Multi-parametric programming and explicit model predictive control of hybrid systems

by Pedro Rivotti

March, 2014

Supervised by Prof. Efstratios N. Pistikopoulos

Submitted in part fulfilment of the requirements for the degree of Doctor of Philosophy in Chemical Engineering of Imperial College London
Declaration of originality

I herewith certify that all material in this dissertation which is not my own work has been properly acknowledged.

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.
Abstract

This thesis is concerned with different topics in multi-parametric programming and explicit model predictive control, with particular emphasis on hybrid systems. The main goal is to extend the applicability of these concepts to a wider range of problems of practical interest, and to propose algorithmic solutions to challenging problems such as constrained dynamic programming of hybrid linear systems and nonlinear explicit model predictive control. The concepts of multi-parametric programming and explicit model predictive control are presented in detail, and it is shown how the solution to explicit model predictive control may be efficiently computed using a combination of multi-parametric programming and dynamic programming. A novel algorithm for constrained dynamic programming of mixed-integer linear problems is proposed and illustrated with a numerical example that arises in the context of inventory scheduling. Based on the developments on constrained dynamic programming of mixed-integer linear problems, an algorithm for explicit model predictive control of hybrid systems with linear cost function is presented. This method is further extended to the design of robust explicit controllers for hybrid linear systems for the case when uncertainty is present in the model. The final part of the thesis is concerned with developments in nonlinear explicit model predictive control. By using suitable model reduction techniques, the model captures the essential nonlinear dynamics of the system, while the achieved reduction in dimensionality allows the use of nonlinear multi-parametric programming methods.
Acknowledgements

The research presented in this thesis has been supported by the European Research Council (mobile, erc Advanced Grant, No: 226462), epsrc (ep 1014640) and the cpse Industrial Consortium.

Working at Imperial College has been a very stimulating and fruitful experience. I would like to thank my supervisor, Prof. Stratos Pistikopoulos, a passionate ambassador and pioneer of the topics explored in this thesis, for the guidance and encouragement provided during the four years of my degree. This experience was made possible by Prof. Carla Pinheiro, who introduced me to Stratos and offered friendly and valuable advice for my early career.

Throughout my stay in London, I had the pleasure of meeting interesting people from all around the world. I thank you all for your friendship and support, and hope that we stay in touch for a long time.

Finally, I would like to thank my parents and sister for their love and friendship. A very special thanks to Inês for being truly wonderful and for keeping me happy day after day.
## Contents

List of acronyms ................................................. 7  
List of tables ............................................... 8  
List of figures ................................................ 9

1 Introduction ................................................. 11
   1.1 Multi-parametric programming .............................. 12  
   1.2 Dynamic programming ...................................... 14  
   1.3 Model predictive control .................................. 16  
      1.3.1 Explicit model predictive control ................. 17  
      1.3.2 Hybrid explicit model predictive control .......... 19  
      1.3.3 Explicit robust model predictive control .......... 20  
      1.3.4 Nonlinear explicit model predictive control ...... 21  
   1.4 Thesis goals and outline .................................. 22

2 Multi-parametric programming and explicit model predictive control 25
   2.1 Fundamentals of multi-parametric linear and quadratic programming .. 26  
   2.2 Explicit model predictive control ........................ 30  
      2.2.1 Closed-loop stability ................................ 31  
      2.2.2 Choice of weights ................................... 32  
   2.3 Explicit model predictive control - a dynamic programming approach .. 33  
      2.3.1 Complexity of explicit model predictive control .... 33  
      2.3.2 Explicit model predictive control by multi-parametric programming and dynamic programming .................. 34  
   2.4 Illustrative example ...................................... 38  
      2.4.1 Solution using explicit model predictive control .... 38  
      2.4.2 Solution using dynamic programming and explicit model predictive control .......................... 39  
   2.5 Concluding remarks ..................................... 41
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Multi-parametric programming example from §1.1</td>
<td>129</td>
</tr>
<tr>
<td>B</td>
<td>Further results for example of §2.4</td>
<td>131</td>
</tr>
<tr>
<td>C</td>
<td>Further results for example of §3.3</td>
<td>133</td>
</tr>
<tr>
<td>D</td>
<td>Further results for example of §4.4</td>
<td>135</td>
</tr>
<tr>
<td>E</td>
<td>Equations and parameters for examples of §6.3</td>
<td>137</td>
</tr>
<tr>
<td>E.1</td>
<td>Distillation column with 32 states</td>
<td>137</td>
</tr>
<tr>
<td>E.2</td>
<td>cstr train with 6 states</td>
<td>138</td>
</tr>
<tr>
<td>F</td>
<td>Further results for examples of §6.3</td>
<td>141</td>
</tr>
<tr>
<td>F.1</td>
<td>Distillation column with 32 states</td>
<td>141</td>
</tr>
<tr>
<td>F.2</td>
<td>cstr train with 6 states</td>
<td>141</td>
</tr>
<tr>
<td>G</td>
<td>Meta-modelling based N-step ahead prediction</td>
<td>145</td>
</tr>
</tbody>
</table>
### List of acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSTR</td>
<td>Continuous stirred-tank reactor.</td>
</tr>
<tr>
<td>HDMR</td>
<td>High dimensional model representation.</td>
</tr>
<tr>
<td>KKT</td>
<td>Karush-Kuhn-Tucker.</td>
</tr>
<tr>
<td>LICQ</td>
<td>Linear independence constraint qualification.</td>
</tr>
<tr>
<td>LP</td>
<td>Linear programming problem.</td>
</tr>
<tr>
<td>MILP</td>
<td>Mixed-integer linear programming problem.</td>
</tr>
<tr>
<td>mp-LP</td>
<td>Multi-parametric linear problem.</td>
</tr>
<tr>
<td>mp-MILP</td>
<td>Multi-parametric mixed-integer linear problem.</td>
</tr>
<tr>
<td>mp-MINLP</td>
<td>Multi-parametric mixed-integer nonlinear problem.</td>
</tr>
<tr>
<td>mp-MIQP</td>
<td>Multi-parametric mixed-integer quadratic problem.</td>
</tr>
<tr>
<td>mp-MPC</td>
<td>Explicit model predictive control.</td>
</tr>
<tr>
<td>mp-NLP</td>
<td>Multi-parametric nonlinear problem.</td>
</tr>
<tr>
<td>mp-NMPC</td>
<td>Nonlinear model predictive control.</td>
</tr>
<tr>
<td>mp-QP</td>
<td>Multi-parametric quadratic problem.</td>
</tr>
<tr>
<td>MPC</td>
<td>Model predictive control.</td>
</tr>
<tr>
<td>NLP</td>
<td>Nonlinear programming problem.</td>
</tr>
<tr>
<td>NLSENS</td>
<td>Nonlinear sensitivity based algorithm.</td>
</tr>
<tr>
<td>NMPC</td>
<td>Nonlinear model predictive control.</td>
</tr>
<tr>
<td>ODE</td>
<td>Ordinary differential equation.</td>
</tr>
<tr>
<td>PID</td>
<td>Proportional-integral-derivative controller.</td>
</tr>
<tr>
<td>SCS</td>
<td>Strict complementary slackness.</td>
</tr>
<tr>
<td>SOSC</td>
<td>Second order sufficiency conditions.</td>
</tr>
</tbody>
</table>
## List of tables

1.1 Applications of multi-parametric programming ................................................. 12  
1.2 Review of algorithms for different classes of multi-parametric programming problems. ................................................................. 14  
1.3 Some applications of mp-MPC ........................................................................ 18  
1.4 Multi-parametric programming in the context of model predictive control. ........... 19  
1.5 Algorithms for robust multi-parametric programming according to the type of uncertainty description .............................................. 21  

A.1 Optimal solution and critical regions of multi-parametric linear problem (A.1) .......... 130  
B.1 Sample of critical regions and corresponding optimal solutions for (2.29), $N = 2$. ................................................................. 131  
B.2 Sample of critical regions and corresponding optimal solutions for (2.29), $N = 25$. ............................................................................. 132  
C.1 Critical regions and corresponding optimal solutions for example of §3.3, $i = 5$, $N = 6$. ............................................................................. 133  
C.2 Critical regions and corresponding optimal solutions for final solution of example of §3.3, $N = 6$. ................................................................. 134  
D.1 Critical regions and corresponding optimal solutions for the solution of (4.20), $N = 2$. ................................................................. 135  
D.2 Critical regions and corresponding optimal solutions for the solution of (4.20), $N = 5$. ................................................................. 136  
F.1 Critical regions for the reduced controllers with 1 state and corresponding optimal solutions. ................................................................. 141  
F.2 Example of critical regions for the reduced controllers with 2 state and corresponding optimal solutions. ................................................................. 142  
F.3 Example of critical regions for the reduced controllers with 2 state and corresponding optimal solutions. ................................................................. 143
List of figures

