k \in \Omega, u_k \in \mathbb{U} : x_{k+1} = Ax_k + Bu_k\right\},$$

it follows that $\mathcal{R}(\Omega)$ is the projection of the polyhedron

$$\Psi \triangleq \left\{[x'_{k+1}, x'_k, u'_k]' \in \mathbb{R}^{2n+m} \,| x_k \in \Omega, u_k \in \mathbb{U}, x_{k+1} = Ax_k + Bu_k\right\}$$

onto the subspace spanned by the first n coordinates.

Alternatively,

$$\mathcal{R}(\Omega) = A\Omega \oplus B\mathbb{U}\,,$$

where $A\Omega$ and $B\mathbb{U}$ can either be computed using a projection algorithm or, more efficiently, as suggested by Propositions 3.6 and 3.7.

3.4 Some Efficient Algorithms

This section describes some less well-known algorithms for subset testing and the computation of the linear map of a polyhedron. Depending on the size of the problem, these algorithms could be more efficient than algorithms based on the methods described in Section 3.3.

3.4.1 Subset Testing

The following necessary and sufficient condition for a polyhedron to be a subset of another is an extension of Farkas' lemma [Sch86].

Lemma 3.1 (Extended Farkas' lemma). *[DH96, Bla99] Given two polyhedra*

$$\Omega \triangleq \{\omega \in \mathbb{R}^n \mid Q\omega \preceq q\}$$

and

$$\Phi \triangleq \{\phi \in \mathbb{R}^n \mid S\phi \preceq s\},$$

with $Q \in \mathbb{R}^{M \times n}$, $q \in \mathbb{R}^M$, $S \in \mathbb{R}^{N \times n}$ and $s \in \mathbb{R}^N$, then

$$\Phi \subseteq \Omega$$

if and only if there exists a non-negative matrix $P \in \mathbb{R}^{M \times N}$ (i.e. $P_{i,j} \geq 0, \forall i, j$) such that

$$PS = Q$$

and

$$Ps \preceq q\,.$$

The existence of the matrix P can be checked by determining whether a solution to a feasibility problem exists. This can be set up as an LP where the decision variables are the elements of P and the constraints are $PS = Q$ and $Ps \preceq q$. A feasible solution exists if and only if $\Phi \subseteq \Omega$.

This implies that instead of having to solve M LPs to check whether Φ is a subset of Ω as in Proposition 3.4, only a single LP is sufficient. The difference is that with Proposition 3.4 each of the M LPs have n decision variables and N inequality constraints, while with Lemma 3.1 the LP has $M \times N$ decision variables and $2(M \times n) + M$ inequality constraints.

Depending on the problem and the LP solver that is used, either method could be faster. If an interior-point method is used, then the time complexity is polynomial in the number of decision variables and constraints, whereas if a simplex-based method is used, the complexity is worst-case exponential, even if termination occurs in polynomial time *on average*. An efficient practical algorithm would compare the sizes of M, N and n before deciding which algorithm to adopt.

This result can easily be extended to allow for the testing of set equality and approximate set equality using a single LP as well.

3.4.2 Linear Mapping of a Polyhedron

This section gives two results on how to compute the polyhedron $B\Omega$ if the number of columns of B is less than or equal to the number of rows. This is a realistic assumption, since in most physical systems the number of control inputs does not exceed the number of states[7].

Proposition 3.6 (Invertible matrix). *[BR71] A polyhedron*

$$\Omega \triangleq \{\omega \in \mathbb{R}^n \mid Q\omega \preceq q\}$$

with $Q \in \mathbb{R}^{N\times n}$ and $q \in \mathbb{R}^N$ is given. If $B \in \mathbb{R}^{n\times n}$ is invertible, then

$$B\Omega = \{x \in \mathbb{R}^n \mid QB^{-1}x \preceq q\}\,. \tag{3.15}$$

Proof. By definition,

$$B\Omega \triangleq \{x \in \mathbb{R}^n \mid \exists\omega \in \Omega : x = B\omega\}\,.$$

Since B^{-1} exists, one can write $\omega = B^{-1}x$. Therefore

$$B\Omega = \{x \in \mathbb{R}^n \mid B^{-1}x \in \Omega\}$$

and by substituting $\omega = B^{-1}x$ into the definition of Ω the result follows:

$$B\Omega = \{x \in \mathbb{R}^n \mid QB^{-1}x \preceq q\}\,.$$

□

Remark 3.10. No LPs are needed to compute $B\Omega$. B^{-1} can be computed using standard numerical methods in polynomial time.

It is trivial that for a given scalar $\alpha \neq 0$ and set $\Omega \triangleq \{\omega \in \mathbb{R}^n \mid Q\omega \preceq q\}$, that

$$\alpha\Omega = \left\{\omega \in \mathbb{R}^n \,\middle|\, \frac{1}{\alpha}Q\omega \preceq q\right\}\,.$$

The next result follows immediately from Proposition 3.6 and states that the number of inequalities describing the image $B\Omega$ is no more than the number of inequalities describing Ω.

Corollary 3.4 (Invertible matrix). *If B is invertible and Ω is given by N linear inequalities, then $B\Omega$ is given by (at most) N linear inequalities.*

If B is not invertible, then the following result can be used to compute the mapping $B\Omega$.

[7] In many cases the number of inputs is more than the number of *outputs*. However, even then the number of inputs very seldom exceeds the number of *states*. If the number of columns of B is greater than the number of rows of B, then the mapping $B\Omega$ can be computed via a projection.

Proposition 3.7 (Singular matrix). *[MB76] A polyhedron*

$$\Omega \triangleq \{\omega \in \mathbb{R}^m \mid Q\omega \preceq q\}$$

with $Q \in \mathbb{R}^{N\times m}$ and $q \in \mathbb{R}^N$ is given. If $B \in \mathbb{R}^{n\times m}$ is given with $m \leq n$ and $r = \operatorname{rank}(B)$, then

$$B\Omega = \{x \in \mathbb{R}^n \mid B_\perp x = 0, QB_I x \preceq q\}, \tag{3.16}$$

where the rows of $B_\perp \in \mathbb{R}^{(n-r)\times n}$ form a basis for the subspace of $\mathbb{R}^n$ which is orthogonal to the subspace spanned by the column vectors of B. The matrix $B_I \in \mathbb{R}^{m\times n}$ is any matrix with the property $B_I B = I_m$.

Proof. A proof is given here, since [MB76] does not contain one.

Define

$$\Phi \triangleq \{x \in \mathbb{R}^n \mid B_\perp x = 0, QB_I x \preceq q\}.$$

It is obvious that Ker $B_\perp$ is orthogonal to Im $B'_\perp$. Since Im $B'_\perp$ is chosen to be orthogonal to Im B, i.e. $B_\perp B = 0$, and the column vectors of $[B'_\perp \; B]$ span $\mathbb{R}^n$, it follows that

$$\operatorname{Ker} B_\perp = \operatorname{Im} B.$$

This allows one to conclude that $x \in \operatorname{Im} B$ if and only if $x \in \operatorname{Ker} B_\perp$, i.e.

$$B_\perp x = 0 \Leftrightarrow \exists \omega \in \mathbb{R}^m : x = B\omega.$$

This implies that

$$\Phi = \{x \in \mathbb{R}^n \mid \exists \omega \in \mathbb{R}^m : x = B\omega, QB_I x \preceq q\}.$$

Recalling that $B_I B = I_m$ and by substituting $x = B\omega$ into $QB_I x \preceq q$, it follows that

$$\Phi = \{x \in \mathbb{R}^n \mid \exists \omega \in \mathbb{R}^m : x = B\omega, Q\omega \preceq q\}.$$

The proof is completed by comparing this to the definition of $B\Omega$ and the fact that $Q\omega \preceq q$, i.e.

$$B\Omega = \{x \in \mathbb{R}^n \mid \exists \omega \in \mathbb{R}^m : x = B\omega, Q\omega \preceq q\}.$$

□

Remark 3.11. Standard numerical methods can be used to compute $B_\perp$ in polynomial time. The matrix B_I is the solution to a set of m^2 equalities in $m \times n$ unknowns and can therefore also be computed in polynomial time using standard numerical linear algebra methods.

This result allows one to give a bound on the number of inequalities describing $B\mathbb{U}$.

Corollary 3.5 (Singular matrix). *If $B \in \mathbb{R}^{n\times m}$, $m \leq n$, $r = \operatorname{rank}(B)$ and Ω is given by N linear inequalities, then $B\Omega$ is given by (at most) $N + 2(n - r)$ linear inequalities.*

This is a tight bound, since it is easy to find a problem where the number of non-redundant inequalities is equal to $N + 2(n - r)$. Note that Corollary 3.4 is a special case of Corollary 3.5 for the case $\operatorname{rank}(B) = n = m$.

3.5 Summary

This chapter deals with linear systems with parametric uncertainty in the pair (A, B) and disturbances acting linearly on the state. The concept of contractive sets was introduced and Theorem 3.1 was given for guaranteeing that the computation of a robust control invariant set will terminate after a finite number of steps.

Some well-known algorithms were described for implementing the three main ingredients identified in Chapter 2 for computing the robust controllable sets. Section 3.3 discussed how to compute the intersection of two polyhedra and test whether one polyhedron is a subset of another (and hence also be able to test for equality and invariance).

It was shown that the Pontryagin difference between two convex polyhedra can be computed using a finite number of LPs. This set can then be used in a projection operation or Minkowski summation to compute the robust one-step set.

Finally, Section 3.4 gave some less well-known results on subset testing and the linear mapping of a polyhedron. The latter result allows one to derive a non-conservative upper bound on the number of inequalities needed to describe the linear map.

Chapter 4

Robust Controllable Sets for Hybrid and Piecewise Affine Systems

This chapter describes how to compute the robust controllable sets for piecewise affine systems. The result is based on computing the Pontryagin difference between the union of convex polyhedra and a convex polyhedron.

4.1 Introduction

In many control applications existing today, there is a high level of interaction between subsystems with continuous dynamics and subsystems with discrete dynamics. These systems are often referred to as hybrid systems.

A system is said to be *hybrid* if it has state variables which can take on values from an *uncountable* set and state variables which can take on values from a *countable* set. State variables whose set of valuations is countable is often referred to as *discrete* and variables whose valuations come from an uncountable set, such as a Euclidean space, as *continuous*. The evolution of the system is usually given by equations which depend on both types of variables, where the dynamics can be continuous-time, discrete-time or sampled-data.

Classical control theory has mainly been concerned with continuous systems and the field of computer science has mainly been concerned with systems with discrete dynamics. As systems are becoming more complex and the interaction of continuous and discrete dynamics is increasing, it is necessary to develop tools for analysing and synthesising controllers for such systems.

At present, there are two main approaches to dealing with hybrid systems; a general, system-based approach [BBM98, LTS99] and a more specific, piecewise-affine (PWA) description [RJ00, BFM00]. Though various theoretical results regarding the undecidability of the controllability and reachability problem for general hybrid systems have been published [BT99, BT00], the reachability problem has been shown to be decidable for some classes of continuous-time, linear hybrid systems [LPY99].

An approach based on game theory is described in [LTS99] for the computation of reachable sets for continuous-time hybrid systems. The authors of [VSLS99] propose the use of quantifier elimination theory for computing robust invariant sets for discrete-time hybrid systems.

In [BTM00a] a procedure which uses mixed-integer programming is described which can be used for computing the reachable sets for discrete-time PWA systems where there is either no control input or no disturbance. The discussion in this chapter is concerned with the computation of robust controllable sets for discrete-time PWA systems, where there is both a control input and a disturbance present[1].

4.2 Mixed Logic Dynamical Systems

The MLD modelling framework, introduced in [BM99a], allows one to represent systems which can be described by interdependent physical laws, logical rules and operating constraints. It allows a large class of systems to be described such as

- constrained linear systems;
- finite state machines;
- some classes of discrete-event systems;
- systems with discrete states and/or inputs;
- nonlinear systems which can be approximated by piecewise affine functions;
- any combination of the above interacting with each another.

The general MLD form is given by

$$x_{k+1} = Ax_k + B_1 u_k + B_2 \delta_k + B_3 z_k \tag{4.1a}$$

$$y_k = Cx_k + D_1 u_k + D_2 \delta_k + D_3 z_k \tag{4.1b}$$

$$E_2 \delta_k + E_3 z_k \preceq E_1 u_k + E_4 x_k + E_5 \tag{4.1c}$$

where $x_k \in \mathbb{R}^{n_c} \times \{0,1\}^{n_l}$ are the continuous and binary state variables, $u_k \in \mathbb{R}^{m_c} \times \{0,1\}^{m_l}$ are the inputs, $y_k \in \mathbb{R}^{p_c} \times \{0,1\}^{p_l}$ the outputs, $\delta_k \in \{0,1\}^{r_l}$ and $z_k \in \mathbb{R}^{r_c}$ represent binary and continuous auxiliary variables. The latter are introduced when propositional logic statements are transformed into linear inequalities. All the constraints on the state, input, δ_k and z_k are contained in (4.1c). The description in (4.1) only appears to be linear; the variables δ_k are constrained to be binary. It

[1]The method described in this chapter is self-contained and does not rely on the results of [BTM00a]. However, it is possible to integrate some of the ideas in [BTM00a] for improving the efficiency of computing the one-step set.

is assumed that the system is completely well-posed [BM99a] in the sense that once x_k and u_k are assigned, x_{k+1} and y_k are uniquely defined.

As mentioned above, the variables δ_k and z_k are introduced when converting statements to inequalities. For example, the statement

$$z = \delta f(x)$$

is equivalent to

$$\begin{aligned} z &\leq M\delta \\ -z &\leq -m\delta \\ z &\leq f(x) - m(1-\delta) \\ -z &\leq -f(x) + M(1-\delta)\,, \end{aligned}$$

where m (M) is a lower (upper) bound of $f(x)$ over some bounded set. As another example, the propositional logic statement

$$[\delta_3 = 1] \leftrightarrow [\delta_1 = 1] \wedge [\delta_2 = 1]$$

is equivalent to

$$\begin{aligned} -\delta_1 + \delta_3 &\leq 0 \\ -\delta_2 + \delta_3 &\leq 0 \\ \delta_1 + \delta_2 - \delta_3 &\leq 1\,. \end{aligned}$$

This ability to convert statements involving logic variables continuous functions to inequalities is what gives the MLD formalism the flexibility to deal with a large range of systems.

A number of controller design techniques has been developed for MLD systems. Controller design can be achieved by

- formulating an MPC problem and solving it *on-line* using a mixed-integer quadratic program (MIQP) solver [BM99a] or via a performance-driven reachability analysis [BGT00];
- formulating an MPC problem as a multi-parametric mixed-integer linear program (mp-MILP) and computing *off-line* a piecewise linear (PWL) optimal control law [BBM00a, BBM00b, BBM00c];
- obtaining the PWA equivalent form of the MLD model [BFM00] and computing a piecewise-linear (PWL) control law by solving *off-line* a set of linear matrix inequalities (LMIs) [MFM00]. This method is based on that of [RJ00] of computing piecewise quadratic Lyapunov functions, but for discrete-time, rather than continuous-time PWA systems.

For further details on propositional logic and how hybrid systems can be modelled in the MLD framework, the reader is referred to the references cited in this section. The reason for introducing MLD systems here is because they have an equivalent PWA description.

4.3 Equivalence Between MLD and Piecewise Affine Systems

In [BFM00] it is shown in a constructive manner that MLD systems are formally equivalent to PWA systems. This fact allows one to use results which have been developed for PWA systems and apply them to a large class of linear hybrid systems which can be modelled as MLD systems, and vice versa. The PWA equivalent form allows one to develop observability and controllability tests for hybrid systems [BFM00], synthesise controllers [MFM00], perform a verification and reachability analysis [BM99b, BTM00a], compute controllable, stabilisable and admissible sets [BTM00a], construct a state estimator and fault detector [BMM99, FMM00] and identify a model from input-output data [FMLM00].

Piecewise affine systems are described by the state-space equations:

$$x_{k+1} = A^i x_k + B^i u_k + E^i w_k + f^i, \quad \text{if } \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathcal{X}_i\,, \tag{4.2}$$

where $\{\mathcal{X}_i\}_{i=0}^{s-1}$ is a *polyhedral* partition[2] of the state and input space. Each $\mathcal{X}_i$ is given by

$$\mathcal{X}_i \triangleq \left\{ \begin{bmatrix} x_k \\ u_k \end{bmatrix} \middle| Q^i \begin{bmatrix} x_k \\ u_k \end{bmatrix} \preceq q^i \right\} \tag{4.3}$$

and the f^i are suitable constant vectors. Each subsystem defined by the 4-tuple (A^i, B^i, E^i, f^i), $i \in \{0, 1, \dots, s-1\}$, is termed a *component* of the PWA system (4.2). If all the $f^i = 0$, then the system (4.2) is said to be piecewise linear (PWL) and if all $A^i = 0$, $B^i = 0$, $E^i = 0$ then the system is piecewise constant. It is still required that the states and inputs satisfy $\mathbb{X}$ and $\mathbb{U}$. It is assumed that $0 \in \mathbb{W}$ and that $\mathbb{W}$ is a compact polyhedron (polytope).

Remark 4.1. A disturbance $w_k \in \mathbb{W}$ has been added to the description of the PWA system. The disturbance affects the state via the matrix E^i, but the partitioning still only depends on the state and the input. In [BFM00, BM99b, BTM00a, BTM00b] it is assumed that there is no control input $B^i = 0$ or that there is no disturbance $E^i = 0$.

Remark 4.2. Note that even though $x_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$ and $w_k \in \mathbb{R}^q$, this model is general enough to represent the discrete variables present in the MLD model; the matrix update equations are well-defined when converting from MLD to PWA form and the state evolution will be well-defined for well-defined initial conditions. For example, consider the simple PWA system

$$x_{k+1} = \begin{cases} 16 & \text{if } x_k \geq 8 \\ 2x_k & \text{if } 0 \leq x_k \leq 8 \\ 0 & \text{if } x_k \leq 0 \end{cases} .$$

If $x_0 \in \{0, 1, 2, 4, 8, 16\}$, then future evolutions of the state variable will take on values from the same discrete set.

[2]The interiors of all the $\mathcal{X}_i$ are pair-wise disjoint and the union $\bigcup_{i=0}^{s-1} \mathcal{X}_i$ covers a polyhedral region of interest in the state and input space [BBM00c, Def. 4].

The common boundaries of $\mathcal{X}_i$ are called *guard-lines*. Without additional continuity assumptions on the PWA system, (4.2) is not well-posed in general, since the function is multiply defined on the guard-lines. This will not affect the computation of the Pontryagin difference as discussed in Section 4.5.1, but might affect the computation of the one-step set as discussed in Section 4.5.2. This is a technical issue which can be avoided in practice. For example, given the ill-posed system

$$x_{k+1} = \begin{cases} 0.8x_k & \text{if } x_k \geq 1 \\ 0.5x_k & \text{if } x_k \leq 1 \end{cases}$$

it can be seen that if $x_k = 1$, then $x_{k+1} \in \{0.8, 0.5\}$. By redefining the system as

$$x_{k+1} = \begin{cases} 0.8x_k & \text{if } x_k \geq 1 \\ 0.5x_k & \text{if } x_k \leq 1 - \epsilon \end{cases}$$

where $\epsilon > 0$ is a sufficiently small number (typically machine precision), the state evolution is well-defined. When converting a well-posed MLD system to PWA form, the resulting description will have a similar form as the latter, thereby guaranteeing that the evolution of the states of the PWA system is well-defined.

See [BFM00] for more details on how to convert from MLD form to PWA form. Converting from PWA form to MLD form is trivial and is discussed in [BM99a].

4.4 Verification and Reachability Analysis of PWA Systems

The following problem is addressed in [BTM00c] (see Figure 4.1).

Problem 4.1 (Reachability analysis problem). *Given a system in MLD or PWA form and a set of initial conditions $\mathbb{X}_0$, a collection of disjoint*[3] *target sets $\mathcal{Z}_1, \mathcal{Z}_2, \dots, \mathcal{Z}_L$, a bounded set of inputs $\mathbb{U}$ and a time horizon T,*

1. *determine whether there exists an input sequence $\{u_k \in \mathbb{U}\}_0^{t-1}$ such that $\mathcal{Z}_j$ is reachable from $\mathbb{X}_0$ within $t \leq T$ steps;*
2. *if such a sequence exists, then compute the subset of initial conditions $\mathbb{X}_{\mathcal{Z}_j}$ of $\mathbb{X}_0$ from which $\mathcal{Z}_j$ can be reached within T steps;*
3. *for a given $x_0 \in \mathbb{X}_{\mathcal{Z}_j}$, compute an input sequence $\{u_k \in \mathbb{U}\}_0^{t-1}$, where $t \leq T$, which drives x_0 to any $x_t \in \mathcal{Z}_j$.*

Remark 4.3. Note that it is assumed in this problem that there is no disturbance in (4.2), i.e. $E^i = 0$. A similar problem can also be formulated for the case when there is a disturbance but no control input, i.e. $B^i = 0$ [BTM00a]. The latter problem is typically known as the *verification* problem [BM99b].

[3] Strictly speaking, it is not necessary for the $\mathcal{Z}_j$ to be disjoint.

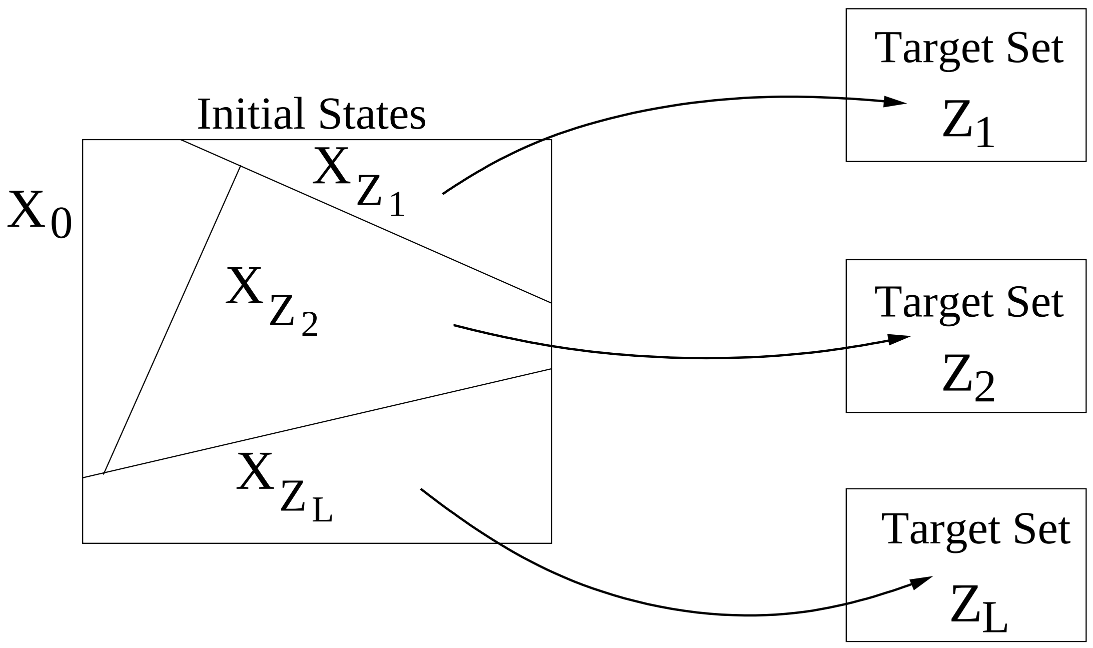

Figure 4.1: Reachability analysis problem

Without wanting to get into too much detail, the point that is being made is that the algorithm presented in [BM99b, BTM00a, BTM00c] can be used with very little modification to compute many of the sets described in Chapter 2. In addition, the iterative procedure of Algorithm 2.1 is not necessary and the sets can be computed more efficiently in a single pass with the algorithm of [BM99b, BTM00c, BTM00a]. However, this algorithm is applicable only if the PWA system has either no control input or no disturbance.

When there are both control inputs and disturbances present and one would like to compute the *robust controllable* sets, a new algorithm is needed. An iterative procedure based on Algorithm 2.1 and the computation of the Pontryagin difference of non-convex sets is discussed next.

4.5 Robust Controllable Sets for PWA Systems

As discussed in Section 2.10, if the system is of the form

$$x_{k+1} = f_{xu}(x_k, u_k) + f_w(w_k)\,, \tag{4.4}$$

as is the case with PWA systems, and one defines

$$\mathbb{D} \triangleq f_w(\mathbb{W})\,, \tag{4.5}$$

then the *robust* one-step set $\tilde{\mathcal{Q}}(\Omega)$ is equal to the *nominal* one-step set $\mathcal{Q}(\Omega \sim \mathbb{D})$, i.e.

$$\tilde{\mathcal{Q}}(\Omega) = \mathcal{Q}(\Omega \sim \mathbb{D}) \triangleq \left\{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}, x_{k+1} \in \Omega \sim \mathbb{D} : x_{k+1} = f_{xu}(x_k, u_k)\right\} . \tag{4.6}$$

This implies that in order to develop an algorithm for computing robust controllable sets for a given PWA system, it is sufficient to develop procedures for computing the Pontryagin difference $\Omega \sim \mathbb{D}$, the nominal one-step set $\mathcal{Q}(\Omega \sim \mathbb{D})$ and the intersection $\mathcal{Q}(\Omega \sim \mathbb{D}) \cap \Omega$. Given a target set $\mathbb{T}$, the iterative procedure of Algorithm 2.1 can then be used to compute the robust controllable set $\tilde{\mathcal{K}}_i(\mathbb{X}, \mathbb{T})$.

For PWA systems, even if $\mathbb{T}$ is a convex polyhedron, all the $\tilde{\mathcal{K}}_i(\mathbb{X}, \mathbb{T})$ are not guaranteed to be convex polyhedra. As a result, the algorithms described in Chapter 3 cannot be applied directly. However, if $\mathbb{T}$ is a convex polyhedron, then all the $\tilde{\mathcal{K}}_i(\mathbb{X}, \mathbb{T})$ can always be described as the union of a number of convex polyhedra. This chapter presents the building blocks of an algorithm for computing the robust controllable sets for PWA systems. Methods for computing the Pontryagin difference $\Omega \sim \mathbb{D}$ and the nominal one-step set $\mathcal{Q}(\Omega \sim \mathbb{D})$, where Ω is given as the union of a finite number of convex polyhedra, will be described in Sections 4.5.1 and 4.5.2.

Before proceeding, the following assumption is made.

Assumption 4.1. $E^i = E, \forall i = 0, \dots, s-1$.

In other words,

$$0 \in \mathbb{D} = E\mathbb{W} .$$

4.5.1 Pontryagin Difference

This section describes a method for computing the Pontryagin difference $\Omega \sim \mathbb{D}$, where Ω is a (possibly non-convex) set which can be described as the union of a set of convex polyhedra, i.e.

$$\Omega \triangleq \bigcup_{j=1}^{N} \Omega_j , \tag{4.7}$$

where Ω_j are convex polyhedra.

Recall the definition of the Pontryagin difference:

$$\Omega \sim \mathbb{D} \triangleq \{x_k \in \mathbb{R}^n \mid x_k + d_k \in \Omega, \forall d_k \in \mathbb{D}\} .$$

The following result states that if Ω is given by the union of disjoint sets, then the Pontryagin difference is given by the union of the Pontryagin difference of each $\Omega_j \sim \mathbb{D}$. If this is the case, then Proposition 3.5 can be used to compute all the $\Omega_j \sim \mathbb{D}$, since all the Ω_j are convex polyhedra.

Proposition 4.1. *If* $0 \in \mathbb{D}$ *and* $\Omega = \bigcup_{j=1}^{N} \Omega_j$, *where all the* Ω_j *are pairwise disjoint, then* $\Omega \sim \mathbb{D} = \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$.

Proof. If $x_k \in \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$, then $x_k \in \Omega_j \sim \mathbb{D}$ for some Ω_j. This implies that $x_k + d_k \in \Omega_j, \forall d_k \in \mathbb{D}$. Combining this result with the fact that $\Omega_j \subseteq \Omega$, it follows that $x_k + d_k \in \Omega, \forall d_k \in \mathbb{D}$ and hence $x_k \in \Omega \sim \mathbb{D}$.

This allows one to conclude that $\bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D}) \subseteq \Omega \sim \mathbb{D}$. The fact that $\Omega \sim \mathbb{D} \subseteq \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$ will be shown by contradiction.

Assume that $\Omega \sim \mathbb{D} \nsubseteq \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$.

If $x_k \in (\Omega \sim \mathbb{D}) \setminus \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$, then either $x_k \in \Omega_j \setminus \Omega_j \sim \mathbb{D}$ for some Ω_j or $x_k \notin \Omega$.

If $x_k \notin \Omega$, then $d_k = 0$ results in $x_k + d_k \notin \Omega$. This implies that $\Omega \sim \mathbb{D} \subseteq \Omega$ and hence that the former is the only other possible case. Assuming this is true, i.e. $x_k \in \Omega_j \setminus \Omega_j \sim \mathbb{D}$ for some Ω_j, then one can always choose a $d_k \in \mathbb{D}$ such that $x_k + d_k \in \partial\Omega$.

However, there also exists an $\epsilon > 0$ such that it is possible to choose a $d_k \in \mathbb{D}$ such that $d_k + \epsilon \in \mathbb{D}$ and $x_k + d_k + \epsilon \notin \Omega$. This follows from the fact that all the Ω_j are disjoint, i.e. $\forall j, x_k \in \partial\Omega_j$, there exists an $\epsilon > 0$ such that $x_k + \epsilon \notin \Omega$. The case $x_k \in \Omega_j \setminus \Omega_j \sim \mathbb{D}$ is therefore also not possible and hence $(\Omega \sim \mathbb{D}) \setminus \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D}) = \emptyset$.

This implies that the assumption $\Omega \sim \mathbb{D} \nsubseteq \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$ is false, thereby concluding the proof. □

However, in general all the Ω_j are not disjoint and as a result, $\Omega \sim \mathbb{D} \neq \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D})$, but

$$\Omega \sim \mathbb{D} \supseteq \bigcup_{j=1}^{N}(\Omega_j \sim \mathbb{D}) .$$

From this point on, it is assumed that there exist $\Omega_i, \Omega_j, i \neq j$ such that $\Omega_i \cap \Omega_j \neq \emptyset$.

Before proceeding to describe the algorithm, the following set is defined:

$$\mathcal{Q}_{\mathbb{D}}(\Omega) \triangleq \left\{x_k \in \mathbb{R}^n \mid \exists d_k \in \mathbb{D} : x_k + d_k \in \Omega\right\} . \tag{4.8}$$

Proposition 4.2. *The Pontryagin difference is given by*

$$\Omega \sim \mathbb{D} = \left[\mathcal{Q}_{\mathbb{D}}\left(\Omega^c\right)\right]^c . \tag{4.9a}$$

Proof. From the definition of the Pontryagin difference it follows that

$$(\Omega \sim \mathbb{D})^c = \{x_k \in \mathbb{R}^n \mid \exists d_k \in \mathbb{D} : x_k + d_k \in \Omega^c\} .$$

This set is the same set as $\mathcal{Q}_{\mathbb{D}}(\Omega^c)$. The proof is concluded by taking the complement. □

Remark 4.4. Note that this result does not require that $0 \in \mathbb{D}$.

Because of the earlier assumption that $0 \in \mathbb{D}$, the Pontryagin difference $\Omega \sim \mathbb{D}$ can therefore be computed by determining all states in Ω for which a disturbance exists which will take the system to Ω^c and then taking the complement.

To summarise, given a set Ω, which is the union of a finite number of convex polyhedra, and a compact polyhedron $\mathbb{D}$, the Pontryagin difference can be computed as follows:

1. Given
$$\Omega \triangleq \bigcup_{j=1}^{N} \Omega_j \ ,$$
compute the complement Ω^c as described in Appendix D. Ω^c is then given by
$$\Omega^c \triangleq \bigcup_{i=1}^{M} \Phi_i \ ,$$
where each Φ_i is an open polyhedron;

2. Given the polytope $\mathbb{D}$, compute
$$\begin{aligned} \mathcal{Q}_{\mathbb{D}}(\Omega^c) &= \mathcal{Q}_{\mathbb{D}}\left(\bigcup_{i=1}^{M} \Phi_i\right) \\ &= \bigcup_{i=1}^{M} \mathcal{Q}_{\mathbb{D}}\left(\Phi_i\right) \ . \end{aligned}$$
The last step is a consequence of Proposition 2.2. As in Chapter 3, each $\mathcal{Q}_{\mathbb{D}}\left(\Phi_i\right)$ can be computed using a projection algorithm or as the Minkowski sum $\mathcal{Q}_{\mathbb{D}}\left(\Phi_i\right) = \Phi_i \oplus (-\mathbb{D})$;

3. Compute $[\mathcal{Q}_{\mathbb{D}}(\Omega^c)]^c$ as described in Appendix D. This gives the Pontryagin difference as the union of a finite number of closed polyhedra:
$$\Omega \sim \mathbb{D} \triangleq \bigcup_{j=1}^{L} \mathcal{Z}_j \ .$$

Remark 4.5. If the Ω_j are closed sets, then the Φ_i are all open sets. Open sets are difficult to work with in computers with finite precision arithmetic. When implementing the algorithm in a computer, the Φ_i can be substituted with their closures to simplify the coding and improve the numerical condition. Similarly, when computing $[\mathcal{Q}_{\mathbb{D}}(\Omega^c)]^c$, the closures of the sets could be used without affecting the result.

Remark 4.6. Note that the procedure described here is not dependent on the system dynamics $x_{k+1} = f_{xu}(x_k, u_k)$ and that $E^i = E$. This implies that the Pontryagin difference is not dependent on the partitioning of the state and input space. The Pontryagin difference $\Omega \sim \mathbb{D}$ for a given Ω is unique, even if the PWA system is not well-posed.

4.5.2 Computation of the One-step Robust Controllable Set

When computing the one-step set, the result is dependent on the system dynamics $x_{k+1} = f_{xu}(x_k, u_k)$ and hence also on the partitioning of the state and input space. This section describes how to compute the one-step set for well-defined PWA systems, given the Pontryagin difference.

If the Pontryagin difference is given as the union of convex polyhedral sets, as in the previous section,

$$\Omega \sim \mathbb{D} \triangleq \bigcup_{j=1}^{L} \mathcal{Z}_j \,, \tag{4.10}$$

then Proposition 2.2 allows one to write

$$\tilde{\mathcal{Q}}(\Omega) = \mathcal{Q}(\Omega \sim \mathbb{D}) = \bigcup_{j=1}^{L} \mathcal{Q}(\mathcal{Z}_j)$$

Here the set $\mathcal{Q}(\mathcal{Z}_j)$ has to be computed while taking the partitioning $\{\mathcal{X}_i\}_{i=0}^{s-1}$ into account. As a result

$$\tilde{\mathcal{Q}}(\Omega) = \mathcal{Q}(\Omega \sim \mathbb{D}) = \bigcup_{j=1}^{L} \bigcup_{i=0}^{s-1} \mathcal{Q}_i(\mathcal{Z}_j) \,,$$

where

$$\mathcal{Q}_i(\mathcal{Z}_j) \triangleq \left\{ x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}, x_{k+1} \in \mathcal{Z}_j : [x_k', u_k']' \in \mathcal{X}_i, x_{k+1} = A^i x_k + B^i u_k \right\} \,. \tag{4.11}$$

In principle an algorithm which is based on this expression will work, but it will require the computation of $s \times L$ one-step sets at each step. This could be computationally expensive. To reduce the number of computations, one can compute which combinations of $\mathbb{X} \cap \mathcal{X}_i$ are non-empty[4]. The one-step robust controllable set is computed by noting that

$$\begin{aligned}
\tilde{\mathcal{K}}_1(\mathbb{X}, \Omega) &= \mathcal{K}_1(\mathbb{X}, \Omega \sim \mathbb{D}) \\
&= \mathcal{Q}(\Omega \sim \mathbb{D}) \cap \mathbb{X} \\
&= \bigcup_{j=1}^{L} \mathcal{Q}(\mathcal{Z}_j) \cap \mathbb{X} \\
&= \bigcup_{j=1}^{L} \bigcup_{i=0}^{s-1} \mathcal{Q}_i(\mathcal{Z}_j) \cap \mathbb{X} \\
&= \bigcup_{j=1}^{L} \bigcup_{\{i \mid \mathbb{X} \cap \mathcal{X}_i \neq \emptyset\}} \mathcal{Q}_i(\mathcal{Z}_j) \cap \mathbb{X} \,.
\end{aligned}$$

The $\mathcal{Q}_i(\mathcal{Z}_j)$ can be computed using a projection algorithm The set $\tilde{\mathcal{K}}_1(\mathbb{X}, \Omega)$ is a non-convex, possibly disjoint set, given by the union of a finite number of convex polyhedra.

[4]Since $\mathbb{X} \subseteq \mathbb{R}^n$ and $\mathcal{X}_i \subseteq \mathbb{R}^{n+m}$, with a slight abuse of notation $\mathbb{X} \cap \mathcal{X}_i \triangleq \mathbb{X} \cap \left\{ x_k \mid \exists u_k : [x_k' \; u_k']' \in \mathcal{X}_i \right\}$.

Remark 4.7. If $\Omega \sim \mathbb{D}$ is given by a set of convex polyhedral sets as above, then the one-step robust controllable set can also be computed using the algorithm in [BM99b, BTM00c, BTM00a]. The algorithm is initialised with $\mathbb{X}_0 = \mathbb{X}$ and target set $\mathbb{T} = \Omega \sim \mathbb{D}$. The output from the algorithm will be $\mathcal{K}_1(\mathbb{X}, \Omega \sim \mathbb{D})$. Some benefit in efficiency might be obtained by exploiting the MLD structure in order to determine which $\mathcal{Q}_i(\mathcal{Z}_j)$ need to be computed. This might involve setting up a mixed-integer feasibility program as in [BM99b, BTM00c, BTM00a]. Furthermore, a convexity recognition algorithm such as the one described in [BFT00] could also be used to reduce the number of polyhedra needed to describe the resulting robust controllable sets.

To summarise, the one-step robust controllable set $K_1(\mathbb{X}, \Omega)$ for a PWA system can be computed as follows:

1. Compute $\mathcal{I} \triangleq \{i \mid \mathbb{X} \cap \mathcal{X}_i \neq \emptyset\}$.
2. Given
$$\Omega \sim \mathbb{D} \triangleq \bigcup_{j=1}^{L} \mathcal{Z}_j$$
and using an appropriate method from Chapter 3, compute $\mathcal{Q}_i(\mathcal{Z}_j) \cap \mathbb{X}$ for $j = 1, \ldots, L$ and $i \in \mathcal{I}$.

The robust controllable set is given by
$$\tilde{\mathcal{K}}_1(\mathbb{X}, \Omega) = \bigcup_{j=1}^{L} \bigcup_{i \in \mathcal{I}} \mathcal{Q}_i(\mathcal{Z}_j) \cap \mathbb{X}\,. \tag{4.12}$$

4.6 Example

Consider the following well-posed PWA system
$$x_{k+1} = \begin{cases} A^1 x_k + B^1 u_k + E^1 w_k, & \text{if } \begin{bmatrix} 1 & 1 \end{bmatrix} x_k \leq 0 \\ A^2 x_k + B^2 u_k + E^2 w_k, & \text{if } \begin{bmatrix} -1 & -1 \end{bmatrix} x_k \leq 0 \end{cases}$$
with
$$A^1 = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, A^2 = \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}, B^1 = B^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, E^1 = E^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
and the constraints
$$\mathbb{X} \triangleq \{x \in \mathbb{R}^2 \mid \|x\|_\infty \leq 10\}$$
$$\mathbb{U} \triangleq \{u \in \mathbb{R}^2 \mid \|u\|_\infty \leq 5\}$$
$$\mathbb{W} \triangleq \{w \in \mathbb{R}^2 \mid \|w\|_\infty \leq 1\}\,.$$

The polyhedral partition of the state and input space is

$$\mathcal{X}_0 = \left\{ \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathbb{R}^4 \middle| \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \leq 0 \right\}$$
$$\mathcal{X}_1 = \left\{ \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathbb{R}^4 \middle| \begin{bmatrix} -1 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \leq 0 \right\} .$$

The one-step robust controllable set $\tilde{\mathcal{K}}_1(\mathbb{X}, \Omega)$ is to be calculated, where

$$\Omega \triangleq \bigcup_{j=1}^{2} \Omega_j ,$$

with

$$\Omega_1 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} 0 & 1 \\ -1 & 0 \\ 0 & -1 \\ 1 & 0 \end{bmatrix} x_k \preceq \begin{bmatrix} 6 \\ 5 \\ 4 \\ 0 \end{bmatrix} \right\}$$
$$\Omega_2 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} -1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} x_k \preceq \begin{bmatrix} 1 \\ 5 \\ 4 \end{bmatrix} \right\}$$

Ω is shown in Figure 4.2.

4.6.1 The Pontryagin Difference

The complement of Ω is computed as in Appendix D. A description of this set is given by

$$\Omega^c \triangleq \bigcup_{i=1}^{5} \Phi_i ,$$

with

$$\Phi_1 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} 1 & 0 \end{bmatrix} x_k < -5 \right\}$$
$$\Phi_2 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} 0 & -1 \end{bmatrix} x_k < -6 \right\}$$
$$\Phi_3 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} 0 & 1 \end{bmatrix} x_k < -4 \right\}$$
$$\Phi_4 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} -1 & 0 \end{bmatrix} x_k < -5 \right\}$$
$$\Phi_5 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} 0 & 1 \\ -1 & 0 \\ 1 & -1 \end{bmatrix} x_k \prec \begin{bmatrix} 6 \\ 0 \\ -1 \end{bmatrix} \right\} .$$

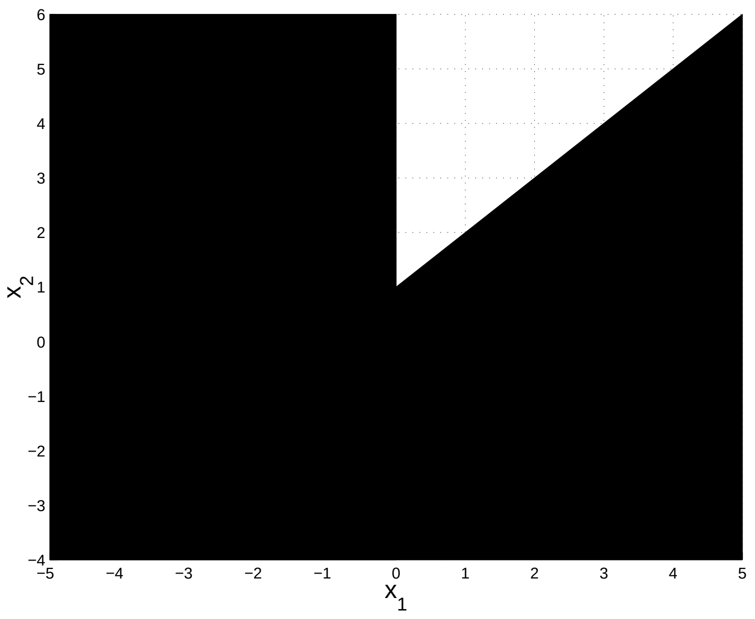

Figure 4.2: The shaded area represents $\Omega = \bigcup_{j=1}^{2} \Omega_j$

Ω^c is shown in Figure 4.3.

With $\mathbb{D} = \mathbb{W}$, the sets

$$\mathcal{Q}_{\mathbb{D}}(\Phi_i) = \Phi_i \oplus (-\mathbb{D}), \quad i = 1, 2, 3, 4, 5$$

are computed in order to obtain

$$\mathcal{Q}_{\mathbb{D}}(\Omega^c) = \bigcup_{i=1}^{5} \mathcal{Q}_{\mathbb{D}}\left(\Phi_i\right) .$$

The complement $[\mathcal{Q}_{\mathbb{D}}(\Omega^c)]^c$ is computed as in Appendix D, giving the Pontryagin difference

$$\Omega \sim \mathbb{D} \triangleq \bigcup_{j=1}^{3} \mathcal{Z}_j ,$$

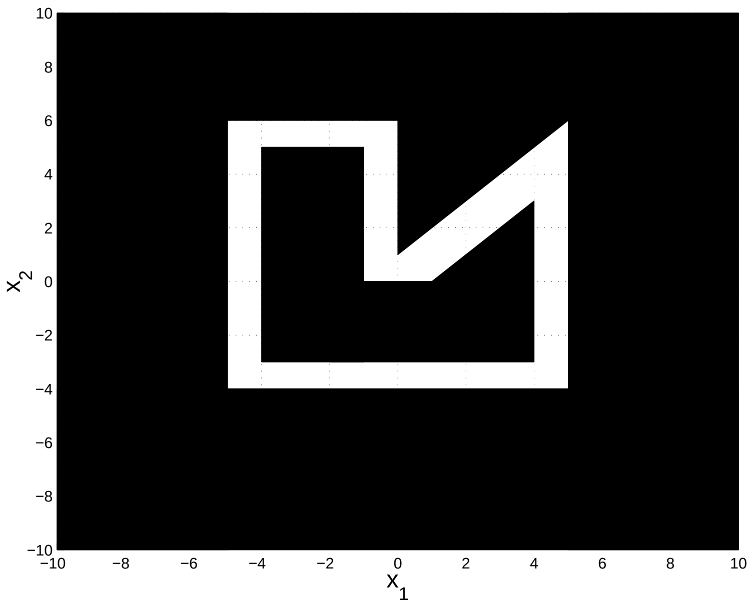

Figure 4.3: The outer, shaded area represents $\Omega^c = \bigcup_{i=1}^{5} \Phi_i$, while the inner, shaded area represents the Pontryagin difference $\Omega \sim \mathbb{D} = \bigcup_{j=1}^{3} \mathcal{Z}_j$

with

$$\mathcal{Z}_1 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} -1 & 0 \\ 0 & 1 \\ 0 & -1 \\ 1 & 0 \end{bmatrix} x_k \preceq \begin{bmatrix} 4 \\ 5 \\ 3 \\ -1 \end{bmatrix} \right\}$$

$$\mathcal{Z}_2 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} 0 & 1 \\ -1 & 0 \\ 1 & -1 \end{bmatrix} x_k \preceq \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \right\}$$

$$\mathcal{Z}_3 = \left\{ x_k \in \mathbb{R}^2 \middle| \begin{bmatrix} -1 & 1 \\ 1 & 0 \\ 0 & -1 \end{bmatrix} x_k \preceq \begin{bmatrix} -1 \\ 4 \\ 3 \end{bmatrix} \right\}.$$

$\Omega \sim \mathbb{D}$ is shown in Figure 4.3.

4.6.2 The One-step Set

Since the partition $\{\mathcal{X}_0, \mathcal{X}_1\}$ is not dependent on u_k, the Minkowski sum can be used to compute the six $\mathcal{Q}_i(\mathcal{Z}_j)$. They are given by

$$\mathcal{Q}_0(\mathcal{Z}_1) = \left\{ x_k \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -1 & 0 \\ 1 & 1 \\ -3 & -4 \\ 1 & 2 \end{bmatrix} x_k \preceq \begin{bmatrix} 10 \\ 0 \\ 8 \\ 4 \end{bmatrix} \right\}$$

$$\mathcal{Q}_0(\mathcal{Z}_2) = \left\{ x_k \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -1 & 0 \\ 1 & 1 \\ 3 & 4 \\ -1 & -2 \\ 1 & 2 \\ -3 & -4 \end{bmatrix} x_k \preceq \begin{bmatrix} 10 \\ 0 \\ 5 \\ 6 \\ 6 \\ 7 \end{bmatrix} \right\}$$

$$\mathcal{Q}_0(\mathcal{Z}_3) = \left\{ x_k \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -1 & 0 \\ 1 & 1 \\ -3 & -4 \\ 3 & 4 \\ -1 & -2 \end{bmatrix} x_k \preceq \begin{bmatrix} 10 \\ 0 \\ 8 \\ 8 \\ 7 \end{bmatrix} \right\}$$

and

$$\mathcal{Q}_1(\mathcal{Z}_1) = \left\{ x_k \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 0 \\ -1 & -1 \\ 5 & 6 \\ -5 & -6 \\ 3 & 4 \end{bmatrix} x_k \preceq \begin{bmatrix} 10 \\ 0 \\ 10 \\ 8 \\ 4 \end{bmatrix} \right\}$$

$$\mathcal{Q}_1(\mathcal{Z}_2) = \left\{ x_k \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 0 \\ -1 & -1 \\ 5 & 6 \\ -3 & -4 \end{bmatrix} x_k \preceq \begin{bmatrix} 10 \\ 0 \\ 5 \\ 6 \end{bmatrix} \right\}$$

$$\mathcal{Q}_1(\mathcal{Z}_3) = \left\{ x_k \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 0 \\ -1 & -1 \\ 5 & 6 \\ -3 & -4 \end{bmatrix} x_k \preceq \begin{bmatrix} 10 \\ 0 \\ 8 \\ 7 \end{bmatrix} \right\}.$$

The robust one-step controllable set

$$\tilde{\mathcal{K}}_1(\mathbb{X}, \Omega) = \bigcup_{j=1}^{3} \bigcup_{i=0}^{1} \mathcal{Q}_i(\mathcal{Z}_j) \cap \mathbb{X}$$

is shown in Figure 4.4.

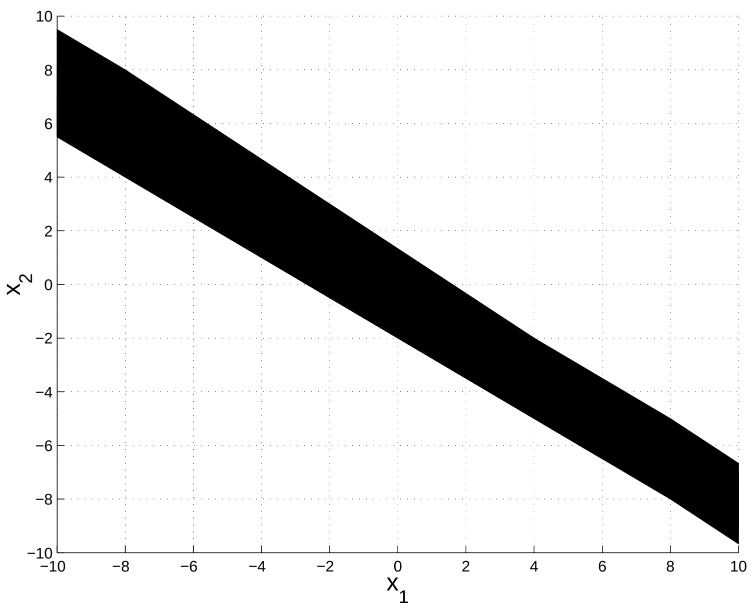

Figure 4.4: The shaded area represents the robust one-step controllable set $\tilde{\mathcal{K}}_1(\mathbb{X}, \Omega)$

4.7 Summary

This chapter started by briefly describing MLD systems. The reason for introducing MLD systems in this chapter is that a large class of hybrid systems can be described using the MLD formalism. Furthermore, MLD systems are equivalent to PWA systems. This implies that if one can compute robust controllable sets for one class of systems, then the same sets can be used in the analysis and synthesis of controllers for the equivalent system. This chapter was concerned with computing the robust controllable sets for PWA systems.

In general, the sets are non-convex at each stage of the iteration in the computation of the robust controllable sets. As a matter of fact, they are given by the union of a set of convex polyhedra.

The main building block in the computation of the robust controllable set is the computation of the Pontryagin difference of the union of convex polyhedra and the disturbance set. The computation of the Pontryagin difference involves computing the complement of the union of a set of polyhedra twice. This appears to be the main bottleneck of the proposed approach.

Nevertheless, it is possible to proceed and complete the computation of the robust controllable set using the results obtained in the previous two chapters. Though this chapter only describes the computation of one step of the algorithm for the robust controllable sets, by repetitively applying the algorithm one can compute all the robust controllable sets. With the appropriate choice of target set, one can also compute the maximal robust control invariant and maximal robust stabilisable sets, provided they are finitely determined.

Part II

Nonlinear Model Predictive Control

Chapter 5

Nominal Feasibility in Model Predictive Control

The nominal MPC regulation problem is introduced. The feasible set of the MPC scheme is defined and the causes of infeasibility in MPC are given. The notion of strong feasibility is introduced and a new sufficient condition is derived for guaranteeing strong feasibility. The effect of the choice of horizons and terminal constraint set on the feasible set and feasibility of the MPC problem is investigated.

5.1 Introduction

This chapter briefly introduces Model Predictive Control (MPC) and proceeds to address some nominal feasibility issues related to solving the MPC problem.

It is assumed that there are no disturbances present, i.e.

$$x_{k+1} = f(x_k, u_k) \, .$$

The MPC control action is determined by solving the following finite horizon optimal control problem at each time step:

Problem 5.1 (Nominal MPC Regulation Problem). *Solve*

$$V^*(x_k) = \min_{\pi_k^N} F(\hat{x}_{P|k}) + \sum_{i=0}^{P-1} L(\hat{x}_{i|k}, \hat{u}_{i|k}) \tag{5.1}$$

subject to

$$\hat{x}_{l+1|k} = f(\hat{x}_{l|k}, \hat{u}_{l|k}), \qquad \hat{x}_{0|k} = x_k \tag{5.2a}$$

$$\hat{x}_{l|k} \in \mathbb{X}, \quad \hat{u}_{l|k} \in \mathbb{U}, \qquad l = 0, \dots, P-1 \tag{5.2b}$$

$$\hat{u}_{l|k} = h(\hat{x}_{l|k}), \qquad l = N, \dots, P-1 \tag{5.2c}$$

$$\hat{x}_{P|k} \in \mathbb{T} \subseteq \mathbb{X}. \tag{5.2d}$$

The decision variable in the MPC problem is the control sequence

$$\pi_k^N \triangleq \left[\hat{u}'_{0|k}, \hat{u}'_{1|k}, \dots, \hat{u}'_{N-1|k}\right]' \tag{5.3}$$

and it is assumed that no disturbances are present. The notation $\hat{x}_{l|k}$ and $\hat{u}_{l|k}$ denote estimates of the state and input at time $k + l$. The variables N and P are the control and prediction horizons, respectively, and it is assumed that $P \geq N \geq 0$. Note that if $P = N$, then constraint (5.2c) is removed. $\mathbb{T}$ is the terminal constraint set and $0 \in \mathbb{T} \subseteq \mathbb{X}$.

N, P, $F(\cdot)$, $L(\cdot,\cdot)$, $h(\cdot)$ and $\mathbb{T}$ are the design variables and $f(\cdot,\cdot)$, $\mathbb{X}$ and $\mathbb{U}$ are fixed. $\mathbb{X}$ and $\mathbb{T}$ are closed and $\mathbb{U}$ is compact. It is assumed that $(0,0) \in \mathbb{X}^\circ \times \mathbb{U}^\circ$ and $0 = f(0,0)$. The aim of the control action is to regulate the states and control inputs to $(0,0)$.

Since the optimisation is over a finite horizon, in the design of the terminal cost $F(\hat{x}_{P|k})$ and the stage cost $L(\hat{x}_{i|k}, \hat{u}_{i|k})$, it is assumed that $\hat{u}_{l|k} = h(\hat{x}_{l|k})$ is a Lyapunov stabilising control law defined on $\mathbb{X}$ that will be applied on the infinite horizon for $l \geq P$. It is assumed that $L(\cdot,\cdot)$ is a continuous, non-negative, time-invariant function defined on $\mathbb{X} \times \mathbb{U}$ and $F(\cdot)$ is a continuous, non-negative, time-invariant function defined on $\mathbb{X}$.

At each time instant k, the current state x_k of the system is measured. The new control input to be applied to the system is the first element of the (not necessarily optimal) solution π_k^{N*} to Problem 5.1, i.e.

$$\kappa(x_k) \triangleq \hat{u}^*_{0|k}\ .$$

Here $\kappa(x)$ implicitly defines the MPC control law, with the closed-loop system being given by $x_{k+1} = f(x_k, \kappa(x_k))$. Feedback is incorporated into MPC by repeating the state measurement and control input calculation at the next time instant. Due to the finite prediction horizon, the computed control at the next time instant $\hat{u}^*_{0|k+1}$ is in general not equal to the previously computed $\hat{u}^*_{1|k}$.

Remark 5.1. Note that the constraint $\hat{x}_{0|k} \in \mathbb{X}$ is included in (5.2). Strictly speaking, this is not necessary since this constraint does not affect the resulting control action, but only affects the region of feasibility. The constraint can be removed to enlarge the region of feasibility of the MPC controller. However, by including this constraint the notation and presentation of the results in this chapter are simplified.

Remark 5.2. The above formulation is the one most commonly adopted in the literature and is similar to those of [May00, MRRS00], but with a prediction horizon which is allowed to be different from the control horizon. There is some benefit to be gained by including a separate prediction horizon, such as increasing the size of the feasible set if $\mathbb{T}$ is a control invariant set. Section 5.8 discusses the effect of the prediction horizon on the properties of the feasible set.

5.2 Nominal Feasibility in MPC

Often one is interested in obtaining the set of states for which the MPC problem is feasible. Before proceeding, it is necessary to assume that the set of ordered pairs (x_k, π_k^N) which satisfy the constraints in (5.2), is non-empty.

The feasible set[1] $\mathbb{X}_F$ of the MPC problem is the set of states x_k for which a feasible control sequence π_k^N to Problem 5.1 exists, i.e.

$$\mathbb{X}_F(\mathbb{T}, N, P) \triangleq \left\{x_k \in \mathbb{R}^n \mid \exists \pi_k^N : (x_k, \pi_k^N) \text{ satisfies (5.2)}\right\} . \tag{5.4}$$

$\mathbb{X}_F$ can therefore be interpreted as the orthogonal projection of (5.2) onto the first coordinate. Note also that the input admissible set of the MPC controller is, by definition, $\mathbb{X}^\kappa = \mathbb{X}_F$.

If a projection algorithm is available then the feasible set can be computed. However, as discussed in Chapter 3, projection is not the most efficient or easiest way to proceed in calculating the feasible set. An alternative method is to compute the N-step *nominal* controllable set to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$. Depending on the problem and the algorithms used, the iterative approach might be more efficient.

Theorem 5.1. *The feasible set $\mathbb{X}_F(\mathbb{T}, N, P)$ of the MPC regulation problem is given by*

$$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))\,. \tag{5.5}$$

Proof. From the constraints (5.2) the solution to the MPC problem has to satisfy $\hat{x}_{l|k} \in \mathbb{X}$ and $\hat{u}_{l|k} = h(\hat{x}_{l|k}) \in \mathbb{U}, \forall l = N, \ldots, P-1$, therefore $\hat{x}_{l|k} \in \mathbb{X}^h, \forall l = N, \ldots, P-1$. It is also required that $\hat{x}_{P|k} \in \mathbb{T}$, therefore $\hat{x}_{N|k} \in \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$.

Furthermore, the constraints $\hat{x}_{l|k} \in \mathbb{X}$ and $\hat{u}_{l|k} \in \mathbb{U}$ have to be satisfied for all $l = 0, \ldots, N-1$. Because the problem does not include the effect of any disturbances, it follows that there exists a control sequence of length N such that these constraints can be satisfied if only if $x_k = \hat{x}_{0|k} \in \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))$. $\square$

Remark 5.3. As mentioned earlier, the constraint $\hat{x}_{0|k} \in \mathbb{X}$ can be removed. If this constraint has been removed, then the feasible set is equal to the one-step set to the $(N-1)$-step controllable set to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$, i.e. $\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{Q}\left(\mathcal{K}_{N-1}\left(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})\right)\right)$.

It is useful to note that the feasible set is equal to the one-step controllable set to the feasible set of the MPC problem with a control and prediction horizon of $N-1$ and $P-1$:

[1]Occasionally, the arguments $(\mathbb{T}, N, P)$ in $\mathbb{X}_F(\mathbb{T}, N, P)$ will be dropped for simplicity of notation.

Corollary 5.1.

$$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N-1, P-1)) \cap \mathbb{X} \tag{5.6a}$$

$$= \mathcal{K}_1(\mathbb{X}_F(\mathbb{T}, N-1, P-1))\,. \tag{5.6b}$$

Proof. This follows by observing that

$$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})) = \mathcal{Q}(\mathcal{K}_{N-1}(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))) \cap \mathbb{X}$$

and

$$\mathbb{X}_F(\mathbb{T}, N-1, P-1) = \mathcal{K}_{N-1}(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))\,.$$

□

Furthermore, if $x_k \in \mathbb{X}_F(\mathbb{T}, N, P)$, then after implementing the resulting control the state at the next time instant will be contained in the feasible set of the MPC problem with a control and prediction horizon of $N-1$ and $P-1$:

Lemma 5.1.

$$\hat{x}_{0|k} \in \mathbb{X}_F(\mathbb{T}, N, P) \Rightarrow \hat{x}_{1|k} \in \mathbb{X}_F(\mathbb{T}, N-1, P-1)\,. \tag{5.7}$$

Proof. Similar to the proof of Theorem 5.1, it can be shown that $\hat{x}_{1|k} \in \mathcal{K}_{N-1}(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})) = \mathbb{X}_F(\mathbb{T}, N-1, P-1)$. Alternatively, it could be argued that one can drive the system from $\hat{x}_{0|k}$ to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$ in N steps only if it is possible to drive the system from $\hat{x}_{1|k} = f(\hat{x}_{0|k}, \hat{u}_{0|k})$ to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$ in $N-1$ steps. □

This result provides one with a possible way of recovering from infeasibility without the need for soft constraints.

Corollary 5.2. *If there are no disturbances present and the MPC problem with horizons N and P is feasible at time k, but infeasible at time $k+1$, then the MPC problem with horizons $N-1$ and $P-1$ is feasible at time $k+1$.*

This process can be repeated until $N = 0$, if necessary, at which point the state will lie inside the set $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$ and one could switch to the control law $u_k = h(x_k)$, in a fashion similar to dual-mode MPC [MM93]. However, so far no assumptions about the invariance of $\mathbb{T}$ has been made and constraint satisfaction for all time cannot be guaranteed.

MPC as a Minimum-time Control Scheme

Another interesting interpretation of the above result is that if $N = P$, then by decreasing the horizon at each time step one can drive the system to $\mathbb{T}$ in N steps. This behaviour is similar to the "minimum-time" control algorithms described in [KG87, MS97].

However, in the latter the controllable sets to $\mathbb{T}$ are computed off-line. It is determined on-line for which pair of sets $x_k \in \mathcal{K}_N(\mathbb{X}, \mathbb{T}) \backslash \mathcal{K}_{N-1}(\mathbb{X}, \mathbb{T})$. A control is then computed such that $x_{k+1} \in \mathcal{K}_{N-1}(\mathbb{X}, \mathbb{T})$. This process is repeated for N steps, at which point $x_{k+N} \in \mathbb{T}$.

Provided $\mathbb{T}$ is control invariant, the same minimum-time behaviour and constraint satisfaction for all time can be achieved using MPC without having to compute the controllable sets off-line and having to search through all controllable sets in memory.

5.3 Causes of Infeasibility in MPC

An important fact to recognise is that infeasibility can occur even if there are no disturbances and no model mismatch. This problem of guaranteeing nominal feasibility is inherent in the MPC formulation.

As was mentioned earlier, due to the *finite-horizon* nature of MPC, the control at the next time instant could be different from the previously computed value. There are basically two ways in which the MPC problem could become infeasible:

- A bad choice of design variables (horizons and cost function) could result in a solution with $\hat{x}^*_{1|k} \in \mathbb{X} \backslash \mathbb{X}_F$. Since $x_{k+1} \notin \mathbb{X}_F$, the MPC problem will be infeasible at the next time instant;
- If $\mathbb{X}_F \backslash \mathcal{C}_\infty(\mathbb{X}) \neq \emptyset$ it is possible that $\hat{x}^*_{1|k} \in \mathbb{X}_F \backslash \mathcal{C}_\infty(\mathbb{X})$, which will result in $x_{k+1} \notin \mathcal{C}_\infty(\mathbb{X})$. Since there does not exist a control sequence which will satisfy the constraints if the state is outside the maximal control invariant set, the MPC problem will become infeasible at some future time, even though it will be feasible at time $k + 1$.

The use of soft constraints [SR99, Mac01] is one way of solving the infeasibility problem and will be discussed in Chapter 7. However, this is not the best approach to addressing nominal feasibility. State constraints will be violated at some future time, even in the absence of disturbances if the solution to the soft-constrained problem results in $\hat{x}^*_{1|k} \in \mathbb{X} \backslash \mathcal{C}_\infty(\mathbb{X})$. This chapter addresses the nominal feasibility issue by providing conditions on N, P and $\mathbb{T}$ under which feasibility (and hence state constraint satisfaction) can be guaranteed for all time, without the need for soft constraints.

Another important issue to consider is the fact that the solution to Problem 5.1 might be sub-optimal. Schemes which rely on the optimality of the solution to guarantee feasibility and stability lose their guarantee of feasibility if the solution is sub-optimal. For example, in [BTM00b] a method is described for analysing the feasibility and stability of a given MPC scheme. It is assumed that either the on-line computed control input is optimal or that the MPC control law has been computed off-line as described in Section 7.4. Sub-optimality of the control law might invalidate the analysis results, depending on the values used for $\mathbb{T}$, N and P.

Finally, the choice of $F(\cdot)$ and $L(\cdot,\cdot)$ also affects the feasibility of the resulting MPC controller. Though it does not affect the feasible set $\mathbb{X}_F$, it will affect whether the feasible set is a positively invariant set for the closed-loop system. Once again, a sub-optimal solution might invalidate feasibility results based on the cost function.

Therefore, one of the aims of the approach adopted in this chapter is to derive conditions based on the standard MPC framework of Problem 5.1 which *do not rely on the cost function or optimality of the solution*.

Before proceeding, some further definitions are needed to define precisely the aspects of feasibility that will be considered.

5.4 Fundamental Definitions and Results for Nominal Feasibility

By definition the MPC regulation problem is feasible at time k if and only if $x_k \in \mathbb{X}_F \neq \emptyset$. However, one is interested in guaranteeing that once feasible, the MPC problem will always be feasible:

Definition 5.1 (Feasible for all time). The MPC problem is *feasible for all time $k \in \mathbb{N}$* if and only if the initial state x_0 belongs to the feasible set and all future evolutions of the state of the closed-loop system belong to the feasible set, i.e. $x_{k+1} = f(x_k, \kappa(x_k)) \in \mathbb{X}_F, \forall k \in \mathbb{N}$

With this definition, the first result follows from the discussion in Section 5.3 and is a necessary and sufficient condition for guaranteeing that the MPC problem is feasible for all time:

Lemma 5.2. *The MPC problem is feasible for all time if and only if $x_0 \in \mathbb{X}_F \cap \mathcal{C}_\infty(\mathbb{X})$ and the solution to the MPC problem results in $\hat{x}^*_{1|k} \in \mathbb{X}_F \cap \mathcal{C}_\infty(\mathbb{X})$ for all $k \geq 0$.*

Definition 5.2 (Feasible control input). Given a state x_k, a control input u_k is *feasible* if and only if the state-input pair (x_k, u_k) is compatible with the constraints of the MPC problem, i.e. u_k is feasible if and only if there exists a control sequence $\pi_k^N = [\hat{u}'_{0|k}, \hat{u}'_{1|k}, \dots, \hat{u}'_{N-1|k}]'$ with $\hat{u}_{0|k} = u_k$ such that (x_k, π_k^N) satisfies (5.2).

In other words, a control input is feasible if and only if it is the first element of a feasible solution to the MPC problem. The feasible set is therefore the set of states for which a feasible control input (and sequence) exists. Note that if a control input is *admissible*, it is not necessarily *feasible*. For a given state the set of *feasible* inputs is a subset of the *admissible* inputs.

As discussed, the MPC problem might become infeasible at some point in time for a *subset* of initial states contained in $\mathbb{X}_F$. It is desirable to design the controller such that *for all initial states* contained in $\mathbb{X}_F$, the MPC problem will be feasible for all time. An infeasible MPC problem can then be treated as a process exception; the constraints should be softened in order to compute a control action and the operator alerted that constraint violation is probable.

Definition 5.3 (Strongly feasible). The MPC problem is *strongly feasible* if and only if *for all* $x_0 \in \mathbb{X}_F$ and *for all* sequences of feasible control inputs the MPC problem is feasible for all time. Equivalently, the MPC problem is strongly feasible if and only if for all $x_k \in \mathbb{X}_F$ and feasible control inputs, $x_{k+1} \in \mathbb{X}_F$.

If the feasible set is strongly feasible, then one can guarantee that the MPC problem will never become infeasible if there are no disturbances or model uncertainty. It is this notion of *strong* feasibility, which is independent of optimality or the cost function, which will be used throughout to investigate feasibility in MPC. This strong feasibility result is also guaranteed in the traditional MPC approaches when using a control invariant terminal set [MRRS00].

Though this definition might result in conservative guarantees for feasibility, it does provide a good basis from which to proceed. By introducing additional assumptions, such as the optimality of the solution or a guarantee that the cost function will decrease at each time step, one might be able to obtain better results.

Set invariance theory immediately provides one with the following condition for guaranteeing that the feasible set will be strongly feasible.

Proposition 5.1. *The MPC problem is strongly feasible only if the feasible set* $\mathbb{X}_F$ *is a* control invariant *set for the system* $x_{k+1} = f(x_k, u_k)$.

It is important to note that control invariance is only a necessary condition for a strongly feasible MPC problem. The design variables which determine whether $\mathbb{X}_F$ is control invariant are N, P, $h(x_k)$ and $\mathbb{T}$. All the design variables, including the cost functions $F(x_k)$ and $L(x_k, u_k)$, and the optimality of the solution determine whether $\mathbb{X}_F$ is positively invariant for the closed-loop system. As discussed in Section 5.3, the aim of this chapter is to determine feasibility conditions independent of the choice of cost function or optimality of the solution.

The set $\mathbb{X}_F$ is a control invariant set only if $\mathbb{X}_F$ is a subset of the maximal control invariant set $\mathcal{C}_\infty(\mathbb{X})$. This means that the feasible set cannot be larger than the maximal control invariant set if the MPC problem is to be strongly feasible. A design goal would therefore be to obtain an MPC control problem with a feasible set as close as possible in size to the maximal control invariant set. The concept of finite-determinedness of controllable sets is useful in obtaining results relating to the size of the feasible set and will be discussed in the following sections.

One might also be interested in determining whether increasing the control and prediction horizons or choosing a new terminal set will significantly increase the size of the feasible set. This can be determined by calculating what fraction of the volume of the maximal control invariant set the new feasible set is, in relation to the old feasible set. Comparing volumes might be misleading and an alternative is to use an approximation test for set equality as in Section 3.3.2. Relevant metrics still need to be developed in order to determine the change in size of the feasible set.

5.5 The Need for a Terminal Constraint Set

The idea of using a terminal constraint to guarantee nominal stability (and feasibility) was introduced in [KG88], where the terminal constraint was chosen to be the origin $\mathbb{T} = \{0_n\}$. However, this constraint reduces the size of the feasible set and could result in numerical convergence problems in the optimisation, especially when working with nonlinear models [May00].

One of the most popular methods for guaranteeing that the MPC problem is strongly feasible, is to choose a control invariant terminal *set* [MM93]. By choosing the terminal constraint to be a set, rather than the origin, the size of the feasible set is increased and most of the numerical convergence problems are addressed.

Though the terminal constraint idea seems to have been embraced by the academic community, it still needs to find its way into industry. This is due to a number of factors:

- The addition of a control invariant set could result in a smaller feasible set for the same control horizon, as stated in Proposition 5.2. However, it might be possible to increase the size of the feasible set with only a small increase in the control or prediction horizon. By increasing the prediction horizon one could get an increase in the size of the feasible set without a large increase in computational overhead, as discussed in [DMMS00, ZA98];

- The computation of a sufficiently large control invariant terminal set is believed to be computationally expensive. However, this computation is done off-line and computation speed is therefore less important;

- The addition of a terminal set increases the overhead in the optimisation. With the availability of efficient interior-point methods [RWR98, Mac01] with a time complexity independent on the number of constraints, this will probably be less of an issue in the future;

- The invariance condition is only sufficient and it would be nice to see under what circumstances it becomes a necessary condition or whether a better solution to the feasibility problem exists;

- By modifying the original MPC formulation and adding more mathematics such as set invariance theory, some transparency is lost. The results and tools from set invariance theory therefore need to presented in the most simplistic form possible, while still capturing the essential concepts. For safety-critical applications guarantees of controller performance is required and set invariance might be able to provide such guarantees.

For the reasons mentioned above, this chapter investigates to what extent the invariance condition on $\mathbb{T}$ is necessary in guaranteeing feasibility.

5.6 A Generalised Sufficient Condition for Strong Feasibility

The following sufficient condition can be thought of as a generalisation of the "control invariant terminal set" condition of [MM93]. The proof differs from the traditional "shifted control" approach generally adopted for proving feasibility for MPC schemes with a control invariant terminal set. The main idea here is to show that if x_k is in the feasible set, then $x_{k+1} = f(x_k, \kappa(x_k))$ is also in the feasible set.

Lemma 5.3. *If $\mathbb{X}_F(\mathbb{T}, N, P)$ is control invariant, then the MPC problem with a control horizon of $\tilde{N} = N + 1$ and a prediction horizon of $\tilde{P} = P + 1$ is strongly feasible.*

Proof. $\mathbb{X}_F(\mathbb{T}, N, P)$ is control invariant if and only if

$$\mathbb{X}_F(\mathbb{T}, N, P) \subseteq \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N, P)) .$$

Recall that $\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))$ and that

$$\mathbb{X}_F(\mathbb{T}, N + 1, P + 1) = \mathcal{K}_{N+1}(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})) = \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X} .$$

If $x_k \in \mathbb{X}_F(\mathbb{T}, N + 1, P + 1)$, then after implementing any feasible control input,

$$x_{k+1} \in \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})) = \mathbb{X}_F(\mathbb{T}, N, P) \subseteq \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N, P)) .$$

Since x_{k+1} must also be contained in $\mathbb{X}$, $x_{k+1} \in \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}$. However, this implies that $x_{k+1} \in \mathbb{X}_F(\mathbb{T}, N+1, P+1)$, since Corollary 5.1 states that $\mathbb{X}_F(\mathbb{T}, N+1, P+1) = \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}$.

The MPC problem is therefore feasible at the next time instant. By induction, the MPC problem is feasible for all time. Since this holds for any arbitrary element $x_k \in \mathbb{X}_F(\mathbb{T}, N + 1, P + 1)$ and any feasible control input, the MPC problem is strongly feasible. □

Remark 5.4. This result holds even if $\mathbb{X}_F(\mathbb{T}, N - 1, P - 1)$ and/or $\mathbb{T}$ are not control invariant.

This result is useful from both a theoretical and practical viewpoint, since one can choose any $\mathbb{T}$ and increase P and N to see whether the feasible set becomes control invariant for some values. If the feasible set is control invariant then by increasing the complexity of the optimisation by a small amount (i.e. increasing the control and prediction horizons by one), one can guarantee that the MPC problem is strongly feasible.

The next result follows immediately.

Theorem 5.2. *If $\mathbb{X}_F(\mathbb{T}, N, P)$ is control invariant, then the MPC problem with a control horizon of $\tilde{N} \geq N + 1$ and a prediction horizon of $\tilde{P} = P + \tilde{N} - N$ is strongly feasible.*

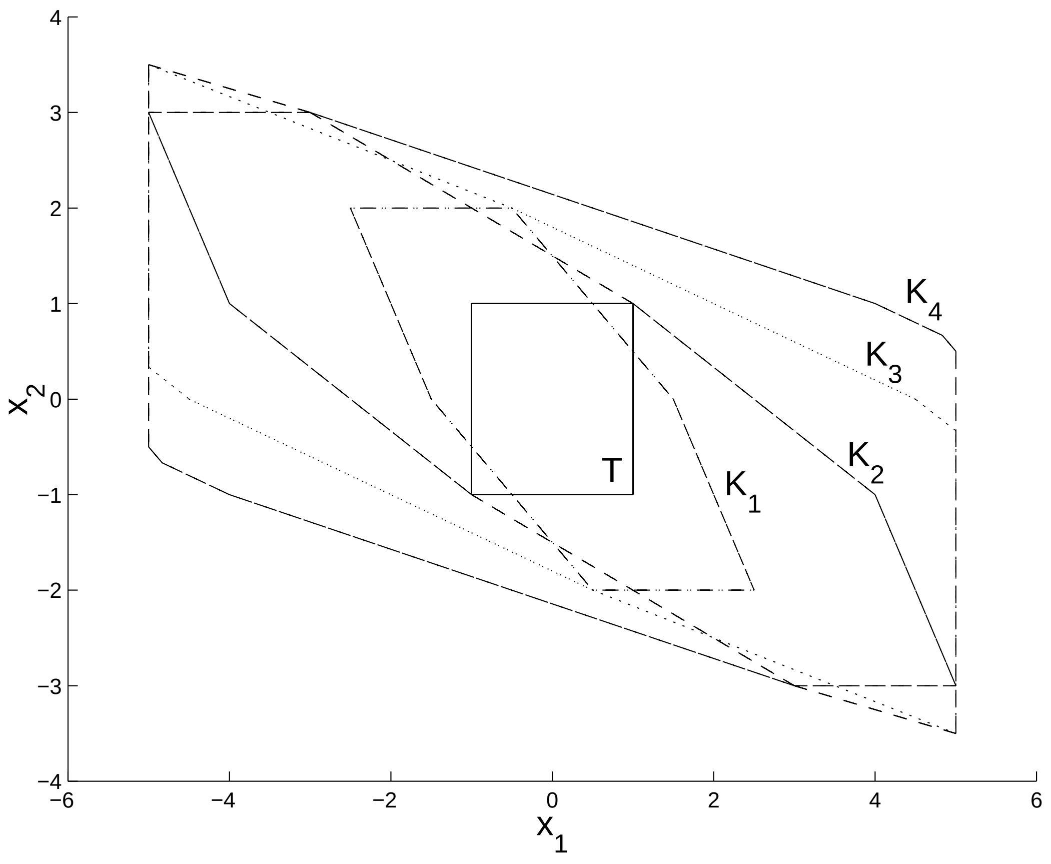

Figure 5.1: Plot showing that even if the terminal set is not control invariant, the MPC problem is strongly feasible if $N = P \geq 4$

Proof. From Lemma 5.3, if $\mathbb{X}_F(\mathbb{T}, N, P)$ is control invariant, then $\mathbb{X}_F(\mathbb{T}, N + 1, P + 1)$ is strongly feasible and hence also control invariant. The result follows by induction. $\square$

This result will be used throughout the chapter and implies that increasing the control and prediction horizons by the same amount will result in a strongly feasible MPC problem.

Remark 5.5. A necessary and sufficient condition for the MPC problem to be strongly feasible, is given later by Corollary 6.1.

Example 5.1. *Consider the system*

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k\,, \tag{5.8}$$

with the input constrained to $\|u\|_\infty \leq 1$ and the states constrained to $\|x\|_\infty \leq 5$. The target set $\mathbb{T} = \{x \in \mathbb{R}^2 \mid \|x\|_\infty \leq 1\}$ is not control invariant and the control and prediction horizons are equal $P = N$. Figure 5.1 is a plot of $\mathbb{T}$ and the controllable sets $\mathcal{K}_N(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, N, N)$ for $N = 1, \ldots, 4$.

Recalling that a set Ω is control invariant if and only if $\Omega \subseteq \mathcal{Q}(\Omega)$ and that the inequality $\mathcal{K}_N(\mathbb{X}, \mathbb{T}) \subseteq \mathcal{Q}(\mathcal{K}_{N-1}(\mathbb{X}, \mathbb{T}))$ holds, one can determine graphically that $\mathcal{K}_1(\mathbb{X}, \mathbb{T})$ and $\mathcal{K}_2(\mathbb{X}, \mathbb{T})$ are not control invariant, but that $\mathcal{K}_3(\mathbb{X}, \mathbb{T})$ is. Theorem 5.2 implies that an MPC problem with the given $\mathbb{T}$ and horizons $N = P \geq 4$ will be strongly feasible, even though $\mathbb{T}$ is not control invariant.

5.7 Equal Control and Prediction Horizons

The terminal controller $h(x_k)$ does not affect the feasible set if the control and prediction horizons are equal. The only design variables that determine the geometrical properties of the feasible set are the control horizon $N = P$ and the terminal constraint set $\mathbb{T}$.

5.7.1 Terminal Set $\mathbb{T} = \mathbb{X}$

The following new result on the feasibility of the MPC problem considers the case when the terminal constraint set is equal to the state constraints. This theorem tells one what happens with the feasibility of the MPC problem if the terminal constraint set is effectively "removed"[2].

Theorem 5.3. *Let $P = N$ and $\mathbb{T} = \mathbb{X}$:*

1. *The feasible set is equal to the N-step admissible set:*

$$\mathbb{X}_F(\mathbb{X}, N, N) = \mathcal{C}_N(\mathbb{X}) .$$

   The feasible set contains the maximal control invariant set:

$$\mathcal{C}_\infty(\mathbb{X}) \subseteq \mathbb{X}_F(\mathbb{X}, N, N) .$$

   The feasible set is control invariant if and only if the maximal control invariant set is finitely determined and the control horizon is equal to or greater than its determinedness index i^*, i.e.

$$\mathbb{X}_F(\mathbb{X}, N, N) \subseteq \mathcal{Q}(\mathbb{X}_F(\mathbb{X}, N, N)) \Leftrightarrow \mathcal{C}_\infty(\mathbb{X}) = \mathcal{C}_{i^*}(\mathbb{X}), N \geq i^* ;$$

2. *The MPC problem is strongly feasible if the control horizon is larger than the determinedness index i^* of the maximal control invariant set $\mathcal{C}_\infty(\mathbb{X})$, i.e. $N \geq i^* + 1$;*

3. *A* larger *control horizon results in a* smaller *feasible set. The size of the feasible set stops decreasing if and only if the maximal control invariant set is finitely determined and the control horizon is larger than its determinedness index, i.e.*

$$i^* \geq N_1 > N_2 \Leftrightarrow \mathbb{X}_F(\mathbb{X}, N_1, N_1) \subset \mathbb{X}_F(\mathbb{X}, N_2, N_2) .$$

   Furthermore,

$$\mathbb{X}_F(\mathbb{X}, N, N) = \mathcal{C}_\infty(\mathbb{X}) = \mathcal{C}_{i^*}(\mathbb{X}), \forall N \geq i^* .$$

[2] In the sense of replacing $\mathbb{T}$ with the original state constraints $\mathbb{X}$.

Proof.

1. From Theorem 5.1 and the definitions of controllable and admissible sets, the feasible set is given by

   $$\mathbb{X}_F(\mathbb{X}, N, N) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}_0^h(\mathbb{X}, \mathbb{X})) = \mathcal{K}_N(\mathbb{X}, \mathbb{X}) = \mathcal{C}_N(\mathbb{X})\,.$$

   By construction $\mathcal{C}_\infty(\mathbb{X}) \subseteq \mathcal{C}_N(\mathbb{X})$, hence $\mathcal{C}_\infty(\mathbb{X}) \subseteq \mathbb{X}_F(\mathbb{X}, N, N)$.

   Since $\mathcal{C}_i(\mathbb{X})$ contains the maximal control invariant set and $\mathcal{C}_i(\mathbb{X})$ is control invariant only if $\mathcal{C}_i(\mathbb{X}) \subseteq \mathcal{C}_\infty(\mathbb{X})$, $\mathcal{C}_i(\mathbb{X})$ is control invariant if and only if $\mathcal{C}_i(\mathbb{X}) = \mathcal{C}_\infty(\mathbb{X})$. However, this is only possible if $\mathcal{C}_\infty(\mathbb{X})$ is finitely determined. $\mathcal{C}_\infty(\mathbb{X})$ is finitely determined if and only if there exists an i such that $\mathcal{C}_i(\mathbb{X}) = \mathcal{C}_{i+1}(\mathbb{X})$. As a consequence, $\mathbb{X}_F(\mathbb{X}, N, N) = \mathcal{C}_N(\mathbb{X})$ is control invariant if and only if $\mathcal{C}_\infty(\mathbb{X})$ is finitely determined and $N \geq i^*$.

2. The first statement says that an MPC problem with $N = P$ and $\mathbb{T} = \mathbb{X}$ is control invariant if and only if $N \geq i^*$. Theorem 5.2 then implies that an MPC problem with control and prediction horizon $P = N \geq i^* + 1$ is strongly feasible.

3. This follows from the fact that $\mathcal{C}_{N+1}(\mathbb{X}) \subseteq \mathcal{C}_N(\mathbb{X})$. The strict inclusion $\mathcal{C}_{N+1}(\mathbb{X}) \subset \mathcal{C}_N(\mathbb{X})$ holds if and only if $N < i^*$, since Theorem 2.3 implies that $\mathcal{C}_{N+1}(\mathbb{X}) = \mathcal{C}_N(\mathbb{X})$ if and only if the maximal control invariant set is finitely determined and $N \geq i^*$.

□

Theorem 5.3 implies that one cannot choose the design variables such that the MPC problem is strongly feasible if and only if the maximal control invariant set is not finitely determined. In general one cannot guarantee finite determinedness or that the determinedness index will be small enough for the controller to be implementable. As such, one cannot choose values for the control horizon which would make the MPC problem strongly feasible. It might be possible that a redesign of the state and/or control constraints or the system might solve the determinedness problem, but it is in general not clear how to proceed if this is the case.

This result also implies that if one were wanting to do without the terminal constraint and keep the control and prediction horizons equal, then a strongly feasible MPC problem will result if the maximal control invariant set is finitely determined and the control horizon is larger than the determinedness index. It could therefore be argued that a terminal constraint set is necessary if the required control horizon is too large for the available computation power. By adding a terminal set and choosing a smaller control horizon, it might be possible to get a strongly feasible MPC controller with a sufficiently large feasible set.

Assuming the Solution is Optimal

A subset of the feasible set might still be positively invariant for the closed-loop system and this region might be large enough for all practical purposes. However, calculating this region is difficult, even if the internal model is LTI. It is shown in [BMDP00a] that for MPC problems with LTI models and polyhedral constraints, the closed-loop system is a piecewise-affine (PWA) function. A method for computing a region of attraction of the origin for PWA systems is described in [BTM00a]. In [BTM00b] this procedure is used to calculate a positively invariant subset of the closed-loop system, where it is assumed that the optimal solution will be obtained at each time step.

Another approach which is based on finding *a priori* a lower bound for the control horizon which guarantees that the finite and infinite horizon costs are equal, given a set of initial states, is described in [CM96]. With the appropriate assumptions on the system and the cost function, if the finite and infinite horizon costs are equal, then the origin of the closed-loop system is an asymptotically stable fixed point (and feasibility for all time is guaranteed). A similar idea is described in [PN00a, PN00b], but allowing for a difference between the finite and infinite horizon costs. Though an explicit terminal constraint is not present in all of these formulations, the results rely on guaranteeing that the terminal state lies in some control invariant set.

5.7.2 Control Invariant Terminal Set

The following theorem contains the well-known control invariant terminal constraint condition [MM93, MRRS00].

Theorem 5.4. *Let $P = N$ and the terminal constraint set be a* control invariant *subset of $\mathbb{X}$, i.e. $\mathbb{T} \subseteq \mathcal{Q}(\mathbb{T}) \cap \mathbb{X}$:*

1. *The feasible set is equal to the N-step stabilisable set:*

$$\mathbb{X}_F(\mathbb{T}, N, N) = \mathcal{S}_N(\mathbb{X}, \mathbb{T}) .$$

   The feasible set is control invariant and contained within the maximal control invariant set:

$$\mathbb{X}_F(\mathbb{T}, N, N) \subseteq \mathcal{C}_\infty(\mathbb{X}) ;$$

2. *The MPC problem is strongly feasible;*

3. *A* larger *control horizon results in a* larger *feasible set. The size of the feasible set stops increasing if and only if the maximal stabilisable set is finitely determined and the control horizon is larger than its determinedness index i^*, i.e.*

$$i^* \geq N_1 > N_2 \Leftrightarrow \mathbb{X}_F(\mathbb{T}, N_1, N_1) \supset \mathbb{X}_F(\mathbb{T}, N_2, N_2) .$$

   Furthermore,

$$\mathbb{X}_F(\mathbb{T}, N, N) = \mathcal{S}_\infty(\mathbb{X}, \mathbb{T}) = \mathcal{S}_{i^*}(\mathbb{X}, \mathbb{T}), \forall N \geq i^* .$$

Proof.

1. From Theorem 5.1 and the definitions of controllable and stabilisable sets, the feasible set is given by

   $$\mathbb{X}_F(\mathbb{T}, N, N) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}_0^h(\mathbb{X}, \mathbb{T})) = \mathcal{K}_N(\mathbb{X}, \mathbb{T}) = \mathcal{S}_N(\mathbb{X}, \mathbb{T})\,.$$

   Since $\mathbb{T}$ is control invariant, it follows from the first property in Proposition 2.7 that $\mathcal{S}_N(\mathbb{X}, \mathbb{T})$ is control invariant. The set $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, N, N)$ is control invariant only if $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) \subseteq \mathcal{C}_\infty(\mathbb{X})$.

2. Since $\mathbb{X}_F(\mathbb{T}, N, N)$ is control invariant for all $N \geq 0$, it follows from Theorem 5.2 that the MPC problem with $N = P \geq 1$ is strongly feasible.

3. This follows from the second property in Proposition 2.7. The strict set inclusion $\mathcal{S}_{N+1}(\mathbb{X}, \mathbb{T}) \supset \mathcal{S}_N(\mathbb{X}, \mathbb{T})$ holds if only if $N < i^*$, since Theorem 2.2 implies that $\mathcal{S}_{N+1}(\mathbb{X}, \mathbb{T}) = \mathcal{S}_N(\mathbb{X}, \mathbb{T})$ if and only if $\mathcal{S}_\infty(\mathbb{X}, \mathbb{T})$ is finitely determined and $N \geq i^*$.

□

In addition to the above result, the following result implies that by changing the terminal constraint set from $\mathbb{T} = \mathbb{X}$ to $\mathbb{T} \subset \mathbb{X}$, given the same control horizon, the feasible set will be contained within the original feasible set:

Proposition 5.2. *Let $\mathbb{T}$ be a control invariant set. If $N = P$, then the feasible set of an MPC problem with $\mathbb{T} \subset \mathbb{X}$ is contained within the feasible set of an MPC problem with $\mathbb{T} = \mathbb{X}$, i.e.*

$$\mathbb{X}_F(\mathbb{T}, N, N) \subseteq \mathbb{X}_F(\mathbb{X}, N, N)\,.$$

Furthermore, if $\mathcal{S}_\infty(\mathbb{X}, \mathbb{T})$ is not finitely determined or $\mathcal{S}_\infty(\mathbb{X}, \mathbb{T})$ is finitely determined with determinedness index i^* and $N < i^*$, then

$$\mathbb{X}_F(\mathbb{T}, N, N) \subset \mathbb{X}_F(\mathbb{X}, N, N)\,.$$

Proof. Recall $\mathcal{C}_N(\mathbb{X}) = \mathbb{X}_F(\mathbb{X}, N, N)$ and if $\mathbb{T}$ is control invariant, then $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, N, N)$.

Since $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) \subseteq \mathcal{C}_\infty(\mathbb{X})$ and $\mathcal{C}_\infty(\mathbb{X}) \subseteq \mathcal{C}_N(\mathbb{X})$ it follows that $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) \subseteq \mathcal{C}_N(\mathbb{X})$. This gives the first inclusion.

$N < i^*$ if and only if $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) \subset \mathcal{S}_{N+1}(\mathbb{X}, \mathbb{T})$. Combining this with the fact that $\mathcal{S}_{N+1}(\mathbb{X}, \mathbb{T}) \subseteq \mathcal{S}_\infty(\mathbb{X}, \mathbb{T}) \subseteq \mathcal{C}_\infty(\mathbb{X}) \subseteq \mathcal{C}_N(\mathbb{X})$ gives $\mathcal{S}_N(\mathbb{X}, \mathbb{T}) \subset \mathcal{C}_N(\mathbb{X})$. This gives the second inclusion. □

Theorem 5.4 and Proposition 5.2 imply that if the maximal stabilisable set is finitely determined, then one could determine the size of the control horizon which will maximise the feasible set. It also tells one that an increase in control horizon will not increase the size of the feasible set. In some cases it happens that $\mathcal{C}_\infty(\mathbb{X}) = \mathcal{S}_\infty(\mathbb{X}, \mathbb{T})$ and that both sets are finitely determined. One can then choose the control horizon which minimises the computational overhead. For example. if i^* and j^* are the determinedness indices of $\mathcal{C}_\infty(\mathbb{X})$ and $\mathcal{S}_\infty(\mathbb{X}, \mathbb{T})$, respectively, then one could choose the $\mathbb{T}$ and N such that $N = \min(i^*, j^*)$.

Increasing the Horizon Length Until $\hat{x}_{N|k} \in \mathbb{T}$

A remark could be made with regards to the discussion in [SR98, Sect. 4A] and [RR99]. The authors argue that a terminal set should not be included in the MPC problem, as it increases computational overhead. They propose that after each optimisation it should be checked whether the terminal state $\hat{x}_{N|k}$ lies in a control invariant terminal set and if not, increase the control horizon by some heuristic and repeat the optimisation until the terminal state lies in a control invariant set. This requirement is mainly due to the fact that the authors require the finite and infinite horizon costs to be equal.