1.1 Two strategies for solving an optimisation problem under uncertainty – (a) grid optimisation; (b) multi-parametric programming. ........................................ 13
1.2 Principle of optimality (Bellman, 1957) for a generic multi-stage decision process. Figure adapted from (Sieniutycz, 2000). ................................. 15
1.3 Model predictive control - receding horizon scheme. ................................. 17
1.4 Framework for the development of explicit model predictive controllers. Figure adapted from (Pistikopoulos, 2009). ............................................. 19
2.1 Model predictive control as a multi-stage process with \(N\) decision stages. .... 33
2.2 Map of critical regions for the solution of (2.29) with \(N = 2\). ............... 39
2.3 Map of critical regions for the solution of (2.29) with \(N = 25\). ............... 40
2.4 Contour plot showing how the first element of the solution of (2.29) with \(N = 25\) is distributed in the parameter space. ................................. 40
2.5 State-space trajectories for different initial conditions (■) converging in the set-point (♦). ................................................................. 41
2.6 Temporal trajectories of the two components of the system state, \(x_1\) and \(x_2\). 42
2.7 Temporal trajectory of the control input of the system, \(u\). ................. 42
2.8 Comparison of computational times required for the solution of problem (2.29) using \texttt{MP-QP}, and dynamic programming based \texttt{MP-QP}. .......... 43
3.1 Schematic depiction of process with 2 stages............................................ 49
3.2 Map of critical regions for iteration \(i = 5\) of dynamic programming based algorithm with \(N = 6\). ......................................................... 54
3.3 Comparison of computational times required for the solution of problem (3.17) using \texttt{MP-MILP} and dynamic programming based \texttt{MP-MILP}. ........ 54
3.4 Algorithm profile for average between \(N = 4, 5, 6\). ............................... 55
4.1 Map of critical regions for the solution of (4.20) with \(N = 2\). ............... 70
4.2 Map of critical regions for the solution of (4.20) with \(N = 5\). ............... 71
4.3 State-space trajectories for different initial conditions (■) converging to the set-point (♦). ................................................................. 71
List of figures

4.4 Temporal trajectories of the two components of the system state, $x_1$ and $x_2$. 72
4.5 Temporal trajectory of the control input of the system, $u$. 72
4.6 Switching between affine dynamics of (4.20). 73
4.7 Comparison of computational times required for the solution of problem (4.22) using MP-MILP and dynamic programming based MP-MILP. 74

5.1 State-space trajectories for different initial conditions (■) converging to the set-point (●) or resulting in infeasible operation (×). 84
5.2 Map of critical regions for the solution of (5.33) with $N = 2$. 85
5.3 Map of critical regions for the solution of (5.33) with $N = 5$. 85
5.4 Map of critical regions for a robust controller with $N = 2$ for different values of $\gamma$: (a) $\gamma = 10\%$; (b) $\gamma = 10\%$; (c) $\gamma = 30\%$. 86
5.5 State-space trajectories for different initial conditions (■) converging to the set-point (●). 86
5.6 Temporal trajectories of the two components of the system state, $x_1$ and $x_2$. 87
5.7 Temporal trajectory of the control input of the system, $u$. 87
5.8 Switching between affine dynamics of (5.33). 88

6.1 Schematic representation of the distillation column in example 6.3.1. 96
6.2 Control inputs as a function of the states for the controllers based on the reduced model with one state. 97
6.3 Control inputs as a function of the states for the controllers based on the reduced model with two states. 98
6.4 Critical regions and system trajectory for different disturbances. 98
6.5 Closed-loop performance for reduced order online NMPC and explicit multi-parametric controllers (2 states). 99
6.6 Closed-loop performance for the reduced order controllers with one state and two states. 100
6.7 Closed-loop controller performance for disturbance rejection. 100
6.8 Schematic representation of a train of CSTRs. 101
6.9 Second component of the control input as a function of the states for reduced order controller with two states. 102
6.10 Critical regions and system trajectory for different disturbances. 102
6.11 Output trajectories of the volume in the second reactor for a disturbance of $+5\%$. 103
6.12 Output trajectories of the temperature in the second reactor for a disturbance of $+5\%$. 103
6.13 Closed-loop controller performance for disturbance rejection. 104
6.14 Closed-loop controller performance for disturbance rejection. 105
Chapter I

Introduction

The concept of optimisation has been expressed in a variety of ways, but perhaps the most eloquent and adequate is the following, attributed to Wilde and Beightler (Wilde and Beightler, 1967).

Man’s longing for perfection finds expression in the theory of optimisation.
It studies how to describe and attain what is Best, once one knows how to measure and alter what is Good or Bad.

While perfection is not always attainable, optimisation provides the mathematical tool that assists in making decisions that minimise undesired outcomes or maximise a certain quality criteria. Mathematically, optimisation corresponds to the problem of finding local or global extreme points of a function, possibly subject to a set of equality or inequality constraints.

Applications of optimisation are numerous and extend to fields of knowledge ranging from production planning, economics, resource allocation, urban planning, engineering, social sciences, and many more. Several textbooks have been devoted to the theory and practice of optimisation techniques (Luenberger, 1973; Bertsimas and Tsitsiklis, 1997; Schrijver, 1998; Winston et al., 2003; Bazaraa et al., 2013).

This thesis is concerned with the concept of multi-parametric programming which, in a way, takes optimisation one step further, by enabling the analysis of the optimal solution of an optimisation problem in face of inexact or unreliable data. The topics explored in this thesis are far from the originally intended purposes of multi-parametric programming, but still maintain the connection to the ideas developed over 50 years ago.

This introductory chapter presents a brief overview of the state of the art on multi-parametric programming, with particular emphasis on its applicability in the context of model predictive control. The concept of dynamic programming is also introduced here due to its importance in most of the developments proposed in this thesis. The shortcomings of the state of the art motivate the work on different aspects of the theory of multi-parametric programming which are outlined in the end of the chapter.
Table 1.1: Applications of multi-parametric programming.

<table>
<thead>
<tr>
<th>Application</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy and environmental analysis</td>
<td>(Pistikopoulos et al., 2007a)</td>
</tr>
<tr>
<td>Process planning</td>
<td>(Hugo and Pistikopoulos, 2005), (Li and Ierapetritou, 2007a)</td>
</tr>
<tr>
<td>Proactive scheduling</td>
<td>(Ryu et al., 2004), (Ryu et al., 2007)</td>
</tr>
<tr>
<td>Multi-stage optimisation</td>
<td>(Bard, 1983), (Vicente, 2001), (Faisca et al., 2007), (Pistikopoulos et al., 2007a)</td>
</tr>
<tr>
<td>Game theory</td>
<td>(Faisca et al., 2008)</td>
</tr>
<tr>
<td>Model predictive control</td>
<td>(Bemporad et al., 2002a), (Pistikopoulos et al., 2007b)</td>
</tr>
</tbody>
</table>

1.1 Multi-parametric programming

The first works on parametric programming date as far back as the 1950s and are often attributed to Saul Gass and Thomas Saaty (Gass and Saaty, 1955).

The idea of moving towards an optimisation strategy that encompasses variations in the objective function or constraints paved the way to many new research directions in optimisation theory. One of the earlier examples of this is attributed to Robinson and Day (Robinson and Day, 1974), who used parametric programming to study the effect of round-off errors in the solution of an optimisation problem.

With the establishment of a solid theory on parametric programming, and its extension to the general case of multi-parametric programming, its ideas became important in several fields of study. Table 1.1 presents a selection of applications in which multi-parametric programming is used.

To illustrate the concept of multi-parametric programming, consider the decision faced by a decision maker when solving an optimisation problem with two uncertain parameters, $\theta_1$ and $\theta_2$, with values in the range $\theta \in [-10, 10]$.

One possible strategy for solving the optimisation problem for the given range of parameters would be to define a grid of points in the space defined by $\theta_1$ and $\theta_2$, as illustrated in Figure 1.1a, and to solve an optimisation problem at each point in the grid. While this approach may be suitable in certain cases, two shortcomings may be identified: a) it is not clear how fine the grid should be in order to capture the most important values of the parameters; b) for the cases in which a fine grid is required, a large number of optimisation problems needs to be solved.

A more elegant solution could be obtained by using multi-parametric programming, with the two uncertain variables, $\theta_1$ and $\theta_2$, being the parameter vector. In contrast to Figure 1.1a, the solution obtained by multi-parametric programming corresponds to a map of regions in the parameter space, denoted critical regions (Figure 1.1b), where a certain solution is valid.

In contrast to the grid optimisation approach, the entire parameter space is explored by using multi-parametric programming. Another important piece of information is the region in the parameter space for which no critical region is shown in Figure 1.1b,
which corresponds to combinations of parameters that lead to infeasible solutions of the optimisation problem. The procedure used to solve multi-parametric programming problems and fully explore a given parameter space is described in Chapter 2. The results of Figure 1.1b may be replicated by solving the example shown in Appendix A.

Consider the general formulation of a multi-parametric programming problem given by (1.1).

\[
\begin{align*}
z(\theta) &= \min_{x,y} f(x, y, \theta) \\
\text{s.t. } g(x, y, \theta) &\leq 0 \\
h(x, y, \theta) &= 0 \\
\theta &\in \Theta
\end{align*}
\]  

In problem (1.1), \( z(\theta) \) is the optimal value of the cost function, \( f \), evaluated at the optimised set of decision variables which may be continuous, \( x \), or discrete, \( y \). The problem is subject to a set of inequality and equality parametric constraints, \( g \) and \( h \), respectively, which may be nonlinear.

The aim of multi-parametric programming is to solve an optimisation problem, such as (1.1) for which the outcome depends on a varying set of parameters, \( \theta \), contained in a set \( \Theta \), usually pre-defined. The equality constraints often contain the equations defining the discrete-time or continuous-time dynamics of the system under study. The set of inequalities may include physical constraints, production requirements, or any generic constraints imposed on the system.

The solution of problem (1.1) comprises (a) the optimal cost function, \( z(\theta) \), and the corresponding optimal decision variables, \( x^*(\theta) \) and \( y^*(\theta) \); (b) the map of regions in the parameter space (critical regions) for which the optimal functions are valid.
Table 1.2: Review of algorithms for different classes of multi-parametric programming problems.

<table>
<thead>
<tr>
<th>Problem class</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MP-LP</strong></td>
<td>(Gass and Saaty, 1955), (Gal and Nedoma, 1972), (Adler and Monteiro, 1992), (Dua and Pistikopoulos, 2000), (Borrelli et al., 2003), (Filippi, 2004)</td>
</tr>
<tr>
<td><strong>MP-QP</strong></td>
<td>(Dua et al., 2002), (Bemporad et al., 2002a), (Tøndel et al., 2003b), (Gupta et al., 2011), (Feller et al., 2013)</td>
</tr>
<tr>
<td><strong>MP-MILP</strong></td>
<td>(Acevedo and Pistikopoulos, 1997), (Dua and Pistikopoulos, 2000), (Li and Ierapetritou, 2007b), (Mitsos and Barton, 2009), (Wittmann-Hohlbein and Pistikopoulos, 2012b)</td>
</tr>
<tr>
<td><strong>MP-MIQP</strong></td>
<td>(Dua et al., 2002)</td>
</tr>
<tr>
<td><strong>MP-NLP</strong></td>
<td>(Kyparisis, 1987), (Fiacco and Kyparisis, 1988), (Acevedo and Salgueiro, 2003), (Bemporad and Filippi, 2006), (Grancharova and Johansen, 2006), (Domínguez et al., 2010)</td>
</tr>
<tr>
<td><strong>MP-MINLP</strong></td>
<td>(Pertsinidis et al., 1998), (Dua and Pistikopoulos, 1999), (Mitsos, 2010)</td>
</tr>
</tbody>
</table>

Depending on properties such as convexity and linearity of the functions $f$, $g$, and $h$, and the presence or not of integer variables, problem (1.1) belongs to a certain class of problems, for which specific algorithms exists.

Table 1.2 presents references to algorithms for solving typical classes of multi-parametric programming problems. The results presented in all chapters of this thesis rely on the existence of algorithms to solve classes of multi-parametric problems presented in Table 1.2.

Despite the significant amount of publications and algorithms proposed for the multi-parametric problems presented in Table 1.2, most classes of problems remain active subjects of research. Even for well established classes of problems, such as multi-parametric linear programming problems, there is continued interest in further improving the efficiency of the algorithm, reducing the complexity of exploring large-dimensional parameter spaces, or extending the approach to wider ranges of uncertainty descriptions in the cost function or constraints of the problem.

### 1.2 Dynamic programming

Multi-stage decision processes occur in different fields of study, such as energy planning (Pereira and Pinto, 1991; Pistikopoulos and Ierapetritou, 1995; Growe-Kuska et al., 2003), computational finance (Pliska, 1997; Seydel, 2012), computer science (Sakoe, 1979; Amini et al., 1990; Leiserson et al., 2001), or optimal control (Bertsekas, 1995; Fleming and Soner, 2006; Powell, 2007).

An illustration of a multi-stage decision process is shown in Figure 1.2. The structure of the problem corresponds to a block diagram in which an initial state of the system, $s_0$, undergoes a sequence of $N$ decision stages in which its value is affected by the decision variables $x_0, \cdots, x_N$. 
1.2. Dynamic programming

Dynamic programming (Bellman, 1957; Bertsekas, 1995; Sniedovich, 2010; Powell, 2007) is an optimisation theory used to efficiently obtain optimal solutions for problems involving multi-stage decision processes by exploring the sequential structure shown in Figure 1.2. Dynamic programming has been used to address problems in a variety of fields, such as process scheduling (Bomberger, 1966; Choi et al., 2004; Herroelen and Leus, 2005), optimal control (Dadebo and McAuley, 1995; Bertsekas, 1995) or robust control (Nilim and El Ghaoui, 2005; Kouramas et al., 2012).

The method is based on the principle of optimality, proposed by Bellman (Bellman, 1957), which states the equivalence of the solution obtained by dynamic programming to the solution obtained using conventional optimisation techniques. The following definition corresponds to the original formulation of the principle of optimality, given by Bellman (Bellman, 1957).

**Principle of optimality:**
An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

The principle of optimality reflects the fact that if a decision taken at stage $N$ of a multi-stage process such as Figure 1.2 is optimal, it will remain optimal regardless of the decisions taken in previous stages. Having this simple principle in mind, it is possible to transform an $N$-dimensional problem into a set of $N$ one-dimensional problems that may be solved sequentially.

The dashed boxes in Figure 1.2 illustrate this concept. The index $i$ represents the progression of an algorithm for the solution of the multi-stage problem based on a backwards dynamic programming recursion. For a certain value of $i$, the optimal decisions of future stages have been determined in previous iterations, and the task is reduced to finding the optimal value of $x_{N-i}$. After solving the iteration $i = N$, the optimal sequence of decision variables, $x_0, \ldots, x_{N-1}$, is obtained.

Despite the advantages of using dynamic programming in the context of multi-stage decision processes, its use is limited in the presence of hard constraints. In this case, at
Chapter 1. Introduction

At each stage of the dynamic programming recursion, non-linear decisions result and non-convex optimisation procedures are required to solve the dynamic programming problem (Faísca et al., 2008). Another important challenge in constrained dynamic programming is that the computation and storage requirements may significantly increase in the presence of hard constraints (Bertsekas, 1995).

To address these issues, different algorithms for constrained dynamic programming have been proposed, combining the principle of optimality of Bellman and multi-parametric programming techniques. By combining these techniques with the principle of optimality, the issues that arise for hard constrained problems are handled in a systematic way, and the shortcomings of conventional dynamic programming techniques are avoided. This method has been used to address constrained dynamic programming problems involving linear/quadratic models (Borrelli et al., 2005; Faisca et al., 2008), and mixed-integer linear/quadratic models (Borrelli et al., 2005).

The use of dynamic programming and multi-parametric programming in the context of explicit model predictive control is described in §2.3. The concept is extended for the case of constrained dynamic programming of mixed-integer linear problems in Chapter 3.