This approach suffers from two main drawbacks. If $x_k \notin \mathcal{C}_\infty(\mathbb{X})$ then the terminal state will never lie in a control invariant set for *any* control horizon and the process of increasing the control horizon will only result in an infeasible problem at some future time. Secondly, the problem is restricted to slower processes, since the control horizon has to be increased repeatedly before applying the control input. In [SR98, Sect. 4B] the authors propose that one switch to an MPC controller which is known to be stabilising, such as one with a terminal constraint, if a control horizon is not obtained which guarantees optimality. This adds unnecessary overhead to the MPC problem.

Furthermore, if the sampling time of the process has already been fixed, then this puts a restriction on the size of the optimisation problem which can be solved between samples. This immediately places an upper bound on the control horizon. By fixing the control horizon to this value and including a control invariant terminal constraint in the optimisation one not only maximises the feasible set, given the available computation power, but can provide a guarantee that the finite horizon cost will be equal to the infinite horizon cost for a subset of the feasible set. This subset of optimality cannot be increased, given the computational power, without changing the cost function and/or computing a larger control invariant set.

The author of this thesis proposes that the computational overhead of adding constraints in the form of a control invariant terminal set is minor compared to the overhead of having to repeatedly increase the control horizon. Recently, in [RWR98] it was shown that if the system is LTI, then interior-point methods can be used to solve for the MPC control action with a time complexity of $O(N(m+n)^3)$. In other words, the time complexity is independent of the number of constraints and linear in the control horizon length. With this new algorithm, the addition of a control invariant terminal set will result in a minor increase in the time taken to compute the control action.

5.8 Different Control and Prediction Horizons

When the prediction horizon is larger than the control horizon, feasibility analysis of the MPC problem is slightly more involved and it is more difficult to obtain many useful results regarding the feasibility of the MPC problem.

Proposition 5.3. *Assume that the maximal control invariant set is finitely determined with determinedness index i^* and that $P > N$. If $\mathbb{T}$ is any subset of $\mathbb{X}$ and $P \geq i^*$, then $\mathbb{X}_F(\mathbb{T}, N, P) \subseteq \mathcal{C}_\infty(\mathbb{X})$.*

Proof. For all $x_k \in \mathbb{X}_F(\mathbb{T}, N, P)$ there exists a control sequence of length P such that the state constraints are satisfied. By recalling the definition of admissible sets it follows that $\mathbb{X}_F(\mathbb{T}, N, P) \subseteq \mathcal{C}_P(\mathbb{X})$. But $P \geq i^*$, therefore $\mathcal{C}_P(\mathbb{X}) = \mathcal{C}_\infty(\mathbb{X})$. □

Note that since Proposition 5.3 does not assume any invariance condition on $\mathbb{T}$, the result does not imply that the feasible set is control invariant. Even with a control invariant $\mathbb{T}$, if $P > N$ one cannot guarantee in general that the MPC problem is strongly feasible or even control invariant without making additional assumptions.

Before proceeding with considering some special cases, the following lemma is useful in understanding Theorem 5.5.

Lemma 5.4. *Let $\mathbb{T} \subseteq \mathbb{X}$.*

1. *If $\mathbb{T} = \mathbb{X}^h$, then $\mathcal{KO}_i^h(\mathbb{X}, \mathbb{T}) = \mathcal{O}_i^h(\mathbb{X})$ for all $i \geq 0$. Furthermore, if the maximal positively invariant set $\mathcal{O}_\infty^h(\mathbb{X})$ is finitely determined with determinedness index i^*, then $\mathcal{KO}_i^h(\mathbb{X}, \mathbb{T}) = \mathcal{O}_\infty^h(\mathbb{X})$ for all $i \geq i^*$.*
2. *If $\mathbb{T} \supset \mathbb{X}^h$ and the maximal positively invariant set $\mathcal{O}_\infty^h(\mathbb{X})$ is finitely determined with determinedness index i^*, then $\mathcal{KO}_i^h(\mathbb{X}, \mathbb{T}) = \mathcal{O}_\infty^h(\mathbb{X})$ for all $i \geq i^* + 1$.*

Proof.

1. Recalling the definitions in Section 2.9, it follows that for all $i \geq 0$:
$$\mathcal{KO}_i^h(\mathbb{X}, \mathbb{T}) = \mathcal{KO}_i^h(\mathbb{X}, \mathbb{X}^h) = \mathcal{KO}_i(\mathbb{X}^h, \mathbb{X}^h) = \mathcal{O}_i(\mathbb{X}^h) = \mathcal{O}_i^h(\mathbb{X})\,.$$
2. Firstly, it will be shown that $\mathcal{O}_\infty^h(\mathbb{X}) \subseteq \mathcal{KO}_i^h(\mathbb{X}, \mathbb{T})$. If $x_k \in \mathcal{O}_\infty^h(\mathbb{X})$ then after applying $u_k = h(x_k)$ for $i \geq i^* + 1$ steps, $x_{k+i} \in \mathcal{O}_\infty^h(\mathbb{X})$. But $\mathcal{O}_\infty^h(\mathbb{X}) \subseteq \mathbb{X}^h \subset \mathbb{T}$, therefore $x_k \in \mathcal{KO}_i^h(\mathbb{X}, \mathbb{T})$.

   Secondly, it will be shown by contradiction that $\mathcal{O}_\infty^h(\mathbb{X}) \supseteq \mathcal{KO}_i^h(\mathbb{X}, \mathbb{T})$. Assume that $\mathcal{O}_\infty^h(\mathbb{X}) \not\supseteq \mathcal{KO}_i^h(\mathbb{X}, \mathbb{T})$. This implies that there exists an $x_k \in \mathcal{KO}_i^h(\mathbb{X}, \mathbb{T})$ for which the evolution of the system leaves $\mathbb{X}^h$ in i^* steps or less. However, $i \geq i^* + 1$, which implies that $\forall x_k \in \mathcal{KO}_i^h(\mathbb{X}, \mathbb{T})$ the system evolution will remain within $\mathbb{X}^h$ for the first i^* steps.

□

5.8.1 Terminal Set $\mathbb{T} = \mathbb{X}$

The next result considers the case when the control and prediction horizons are different and the terminal constraint set is equal to $\mathbb{X}$.

Theorem 5.5. *Let $P > N$ and $\mathbb{T} = \mathbb{X}$:*

1. *The feasible set is equal to the N-step* controllable *set to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{X})$ for the closed-loop system $x_{k+1} = f(x_k, h(x_k))$, i.e.*

   $$\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{X}))\,.$$

   The feasible set is not necessarily control invariant;

2. *The MPC problem is strongly feasible if the* difference *between the prediction and control horizons is larger than the determinedness index i^* of the maximal positively invariant set $\mathcal{O}^h_\infty(\mathbb{X})$, i.e. $P - N \geq i^* + 1$. The condition relaxes to $P - N \geq i^*$ if $\mathbb{T} = \mathbb{X}^h$;*

3. *Assume that N is fixed. By* increasing *the prediction horizon, the size of the feasible set* does not increase. *If the maximal positively invariant set $\mathcal{O}^h_\infty(\mathbb{X})$ is finitely determined and the* difference *between the prediction and control horizons is larger than its determinedness index i^*, then the feasible set is equal to the N-step stabilisable set to $\mathcal{O}^h_\infty(\mathbb{X})$, i.e.*

   $$N + i^* + 1 \geq P_1 > P_2 > N \Rightarrow \mathbb{X}_F(\mathbb{X}, N, P_1) \subseteq \mathbb{X}_F(\mathbb{X}, N, P_2)$$

   and

   $$\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{S}_N(\mathbb{X}, \mathcal{O}^h_\infty(\mathbb{X})), \forall P \geq N + i^* + 1\,;$$

4. *Assume that $P \geq N + i^* + 1$. A larger* control *horizon results in a* larger *feasible set. The size of the feasible set stops increasing if and only if the maximal stabilisable set $\mathcal{S}_\infty(\mathbb{X}, \mathcal{O}^h_\infty(\mathbb{X}))$ is finitely determined and the control horizon is larger than its determinedness index j^*, i.e.*

   $$j^* \geq N_1 > N_2 \Leftrightarrow \mathbb{X}_F(\mathbb{X}, N_1, P) \supset \mathbb{X}_F(\mathbb{X}, N_2, P)\,.$$

   Furthermore,

   $$\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{S}_\infty(\mathbb{X}, \mathcal{O}^h_\infty(\mathbb{X})), \forall N \geq j^*, P \geq N + i^* + 1\,.$$

Proof.

1. From Theorem 5.1 and the definitions of controllable sets, the feasible set is given by

   $$\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{X}))\,.$$

   Since no further assumptions have been made, one cannot deduce anything about the invariance of the feasible set.

2. If $P - N \geq i^* + 1$, then by the second statement in Lemma 5.4 it follows that $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{X}) = \mathcal{O}^h_\infty(\mathbb{X})$. This implies that $\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{O}^h_\infty(\mathbb{X}))$. But $\mathcal{O}^h_\infty(\mathbb{X})$ is control invariant for the system $x_{k+1} = f(x_k, u_k)$, therefore $\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{S}_N(\mathbb{X}, \mathcal{O}^h_\infty(\mathbb{X}))$.

   The feasible set can be seen to be equal to that of an MPC problem with $N = P$ and a control invariant $\mathbb{T} = \mathcal{O}^h_\infty(\mathbb{X})$. Strong feasibility follows from Theorem 5.4. The relaxation follows from the first statement in Lemma 5.4.

3. If $P_1 > P_2$ then $\mathcal{KO}^h_{P_1-N}(\mathbb{X}, \mathbb{X}) \subseteq \mathcal{KO}^h_{P_2-N}(\mathbb{X}, \mathbb{X})$ follows from the definition of the sets. As a result of applying Proposition 2.1 repetitively, it follows that $\mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P_1-N}(\mathbb{X}, \mathbb{X})) \subseteq \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P_2-N}(\mathbb{X}, \mathbb{X}))$ for all N.

   This implies that $\mathbb{X}_F(\mathbb{X}, N, P_1) \subseteq \mathbb{X}_F(\mathbb{X}, N, P_2)$. The feasible set for the case when $P \geq N + i^* + 1$ was derived in the previous statement.

4. The proof is the same as for the third statement in Theorem 5.4, since $\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{S}_N(\mathbb{X}, \mathbb{T})$ with $\mathbb{T} = \mathcal{O}^h_\infty(\mathbb{X})$.

□

Remark 5.6. The third statement in Theorem 5.5 says that an increase in the prediction horizon will *not* result in an increase in the size of the feasible set. Depending on the size of N, it is more likely that an increase in the prediction horizon leads to a decrease in the size of the feasible set. The decrease in the size of the feasible set then stops if and only if $P - N$ is larger than the determinedness index of $\mathcal{O}^h_\infty(\mathbb{X})$.

Note that if $P - N \leq i^*$, then one cannot guarantee that the feasible set is control invariant, except that there exists a subset of the feasible set which is control invariant. It is also difficult to say anything useful about the size of the feasible set with respect to the length of the horizons.

Theorem 5.5 leads to the following well-known result which is useful when the determinedness index of $\mathcal{O}^h_\infty(\mathbb{X})$ is known.

Corollary 5.3. *If $\mathcal{O}^h_\infty(\mathbb{X})$ is finitely determined with determinedness index i^*, then the feasible set of an MPC problem with terminal constraint $\mathbb{T} = \mathcal{O}^h_\infty(\mathbb{X})$ and $N = P$ is equal to the feasible set of an MPC problem with $\mathbb{T} = \mathbb{X}$ and $P \geq N + i^* + 1$, i.e. $\mathbb{X}_F(\mathcal{O}^h_\infty(\mathbb{X}), N, N) = \mathbb{X}_F(\mathbb{X}, N, P)$ for all $P \geq N + i^* + 1$. Both problems are strongly feasible.*

This result implies that if the system is LTI, the constraints are given by linear inequalities and the control law is linear $h(x_k) = Kx_k$, then it is probably more efficient to use a terminal set rather than setting $P - N$ to be larger than the determinedness index of $\mathcal{O}^h_\infty(\mathbb{X})$. This is because the number of inequalities describing $\mathcal{O}^h_\infty(\mathbb{X})$ will be no more than the extra number of inequalities in the MPC problem with $P - N \geq i^* + 1$. When computing $\mathcal{O}^h_\infty(\mathbb{X})$, it is nearly always the case that the number of inequalities are less, therefore making the MPC problem with $N = P$ and $\mathbb{T} = \mathcal{O}^h_\infty(\mathbb{X})$ more efficient than one with $P - N \geq i^* + 1$ and $\mathbb{T} = \mathbb{X}$.

5.8.2 Control Invariant Terminal Set

In general, if $\mathbb{T}$ is any control invariant subset of $\mathbb{X}$ and $P > N$, it is difficult to say anything about the feasibility of the MPC problem. However, the following theorem is useful if $\mathcal{O}^h_\infty(\mathbb{X})$ is complex and it is easy to obtain a simple expression for a positively invariant subset of $\mathcal{O}^h_\infty(\mathbb{X})$. It also says that an increase in the prediction horizon will not result in a decrease in the feasible set.

Theorem 5.6. *Let $P > N$ and the terminal constraint set $\mathbb{T}$ be a positively invariant set for the closed-loop system $x_{k+1} = f(x_k, h(x_k))$, i.e. $\mathcal{O}^h_\infty(\mathbb{T}) = \mathbb{T} \subseteq \mathcal{O}^h_\infty(\mathbb{X})$:*

1. *The feasible set is equal to the N-step* stabilisable *set to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$, i.e.*

   $$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{S}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))\,.$$

   The feasible set is control invariant;

2. *The MPC problem is strongly feasible;*

3. *Assume that N is fixed. By* increasing *the prediction horizon, the size of the feasible set* does not decrease. *If the set $\mathcal{KO}^h_\infty(\mathbb{X}, \mathbb{T})$ is finitely determined and the* difference *between the prediction and control horizons is larger than or equal to its determinedness index i^*, then the feasible set is equal to the N-step stabilisable set to $\mathcal{KO}^h_\infty(\mathbb{X}, \mathbb{T})$, i.e.*

   $$N + i^* \geq P_1 > P_2 > N \Rightarrow \mathbb{X}_F(\mathbb{T}, N, P_1) \supseteq \mathbb{X}_F(\mathbb{T}, N, P_2)$$

   and

   $$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{S}_N(\mathbb{X}, \mathcal{KO}^h_\infty(\mathbb{X}, \mathbb{T})), \forall P - N \geq i^*\,;$$

4. *Assume that $P - N$ is fixed. A larger* control *horizon results in a* larger *feasible set. The size of the feasible set stops increasing if and only if $\mathcal{S}_\infty(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))$ is finitely determined and the control horizon is larger than its determinedness index j^*, i.e.*

   $$j^* \geq N_1 > N_2 \Leftrightarrow \mathbb{X}_F(\mathbb{T}, N_1, P) \supset \mathbb{X}_F(\mathbb{T}, N_2, P)$$

   and

   $$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{S}_\infty(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})), \forall N \geq j^*\,.$$

   Furthermore, if $\mathcal{S}_\infty(\mathbb{X}, \mathcal{KO}^h_\infty(\mathbb{X}, \mathbb{T})) = \mathcal{S}_{s^*}(\mathbb{X}, \mathcal{KO}^h_\infty(\mathbb{X}, \mathbb{T})$, then

   $$\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{S}_\infty(\mathbb{X}, \mathcal{KO}^h_\infty(\mathbb{X}, \mathbb{T})), \forall N \geq s^*, P - N \geq i^*\,.$$

Proof.

1. Since $\mathbb{T}$ is positively invariant for $x_{k+1} = f(x_k, h(x_k))$, $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$ is also positively invariant and hence control invariant for $x_{k+1} = f(x_k, u_k)$. From Theorem 5.1 and the definitions of controllable and stabilisable sets, the feasible set is given by

$$\mathbb{X}_F(\mathbb{X}, N, P) = \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})) = \mathcal{S}_N(\mathbb{X}, \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}))\,.$$

2. Since $\mathbb{X}_F(\mathbb{T}, N, P)$ is control invariant for all $N \geq 0$, it follows from Theorem 5.2 that the MPC problem with $P > N \geq 1$ is strongly feasible.

3. If $P_1 > P_2$, then $\mathcal{KO}^h_{P_1-N}(\mathbb{X}, \mathbb{T}) \supseteq \mathcal{KO}^h_{P_2-N}(\mathbb{X}, \mathbb{T}) \supseteq \mathcal{KO}^h_1(\mathbb{X}, \mathbb{T}) \supseteq \mathbb{T}$, since $\mathbb{T}$ is positively invariant. As a result of applying Proposition 2.1 repetitively, it follows that for all N, $\mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P_1-N}(\mathbb{X}, \mathbb{T})) \supseteq \mathcal{K}_N(\mathbb{X}, \mathcal{KO}^h_{P_2-N}(\mathbb{X}, \mathbb{T}))$. This implies that $\mathbb{X}_F(\mathbb{T}, N, P_1) \supseteq \mathbb{X}_F(\mathbb{T}, N, P_2)$.

   The feasible set for the case when $P \geq N + i^*$ follows from the fact that $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T}) = \mathcal{KO}^h_{\infty}(\mathbb{X}, \mathbb{T}), \forall P - N \geq i^*$.

4. The proof proceeds along similar lines as for the third statement in Theorem 5.4, since the feasible set $\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{S}_N(\mathbb{X}, \mathbb{T})$ with a control invariant $\mathbb{T} = \mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$.

$\square$

The conclusion that increasing the difference between the control and prediction horizon could result in a larger feasible set, provided $\mathbb{T}$ is positively invariant for the system $x_{k+1} = f(x_k, h(x_k))$, is also reported in [DMMS00]. This idea of using different control and prediction horizons to reduce the computational burden in MPC, while enlarging the region of feasibility, is also discussed in [ZA98].

5.9 Nominal Stability in MPC

This chapter deals mainly with feasibility in MPC. As such, the results in this chapter do not necessarily imply anything about the stability of the closed-loop system. Strong feasibility does not imply stability.

If the feasible set is bounded and the MPC problem is strongly feasible, then one can think of the system as being nominally stable in a weak Lyapunov sense. However, one is often interested in obtaining stronger stability guarantees, such as asymptotic and exponential stability.

One way in which stability can be ensured for suboptimal solutions, is to add constraints to the MPC problem which ensure that the cost will not increase with time [SMR99]. With some additional assumptions on the system and cost function, it can be shown that feasibility implies asymptotic stability (and feasibility for all time).

Another way of ensuring exponential stability for suboptimal MPC, is to choose the terminal set to be a *contractive constraint* [dM00], e.g. $\mathbb{T} \triangleq \{\hat{x}_{P|k} \in \mathbb{R}^n \mid \|\hat{x}_{P|k}\| \leq \alpha \|\hat{x}_{0|k}\|\}$. Feasibility could possibly be a problem, but with a proper choice of the contraction parameter $\alpha \in [0, 1)$, feasibility for all time can be guaranteed.

The usual way of ensuring stability is via some kind of Lyapunov argument. The reader is referred to [MRRS00, DMS00, May00] for surveys on stability results in MPC which are based on this approach.

Usually the following assumptions[3], together with optimality of the solution at each time step, are made when using a direct Lyapunov argument to prove that the origin is an asymptotically stable fixed point with region of attraction $\mathbb{X}_F$ [MRRS00]:

1. $h(x) \in \mathbb{U}, \forall x \in \mathbb{T}$, i.e. the control law $h(x)$ is admissible in $\mathbb{T}$;
2. $f(x, h(x)) \in \mathbb{T}, \forall x \in \mathbb{T}$, i.e. $\mathbb{T}$ is positively invariant for the system $x_{k+1} = f(x, h(x))$;
3. There exists a positive c such that the stage cost $L(x, u) \geq c\|(x, u)\|^2$ and $L(0, 0) = 0$.
4. $F(x)$ is positive definite and $F(f(x, h(x))) - F(x) \leq -L(x, h(x)), \forall x \in \mathbb{T}$, i.e. $F(\cdot)$ is a control Lyapunov function in a neighbourhood of the origin;

It can be shown that the above assumptions allow one to use $V^*(x_k)$ as a Lyapunov function for the closed-loop system.

Strictly speaking, optimality at each time step (and hence uniqueness of the solution) is not needed to ensure convergence to the origin. The Lyapunov method is based on guaranteeing that at each time step, the new control sequence is such that the cost decreases, i.e.

$$V^*(x_{k+1}) < V^*(x_k)\,.$$

If the above conditions hold, and a feasible control sequence π_k^N was found at time k, then the control sequence

$$\pi_{k+1}^N = \left[\hat{u}'_{1|k}, \hat{u}'_{2|k}, \ldots, \hat{u}'_{N-1|k}, h(\hat{x}_{N|k})'\right]'$$

is feasible at time $k+1$ and results in a lower cost than the cost obtained at time k with π_k^N. By initialising the problem with the time-shifted control sequence found at the previous time step and appending it with $h(\cdot)$, convergence to the origin is guaranteed even if the solution is suboptimal.

If the following additional assumptions hold, then the origin of the closed-loop system is an exponentially stable fixed point: there exist positive constants a, b and c such that

1. $a\|x\|^2 \leq V^*(x) \leq b\|x\|^2, \forall x \in \mathbb{X}_F$;
2. $V^*(f(x, h(x))) - V^*(x) \leq -c\|x\|^2, \forall x \in \mathbb{X}_F$.

[3]Sometimes some continuity assumptions on f, L and F are included to guarantee existence of a unique solution to Problem 5.1 [Rao00, Chap. 5], though these can often be dropped in practice.

It is interesting to note that the condition $L(x,u) \geq c\|(x,u)\|^2 \geq c\|x\|^2$ is sufficient to guarantee exponential stability[4], provided $F(\cdot)$ is chosen such that $V^*(x) \leq b\|x\|^2, \forall x \in \mathbb{T}$ [MRRS00, App. A]. For example, often $F(x) = x'Q_F x$ with $Q_F \succ 0$ is the chosen control Lyapunov function.

Because of the fact that asymptotic stability is guaranteed, for all $x_0 \in \mathbb{X}_F \backslash \mathbb{T}$, the system is guaranteed to enter $\mathbb{T}$ after a finite number of steps. Exponential stability follows since the conditions for exponential stability are satisfied for all $x \in \mathbb{T}$.

As before, optimality and uniqueness of the solution is not required to guarantee exponential stability. The optimisation problem need only be initialised with the shifted feasible control sequence found at the previous time step.

5.10 Summary

A standard formulation for the nominal MPC regulator was given. The formulation allows for different control and prediction horizons as well as the inclusion of a terminal constraint set.

Even in the absence of disturbances, infeasibility occurs in MPC mainly because of the finite horizon nature of the problem. The feasible set of the MPC problem was defined and the reasons for infeasibility occurring in MPC were discussed.

The notion of strong feasibility was introduced. An MPC problem is strongly feasible if and only if it is feasible for all time, even if the solution is sub-optimal. A new sufficient condition was derived for guaranteeing strong feasibility, even if the terminal constraint is not control invariant. An equivalent statement of the condition is that if

$$\mathbb{X}_F(\mathbb{T}, N-1, P-1) \subseteq \mathbb{X}_F(\mathbb{T}, N, P),$$

then the MPC scheme with control and prediction horizons of N and P is strongly feasible.

The effect of the horizons and terminal constraint set on the geometrical properties of the feasible set was investigated. A new result on the possible need for some kind of "feasibility constraint" was found during this study. If the control and prediction horizons are equal and the terminal constraint set is equal to the state constraints, then the MPC problem can be made to be strongly feasible if and only if there exists a finite N such that

$$\mathbb{X}_F(\mathbb{X}, N, N) = \mathbb{X}_F(\mathbb{X}, N-1, N-1).$$

In general, such an N is not guaranteed to exist.

Finally, some well-known conditions on guaranteeing nominal stability in MPC were given.

[4]By replacing a with c.

Chapter 6

Robust Feasibility in Model Predictive Control

A necessary and sufficient condition for robust feasibility is given. The design of robustly feasible MPC controllers via the addition of a robustness constraint is discussed. A new necessary and sufficient and some new sufficient conditions are given for the proposed scheme to be robustly feasible. The implementation of the scheme for linear systems with parametric uncertainty is given. A procedure for computing a setpoint which is compatible with the constraints and disturbances is given.

6.1 Introduction

Recall the definitions for the feasible set and feasible control inputs, as in Section 5.4. This chapter deals with determining whether the MPC problem will be feasible for all time, despite any disturbance sequences that might occur. Feasibility must also be independent of the optimality of the solution to the MPC problem. The definition of strong feasibility in the presence of disturbances is extended to the following:

Definition 6.1 (Robust strongly feasible). The MPC problem is *robust strongly feasible* if and only if for all feasible state-input pairs and allowable disturbances the MPC problem is guaranteed to be feasible at the next time instant.

Sections 6.2–6.4 will mainly be concerned with a system given by

$$x_{k+1} = f_{xu}(x_k, u_k) + f_w(w_k)\,. \tag{6.1}$$

LTI systems with polytopic uncertainty and state disturbances will be considered in Section 6.5.

Before proceeding, define

$$\mathbb{D} \triangleq f_w(\mathbb{W})\,. \tag{6.2}$$

It is assumed that $0 \in \mathbb{W}$ and $0 = f_w(0)$ and that the state is measured.

6.2 A Necessary and Sufficient Condition for Robust Feasibility

This section deals with deriving a necessary and sufficient condition for strong robust feasibility for the MPC problem considered in Chapter 5.

By recalling the definition of the nominal reach set from Section 2.3

$$\mathcal{R}(\Omega) \triangleq \{x_{k+1} \in \mathbb{R}^n \mid \exists x_k \in \Omega, u_k \in \mathbb{U} : x_{k+1} = f_{xu}(x_k, u_k)\}, \tag{6.3}$$

it is possible to state the following:

Lemma 6.1. *For Problem 5.1 the following holds true:*

1. *For all $\hat{x}_{0|k} \in \mathbb{X}_F(\mathbb{T}, N, P)$ and for all corresponding feasible control inputs $\hat{u}_{0|k}$, $\hat{x}_{1|k} = f_{xu}(\hat{x}_{0|k}, \hat{u}_{0|k}) \in \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$.*
2. *For all $\hat{x}_{1|k} \in \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ there exist an $\hat{x}_{0|k} \in \mathbb{X}_F(\mathbb{T}, N, P)$ and a corresponding feasible control input $\hat{u}_{0|k}$ such that $\hat{x}_{1|k} = f_{xu}(\hat{x}_{0|k}, \hat{u}_{0|k})$.*
3. *For all states $x_{k+1} \in (\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \oplus \mathbb{D}$ there exist a state $x_k \in \mathbb{X}_F(\mathbb{T}, N, P)$, a corresponding feasible control input $\hat{u}_{0|k}$ and an allowable disturbance $w_k \in \mathbb{W}$ such that $x_{k+1} = f_{xu}(x_k, \hat{u}_{0|k}) + f_w(w_k)$.*

Proof.

1. This follows from Lemma 5.1 and the definition of the reach set $\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P))$. If $\hat{x}_{0|k} \in \mathbb{X}_F(\mathbb{T}, N, P)$ and a feasible control input has been found, then $\hat{x}_{1|k} \in \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P))$ has to be true, since a feasible control input is also admissible.
2. If $\hat{x}_{1|k} \in \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ then there exists a feasible control sequence $\{\hat{u}_{l|k}\}_1^{N-1}$ which will take the system from $\hat{x}_{1|k}$ to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$ in $N-1$ steps. Also, there exists an *admissible* control input $\hat{u}_{0|k}$ and an $\hat{x}_{0|k} \in \mathbb{X}_F(\mathbb{T}, N, P)$ such that $\hat{x}_{1|k} = f_{xu}(\hat{x}_{0|k}, \hat{u}_{0|k})$.

   However, the appended sequence $\{\hat{u}_{l|k}\}_0^{N-1}$ is a feasible control sequence which will drive the system from the given $\hat{x}_{0|k}$ to $\mathcal{KO}^h_{P-N}(\mathbb{X}, \mathbb{T})$ in N steps, via $\hat{x}_{1|k}$. Hence the same $\hat{u}_{0|k}$ is also *feasible*.
3. Recalling the definition of $\mathbb{D}$ and the Minkowski sum, it follows immediately that $\forall x_{k+1} \in (\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \oplus \mathbb{D}$ there exists an $\hat{x}_{1|k} \in \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ and a disturbance $w_k \in \mathbb{W}$ such that $x_{k+1} = \hat{x}_{1|k} + f_w(w_k)$.

   From the second result, it follows that there must also exist an $\hat{x}_{0|k} \in \mathbb{X}_F(\mathbb{T}, N, P)$ and a feasible control input $\hat{u}_{0|k}$ such that $\hat{x}_{1|k} = f_{xu}(\hat{x}_{0|k}, \hat{u}_{0|k})$. Since $\hat{x}_{0|k} = x_k$, the result follows.

□

Remark 6.1. Note that, in general,

$$\mathbb{X}_F(\mathbb{T}, N-1, P-1) \nsubseteq \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \ .$$

If this set inclusion does not hold, then

$$\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1) \neq \mathbb{X}_F(\mathbb{T}, N-1, P-1) \ .$$

Definition 6.2. Assuming no disturbances, the set of states $\mathcal{R}_{MPC}$ reachable from $\mathbb{X}_F(\mathbb{T}, N, P)$ using control inputs which are feasible for the MPC problem is

$$\mathcal{R}_{MPC} \triangleq \left\{x_{k+1} \in \mathbb{R}^n \mid \exists (x_k, \pi_k^N) \text{ which satisfies (5.2)} : x_{k+1} = f_{xu}(x_k, \hat{u}_{0|k})\right\} \ . \tag{6.4}$$

Given the above result and definition, the following result can be given.

Proposition 6.1. *Assuming no disturbances, the set of states reachable from* $\mathbb{X}_F(\mathbb{T}, N, P)$ *using feasible control inputs is*

$$\mathcal{R}_{MPC} = \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1) \ . \tag{6.5}$$

Proof. The fact that $\mathcal{R}_{MPC} \supseteq \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ follows from the second statement in Lemma 6.1.

If $\mathcal{R}_{MPC} \nsubseteq \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$, then either $x_{k+1} \notin \mathbb{X}_F(\mathbb{T}, N-1, P-1)$, which contradicts Lemma 5.1, or $x_{k+1} \notin \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P))$ which contradicts the definition of the reach set. Therefore, $\mathcal{R}_{MPC} \subseteq \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$. □

$\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P))$ is the set of states reachable from the feasible set $\mathbb{X}_F(\mathbb{T}, N, P)$ using *admissible* control inputs, while the set $\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ is the subset which is reachable using *feasible* control inputs. The set $\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \setminus \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ is the set of states reachable using admissible control inputs which are incompatible with the constraints of the MPC problem over the prediction horizon.

With this understanding of the set of states reachable using feasible control inputs, one can derive a necessary and sufficient condition for strong robust feasibility.

Theorem 6.1 (Robust strongly feasible). *The nominal MPC regulator is robust strongly feasible if and only if*

$$(\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \oplus \mathbb{D} \subseteq \mathbb{X}_F(\mathbb{T}, N, P) \ . \tag{6.6}$$

Proof. ($\Rightarrow$) If the problem is robust strongly feasible then for all $x_k \in \mathbb{X}_F(\mathbb{T}, N, P)$ it is true that for all corresponding feasible control inputs and for all allowable disturbances $x_{k+1} = \hat{x}_{1|k} + f_w(w_k) \in \mathbb{X}_F(\mathbb{T}, N, P)$.

Assume that the set inclusion does not hold. The third statement in Lemma 6.1 implies that for all

$$x_{k+1} \in \{(\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \oplus \mathbb{D}\} \setminus \mathbb{X}_F(\mathbb{T}, N, P)$$

there exist an $x_k \in \mathbb{X}_F(\mathbb{T}, N, P)$, a corresponding feasible control input and an allowable disturbance which will result in $x_{k+1} = f_{xu}(x_k, \hat{u}_{0|k}) + f_w(w_k) \notin \mathbb{X}_F(\mathbb{T}, N, P)$. This contradicts the assumption that the MPC problem is robust strongly feasible and the set inclusion therefore has to hold.

($\Leftarrow$) By the first statement in Lemma 6.1,

$$x_k \in \mathbb{X}_F(\mathbb{T}, N, P) \Rightarrow \hat{x}_{1|k} \in \mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1) \ .$$

After applying a feasible control input, then for all allowable disturbances it is true that $x_{k+1} = \hat{x}_{1|k} + f_w(w_k) \in (\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \oplus \mathbb{D}$. But this set is contained inside $\mathbb{X}_F(\mathbb{T}, N, P)$, hence the problem is feasible at time $k+1$, despite the presence of a disturbance.

□

This statement says that a nominal MPC scheme is robust strongly feasible if and only if the Minkowski sum of $\mathbb{D}$ and the intersection of $\mathbb{X}_F(\mathbb{T}, N-1, P-1)$ with the set reachable from $\mathbb{X}_F(\mathbb{T}, N, P)$ is a subset of the feasible set $\mathbb{X}_F(\mathbb{T}, N, P)$.

Corollary 6.1. *Assuming there are no disturbances present, then the nominal MPC regulator of Problem 5.1 is strongly feasible if and only if*

$$\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1) \subseteq \mathbb{X}_F(\mathbb{T}, N, P) \ . \tag{6.7}$$

Remark 6.2. This result is stronger than Theorem 5.2 and can be used to prove Theorem 5.2.

Theorem 6.1 is useful for analysing the robust feasibility of a given MPC regulator. If the nominal MPC problem satisfies this criterion, then no modifications need to be made in order to robustify the controller. By increasing the size of $\mathbb{W}$ until (6.6) is violated one can calculate the size of disturbances to which the closed-loop system will be robust.

Theorem 6.1 was derived for obtaining a condition for guaranteeing strong feasibility, i.e. *all* feasible (optimal and suboptimal) control inputs are considered. This result could therefore be conservative in practice. It is possible that the MPC scheme will reject a larger set of disturbances when implemented, due to the fact that the optimisation routine might try to steer the system towards the origin, rather than towards the boundary of $\mathcal{R}(\mathbb{X}_F(\mathbb{T}, N, P)) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)$.

The following example shows that a nominal MPC scheme can be robust strongly feasible without having to make any modifications to the original formulation.

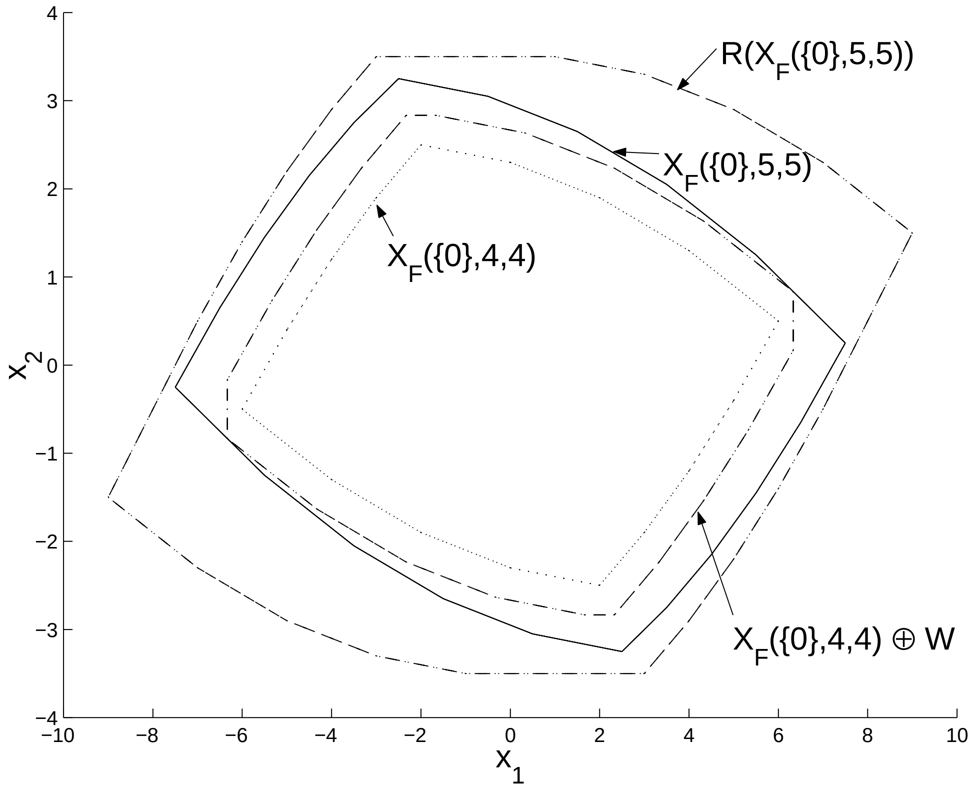

Figure 6.1: Plot showing that the given nominal MPC scheme is robust strongly feasible for an unknown disturbance with $\|w\|_\infty \leq 0.333$

Example 6.1. *Consider the system:*

$$x_{k+1} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} x_k + \begin{bmatrix} 1 & 0.5 \\ 0 & 0.5 \end{bmatrix} u_k + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w_k \,, \tag{6.8}$$

with no constraints on the states.

The input is constrained to $\|u\|_\infty \leq 1$ and the disturbance $\|w\|_\infty \leq \gamma$. The target set $\mathbb{T} = \{0_2\}$ and the control and prediction horizons are equal $P = N$.

Figure 6.1 is a plot of the reach set $\mathcal{R}(\mathbb{X}_F(\{0_2\}, 5, 5))$ and the feasible sets $\mathbb{X}_F(\{0_2\}, N, N) = \mathcal{S}_N(\mathbb{R}^2, \{0_2\})$ for $N = 4$ and 5. As the figure shows,

$$\mathcal{R}(\mathbb{X}_F(\{0_2\}, 5, 5)) \cap \mathbb{X}_F(\{0_2\}, 4, 4) = \mathbb{X}_F(\{0_2\}, 4, 4)\,.$$

It was found that

$$\mathbb{X}_F(\{0_2\}, 4, 4) \oplus E\mathbb{W} \subseteq \mathbb{X}_F(\{0_2\}, 5, 5) \text{ } \textit{if} \, \gamma \leq 0.333$$

and

$$\mathbb{X}_F(\{0_2\}, 4, 4) \oplus E\mathbb{W} \not\subseteq \mathbb{X}_F(\{0_2\}, 5, 5) \text{ } \textit{if} \, \gamma > 0.333\,.$$

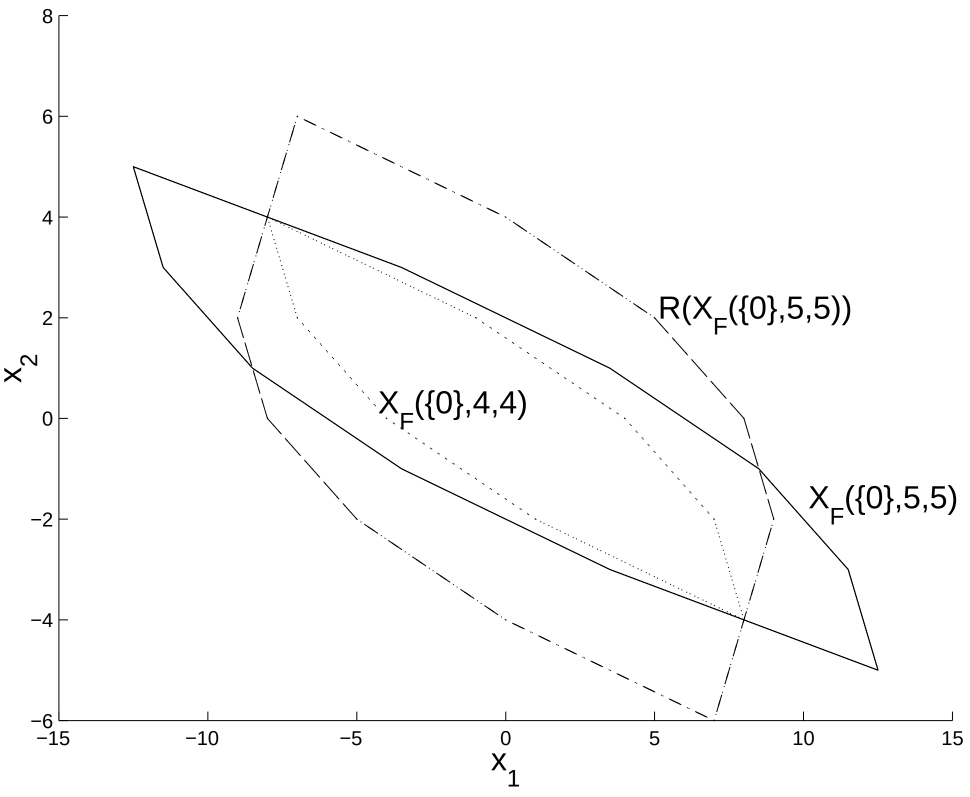

Figure 6.2: Plot showing that the given nominal MPC scheme for the double integrator is not robust strongly feasible for any size of disturbance

This implies that the nominal MPC regulator with $N = 5$ is robust strongly feasible for all $\|w\|_\infty \leq 0.333$

The next example demonstrates that a nominal MPC scheme for the double integrator is not robust strongly feasible given any arbitrarily small disturbance set. It is only strongly feasible in the nominal sense.

Example 6.2. *Consider the double integrator:*

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w_k\,, \tag{6.9}$$

with no constraints on the states. The input is constrained to $\|u\|_\infty \leq 1$ and the disturbance $\|w\|_\infty \leq \gamma$. The target set $\mathbb{T} = \{0_2\}$ and the control and prediction horizons are equal $P = N$.

Figure 6.2 is a plot of the reach set $\mathcal{R}(\mathbb{X}_F(\{0_2\}, 5, 5))$ and the feasible sets $\mathbb{X}_F(\{0_2\}, N, N) = \mathcal{S}_N(\mathbb{R}^2, \{0_2\})$ for $N = 4$ and 5.

As can be seen,

$$\mathcal{R}(\mathbb{X}_F(\{0_2\}, 5, 5)) \cap \mathbb{X}_F(\{0_2\}, 4, 4) = \mathbb{X}_F(\{0_2\}, 4, 4)$$

and $\mathbb{X}_F(\{0_2\}, 4, 4)$ *intersects the boundary of* $\mathbb{X}_F(\{0_2\}, 5, 5)$*, hence*

$$\mathbb{X}_F(\{0_2\}, 4, 4) \oplus E\mathbb{W} \nsubseteq \mathbb{X}_F(\{0_2\}, 5, 5)$$

for any $\gamma > 0$*.*

This implies that the given MPC controller with $N = 5$ *is not robust strongly feasible, even though it has nominal strong feasibility.*

It would be desirable to determine whether one can synthesise a predictive controller to be robust to an *a priori* determined disturbance set. The robust synthesis problem is the focus of the rest of this chapter.

6.3 Min-max Robust MPC Schemes

This section briefly describes the two main robust model predictive control (RMPC) schemes found in the literature - open-loop and feedback RMPC. Both approach the problem from a min-max point of view. The control tries to minimise the worst-case cost that could result from a future disturbance sequence.

In both cases it is usually assumed that the control and prediction horizons are equal, i.e.

$$N = P\,.$$

In order to guarantee that the RMPC scheme is robust strongly feasible the terminal constraint $\mathbb{T}$ is chosen to be a robust control invariant set

$$\mathbb{T} \subseteq \tilde{\mathcal{Q}}(\mathbb{T})\,.$$

The open-loop RMPC problem is given by:

Problem 6.1 (Open-loop RMPC). *Solve*

$$\min_{\pi_k^N} \max_{\{\hat{w}_{l|k} \in \mathbb{W}\}_0^{N-1}} F(\hat{x}_{N|k}) + \sum_{i=0}^{N-1} L(\hat{x}_{i|k}, \hat{u}_{i|k}) \tag{6.10}$$

subject to

$$\hat{x}_{l+1|k} = f(\hat{x}_{l|k}, \hat{u}_{l|k}, \hat{w}_{l|k}), \qquad \hat{x}_{0|k} = x_k \tag{6.11a}$$

$$\hat{x}_{l|k} \in \mathbb{X}, \quad \hat{u}_{l|k} \in \mathbb{U}, \qquad l = 0, \dots, N-1 \tag{6.11b}$$

$$\hat{x}_{N|k} \in \mathbb{T}\,. \tag{6.11c}$$

The decision variable is

$$\pi_k^N \triangleq \left[\hat{u}'_{0|k}, \hat{u}'_{1|k}, \dots, \hat{u}'_{N-1|k}\right]'\,. \tag{6.12}$$

Following the discussion in Section 2.6, the feasible set of open-loop RMPC is

$$\mathbb{X}_F^{ol} = \left\{x_0 \in \mathbb{R}^n \mid \exists \{u_k \in \mathbb{U}\}_0^{N-1} : \{x_k \in \mathbb{X}\}_0^{N-1}, x_N \in \mathbb{T}, \forall \{w_k \in \mathbb{W}\}_0^{N-1}\right\} . \tag{6.13}$$

The feedback RMPC problem is given by:

Problem 6.2 (Feedback RMPC). *Solve*

$$\min_{\pi_k^N} \max_{\{\hat{w}_{l|k} \in \mathbb{W}\}_0^{N-1}} F(\hat{x}_{N|k}) + \sum_{i=0}^{N-1} L(\hat{x}_{i|k}, \hat{u}_{i|k}) \tag{6.14}$$

subject to

$$\hat{x}_{l+1|k} = f(\hat{x}_{l|k}, \hat{u}_{l|k}, \hat{w}_{l|k}), \qquad \hat{x}_{0|k} = x_k \tag{6.15a}$$

$$\hat{x}_{l|k} \in \mathbb{X}, \quad \hat{u}_{l|k} \in \mathbb{U}, \qquad l = 0, \dots, N-1 \tag{6.15b}$$

$$\hat{u}_{l|k} = h(\hat{x}_{l|k}), \qquad l = 1, \dots, N-1 \tag{6.15c}$$

$$\hat{x}_{N|k} \in \mathbb{T} . \tag{6.15d}$$

The decision variable is

$$\pi_k^N \triangleq \left[\hat{u}'_{0|k}, h\left(\hat{x}_{1|k}\right)', \dots, h\left(\hat{x}'_{N-1|k}\right)\right]' . \tag{6.16}$$

The only real, but very important, difference between Problems 6.1 and 6.2 is the choice of decision variable. In open-loop RMPC the decision variable is a control *sequence* of length N and in feedback RMPC the decision variable is the control *law* $h(\cdot)$.