1.3 Model predictive control

Model predictive control (Maciejowski and Huzmezan, 1997; Mayne et al., 2000; Camacho and Bordons, 2004; Rawlings and Mayne, 2009) is an advanced control strategy used for the regulation of multi-variable complex plants with strict standards in terms of product specifications and safety requirements. The control problem is formulated as an optimisation problem, which results in optimality of the control inputs with respect to a certain quality criteria, while guaranteeing constraint satisfaction and inherent ability to handle a certain degree of model uncertainty and unknown disturbances (Magni and Sepulchre, 1997; Findeisen and Allgöwer, 2002).

The main concept behind model predictive control is illustrated in Figure 1.3. The optimisation problem takes place at each time instant, \( t \). The state of the system at time \( t \) is either directly measured or, more commonly, estimated based on the measured output (Lee and Ricker, 1994; Mayne et al., 2000). Using this information and the model of the system, the optimiser projects the output of the system over a specified prediction horizon and determines the optimal sequence of inputs that drive the output as close as possible to the desired reference output.

It is possible to operate a model predictive controller in an open-loop fashion, in which the sequence of optimal control inputs determined at time \( t \) is only determined once. However, in face of model uncertainty and unknown disturbances, it is more common to apply the scheme in a closed-loop manner: only the first element of the sequence of optimal inputs is applied to the system, and the optimisation procedure is repeated at time \( t + 1 \). As the optimisation is repeated at time \( t + 1 \), the prediction horizon also shifts in time, which is the reason for model predictive control often being referred to as receding horizon control.
Despite the well established benefits of using model predictive control, it has not seen a widespread adoption in industrial processes, particularly when the sampling rates of the process are fast. Part of the reason for this is related to the large amounts of legacy controllers based on PID, or other classic controller schemes, and the difficulty in training the plant personnel in the use of a control scheme as drastically different as model predictive control. Another important limitation is related to the computational requirements associated with running an online optimiser at every instance of the sampling time (Engell, 2007), despite the recent advances in fast online optimisation (Wang and Boyd, 2010).

To address these issues, the idea of explicit model predictive control was developed, combining the principles of model predictive control and multi-parametric programming. These concepts are introduced in §1.3.1.

1.3.1 Explicit model predictive control

Explicit model predictive control (Bemporad et al., 2002a; Pistikopoulos et al., 2002, 2007b) is a relatively recent concept, as testified by its absence in important survey papers, such as (Morari and Lee, 1999; Mayne et al., 2000). It has, however, been a very significant advance in control theory, and the drive behind much research both in academic topics and applications. A selection of applications reported in the literature, where explicit model predictive control is used, is presented in Table 1.3.

The idea behind explicit model predictive control is to link the theory of multi-parametric programming, presented in §1.1, and model predictive control.

As mentioned in §1.3, a closed-loop model predictive control scheme involves solving an optimisation problem whenever a sample of the system state is available. By formulating the optimisation problem as a multi-parametric programming problem such as (1.1), with the state of the system being the vector of parameters (Bemporad et al.,
Chapter 1. Introduction

Table 1.3: Some applications of mp-MPC.

<table>
<thead>
<tr>
<th>Application</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active valve train control</td>
<td>(Kosmidis et al., 2006)</td>
</tr>
<tr>
<td>Cruise control</td>
<td>(Möbus et al., 2003)</td>
</tr>
<tr>
<td>Traction control</td>
<td>(Borrelli et al., 2006)</td>
</tr>
<tr>
<td>Direct torque control of induction motors</td>
<td>(Papafotiou et al., 2007)</td>
</tr>
<tr>
<td>Biomedical drug delivery systems</td>
<td>(Dua et al., 2006; Krieger et al., 2013)</td>
</tr>
<tr>
<td>Hydrogen storage</td>
<td>(Panos et al., 2010)</td>
</tr>
<tr>
<td>Marine vessels with rudders</td>
<td>(Johansen et al., 2008)</td>
</tr>
</tbody>
</table>

(2002a), it is possible to shift the computational effort involved in online optimisation to an offline step in which the optimal solutions for every possible realisation of the state vector are pre-computed.

The use of an explicit model predictive controller as a control device consists therefore of evaluating the state of the system, at every sampling instance, and looking-up the corresponding optimal control input in the pre-computed map of critical regions. This operation is usually significantly faster than repeatedly solving optimisation problems, and therefore the method may be used for systems with more frequent sampling times.

Apart from the reduced computational costs, there is also a benefit in terms of portability. The storage requirements and the processing power required to run an explicit model predictive controller online are relatively low, and the required infrastructure is significantly lower than in the case of conventional model predictive control. This feature motivated the use of the method in applications that require the high performance standard of model predictive control to be achieved in a single chip (Dua et al., 2008).

According to a recent survey paper (Alessio and Bemporad, 2009b), the currently available tools for the design of explicit model predictive controllers (ParOS, 2004; Kvasnica et al., 2004) are suitable for applications with sampling times larger than 50ms and relatively small size (1-2 input variables and 5-10 parameters). For a study on the hardware implementation of explicit model predictive controllers see (Johansen et al., 2007).

The development of explicit model predictive controllers usually follows the work flow presented by Pistikopoulos (Pistikopoulos, 2009). A schematic representation of the framework is presented in Figure 1.4.

The need for an intermediate step that reduces the order of the high-fidelity model for which a controller is being designed arises from the limitations of multi-parametric programming algorithms in terms of the size of the problem to be solved, as mentioned above. By using model reduction and system identification techniques, a more tractable problem is defined and currently available software tools may be used to design the controller. To guarantee that this intermediate step does not affect the performance of the controller, closed-loop simulations are performed against the original high-fidelity model, and the entire design procedure repeated, if the results are found to be unsatisfactory.

The benefits of using explicit model predictive control motivated research aimed at
Hybrid explicit model predictive control

Hybrid systems correspond to a class of systems that are described by a combination of continuous variables and logical components. These logical components may result from the presence of discontinuous operating conditions of the equipment, discrete decisions related to availability of components in the system, valves and switches, or the presence of boolean decision rules, such as if-then-else statements.

Hybrid systems find relevance in most processes of practical interest (Pantelides et al., 1999; Branicky et al., 1998). Due to this importance, including integer decision variables in an explicit model predictive control framework has been identified as an
important research direction (Morari and Lee, 1999; Pistikopoulos, 2009). However, the modelling of hybrid systems results in models with integer variables (Raman and Grossmann, 1992; Williams, 1999), and therefore in the need to use computationally complex multi-parametric mixed-integer programming algorithms to design the controllers. Additionally, as pointed out by Mayne et al. (Mayne et al., 2000), many aspects of conventional model predictive control, such as stability or robustness, require especial treatment in the case of systems involving both continuous and discrete variables.

For this reason, hybrid explicit model predictive control remains an open research topic, and the available theory and algorithms are limited to only a few contributions. The problem of hybrid explicit model predictive control with a linear cost function has been addressed by Bemporad et al. (Bemporad and Morari, 1999a) and Baotic et al. (Baotic et al., 2006). Sakizlis et al. (Sakizlis et al., 2002) presented a method based on a mixed-integer quadratic programming algorithm (Dua et al., 2002) that handles quadratic cost functions.

The ability of any algorithm to convert the logical components of the hybrid system into a suitable formulation relies on the equivalence between propositional logic statements and linear constraints (Cavalier et al., 1990; Raman and Grossmann, 1991; Bemporad and Morari, 1999a). This property is explored by the mixed logical dynamical framework (Bemporad and Morari, 1999a) that provides a systematic method of converting logical propositions into a mixed-integer linear formulation.

These ideas are covered in more detail in Chapter 4, which presents a novel algorithm for hybrid model predictive control based on multi-parametric programming and dynamic programming.

### 1.3.3 Explicit robust model predictive control

As discussed in §1.3, one of the main drawbacks of an open-loop model predictive control implementation is that it assumes the absence of unknown uncertainties and model mismatch. By implementing a closed-loop formulation, in which the optimisation is repeated at each time step, it is possible to reduce the effect of these uncertainties to some extent. This property of model predictive control is referred to as inherent robustness (Mayne et al., 2000).

Despite the inherent robustness of model predictive control, possible model mismatch or external disturbances are not taken into account while optimising the control inputs, which may result in infeasible operation.

Robust model predictive control has the objective of deriving formulations that explicitly take into account uncertainties and guarantee feasible performance for a range of model variations and exterior disturbances.

Several methods for designing robust model predictive controllers have been proposed (Campo and Morari, 1987; Zafiriou, 1990; Kothare et al., 1994; Scokaert and Rawlings, 1998; Wang and Rawlings, 2004). Despite the wealth of publications on the subject, robust model predictive control remains a challenging problem and the existing methods are not at a stage of development suitable for industrial application, except in
1.3. Model predictive control

Table 1.5: Algorithms for robust multi-parametric programming according to the type of uncertainty description.

<table>
<thead>
<tr>
<th>References</th>
<th>Additive disturbances</th>
<th>Polytopic uncertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Bemporad et al., 2003)</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>(Grancharova and Johansen, 2003)</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>(Kerrigan and Maciejowski, 2003)</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>(Sakizlis et al., 2004)</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>(Alamo et al., 2005)</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>(Manthanwar et al., 2005b)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(de la Peña et al., 2005)</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>(de la Peña et al., 2007)</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>(Pistikopoulos et al., 2009)</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>(Kouramas et al., 2012)</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

very specific cases (Bemporad and Morari, 1999b). For a review of the theory and algorithms for robust model predictive control see (Bemporad and Morari, 1999b; Rawlings and Mayne, 2009).

The extension of this methodology to robust explicit model predictive controllers involves further challenges and only recently began to attract the attention of the research community. A selection of publications on this subject is classified in Table 1.5 according to the type of uncertainty description addressed. These types of uncertainty description are presented in more detail in §5.1.

Despite these efforts, many issues and areas of robust explicit model predictive control remain to be addressed (Pistikopoulos, 2009). One particular aspect in which theory is lacking is the extension of explicit robust control methods to the challenging problem of hybrid explicit model predictive control, for which very few attempts have been published in the literature (Manthanwar et al., 2005b).

Another limitation of conventional robust model predictive control techniques is that the optimisation is performed considering an open-loop control formulation, despite the fact that only the first control input is implemented in the system. The information about past uncertainty values is not taken into account in the optimisation problem, resulting in poor performance of the controller (Lee and Yu, 1997). To overcome this limitation, closed-loop formulations based on dynamic programming have been proposed (Lee and Yu, 1997; Bemporad et al., 2003).

Chapter 5 describes a novel algorithm proposed for explicit robust model predictive control of hybrid systems with linear cost function, based on multi-parametric programming and dynamic programming.

1.3.4 Nonlinear explicit model predictive control

The extension of explicit model predictive control to systems described by nonlinear dynamics is of especial importance (Biegler and Rawlings, 1991). For these systems, the use of online model predictive control is particularly challenging, since the time required
to compute the solution of the underlying optimisation problem may be significantly larger than the sampling time (Findeisen et al., 2007).

However, the theoretical basis required for applying multi-parametric programming to nonlinear model predictive control is far from being well established (Domínguez et al., 2010). One of the reasons for this is that, as mentioned in §1.3.1, the aim of explicit model predictive control is to determine the complete map of optimal control actions for every possible realisation of the system state. While this may be achieved for linear systems, in the case of nonlinear systems only an approximation of the map of optimal control inputs may be expected to be obtained. The choice of a general approximate algorithm for such task is not simple, since the type of nonlinearities in the model varies from system to system.

By considering different approximation methods, several algorithms for approximate nonlinear explicit model predictive control have been proposed in the literature (Johansen, 2002, 2004; Sakizlis et al., 2007; Domínguez et al., 2010). For an overview and comparison of these algorithms see (Domínguez and Pistikopoulos, 2011).

An additional challenge in designing nonlinear explicit model predictive controllers is due to the limitations of multi-parametric programming known for systems with high dimensionality (Alessio and Bemporad, 2009a), which are especially relevant in the case of nonlinear systems.

In Chapter 6, a method is presented that combines model approximation techniques and nonlinear multi-parametric programming algorithms to derive explicit controllers for nonlinear systems. One of the key challenges in this aspect is to select a model approximation methodology that effectively reduces the dimensionality of the model, but keeps track of the main nonlinear dynamics of the system.

1.4 Thesis goals and outline

The fundamental concepts of multi-parametric programming for linear and quadratic problems are presented in Chapter 2. The chapter begins with the presentation of a general multi-parametric programming problem, for which the optimality conditions are derived. By using local sensitivity analysis results, it is shown how, under certain assumptions, the optimal solution of the general problem may be expressed as a piece-wise affine function of the varying parameters. It is also shown how the solution of model predictive control problems may be obtained by recasting the optimal control formulation as a multi-parametric programming problem where the initial state of the system is the vector of parameters. The chapter concludes with the presentation of a methodology for the solution of explicit model predictive control problems that combines multi-parametric programming and dynamic programming. An illustrative example demonstrates the benefits of using the approach based on multi-parametric programming and dynamic programming, as opposed to conventional methods for explicit model predictive control.

Chapter 3 addresses the topic of constrained dynamic programming for problems involving multi-stage mixed-integer linear formulations with a linear objective function.
It is shown that such problems may be decomposed into a series of multi-parametric mixed-integer linear problems, of lower dimensionality, that are sequentially solved to obtain the globally optimal solution of the original problem. At each stage, the dynamic programming recursion is reformulated as a convex multi-parametric programming problem, therefore avoiding the need for global optimisation that usually arises in hard constrained problems. The proposed algorithm is applied to a problem of mixed-integer linear nature that arises in the context of inventory scheduling. The example also highlights how the complexity of the original problem is reduced by using dynamic programming and multi-parametric programming.

Based on the developments of Chapter 3, an algorithm for explicit model predictive control of hybrid linear systems is presented in Chapter 4. The proposed method employs multi-parametric and dynamic programming techniques to disassemble the original model predictive control formulation into a set of smaller problems, which can be efficiently solved using suitable multi-parametric mixed integer programming algorithms. The proposed developments are demonstrated with an example of the optimal control of a piece-wise affine system with a linear cost function.

Chapter 5 builds on the methodology presented in Chapter 4 and extends it to the problem of explicit robust model predictive control of hybrid systems where uncertainty is present in the model. To immunise the explicit controller against uncertainty, the constraints are reformulated taking into account the worst-case realisation of the uncertainty in the model, while the objective function is considered to have its nominal value. It is shown how the reformulation leads to an explicit hybrid model predictive control problem that may be solved using the methods proposed in Chapter 4.

Chapter 6 presents a methodology to derive explicit multi-parametric controllers for nonlinear systems, by combining model approximation techniques and multi-parametric model predictive control. Particular emphasis is given to an approach that applies a nonlinear model reduction technique, based on balancing of empirical gramians, which generates a reduced order model suitable for explicit nonlinear model predictive control algorithms. This approach is compared with a recently proposed method that uses a meta-modelling based model approximation technique which can be directly combined with standard multi-parametric programming algorithms. The methodology is illustrated for two nonlinear models, of a distillation column and a train of CSTRs, respectively.

Chapter 7 presents a summary of the main developments presented in this thesis, and indications of future research directions in the topics of nonlinear explicit model predictive control, robust explicit model predictive control, and constrained dynamic programming of hybrid systems.
The goals of this thesis are summarised as follows.

- Provide a qualitative discussion of the benefits in terms of computational time of a methodology for the solution of constrained multi-stage optimisation problems by dynamic programming and multi-parametric programming.

- Propose an algorithm for constrained dynamic programming for problems involving multi-stage mixed-integer linear formulations and a linear objective function.

- Demonstrate by means of an illustrative example the computational benefits of using the proposed algorithm for constrained dynamic programming of mixed-integer linear problems.

- Apply the developments proposed for constrained dynamic programming of hybrid linear problems as the basis for an algorithm for explicit model predictive control of hybrid systems with linear cost function.

- Present an algorithm, based on the proposed developments in constrained dynamic programming and explicit model predictive control, for robust explicit model predictive control of hybrid systems in the case where the model dynamics are affected by worst-case type of uncertainty.

- Develop an algorithm for explicit nonlinear model predictive control by combining multi-parametric programming and model approximation techniques. In this context, compare the use of a nonlinear model reduction technique with a meta-modelling based model approximation technique.
This chapter presents fundamental concepts of multi-parametric programming and explicit model predictive control. A general formulation of a multi-parametric problem is shown in §2.1 and it is shown how sensitivity analysis results may be used to derive the explicit solution for the particular case of linear or quadratic multi-parametric programming.

In §2.2, a procedure is presented for reformulating a model predictive control problem with quadratic objective function as a multi-parametric programming problem for which the explicit solution is obtained. Some properties of model predictive control, such as stability and importance of the choice of weights in the objective function, are also discussed in this section.

In §2.3 it is shown how explicit model predictive controllers may be efficiently designed using a combination of multi-parametric programming and dynamic programming.