Some authors, such as [MRRS00], prefer using the more general sequence of control laws

$$\pi_k^N \triangleq \left[\hat{u}'_{0|k}, h_1\left(\hat{x}_{1|k}\right)', \dots, h_{N-1}\left(\hat{x}'_{N-1|k}\right)\right]'$$

as the decision variable for feedback RMPC. By choosing a single control law as the feedback policy implicitly puts a causality constraint [SM98] on the sequence of control laws in the sense that the control is independent on the path taken to reach the state, i.e. if $\hat{x}^1_{l|k}$ and $\hat{x}^2_{l|k}$ are the estimates of the state for two different disturbance sequences, then

$$\hat{x}^1_{l|k} = \hat{x}^2_{l|k} \Rightarrow \hat{u}^1_{l|k} = \hat{u}^2_{l|k} .$$

As is often the case, if the system is time-invariant, the terminal constraint is robust control invariant and the disturbance sequence does not depend on previous values of the disturbance, then no benefit in terms of the size of the feasible set of the RMPC scheme is gained from using a controller with memory. A memoryless control law is sufficient for guaranteeing that the region in which constraint satisfaction for all time can be guaranteed, is maximised. By solving for a single, memoryless controller as in Problem 6.2 the complexity of the min-max problem is reduced and the presentation and development of theoretical results is simplified. The idea of optimising over a single feedback policy for MPC of LTI systems is adopted in [KBM96, SM98], the former for polytopic uncertainty and the latter for bounded state disturbances.

Following the discussion in Section 2.6, since $\mathbb{T}$ is robust control invariant, the feasible set of the feedback RMPC problem stated above is

$$\mathbb{X}_F^{fb} = \tilde{\mathcal{K}}_N(\mathbb{X}, \mathbb{T})\,. \tag{6.17}$$

As mentioned before, the main difference is that open-loop RMPC tries to find a sequence of control inputs, whereas feedback RMPC tries to find a control law which will guarantee constraint satisfaction over the control horizon. A simplistic way of appreciating the difference between the two schemes is to realise that open-loop RMPC assumes that the control sequence computed at time k will be applied blindly for N steps, without measuring the state and recomputing a new control sequence at each of the subsequent time steps. A *single* control sequence is chosen such that *for all* allowable disturbance sequences the constraints will be satisfied.

Clearly, by choosing different control sequences for different disturbance sequences will be less conservative. Feedback RMPC takes into account that at each point in the future the state will be measured to determine which disturbance has occurred. Based on this knowledge of where the actual state lies compared to the previously predicted range of possible values, a different control can be computed. Feedback RMPC assumes that feedback will be used over the next N steps and incorporates this into the prediction. As a result, the feasible set of feedback RMPC is often much larger than for open-loop RMPC, i.e.

$$\mathbb{X}_F^{ol} \subset\subset \mathbb{X}_F^{fb}\,.$$

Though feedback RMPC is in principle a good idea, it is fairly difficult to implement and computationally expensive. Min-max RMPC schemes require determining all possible future evolutions of the disturbance sequence over the control horizon. Even if some special properties about the system and disturbances hold, such as linearity and convexity [SM98], the computations quickly become intractable as the horizon is increased.

The aim of the next section is to describe a method for robustifying MPC via the inclusion of a “robustness constraint”. The proposed scheme does not suffer from having to predict all possible future disturbance evolutions on-line, but relies on the off-line computation of a robust control invariant set. The addition of this constraint to the original MPC problem usually increases the computational load by a minimal amount compared to traditional min-max RMPC schemes. Furthermore, in principle the feasible set of the modified MPC problem can be made to be as large as possible.

6.4 Robust Feasibility via a Robustness Constraint

The idea of adding a constraint to the nominal MPC problem to robustify the system against persistent state disturbances was proposed in [CZ99]. This approach to solving the robust feasibility problem has a number of benefits over the traditional robust MPC schemes.

This section also gives a new necessary and sufficient condition for the MPC problem with a robustness constraint to be robust strongly feasible. Some new sufficient conditions are also given in Theorem 6.4, which are generalisations of [CZ00c, Thm. 5] and therefore less conservative. In Section 6.5 it will be shown that this approach can also be extended to LTI systems with parametric uncertainty and state disturbances.

The original MPC problem of Chapter 5 is modified by placing an additional constraint on $\hat{x}_{1|k}$. Typically the constraint is derived from a robust control invariant set contained in $\mathbb{X}$ or $\mathbb{X}_F(\mathbb{T}, N, P)$, depending on the problem at hand. It is then required that $\hat{x}_{1|k}$ lie inside the Pontryagin difference of this pre-computed set and the disturbance set. As will be shown in the sequel, this constraint allows one to modify a nominal MPC scheme in order to guarantee robust strong feasibility.

Problem 6.3 (MPC with a Robustness Constraint). *[CZ99, CZ00c] Solve*

$$\min_{\pi_k^N} F(\hat{x}_{P|k}) + \sum_{i=0}^{P-1} L(\hat{x}_{i|k}, \hat{u}_{i|k}) \tag{6.18}$$

subject to

$$\hat{x}_{l+1|k} = f_{xu}(\hat{x}_{l|k}, \hat{u}_{l|k}), \qquad \hat{x}_{0|k} = x_k \tag{6.19a}$$

$$\hat{x}_{1|k} \in \mathbb{X}_R \sim \mathbb{D} \tag{6.19b}$$

$$\hat{x}_{l|k} \in \mathbb{X}, \hat{u}_{l|k} \in \mathbb{U}, \qquad l = 0, \dots, P-1 \tag{6.19c}$$

$$\hat{u}_{l|k} = h(\hat{x}_{l|k}), \qquad l = N, \dots, P-1 \tag{6.19d}$$

$$\hat{x}_{P|k} \in \mathbb{T} \subseteq \mathbb{X} \tag{6.19e}$$

The decision variable in the above MPC problem is the control sequence

$$\pi_k^N = \left[\hat{u}'_{0|k}, \hat{u}'_{1|k}, \dots, \hat{u}'_{N-1|k}\right]' .$$

The problem posed above is the same as Problem 5.1, but with the robustness constraint (6.19b) added to the original MPC constraints. It is assumed that $\mathbb{X}_R \subseteq \mathbb{X}$. No assumption about strong feasibility of the original MPC problem is made.

Remark 6.3. Note that, in contrast with the min-max RMPC schemes, Problem 6.3 does not minimise the worst case cost, nor does it make use of explicit predictions of the future behaviour of the disturbance. Typically, $\mathbb{X}_R$ is a robust control invariant set and by choosing the parameters appropriately, strong robust feasibility can be guaranteed. The effect of the disturbance is implicitly taken into account by requiring that the predicted state at the next time instant lie inside a robust control invariant set.

The feasible set[1] is defined in a similar fashion as in Section 5.2 to be the set of states for which a control sequence exists which will satisfy the constraints in Problem 6.3.

Theorem 6.2 (Feasible set of MPC with robustness constraint). *The feasible set $\mathbb{X}_F^{rc}$ of the MPC controller defined by Problem 6.3 is given by*

$$\begin{aligned}\mathbb{X}_F^{rc} &= \mathcal{K}_1(\mathbb{X}, (\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \\ &= \mathcal{Q}((\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \cap \mathbb{X}\,.\end{aligned} \tag{6.20}$$

Proof. The fact that $\hat{x}_{1|k} \in \mathbb{X}_F(\mathbb{T}, N-1, P-1))$ follows as with Theorem 5.1. Additionally, it is required that $\hat{x}_{1|k} \in \mathbb{X}_R \sim \mathbb{D}$, hence $\hat{x}_{0|k} \in \mathcal{Q}((\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1))$. Finally, $\hat{x}_{0|k} \in \mathbb{X}$, hence $\hat{x}_{0|k} \in \mathcal{Q}((\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1)) \cap \mathbb{X}$ and the result follows from the definition of controllable sets. □

Remark 6.4. If the constraint $\hat{x}_{0|k} \in \mathbb{X}$ is removed, then

$$\mathbb{X}_F^{rc} = \mathcal{Q}((\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1))\,. \tag{6.21}$$

Furthermore, since $\mathbb{X}_R \sim \mathbb{D} \subseteq \mathbb{X}$, the explicit constraint $\hat{x}_{1|k} \in \mathbb{X}$ can be removed without changing the problem.

The next result is a necessary and sufficient condition for Problem 6.3 to be strongly feasible.

Theorem 6.3 (Feasibility of MPC with robustness constraint). *Problem 6.3 is robust strongly feasible if and only if*

$$\left(\mathcal{R}\left(\mathbb{X}_F^{rc}\right) \cap (\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F\left(\mathbb{T}, N-1, P-1\right)\right) \oplus \mathbb{D} \subseteq \mathbb{X}_F^{rc}\,. \tag{6.22}$$

Proof. The proof follows the same argument as that of Theorem 6.1. □

The important thing to note about this result is that it does not require $\mathbb{X}_R$ or the original $\mathbb{X}_F$ to be robust control invariant and can hence also be used for analysis. If this condition is satisfied, then $\mathbb{X}_F^{rc}$ is robust control invariant. Furthermore, none of the following conditions on their own are necessary nor sufficient for Problem 6.3 to be robust strongly feasible, since one can find counter-examples to these conditions:

$$\begin{aligned}\mathbb{X}_R &\subseteq \mathbb{X}_F(\mathbb{T}, N, P)\,, \\ \mathbb{X}_R \sim \mathbb{D} &\subseteq \mathbb{X}_F(\mathbb{T}, N, P)\,.\end{aligned}$$

However, the following theorem provides some sufficient conditions to guarantee that Problem 6.3 is robust strongly feasible.

[1]The notation $\mathbb{X}_F(\mathbb{T}, N, P)$ is still meant to denote the feasible set of Problem 5.1.

Theorem 6.4 (Sufficient conditions for MPC with a robustness constraint).

1. *If $\mathbb{X}_R$ is robust control invariant and*
$$\mathbb{X}_R \sim \mathbb{D} \subseteq \mathbb{X}_F(\mathbb{T}, N-1, P-1)\,, \tag{6.23}$$
then Problem 6.3 is robust strongly feasible and
$$\mathbb{X}_F^{rc} = \mathcal{K}_1(\mathbb{X}, \mathbb{X}_R \sim \mathbb{D}) = \mathcal{Q}(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}\,. \tag{6.24}$$

2. *If $\mathbb{X}_R$ is robust control invariant and*
$$\mathbb{X}_R \subseteq \mathbb{X}_F(\mathbb{T}, N-1, P-1)\,, \tag{6.25}$$
then Problem 6.3 is robust strongly feasible and
$$\mathbb{X}_F^{rc} = \mathcal{K}_1(\mathbb{X}, \mathbb{X}_R \sim \mathbb{D}) = \mathcal{Q}(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}\,. \tag{6.26}$$

3. *If*
$$\mathbb{X}_R \subseteq \mathbb{X}_F(\mathbb{T}, N, P) \tag{6.27}$$
and
$$\mathbb{X}_F(\mathbb{T}, N-1, P-1) \subseteq \mathbb{X}_R \sim \mathbb{D}\,, \tag{6.28}$$
then Problem 6.3 is robust strongly feasible and
$$\mathbb{X}_F^{rc} = \mathbb{X}_F(\mathbb{T}, N, P)\,. \tag{6.29}$$

Proof.

1. If $\mathbb{X}_R \sim \mathbb{D} \subseteq \mathbb{X}_F(\mathbb{T}, N-1, P-1)$, then it follows that $(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1) = \mathbb{X}_R \sim \mathbb{D}$ and hence from Theorem 6.2 that $\mathbb{X}_F^{rc} = \mathcal{Q}(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}$.

   Recall that $(\mathbb{X}_R \sim \mathbb{D}) \oplus \mathbb{D} \subseteq \mathbb{X}_R \subseteq \mathbb{X}$ and from the geometric condition for robust control invariance that $\mathbb{X}_R \subseteq \tilde{\mathcal{Q}}(\mathbb{X}_R) = \mathcal{Q}(\mathbb{X}_R \sim \mathbb{D})$, which implies that $\mathbb{X}_R \subseteq \mathcal{Q}(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}$.

   If $x_k \in \mathbb{X}_F^{rc}$, then for all feasible inputs $\hat{x}_{1|k} \in \mathbb{X}_R \sim \mathbb{D}$ and for all allowable disturbances $x_{k+1} \in (\mathbb{X}_R \sim \mathbb{D}) \oplus \mathbb{D} \subseteq \mathbb{X}_R \subseteq \mathcal{Q}(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X} = \mathbb{X}_F^{rc}$.

2. This result follows immediately from the first statement, since $\mathbb{X}_R \sim \mathbb{D} \subseteq \mathbb{X}_F(\mathbb{T}, N-1, P-1)$.

3. If $\mathbb{X}_F(\mathbb{T}, N-1, P-1) \subseteq \mathbb{X}_R \sim \mathbb{D}$, then it follows that $(\mathbb{X}_R \sim \mathbb{D}) \cap \mathbb{X}_F(\mathbb{T}, N-1, P-1) = \mathbb{X}_F(\mathbb{T}, N-1, P-1)$ and hence from Theorem 6.2 that $\mathbb{X}_F^{rc} = \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N-1, P-1)) \cap \mathbb{X} = \mathbb{X}_F(\mathbb{T}, N, P)$.

If $\mathbb{X}_F(\mathbb{T}, N-1, P-1) \subseteq \mathbb{X}_R \sim \mathbb{D}$, then $\mathbb{X}_F(\mathbb{T}, N-1, P-1) \oplus \mathbb{D} \subseteq (\mathbb{X}_R \sim \mathbb{D}) \oplus \mathbb{D}$. Recall also that $(\mathbb{X}_R \sim \mathbb{D}) \oplus \mathbb{D} \subseteq \mathbb{X}_R \subseteq \mathbb{X}_F(\mathbb{T}, N, P)$.

If $x_k \in \mathbb{X}_F^{rc}$, then for all feasible inputs $\hat{x}_{1|k} \in \mathbb{X}_F(\mathbb{T}, N-1, P-1) \subseteq \mathbb{X}_R \sim \mathbb{D}$ and for all allowable disturbances, $x_{k+1} \in \mathbb{X}_F(\mathbb{T}, N-1, P-1) \oplus \mathbb{D} \subseteq (\mathbb{X}_R \sim \mathbb{D}) \oplus \mathbb{D} \subseteq \mathbb{X}_R \subseteq \mathbb{X}_F(\mathbb{T}, N, P) = \mathbb{X}_F^{rc}$.

□

Remark 6.5. The method for constructing $\mathbb{X}_R$ given in [CZ00c] satisfies the second condition in Theorem 6.4. Given an *a priori* chosen $N = P$ and $\mathbb{T} = \mathcal{O}_\infty^h(\mathbb{X})$, the authors propose setting $\mathbb{X}_R = \tilde{\mathcal{S}}_{M^*}(\mathbb{X}, \tilde{\mathcal{O}}_\infty^h(\mathbb{X}))$, where M^* is the largest M such that $\tilde{\mathcal{S}}_M(\mathbb{X}, \tilde{\mathcal{O}}_\infty^h(\mathbb{X})) \subseteq \mathcal{S}_{N-1}(\mathbb{X}, \mathcal{O}_\infty^h(\mathbb{X})) = \mathbb{X}_F(\mathbb{T}, N-1, N-1)$. A better choice would be to set $\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X}_F(\mathbb{T}, N-1, N-1))$ or to set $\mathbb{X}_R = \tilde{\mathcal{S}}_\infty(\mathbb{X}_F(\mathbb{T}, N-1, N-1), \tilde{\mathcal{O}}_\infty^h(\mathbb{X}))$, since it is easy to show via contradiction that $\tilde{\mathcal{S}}_{M^*}(\mathbb{X}, \tilde{\mathcal{O}}_\infty^h(\mathbb{X})) \subseteq \tilde{\mathcal{S}}_\infty(\mathbb{X}_F(\mathbb{T}, N-1, N-1), \tilde{\mathcal{O}}_\infty^h(\mathbb{X})) \subseteq \tilde{\mathcal{C}}_\infty(\mathbb{X}_F(\mathbb{T}, N-1, N-1))$.

Remark 6.6. If $\mathbb{X}_F(\mathbb{T}, N-1, P-1) = \mathcal{S}_{N-1}(\mathbb{X}, \mathcal{O}_\infty^h(\mathbb{X}))$ as in Remark 6.5, then another method which improves on the one given in [CZ00c] is to find M^*, the largest M such that $\tilde{\mathcal{S}}_M(\mathbb{X}, \tilde{\mathcal{O}}_\infty^h(\mathbb{X})) \sim \mathbb{D} \subseteq \mathcal{S}_{N-1}(\mathbb{X}, \mathcal{O}_\infty^h(\mathbb{X}))$ and setting $\mathbb{X}_R = \tilde{\mathcal{S}}_{M^*}(\mathbb{X}, \tilde{\mathcal{O}}_\infty^h(\mathbb{X}))$. Similarly, an improvement on the latter scheme is to find M^*, the largest M such that $\tilde{\mathcal{S}}_M(\mathbb{X}_F(\mathbb{T}, N, P), \tilde{\mathcal{O}}_\infty^h(\mathbb{X})) \sim \mathbb{D} \subseteq \mathcal{S}_{N-1}(\mathbb{X}, \mathcal{O}_\infty^h(\mathbb{X}))$ and setting $\mathbb{X}_R = \tilde{\mathcal{S}}_{M^*}(\mathbb{X}_F(\mathbb{T}, N, P), \tilde{\mathcal{O}}_\infty^h(\mathbb{X}))$. Strong robust feasibility is then guaranteed in both cases by the first condition in Theorem 6.4.

Remark 6.7. The last result in Theorem 6.4 does not require $\mathbb{X}_R$ to be robust control invariant. Furthermore, the robustness constraint is effectively redundant and the constraint $\hat{x}_{1|k} \in \mathbb{X}_R \sim \mathbb{D}$ can be removed if the third statement holds for the given MPC scheme with robustness constraint. Theorem 6.1 then guarantees strong robust feasibility.

6.4.1 Implementation of MPC with a Robustness Constraint

The idea of using a constraint to guarantee feasibility can be implemented in one of two ways:

- Given $\mathbb{X}_R$, choose an N, P and $\mathbb{T}$ such that one of the conditions in Theorems 6.3 or 6.4 holds;
- Given an MPC controller, choose $\mathbb{X}_R$ such that one of the conditions in Theorems 6.3 or 6.4 holds.

Which approach is the most appropriate is dependent on the structure of the system. For example, for a general nonlinear system, if $\mathbb{X}_R$ is the maximal robust control invariant set, then the first approach might not work if the terminal constraint is chosen such that $\mathcal{K}_\infty(\mathbb{X}, \mathbb{T}) \subseteq \mathbb{X}_R \sim \mathbb{D}$, since no choice of control horizon will result in $\mathbb{X}_R \sim \mathbb{D} \subseteq \mathbb{X}_F(\mathbb{T}, N-1, P-1)$. On the other hand, experience has shown that for LTI systems there nearly always exists a choice of horizons which results in one of the conditions in Theorems 6.3 or 6.4 holding.

The first approach can be implemented as follows:

1. Given: $\mathbb{X}_R$ and the resulting $\mathbb{X}_R \sim \mathbb{D}$;
2. Choose/compute a terminal constraint $\mathbb{T}$, the maximum allowed control horizon $N_{\max}$ and a value for the difference $P - N$;
3. Set $N \leftarrow 1$;
4. Compute $\mathbb{X}_F(\mathbb{T}, N-1, P-1)$;
5. Compute $\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N-1, P-1)) \cap \mathbb{X}$;
6. If any of the robust feasibility conditions in Theorems 6.3 or 6.4 hold, then stop;
7. If $N < N_{\max}$ then set $N \leftarrow N + 1$ and go to step 5, else go to step 2.

The second approach can be implemented as follows:

1. Given: a terminal constraint $\mathbb{T}$ and values for the horizons P and N.
2. Compute $\mathbb{X}_F(\mathbb{T}, N-1, P-1)$ and $\mathbb{X}_F(\mathbb{T}, N, P) = \mathcal{Q}(\mathbb{X}_F(\mathbb{T}, N-1, P-1)) \cap \mathbb{X}$;
3. Compute $\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X})$;
4. If any of the robust feasibility conditions in Theorems 6.3 or 6.4 hold, then stop;
5. Compute $\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X}_F(\mathbb{T}, N, P))$;
6. If any of the robust feasibility conditions in Theorems 6.3 or 6.4 hold, then stop;
7. Compute $\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X}_F(\mathbb{T}, N-1, P-1))$;

Provided $\tilde{\mathcal{C}}_\infty(\mathbb{X}_F(\mathbb{T}, N-1, P-1)) \neq \emptyset$, the last choice for $\mathbb{X}_R$ will always work, since the second statement in Theorem 6.4 will hold. Obviously, alternative choices for $\mathbb{X}_R$ are possible, such as those proposed in Remarks 6.5 and 6.6.

6.4.2 Benefits of MPC with a Robustness Constraint

The following are some benefits of using the robustness constraint approach in guaranteeing robust feasibility in MPC:

- Traditional robust MPC schemes based on the min-max approaches discussed in Section 6.3 typically result in computationally impractical implementations. This is because, as the horizon increases, the number of possible sequences of disturbances can grow exponentially and often also the number of steps required in solving the min-max problem.

What the robustness constraint approach offers is a guarantee for robust feasibility with the addition of only a minimal amount of on-line computational effort. For example, if the cost function is quadratic, the system is LTI and the constraints are given by polyhedra, then a single QP is sufficient for solving for the MPC control action.

- An additional benefit of the robustness constraint approach is that one can robustify an existing MPC controller without having to redefine the problem in a substantial way. A new choice of terminal constraint, horizons or cost function is not necessary.
- The use of a terminal constraint $\mathbb{T}$ alone does not give a robust feasibility guarantee. The robustness constraint $\mathbb{X}_R \sim \mathbb{D}$ does away with the need for relying on a terminal constraint to guarantee feasibility.

  However, often the terminal constraint is used to provide a stability guarantee. The robustness constraint allows one to seek alternative ways of guaranteeing stability without having to rely on the use of a terminal constraint.
- In principle (particularly for LTI systems) the MPC problem can be made to be robust strongly feasible over as large a subset of $\mathbb{X}$ as possible. For example, by setting $\mathbb{T} = \mathbb{X}$ and $N = P$ one can choose $\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X})$. The MPC problem will be robust strongly feasible with a feasible set $\mathbb{X}_F^{rc} = \mathcal{Q}(\tilde{\mathcal{C}}_\infty(\mathbb{X}) \sim \mathbb{D}) \cap \mathbb{X}$, for any choice of N.

6.5 LTI Systems with Parametric Uncertainty

If the system is LTI with no uncertainty in the matrices (A, B) and only additive state disturbances are present, then all the results in Sections 6.2 and 6.4 can be used to guarantee robust strong feasibility. However, if there is parametric uncertainty in (A, B) as in Section 3.1, then a few small modifications need to be made to Problem 6.3 and care has to be taken which matrices are to be used in the different parts of the MPC problem.

It is assumed that the actual system is given by

$$x_{k+1} = Ax_k + Bu_k + Ew_k \tag{6.30}$$

where

$$(A, B) \in \Delta \triangleq \operatorname{conv}\left\{(A_1, B_1), \dots, \left(A_p, B_p\right)\right\} \tag{6.31}$$

and $w_k \in \mathbb{W}$, where $\mathbb{W}$ is a polytope containing the origin.

Before proceeding, one has to choose a nominal matrix pair

$$(A_0, B_0) \in \Delta \tag{6.32}$$

which will be used in the constraints and cost function of the MPC problem. The MPC problem then becomes:

Problem 6.4 (Robustly feasible MPC for LTI systems with parametric uncertainty). *Solve*

$$\min_{\pi_k^N} F(\hat{x}_{P|k}) + \sum_{i=0}^{P-1} L(\hat{x}_{i|k}, \hat{u}_{i|k}) \tag{6.33}$$

subject to

$$\begin{aligned}
&\hat{x}_{l+1|k} = A_0\hat{x}_{l|k} + B_0\hat{u}_{l|k}, && \hat{x}_{0|k} = x_k && \text{(6.34a)}\\
&A_j\hat{x}_{0|k} + B_j\hat{u}_{0|k} \in \tilde{\lambda}\mathbb{X}_R \sim \mathbb{D}, && j = 1, \dots, p && \text{(6.34b)}\\
&\hat{x}_{l|k} \in \mathbb{X}, \hat{u}_{l|k} \in \mathbb{U} && l = 0, \dots, P-1 && \text{(6.34c)}\\
&\hat{u}_{l|k} = h(\hat{x}_{l|k}), && l = N, \dots, P-1 && \text{(6.34d)}\\
&\hat{x}_{P|k} \in \mathbb{T} \subseteq \mathbb{X} && && \text{(6.34e)}
\end{aligned}$$

The decision variable in the MPC problem is still the control sequence $\pi_k^N = \left[\hat{u}'_{0|k}, \hat{u}'_{1|k}, \dots, \hat{u}'_{N-1|k}\right]'$. The vertices of the matrix polytope are included in the robustness constraint (6.34b). Due to convexity, if the constraints are satisfied for all vertices, then the constraints will be satisfied for all points contained in the convex hull.

Some slight modifications need to be made to the results in Section 6.4. It is easy to verify that the feasible set of the robust MPC problem is given by

$$\mathbb{X}_F^{rc} = \left\{ x_k \in \mathbb{X} \,\middle|\, \exists u_k \in \mathbb{U} : \begin{array}{l} A_j x_k + B_j u_k \in \tilde{\lambda}\mathbb{X}_R \sim \mathbb{D}, \, j = 1, \dots, p, \\ A_0 x_k + B_0 u_k \in \mathbb{X}_F(\mathbb{T}, N-1, P-1) \end{array} \right\}, \tag{6.35}$$

where $\mathbb{X}_F(\mathbb{T}, N-1, P-1)$ is computed using the nominal matrix pair (A_0, B_0) and $\mathbb{X}_F^{rc}$ can then be computed using a projection method.

Due to plant-model mismatch the necessary and sufficient condition of Theorem 6.3 does not necessarily hold. However, due to convexity the sufficient conditions of Theorem 6.4 hold if $\mathbb{X}_R$ is λ-contractive and $\tilde{\lambda}$ is such that $\lambda \leq \tilde{\lambda} \leq 1$. The following substitutions need to be made:

$$\mathbb{X}_R \sim \mathbb{D} \leftarrow \tilde{\lambda}\mathbb{X}_R \sim \mathbb{D}$$

and

$$\mathcal{Q}(\mathbb{X}_R \sim \mathbb{D}) \leftarrow \mathcal{Q}_\Delta(\tilde{\lambda}\mathbb{X}_R \sim \mathbb{D})\,.$$

6.6 Robust Stability

As with nominal stability, it is desirable to obtain stability results for the various RMPC schemes. The conditions in Section 5.9 need to be strengthened as follows to guarantee robust asymptotic stability for the open-loop and feedback RMPC schemes:

1. $h(x) \in \mathbb{U}, \forall x \in \mathbb{T}$, i.e. the control law $h(x)$ is admissible in $\mathbb{T}$;
2. $f(x, h(x), w) \in \mathbb{T}, \forall x \in \mathbb{T}, \forall w \in \mathbb{W}$, i.e. $\mathbb{T}$ is robust positively invariant for the system $x_{k+1} = f(x, h(x))$;
3. There exists a positive c such that the stage cost $L(x, u) \geq c\|(x, u)\|^2$ and $L(0, 0) = 0$.
4. $F(x)$ is positive definite and $F(f(x, h(x), w)) - F(x) \leq -L(x, h(x)), \forall x \in \mathbb{T}, \forall w \in \mathbb{W}$, i.e. $F(\cdot)$ is a robust control Lyapunov function in a neighbourhood of the origin;

These conditions guarantee that the worst-case cost in the min-max RMPC schemes will decrease at each time step. Robust asymptotic stability follows.

However, with the robustness constraint MPC scheme of Section 6.4, the worst-case cost does not come into play, since the scheme does not rely on explicit predictions of the disturbance. Nevertheless, a robust stability result can be obtained.

Definition 6.3 (Asymptotic stability of a perturbed system). [SRM97] The origin is an asymptotically stable fixed point of the perturbed system $\bar{x}_{k+1} = G(\bar{x}_k) + w_k$ if and only if:

1. there exists strictly positive constants r and μ such that the solution of the perturbed system $\bar{x}_{k+1} = G(\bar{x}_k) + w_k$ remains in a ball B_r for all $k \geq 0$, if $\bar{x}_0 \in B_q$, $q \neq r$ for some q, and $w_k \in B_\mu$ for all k;
2. the solution of the perturbed system $\bar{x}_{k+1} = G(\bar{x}_k) + w_k$ converges asymptotically to the origin, if $\bar{x}_0 \in B_q$, $w_k \in B_\mu$ for all k and $w_k \to 0$ as $k \to \infty$.

Theorem 6.5 (Asymptotic stability of a perturbed system). *[SRM97] Let $G : \mathbb{R}^n \mapsto \mathbb{R}^n$ satisfy a Lipschitz condition in a neighbourhood of the origin with $F(0) = 0$. If the origin is an exponentially stable fixed point of $x_{k+1} = G(x_k)$, it is an asymptotically stable fixed point of the perturbed system $\bar{x}_{k+1} = G(\bar{x}_k) + w_k$.*

Let $G(x_k) = f_{xu}(x_k, \kappa(x_k))$ be the description of the nominal system in closed-loop with the robustness constraint MPC control law of Section 6.4, where the parameters have been chosen such that the origin is an exponentially stable fixed point (see Section 5.9). An additional Lipschitz continuity assumption on $G(\cdot)$ guarantees robust asymptotic stability of the system, provided w_k is an asymptotically decaying disturbance and $\mathbb{W}$ is bounded.

An unresolved question is for which class of systems the Lipschitz continuity of the resulting closed-loop system holds. For LTI systems where the inequalities are linear and the cost is quadratic, the optimisation problem becomes a QP. It can be shown that the solution of the QP is Lipschitz continuous over the feasible set [Hag79, Mea94, BMDP00a]. As a result, the closed-loop system is Lipschitz continuous and the stability result is applicable.

This observation that if the system is LTI then the resulting closed-loop system is robust asymptotically stable, is also noted in [SR98, Thm. 2]. However, feasibility for all time is not guaranteed in [SR98]. With the addition of a robustness constraint to the original (nominal) MPC problem, both strong robust feasibility and robust asymptotic stability can be guaranteed for LTI systems with an asymptotically decaying disturbance.

As mentioned in [Mea94, SRM97, SM98], the output feedback case can be addressed by cascading an asymptotically stable state estimator with an exponentially stable MPC scheme. The errors in the estimation can be treated as disturbances on the state. By assuming a bound on the effect of the errors on the state and incorporating this into the MPC controller, a stable closed-loop system results with guaranteed feasibility.

6.7 Output Feedback

The case of output feedback has always been one of the main problems in MPC because of the fact that there is always some error between the actual and estimated state. All guarantees of feasibility, even if there is no plant-model mismatch or disturbances, are lost if the initial state estimate differs from the true state.

However, if one has an asymptotically stable estimator and one can place a bound on the error, then it is easy to see that by defining the error as a bounded state disturbance of an *a priori* chosen size, then one can synthesise a predictive controller with a robustness constraint which incorporates this fact. Furthermore, if the nominal MPC scheme is exponentially stable and satisfies some Lipschitz conditions, then the origin of the estimator-controller-plant system is an asymptotically stable fixed point, as mentioned in Section 6.6.

To see why errors in output feedback can be treated as a state-disturbance, consider the LTI system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k \end{aligned}$$

in closed-loop with a feedback control law

$$u_k = K\hat{x}_{k|k} \,,$$

where the estimate of the current state $\hat{x}_{k|k}$ is provided by an observer of the form

$$\begin{aligned} \hat{x}_{k|k-1} &= A\hat{x}_{k-1|k-1} + Bu_{k-1} \\ \hat{x}_{k|k} &= l(y_k, \hat{x}_{k|k-1}) \,. \end{aligned}$$

The error between the estimated and the actual state is given by

$$e_k \triangleq \hat{x}_{k|k} - x_k \,. \tag{6.36}$$

If one implements the control law, then the closed-loop system

$$\begin{aligned} x_{k+1} &= Ax_k + BK(x_k + e_k) \\ y_k &= Cx_k \end{aligned}$$

is equivalent to

$$\begin{aligned} x_{k+1} &= (A + BK)x_k + Ew_k \\ y_k &= Cx_k \, , \end{aligned}$$

with $E = BK$ and $w_k = e_k$. The error in the estimate is scaled by the control law and produces a control input which is slightly perturbed from the ideal control law $u_k = Kx_k$, which would have been possible if there were state feedback.

If the control law or plant is nonlinear as with MPC, then an analysis of this kind is more difficult. In the case of an MPC controller in closed-loop with an LTI system, the controller can be computed off-line and it results in a piecewise affine control law, as discussed in Section 7.4.2, i.e.

$$u_k = K^i \hat{x}_{k|k} + g^i, \quad \text{if } \hat{x}_{k|k} \in \mathcal{CR}_i \, .$$

The closed-loop system is then also a piecewise affine system

$$\begin{aligned} x_{k+1} &= (A + BK^i)x_k + Bg^i + E^i w_k, \quad \text{for } \hat{x}_{k|k} \in \mathcal{CR}_i \\ y_k &= Cx_k \, , \end{aligned}$$

where the estimation error $e_k = w_k$ is still treated as a state disturbance with $E^i = BK^i$. Given a bound on the error $e_k \in \mathbb{E}$, this analysis can be performed for all critical regions $\mathcal{CR}_i$ and a disturbance set $\mathbb{W}$ computed.

For large systems this kind of analysis might be impractical. Prior to controller design, a $\mathbb{W}$ which is 'sufficiently large' to include the effect of state estimation errors and actual state disturbances can be chosen. Combining this heuristic approach with a robustness constraint already provides one with some kind of robustness guarantee, compared to a standard MPC scheme which has no robust feasibility guarantee.

Finally, one could design the estimator such that the error size is robust to the unmeasured state disturbances [Bla90, Sect. IV]. The estimator parameters have to be chosen such that there exists a robust positively invariant set contained within the *a priori* chosen bounds on e_k. An alternative way of including output feedback in MPC is to incorporate set-based estimation techniques [Sch68, Hny69, Sch73, CGZ96, CGVZ98] into the predictive controller [BG00, CZ00b]. However, the methods proposed in [BG00, CZ00b] propagate the uncertainty in the state forward in time using open-loop predictions. Because of the fact that the predictions are *open-loop*, this approach is conservative and the controller could therefore have a small feasible set. An MPC scheme with a robustness constraint which assumes a bound on the size of the estimation error can be designed to have a larger feasible set, given the same control horizon.

6.8 Setpoint Calculation

In most applications the operating level as required by the operator changes during the lifetime of the process. A controller which has been designed to operate around a single set-point is therefore not very practical. The problem with many MPC schemes is that a large change of set-point could result in an infeasible MPC problem at some future time. The MPC controller has therefore got to be designed to drive the system from one operating point to another without violating the constraints.

Many approaches have been proposed for designing MPC controllers to allow for varying set-points. One of the solutions which has received a large amount of attention is the concept of using a reference governor [GKT95, GK99]. In the standard approach it is assumed that some stabilising controller, which does not explicitly take account of the constraints, has been designed *a priori*. The reference governor then modifies the reference at each time step in order to avoid the violation of constraints. These ideas have been applied in a predictive control context [BCM97, BM98, Bem98], where some form of uncertainty in the impulse/step response can also be assumed.

In [CZ00a] a method is described which combines the reference governor approach with an MPC controller. The region of attraction of the reference governor is enlarged by allowing the controller to not only modify the reference, but the input as well. Bounded state disturbances are dealt with by adding a robustness constraint to the original MPC problem, as in Section 6.4.

An alternative solution to the set-point tracking problem is to derive an MPC controller for a family of set-points [FCA00]. In this approach, a "pseudo-linearisation" of the plant is used to obtain a closed form expression for the MPC controller parameters as a function of the set-point.

More fundamental than taking the system from one set-point to another, is that of determining a set-point which is compatible with the constraints and disturbances [MR93]. The problem of determining a setpoint, assuming no disturbances, is discussed in [Mus97]. A procedure for computing the setpoint for systems with measured disturbances is described in [RR99] and an algorithm which explicitly accounts for model uncertainty is given in [KBH00]. This section discusses the problem of determining a setpoint when there are unknown, but bounded state disturbances.

6.8.1 Computation of a Compatible Setpoint for LTI Systems

Consider the LTI system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Ew_k \\ y_k &= Cx_k + Fv_k\,, \end{aligned} \tag{6.37}$$

where $w_k \in \mathbb{W}$ and $v_k \in \mathbb{V}$ are the disturbances with $(0, 0) \in \mathbb{W} \times \mathbb{V}$. It is required that $u_k \in \mathbb{U}$ and $y_k \in \mathbb{Y}$ for all time. The desired set-points for the inputs and outputs are given by u_d and y_d, respectively.

The problem becomes that of determining a steady-state equilibrium for the state $x_{ss} = Ax_{ss} + Bu_{ss}$ and input u_{ss} such that the steady-state output $y_{ss} = Cx_{ss}$ is close to y_d in some sense and that the constraints can be satisfied for all allowable disturbances[2].

Note that the disturbances are unknown. If the output constraints are to be satisfied for all state and output disturbances, then the state has to be kept inside a robust control invariant set contained inside the output admissible set, i.e.

$$x_k \in \tilde{\mathcal{C}}_\infty^\lambda(\mathbb{X}^\phi), \forall k \in \mathbb{N} \tag{6.38}$$

where the output admissible set $\mathbb{X}^\phi$ is given by

$$\mathbb{X}^\phi = \left\{x_k \in \mathbb{R}^n \mid Cx_k \in \mathbb{Y} \sim F\mathbb{V}\right\}. \tag{6.39}$$

An inner approximation to $\tilde{\mathcal{C}}_\infty^\lambda(\mathbb{X}^\phi)$ can be computed using the algorithms given in Chapter 3. Let Ω denote an inner approximation to the maximal λ-contractive set $\tilde{\mathcal{C}}_\infty^\lambda(\mathbb{X}^\phi)$. The problem can then be restated as finding an x_{ss} and admissible u_{ss} such that the constraints

$$x_{ss} \in \Omega \sim E\mathbb{W} \tag{6.40a}$$

$$x_{ss} = Ax_{ss} + Bu_{ss} \tag{6.40b}$$

are satisfied and y_{ss} and u_{ss} are as close as possible to y_d and u_d.

Remark 6.8. Note that it is required that $x_{ss} \in \Omega \sim E\mathbb{W}$ and not just $x_{ss} \in \Omega$. If only the latter were enforced, then it is possible that at steady-state, a state disturbance could drive the system outside the output admissible set.

The issue is complicated by the fact that the number of inputs and outputs often differ. If there are more inputs than outputs, then multiple combinations of inputs may produce the same output. If there are less inputs than outputs, then it is possible that there does not exist a combination of inputs which will ensure that all desired output values are met. Furthermore, it is often more desirable to satisfy some steady states and give up on others if it is not possible to get an exact solution[3].

If $\Omega \sim E\mathbb{W}$ and $\mathbb{U}$ are given by linear inequalities, then an *ad hoc* way of computing the optimal setpoint is to solve the following soft-constrained quadratic program:

$$\min_{x_{ss}, u_{ss}, \varepsilon} \frac{1}{2}\left[\varepsilon' Q_{ss}\varepsilon + (u_{ss} - u_d)' R_{ss}(u_{ss} - u_d)\right] + q_{ss}'\varepsilon \tag{6.41}$$

[2]Recall from Section 2.2 that the output constraints can be recast as constraints on the state, i.e. $\mathbb{X} = \mathbb{R}^n$ is replaced with $\mathbb{X}^\phi$, the output admissible set.

[3]The setpoint determination then becomes one of a multi-objective optimisation problem [MSB92]. There are several proposals for dealing with multi-objective problems and an approach based on mixed-integer programming is discussed in Chapter 8.

subject to the constraints

$$x_{ss} = Ax_{ss} + Bu_{ss} \tag{6.42a}$$

$$x_{ss} \in \Omega \sim E\mathbb{W} \tag{6.42b}$$

$$u_{ss} \in \mathbb{U} \tag{6.42c}$$

$$y_d - Cx_{ss} \preceq \varepsilon \tag{6.42d}$$

$$y_d - Cx_{ss} \succeq -\varepsilon \tag{6.42e}$$

$$0 \preceq \varepsilon\,, \tag{6.42f}$$

where $Q_{ss} \succ 0$ and $R_{ss} \succ 0$. The weight $q_{ss} \succeq 0$ is chosen sufficiently large such that the soft constraint is guaranteed to be exact[4]. The uniqueness of the solution is guaranteed if the system is detectable [RR99, App. A].

Due to the exact penalty nature of the problem, the optimisation routine tries to minimise the slack variables ε before minimising $u_{ss} - u_d$, thereby assigning a high priority to all the outputs and putting all the inputs on the same, but lower priority level.

Remark 6.9. Provided q_{ss} is large enough, if any of the slack variables of the solution are non-zero, then it indicates that the computed steady state is incompatible with the output constraints and disturbances. Such a violation should be used to indicate a process exception and the operator should be notified. Furthermore, an infeasible solution indicates that a steady state is not possible.

Example 6.3. *Consider the system:*

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} x_k + \begin{bmatrix} 1 & 0.5 \\ 0 & 0.5 \end{bmatrix} u_k + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w_k \\ y_k &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} v_k\,, \end{aligned}$$

with

$$\begin{aligned} \mathbb{Y} &\triangleq \left\{ y \in \mathbb{R}^2 \,|\, \|y\|_\infty \le 5 \right\} \\ \mathbb{U} &\triangleq \left\{ u \in \mathbb{R}^2 \,|\, \|u\|_\infty \le 1 \right\} \\ \mathbb{W} &\triangleq \left\{ w \in \mathbb{R}^2 \,|\, \|w\|_\infty \le 0.5 \right\} \\ \mathbb{V} &\triangleq \left\{ v \in \mathbb{R}^2 \,|\, \|v\|_\infty \le 1 \right\}\,. \end{aligned}$$

The desired output y_d and input u_d values at steady-state are given as

$$y_d = \begin{bmatrix} 2 \\ 4 \end{bmatrix}, u_d = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

[4]See Chapter 7 for a discussion on how to compute such a q_{ss}.

The first step is to calculate the output admissible set. The output admissible set is

$$\begin{aligned}\mathbb{X}^{\phi} &\triangleq \left\{x \in \mathbb{R}^2 \mid Cx \in \mathbb{Y} \sim F\mathbb{V}\right\} \\ &= \left\{x \in \mathbb{R}^2 \mid \|x\|_\infty \leq 4\right\} .\end{aligned}$$

The second step is to calculate the maximal robust control invariant set $\tilde{\mathcal{C}}_\infty(\mathbb{X}^\phi)$ contained in the output admissible set. It turns out that an inner approximation is not necessary, since $\tilde{\mathcal{C}}_\infty(\mathbb{X}^\phi)$ is finitely determined.

The third step is to compute the Pontryagin difference between $\tilde{\mathcal{C}}_\infty(\mathbb{X}^\phi)$ and $E\mathbb{W}$. This set is

$$\Omega \sim E\mathbb{W} = \tilde{\mathcal{C}}_\infty(\mathbb{X}^\phi) \sim E\mathbb{W} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -0.8 & -2 \\ 0.8 & 2 \\ -0.6 & -2 \\ 0.6 & 2 \\ -0.4 & -2 \\ 0.4 & 2 \\ -0.2 & -2 \\ 0.2 & 2 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \preceq \begin{bmatrix} 7.8 \\ 7.8 \\ 7.3 \\ 7.3 \\ 7 \\ 7 \\ 6.9 \\ 6.9 \\ 3.5 \\ 3.5 \\ 3.5 \\ 3.5 \end{bmatrix} \right\} .$$

Finally, solving the soft-constrained QP given in Section 6.8.1 with

$$Q_{ss} = R_{ss} = I, q_{ss} = 100 \cdot [1, 1]' .$$

the steady-state

$$x_{ss} = \begin{bmatrix} 2 \\ 3.05 \end{bmatrix}, u_{ss} = \begin{bmatrix} 0.2 \\ -0.4 \end{bmatrix}$$

is obtained.

Figure 6.3 shows the various sets considered in the computation of the setpoint as well as the location of the final steady state x_{ss} and the state which corresponds to the desired y_d.

Staying Away From the Constraints

In many industrial processes the operating points y_d and u_d of the plant are calculated on a higher level to minimise some economic cost. This optimisation is often posed as an LP and as a result the most economic operating point is always on the boundary or intersection of some of the constraints.

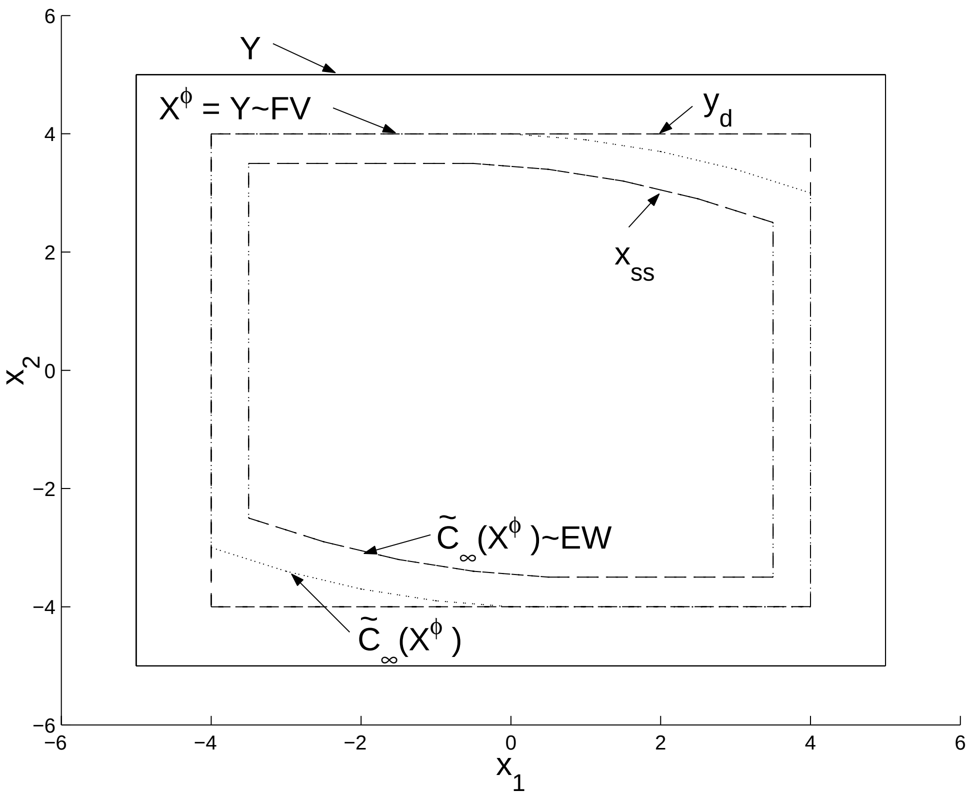

Figure 6.3: The sets used for calculating the setpoint in Example 6.3

On the other hand, one cannot drive the system too close to the boundary since an unknown disturbance could push the system outside the constraints. If the constraint is a safety constraint, then this could result in system failure.

If the $\mathbb{W}$ contains the origin in its interior, then the new steady state will be contained in the interior of the output admissible set $\mathbb{X}^{\phi}$. This agrees with intuition in the sense that in order to satisfy the constraints in the presence of disturbances, the set-point has to be some distance from the boundary.

The soft-constrained optimisation problem posed above will result in a setpoint which is as close to the desired set-point as possible. As can be seen, there will always be some tradeoff between optimality and robustness, since a larger disturbance set will result in a setpoint which is further away from the boundary and hence less optimal in an economic sense.

6.8.2 The MPC Problem With a New Setpoint

Given the new steady-state pair (x_{ss}, u_{ss}) as computed using the above soft-constrained QP, the origin of the system and the input- and state constraints need to be translated and a new MPC problem needs to be set up to regulate the system to the new setpoint.

Problem 6.5 (Robustly feasible MPC with a new setpoint). *Solve*

$$\min_{\pi_k^N} F(\hat{x}_{P|k}) + \sum_{i=0}^{P-1} L(\hat{x}_{i|k}, \hat{u}_{i|k}) \tag{6.43}$$

subject to

$$\begin{aligned}
&\hat{x}_{l+1|k} = A\hat{x}_{l|k} + B\hat{u}_{l|k}, && \hat{x}_{0|k} + x_{ss} = x_k && \text{(6.44a)}\\
&\hat{x}_{1|k} \in \mathbb{X}_R \sim \mathbb{D} && && \text{(6.44b)}\\
&\hat{x}_{l|k} + x_{ss} \in \mathbb{X},\ \hat{u}_{l|k} + u_{ss} \in \mathbb{U}, && l = 0, \dots, P-1 && \text{(6.44c)}\\
&\hat{u}_{l|k} = h(\hat{x}_{l|k}), && l = N, \dots, P-1 && \text{(6.44d)}\\
&\hat{x}_{P|k} \in \mathbb{T}. && && \text{(6.44e)}
\end{aligned}$$

The input that is implemented is given by $u_k = \hat{u}^*_{0|k} + u_{ss}$.

Remark 6.10. The sets $\mathbb{X}_R$ and $\mathbb{T}$ might have to be recomputed for the new setpoint and translated constraints. It is also possible that the horizon lengths need to be increased in order to make the new problem feasible.

If the number of possible operating points are finite, then an off-line design could be carried out to determine all possible values for the constraints and horizons to guarantee feasibility for all cases. If the possible operating points are not known before-hand, then an on-line computation has to be done with each set-point change.

If $\mathbb{T} = \mathbb{X}$, $N = P$ and one would like to keep the current horizon length and robustness constraint, then a steady-state would have to be computed which is compatible with the constraints of the original MPC problem. This is achieved by adding the constraint $x_{ss} \in \mathbb{X}_R \sim \mathbb{D}$ to the steady-state computation of Section 6.8.1.

An important further point which needs mentioning is that "any domain of attraction for a linear constrained system is a tracking domain of attraction" [BM00]. What this implies is that if the set-point of the system changes, the shape and size of the maximal stabilisable and control invariant set does not change and therefore does not need to be recomputed. Though [BM00] discusses only the nominal case, it should be possible to extend the results to the case with disturbances. If this does hold true, then the following remark is also true:

Remark 6.11. If $\mathbb{T} = \mathbb{X}$, $N = P$ and $\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X})$ in the original robust MPC problem, then the new MPC problem will be feasible at the next time instant and it will be robust strongly feasible as well. No new calculations for N or $\mathbb{X}_R$ need to be made. The new x_{ss} is a compatible steady-state only if $x_{ss} \in \tilde{\mathcal{C}}_\infty(\mathbb{X})$.

6.9 Robust MPC Design Examples

This section shows how a robust strongly feasible MPC controller can be designed by adding a robustness constraint to the nominal controller.

6.9.1 The Double Integrator

Consider the double integrator:

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w_k \,.$$

It was shown in Example 6.2 that a nominal MPC controller cannot be designed to be robust strongly feasible for any size of disturbance. It will be shown how adding a robustness constraint guarantees robust strong feasibility of the closed-loop system.

The constraints are given by

$$\begin{aligned} \mathbb{X} &\triangleq \left\{x \in \mathbb{R}^2 \,|\, \|x\|_\infty \leq 5\right\} \\ \mathbb{U} &\triangleq \left\{u \in \mathbb{R}^2 \,|\, \|u\|_\infty \leq 1\right\} \\ \mathbb{W} &\triangleq \left\{w \in \mathbb{R}^2 \,|\, \|w\|_\infty \leq 0.5\right\} \,. \end{aligned}$$

The first step is to design an MPC controller with nominal exponential stability. For this purpose, the stage cost is chosen to be quadratic

$$L(x, u) = x'Qx + u'Ru$$

with

$$Q = I_2, \, R = 1 \,.$$

The terminal controller is chosen to be the solution of the unconstrained, infinite horizon LQR problem with weights Q and R, as in (7.4):

$$h(x_k) = K_\infty x_k = \begin{bmatrix} -0.4345 & -1.0285 \end{bmatrix} x_k \,.$$

The terminal cost is chosen to correspond to be the control Lyapunov function $F(x) = x'Q_F x$ with

$$Q_F = \begin{bmatrix} 2.3671 & 1.1180 \\ 1.1180 & 2.5875 \end{bmatrix} ,$$

where Q_F is found as part of the solution to the Algebraic Riccati Equation (7.4).

The terminal constraint is chosen to be the maximal positively invariant set for the unconstrained LQR controller contained in $\mathbb{X}$:

$$\mathbb{T} = \mathcal{O}^h_\infty(\mathbb{X}) = \mathcal{O}^h_1(\mathbb{X}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ -0.4345 & -1.0285 \\ 0.4345 & 1.0285 \\ 0.1068 & -0.1818 \\ -0.1068 & 0.1818 \end{bmatrix} x \preceq \begin{bmatrix} 5 \\ 5 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\} .$$

The control and prediction horizons are chosen to be equal, i.e. $P = N$. It is desired that the smallest control horizon be chosen such that the feasible set is as large as possible, while still being robust strongly feasible. The largest that the feasible set can be, is equal to $\mathcal{Q}(\tilde{\mathcal{C}}_\infty(\mathbb{X}) \sim \mathbb{D}) \cap \mathbb{X}$.

The maximal robust control invariant set $\tilde{\mathcal{C}}_\infty(\mathbb{X})$ has a determinedness index of 6 and the robustness constraint is chosen to be

$$\mathbb{X}_R \sim \mathbb{D} = \tilde{\mathcal{C}}_\infty(\mathbb{X}) \sim \mathbb{D} .$$

It is found that

$$\mathbb{X}_R \sim \mathbb{D} \not\subseteq \mathcal{S}_0(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, 0, 0)$$

but that

$$\mathbb{X}_R \sim \mathbb{D} \subseteq \mathcal{S}_1(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, 1, 1) .$$

The first statement in Theorem 6.4 implies that if the control horizon $N \geq 2$, then the MPC scheme with the given robustness constraint is robust strongly feasible with feasible set

$$\mathbb{X}^{rc}_F = \mathcal{Q}(\tilde{\mathcal{C}}_\infty(\mathbb{X}) \sim \mathbb{D}) \cap \mathbb{X} .$$

The control horizon is therefore set to $N = P = 2$. Figure 6.4 shows the corresponding sets used in deriving the MPC control law with robustness constraint. Figure 6.5 shows the state evolution starting from a number of initial states inside the feasible set. The MPC problem remains feasible for a sequence of random state disturbances.

It is interesting to note that even though the disturbance does not decay to zero, the system is stable in the sense that every trajectory enters a bounded subset containing the origin.

6.9.2 A System With Three States and Two Inputs

This section illustrates that it is not necessary to visualise the sets in order to design a robust MPC controller. The tools developed in this thesis can be used to obtain values for the MPC parameters such that the feasible set is maximised, while still guaranteeing strong robust feasibility.

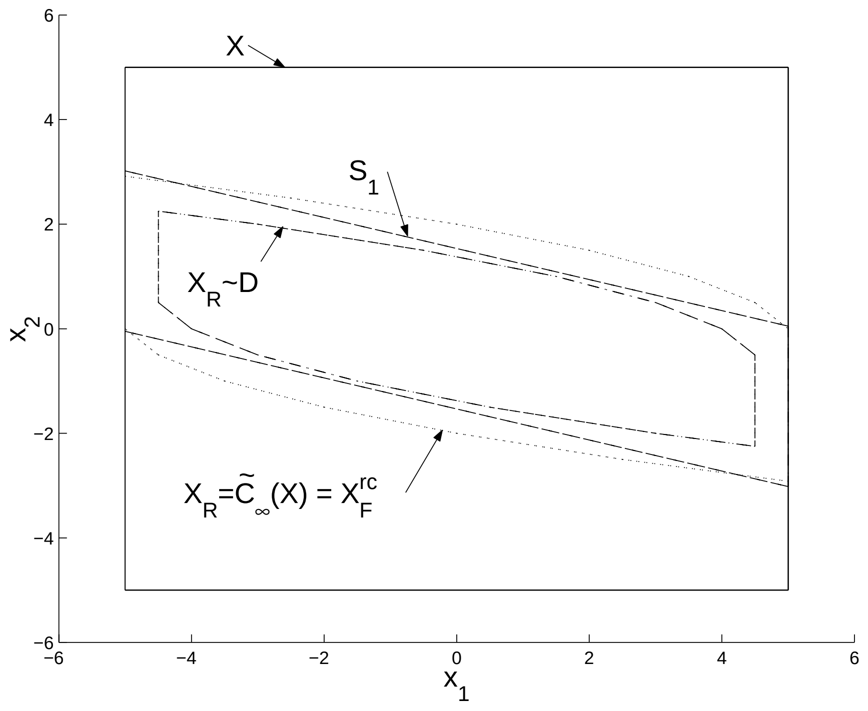

Figure 6.4: Some of the sets used in Section 6.9.1 for designing an MPC controller with a robustness constraint

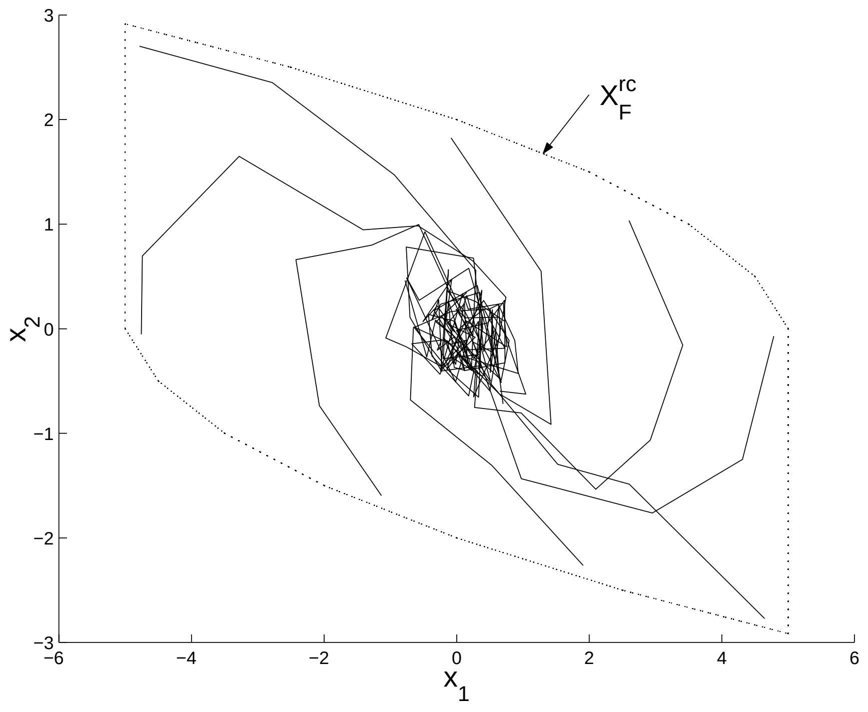

Figure 6.5: The evolution of the system from a number of initial states inside the feasible set of the robust MPC controller designed in Section 6.9.1

Consider the arbitrary system:

$$x_{k+1} = \begin{bmatrix} -0.1 & 0.2 & 0.1 \\ -0.3 & -0.5 & 0.3 \\ 0.5 & -0.6 & 0.7 \end{bmatrix} x_k + \begin{bmatrix} 1 & 2 \\ 5 & 8 \\ 8 & 2 \end{bmatrix} u_k + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} w_k \,,$$

where the constraints are given by

$$\begin{aligned} \mathbb{X} &\triangleq \left\{ x \in \mathbb{R}^3 \,|\, \|x\|_\infty \leq 100 \right\} \\ \mathbb{U} &\triangleq \left\{ u \in \mathbb{R}^2 \,|\, \|u\|_\infty \leq 1 \right\} \\ \mathbb{W} &\triangleq \left\{ w \in \mathbb{R}^3 \,|\, \|w\|_\infty \leq 10 \right\} \,. \end{aligned}$$

The first step is to design an MPC controller with nominal exponential stability. For this purpose, the stage cost is chosen to be quadratic

$$L(x,u) = x'Qx + u'Ru$$

with

$$Q = I_3,\, R = I_2 \,.$$

The terminal constraint is chosen to be

$$\mathbb{T} = \{0\}$$

with the terminal cost

$$F(x) = 0$$

and terminal controller

$$h(x_k) = 0 \,.$$

The maximal robust control invariant set is finitely determined with a determinedness index of 1. By choosing

$$\mathbb{X}_R = \tilde{\mathcal{C}}_\infty(\mathbb{X}) = \tilde{\mathcal{C}}_1(\mathbb{X})$$

the robustness constraint is

$$\mathbb{X}_R \sim \mathbb{D} = \left\{ x \in \mathbb{R}^3 \,\middle|\, \begin{bmatrix} -0.2913 & -0.4854 & 0.2913 \\ 0.5328 & -0.3651 & 0.5920 \\ 0.5000 & -0.6000 & 0.7000 \\ -0.5328 & 0.3651 & -0.5920 \\ 0.2913 & 0.4854 & -0.2913 \\ -0.5000 & 0.6000 & -0.7000 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} x \preceq \begin{bmatrix} 89.3204 \\ 85.1011 \\ 82.0000 \\ 85.1011 \\ 89.3204 \\ 82.0000 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \end{bmatrix} \right\}.$$

By increasing $N = P$ from 1 to 5 it is found that

$$\mathbb{X}_R \sim \mathbb{D} \not\subseteq \mathcal{S}_N(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, N, P), \qquad N = 1, 2, 3, 5$$

but that

$$\mathbb{X}_R \sim \mathbb{D} \subseteq \mathcal{S}_6(\mathbb{X}, \mathbb{T}) = \mathbb{X}_F(\mathbb{T}, 6, 6)\,.$$

The first statement in Theorem 6.4 implies that if the control horizon $N \geq 7$, then the MPC scheme with the given robustness constraint is robust strongly feasible with feasible set

$$\mathbb{X}_F^{rc} = \mathcal{Q}(\tilde{\mathcal{C}}_\infty(\mathbb{X}) \sim \mathbb{D}) \cap \mathbb{X}\,.$$

Figure 6.6 shows the state evolution starting from a number of initial states inside the feasible set. The MPC problem remains feasible for a sequence of random state disturbances.

As with the double integrator, it is interesting to note that even though the disturbance does not decay to zero, the system is stable in the sense that every trajectory enters a bounded subset containing the origin.

6.10 Summary

The notion of strong feasibility defined in the previous chapter was extended to the notion of robust strong feasibility. A necessary and sufficient condition was derived for a given nominal MPC scheme to be robust strongly feasible. This condition reduces to a necessary and sufficient condition on the strong feasibility of the MPC controller in the absence of disturbances.

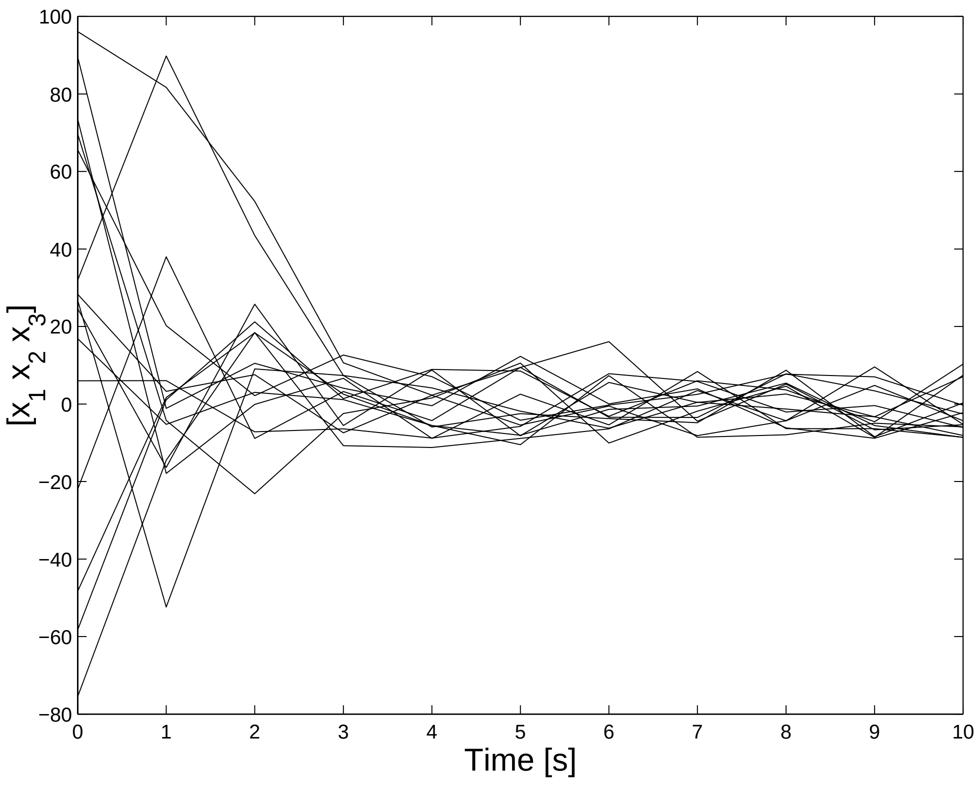

Figure 6.6: The evolution of the system from a number of initial states inside the feasible set of the robust MPC controller designed in Section 6.9.2

The differences between open-loop robust MPC and feedback robust MPC schemes were discussed. These schemes suffer from having to predict all possible future disturbance evolutions at each time step, thereby making on-line implementation very difficult.

The addition of a robustness constraint to the nominal MPC scheme for guaranteeing robust strong feasibility and reducing the computational effort was discussed. The idea relies on the off-line computation of a robust control invariant set. This constraint is used to modify the original MPC scheme by requiring the predicted state at the next time instant to lie inside the Pontryagin difference of this pre-computed set and the disturbance set. A new necessary and sufficient condition and some new sufficient conditions were derived for guaranteeing the robust strong feasibility of the proposed scheme.

It was then shown how this scheme can be applied to guaranteeing robust strong feasibility for MPC of systems with parametric uncertainty and state disturbances. If the constraints are given by convex polyhedra and the cost function is quadratic, then a single QP at each time step is sufficient to compute an MPC control which will guarantee that the MPC problem is feasible at the next time instant, despite the presence of uncertainty and disturbances. This makes the on-line implementation of the robustness constraint MPC approach feasible, since the addition of the extra constraint adds minimal overhead to the computational effort required.

Some well-known conditions for guaranteeing robust stability were also given. It was briefly discussed how the robustness constraint approach can be used to guarantee robust feasibility and stability in the case of output feedback with an asymptotically stable observer.

Finally, some of the ideas from set invariance theory were applied to the computation of a setpoint which is compatible with the constraints of the system, while bearing in mind that there are unknown disturbances on the state and output.

Part III

Recovering from Constraint Violations

Chapter 7

Soft Constraints and Exact Penalty Functions

Soft constraints and exact penalty functions are introduced. It is shown how to compute a lower bound on the penalty weight such that the soft-constrained MPC is such that constraint satisfaction is guaranteed if possible.

7.1 Introduction

The success of Model Predictive Control (MPC) in industry is primarily due to the ease with which constraints on the inputs and states can be included in the control problem formulation. However, sometimes a disturbance drives the plant into a state for which the control problem is infeasible and hence a new control input cannot be computed. Heuristic methods such as removing constraints or repeating the previously computed input are sub-optimal and could lead to unpredictable closed-loop behaviour.

A more systematic method for dealing with infeasibility is to “soften” the constraints by adding slack variables to the problem, where the size of the slack variables correspond to the size of the associated constraint violations [dOB94, SR99, Mac01]. The slack variables are added to the MPC cost function and the optimiser searches for a solution which minimises the original cost function, while keeping the constraint violations as small as possible.

Additionally, it is desirable that the solution to the soft-constrained MPC problem be the same as the solution to the original hard-constrained MPC problem, if the latter were feasible. The theory of exact penalty functions can be used to derive a lower bound for the constraint violation weight such that equality is guaranteed [Fle87, Sect. 14.3]. However, in MPC this weight is dependent on the current state of the system. It is therefore necessary to calculate a lower bound for the whole of the feasible set of the hard-constrained problem.

A naive and impractical solution would be to grid the state space region of interest and compute the optimal Lagrange multipliers at each point. This method is computationally demanding and due to the finite nature of the grid one cannot guarantee that the true lower bound on the weight will be found. As mentioned in [SR96], a conservative state-dependent lower bound might be obtainable by exploiting the Lipschitz continuity of the quadratic program [Hag79]. However, it is unclear as to how exactly one would proceed to implement this for the entire feasible state space.

Furthermore, it is shown in Section 7.6 that the norm of the Lagrange multipliers of the optimal solution are, in general, non-convex over the feasible set. This further complicates the problem.

This chapter shows how the Karush-Kuhn-Tucker (KKT) conditions can be used to compute a lower bound by solving a finite number of linear programs (LPs). This method is therefore computationally less demanding than gridding and provides a guarantee that the lower bound has been found.

Once a lower bound has been computed, the soft-constrained MPC problem can be set up. This new MPC problem will produce a result where the original hard-constrained MPC problem would have been infeasible. The important result is that one can guarantee that the soft- and hard-constrained MPC problems will produce the same result for the region in which the latter would have been feasible.

Section 7.2 defines a standard formulation of MPC with an LTI model subject to linear inequality constraints. It is shown that the cost function and constraints of the resulting quadratic program (QP) are dependent on the current plant state. More precisely, the MPC problem can be treated as a multi-parametric quadratic program (mp-QP) [BMDP00b]. This allows one to gain additional insight into the structure of the problem and develop a systematic approach for computing a lower bound for the violation weight.

Exact penalty functions are introduced in Section 7.3 in order to find a condition on the lower bound for the violation weight. By introducing slack variables the non-smooth[1], exact penalty function can be converted into an easily-solvable, soft-constrained QP.

A procedure for setting up an optimisation routine for computing a non-conservative lower bound for the violation weight is described in Section 7.5. This weight guarantees the exactness of the penalty function over an *a priori* chosen subset of feasible states.

A simple example is presented in Section 7.6 to show how a soft-constrained mp-QP could be set up to have the same solution as the original hard-constrained mp-QP. The chapter concludes with a summary of the results.

7.2 Model Predictive Control of LTI Systems

A standard formulation for MPC will be described below. The cost function and constraints of the optimisation problem will be shown to be dependent on the system state.

[1] In the sense of not being differentiable everywhere.

Consider the following discrete-time, LTI system:

$$x_{k+1} = Ax_k + Bu_k \tag{7.1a}$$

where $x_k \in \mathbb{R}^n$ denotes the state and $u_k \in \mathbb{R}^m$ is the input. The system is subject to linear inequality constraints on the control inputs and/or the states over the whole time horizon $k \in \mathbb{N}$, as in (3.2).

Assuming that a full measurement of the state is available, the MPC problem to be solved at each time step is given by:

Problem 7.1 (Hard-constrained MPC with quadratic cost). *Solve*

$$\mathcal{U}_H^*(x_k) = \arg\min_{\mathcal{U}} \hat{x}'_{P|k} F \hat{x}_{P|k} + \sum_{i=0}^{P-1} \hat{x}'_{i|k} Q \hat{x}_{i|k} + \hat{u}'_{i|k} R \hat{x}_{i|k} \tag{7.2}$$

subject to

$$\begin{aligned}
&\hat{x}_{l+1|k} = A\hat{x}_{l|k} + B\hat{u}_{l|k}, && \hat{x}_{0|k} = x_k && \text{(7.3a)}\\
&\hat{x}_{l|k} \in \mathbb{X}, \quad \hat{u}_{l|k} \in \mathbb{U}, && l = 0, \dots, P-1 && \text{(7.3b)}\\
&\hat{u}_{l|k} = K\hat{x}_{l|k}, && l = N, \dots, P-1 && \text{(7.3c)}\\
&\hat{x}_{P|k} \in \mathbb{T} \subseteq \mathbb{X}, && && \text{(7.3d)}
\end{aligned}$$

where $Q \succeq 0$, $R \succ 0$, $F \succeq 0$ and K is a feedback gain.

The decision variable is the control sequence

$$\mathcal{U} \triangleq \left[\hat{u}'_{0|k}, \hat{u}'_{1|k}, \dots, \hat{u}'_{N-1|k}\right]' .$$

Various possibilities exist in choosing K and F in order to guarantee nominal stability. A popular choice is to set $K = K_\infty$, where K_∞ and F are the solutions of the unconstrained, infinite horizon LQR problem with weights Q and R:

$$K_\infty = -(R + B'FB)^{-1}B'FA \tag{7.4a}$$

$$F = (A + BK_\infty)'F(A + BK_\infty) + K'_\infty RK_\infty + Q\,. \tag{7.4b}$$

The horizon lengths and $\mathbb{T}$ are then chosen such that the feasible set is strongly feasible, as discussed in detail in Chapter 5. It is then straightforward to show via a Lyapunov argument that with this choice of F and K the origin of the nominal closed-loop system will be an exponentially stable fixed point [MRRS00].

In addition to exponential stability, it is also possible to check whether it is necessary to add a robustness constraint to guarantee that the the MPC problem is robust strongly feasible, as discussed in Chapter 6. If the (possibly modified) MPC problem is robust strongly feasible and the additive disturbance decays asymptotically to zero, then the origin of the closed-loop system will be an asymptotically stable fixed point [SRM97].

By substituting

$$\hat{x}_{l|k} = A^l x_k + \sum_{j=0}^{l-1} A^j B \hat{u}_{l-1-j|k} \tag{7.5}$$

into the cost function of Problem 7.1, the optimisation can be rewritten as

$$\mathcal{U}_H^*(x_k) = \arg\min_{\mathcal{U}} \frac{1}{2}\mathcal{U}'\mathcal{H}\mathcal{U} + \mathcal{U}'\mathcal{G}x_k + x_k'\mathcal{F}x_k \tag{7.6a}$$

subject to

$$E\mathcal{U} \preceq f + Gx_k \,. \tag{7.6b}$$

The matrices and vectors $\mathcal{F}$, $\mathcal{G}$, E, f, G and $\mathcal{H} \succ 0$ are obtained by collecting terms. The term involving $\mathcal{F}$ is usually dropped, since it does not affect the optimal solution $\mathcal{U}_H^*(x_k)$.

Remark 7.1. Note that both the cost function and constraints, and hence the optimal solution, are dependent on x_k. The MPC problem can therefore be treated as an mp-QP for which an explicit solution can be computed off-line [BMDP00a, BMDP00b] as will be discussed in Section 7.4. Additionally, it can also be shown that for the reference tracking case, the mp-QP is dependent on the current state, past input and reference [BMDP00a, KM00b]. If a measured disturbance is assumed, then the disturbance also enters as a parameter of the mp-QP.

The feasible set of the hard-constrained mp-QP is defined as in Chapter 5:

$$\mathbb{X}_F \triangleq \{x_k \in \mathbb{R}^n \mid \exists \mathcal{U} : E\mathcal{U} \preceq f + Gx_k\} \,. \tag{7.7}$$

Even if the MPC problem has been designed to be strongly feasible as discussed in Chapter 5, it is still possible that a disturbance or modelling error could result in the system being driven to a state outside $\mathbb{X}_F$, where the hard-constrained mp-QP is infeasible and hence no solution exists. One possible way of dealing with this situation is to soften some or all of the constraints, as described in the sequel.

7.3 Soft Constraints

A straightforward way of softening constraints is to introduce slack variables which are defined such that they are non-zero only if the corresponding constraints are violated. If the original, hard-constrained solution is feasible, one would like the soft-constrained problem to produce the same control action. In order to guarantee this the weights in the cost function have to be chosen large enough such that the optimiser tries to keep the slack variables at zero, if possible. Exact penalty functions can be used to guarantee this behaviour [Fle87, Sect. 14.3].

7.3.1 Exact Penalty Functions

The general non-linear, constrained minimisation problem can be stated as:

$$\theta^* = \arg\min_{\theta} V(\theta) \tag{7.8a}$$

subject to

$$c(\theta) \preceq 0 \ . \tag{7.8b}$$

This optimisation problem can be recast into the following equivalent unconstrained, non-smooth penalty function minimisation:

$$\theta_s^* = \arg\min_{\theta} V(\theta) + \rho \|c(\theta)^+\| \tag{7.9}$$

where the vector $c(\theta)^+$ contains the magnitude of the constraint violations for a given θ and $c_i^+ \triangleq \max(c_i, 0)$. The scalar ρ is the constraint violation penalty weight.

The *dual norm* is used in the condition on ρ which guarantees that the solution θ_s^* to (7.9) is equal to the solution θ^* to (7.8). The dual of a given norm $\|\cdot\|$ is defined as

$$\|u\|_D \triangleq \max_{\|v\| \leq 1} u'v \ . \tag{7.10}$$

It can be shown that the dual of $\|\cdot\|_1$ is $\|\cdot\|_\infty$ and vice versa, and that $\|\cdot\|_2$ is the dual of itself [HJ85].

If θ^* denotes the optimal solution to (7.8) and λ^* is the corresponding Lagrange multiplier vector, then the following well-known result gives a condition under which the solutions to (7.8) and (7.9) are equal:

Theorem 7.1 (Exact penalty function). *If the penalty weight $\rho > \|\lambda^*\|_D$ and $c(\theta_s^*) \preceq 0$, then the solution θ^* to* (7.8) *is equal to the solution θ_s^* to* (7.9).

Proof. See [Fle87, Thm. 14.3.1]. □

If $\rho > \|\lambda^*\|_D$, then (7.9) is called an *exact penalty function*. The cost function (7.9) is non-smooth and therefore not as easy to solve for as, say, a QP. One way to overcome this difficulty is to introduce slack variables into the problem.

7.3.2 Slack Variables as Soft Constraints

The non-smooth, unconstrained minimisation (7.9) can be cast into the following equivalent constrained problem:

$$\min_{(\theta,\epsilon)} V(\theta) + \rho\|\epsilon\| \tag{7.11a}$$

subject to

$$c(\theta) \preceq \epsilon \tag{7.11b}$$

$$0 \preceq \epsilon\,, \tag{7.11c}$$

where ϵ are the slack variables representing the constraint violations, i.e. $\epsilon = 0$ if the constraints are satisfied.

The hard-constrained MPC problem can now be formulated as the soft-constrained MPC problem:

Problem 7.2 (Soft-constrained MPC). *Solve*

$$(\mathcal{U}_S^*(x_k), \epsilon^*(x_k)) = \arg \min_{(\mathcal{U},\epsilon)} \frac{1}{2}\mathcal{U}'\mathcal{H}\mathcal{U} + \mathcal{U}'\mathcal{G}x_k + \rho\|\epsilon\| \tag{7.12a}$$

subject to

$$E\mathcal{U} \preceq f + Gx_k + \epsilon \tag{7.12b}$$

$$0 \preceq \epsilon\,. \tag{7.12c}$$

If $\|\epsilon\|_1$ or $\|\epsilon\|_\infty$ is used in (7.12a) to penalise the constraint violations, then the soft-constrained problem can be formulated as a QP and solved using standard techniques [dOB94, SR99, Mac01].

Remark 7.2. Even though the l_2-norm $\|\epsilon\|_2 \triangleq \sqrt{\epsilon'\epsilon}$ will result in a non-smooth penalty function, one cannot formulate the soft-constrained MPC problem as a QP because the hard-constrained MPC cost function is quadratic and $\|\epsilon\|_2$ has a square root. Using the l_2^2 quadratic norm $\|\epsilon\|_2^2 \triangleq \epsilon'\epsilon$ one can express the problem as a QP, but this does not result in an exact penalty function since (7.9) will be smooth; it is the non-smoothness of the penalty function which allows it to be exact[2].

7.4 Explicit Solution of the MPC Control Law

In MPC, the optimal solution $\mathcal{U}_H^*$ is dependent on the current state x_k, as discussed in Section 7.2, and hence the corresponding Lagrange multiplier λ^* is also dependent on x_k. The lower bound for ρ is therefore dependent on x_k.

One would have to calculate a lower bound for ρ which guarantees that the soft-constrained MPC will produce the same solution as the original hard-constrained MPC for all $x_k \in \mathbb{X}_F$. The Karush-Kuhn-Tucker (KKT) conditions provide some insight into the relation of the Lagrange multipliers to x_k. This section gives an explicit expression for the Lagrange multiplier in terms of x_k, as well as the region in which the expression is valid.

[2]In [SR99], $\|\epsilon\|_S^2$ is added to the cost function, together with a weighted l_1-norm; the l_1-norm guarantees an exact penalty function and S is an extra tuning weight used to penalise the constraint violations.

7.4.1 KKT Conditions for mp-QP Problems

The Lagrangian of optimisation problem (7.6) is

$$\mathscr{L}(\mathcal{U},\lambda,x_k) = \frac{1}{2}\mathcal{U}'\mathcal{H}\mathcal{U} + \mathcal{U}'\mathcal{G}x_k + x_k'\mathcal{F}x_k + \lambda'(E\mathcal{U} - f - Gx_k)\,. \tag{7.13}$$

A stationary point for the Lagrangian occurs when $\nabla_{\mathcal{U}}\mathscr{L}(\mathcal{U},\lambda,x_k) = 0$, hence the corresponding KKT optimality conditions are [Fle87]:

$$\mathcal{H}\mathcal{U} + \mathcal{G}x_k + E'\lambda = 0 \tag{7.14a}$$

$$\lambda \succeq 0, \lambda \in \mathbb{R}^q \tag{7.14b}$$

$$E\mathcal{U} - f - Gx_k \preceq 0 \tag{7.14c}$$

$$\operatorname{diag}(\lambda)(E\mathcal{U} - f - Gx_k) = 0 \tag{7.14d}$$

where q is the number of non-redundant[3] linear inequalities in (7.6b).

Provided $\mathcal{H} \succ 0$ (as is the case when $R \succ 0$), from (7.14a) one can solve for the unique

$$\mathcal{U} = -\mathcal{H}^{-1}(\mathcal{G}x_k + E'\lambda) \tag{7.15}$$

and substitute $\mathcal{U}$ back into (7.14), if desired. For a given x_k, the $\mathcal{U}$ and λ which solve (7.14) are equal to the solution $\mathcal{U}_H^*(x_k)$ and Lagrange multipliers λ^* of (7.6).

7.4.2 Expressions for the Optimal Solution and Lagrange Multipliers

Before proceeding to use the KKT conditions to derive an explicit expression for the optimal solution, the following non-degeneracy assumption is made in order to guarantee that the Lagrange multipliers are unique at the optimum.

Assumption 7.1. For all $x_k \in \mathbb{X}_F$ and for all admissible combinations of active constraints at the optimal solution of (7.6), the corresponding rows of matrix E are linearly independent.

It might be possible to relax this assumption, as can be done for the case of computing the explicit solution of an mp-LP [BBM00c]. However, this assumption seems to be valid in most practical cases.

Theorem 7.2 (Explicit solution of the MPC control law). *[BMDP00a, BMDP00b] Let $\mathcal{H} \succ 0$ and E satisfy Assumption 7.1. For a given x_k, let $\breve{\lambda}(x_k) = 0$ and $\tilde{\lambda}(x_k)$ denote the Lagrange multipliers corresponding to the inactive and active constraints at the optimal solution, respectively. The Lagrange multipliers corresponding to the active constraints are given by*

$$\tilde{\lambda}(x_k) = Sx_k + t \tag{7.16}$$

[3]It is assumed that the non-redundant inequalities are removed from (7.6b) before analysis and implementation.

and the optimal solution[4] *is given by*

$$\mathcal{U}_H^*(x_k) = \left(-\mathcal{H}^{-1}\mathcal{G} - \mathcal{H}^{-1}\tilde{E}'S\right)x_k - \mathcal{H}^{-1}\tilde{E}'t \tag{7.17}$$

where

$$S = -\left(\tilde{E}\mathcal{H}^{-1}\tilde{E}'\right)^{-1}\left(\tilde{G} + \tilde{E}\mathcal{H}^{-1}\mathcal{G}\right) \tag{7.18a}$$

$$t = -\left(\tilde{E}\mathcal{H}^{-1}\tilde{E}'\right)^{-1}\tilde{f} \tag{7.18b}$$

and $\tilde{E}$, $\tilde{f}$ and $\tilde{G}$ correspond to the set of active constraints. Furthermore, these expressions are valid for all x_k contained in the polyhedron

$$\mathcal{CR} = \left\{x_k \in \mathbb{R}^n \left| \begin{bmatrix} -E\mathcal{H}^{-1}\mathcal{G} - E\mathcal{H}^{-1}\tilde{E}'S - G \\ -S \end{bmatrix} x_k \preceq \begin{bmatrix} f + E\mathcal{H}^{-1}\tilde{E}'t \\ t \end{bmatrix} \right.\right\}. \tag{7.19}$$

Proof. Substitute (7.15) into (7.14d) to obtain the complementary slackness condition

$$\operatorname{diag}(\lambda)\left(E\left(-\mathcal{H}^{-1}\left(\mathcal{G}x_k + E'\lambda\right)\right) - f - Gx_k\right) = 0\,.$$

For the inactive constraints

$$\breve{\lambda}(x_k) = 0\,.$$

Let the rows of $\tilde{E}$, $\tilde{f}$ and $\tilde{G}$ correspond to the set of active constraints. For the active constraints $\tilde{\lambda} \succ 0$ and hence (7.14d) implies that

$$\tilde{E}\left(-\mathcal{H}^{-1}\left(\mathcal{G}x_k + \tilde{E}'\tilde{\lambda}\right)\right) - \tilde{f} - \tilde{G}x_k = 0$$

and solving[5] for $\tilde{\lambda}$ it follows that

$$\tilde{\lambda}(x_k) = -\left(\tilde{E}\mathcal{H}^{-1}\tilde{E}'\right)^{-1}\left(\tilde{f} + \left(\tilde{G} + \tilde{E}\mathcal{H}^{-1}\mathcal{G}\right)x_k\right)\,.$$

By defining S and t as in (7.18), the expression $\tilde{\lambda}(x_k) = Sx_k + t$ results.

Substituting this expression for $\tilde{\lambda}(x_k)$ into (7.15) one gets

$$\mathcal{U}_H^*(x_k) = -\mathcal{H}^{-1}\left(\mathcal{G}x_k + \tilde{E}'(Sx_k + t)\right) = \left(-\mathcal{H}^{-1}\mathcal{G} - \mathcal{H}^{-1}\tilde{E}'S\right)x_k - \mathcal{H}^{-1}\tilde{E}'t\,.$$

$\mathcal{U}_H^*(x_k)$ has to satisfy the constraints (7.6b) and the Lagrange multipliers $\tilde{\lambda}(x_k)$ corresponding to the active constraints have to be non-negative. These two constraints combine to define the critical region

$$\begin{aligned}\mathcal{CR} &= \left\{x_k \in \mathbb{R}^n \left| E\left(\left(-\mathcal{H}^{-1}\mathcal{G} - \mathcal{H}^{-1}\tilde{E}'S\right)x_k - \mathcal{H}^{-1}\tilde{E}'t\right) \preceq f + Gx_k,\, Sx_k + t \succeq 0\right.\right\} \\ &= \left\{x_k \in \mathbb{R}^n \left| \begin{bmatrix} -E\mathcal{H}^{-1}\mathcal{G} - E\mathcal{H}^{-1}\tilde{E}'S - G \\ -S \end{bmatrix} x_k \preceq \begin{bmatrix} f + E\mathcal{H}^{-1}\tilde{E}'t \\ t \end{bmatrix} \right.\right\}.\end{aligned}$$

□

[4] Note that the control $u_k = \kappa(x_k)$ to be implemented is given by the first m components of $\mathcal{U}_H^*(x_k)$.

[5] $(\tilde{E}\mathcal{H}^{-1}\tilde{E}')^{-1}$ exists because the rows of $\tilde{E}$ are linearly independent.

This result implies that the resulting MPC control law is a continuous, piecewise-affine function with domain $\mathbb{X}_F$. In order to determine the complete expression over all of $\mathbb{X}_F$ it is necessary to determine all feasible combinations of active constraints. Rather than trying out all $2^q - 1$ combinations of possible active constraints, an efficient procedure can be described as follows:

1. Set $i \leftarrow 0$.
2. Choose an arbitrary $x_k \in \mathbb{X}_F$.
3. Solve the corresponding QP.
4. By looking at the constraints which are active at the solution of this QP, compute the affine functions for $\mathcal{U}_H^*(x_k)$ and $\lambda^*(x_k)$ as in Theorem 7.2.
5. Compute the resulting critical region $\mathcal{CR}_i$ and remove the redundant constraints.
6. Terminate if $\bigcup_{j=0}^{i} \mathcal{CR}_j = \mathbb{X}_F$, else set $i \leftarrow i + 1$ and continue.
7. Choose an arbitrary $x_k \in \mathbb{X}_F \setminus \bigcup_{j=0}^{i-1} \mathcal{CR}_j$ and go to Step 3.

This procedure guarantees that all feasible combinations of active constraints will be computed. The number of feasible combinations is often many orders of magnitude less than $2^q - 1$. A systematic procedure for choosing the x_k in Step 7 is described in [BMDP00a, BMDP00b] and involves the computation of a sensible partitioning of $\mathbb{X}_F$.

7.5 Computing a Lower Bound for the Penalty Weight

The problem of guaranteeing the exactness of the soft-constrained MPC problem can be restated as:

Problem 7.3. *Given $\mathbb{X}_0$, a closed,* bounded[6] *polyhedral subset of the feasible set*

$$\mathbb{X}_0 \subseteq \mathbb{X}_F \,,$$

find a ρ such that

$$x_k \in \mathbb{X}_0 \Rightarrow \mathcal{U}_H^*(x_k) = \mathcal{U}_S^*(x_k)\,.$$

In other words, find a ρ such that

$$\rho > \max_{\mathcal{U}, x_k, \lambda} \|\lambda\|_D \tag{7.20}$$

[6]The requirement that $\mathbb{X}_0$ is bounded, is sufficient to guarantee that the maximisation in (7.20) is bounded from above. To determine whether a given $\mathbb{X}_0$ is contained in $\mathbb{X}_F$, one can test whether the hard-constrained MPC problem is feasible at each one of the vertices of $\mathbb{X}_0$.

with the maximisation subject to the KKT optimality conditions (7.14) and the additional constraint $x_k \in \mathbb{X}_0$. This value for ρ will guarantee that the soft- and hard-constrained QP problems produce the same solution for all feasible $x_k \in \mathbb{X}_0$, since all $\mathcal{U}$ and λ which satisfy the KKT conditions for a given x_k solve the corresponding primal and dual problems.

The optimisation in (7.20) is difficult, since it is the maximisation of the norm of a piecewise affine function, which is not necessarily convex or concave. Furthermore, the number of possible active constraint combinations is exponential in the worst case ($2^q - 1$) and checking each combination of active constraints is therefore impractical.

However, despite this inherent complexity of the optimisation problem, the explicit solution derived in Section 7.4 can be used to develop a systematic procedure for computing a lower bound for ρ:

1. Using the KKT conditions, compute off-line the explicit solution to the mp-QP (7.6):
   (a) Identify, for $\mathbb{X}_0$, all possible combinations of active constraints and the corresponding critical regions $\mathcal{CR}_i$ via the procedure described in [BMDP00a, BMDP00b];
   (b) For each critical region $\mathcal{CR}_i$ that intersects $\mathbb{X}_0$, obtain the explicit affine expression for the Lagrange multipliers corresponding to the set of active constraints:
   $$\tilde{\lambda}(x_k) = S^i x_k + t^i\,, \tag{7.21}$$
   where S^i and t^i are as in Theorem 7.2 and the superscript i denotes the corresponding active region.
2. Choose a lower bound on ρ such that
   $$\rho > \max_i \max_{x_k \in \mathcal{CR}_i} \|\lambda^*(x_k)\|_D = \max_i \max_{x_k \in \mathcal{CR}_i} \|\tilde{\lambda}(x_k) = S^i x_k + t^i\|_D\,. \tag{7.22}$$
   If $\|\cdot\|_1$ or $\|\cdot\|_\infty$ is used to penalise the constraint violations in (7.12a), then the maximum can be found by solving a finite number of LPs for each critical region.

The authors of [BMDP00a, BMDP00b] discuss the computational complexity of computing the explicit solution of the mp-QP and give a bound on the maximum number of possible active constraint combinations. Though it is possible that the computation of the solution could take a long time, for off-line design and analysis the computation speed is less of an issue. The method outlined here is more efficient than the brute force approach of gridding and provides a guarantee that a lower bound has been found.