The two approaches used to derive explicit model predictive controllers are compared in §2.4 by using an illustrative example, and conclusions are drawn regarding the computational benefits of using the approach based on dynamic programming.
2.1 Fundamentals of multi-parametric linear and quadratic programming

A general multi-parametric programming problem with a vector of continuous decision variables, \( x \in \mathbb{R}^n \), and a vector of parameters, \( (\theta \in \Theta) \in \mathbb{R}^m \), may be represented in the form (2.1).

\[
\begin{align*}
z(\theta) &= \min_x f(x, \theta) \\
\text{s. t.} & \quad g(x, \theta) \leq 0 \\
& \quad h(x, \theta) = 0 \\
& \quad \theta \in \Theta
\end{align*}
\]

In (2.1), \( z(\theta) \in \mathbb{R} \) is the optimal value of the cost function, \( f(x, \theta) \in \mathbb{R} \), evaluated at the optimal set of continuous decision variables \( x \in \mathbb{R}^n \). The problem is subject to a set of inequality and equality parametric constraints, \( g(x, \theta) \in \mathbb{R}^p \) and \( h(x, \theta) \in \mathbb{R}^q \), respectively.

In the remaining of this section, it is shown how the solution of problem (2.1) may be computed, under certain conditions, using principles of sensitivity analysis. The two components that define the solution of (2.1) are:

(a) The explicit expressions of optimal cost function, \( z^*(\theta) \), and the corresponding optimal decision variables, \( x^*(\theta) \);

(b) The map of regions in the parameter space (critical regions) for which the optimal functions are valid.

The procedure for solving problem (2.1) is based on the principles of local sensitivity analysis and parametric nonlinear programming. The general idea of the procedure is to derive the optimality conditions of (2.1) and analyse how these are affected by perturbations in the parameter vector.

The Lagrangian function of problem (2.1), \( L(x, \theta, \lambda, \mu) \) is defined as (2.2).

\[
L(x, \theta, \lambda, \mu) = f(x, \theta) + \sum_{i=1}^{p} \lambda_i^T g_i(x, \theta) + \sum_{j=1}^{q} \mu_j h_j(x, \theta)
\]

The first order Karush-Kuhn-Tucker optimality conditions (Bazaraa et al., 2013) for problem (2.1) have the form of (2.3).

\[
\begin{align*}
\nabla_x L(x, \theta, \lambda, \mu) &= 0 \\
\lambda_i g_i(x, \theta) &= 0, \quad \forall i = 1, \ldots, p \\
h_j(x, \theta) &= 0, \quad \forall j = 1, \ldots, q \\
\lambda_i &\geq 0 \\
g_i(x, \theta) &\leq 0
\end{align*}
\]
2.1. Fundamentals of multi-parametric linear and quadratic programming

The vectors \( \lambda_i \) and \( \mu_j \) in (2.3) correspond to the Lagrange multipliers of the inequality and equality constraints, respectively.

Under certain assumptions, the optimality conditions of the general problem (2.1) may be tracked in the neighbourhood of a certain parameter realisation, \( \theta_0 \), providing an explicit function of the optimizer, \( x(\theta) \), and the Lagrangian multipliers, \( \lambda(\theta) \) and \( \mu(\theta) \), as a function of the parameters. The existence of this function is ensured by Theorem 1.

**Theorem 1. Local Sensitivity Theorem (Fiacco, 1976)**

Let \( \theta_0 \) be a particular parameter realisation of (2.1) and \( \eta = [x_0, \lambda_0, \mu_0]^T \) the solution of (2.3).

Under the following assumptions:

1. **Assumption 1.** Strict complementary slackness (scs) (Tucker, 1956).
2. **Assumption 2.** Linear independence constraint qualification (licq).
3. **Assumption 3.** Second order sufficiency condition (sosc).

In the neighbourhood of \( \theta_0 \), there exist unique and once continuously differentiable functions \( x(\theta) \), \( \lambda(\theta) \) and \( \mu(\theta) \).

Moreover, the jacobian of system (2.3) is defined by matrices \( M_0 \) and \( N_0 \), such as:

\[
M_0 = \begin{bmatrix}
\nabla^2_{xL} & \nabla_x g_1 & \ldots & \nabla_x g_p & \nabla_x h_1 & \ldots & \nabla_x h_q \\
-\lambda_1 \nabla^T_{x} g_1 & -V_1 \\
\vdots & \ddots & 0 \\
-\lambda_p \nabla^T_{x} g_p & -V_p \\
\nabla^T_{x} h_1 \\
\vdots \\
\nabla^T_{x} h_q
\end{bmatrix}
\]

(2.4)

where \( V_i = g_i(\theta_0) \),

\[
N_0 = [\nabla^2_{xL} \nabla^T_{x} g_1, \ldots, -\lambda_p \nabla^T_{x} g_p, \nabla^T_{x} h_1, \ldots, \nabla^T_{x} h_q]^T
\]

(2.5)

The following corollary shows that the explicit parametric expressions mentioned in Theorem 1, \( x(\theta) \), \( \lambda(\theta) \) and \( \mu(\theta) \), are piece-wise affine functions of the parameter \( \theta \).

**Corollary 1. First-order estimation of \( x(\theta) \), \( \lambda(\theta) \) and \( \mu(\theta) \) in the neighbourhood of \( \theta_0 \) (Fiacco, 1976).**

Under the assumptions of Theorem 1, the first-order approximation of \( x(\theta) \), \( \lambda(\theta) \) and \( \mu(\theta) \) in the neighbourhood of \( \theta_0 \) is given by:

\[
\begin{bmatrix}
x(\theta) \\
\lambda(\theta) \\
\mu(\theta)
\end{bmatrix} =
\begin{bmatrix}
x(\theta_0) \\
\lambda(\theta_0) \\
\mu(\theta_0)
\end{bmatrix} - M_0^{-1} N_0 (\theta - \theta_0) + o(\|\theta\|) \quad (2.6)
\]

where \( o(\|\theta\|) \) is a term \( \phi(\theta) \) such that \( \lim_{\theta \to 0} \frac{\phi(\theta)}{\|\theta\|} = 0 \).
To determine the optimal expressions of $x(\theta), \lambda(\theta),$ and $\mu(\theta)$ using (2.6), it is required to compute the inverse of (2.4). The existence of such inverse is guaranteed by the assumptions of Theorem 1 (McCormick, 1976).

The results guaranteed by Theorem 1 and Corollary 1 are important in the sense that they provide means of determining the optimal values of any general problem such as (2.1) as affine expressions of the parameters. However, the need to compute matrices (2.4) and (2.5) may be avoided in the special case of multi-parametric linear or quadratic problems.

A multi-parametric quadratic problem with linear constraints may be written in the form (2.7). Note that a multi-parametric linear problem may be obtained from (2.7) by setting $Q = 0.$

$$z(\theta) = \min_x c^T x + \frac{1}{2} x^T Q x$$
$$\text{s. t. } Ax \leq b + F\theta$$
$$\theta \in \Theta$$

In problem (2.7), $c \in \mathbb{R}^n$ is the cost associated with the linear term, $Q \in \mathbb{R}^{n \times n}$ is a positive definite matrix defining the cost of the quadratic term, and $A \in \mathbb{R}^{p \times n}, b \in \mathbb{R}^p,$ and $F \in \mathbb{R}^{p \times m}$ are linear inequality coefficients.

For a problem such as (2.7), it is possible to prove that the explicit optimal function is an affine function of the parameters by writing the Karush-Kuhn-Tucker optimality conditions and performing algebraic operations. These results are given by Theorem 2.