7.6 Example

Consider the system:

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k \tag{7.23}$$

with constraints on the input

$$\mathbb{U} = \{u \in \mathbb{R} \mid -1 \leq u \leq 1\} \tag{7.24}$$

and the state

$$\mathbb{X} = \left\{ x \in \mathbb{R}^2 \,\middle|\, \begin{bmatrix} -25 \\ -5 \end{bmatrix} \leq x \leq \begin{bmatrix} 25 \\ 5 \end{bmatrix} \right\}. \tag{7.25}$$

The weights for the MPC controller are chosen as

$$Q = I_2, R = 1 \tag{7.26}$$

with the terminal weight

$$F = \begin{bmatrix} 2.3671 & 1.1180 \\ 1.1180 & 2.5875 \end{bmatrix} \tag{7.27}$$

corresponding to the unconstrained, infinite-horizon LQR cost, obtained from solving (7.4). The unconstrained LQR controller is

$$K_\infty = \begin{bmatrix} -0.4345 & -1.0285 \end{bmatrix} \tag{7.28}$$

and the maximal positively invariant set using this controller is

$$\mathcal{O}_\infty^{K_\infty}(\mathbb{X}) = \mathcal{O}_1^{K_\infty}(\mathbb{X}) = \left\{ x \in \mathbb{R}^2 \,\middle|\, \begin{bmatrix} -0.4345 & -1.0285 \\ 0.4345 & 1.0285 \\ 0.1068 & -0.1818 \\ -0.1068 & 0.1818 \end{bmatrix} x \preceq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\}. \tag{7.29}$$

If the terminal set is chosen to be

$$\mathbb{T} = \mathcal{O}_\infty^{K_\infty}(\mathbb{X}), \tag{7.30}$$

then the maximal stabilisable set $\mathcal{S}_\infty(\mathbb{X}, \mathbb{T})$ has a determinedness index of 13:

$$\mathcal{S}_\infty(\mathbb{X}, \mathbb{T}) = \mathcal{S}_{13}(\mathbb{X}, \mathbb{T}) = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 5 \\ -1 & -5 \\ 1 & 4 \\ -1 & -4 \\ 1 & 3 \\ -1 & -3 \\ 1 & 2 \\ -1 & -2 \\ 1 & 1 \\ -1 & -1 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \preceq \begin{bmatrix} 37.5 \\ 37.5 \\ 33 \\ 33 \\ 29.5 \\ 29.5 \\ 27 \\ 27 \\ 25.5 \\ 25.5 \\ 25 \\ 5 \\ 25 \\ 5 \end{bmatrix} \right\}. \tag{7.31}$$

In addition, the maximal stabilisable set is equal to the maximal control invariant set $\mathcal{C}_\infty(\mathbb{X})$, which has a determinedness index of 5:

$$\mathcal{S}_\infty(\mathbb{X}, \mathbb{T}) = \mathcal{C}_\infty(\mathbb{X}) = \mathcal{C}_5(\mathbb{X}). \tag{7.32}$$

The feasible set of an MPC controller with horizon

$$P = N = 13 \tag{7.33}$$

and terminal constraint $\mathbb{T}$ as above is maximal in the sense that

$$\mathbb{X}_F = \mathcal{C}_\infty(\mathbb{X}) = \mathcal{S}_\infty(\mathbb{X}, \mathbb{T}). \tag{7.34}$$

Because of the choice of terminal constraint $\mathbb{T}$ and cost F, the origin will be an exponentially stable fixed point of the closed-loop system, with region of attraction equal to the maximal control invariant set.

As mentioned earlier, the norm of the Lagrange multipliers is not guaranteed to be convex over $\mathbb{X}_F$. Figure 7.1 shows the value of the infinity norm of the Lagrange multipliers for the range

$$x_k = \alpha x_1 + (1-\alpha)x_2, \alpha \in [0, 1], x_1 = [19, -1]', x_2 = [19, -3.6]'.$$

The figure shows that $\|\lambda^*(x_k)\|_\infty$ is non-convex over a small, convex subset of $\mathbb{X}_F$. This implies that problems exist for which the optimisation in (7.20) is inherently complex, thereby ruling out the possibility of using convex optimisation techniques.

Figure 7.2 depicts the feasible set and the critical regions for different combinations of active constraints for the above MPC controller. The states at which the 1-norm and infinity-norm of the La-

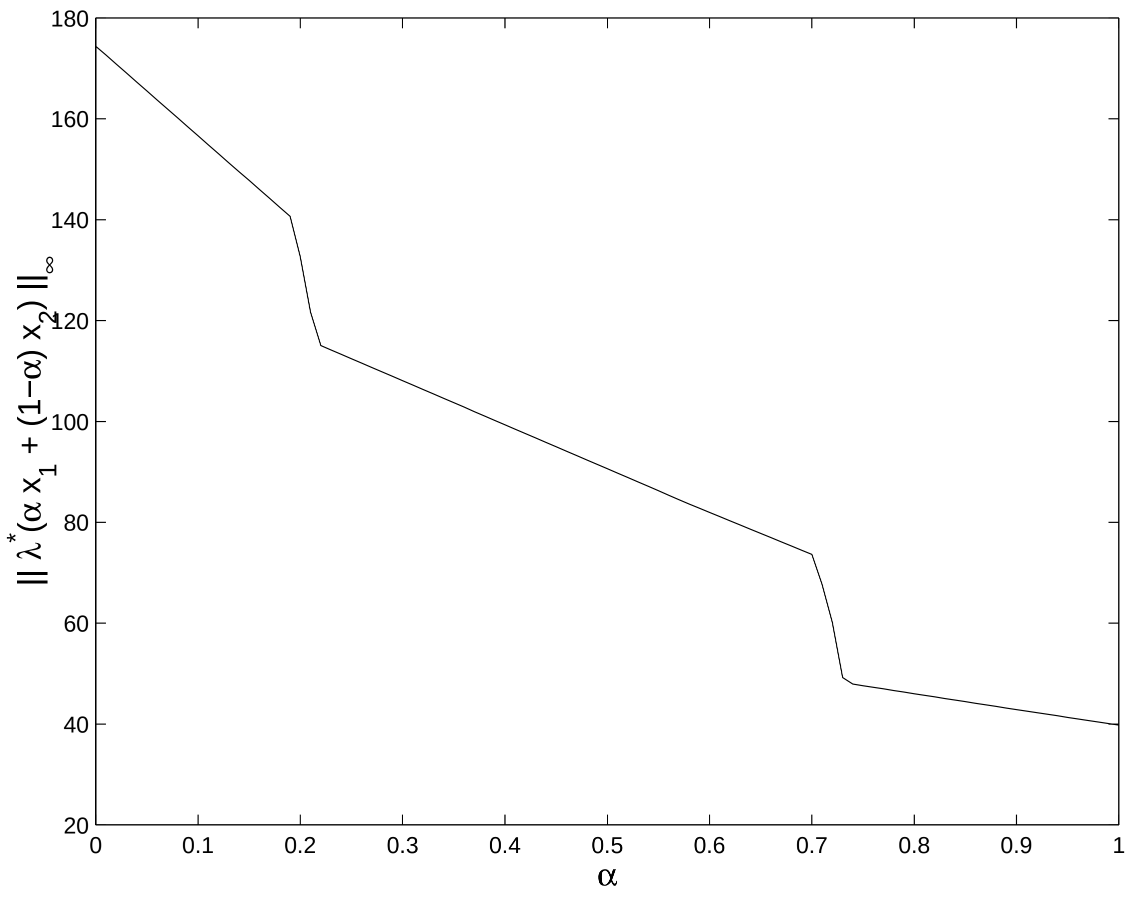

Figure 7.1: Plot showing that $\|\lambda^*(x_k)\|_\infty$ is non-convex over $\mathbb{X}_F$ for the given example. The state x_k is varied from $x_1 = [19, -1]'$ to $x_2 = [19, -3.6]'$, by choosing $x_k = \alpha x_1 + (1 - \alpha)x_2, \alpha \in [0, 1]$

grange multipliers are maximised are

$$\arg\max_{x_k \in \mathbb{X}_F} \|\lambda^*(x_k)\|_\infty = \arg\max_{x_k \in \mathbb{X}_F} \|\lambda^*(x_k)\|_1 = \pm[12.5, 5]' , \tag{7.35}$$

with the maximum norms

$$\max_{x_k \in \mathbb{X}_F} \|\lambda^*(x_k)\|_\infty = 2.188 \times 10^3 \tag{7.36a}$$

$$\max_{x_k \in \mathbb{X}_F} \|\lambda^*(x_k)\|_1 = 8.162 \times 10^3 . \tag{7.36b}$$

This implies that if $\|\epsilon\|_1$ is used in (7.12) to penalise the constraint violations, then

$$\rho > 2.189 \times 10^3, x_k \in \mathbb{X}_F \Rightarrow \mathcal{U}_S^*(x_k) = \mathcal{U}_H^*(x_k) .$$

Similarly, if $\|\epsilon\|_\infty$ is used, then

$$\rho > 8.163 \times 10^3, x_k \in \mathbb{X}_F \Rightarrow \mathcal{U}_S^*(x_k) = \mathcal{U}_H^*(x_k) .$$

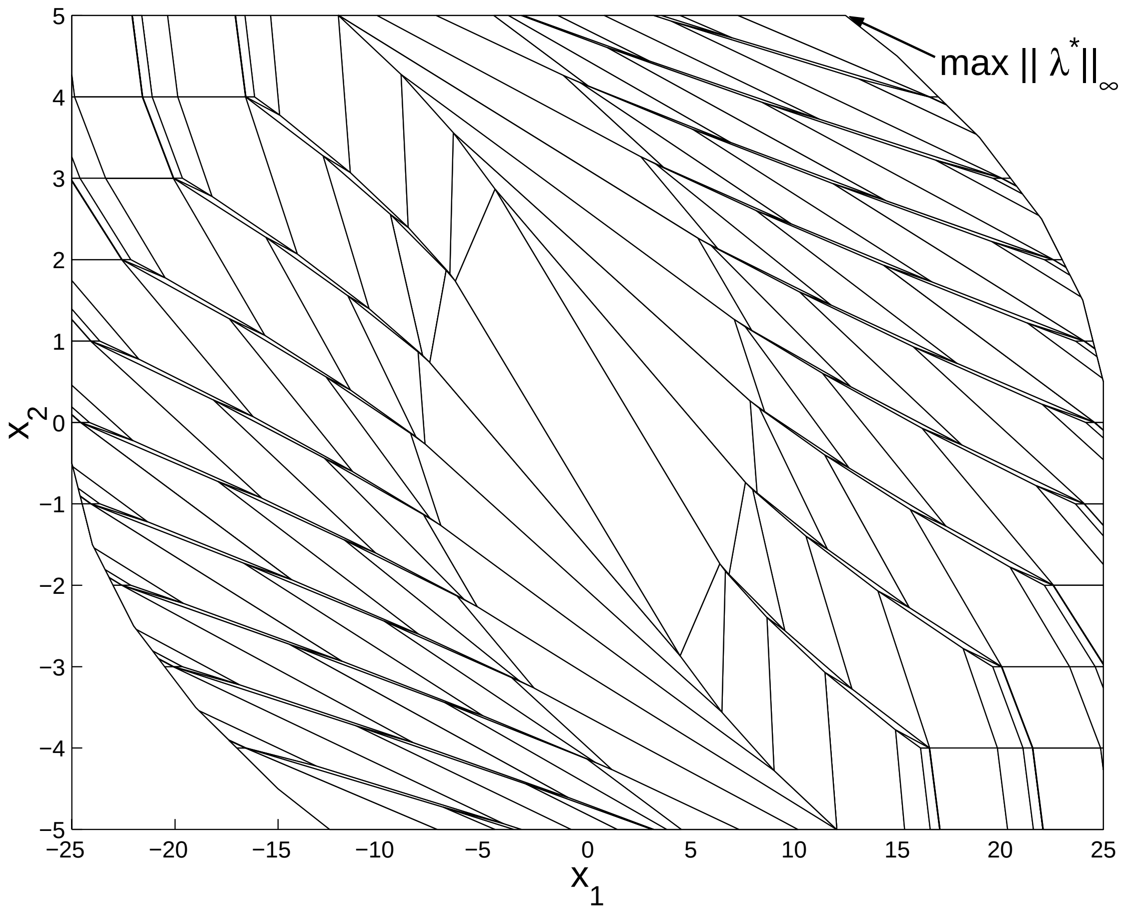

Figure 7.2: The feasible set $\mathbb{X}_F$ and critical regions of the resultant MPC control law with $P = N = 13$ and $\mathbb{T} = \mathcal{O}_{\infty}^{K_\infty}(\mathbb{X})$. The location of the maximising solution to (7.22) is indicated

7.7 Summary

The problem investigated in this chapter is how to choose the weights in a soft-constrained MPC problem such that the resulting control action would be equal to the solution of the original, hard-constrained MPC problem. The theory of exact penalty functions say that if the the constraint violation weight of the soft-constraint problem is larger than the norm of the Lagrange multipliers of the original, hard-constrained problem, then the two solutions will be equal.

A standard formulation of an MPC controller for LTI systems subject to polyhedral constraints was given. It was shown that both the cost function and the constraints of the resulting optimisation problem are dependent on the current state. This implies that the Lagrange multipliers are also dependent on the state. It is therefore necessary to compute an upper bound on the norm of the Lagrange multipliers for all feasible states.

A method for computing the upper bound of the norm of the Lagrange multipliers over a bounded subset of the feasible states was presented. The region of interest can be divided into polytopes in which different combinations of constraints become active at the solution and the Lagrange multipliers are given by an affine expression in the state. The problem of finding the maximum norm of the Lagrange multipliers therefore reduces to solving a finite number of LPs.

If the constraint violation weight that is used in the soft-constrained problem is larger than the computed bound, the solution is guaranteed to be equal to the hard-constrained solution for all feasible conditions that were considered.

Chapter 8

Optimisation Subject to Prioritised Constraints

Multi-objective problems and prioritised solutions are introduced. A mixed-integer approach is described for finding a solution to a constrained optimisation problem which minimises the number of violations in a set of prioritised constraints. The same idea is applied in the computation of a minimum-time, output-prioritised MPC control law for hybrid systems which can be modelled in MLD form.

8.1 Introduction

In most practical applications there is usually a large number of control objectives. The nature of these objectives vary widely from time and frequency domain constraints to the minimisation of a number of cost functions.

The issue is further complicated by the fact that often the objectives cannot be met simultaneously and a solution therefore does not exist. The question then becomes how the objectives should be modified in order for a solution to exist.

The usual approach to attacking an infeasible controller design problem is for the designer to re-specify the objectives and then determine whether a solution to the new problem exists. The choice of which objective to change and how to change it is usually based on the designer's experience and insight into the physical process. This re-specification of the objectives could involve a number of iterations and some systematic method which would reduce the number of iterations is therefore highly desirable.

The area of multi-objective optimisation attempts to provide insight and tools for automating the controller design problem. The need for multi-objective optimisation problems to incorporate the fact that certain objectives are more important than others further complicates the problem. Section 8.2

defines an abstract framework for handling some of these *prioritised*, multi-objective problems.

Finding a general approach to solving a general multi-objective problem is extremely difficult and therefore this chapter deals mainly with the problem of satisfying prioritised constraints, rather than the minimisation of a number of continuous cost functions. Section 8.3 defines a number of related prioritised constraint satisfaction problems and Section 8.4 provides some solutions.

Most of the results in this chapter apply to the general class of multi-parametric, mixed-integer, non-linear programs (mp-MINLPs). Section 8.5 discusses some practical issues for the special case when the problem is an mp-MIQP or mp-MILP, as occurs when implementing MPC controllers for hybrid systems.

While controlling a system, often a disturbance or fault occurs which drives the system outside the maximal control invariant set, thereby making the satisfaction of all the constraints impossible. A control sequence then has to be chosen which will bring the system into the desired region as soon as possible, while bearing in mind that the constraints on output variables have different priorities. An MPC solution to this *minimum-time, output-prioritised* problem is presented in Section 8.6.

One of the motivations for this chapter was to develop a framework for the optimal reconfiguration of a control system in the event of a fault occurring. In Section 8.7 the results of this chapter are applied to the steady-state computation for a faulty three-tank system.

8.2 Prioritised, Multi-Objective Problems

Given a cost function vector $v(\theta) \in \mathbb{R}^r$, where $\theta \in \Theta$ is the decision variable, the multi-objective optimisation problem is often defined as finding the set of all θ^* such that

$$\theta^* = \arg\min_{\theta}[v_1(\theta), v_2(\theta), \dots, v_r(\theta)]\,. \tag{8.1}$$

At this stage it is unclear what is meant by an optimal solution of a cost function vector. When working with multi-objective optimisation problems one therefore needs a definition for optimality. A notion of optimality which is often used is that of Pareto-optimality.

8.2.1 Pareto-Optimal Solutions

Definition 8.1 (Pareto-optimal solution). A solution θ^* is *Pareto-optimal* if and only if $\forall\theta \neq \theta^*$ there exists an i such that $v_i(\theta) > v_i(\theta^*)$ or $v_i(\theta) \geq v_i(\theta^*)$ for all i.

Remark 8.1. This definition is probably easier to understand by noting that a solution θ^* is *not* Pareto-optimal if and only if $\exists\theta \neq \theta^*$ such that $\forall i : v_i(\theta) \leq v_i(\theta^*)$ and $\exists i : v_i(\theta) < v_i(\theta^*)$.

A solution is therefore Pareto-optimal if and only if one cannot find another solution which improves uniformly on all the $v_i(\theta)$. Equivalently, a solution is Pareto-optimal if and only if a decrease in any one of the component cost functions will result in an increase in at least one of the other cost components.

The easiest way of finding an *unprioritised*, Pareto-optimal solution is to solve for

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{r} w_i v_i(\theta)\,, \tag{8.2}$$

where the weights are any $w_i \in \mathbb{R}_+$. By varying the w_i one can generate a set of Pareto-optimal solutions.

However, this chapter is concerned with finding the subset of Pareto-optimal solutions which are optimal with respect to the relative *priorities* of all the cost functions $v_i(\theta)$.

8.2.2 Prioritised-Optimal Solutions

Before giving a definition of a prioritised-optimal solution, the following assumption is made:

Assumption 8.1. The objective associated with cost function $v_i(\theta)$ has a higher priority than the one associated with $v_{i+1}(\theta)$.

A formal definition of *priority* will not be given. However, the following implicitly defines what is meant by priority.

Definition 8.2 (Prioritised-optimal solution). A solution θ^* is a *prioritised-optimal* solution if and only if $\nexists \theta \neq \theta^*$ such that $v_{i^*}(\theta) < v_{i^*}(\theta^*)$, where i^* is the index of the first element where $v(\theta)$ and $v(\theta^*)$ differ.

The process of finding the set of prioritised-optimal solutions can be described as follows: A subset $\Theta_1 \subseteq \Theta$ is chosen for which all $\theta \in \Theta_1$ are such that $v_1(\theta)$ is minimised. The subset $\Theta_2 \subseteq \Theta_1$ is then chosen such that $\forall \theta \in \Theta_2$, $v_2(\theta)$ is minimised. This process is continued until all $v_i(\theta)$ have been minimised[1]. Determining the prioritised-optimal solution is equivalent to finding the lexicographic minimum[2] of a set [VSJ99, Def. 1].

A single *prioritised*, Pareto-optimal solution is therefore obtained by solving the sequence of optimisation problems for $i = 1, \dots, r$:

$$v_i^* = \min_{\theta} v_i(\theta) \tag{8.3}$$

subject to the set of constraints

$$v_j(\theta) = v_j^*, \quad j = 1, \dots, i-1\,. \tag{8.4}$$

[1]An implemented algorithm will not necessarily follow this recipe, but the result would be the same.

[2]This process is analogous to arranging a set of words alphabetically, hence the use of the word ‘lexicographic’. For example, the lexicographic minimum of the set $\{[2,3,1], [3,2,1], [2,2,4], [2,2,1], [2,2,3]\}$ is $[2,2,1]$.

The solution to the r'th optimisation problem is a prioritised-optimal solution. This approach is also the method that is used in [MSB92, VSF99]. Though easy to implement, this method will always require r optimisation problems to be solved.

It would therefore be desirable if one could find a set of weights for (8.2) such that one could guarantee that the solution to (8.2) is a prioritised, Pareto-optimal solution to (8.1), as was done in [VSJ99] for the special case of a prioritised LP. It turns out that this is relatively easy if the cost function is such that $\forall\theta,\ v(\theta) \in \mathbb{N}^r$. A choice of weights which guarantees a prioritised-optimal solution is given by the following theorem.

Theorem 8.1 (Weights for the prioritised, multi-objective problem).
Let $v_i(\theta) \in \mathbb{N}$ and $v_i(\theta) \leq t_i, \forall\theta \in \Theta$. If

$$\theta^* = \arg\min_{\theta\in\Theta} W' v(\theta)\,, \tag{8.5a}$$

where

$$W \triangleq \begin{bmatrix} w_1 \\ \vdots \\ w_i \\ \vdots \\ w_r \end{bmatrix} \tag{8.5b}$$

with $w_i \in \mathbb{R}_+$ and

$$w_i > \sum_{j=i+1}^{r} t_j w_j\,, \tag{8.5c}$$

then θ^* is a prioritised, Pareto-optimal solution to (8.1).

Proof. Assume that θ^* is an optimal solution to (8.5a), but that it is not prioritised-optimal, i.e. there exists a $\theta \neq \theta^*$ such that $v_{i^*}(\theta) < v_{i^*}(\theta^*)$, where i^* is the index of the first element where $v(\theta)$ and

$v(\theta^*)$ differ, i.e. $v_{i^*}(\theta) \leq v_{i^*}(\theta^*) - 1$. If this is the case, then

$$\begin{aligned}
W'v(\theta) - W'v(\theta^*) &= \sum_{j=1}^{r} w_j(v_j(\theta) - v_j(\theta^*)) \\
&= \sum_{j=i^*}^{r} w_j(v_j(\theta) - v_j(\theta^*)), \text{ since } v_j(\theta) = v_j(\theta^*), \, j = 1, \dots, i^* - 1 \\
&= w_{i^*}(v_{i^*}(\theta) - v_{i^*}(\theta^*)) + \sum_{j=i^*+1}^{r} w_j(v_j(\theta) - v_j(\theta^*)) \\
&\leq w_{i^*}(v_{i^*}(\theta) - v_{i^*}(\theta^*)) + \sum_{j=i^*+1}^{r} w_j t_j, \text{ since } v_j(\theta) - v_j(\theta^*) \leq t_j \\
&< w_{i^*}(v_{i^*}(\theta) - v_{i^*}(\theta^*)) + w_{i^*} \\
&= w_{i^*}(v_{i^*}(\theta) - v_{i^*}(\theta^*) + 1) \\
&\leq 0, \text{ since } w_{i^*} > 0, \, v_{i^*}(\theta) - v_{i^*}(\theta^*) \leq -1 \, .
\end{aligned}$$

This implies that $W'v(\theta) < W'v(\theta^*)$. This contradicts the assumption that θ^* is optimal, thereby concluding the proof. □

This idea that the weight for a certain priority level must be larger than the weighted sum of the number of lower-prioritised objectives, will be used frequently in the subsequent sections.

8.2.3 Constraint Satisfaction

It might seem that by restricting the cost functions to bounded $v_i(\theta) \in \mathbb{N}$ very few multi-objective problems will fall into this class. However, note that the satisfaction of a constraint can be represented as the minimisation of a cost function, e.g. if the constraint $g_i(\theta) \leq 0$ is given and one defines

$$v_i(\theta) \triangleq \begin{cases} 0 & \text{if } g_i(\theta) \leq 0 \\ 1 & \text{if } g_i(\theta) > 0 \end{cases} \tag{8.6}$$

then $v_i(\theta) = 1$ if the constraint is violated and $v_i(\theta) = 0$ if it is satisfied. For a more complex example, assume that the objectives consist only of constraints and that there are r priority levels, with the possibility of some constraints having the same priority. Given a candidate solution θ, one can define $v_i(\theta) \in \mathbb{N}$ to denote the number of violated constraints on priority level i, hence $v(\theta) \in \mathbb{N}^r$ represents the number of violated constraints on each of the priority levels. Hence, a solution is prioritised-optimal if and only if there does not exist another solution which will violate less constraints on any level, without increasing the number of violated constraints on a higher level.

Theorem 8.1 therefore allows one to define multi-objective problems in terms of the *number* of constraint satisfactions, violations or relaxations. The problem of designing a single optimisation which minimises the number[3] of prioritised constraint violations seems to have received very little attention

[3]In [VSJ99] the problem of minimising the *size* of the constraint violations in a prioritised fashion is considered.

in the optimisation and control literature. The subsequent sections present a method for solving this and related problems.

Though Theorem 8.1 is the solution to an abstract problem, it will be shown how one can modify a multi-parametric, soft-constrained optimisation problem so that the solution is such that the number of constraint violations is minimised in a prioritised-optimal fashion. This is achieved by introducing logic variables into the problem such that the value of the logic variable at the solution indicates which constraints have been satisfied or violated.

Remark 8.2. Note that in this context the usual concept of a constrained optimisation problem can be interpreted as a prioritised multi-objective optimisation problem, with the satisfaction of the constraints taking higher priority than the optimisation of the cost function. As a result, in subsequent sections only the prioritised satisfaction of constraints will be considered. The optimisation problem will be constructed such that minimisation of the cost function effectively has the lowest priority. The cost function will only be minimised after a set of solutions has been found that guarantees constraint satisfaction.

8.2.4 Numerical Conditioning of the Proposed Choice of Weights

Though Theorem 8.1 is a simple result, it has a drawback in the sense that the weights can grow to be very large if there are a large number of priority levels, as shown in the next example.

Example 8.1. *Let a problem contain 100 objectives and choose* $w_r = 1$.

- *Let each of the objectives be assigned its own priority, i.e.* $t_i = 1$. *If one chooses* $w_i = 1 + \sum_{j=i+1}^{r} t_j w_j$, *then* $w_1 \approx 6.338 \times 10^{29}$. *If* $w_i = 0.001 + \sum_{j=i+1}^{r} t_j w_j$, *then* $w_1 \approx 3.172 * 10^{29}$, *which is not much better.*
- *If the objectives can be divided into 10 different priorities, i.e.* $t_i = 10$, *and one chooses* $w_i = 1 + \sum_{j=i+1}^{r} t_j w_j$, *then* $w_1 = 2.357947691 \times 10^9$. *If* $w_i = 0.001 + \sum_{j=i+1}^{r} t_j w_j$, *then* $w_1 \approx 2.144 \times 10^9$. *These two choices of weights are slightly more acceptable.*
- *Let the objectives be such that the first 10 have the same, but higher priority than the next 90, i.e.* $t_1 = 10$ *and* $t_2 = 90$. *If one chooses* $w_i = 0.001 + \sum_{j=i+1}^{r} t_j w_j$, *then* $w_i = 90.001$ *for* $i = 1 \ldots 10$ *and* $w_i = 1$ *for* $i = 11 \ldots 100$. *This choice of weights is a lot more preferable than for the previous two cases, where the choice of weights could result in an ill-conditioned problem.*

One can therefore conclude that a large problem with many priority levels might result in an ill-conditioned optimisation problem. The proposed choice of weights therefore works best either when there are a small number of priority levels or a small number of objectives with a high priority. This is typically the case in many practical applications where there are a large number of lower-prioritised performance constraints and a small number of higher-prioritised safety constraints.

8.3 Problem Formulation

The main aim of this chapter is to design a prioritised, soft-constrained problem for a given hard-constrained, multi-parametric, mixed-integer, nonlinear program (mp-MINLP). A *multi-parametric* optimisation problem is one where the cost function and/or constraints are dependent on one or more variable. A different optimal solution set will exist for each one of these variables. The motivation here for working with multi-parametric programs is due to the fact that the optimisation problem in MPC controller design is dependent on the current state and hence the solution is also dependent on the current state. The current state therefore parametrises the solution.

This section is concerned with defining the scope of objectives and problems which this chapter attempts to solve. The hard-constrained problem and prioritisation scheme is described. This is followed with the setting up of the prioritised, soft-constrained problem and the types of problems which will be addressed in the next section.

Consider the following hard-constrained mp-MINLP:

Problem 8.1 (Hard-constrained mp-MINLP). *Solve*

$$\theta^*(x) = \arg\min_{\theta} f(\theta, x) \tag{8.7a}$$

subject to

$$g(\theta, x) \preceq 0 \tag{8.7b}$$

where $\theta \in \mathbb{R}^{d_1} \times \mathbb{Z}^{d_2}$ *is the decision variable and* $x \in \mathbb{R}^{p_1} \times \mathbb{Z}^{p_2}$ *is the parameter vector of the mp-MINLP and* $f : (\mathbb{R}^{d_1} \times \mathbb{Z}^{d_2}) \times (\mathbb{R}^{p_1} \times \mathbb{Z}^{p_2}) \mapsto \mathbb{R}$. *The constraints* (8.7b)*, where* $g : (\mathbb{R}^{d_1} \times \mathbb{Z}^{d_2}) \times (\mathbb{R}^{p_1} \times \mathbb{Z}^{p_2}) \mapsto \mathbb{R}^c$*, define a closed and bounded, non-empty set* $\mathcal{F} \triangleq \{(\theta, x) : g(\theta, x) \preceq 0\} \neq \emptyset$ *and all the constraints are unique, i.e.* $g_i(\cdot, \cdot) = g_j(\cdot, \cdot) \Leftrightarrow i = j$.

The constraints implicitly define the set of feasible parameters for the hard-constrained mp-MINLP:

$$\mathbb{X}_{FH} \triangleq \{x : \exists\theta \text{ such that } g(\theta, x) \preceq 0\} . \tag{8.8}$$

It is assumed that both $\min_\theta f(\theta, x)$ and $\max_\theta f(\theta, x)$ exist $\forall x \in \mathbb{X}_{FH}$. No continuity assumptions are made.

The constraints (8.7b) usually reflect desired constraints which the decision variable has to satisfy. However, sometimes a parameter x is passed to the optimisation routine for which no feasible solution exists, i.e. $x \notin \mathbb{X}_{FH}$. It is therefore necessary to either redefine the problem or, more likely, relax some of the constraints and allow for the violation of some of the constraints in the final solution.

8.3.1 Prioritised Constraints

Often a hierarchy of priorities can be assigned to the set of constraints, e.g. it is more important to satisfy the constraint $g_1(\theta, x) \leq 0$ than the constraint $g_2(\theta, x) \leq 0$. A solution which violates

$g_2(\theta, x) \leq 0$ but satisfies $g_1(\theta, x) \leq 0$ is therefore preferred. For the purpose of rigorously defining and implementing these priorities, the following definitions are given:

- The set of indices of the constraints is given by $\mathcal{C} \triangleq \{1, 2, \dots, c\}$. If the set of indices of the soft constraints is given by $\mathcal{S}$, then the set of indices of the hard constraints is $\mathcal{H} \triangleq \mathcal{C} \backslash \mathcal{S}$;

- There are r priority levels, ordered such that level i has a higher priority than level $i + 1$. The set of indices of constraints on priority level i is given by $\mathcal{P}_i \subseteq \mathcal{C}$, with $\mathcal{P}_i \cap \mathcal{P}_j = \emptyset, i \neq j$, i.e. a constraint cannot be associated with more than one priority level.

  Let c_i be the number of constraints associated with priority level i, i.e. $c = \sum_{i=1}^{r} c_i$;

- The vector of slack variables $\epsilon \in \mathbb{R}^s$ is defined as $\epsilon_m(\theta, x) \triangleq \max_{k \in \mathcal{S}_m}(g_k(\theta, x), 0)$, where $\mathcal{S}_m$ is the set of indices of soft constraints associated with slack variable ϵ_m. $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset, i \neq j$ and $\mathcal{S} = \bigcup_{m=1}^{s} \mathcal{S}_m$.

  For a given (θ, x), each slack variable represents the largest constraint violation of a set of constraints, hence $[\epsilon_m(\theta, x) = 0] \leftrightarrow \bigwedge_{k \in \mathcal{S}_m}[g_k(\theta, x) \leq 0]$.

  All constraints associated with a slack variable have to be associated with the same priority level, i.e. $\forall \mathcal{S}_m$, $\exists \mathcal{P}_i$ such that $\mathcal{S}_m \subseteq \mathcal{P}_i$.

  Let s_i be the number of slack variables associated with priority level i, i.e. $s = \sum_{i=1}^{r} s_i$;

- Each element of the vector of logic variables $\delta \in \{0, 1\}^t$ is associated with one or more slack variables on the same priority level such that

  $$[\delta_n = 0] \leftrightarrow \bigwedge_{m \in \mathcal{T}_n} [\epsilon_m = 0], \tag{8.9}$$

  where $\mathcal{T}_n$ is the set of indices of the *slack variables* associated with δ_n and $\mathcal{T}_i \cap \mathcal{T}_j = \emptyset, i \neq j$, i.e. a set of slack variables (and the associated set of soft constraints) cannot be associated with more than one logic variable.

  The set of indices of *constraints* associated with δ_n is given by $\mathcal{D}_n \triangleq \bigcup_{m \in \mathcal{T}_n} \mathcal{S}_m$ and $\forall \mathcal{D}_n$, $\exists \mathcal{P}_i$ such that $\mathcal{D}_n \subseteq \mathcal{P}_i$, hence

  $$[\delta_n = 0] \leftrightarrow \bigwedge_{k \in \mathcal{D}_n} [g_k(\theta, x) \leq 0]. \tag{8.10}$$

  Let t_i be the number of logic variables associated with priority level i, i.e. $t = \sum_{i=1}^{r} t_i$.

Remark 8.3. From the definitions above, it can be seen that $c \geq s \geq t \geq r$.

Example 8.2. *A given problem has $c = 10$ constraints:*

$$\mathcal{C} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\},$$

where the first eight are allowed to be softened and the rest are to remain as hard constraints:

$$\begin{aligned}\mathcal{S} &= \{1, 2, 3, 4, 5, 6, 7, 8\} \\ \mathcal{H} &= \{9, 10\}\,.\end{aligned}$$

The soft constraints can be ordered in a hierarchy of $r = 4$ *priority levels, from highest to lowest:*

$$\mathcal{P}_1 = \{1, 2\}, \mathcal{P}_2 = \{3, 4\}, \mathcal{P}_3 = \{5, 6, 7\}, \mathcal{P}_4 = \{8\}\,.$$

Each constraint is associated with its own slack variable, except constraints 5 and 6 (hence $s = 7$*), since they cannot be violated simultaneously and introducing an additional slack variable is therefore unnecessary:*

$$\mathcal{S}_1 = \{1\}, \mathcal{S}_2 = \{2\}, \mathcal{S}_3 = \{3\}, \mathcal{S}_4 = \{4\}, \mathcal{S}_5 = \{5, 6\}, \mathcal{S}_6 = \{7\}, \mathcal{S}_7 = \{8\}\,.$$

A logic variable is associated with each slack variable, except slack variables 3 and 4, which share a logic variable:

$$\mathcal{T}_1 = \{1\}, \mathcal{T}_2 = \{2\}, \mathcal{T}_3 = \{3, 4\}, \mathcal{T}_4 = \{5\}, \mathcal{T}_5 = \{6\}, \mathcal{T}_6 = \{7\}\,.$$

Hence, the indices of the constraints associated with the $t = 6$ *logic variables are:*

$$\mathcal{D}_1 = \{1\}, \mathcal{D}_2 = \{2\}, \mathcal{D}_3 = \{3, 4\}, \mathcal{D}_4 = \{5, 6\}, \mathcal{D}_5 = \{7\}, \mathcal{D}_6 = \{8\}\,.$$

It can be seen that each $\mathcal{D}_n$ *and* $\mathcal{S}_m$ *is a subset of some* $\mathcal{P}_i$*. There are two logic variables and two slack variables associated with priority level 1, one logic variable and two slack variables with priority level 2, two logic variables and two slack variables (but 3 constraints) with priority level 3 and, finally, one logic variable and one slack variable with priority level 4.*

Note that if a solution has been found and $\delta_3 = 1$*, then one cannot deduce whether only one or both of constraints 3 and 4 are violated - one would have to look at the values of the associated slack variables. If* $\delta_4 = 1$*, then either constraint 5 or 6 is violated, but not both. However, if it were possible for constraints 5 and 6 to be violated simultaneously, then one also cannot tell whether one or both constraints have been violated. Since they both share the same slack variable, one cannot gain any information from examining it.*

These definitions have been given for the sake of rigour. It will become clear later on how they are used to set up prioritised, soft-constrained problems.

8.3.2 Setting up a Soft-Constrained Problem

The slack variables are used to soften the constraints. However, the following assumption is made in order to guarantee that the scalar ρ, which is used later in (8.16a), has a finite lower bound.

Assumption 8.2. All the slack variables $\epsilon(\theta, x)$ are bounded above:

$$0 \le \epsilon_m(\theta, x) \le M_m \, . \tag{8.11}$$

This is a realistic assumption, since in most applications the constraints are associated with a physical parameter which is bounded. One can now proceed to define the soft-constrained problem.

Problem 8.2 (Soft-constrained mp-MINLP). *Solve*

$$(\theta_s^*(x), \epsilon^*(x), \delta^*(x)) = \arg\min_{\theta,\epsilon,\delta} \tilde{f}(\theta, x, \epsilon) + \rho W'\delta \tag{8.12}$$

subject to the inequalities

$$g_k(\theta, x) \le \epsilon_m \tag{8.13a}$$
$$g_l(\theta, x) \le 0 \tag{8.13b}$$
$$0 \le \epsilon_m \le M_m \delta_n \tag{8.13c}$$
$$k \in \mathcal{S}_m \tag{8.13d}$$
$$l \in \mathcal{H} \tag{8.13e}$$
$$m \in \mathcal{T}_n \tag{8.13f}$$
$$n \in \{1, 2, \dots, t\} \tag{8.13g}$$

where it is desired that the logic variable $\delta^*(x) \in \{0,1\}^t$ indicates whether any of the associated constraints have been violated in the sense of (8.10) *and $\epsilon_m^*(x) = \epsilon_m(\theta_s^*(x), x)$ represents the largest violation in the m'th subset of constraints. The weights $\rho \in \mathbb{R}_+$ and $W \in \mathbb{N}_+^t$ should be chosen to minimise constraint violations in $g(\theta_s^*(x), x)$, while satisfying the given priorities.*

Since $\mathcal{F}$ is bounded and the slack variables are bounded, the feasible set of parameters x for Problem 8.2, denoted by $\mathbb{X}_{FS}$, is also bounded ($\mathbb{X}_{FS} \supseteq \mathbb{X}_{FH}$). One has to choose a compact subset of states $\mathbb{X}_0 \subseteq \mathbb{X}_{FS}$, with $\mathbb{X}_0 \backslash \mathbb{X}_{FH} \neq \emptyset$, for which one would like to design a prioritised, soft-constrained optimisation problem.

Obviously, a good choice would be $\mathbb{X}_0 = \mathbb{X}_{FS}$ or $\mathbb{X}_0 \supset \mathbb{X}_{FH}$. However, for computation of $\rho \in \mathbb{R}_+$ this might not always be practical and some trade-off in the size of $\mathbb{X}_0$ has to be made. The role of ρ will become clearer in subsequent sections.

Before proceeding, since the original definition of the slack variable $\epsilon_m(\theta, x) \triangleq \max_{k \in \mathcal{S}_m}(g_k(\theta, x), 0)$ results in a non-smooth optimisation problem, one would like to pose it as an easier problem.

Lemma 8.1. *If $\|\cdot\|$ is a suitably defined norm, then the size of the largest violation in the m'th subset of constraints is given by*

$$\epsilon_m(\theta, x) = \arg\min_{\alpha \in \mathbb{R}} \|\alpha\| \, , \tag{8.14a}$$

where the optimisation is subject to

$$g_k(\theta, x) \le \alpha, \quad \forall k \in \mathcal{S}_m \tag{8.14b}$$

$$0 \le \alpha\,. \tag{8.14c}$$

Using this result, one can choose the cost function of the soft-constrained problem such that the components of the optimal ϵ are equal to the sizes of the constraint violations.

Lemma 8.2. *Let*

$$\tilde{f}(\theta, x, \epsilon) = f(\theta, x) + \sum_{m=1}^{s} \|\epsilon_m\|\,, \tag{8.15a}$$

where $\|\cdot\|$ is any suitably defined norm. If the optimal solution to Problem 8.2 has been found, then the vector $\delta^*(x)$ reflects which sets of constraints have been satisfied or violated in the sense that

$$\left[\delta_n^*(x) = 0\right] \Leftrightarrow \bigwedge_{k \in \mathcal{D}_n} \left[g_k(\theta_s^*(x), x) \le 0\right]\,. \tag{8.15b}$$

Furthermore, $\epsilon_m^*(x)$ represents the size of the largest violation in the m'th subset of constraints.

Proof. If (8.15b) does not hold, then the only other possibility is that the constraints are satisfied with the associated logic variable equal to 1. This contradicts the optimality assumption, since it is possible to set the associated slack vectors and logic variables to 0 for all satisfied sets of constraints, resulting in a lower cost. If any of the constraints associated with a logic variable are not satisfied, then the logic variable has to be equal to 1.

For the given $\theta^*(x)$, by application of Lemma 8.1 and the optimality assumption, it can be seen that $\epsilon_m^*(x) = \epsilon_m(\theta^*(x), x)$ represents the size of the largest violation in the m'th subset of constraints. $\square$

8.3.3 Prioritised-Optimal Soft-Constrained Problems

The following problems will be considered in the sequel. The first problem is the same as the one defined in Chapter 7, of guaranteeing that the hard- and soft-constrained solutions will be equal for a given subset of $\mathbb{X}_{FH}$.

Problem 8.3 (Exact penalty function). *Set up Problem 8.2 such that $\forall x \in \mathbb{X}_0 \cap \mathbb{X}_{FH}$, $\theta_s^*(x) = \theta^*(x)$ of Problem 8.1 and hence all constraints in $g(\theta, x) \preceq 0$ are satisfied.*

The next problem is related to the above problem, but also addresses the case when $x \notin \mathbb{X}_{FH}$. No prioritisation is required.

Problem 8.4 (Minimum number of constraint violations). *Set up Problem 8.2 such that $\forall x \in \mathbb{X}_0$ the solution is such that the minimum number of constraints in $g(\theta, x) \preceq 0$ are violated.*

The next two problems require the solution to be prioritised-optimal. Problem 8.4 can be seen to be a special case of Problem 8.5, which in turn is a special case of Problem 8.6.

Problem 8.5 (Uniquely prioritised constraints). *Given a set of constraints with each constraint associated with a different priority level, set up Problem 8.2 such that $\forall x \in \mathbb{X}_0$ the solution minimises the number of constraint violations in $g(\theta, x) \preceq 0$ in a prioritised-optimal fashion.*

Problem 8.6 (Multiple constraints with the same priority). *Given a set of constraints with subsets of constraints associated with the same priority level, set up Problem 8.2 such that $\forall x \in \mathbb{X}_0$ the solution minimises the number of constraint violations on each level in a prioritised-optimal fashion.*

Some applications often have lower and upper bounds on a variable. For a given θ and x, either both constraints are satisfied or only one violated. It is not possible for both to be violated at the same time. One can exploit this structure by putting both constraints on the same priority level and associating a single slack variable with the constraints. This is the motivation for the following problem.

Problem 8.7 (Exclusive constraint violations). *Given a set of constraints with subsets of constraints which cannot be violated at the same time but that are associated with the same priority level as other subsets of constraints, set up Problem 8.2 such that $\forall x \in \mathbb{X}_0$ the solution minimises the number of constraint violations in $g(\theta_s^*(x), x)$ in a prioritised-optimal sense.*

8.4 Main Results

The next result gives a condition on the scalar ρ such that the solution which minimises the cost function in Problem 8.2, also minimises $W'\delta$.

Lemma 8.3. *Let $W \in \mathbb{N}_+^t$ and*

$$\rho > \sup_{\theta,x,\epsilon} \tilde{f}(\theta, x, \epsilon) - \inf_{\theta,x,\epsilon} \tilde{f}(\theta, x, \epsilon) \tag{8.16a}$$

where it is assumed that the optimisations are bounded from above and below, subject to

$$x \in \mathbb{X}_0 \tag{8.16b}$$

$$g_k(\theta, x) \le \epsilon_m \tag{8.16c}$$

$$g_l(\theta, x) \le 0 \tag{8.16d}$$

$$0 \le \epsilon_m \le M_m \tag{8.16e}$$

$$k \in \mathcal{S}_m \tag{8.16f}$$

$$l \in \mathcal{H} \tag{8.16g}$$

$$m \in \mathcal{T}_n \tag{8.16h}$$

$$n \in \{1, 2, \ldots, t\}. \tag{8.16i}$$

If $(\theta_s^*(x), \epsilon^*(x), \delta^*(x))$ is an optimal solution to Problem 8.2 and $x \in \mathbb{X}_0$, then there does not exist another feasible solution $(\theta_s(x), \epsilon(x), \delta(x))$ such that $W'\delta(x) < W'\delta^*(x)$.

Proof. Assume that $(\theta_s^*(x), \epsilon^*(x), \delta^*(x))$ is an optimal solution to Problem 8.2 and that there exists another feasible candidate solution $(\theta_s(x), \epsilon(x), \delta(x))$ with $W'\delta(x) < W'\delta^*(x)$.

Let $V^* = \tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) + \rho W'\delta^*(x)$ and $V' = \tilde{f}(\theta_s(x), x, \epsilon(x)) + \rho W'\delta(x)$ be the values of the cost function in (8.12) for the two feasible solutions.

Since ρ is given by (8.16a), it follows that

$$\rho > |\tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) - \tilde{f}(\theta(x), x, \epsilon(x))|, \ \forall x \in \mathbb{X}_0 .$$

This allows one to proceed as follows:

$$\begin{aligned} V^* - V' &= \tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) + \rho W'\delta^*(x) - \tilde{f}(\theta(x), x, \epsilon(x)) - \rho W'\delta(x) \\ &= \tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) - \tilde{f}(\theta(x), x, \epsilon(x)) + \rho(W'\delta^*(x) - W'\delta(x)) \\ &\geq \tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) - \tilde{f}(\theta(x), x, \epsilon(x)) + \rho, \text{ since } W'\delta^*(x) - W'\delta(x) \geq 1 \\ &> \tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) - \tilde{f}(\theta(x), x, \epsilon(x)) + |\tilde{f}(\theta_s^*(x), x, \epsilon^*(x)) - \tilde{f}(\theta(x), x, \epsilon(x))| \\ &\geq 0 . \end{aligned}$$

This implies that $V' < V^*$ and that $(\theta(x), \epsilon(x), \delta(x))$ results in a lower cost function. This contradicts the assumption that $(\theta_s^*(x), \epsilon^*(x), \delta^*(x))$ is an optimal solution, thereby concluding the proof. □

The following theorem gives conditions on W and tells one how to set up Problem 8.2 such that the problems of Section 8.3.3 can be solved. Without loss of generality, it is assumed that all constraints are softened and ordered from highest to lowest priority.

Theorem 8.2. *With the given $\tilde{f}(\theta, x, \epsilon)$ as in Lemma 8.2 and ρ as in Lemma 8.3, assuming the optimal solution to Problem 8.2 can be found, one can set up Problem 8.2 to solve a number of related problems:*

1. *If $W \in \mathbb{N}_+^t$, then Problem 8.3 is solved;*
2. *Associate a unique slack variable, logic variable and priority level with each constraint, i.e. $\mathcal{P}_i = \{i\}$, $\mathcal{S}_m = \{m\}$, $\mathcal{T}_n = \{n\}$, $\mathcal{D}_n = \{n\}$ and hence $c = s = t = r$. If*

$$W = \mathbf{1}_t , \tag{8.17a}$$

then Problem 8.4 is solved;

3. *Associate a unique slack variable, logic variable and priority level with each constraint, i.e.* $\mathcal{P}_i = \{i\}$, $\mathcal{S}_m = \{m\}$, $\mathcal{T}_n = \{n\}$, $\mathcal{D}_n = \{n\}$ *and hence* $c = s = t = r$. *If*

$$W = \begin{bmatrix} 2^{t-1} \\ \vdots \\ 2^{t-i} \\ \vdots \\ 2^0 \end{bmatrix}, \tag{8.17b}$$

then Problem 8.5 is solved;

4. *Associate a unique slack variable and logic variable with each constraint and include the possibility that multiple constraints are associated with the same priority level, i.e.* $\mathcal{P}_i$ *is given by the constraints associated with priority level* i, $\mathcal{S}_m = \{m\}$, $\mathcal{T}_n = \{n\}$, $\mathcal{D}_n = \{n\}$ *and hence* $c = s = t \geq r$. *If*

$$W = \begin{bmatrix} w_1 \mathbf{1}_{t_1} \\ \vdots \\ w_i \mathbf{1}_{t_i} \\ \vdots \\ w_r \mathbf{1}_{t_r} \end{bmatrix}, \tag{8.17c}$$

with

$$w_i \geq 1 + \sum_{j=i+1}^{r} t_j w_j \tag{8.17d}$$

and $w_i \in \mathbb{N}_+$, *then Problem 8.6 is solved;*

5. *Associate a unique slack vector with each subset of constraints on the same priority level which cannot be violated at the same time. Associate a unique logic variable with each slack variable and include the possibility that multiple subsets of constraints are associated with the same priority level, i.e.* $\mathcal{P}_i$ *is given by the constraints associated with priority level* i, $\mathcal{S}_m$ *and* $\mathcal{D}_n$ *are given by the constraints associated with slack vector* m *and logic variable* n, $\mathcal{T}_n = \{n\}$ *and hence* $c \geq s = t \geq r$. *If* W *is chosen as in* (8.17c), *then Problem 8.7 is solved;*

Proof.

1. If $x \in \mathbb{X}_0 \cap \mathbb{X}_{FH}$, then $\theta^*(x)$, the optimal solution from Problem 8.1, satisfies $g(\theta^*(x), x) \preceq 0$ and hence $(\theta_s^*(x), \epsilon^*(x), \delta^*(x)) = (\theta^*(x), 0, 0)$ is a feasible solution to Problem 8.2. By considering Lemma 8.3, $(\theta^*(x), 0, 0)$ is also a candidate optimal solution, since $W'\delta^*(x) = 0$.

   This implies that the optimal solution has to satisfy $\delta^*(x) = 0$ and that $(\theta_s^*(x), \epsilon^*(x), \delta^*(x))$ is a solution to Problem 8.2 with the added constraints $\delta^*(x) = 0$ and $\epsilon^*(x) = 0$.

The new Problem 8.2 is now equivalent to Problem 8.1, hence the optimal solution to Problem 8.2 is given by $(\theta_s^*(x), \epsilon^*(x), \delta^*(x)) = (\theta^*(x), 0, 0)$. This solves Problem 8.3, by showing that the solution to the soft-constrained problem is equal to the solution of the hard-constrained problem.

2. If $x \in \mathbb{X}_0 \cap \mathbb{X}_{FH}$, then the proof is as above and all constraints are satisfied. For proving the result when $x \in \mathbb{X}_0 \backslash \mathbb{X}_{FH}$, note that if $W = \mathbf{1}_t$, then $W'\delta = \sum_{n=1}^t \delta_n$.

   By application of Lemma 8.2 and the fact that each constraint is associated with its own logic variable, $W'\delta^*(x)$ is equal to the number of constraint *violations* in $g(\theta_s^*(x), x)$.

   Lemma 8.3 implies that if the optimal solution has been found, then there does not exist another feasible solution with a lower $W'\delta^*(x)$, hence the optimal solution also minimises the number of constraint violations.

3. Since each constraint is associated with a unique priority level and logic variable, the optimal $\delta^*(x)$ indicates whether or not the associated priority level has been satisfied. As in Section 8.2, let $v_i(\theta(x))$ represent the number of violated constraints on priority level i for a given $\theta(x)$, then $v_i(\theta_s^*(x)) = \delta_i^*(x)$.

   Assume that the solution is optimal, but not prioritised-optimal. By Definition 8.2, this implies that there exists a $(\theta_s(x), \epsilon(x), \delta(x))$ with $\delta_{i^*}(x) < \delta_{i^*}^*(x)$, where i^* is the index of the first element where $\delta(x)$ and $\delta^*(x)$ differ, i.e. $\delta_{i^*}(x) = 0$ and $\delta_{i^*}^*(x) = 1$.

   By noting that $W'\delta = \sum_{n=1}^t 2^{t-n}\delta_n$ and that $(\delta_n(x) - \delta_n^*(x)) \in \{-1, 0, 1\}$, one can show the following:

$$
\begin{aligned}
W'\delta(x) - W'\delta^*(x) &= \sum_{n=1}^{t} 2^{t-n} \left(\delta_n(x) - \delta_n^*(x)\right) \\
&= \sum_{n=i^*}^{t} 2^{t-n} \left(\delta_n(x) - \delta_n^*(x)\right) \\
&= 2^{t-i^*} \left(\delta_{i^*}(x) - \delta_{i^*}^*(x)\right) + \sum_{n=i^*+1}^{t} 2^{t-n} \left(\delta_n(x) - \delta_n^*(x)\right) \\
&\leq 2^{t-i^*} \left(\delta_{i^*}(x) - \delta_{i^*}^*(x)\right) + \sum_{n=i^*+1}^{t} 2^{t-n} \\
&< 2^{t-i^*} \left(\delta_{i^*}(x) - \delta_{i^*}^*(x)\right) + 2^{t-i^*} \\
&= 2^{t-i^*}(-1+1) \\
&= 0\,.
\end{aligned}
$$

   This implies that $W'\delta(x) < W'\delta^*(x)$. The assumption that $\delta^*(x)$ is part of the optimal solution is contradicted, as implied by Lemma 8.3, hence the optimal solution is also prioritised-optimal.

4. Since each constraint is associated with a unique logic variable, the optimal $\delta^*(x)$ indicates whether the associated constraint has been satisfied. However, more than one constraint can be associated with a given priority level. As in Section 8.2, let $v_i(\theta(x))$ represent the number of violated constraints on priority level i for a given $\theta(x)$, then $v_i(\theta_s^*(x)) = \sum_{n\in\mathcal{P}_i} \delta_n^*(x)$.

   Assume that the solution is optimal, but not prioritised-optimal. By Definition 8.2, this implies that there exists a $(\theta_s(x), \epsilon(x), \delta(x))$ with $v_{i^*}(\theta_s(x)) < v_{i^*}(\theta_s^*(x))$, where i^* is the index of the first element where $v(\theta_s(x))$ and $v(\theta_s^*(x))$ differ.

   By noting that

$$W'\delta = \sum_{i=1}^{r}\sum_{n\in\mathcal{P}_i} w_i\delta_n = \sum_{i=1}^{r} w_i \sum_{n\in\mathcal{P}_i}\delta_n = \sum_{i=1}^{r} w_i v_i(\theta_s)\,,$$

   one can show the following:

$$\begin{aligned}
W'\delta(x) - W'\delta^*(x) &= \sum_{j=1}^{r} w_j\left[v_j(\theta_s(x)) - v_j(\theta_s^*(x))\right] \\
&= \sum_{j=i^*}^{r} w_j\left[v_j(\theta_s(x)) - v_j(\theta_s^*(x))\right] \\
&= w_{i^*}\left[v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x))\right] + \sum_{j=i^*+1}^{r} w_j\left[v_j(\theta_s(x)) - v_j(\theta_s^*(x))\right] \\
&= w_{i^*}\left[v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x))\right] + \sum_{j=i^*+1}^{r} w_j \sum_{n\in\mathcal{P}_j}(\delta_n(x) - \delta_n^*(x)) \\
&\le w_{i^*}\left[v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x))\right] + \sum_{j=i^*+1}^{r} w_j \sum_{n\in\mathcal{P}_j} 1 \\
&\le w_{i^*}\left[v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x))\right] + \sum_{j=i^*+1}^{r} w_j t_j \\
&< w_{i^*}\left[v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x))\right] + w_{i^*} \\
&= w_{i^*}\left\{\left[v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x))\right] + 1\right\} \\
&\le 0,\ \text{since } w_{i^*} \ge 1 \text{ and } v_{i^*}(\theta_s(x)) - v_{i^*}(\theta_s^*(x)) \le -1.
\end{aligned}$$

   This implies that $W'\delta(x) < W'\delta^*(x)$. The assumption that $\delta^*(x)$ is part of the optimal solution is contradicted, as implied by Lemma 8.3, hence the optimal solution is also prioritised-optimal.

5. The proof in the previous result can easily be extended for this case. The new objective that is introduced is that all the constraints associated with some single logic variable be satisfied. All of the constraints associated with the logic variable will be satisfied, if possible. If this is not possible, only one of them will be violated for a given $\theta_s^*(x)$ and the violation of that constraint implies the satisfaction of the other constraints associated with the logic variable.

□

Remark 8.4. Note that (8.17a) and (8.17b) are special cases of (8.17c).

Remark 8.5. Problem 8.7 can also be solved by treating it as a special case of Problem 8.6. However, treating Problem 8.7 as a special case of Problem 8.6 would introduce more slack variables and logic variables than are necessary.

Remark 8.6. It is trivial to extend Theorem 8.2 to the case where more than one slack variable is associated with a single logic variable, if the application requires this. The weight W remains as in (8.17c). However, if one of the constraints associated with the logic variable is violated, one cannot guarantee that the other constraints associated with the logic variable are satisfied, unless the violation is exclusive as in Problem 8.7.

8.5 Special Cases and Simplifications

Up to now, the case of a general mp-MINLP has been considered. In general it is difficult to implement and compute the resulting soft-constrained problem if the cost function and constraints do not take on a special form. This section discusses the special case of when the original problem is an mp-MIQP or mp-MILP, as occurs when setting up MPC problems, and how one could proceed in computing a value for ρ.

8.5.1 The Model Predictive Control Problem as an mp-MIQP or mp-MILP

Since integer variables can be represented by an appropriate number of binary variables, it is assumed from this point on that $\theta \in \mathbb{R}^{d_1} \times \{0,1\}^{d_2}$ and $x \in \mathbb{R}^{p_1} \times \{0,1\}^{p_2}$. The class of MPC problems with quadratic costs and linear inequality constraints results in optimisation problems of the form

$$f(\theta, x) = \frac{1}{2}\theta' H\theta + x'F'\theta\,, \tag{8.18a}$$

where the cost function is convex in the decision variable θ, i.e. $H \succeq 0$, and the linear part is dependent on the parameter x, which is usually the current state of the plant. Furthermore, the constraints can often be written in the form:

$$G_1\theta \preceq g_2 + G_3 x\,, \tag{8.18b}$$

with $G_1 \in \mathbb{R}^{c\times(d_1+d_2)}$, $g_2 \in \mathbb{R}^c$ and $G_3 \in \mathbb{R}^{c\times(p_1+p_2)}$. Note that the right hand side is dependent on x.

This is a hard-constrained mp-MIQP (multi-parametric Mixed-Integer Quadratic Program) and a soft-constrained mp-MIQP can be set up as in Sections 8.3 and 8.4 by introducing appropriate norms for the slack variables and additional logic variables such that

$$\tilde{f}(\theta, x, \epsilon) = \frac{1}{2}\begin{bmatrix}\theta\\ \epsilon\end{bmatrix}'\begin{bmatrix}H & 0\\ 0 & S_2\end{bmatrix}\begin{bmatrix}\theta\\ \epsilon\end{bmatrix} + \begin{bmatrix}Fx\\ S_1\end{bmatrix}'\begin{bmatrix}\theta\\ \epsilon\end{bmatrix}, \tag{8.19}$$

where $S_1 \in \mathbb{R}^s$ and $S_2 \in \mathbb{R}^{s \times s}$ are determined from the (weighted) norm used in penalising the constraint violations.

Even in this simple form, the computation of ρ in (8.16a) is not easy, since the cost function $\tilde{f}(\theta, x, \epsilon)$ is not necessarily convex in θ, ϵ and x, even if $H \succeq 0$ and $S_2 \succeq 0$, as can be seen by rewriting (8.19) as:

$$\tilde{f}(\theta, x, \epsilon) = \frac{1}{2} \begin{bmatrix} \theta \\ x \\ \epsilon \end{bmatrix}' \begin{bmatrix} H & F & 0 \\ F' & 0 & 0 \\ 0 & 0 & S_2 \end{bmatrix} \begin{bmatrix} \theta \\ x \\ \epsilon \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ S_1 \end{bmatrix}' \begin{bmatrix} \theta \\ x \\ \epsilon \end{bmatrix} . \tag{8.20}$$

It is easy to find values for F that result in a Hessian which is not positive semi-definite.

If $H = 0$ and $S_2 = 0$ the soft-constrained problem becomes that of an mp-MILP (multi-parametric Mixed-Integer Linear Program). However, the computation of ρ is still not easy, since the cost functions in (8.16a) are still indefinite quadratics in x.

In both the mp-MIQP and mp-MILP, unless some structure about the problem is known, the most practical solution might be to make a conservative guess at the value of ρ.

8.5.2 Decomposing the Soft-Constrained Problem

If the computation of ρ is difficult and it is crucial that the subset of parameters for which one can guarantee that the solution is prioritised-optimal is maximal, i.e. $\mathbb{X}_0 = \mathbb{X}_{FS}$, then one can decompose Problem 8.2 into two (or more) steps. The first step would be to solve for

$$(\tilde{\theta}(x), \epsilon^*(x), \delta^*(x)) = \arg\min_{\theta,\epsilon,\delta} \sum_{m=1}^{s} \|\epsilon_m\| + \rho W'\delta \tag{8.21}$$

subject to the original soft constraints (8.13) in Problem 8.2. This step finds a solution which is prioritised-optimal in terms of the number of satisfied constraints. The next step would be to solve for

$$\theta_s^*(x) = \arg\min_{\theta} f(\theta, x) \tag{8.22}$$

subject to (8.13), but with the solutions $\epsilon^*(x)$ and $\delta^*(x)$ to the first part substituted into the constraints. In this case, since the slack vectors are known to be bounded, it is easy to find

$$\rho > \sum_{m=1}^{s} \|M_m\| . \tag{8.23}$$

Whether the constraint violations should be penalised in the first or second step is problem-dependent - the same $\delta^*(x)$ will result. However, it can be seen that by penalising the violations in the second step instead of the first, i.e. $\min_{\theta,\delta,\epsilon} W'\delta$ followed by $\min_{\theta,\epsilon} f(\theta, x) + \sum_{m=1}^{s} \|\epsilon_m\|$, that it is *not* necessary to calculate a value for ρ. In addition, the restriction that $w_i \in \mathbb{N}_+$ can also then be relaxed to $w_i \in \mathbb{R}_+$, as in Theorem 8.1.

One does not need to have slack variables for the solution to be prioritised-optimal with respect to the number of constraint violations. If $\tilde{f}(\theta, x, \epsilon) = f(\theta, x) + \sum_{m=1}^{s} \|\epsilon_m\|$, then a trade-off between $f(\theta, x)$ and the size of the constraint violations has to be made. Adding slack vectors also adds decision variables to the optimisation problem, thereby increasing the computational effort. Removing the slack variables and not caring about the size of the constraint violations, amounts to setting $\tilde{f}(\theta, x, \epsilon) = f(\theta, x)$ and replacing (8.13a) and (8.13c) with

$$g_k(\theta, x) \leq M_m \delta_n \, .$$

By decomposing the problem as in the above two approaches, one can guarantee that the solution is prioritised-optimal in terms of constraint satisfaction for all $x \in \mathbb{X}_{FS}$.

8.6 Model Predictive Control of Hybrid Systems

The MLD modelling framework mentioned in Section 4.2 allows one to design MPC controllers for hybrid systems. The following problem has to be solved in order to compute an MPC controller for an MLD system:

Problem 8.8. *[BM99a] Given the initial state x_k and a control horizon N, find (if it exists) the control sequence $\pi_k^N \triangleq \{\hat{u}_{0|k}, \hat{u}_1, \ldots, \hat{u}_{N-1|k}\}$ which transfers the state from x_k to x_f and minimises the performance index*

$$V(\theta, x_k) \triangleq \sum_{l=0}^{N-1} \|\hat{u}_{l|k} - u_f\|_{Q_1}^2 + \|\hat{\delta}_{l|k} - \delta_f\|_{Q_2}^2 + \|\hat{z}_{l|k} - z_f\|_{Q_3}^2 + \|\hat{x}_{l|k} - x_f\|_{Q_4}^2 + \|\hat{y}_{l|k} - y_f\|_{Q_5}^2 \tag{8.24a}$$

subject to

$$\hat{x}_{N|k} = x_f \tag{8.24b}$$

and the MLD system dynamics (4.1)*, where $\|\alpha\|_Q^2 \triangleq \alpha' Q \alpha$, $Q_i = Q_i' \succeq 0$, $i = 1 \ldots 5$ are given weight matrices, and x_f, u_f, δ_f, z_f, y_f are given offset vectors*[4] *satisfying* (4.1b) *and* (4.1c)*. The decision variable θ is made up of all the $\hat{u}_{l|k}$, $\hat{\delta}_{l|k}$ and $\hat{z}_{l|k}$.*

As shown in [BM99a, Sect. 5], this problem is an mp-MIQP. Solving the problem is equivalent to minimising an appropriate cost function in the form (8.18a) subject to constraints (8.18b), with the parameter $x = x_k$. It is also possible to add additional performance or safety constraints on the states and inputs of the system, i.e. $\hat{x}_{l|k} \in \mathbb{X}$ and $\hat{u}_{l|k} \in \mathbb{U}$

[4]These vectors correspond to a steady state which is compatible with the constraints. An MIQP can be set up for computing these values [BM99a, Sect. 6.1].

Since the constraints in the problem are dependent on the current state, it is possible that a disturbance could drive the system outside $\mathbb{X}_{FH}$, resulting in it being impossible to compute a solution to the original hard-constrained problem. Assuming it is still possible to drive the system to x_f in N steps when some or all of the performance constraints have been relaxed, one would like to design a soft-constrained problem which prioritises the soft constraints. The procedure described in Sections 8.3–8.5 allows one to construct such a problem.

Remark 8.7. When dealing with MPC problems, it is relatively easy to get a conservative lower bound for ρ, as the expressions in the cost function represent physical variables which are bounded. Since each $\|\cdot\|^2_{Q_i}$ is convex, it is relatively easy to obtain an upper bound on the maximum and likewise when introducing slack variables. Additionally, the cost function is also always bounded below by 0.

Various priorities can usually be assigned to the soft constraints on the inputs and states. For example:

- It might be less desirable to violate the performance constraints on a given output than constraints on other outputs and therefore the soft constraints associated with the first output have a higher priority than the soft constraints of the other outputs.
- Another design requirement might be that if the performance constraints on an output have to be violated, that it be brought back into the desired region as soon as possible, regardless of the satisfaction by other outputs of their corresponding performance constraints. In this case, constraints in the future have a higher priority than constraints closer to the current time. In addition, all the constraints on the output have a higher priority than constraints on the other outputs.
- A third case would be where redundant hardware has been installed for safety purposes and one would like to use the hardware only to prevent a fault from developing into plant failure. Constraints on inputs and outputs associated with the redundant hardware therefore have higher priority than all other performance constraints.

If there are a large number of inputs and outputs and a large horizon N and one tries to associate a separate priority level with each constraint, the weights in W will become very large and the problem ill-conditioned. However, in practical situations one can rarely associate a large number of distinct priority levels with all the inputs and states, and this is therefore not a serious problem. It is also possible to include time priorities without the need for separate weights for each time constraint and this will be discussed next.

8.6.1 Minimum-Time Output-Prioritised Solutions

Assume that the system has only one output and that one prioritises the constraints on the output such that a constraint at time $k+1$ has a higher priority than a constraint at time k, and one chooses the W appropriately to reflect this priority. If the only constraints that have been softened are the performance constraints related to the output, then it can be seen that the solution is *minimum-time* optimal in the sense that the duration of constraint violations has been minimised. The same minimum-time optimal solution will result if one adopts the approach of [VSJ99], [TM99, Sect. 3.2] or [BM99a, Sect. 5.1] for a MISO system.

However, if one has a MIMO system and the inputs and states are prioritised then the problem becomes the following:

Problem 8.9 (Minimum-time output-prioritised). *Set up a soft-constrained problem for which the solution is output-prioritised-optimal with respect to the duration of constraint violations over the horizon N.*

Note that the problem has changed from trying to find a solution which is prioritised-optimal in the *number* of constraint violations to minimising the *duration* of constraint violations subject to the prioritisation. These are two different problems and adopting the unmodified approach of Sections 8.3 and 8.4 does not solve the latter problem. One has to redefine what $v(\theta)$ represents in order to understand why this is the case.

Let $v_i(\theta)$ now represent the sum of the *durations* of constraint *relaxation* for the inputs and states associated with priority level i. The solution is therefore required to be prioritised-optimal with respect to $v(\theta)$. For example, assume that the soft-constrained problem has been set up such that $[\delta_j^k = 1]$ is true if the associated constraint on the j'th input or state at time k has been *relaxed*[5]. Let $k_j^* = \max_{k\in\{0,\dots,N\}} k$ such that $[\delta_j^k = 1]$, then k_j^* is the duration of constraint relaxation of the j'th state or input on priority level i. If this is the case, then $v_i(\theta) = \sum_j k_j^*$.

In light of this, it can be seen that the original problem is such that $[\delta^*(x) = 1]$ is true if and only if the desired input or state constraint has been violated, not just relaxed. By forcing higher-prioritised constraints to be satisfied, it could result in a 'water bed' effect where constraints on lower-prioritised states or inputs cannot be satisfied, thereby possibly increasing the duration of constraint violation for those inputs or states. The same effect will occur when the approach in [VSJ99] is used.

All that remains is to modify Problem 8.2 such that $[\delta^*(x) = 1]$ is true if the associated input or state constraint has been relaxed, but not necessarily violated. The solution should be such that the duration of constraint relaxation is prioritised-optimal. A modification of the approach in [TM99, Sect. 3.2] and [BM99a, Sect. 5.1] allows one to do just this. Additionally, it will be shown that it is sufficient for all constraints and logic variables associated with the state or input on priority level i to have the same weight w_i.

Theorem 8.3 (Minimum-time output-prioritised). *Let the upper and lower bounds of the r outputs of the MLD system* (4.1) *be given by $\overline{y}$ and $\underline{y}$, respectively.*

[5]Note that a relaxed constraint is not necessarily violated. If a constraint has been relaxed, it implies that violation is allowed, but satisfaction is still possible. For example, a constraint $g(\theta) \leq 0$ has been relaxed if it has been replaced by $g(\theta) \leq M$, where M is some positive number.

Let the violation of output i at time $k+l$ be given by ϵ_i^l and the maximum allowed violation of the constraints of the i'th output be M_i. Furthermore, the outputs are such that output i has a higher priority than output $i+1$.

A control sequence which minimises the duration of output constraint violations in an output-prioritised fashion and optimally transfers the state from x_k to x_f in N steps is found by solving the following MIQP:

$$\min_{\theta,\epsilon,\delta} J(\theta, x_k) + \sum_{l=1}^{N-1} \|\epsilon^l\|_{Q_6}^2 + \rho W'\delta \tag{8.25a}$$

subject to

$$\underline{y} - \epsilon^l \preceq \hat{y}_{l|k} \preceq \overline{y} + \epsilon^l, \quad l = 1, \dots, N-1 \tag{8.25b}$$

$$0 \leq \epsilon_i^l \leq M_i \delta_i^l, \quad i = 1, \dots, r,\ l = 1, \dots, N-1 \tag{8.25c}$$

$$\delta_i^{l+1} \leq \delta_i^l, \quad i = 1, \dots, r,\ l = 1, \dots, N-2 \tag{8.25d}$$

$$\hat{x}_{N|k} = x_f \tag{8.25e}$$

and the MLD system dynamics (4.1)*, with $Q_6 \succeq 0$.*

The prioritised logic vector $\delta \in \{0,1\}^{r(N-1)}$ is defined as

$$\delta \triangleq \begin{bmatrix} \delta_1 \\ \vdots \\ \delta_r \end{bmatrix} \tag{8.25f}$$

with

$$\delta_i \triangleq \begin{bmatrix} \delta_i^1 \\ \vdots \\ \delta_i^{N-1} \end{bmatrix}. \tag{8.25g}$$

The priority weight vector is defined as

$$W \triangleq \begin{bmatrix} w_1 \mathbf{1}_{N-1} \\ \vdots \\ w_i \mathbf{1}_{N-1} \\ \vdots \\ w_r \mathbf{1}_{N-1} \end{bmatrix} \tag{8.25h}$$

with

$$w_i \geq 1 + \sum_{j=i+1}^{r} (N-1) w_j \tag{8.25i}$$

and $w_i \in \mathbb{N}_+$.

The vector of constraint violations ϵ *is defined as:*

$$\epsilon \triangleq \begin{bmatrix} \epsilon^1 \\ \vdots \\ \epsilon^{N-1} \end{bmatrix} \tag{8.25j}$$

with

$$\epsilon^l \triangleq \begin{bmatrix} \epsilon^l_1 \\ \vdots \\ \epsilon^l_r \end{bmatrix}. \tag{8.25k}$$

The optimisation in

$$\rho > \max_{\theta,\epsilon,\delta} J(\theta, x_k) + \sum_{l=1}^{N-1} \|\epsilon^l\|^2_{Q_6} \tag{8.25l}$$

is subject to the same constraints as above.

Proof. If the constraints (8.25d) are removed, then it can be seen that the solution solves a special case of Problem 8.6. All the constraints associated with output i have the same priority, i.e. $t_i = N - 1$. If these constraints weren't included, then the solution would minimise the number of constraint violations in an output-prioritised fashion. With the addition of the constraints (8.25d) the problem is slightly modified.

The constraints (8.25d) are equivalent to the propositional logic statements

$$[\delta^l_i = 0] \rightarrow [\delta^{l+1}_i = 0], \quad i = 1, \dots, r, \ l = 1, \dots, N-2$$

and imply that if the constraints of output i are satisfied at time $k + l$, then the constraints of output i are satisfied from time $k + l + 1$ to $k + N - 1$.

If $[\delta^l_i = 1]$, then this implies that $[\delta^j_i = 1]$ for $j = 1, \dots, l-1$ and hence the constraints for output i have been *relaxed*[6] from time $k + 1$ to time $k + l$.

With the above choice of ρ and w_i and the fact that the cost function is always non-negative, it follows from the same argument as in the proof of Lemma 8.3 that the optimal solution also minimises $W'\delta$.

Note that $\sum_{l=1}^{N-1} \delta^l_i$ is now equal to the duration of constraint relaxation for output i. If one defines

$$v_i(\theta) \triangleq \sum_{l=1}^{N-1} \delta^l_i \,,$$

[6]It is stressed again that this *does not imply* that the constraints have been *violated*.

then

$$(w_i \mathbf{1}_{N-1})' \delta_i = w_i v_i(\theta) \, .$$

By Theorem 8.1 and the fact that $W'\delta$ is minimised, it follows that the optimal solution minimises the duration of constraint relaxations (and hence the *duration* of constraint violations[7]) in a prioritised-optimal fashion. The problem of minimising the duration of constraint relaxations of output i takes higher priority than the minimisation of the duration of constraint relaxations of output $i+1$.

□

This result illustrates how constraints can be added to include time priorities, but reduce the size of the components in W. Variations on this theme are possible by combining it with ideas from the previous sections.

8.7 Fault-Tolerant Control Example: The Three-Tank Benchmark

In [HL99] a benchmark problem was formulated as part of the COSY (Control of Complex Systems) project in order to compare reconfiguration strategies for fault-tolerant control. A number of solutions to this problem are given in [LAC+00]. This section demonstrates how a prioritised optimisation problem can be set up to determine a prioritised-optimal steady-state under the various fault scenarios as defined in [HL99].

8.7.1 Description of the Tank System

The benchmark problem consists of three coupled tanks, as shown in Figure 8.1. The tanks are connected by pipes and the flows through these pipes are controlled by switching valves (V_1, V_{13}, V_2, V_{32}) which can only be completely opened or completely closed. The left and right tanks can be filled using two identical pumps (P_1 and P_2). The continuous measurements of the levels in the tanks h_1, h_2 and h_3 are available. The system is hybrid by nature, since there are both continuous and discrete inputs and states and an MLD model of the system is given in [BMM99, Mig99].

The level in each tank and the flow rate of the pumps are bounded:

- The height of each tank is 62 cm, i.e. $0 \leq h_i \leq 0.62$ [m], $i = 1, 2, 3$;
- The inflows into tanks T_1 and T_2 are limited to the range $0 \leq Q_i \leq 0.1 \times 10^{-3}$ [m^3/s], $i = 1, 2$;

The connection pipes between the tanks are placed at the bottom of the tanks (pipes with valves V_{13} and V_{32} and at a height of 30 cm (pipes with valves V_1 and V_2). Valve V_{1L} can be used to simulate a leak in tank T_1. If there is no water flowing through the leak, then $Q_{1L} = 0$, otherwise $Q_{1L} > 0$.

[7] Not the *number* of constraint violations.

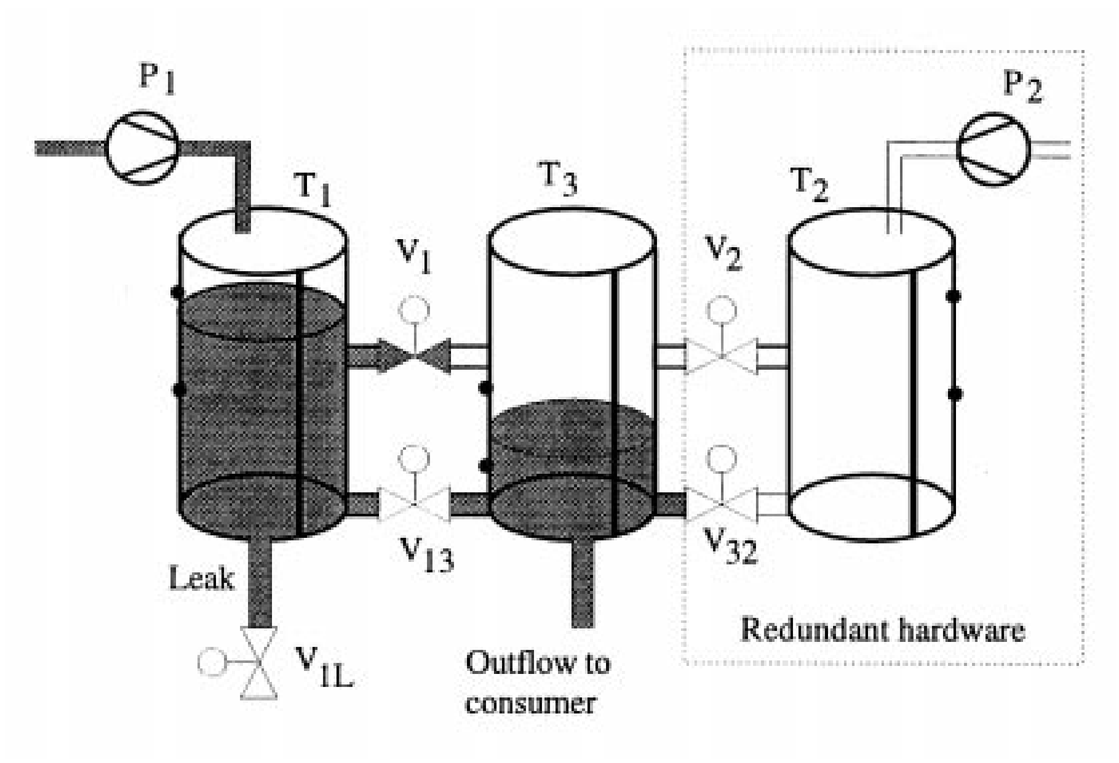

Figure 8.1: The Three-tank Benchmark Problem

The main aim of the tank system is to provide a continuous water flow Q_N to the consumer. In the nominal mode of operation tank T_1 is used as a buffer to control the level of tank T_3 in the range $9\,\text{cm} \leq h_3 \leq 11\,\text{cm}$. The nominal level for tank T_1 is $h_1 = 50\,\text{cm}$. Tank T_2 and pump P_2 are not used and act as redundant hardware.

In normal operation, a PI controller is used to control the flow rate Q_1 of pump P_1 in order to keep $h_1 = 50\,\text{cm}$ and valve V_1 is used to control h_3. All other valves are closed and pump P_2 is switched off, i.e. $Q_2 = 0$.

8.7.2 The Reconfiguration Problem

For the reconfiguration problem, three different fault scenarios are given:

1. Valve V_1 is blocked in the closed position, i.e. $V_1 = 0$;
2. Valve V_2 is blocked in the open position, i.e. $V_2 = 1$;
3. Valve V_{1L} is open, i.e. $V_{1L} = 1$, thereby simulating a leak in tank T_1.

The reconfiguration task, as defined in [HL99], is to automatically find a new control configuration of the three-tank system for each one of the scenarios above such that:

- The level h_3 remains within the nominal operating range, if possible;
- The loss of water is minimised, given the last scenario.

The reconfiguration problem involves determining the set of actuators, sensors, control laws and set-points such that the control aims above are attainable. The use of the redundant hardware is allowed.

8.7.3 Steady-State Analysis

As can be seen in Problem 8.8, the deviations of the predicted from the steady-state values are penalised when determining an MPC control action. A steady-state value for an MLD system can be determined with an MIQP:

$$\min_{x_f, u_f, \delta_f, z_f} \|y_f - r\|^2 + \|x_f\|^2_{\rho_4} + \|u_f\|^2_{\rho_1} + \|z_f\|^2_{\rho_3} + \|\delta_f\|^2_{\rho_2} \tag{8.26}$$

subject to

$$x_f = Ax_f + B_1 u_f + B_2 \delta_f + B_3 z_f \tag{8.27a}$$

$$y_f = Cx_f + D_1 u_f + D_2 \delta_f + D_3 z_f \tag{8.27b}$$

$$E_2 \delta_f + E_3 z_f \preceq E_1 u_f + E_4 x_f + E_5 \tag{8.27c}$$

where ρ_i are small, positive definite weighting matrices and r is a constant reference.

It is possible that the resulting steady-state is unreachable or gives poor performance, or even that a steady-state does not exists but that a cyclical steady-state is possible [TMFM01]. For the initial investigation presented here, it is assumed that a reachable steady-state exists.

Furthermore, it will be assumed that an FDI (fault diagnosis and identification) routine is available and that the fault is correctly identified and modelled. A steady-state for each of the fault conditions can then be computed using the above MIQP.

Defining the Priorities

In many practical systems some degree of redundancy is available and it is possible that many optimal steady-states exist. Additionally, it is often also possible that certain steady-states are preferred over others.

In the three-tank benchmark problem it is possible to define the following constraint objectives from highest to lowest priority:

1. Minimise the water loss. Obviously water loss is minimised when

$$Q_{1L} = 0\,;$$

2. Maintain a good rate of water flow to the consumer. Within this context it is possible to say that a high rather than a low rate of flow is preferred. The highest priority is therefore to keep

$$h_3 \geq 9\,\text{cm};$$

3. Do not use the redundant hardware. This translates into assigning the following two constraints[8] the same priority:

$$V_{32} = 0,\ V_2 = 0\,;$$

4. Close all valves not used in steady-state[9], such as valve V_{13} connecting tanks T_1 and T_3:

$$V_{13} = 0\,;$$

5. Keep the flow rate to the consumer below a certain level. This translates into keeping

$$h_3 \leq 11\,\text{cm}\,;$$

6. Minimise the fluctuation in flow rate to the consumer. This could be achieved by keeping the level of tank T_3 at some constant value, say

$$h_3 = h_{3\text{nom}} = 10\,\text{cm}\,;$$

7. Keep the level of tank T_1 at the nominal value:

$$h_1 = h_{1\text{nom}} = 50\,\text{cm}\,.$$

Since some of the above constraints are not defined in the original benchmark, it is possible to choose any other sensible combination. It is felt that the above list of prioritised objectives reflect what a plant operator would try to achieve with manual control.

The Prioritised Optimisation Problem

For each of the fault scenarios, the following prioritised MIQP can be solved for computing the optimal steady-state $(x_f, u_f, \delta_f, z_f)$:

$$\min_{x_f, u_f, \delta_f, z_f, \epsilon, \delta} \|y_f - r\|^2 + \|x_f\|^2_{\rho_4} + \|u_f\|^2_{\rho_1} + \|z_f\|^2_{\rho_3} + \|\delta_f\|^2_{\rho_2} + \|\epsilon\|^2_{\rho_5} + \rho W'\delta$$

[8]The inclusion of the constraint $Q_2 = 0$ is not necessary, since one can include $\|Q_2\|^2$ in the cost function. If $V_{32} = 0$ and $V_2 = 0$ then the optimal solution would be such that $Q_2 = 0$.

[9]The use of valve V_1 is necessary in steady-state.

where the minimisation is subject to the following constraints:

$$\begin{aligned}
Q_{1L} &\leq \epsilon_{Q_{1L}} \\
h_3 &\geq 0.09 - \epsilon_{h_{3l}} \\
V_{32} &\leq \epsilon_{V_{32}} \\
V_2 &\leq \epsilon_{V_2} \\
V_{13} &\leq \epsilon_{V_{13}} \\
h_3 &\leq 0.11 + \epsilon_{h_{3h}} \\
0.1 - \epsilon_{h_3} \leq h_3 &\leq 0.1 + \epsilon_{h_3} \\
0.5 - \epsilon_{h_1} \leq h_1 &\leq 0.5 + \epsilon_{h_1} \,,
\end{aligned}$$

the lower bounds on the slack variables $\epsilon \triangleq [\epsilon_{Q_{1L}}\ \epsilon_{h_{3l}}\ \epsilon_{V_{32}}\ \epsilon_{V_2}\ \epsilon_{V_{13}}\ \epsilon_{h_{3h}}\ \epsilon_{h_3}\ \epsilon_{h_1}]'$:

$$\epsilon \succeq 0 \,,$$

the upper bounds on the slack variables which also associate the slack variables with the prioritised logic vector $\delta \triangleq [\delta_{Q_{1L}}\ \delta_{h_{3l}}\ \delta_{V_{32}}\ \delta_{V_2}\ \delta_{V_{13}}\ \delta_{h_{3h}}\ \delta_{h_3}\ \delta_{h_1}]'$:

$$\begin{aligned}
\epsilon_{Q_{1L}} &\leq (Q_{1Lmax})\delta_{Q_{1L}} \\
\epsilon_{h_{3l}} &\leq 0.09\delta_{h_{3l}} \\
\epsilon_{V_{32}} &\leq 1\delta_{V_{32}} \\
\epsilon_{V_2} &\leq 1\delta_{V_2} \\
\epsilon_{V_{13}} &\leq 1\delta_{V_{13}} \\
\epsilon_{h_{3h}} &\leq (0.62 - 0.11)\delta_{h_{3h}} \\
\epsilon_{h_3} &\leq \max(0.62 - h_{3\text{nom}}, h_{3\text{nom}})\delta_{h_3} \\
\epsilon_{h_1} &\leq \max(0.62 - h_{1\text{nom}}, h_{1\text{nom}})\delta_{h_1} \,,
\end{aligned}$$

and the MLD steady-state equations of the faulty tank system in the form

$$\begin{aligned}
x_f &= Ax_f + B_1u_f + B_2\delta_f + B_3z_f \\
y_f &= Cx_f + D_1u_f + D_2\delta_f + D_3z_f \\
E_2\delta_f + E_3z_f &\preceq E_1u_f + E_4x_f + E_5 \,.
\end{aligned}$$

The physical constraints on the inputs and states (such as $Q_{1L} \geq 0$ and $0 \leq h_i \leq 0.62$) are included either directly or implicitly in the MLD model and are therefore not listed above.

If the output y_k is

$$y_k = [h_1\ h_2\ h_3]'$$

and the reference r is

$$r = [h_{1\text{nom}}\ 0\ h_{3\text{nom}}]' = [0.5\ 0\ 0.1]' \,,$$

Table 8.1: Results from the steady-state computation for the Three-Tank Benchmark

Fault Condition	h_1 m	h_2 m	h_3 m	Q_1 m^3/s	Q_2 m^3/s	Q_{1L} m^3/s	V_1	V_{13}	V_2	V_{32}
No Fault	0.4	0.0	0.1	1.13×10^{-5}	0	0	open	closed	closed	closed
V_1 open	0.4	0.0	0.1	1.13×10^{-5}	0	0	open	closed	closed	closed
V_1 closed	0.2	0.0	0.1	1.13×10^{-5}	0	0	closed	open	closed	closed
Leak in T_1	0.0	0.2	0.1	0	1.13×10^{-5}	0	closed	closed	closed	open

then the the scalar ρ has to be chosen such that

$$\rho > \max_{x_f, u_f, \delta_f, z_f, \epsilon, \delta} \|y_f - r\|^2 + \|x_f\|^2_{\rho_4} + \|u_f\|^2_{\rho_1} + \|z_f\|^2_{\rho_3} + \|\delta_f\|^2_{\rho_2} + \|\epsilon\|^2_{\rho_5}$$

where the optimisation also has to be performed subject to the above constraints. For the three-tank system it is possible to get a lower bound for ρ simply by inspection[10].

A W which reflects the priorities defined earlier is given by

$$W = [96\ 48\ 16\ 16\ 8\ 4\ 2\ 1]' .$$

Note that the same weight is assigned to both $\delta_{V_{32}}$ and δ_{V_2}, since the associated constraints have the same priority.

The above MIQP was formulated only as an example. Though this MIQP would solve the problem, it was not the one that was implemented. Several computational simplifications can be made by noting that some of the auxiliary variables will be equal to the states of the system at the optimal solution, thereby allowing one to reduce the number of decision variables, e.g.

$$\delta^*_{V_2} = \epsilon^*_{V_2} = V_2 \in \{0, 1\} .$$

Discussion of the Steady-State Computation

Table 8.1 gives a summary of the solutions to the above optimisation problem for each of the three fault scenarios. As can be seen, the computed steady-states satisfy both of the reconfiguration criteria defined in Section 8.7.2, namely keeping h_3 in the nominal range and minimising water loss in the case of a leak occurring (in fact $Q_{1L} = 0$). Valve V_{13} is used only when valve V_1 is blocked closed, thereby allowing water to flow from tank T_1 to tank T_3.

The redundant hardware is only required in steady-state when there is a leak, with tank T_2 acting as buffer instead of tank T_1. Valve V_{32} is used instead of valve V_2, since they both have the same priority. This configuration allows h_2 to be closer to the set-point of 0 m.

[10]For the actual implementation with all the $\rho_i = I$, a value of $\rho = 3$ was sufficient.

Note that the steady-state for the case with no fault is the same as the steady-state for when valve V_1 is blocked open. This is exactly what one would expect, since in steady-state V_1 is open for the nominal case.

It should also be observed that even for the nominal case h_1 cannot be kept at the desired level of 0.5 m at steady-state. This is due to Toricelli's law

$$Q = aS\sqrt{2gh}\,,$$

which says that the flow rate Q through an opening with cross-section S is proportional to the square root of the height h of liquid above the opening[11].

In the three-tank system, the pipes and their openings are identical. This implies that, in steady-state, the flow rate to the consumer must be equal to the flow rate of water coming into tank T_3. This in turn implies that the level of water above the pipe with valve V_1 must be equal to h_3. Since this pipe is placed at 0.3 m, one would expect the level of tank T_1 to be $h_1 = h_3 + 0.3 = 0.4$, which agrees with the computed value.

As a further consequence, one would expect the flow rate of pump P_1 to be equal to the flow rate to the consumer. Using Toricelli's law, one would expect $Q_1 = 2.80 \times 10^{-5}$ m^3/s. However, the MLD model uses a linearised approximation of Toricelli's law [BMM99, Mig99] and hence the computed value of $Q_1 = 1.13 \times 10^{-5}$ m^3/s is different from the ideal value, but correct for the model as implemented. Obviously a better approximation will result in a more accurate estimate of the flow rate.

This example showed that by a careful choice of objectives and priorities a single optimisation could be set up to calculate a sensible steady-state which satisfies as many of the objectives as possible. A change in priorities is easily reflected by a suitable change in the weight W. The scheme proposed in this chapter therefore allows one to add or remove constraints and change priorities in a simple, transparent fashion.

8.8 Summary

Prioritised, multi-objective problems were introduced and it was shown how weights can be chosen such that, for a class of problems where the cost function only takes on integer values, the solution is prioritised-optimal. A soft-constrained mixed-integer programming problem was then formulated and the problem of finding a solution which minimises the number of constraint violations in a prioritised-optimal sense was posed.

The ideas from the first part of the chapter were applied to the choice of weights in the soft-constrained problem. This was then further applied to the case of designing an MPC controller for the control of hybrid systems, where some outputs have higher priority than others. Finally, the ideas were applied to the problem of determining the setpoints for a three-tank system given a number of fault scenarios.

[11]The other terms in the equation are the gravity constant g and a flow correction term a.

Chapter 9

Concluding Remarks

In conclusion, the main contributions of this thesis are summarised and suggestions for possible future directions are outlined.

9.1 Contributions

The central idea behind this thesis was to develop a framework for the synthesis of robust controllers which guarantee constraint satisfaction. The main contributions of this thesis are summarised below.

Invariant Set Theory

- A number of important ideas from set invariance theory were brought together and placed in a general, nonlinear setting. The essential ingredients required for computing robust controllable and invariant sets were identified and discussed.
- Some less well-known results regarding the efficient computation of the linear map of a polyhedron and subset testing were given.
- A method for computing the Pontryagin difference between the union of a set of convex polyhedra and a convex polyhedron was described. It was shown how this allows one to compute robust controllable sets for piecewise affine systems.

Model Predictive Control

- A new sufficient condition was derived for guaranteeing that a given MPC controller will be feasible for all time, despite the possible sub-optimality of the solution at each time step. The effect of the length of the horizons and choice of terminal constraint on the behaviour of the feasible set and the feasibility of the MPC controller was also investigated.