**Theorem 2.** Explicit optimal solution of (2.7) (Dua et al., 2002)

Let $Q$ be a symmetric and positive definite matrix and the assumptions of Theorem 1 hold. Then the optimal vector, $x,$ and the Lagrange multipliers, $\lambda,$ are affine expressions of the parameter vector, $\theta,$ in a neighbourhood of $\theta_0.$

**Proof.** The first order Karush-Kuhn-Tucker conditions of (2.7) are given by:

$$c + Qx + A^T \lambda = 0$$
$$\lambda_i (A_i x - b_i - F_i \theta) = 0, \quad \forall i = 1, \ldots, p$$
$$\lambda_i \geq 0, \quad \forall i = 1, \ldots, p$$

Since $Q$ is a symmetric and positive definite matrix, it is possible to rearrange (2.8) in a form that shows $x$ as an affine function of $\lambda$:

$$x = -Q^{-1}(A^T \lambda + c)$$

Let $\tilde{\lambda}$ denote Lagrange multipliers corresponding to active inequality constraints. For active constraints the following relation shows that $x$ is an affine function of $\theta$:

$$\tilde{A} x - \tilde{b} - F \theta = 0$$
Replacing (2.11) in (2.12) we obtain:

\[- \tilde{A}Q^{-1}(A^T \tilde{\lambda} + c) - \tilde{b} - \tilde{f}\theta = 0 \]  
(2.13)

\[\tilde{\lambda} = -(\tilde{A}Q^{-1}\tilde{A}^T)^{-1} \tilde{f}\theta - (\tilde{A}Q^{-1}\tilde{A}^T)^{-1}(\tilde{A}Q^{-1}c + \tilde{b})\]  
(2.14)

Equation (2.14) shows the affine relation between \(\lambda\) and \(\theta\). Note that the existence of the term \((\tilde{A}Q^{-1}\tilde{A}^T)^{-1}\) is guaranteed by the assumption that the rows of \(\tilde{A}\) are linearly independent and \(Q\) is a positive definite matrix.

**Remark 1.** It is possible to obtain a solution for (2.7) in the case when \(Q\) is a positive semi-definite matrix. Algorithms that handle this case, referred to as dual degeneracy, are discussed in (Tondel et al., 2003).

As mentioned above, the affine expressions obtained using (2.6), or (2.12) and (2.14), are valid in a neighbourhood of \(\theta_0\). To obtain the region in the parameter space (critical region, CR), where each affine expression is valid, feasibility and optimality conditions are enforced (Dua et al., 2002; Bemporad et al., 2002a) as shown in (2.15).

\[\text{CR} = \{\theta \mid \tilde{g}(x(\theta), \theta) \leq 0, h(x(\theta), \theta) = 0, \tilde{\lambda}(\theta) \geq 0, \text{CR}_I\}\]  
(2.15)

In (2.15), \(\tilde{g}(x(\theta), \theta)\) corresponds to the inactive inequality constraints of (2.1), \(\tilde{\lambda}(\theta)\) are the Lagrange multipliers corresponding to the active inequality constraints and \(\text{CR}_I\) corresponds to a user-defined initial region in the parameter space that is to be explored.

Having defined the critical region in which the affine expressions are valid, a strategy is required to fully explore the pre-defined parameter space, \(\text{CR}_I\), in order to obtain a complete map, such as in Figure 1.1b.

Dua et al. (Dua et al., 2002) proposed an algorithm that geometrically partitions the parameter space and recursively explores the newly defined partitions until the entire space is explored. This approach is usually preferred to sub-optimal methods (Johansen et al., 2000), or methods that are applicable for problems with constraints only in the decision variables (Seron et al., 2000).

A different approach to exploring the entire parameter space, \(\text{CR}_I\), has recently been suggested, motivated by the exponential increase in computational complexity observed for parameter spaces of large dimensionality (Gupta et al., 2011; Feller et al., 2012). The idea behind this method is to enumerate all possible combinations of active constraints of (2.1) and directly compute the explicit solutions using the relations (2.6), or (2.12) and (2.14). This method was shown to be effective for problems with large dimensionality of the parameter space, but small number of constraints.

Regardless of the procedure used to partition the parameter space, the final map of critical regions is identical, given the uniqueness of the optimal piece-wise affine solution, guaranteed by the convexity and continuity of (2.7) (Fiacco and Ishizuka, 1990; Dua et al., 2002). To achieve this minimal representation, software solutions for multi-parametric
programming (ParOS, 2004; Kvasnica et al., 2004) usually include a post-processing step that merges partitions with the same optimal solution into a convex critical region.

Despite the usefulness of the theory presented in this section, it should be noted that the first order approximation is made due to its practical usefulness, even though a piece-wise affine form is not the natural representation of the solution of (2.7). One possible consequence of this is that the neighbourhoods in which the affine expressions are valid may be narrow, resulting in a large number of critical regions in the final map. For implementation purposes, having a large number of critical regions may imply that the time required to retrieve the optimal solution from the map of critical regions is larger than the sampling time of the system.

To address the issue of point location in highly partitioned maps of critical regions, a method presented in the literature (Tøndel et al., 2003a) proposed the computation of a binary search tree that allows the efficient retrieval of the optimal solution.

2.2 Explicit model predictive control

Consider a discrete-time linear system described by the dynamic equations (2.16).

\[
\begin{align*}
    x_{k+1} &= Ax_k + Bu_k \\
    y_k &= Cx_k + Du_k
\end{align*}
\] (2.16)

The index \( k \) represents the temporal coordinate of the state vector, \( x_k \in \mathbb{R}^m \), input vector, \( u_k \in \mathbb{R}^n \), and output vector, \( y_k \in \mathbb{R}^m \). In this section, the coefficients \( A \in \mathbb{R}^{p \times m} \), \( B \in \mathbb{R}^{p \times n} \), \( C \in \mathbb{R}^{p_y \times m} \), and \( D \in \mathbb{R}^{p_y \times n} \) are assumed to be time invariant and unaffected by uncertainty. A discussion of explicit model predictive control where the model is unreliable is presented in Chapter 5.

A model predictive control problem is an optimisation formulation used to design a controller based on a dynamical model such as (2.16). The problem of regulating (2.16) to the origin, \( x = 0 \), subject to constraints in the input and state vectors and using a quadratic cost function, is described as (2.17).

\[
    U = \arg\min_{u_0, \ldots, u_{N-1}} \|x_N\|_P^2 + \|u_{N-1}\|_R^2 + \sum_{i=1}^{N-1} \|x_i\|_Q^2 + \|u_{i-1}\|_R^2
\]

s. t. \[
    \begin{align*}
    x_{k+1} &= Ax_k + Bu_k, & k = 0, \ldots, N - 1 \\
    x_{\min} &\leq x_k \leq x_{\max}, & k = 1, \ldots, N \\
    u_{\min} &\leq u_k \leq u_{\max}, & k = 0, \ldots, N - 1
    \end{align*}
\] (2.17)

In (2.17), \( \|a\|_X^2 \) is the square of the Euclidean norm, \( (a^T X a)^{\frac{1}{2}} \), and \( N \) corresponds to the output horizon, which for simplicity is assumed to be equal to the control horizon.

The strategy to reformulate (2.17) as an explicit model predictive control problem involves re-writing (2.17) as a multi-parametric quadratic problem, such as (2.7), with the parameter vector corresponding to the initial state of system, \( x_0 \) (Pistikopoulos et al., 2000; Bemporad et al., 2002a).
The reformulation of (2.17) as a multi-parametric quadratic problem is given by (2.18).

\[
U(\theta) = \arg\min_U \frac{1}{2} U^T H U + \theta^T F U
\]

s.t. \( X_{\text{min}} \leq \hat{A}\theta + \hat{B}U \leq X_{\text{max}} \)

\( U_{\text{min}} \leq U \leq U_{\text{max}} \)

\( \theta = x_0 \)

\( F = 2\hat{A}^T \hat{Q}\hat{B}, \quad H = 2(\hat{B}^T \hat{Q}\hat{B} + \hat{R}) \)

\[
\hat{A} = \begin{bmatrix}
A \\
A^2 \\
A^3 \\
\vdots \\
A^N
\end{bmatrix}, \quad \hat{B} = \begin{bmatrix}
B & 0 \cdot B & \cdots & 0 \cdot B \\
AB & B & \cdots & 0 \cdot B \\
A^2B & AB & B & \cdots & 0 \cdot B \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
A^{N-1}B & A^{N-2}B & \cdots & AB & B
\end{bmatrix}
\]

\[
\hat{Q} = \begin{bmatrix}
Q & 0 \cdot Q & \cdots & 0 \cdot Q \\
0 \cdot Q & Q & \cdots & 0 \cdot Q \\
\vdots & \vdots & \ddots & 0 \cdot Q \\
0 \cdot Q & 0 \cdot Q & \cdots & P
\end{bmatrix}, \quad \hat{R} = \begin{bmatrix}
0 \cdot R & 0 \cdot R & \cdots & 0 \cdot R \\
0 \cdot R & 0 \cdot R & \cdots & \vdots \\
\vdots & \vdots & \ddots & \vdots \\
0 \cdot R & \cdots & \cdots & R
\end{bmatrix}
\]

\( X = [x_1 \ x_2 \ \cdots \ x_N]^T, \quad U = [u_0, u_1, \cdots, u_{N-1}]^T \)

Note that the term \( \theta^T F U \) in the objective function of (2.18) may be handled in a multi-parametric quadratic programming formulation by introducing the change of variable \( U = Z - H^{-1}F^T \theta \) (Dua et al., 2002), and taking \( Z \) as the new optimisation variable.