- A necessary and sufficient condition was derived for analysing whether a given MPC controller will be feasible for all time, despite the presence of disturbances and the possible sub-optimality of the solution at each time step. This allows one to determine whether or not it is necessary to modify the given MPC controller in order to robustify it against disturbances.
- The robustification of the standard MPC scheme via the addition of a robustness constraint was discussed. A new necessary and sufficient condition as well as some new sufficient conditions were given in order to guarantee that the new controller would be robust strongly feasible. It was also shown how to modify the controller to guarantee strong robust feasibility for LTI systems in the presence of state disturbances and parametric uncertainty.
- Ideas from set invariance theory were applied to the problem of computing a steady-state set-point which is compatible with the constraints, while bearing in mind that there are unknown disturbances on the state and output.

Constrained Optimisation

- An algorithm was described for guaranteeing that the solution to a soft-constrained quadratic program is equal to the solution to the original hard-constrained, multi-parametric quadratic program (mp-QP) over a subset of the latter problem's feasible set. This allows one to soften the constraints of an MPC problem, guarantee constraint satisfaction if possible, but also guarantee that the problem will not be infeasible if constraint violation is inevitable.
- A method was described for setting up a mixed-integer optimisation problem such that the solution minimises the number of constraint violations in a prioritised-optimal fashion. It was shown how this method can be applied to the control of hybrid systems for recovering from constraint violations in an optimal fashion, while bearing the priorities of the different constraints in mind.

9.2 Directions for Future Research

Some possible directions for future research are outlined below.

Invariant Set Theory

- Efficient algorithms need to be developed for the computation of robust controllable and invariant sets for linear and nonlinear systems. The class of systems for which these sets can be computed should also be expanded, such as bilinear systems. As shown in Chapter 4 it is possible to compute these sets exactly for piecewise affine systems. However, these sets are generally non-convex and as a result the algorithms are more complex than for linear systems, where the sets are always convex. It might be possible to use the equivalent MLD model of the PWA system to develop more efficient algorithms.

- Different classes of uncertainty to those discussed in this thesis should also be investigated. The effect of uncertainty in the model for each of the regions of the PWA systems could also be included in the computation of the invariant sets.
- The feasibility of using invariant sets in the synthesis of robust controllers for piecewise affine and hybrid systems should be investigated.
- The possibility of obtaining robust performance guarantees from the use of invariant sets should be investigated. It is already known that some guarantees are possible when working with linear systems. The extension of these results to PWA systems could prove to be interesting.

Model Predictive Control

- The use of invariant sets and the robustness constraint as in Section 6.4 could provide the designer with a robust performance guarantee for the closed-loop system. More results regarding the robust stability and performance of the robustness constraint approach need to be developed.
- The case of the robust stability and feasibility of MPC with output feedback needs to be investigated. Section 6.7 briefly alluded as to how a robustly feasible output feedback MPC scheme could be designed and a more thorough investigation into this field needs to be undertaken. The simultaneous design of an MPC controller and observer might prove beneficial in enlarging the region of guaranteed robust feasibility and stability.

Constrained Optimisation

- Though the problem described in Chapter 7 of finding a lower bound on the penalty weight appears to be a difficult, non-convex optimisation problem, there does seem to be some structure in the behaviour of the Lagrange multiplier over the region of interest. It would be useful if one could determine whether the problem is quasi-convex or has some other property which could be exploited in finding a more efficient algorithm for determining a lower bound on the penalty weight.
- The choice of weights proposed in Chapter 8 is impractical for systems with many levels of prioritised constraints. It might be possible to compute an optimal set of weights. This would make the proposed approach feasible for large, complex systems.

Appendix A

Time-Varying Systems

If the system is time-varying

$$x_{k+1} = f_k(x_k, u_k, w_k) \tag{A.1}$$

and the constraints are also time varying

$$u_k \in \mathbb{U}_k \tag{A.2a}$$
$$x_k \in \mathbb{X}_k \tag{A.2b}$$
$$w_k \in \mathbb{W}_k \,. \tag{A.2c}$$

then Algorithm 2.1 requires only a minor modification [BR71, GS71].

Before proceeding, the definition of the robust one-step set is modified to account for the time-varying nature of the system and constraints:

$$\tilde{\mathcal{Q}}_k(\Omega) \triangleq \{x_k \in \mathbb{R}^n \mid \exists u_k \in \mathbb{U}_k : f_k(x_k, u_k, w_k) \in \Omega, \forall w_k \in \mathbb{W}_k\} \,. \tag{A.3}$$

Given a target set $\mathbb{T}$, one is interested in computing the set of states which can be robustly steered to $\mathbb{T}$ in a finite number of steps. Algorithm 2.1 is replaced by the following:

Algorithm A.1 (Robust controllable sets for time-varying systems). *The N-step robust controllable set $\tilde{\mathcal{K}}_N$ can be computed via the following iterative procedure:*

$$\tilde{\mathcal{K}}_0 = \mathbb{T} \tag{A.4a}$$
$$\tilde{\mathcal{K}}_{i+1} = \tilde{\mathcal{Q}}_{k+N-i-1}(\tilde{\mathcal{K}}_i) \cap \mathbb{X}_{k+N-i-1} \,. \tag{A.4b}$$

If $\tilde{\mathcal{K}}_{i+1} = \emptyset$, then terminate.

In order to steer the system to $\mathbb{T}$ in N steps, the control $u_k \in \mathbb{U}_k$ has to be chosen such that $x_{k+1} \in \tilde{\mathcal{K}}_{N-1}, \forall w_k \in \mathbb{W}_k$. At time $k+1$, the state is measured and the control $u_{k+1} \in \mathbb{U}_{k+1}$ has to be chosen such that $x_{k+2} \in \tilde{\mathcal{K}}_{N-2}, \forall w_{k+1} \in \mathbb{W}_{k+1}$. This process is repeated for N steps.

If the system has the structure

$$x_{k+1} = f_k(x_k, u_k) + g_k(w_k)\,, \tag{A.5}$$

then one can use the Pontryagin difference to compute the sequence of valid control moves which will steer the system to $\mathbb{T}$ in N steps. Assuming $x_k \in \tilde{\mathcal{K}}_N$, the sequence of controls which satisfies

$$u_{k+i}\,(x_{k+i}) \in \left\{u \in \mathbb{U}_{k+i} \mid f_{k+i}(x_{k+i}, u) \in \tilde{\mathcal{K}}_{N-i-1} \sim g_{k+i}(\mathbb{W}_{k+i})\right\}, \quad i = 0, \dots, N-1 \tag{A.6}$$

will robustly drive the system to $\mathbb{T}$ in N steps [BR71, GS71]. The set of valid control inputs at time $k+i$ is dependent on the actual measured state x_{k+i}.

As can be seen, future knowledge of the time-varying nature of the system and constraints are necessary and the valid set of controls needs to be re-computed on-line at each time step. Furthermore, once $\mathbb{T}$ is reached, one cannot guarantee that a control sequence will exist which will keep the system inside $\mathbb{T}$ unless further knowledge about the time-varying nature of the system and constraints are known. Concepts like invariance for time-varying systems therefore become a lot more complicated and fall outside the scope of this thesis.

Appendix B

Removing Redundant Constraints

This appendix describes a simple redundancy removal routine which is easy to implement. For a more efficient method, see [CMP89].

A convex polyhedron Ω described by $N \geq 2$ linear inequalities is given:

$$\Omega \triangleq \left\{\omega \in \mathbb{R}^n \mid A\omega \preceq b,\, A \in \mathbb{R}^{N\times n},\, b \in \mathbb{R}^N\right\} . \tag{B.1}$$

The problem is to remove all redundant inequalities in Ω to obtain an *irredundant* description Φ such that $\Phi = \Omega$. The i'th inequality

$$A_i'\omega \leq b_i$$

is redundant if and only if by removing it from the description of Ω the same set results, i.e.

$$\Omega = \Omega_i \triangleq \left\{\omega \in \mathbb{R}^n \mid A_j'\omega \leq b_j,\, j = 1, \dots, i-1, i+1, \dots, N\right\} .$$

Equivalently, the i'th inequality is redundant if and only if

$$\nexists \omega \in \Omega_i : A_i'\omega > b_i . \tag{B.2}$$

The irredundant polyhedron is therefore given by

$$\Phi \triangleq \bigcap_{j\in\left\{i \mid \nexists\omega\in\Omega_i : A_i'\omega > b_i\right\}} \left\{\omega \in \mathbb{R}^n \mid A_j'\omega \leq b_j\right\} . \tag{B.3}$$

Testing whether (B.2) is true can be done by finding the maximum of $A_i'\omega$ over Ω_i. A simple procedure for removing the redundant constraints can therefore be implemented as follows:

1. Set $i \leftarrow 1$, $C \leftarrow [\,]$ and $d \leftarrow [\,]$;
2. If $\max_{\omega\in\Omega_i} A_i'\omega > b_i$, then set $C \leftarrow \begin{bmatrix} C \\ A_i \end{bmatrix}$ and $d \leftarrow \begin{bmatrix} d \\ d_i \end{bmatrix}$;

3. If $i < N$ then set $i \leftarrow i + 1$ and go to step 2, else terminate;

4. The irredundant description of Ω is given by $\Phi = \{\omega \in \mathbb{R}^n \mid C\omega \preceq d\}$.

The maximisation in step 2 can be implemented as an LP. The maximisation can be terminated as soon as an ω has been found such that $A_i'\omega > b_i$.

Appendix C

Fourier-Motzkin Elimination

Fourier elimination can be thought of as the equivalent of Gaussian elimination for solving a set of linear inequalities. A brief sketch behind the idea of Fourier elimination is given here. See [KG87] for a more detailed description of the algorithm.

Let $x, y, z, \ldots, u, t$ denote some scalar variables which are required to satisfy a set of linear inequalities. The aim is to successively eliminate $x, y, z, \ldots$ from the inequalities to obtain inequalities in which only t enters.

Each of the inequalities, in relation to x is either of the form

$$x \geq A + By + Cz + \cdots \tag{C.1}$$

or

$$x \leq \alpha + \beta y + \gamma z + \cdots . \tag{C.2}$$

Each of the constraints in the form (C.1) is taken with each of the constraints in the form (C.2) to form new inequalities in which x does not appear, i.e.

$$\alpha + \beta y + \gamma z + \cdots \geq A + By + Cz + \cdots . \tag{C.3}$$

The inequalities which contained $x, y, z, \ldots, u, t$ are now replaced by those which contain only $y, z, \ldots, u, t$. This process of eliminating the variables continues until only t is present in the inequalities.

Appendix D

The Complement of the Union of a Set of Polyhedra

Given a non-convex set

$$\Omega \triangleq \bigcup_{j=1}^{N} \Omega_j$$

where each Ω_j is a closed, convex polyhedron, this appendix describes how to find the complement

$$\Omega^c \triangleq \bigcup_{i=1}^{M} \Phi_i$$

where each Φ_i is an open, convex polyhedron.

A closed, convex polyhedron can be described as the intersection of a finite number of closed half-spaces:

$$\begin{aligned}\Omega_j &\triangleq \left\{\omega \in \mathbb{R}^n \mid Q^j \omega \preceq q^j,\, Q^j \in \mathbb{R}^{L_j \times n}, q^j \in \mathbb{R}^{L_j}\right\} \\ &= \bigcap_{\ell=1}^{L_j} \left\{\omega \in \mathbb{R}^n \mid Q^j_\ell \omega \leq q^j_\ell\right\} ,\end{aligned}$$

where Q^j_ℓ is the ℓ'th row of Q^j and q^j_ℓ is the ℓ'th component of q^j.

By De Morgan's law

$$\Omega^c_j = \bigcup_{\ell=1}^{L_j} \left\{\omega \in \mathbb{R}^n \mid Q^j_\ell \omega > q^j_\ell\right\}$$

and

$$\begin{aligned}\Omega^c &= \bigcap_{j=1}^{N} \Omega_j^c \\ &= \bigcap_{j=1}^{N} \bigcup_{\ell=1}^{L_j} \left\{ \omega \in \mathbb{R}^n \mid Q_\ell^j \omega > q_\ell^j \right\} .\end{aligned}$$

Rewriting this as

$$\Omega^c = \left[\bigcup_{\ell=1}^{L_1} \{ \omega \in \mathbb{R}^n \mid Q_\ell^1 \omega > q_\ell^1 \} \right] \bigcap \cdots \bigcap \left[\bigcup_{\ell=1}^{L_N} \{ \omega \in \mathbb{R}^n \mid Q_\ell^N \omega > q_\ell^N \} \right]$$

one can proceed with the development of a systematic procedure for finding the complement by repeatedly applying the distributive law and computing the intersections. The resulting set will be the union of a finite number of open, convex polyhedra.

Example D.1. *Consider the set*

$$(A \cup B) \cap (C \cup D) ,$$

where A, B, C and D are open, convex polyhedra. By repeatedly applying the distributive law, it follows that

$$(A \cup B) \cap (C \cup D) = (A \cap C) \cup (A \cap D) \cup (B \cap C) \cup (B \cap D) .$$

The sets $A \cap C$, $A \cap D$, $B \cap C$ and $B \cap D$ are easy to represent as convex polyhedra. Each set is given by appending the strict inequalities which describe the corresponding polyhedra, as discussed in Section 3.3.1.

Example D.2. *Consider the very simple example of computing the complement of*

$$\Omega = [0, 1] \cap [2, 3] .$$

The complement is found by applying De Morgan's and the distributive laws:

$$\begin{aligned}\Omega^c &= [0, 1]^c \cup [2, 3]^c \\ &= \{(-\infty, 0) \cap (1, \infty)\} \cup \{(-\infty, 2) \cap (3, \infty)\} \\ &= \{(-\infty, 0) \cap (-\infty, 2)\} \cup \{(-\infty, 0) \cap (3, \infty)\} \cup \{(1, \infty) \cap (-\infty, 2)\} \cup \{(1, \infty) \cap (3, \infty)\} \\ &= (-\infty, 0) \cup (1, 2) \cup (3, \infty) .\end{aligned}$$

Example D.3. *Consider determining the complement of*

$$\Omega \triangleq \bigcup_{j=1}^{3} \Omega_j$$

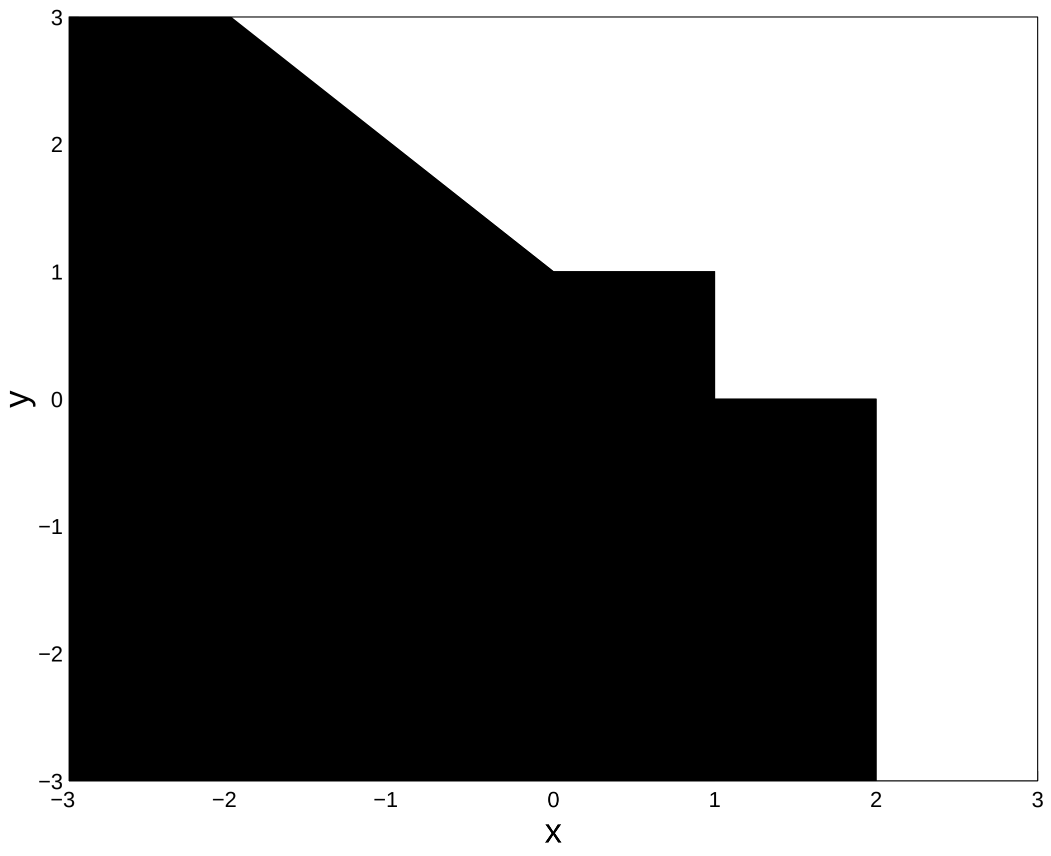

Figure D.1: The shaded area represents $\Omega = \bigcup_{j=1}^{3} \Omega_j$ of Example D.3

where

$$\Omega_1 = \{(x,y)\,|-x+y \le 1, x \le 0\}$$
$$\Omega_2 = \{(x,y)\,|x \le 1, y \le 1\}$$
$$\Omega_3 = \{(x,y)\,|x \le 2, y \le 0\}\ .$$

The set Ω is shown in Figure D.1.

The complement is found by applying De Morgan's laws:

$$\Omega^c = \bigcap_{j=1}^{3} \Omega_j^c\ ,$$

where

$$\Omega_1^c = \{(x,y)\,|x+y > 1\} \cup \{(x,y)\,|x > 0\}$$
$$\Omega_2^c = \{(x,y)\,|x > 1\} \cup \{(x,y)\,|y > 1\}$$
$$\Omega_3^c = \{(x,y)\,|x > 2\} \cup \{(x,y)\,|y > 0\}\ .$$

By applying the distributive law and forming the intersections, one gets

$$\Omega^c = \bigcup_{i=1}^{8} \Phi_i \ ,$$

where

$$\begin{aligned}
\Phi_1 &= \{(x,y)\,|x+y>1, x>1, x>2\} \\
\Phi_2 &= \{(x,y)\,|x+y>1, y>1, x>2\} \\
\Phi_3 &= \{(x,y)\,|x>0, x>1, x>2\} \\
\Phi_4 &= \{(x,y)\,|x>0, y>1, x>2\} \\
\Phi_5 &= \{(x,y)\,|x+y>1, x>1, y>0\} \\
\Phi_6 &= \{(x,y)\,|x+y>1, y>1, y>0\} \\
\Phi_7 &= \{(x,y)\,|x>0, x>1, y>0\} \\
\Phi_8 &= \{(x,y)\,|x>0, y>1, y>0\} \ .
\end{aligned}$$

It is possible to simplify this further by removing redundant inequalities. This results in

$$\begin{aligned}
\Phi_1 &= \{(x,y)\,|x+y>1, x>2\} \\
\Phi_2 &= \{(x,y)\,|y>1, x>2\} \\
\Phi_3 &= \{(x,y)\,|x>2\} \\
\Phi_4 &= \{(x,y)\,|y>1, x>2\} \\
\Phi_5 &= \{(x,y)\,|x>1, y>0\} \\
\Phi_6 &= \{(x,y)\,|x+y>1, y>1\} \\
\Phi_7 &= \{(x,y)\,|x>1, y>0\} \\
\Phi_8 &= \{(x,y)\,|x>0, y>1\} \ .
\end{aligned}$$

The number of polyhedra in the union can be reduced by testing whether the union of some of the sets is convex.

The resulting complement Ω^c, which is the union of the above Φ_i, is shown in Figure D.2. As can be seen, the above 8 sets can be reduced to 3 convex polyhedra.

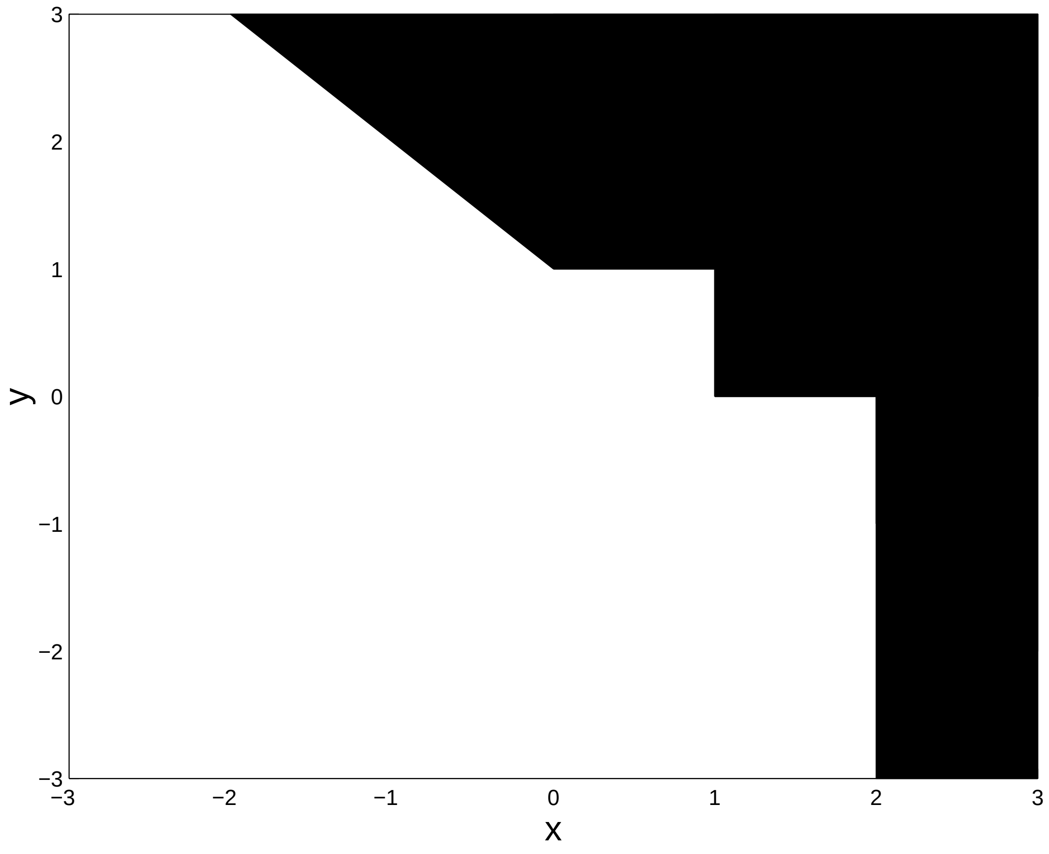

Figure D.2: The shaded area represents $\Omega^c = \bigcup_{i=1}^{8} \Phi_i$ of Example D.3

Appendix E

A Set Invariance Toolbox for LTI Systems

A Matlab toolbox has been developed for the computation of many of the sets described in Chapters 2 and 3 and can be downloaded from the author's Internet site at

http://www-control.eng.cam.ac.uk/eck21/ .

The toolbox handles LTI systems with state disturbances

$$x_{k+1} = Ax_k + Bu_k + Ew_k$$

and/or parametric (polytopic) uncertainty

$$(A, B) \in \operatorname{conv}\left\{(A_1, B_1), \dots, \left(A_p, B_p\right)\right\} .$$

If there is uncertainty in the pair (A, B) then this fact can be passed to the toolbox by stacking the vertices of the matrix polytope on top of each other[1], i.e.

$$A = \begin{bmatrix} A_1 \\ \vdots \\ A_p \end{bmatrix}, B = \begin{bmatrix} B_1 \\ \vdots \\ B_p \end{bmatrix} .$$

The main functions in the toolbox are K1SET and KINFSET for computing all the $\tilde{\mathcal{K}}_i^\lambda(\Omega, \mathbb{T})$. As shown in Chapter 2, nearly all of the sets can be found by computing the robust controllable sets with different target sets.

The basic object of the toolbox is the n-dimensional polyhedron given in augmented form

$$[C\; d] ,$$

with the function STD2AUG converting polyhedra from standard form

$$Cx \preceq d$$

[1] No uncertainty in E is assumed.

to augmented form.

The help files included with the functions are self-explanatory. Following is a list of the functions available in the toolbox.

Initialisation of Polyhedra

STD2AUG	Converts from standard to augmented form
AUG2STD	Converts from augmented to standard form
DEFINEQ	Converts lower and upper bounds on variables into a polyhedron
SYMINEQ	Converts upper bounds to a symmetric polyhedron
NORMALISE	Computes the normalised form of a polyhedron

Operations on Polyhedra

SCALESET	Scales a polyhedron
POLYMAP	Linear map
TRANSLATE	Affine translation
INTSECT	Intersection of two polyhedra
PDIFF	Pontryagin difference of two polyhedra
POLYSUM	Minkowski (vector) sum of two polyhedra
SUPPORT	Value of the support function
INEQPROJ	Projection via Fourier elimination
ISREDUNDANT	True if a linear inequality is redundant
REMRED	Removes redundant inequalities
LPSOLVER	Uses your favourite LP solver

Computation of Various Sets

REACH	Reach set
K1SET	One-step robust controllable and robust one-step set
ONESTEPAUT	One-step set of an autonomous system
KINFSET	The i-step robust controllable (contractive) sets
CINFSET	Maximal control invariant (contractive) and admissible sets
SINFSET	Maximal and i-step stabilisable (contractive) sets
OINFSET	Maximal positively invariant set
OINFSETCL	Maximal input admissible positively invariant set
OINFDIST	Maximal robust positively invariant set
OINFDISTCL	Maximal robust input-output admissible positively invariant set

Tests on Polyhedra

ISINVERTIBLE	True if a given matrix is invertible
ISILLCON	True if a given matrix is ill-conditioned
ISINSET	True if a vector is an element of a polyhedron

ISSUBSET	True if polyhedron is a subset of another
IS0ININT	True if the origin is contained in the interior
ISEMPTYSET	True if empty
ISEQUALSETS	True if two sets are equal
ISCTRLINV	True if control invariant (contractive)
ISROBCTRLINV	True if robust control invariant (contractive)
ISPOSINV	True if positively invariant
ISPOSINVCL	True if positively invariant for closed-loop system
ISROBPOSINV	True if robust positively invariant

Bibliography

[BBM98] M.S. Branicky, V.S. Borkar, and S.K. Mitter. A unified framework for hybrid control: Modal and optimal control theory. *IEEE Transactions on Automatic Control*, 1998.

[BBM00a] A. Bemporad, F. Borrelli, and M. Morari. Optimal controllers for hybrid systems: Stability and piecewise linear explicit form. In *Proceedings of the Conference on Decision and Control*, Sydney, Australia, December 2000.

[BBM00b] A. Bemporad, F. Borrelli, and M. Morari. Piecewise linear optimal controllers for hybrid systems. In *Proceedings of the American Control Conference*, Chicago, Illinois, USA, June 2000.

[BBM00c] F. Borrelli, A. Bemporad, and M. Morari. A geometric algorithm for multi-parametric linear programming. Technical Report AUT00-04, Automatic Control Laboratory, ETH, Swiss Federal Institute of Technology, ETH Zentrum - ETL I26, CH8092 Zürich, Switzerland, May 2000.

[BCM97] A. Bemporad, A. Casavola, and E. Mosca. Nonlinear control of constrained linear systems via predictive reference management. *IEEE Transactions on Automatic Control*, 42(3):340–349, March 1997.

[Bem98] A. Bemporad. Reference governor for constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 43(3):415–419, March 1998.

[Ber72] D.P. Bertsekas. Infinite-time reachability of state-space regions by using feedback control. *IEEE Transactions on Automatic Control*, AC-17(5):604–613, October 1972.

[BFM00] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, 45(10):1864–1876, October 2000.

[BFT00] A. Bemporad, K. Fukuda, and F.D. Torrisi. Convexity recognition of the union of polyhedra. Technical Report AUT00-13, Automatic Control Laboratory, ETH, Swiss Federal Institute of Technology, ETH Zentrum - ETL I26, CH8092 Zürich, Switzerland, April 2000.

[BG00] A. Bemporad and A. Garulli. Output-feedback predictive control of constrained linear systems via set-membership state estimation. *International Journal of Control*, 73(8):655–665, 2000.

[BGT00] A. Bemporad, L. Giovanardi, and F.D. Torrisi. Performance driven reachability analysis for optimal scheduling and control of hybrid systems. In *Proceedings of the Conference on Decision and Control*, Sydney, Australia, December 2000.

[Bla90] F. Blanchini. Feedback control for linear time-invariant systems with state and control bounds in the presence of disturbances. *IEEE Transactions on Automatic Control*, 35(11):1231–1234, November 1990.

[Bla94] F. Blanchini. Ultimate boundedness control for uncertain discrete-time systems via set-induced Lyapunov functions. *IEEE Transactions on Automatic Control*, 39(2):428–433, February 1994.

[Bla99] F. Blanchini. Set invariance in control. *Automatica*, 35:1747–1767, 1999.

[BM98] A. Bemporad and E. Mosca. Fulfilling hard constraints in uncertain linear systems by reference managing. *Automatica*, 34(4):451–461, 1998.

[BM99a] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–427, 1999.

[BM99b] A. Bemporad and M. Morari. Verification of hybrid systems via mathematical programming. In F.W. Vaandrager and J.H. van Schuppen, editors, *Hybrid Systems: Computation and Control*, volume 1569 of *Lecture Notes in Computer Science*. Springer-Verlag, 1999.

[BM00] F. Blanchini and S. Miani. Any domain of attraction for a linear constrained system is a tracking domain of attraction. *SIAM Journal of Control and Optimization*, 38(3):971–994, 2000.

[BMDP00a] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. Technical Report AUT99-16, Automatic Control Lab, ETH Zürich, CH-8092 Zürich, Switzerland, March 2000.

[BMDP00b] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit solution of model predictive control via multiparametric quadratic programming. In *Proceedings of the American Control Conference*, Chicago, USA, June 2000.

[BMM99] A. Bemporad, D. Mignone, and M. Morari. Moving horizon estimation for hybrid systems and fault detection. In *Proceedings of the American Control Conference*, pages 2471–2475, San Diego, CAlifornia, USA, June 1999.

[BR71] D.P. Bertsekas and I.B. Rhodes. On the minimax reachability of target sets and target tubes. *Automatica*, 7:233–247, 1971.

[BT99] V.D. Blondel and J.N. Tsitsiklis. Complexity of stability and controllability of elementary hybrid systems. *Automatica*, 35:479–489, 1999.

[BT00] V.D. Blondel and J.N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36:1249–1274, 2000.

[BTM00a] A. Bemporad, F.D. Torrisi, and M. Morari. Optimization-based verification and stability characterization of piecewise affine and hybrid systems. In B.H. Krogh and N. Lynch, editors, *Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science. Springer-Verlag, 2000.

[BTM00b] A. Bemporad, F.D. Torrisi, and M. Morari. Performance analysis of piecewise linear systems and model predictive control systems. In *Proceedings of the Conference on Decision and Control*, Syndey, Australia, December 2000. IEEE.

[BTM00c] A. Bemporad, F.D. Torrisi, and M. Morari. Verification of mixed logical dynamical models – The batch evaporator process benchmark. Technical Report AUT00-04, Automatic Control Lab, ETH Zürich, CH-8092 Zürich, Switzerland, February 2000.

[CG86] M. Cwikel and P. Gutman. Convergence of an algorithm to find maximal state constraint sets for discrete-time linear dynamical systems with bounded controls and states. *IEEE Transactions on Automatic Control*, AC-31(5):457–459, May 1986.

[CGVZ98] L. Chisci, A. Garulli, A. Vicino, and G. Zappa. Block recursive parallelotopic bounding in set membership identification. *Automatica*, 34:15–22, 1998.

[CGZ96] L. Chisci, A. Garulli, and G. Zappa. Recursive state bounding by parallelotopes. *Automatica*, 32:1049–1056, 1996.

[Chv83] V. Chvátal. *Linear Programming*. W.H. Freeman and Company, 1983.

[CM96] D. Chmielewski and V. Manousiouthakis. On constrained infinite-time linear quadratic optimal control. *Systems & Control Letters*, 29:121–129, 1996.

[CMP89] R.J. Caron, J.F. McDonald, and C.M. Ponic. A degenerate extreme point strategy for the classification of linear constraints as redundant or necessary. *Journal of Optimization Theory and Applications*, 62(2):225–237, August 1989.

[CS00] P. Caravani and E. De Santis. A polytopic game. *Automatica*, 36:973–981, 2000.

[CZ99] L. Chisci and G. Zappa. Robustifying a predictive controller against persistent disturbances. In *Proceedings of the European Control Conference*, Karlsruhe, Germany, 1999.

[CZ00a] L. Chisci and G. Zappa. Fast predictive tracking of constrained systems with bounded disturbances. In *Proceedings of the United Kingdom Automatic Control Conference*, Cambridge, UK, September 2000. UKACC, The Institution of Electrical Engineers, London, UK.

[CZ00b] L. Chisci and G. Zappa. Predictive regulation of constrained linear systems: the output feedback case. *IEEE Transactions on Automatic Control*, submitted 2000.

[CZ00c] L. Chisci and G. Zappa. Robust predictive regulation with invariance constraint. *European Journal of Control*, submitted 2000.

[DDD89] P. D'Allesandro, M. Dalla Mora, and E. De Santis. Techniques of linear programming based on the theory of convex cones. *Optimization*, 20:761–777, 1989.

[De 94] E. De Santis. On positively invariant sets for discrete-time linear systems with disturbance: An application of maximal disturbance sets. *IEEE Transactions on Automatic Control*, 39(1):245–249, January 1994.

[De 97] E. De Santis. On invariant sets for constrained discrete time linear systems with disturbances and parametric uncertainties. *Automatica*, 33(11):2033–2039, 1997.

[DH96] C.E.T. Doréa and J.C. Hennet. Computation of maximal admissible sets of constrained linear systems. Technical Report 96017, LAAS-CNRS, 7, avenue du colonel-Roche, 31077 Toulouse Cedex, France, February 1996.

[DH99] C.E.T. Doréa and J.C. Hennet. (A, B)-invariant polyhedral sets of linear discrete-time systems. *Journal of Optimization Theory and Applications*, 103(3):521–542, December 1999.

[DM69] M.C. Delfour and S.K. Mitter. Reachability of perturbed systems and min sup problems. *SIAM Journal of Control*, 7(4):521–533, 1969.

[dM00] S.L. de Oliveira Kothare and M. Morari. Contractive model predictive control for constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 2000.

[DMMS00] G. De Nicolao, L. Magnani, L. Magni, and R. Scattolini. A stabilizing receding horizon controller for nonlinear discrete time systems. In *Proceedings of the American Control Conference*, pages 270–271, Chicago IL, USA, June 2000.

[DMS00] G. De Nicolao, L. Magni, and R. Scattolini. Stability and robustness of nonlinear receding horizon control. In F. Allgöwer and A. Zheng, editors, *Nonlinear Model Predictive Control*, volume 26 of *Progress in Systems and Control Theory*, pages 23–44. Birkhäuser Verlag, P.O. Box 133, CH-4010 Basel, Switzerland, 2000.

[dOB94] N.M.C. de Oliveira and L.T. Biegler. Constraint handling and stability properties of model-predictive control. *AIChE Journal*, 40(7):1138–1155, July 1994.

[FCA00] R. Findeisen, H. Chen, and F. Allgöwer. Nonlinear predictive control for setpoint families. In *Proceedings of the American Control Conference*, Chicago, Illinois, USA, June 2000.

[FG97] I.J. Fialho and T.T. Georgiou. l_1 state-feedback control with a prescribed rate of exponential convergence. *IEEE Transactions on Automatic Control*, 42(10):1476–1481, October 1997.

[Fle87] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition, 1987.

[FMLM00] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. Identification of piecewise affine and hybrid systems. Technical Report AUT00-21, Automatic Control Lab, ETH, Swiss Federal Institute of Technology, ETH Zentrum - ETL I26, CH8092 Zürich, Switzerland, August 2000.

[FMM00] G. Ferrari-Trecate, D. Mignone, and M. Morari. Moving horizon estimation for hybrid systems. In *Proceedings of the American Control Conference*, Chicago, Illinois, USA, June 2000.

[GC87] P. Gutman and M. Cwikel. An algorithm to find maximal state constraint sets for discrete-time linear dynamical systems with bounded controls and states. *IEEE Transactions on Automatic Control*, AC-32(3):251–254, March 1987.

[GK99] E.G. Gilbert and I. Kolmanovsky. Fast reference governors for systems with state and control constraints and disturbance inputs. *International Journal of Robust and Nonlinear Control*, 9:1117–1141, 1999.

[GKT95] E.G. Gilbert, I. Kolmanovsky, and K.T. Tan. Discrete-time reference governors and the nonlinear control of systems with state and control constraints. *International Journal of Robust and Nonlinear Control*, 5:487–504, 1995.

[Grü67] B. Grünbaum. *Convex Polytopes*. John Wiley & Sons, 1967.

[GS71] J.D. Glover and F.C. Schweppe. Control of linear dynamic systems with set constrained disturbances. *IEEE Transactions on Automatic Control*, AC-16(5):411–423, October 1971.

[GS93] P. Gritzmann and B. Sturmfels. Minkowski addition of polytopes: Computational complexity and applications to Gröbner bases. *SIAM Journal of Discrete Mathematics*, 6(2):246–269, May 1993.

[GT91] E.G. Gilbert and K.T. Tan. Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9):1008–1020, September 1991.

[Hag79] W.W. Hager. Lipschitz continuity for constrained processes. *SIAM Journal of Control and Optimization*, 17(3):321–338, May 1979.

[HJ85] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[HL99] B. Heiming and J. Lunze. Definition of the three-tank benchmark problem for controller reconfiguration. In *Proceedings of the European Control Conference*, Karslruhe, Germany, September 1999.

[Hny69] E. Hnyilicza. A set-theoretic approach to state estimation. Master's thesis, Massachusetts Institute of Technology, June 1969.

[KBH00] D.E. Kassmann, T.A. Badgwell, and R.B. Hawkins. Robust steady-state target calculation for model predictive control. *AIChE Journal*, 46(5):1007–1024, May 2000.

[KBM96] M.V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10):1361–1379, 1996.

[KG87] S.S. Keerthi and E.G. Gilbert. Computation of minimum-time feedback control laws for discrete-time systems with state-control constraints. *IEEE Transactions on Automatic Control*, AC-32(5):432–435, 1987.

[KG88] S.S. Keerthi and E.G. Gilbert. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations. *Journal of Optimisation Theory and Applications*, 57(2):265–293, May 1988.

[KG95] I. Kolmanovksy and E.G. Gilbert. Maximal output admissible sets for discrete-time systems with disturbance inputs. In *Proceedings of the American Control Conference*, pages 1995–2000, 1995.

[KG98] I. Kolmanovsky and E.G. Gilbert. Theory and computation of disturbance invariant sets for discrete-time linear systems. *Mathematical Problems in Engineering: Theory, Methods and Applications*, 4:317–367, 1998.

[KM00a] E.C. Kerrigan and J.M. Maciejowski. Invariant sets for constrained nonlinear dicrete-time systems with application to feasibility in model predictive control. In *Proceedings of the Conference on Decision and Control*, Sydney, Australia, December 2000. IEEE.

[KM00b] E.C. Kerrigan and J.M. Maciejowski. Soft constraints and exact penalty functions in model predictive control. In *Proceedings of the United Kingdom Automatic Control Conference (Control 2000)*, Cambridge, UK, September 2000.

[KS90] S.S. Keerthi and K. Sridharan. Solution of parametrized linear inequalities by fourier elimination and its applications. *Journal of Optimization Theory and Applications*, 65(1):161–169, April 1990.

[LAC$^+$00] J. Lunze, J. Askari-Marnani, A. Cela, P.M. Frank, A.-L. Gehin, B. Heiming, J.M. Lemos, T. Marcu, L. Rato, and M. Staroswiecki. Three-tank control reconfiguration. In K.J. Åström, P. Albertos, M. Blanke, A. Isidori, W. Schaufelberger, and R. Sanz, editors, *Control of Complex Systems*. Springer-Verlag, 2000.

[Las86] J.B. Lasserre. Consistency of a linear system of inequalities. *Journal of Optimization Theory and Applications*, 49(1):177–179, April 1986.

[Las93] J.B. Lasserre. Reachable, controllable sets and stabilizing control of constrained linear systems. *Automatica*, 29(2):531–536, 1993.

[LPY99] G. Lafferriere, G.J. Pappas, and S. Yovine. Reachability computation for linear hybrid systems. In *Proceedings of the 14th IFAC World Congress*, Beijing, China, July 1999. IFAC.

[LTS99] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35:349–370, 1999.

[Mac01] J.M. Maciejowski. *Predictive Control with Constraints*. Addison Wesley Longman, in press 2001.

[May00] D.Q. Mayne. Nonlinear model predictive control: Challenges and opportunities. In F. Allgöwer and A. Zheng, editors, *Nonlinear Model Predictive Control*, volume 26 of *Progress in Systems and Control Theory*, pages 23–44. Birkhäuser Verlag, P.O. Box 133, CH-4010 Basel, Switzerland, 2000.

[MB76] R.J.T. Morris and R.F. Brown. Extension of validity of the GRG method in optimal control calculation. *IEEE Transactions on Automatic Control*, pages 420–422, June 1976.

[Mea94] E.S. Meadows. *Stability and Continuity of Nonlinear Model Predictive Control*. PhD thesis, The University of Texas at Austin, December 1994.

[MFM00] D. Mignone, G. Ferrari-Trecate, and M. Morari. Stability and stabilization of piecewise affine and hybrid systems: An LMI approach. In *Proceedings of the Conference on Decision and Control*, Sydney, Australia, December 2000.

[Mig99] D. Mignone. Moving horizon estimation and fault detection of mixed logic dynamical systems. Postdiploma Thesis (Nachdiplomstudium Informationstechnik), Automatic Control Laboratory, ETH Zürich, Switzerland, 1999.

[MM93] H. Michalska and D.Q. Mayne. Robust receding horizon control of constrained systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, November 1993.

[MR93] K.R. Muske and J.B. Rawlings. Model predictive control with linear models. *AIChE Journal*, 39(2):262–287, February 1993.

[MRRS00] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[MS97] D.Q. Mayne and W.R. Schroeder. Robust time-optimal control of constrained linear systems. *Automatica*, 33(12):2103–2118, December 1997.

[MSB92] T.A. Meadowcroft, G. Stephanopoulos, and C. Brosilow. The modular multivariable controller: I : Steady-state properties. *AIChE Journal*, 38(8):1254–1278, August 1992.

[Mus97] K.R. Muske. Steady-state target optimization in linear model predictive control. In *Proceedings of the American Control Conference*, Albuquerque, New Mexico, USA, June 1997.

[PN00a] J.A. Primbs and V. Nevistić. Feasibility and stability of constrained finite receding horizon control. *Automatica*, 36:965–971, 2000.

[PN00b] J.A. Primbs and V. Nevistić. A new approach to stability analysis for constrained finite receding horizon control without end constraints. *IEEE Transactions on Automatic Control*, 45(8):1507–1512, August 2000.

[Rao00] C.V. Rao. *Moving Horizon Strategies for the Constrained Monitoring and Control of Nonlinear Discrete-Time Systems*. PhD thesis, University of Wisconsin-Madison, USA, February 2000.

[RJ00] A. Rantzer and M. Johansson. Piecewise linear quadratic optimal control. *IEEE Transactions on Automatic Control*, 45(4):629–637, April 2000.

[RR99] C.V. Rao and J.B. Rawlings. Steady states and constraints in model predictive control. *AIChE Journal*, 45(6):1266–1278, June 1999.

[RWR98] C.V. Rao, S.J. Wright, and J.B. Rawlings. Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications*, 99(3):723–757, December 1998.

[Sch68] F.C. Schweppe. Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE Transactions on Automatic Control*, AC-13(1):22–28, February 1968.

[Sch73] F.C. Schweppe. *Uncertain Dynamic Systems*. Prentice Hall, Inc., 1973.

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.

[SM98] P.O.M. Scokaert and D.Q. Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, 43(8):1136–1142, August 1998.

[SMR99] P.O.M. Scokaert, D.W. Mayne, and J.B. Rawlings. Suboptimal model predictive control (Feasibility implies stability). *IEEE Transactions on Automatic Control*, 44(3):648–654, March 1999.

[SR96] P.O.M. Scokaert and J.B. Rawlings. Inifinite horizon linear quadratic control with constraints. In *Proceedings of the 13th Triennial IFAC World Congress*, pages 109–114, San Francisco, USA, 1996. IFAC.

[SR98] P.O.M. Scokaert and J.B. Rawlings. Constrained linear quadratic regulation. *IEEE Transactions on Automatic Control*, 43(8):1163–1169, August 1998.

[SR99] P.O.M. Scokaert and J.B. Rawlings. Feasibility issues in model predictive control. *AIChE Journal*, 45(8):1649–1659, August 1999.

[SRM97] P.O.M. Scokaert, J.B. Rawlings, and E.S. Meadows. Discrete-time stability with perturbations: Application to model predictive control. *Automatica*, 33(3):463–470, 1997.

[TM99] M.L. Tyler and M. Morari. Propositional logic in control and monitoring problems. *Automatica*, 35:565–582, 1999.

[TMFM01] Kazuro Tsuda, Domenico Mignone, Giancarlo Ferrari-Trecate, and Manfred Morari. Reconfiguration strategies for hybrid systems. In *Proceedings of the American Control Conference*, Arlington, VA, USA, June 2001.

[VKV$^+$] S.M. Veres, A.V. Kuntsevich, I. Vályi, D.S. Wall, S. Hermsmeyer, and S.H. Sheng. *Geometric Bounding Toolbox*. University of Birmingham, Edgbaston, UK. Information available at `http://www.eee.bham.ac.uk/gbt/`.

[VSF99] J. Vada, O. Slupphaug, and B.A. Foss. Infeasibility handling in linear MPC subject to prioritized constraints. In *Proceedings of the IFAC'99 World Congress*, Beijing, China, July 1999.

[VSJ99] J. Vada, O. Slupphaug, and T.A. Johansen. Efficient infeasibility handling in linear MPC subject to prioritized constraints. In *Proceedings of the European Control Conference*, Karlsruhe, Germany, August 1999.

[VSLS99] R. Vidal, S. Schaffert, J. Lygeros, and S. Sastry. Controlled invariance of discrete time systems. Technical Report UCB/ERL M99/65, Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720-1774, USA, December 1999.

[Wit80] H.S. Witsenhausen. Some aspects of convexity useful in information theory. *IEEE Transactions on Information Theory*, IT-26(3):265–271, May 1980.

[ZA98] A. Zheng and F. Allgöwer. Towards a practical nonlinear predictive control algorithm with guaranteed stability for large-scale systems. In *Proceedings of the American Control Conference*, pages 2534–2538, Philadelphia, Pennsylvania, June 1998.