Problem (2.18) is in the form of the multi-parametric quadratic programming formulation (2.7) and may be solved using the method presented in §2.1.

One of the advantages of model predictive control is the ability to handle generic constraints. Additional constraints, specific to the application being considered, may be easily added to the formulation (2.18) without loss of generality.

The reformulation presented in this section may be adapted in a straightforward way to different model predictive control strategies, such as reference output tracking, constraint softening, or to include penalties to the rate of change in the input vector (Bemporad et al., 2002a).

### 2.2.1 Closed-loop stability

Stability is the study of the convergence of the state of the system to the origin and is concerned with providing results that guarantee such convergence under certain conditions.
Since the explicit solution of (2.17), obtained using multi-parametric programming, is an exact solution, well established stability results for model predictive control are inherited in the case of explicit model predictive control.

There are different methods of guaranteeing closed-loop stability for constrained receding horizon model predictive control problems.

When the system is open-loop stable, linear, and with convex control constraints, stability may be guaranteed by choosing the terminal weight in (2.17), $P$, as the solution of the Lyapunov equation (Rawlings and Muske, 1993).

Another possibility is to introduce an end-constraint that forces the state to the origin at the end of the control horizon (Kwon and Pearson, 1977; Kwon et al., 1983), which guarantees stability in a straightforward way, but may cause infeasible solutions for low values of the control horizon. A more common approach is to define a region, for example the maximal constraint admissible set (Gilbert and Tan, 1991), and to replace the terminal equality constraint by an inequality that forces the final state to lie in such region (Michalska and Mayne, 1993).

In cases for which it is computationally undesirable to include a terminal constraint, or a terminal set constraint, it is possible to introduce a stabilizing feedback based on the solution of the infinite-horizon quadratic regulator and choose an output horizon large enough to guarantee stability (Chmielewski and Manousiouthakis, 1996; Scokaert and Rawlings, 1998).

A discussion of different approaches to stability of closed-loop systems may be found in the model predictive control literature survey papers (Garcia et al., 1989; Morari and Lee, 1999; Mayne et al., 2000).

### 2.2.2 Choice of weights

The choice of matrices $Q$ and $R$ in (2.17), reflects the relative penalties attributed to deviations in the state of the system and high magnitude of inputs actions, respectively, and is part of the procedure referred to as the tuning of model predictive control. Despite the importance of such procedure, the strategies used for tuning model predictive are often based on heuristics or trial and error, and robust performance is not guaranteed even for the case of unconstrained systems (Rowe and Maciejowski, 2000).

For practical purposes, the most common strategy is to tune the control horizon and terminal weight, $P$, for stability purposes, as discussed in §2.2.1, and leave the task of fine-tuning parameters $Q$ and $R$ to the control engineers who are familiar with the production requirements (Dua et al., 2002; Garriga and Soroush, 2010).

There have been, nevertheless, some attempts at providing systematic rules for tuning the objective parameters (Rustem et al., 1978; Lee and Ricker, 1994; Rustem, 1998; Rowe and Maciejowski, 1999; Trierweiler and Farina, 2003; Chmielewski and Manthanwar, 2004; Baric et al., 2005).

For an overview of the literature on tuning different model predictive control parameters, see the recent survey paper (Garriga and Soroush, 2010).
2.3. Explicit model predictive control - a dynamic programming approach

2.3.1. Complexity of explicit model predictive control

The complexity of the method used to derive explicit model predictive control by multi-parametric programming, presented in §2.2, has been analysed in (Pistikopoulos et al., 2007b). It is remarked that the upper bound on the number of critical regions comprising the map of optimal solutions is given by expression (2.19), where \( \eta \) is given by (2.20).

\[
\#CR \leq \sum_{k=0}^{\eta-1} k!(m_M + n_g)^k \tag{2.19}
\]

\[
\eta = \sum_{i=0}^{N-m} \frac{N(m_M + n_g)!}{(N \cdot m_M + N \cdot m_g - i)!i!} \tag{2.20}
\]

In (2.19) and (2.20), \( m_M \) is the number of input constraints, \( n_g \) is the number of state constraints, and \( N \) is the control horizon.

It may be observed from (2.19) and (2.20) that the upper bound in the number of critical regions is influenced by the control horizon of the problem. While the maximum number of critical regions is not an exact measure of the complexity of the algorithm, these relations hint that the computational time involved in solving the multi-parametric programming problem increases significantly with the increase of the control horizon. This is an important limitation of the method because, as discussed in §2.2.1, it is common practice to design controllers with relatively large control and output horizons, in order to guarantee stability properties.

These issues have been addressed in the literature by considering approaches that combine multi-parametric programming and dynamic programming for the design of explicit model predictive controllers.

Dynamic programming, introduced in §1.2, is a method that finds relevance in the context of multi-stage decision processes. These processes involve a special structure in which the state of the system is affected by decisions taken sequentially, and therefore large problems may be decomposed into a set of sub-problems of smaller dimensionality.

Model predictive control is a multi-stage process in the sense that each instance of the sampling time may be interpreted as a decision stage and that these decisions are taken sequentially. A diagram that illustrates this structure is presented in Figure 2.1.
In the following section it is shown how explicit model predictive control problems written in a recursive form may be solved using principles of dynamic programming.

### 2.3.2 Explicit model predictive control by multi-parametric programming and dynamic programming

Consider the model predictive control problem (2.21).

\[
V(x_0) = \min_{u_0,\ldots,u_{N-1}} \|x_N\|^2_P + \|u_{N-1}\|^2_R + \sum_{i=1}^{N-1} \|x_i\|^2_Q + \|u_{i-1}\|^2_R \\
\text{s.t. } x_{k+1} = Ax_k + Bu_k, \quad k = 0,\ldots,N-1 \\
x_{\text{min}} \leq x_k \leq x_{\text{max}}, \quad k = 1,\ldots,N \\
u_{\text{min}} \leq u_k \leq u_{\text{max}}, \quad k = 0,\ldots,N-1
\]  

(2.21)

By noting that (2.21) is a multi-stage decision process and applying the principle of optimality, (2.21) may be re-written in the recursive form (2.22) (Kouramas et al., 2011; Bertsekas, 1995).

\[
V_i(x_{i-1}) = \min_{u_{i-1}} \begin{cases} 
\|x_i\|^2_Q + \|u_{i-1}\|^2_R + V_{i+1}(x_i) & \text{if } i < N \\
\|x_N\|^2_P + \|u_{N-1}\|^2_R & \text{if } i = N 
\end{cases} \\
\text{s.t. } x_i = Ax_{i-1} + Bu_{i-1} \\
x_{\text{min}} \leq x_i \leq x_{\text{max}} \\
u_{\text{min}} \leq u_{i-1} \leq u_{\text{max}}
\]  

(2.22)

Note that problem (2.22) is a sub-problem of (2.21) with decision variables and constraints related only to stage \(i\). To fully solve the recursion, (2.22) is solved for \(i = N, N-1, \ldots, 1\), and the final solution corresponds to the global optimal solution of (2.21) (Bellman, 1957).

The reformulation of (2.22) as a multi-parametric programming problem has been suggested in the literature using different methods (Munoz de la Pena et al., 2004; Borrelli et al., 2005; Faísca et al., 2008; Kouramas et al., 2011).

One possible alternative is to reformulate (2.22) as a multi-parametric problem with \(x_{i-1}\) being the vector of parameters and to introduce the explicit expression of \(V_{i+1}(x_i)\), obtained at iteration \(i + 1\), in the objective function (Munoz de la Pena et al., 2004; Borrelli et al., 2005). However, \(V_{i+1}(x_i)\) is defined as a piece-wise quadratic function, which results in a nonlinear formulation of (2.21). The authors propose a solution for this issue, consisting of solving multiple multi-parametric quadratic programming problems, over each partition in which \(V_{i+1}(x_i)\) is defined. As noted in (Faísca et al., 2008), this method requires a procedure to combine the solutions obtained for each partition which involves comparing quadratic objective functions, and therefore requires global optimisation techniques to solve.
2.3. Explicit model predictive control - a dynamic programming approach

An alternative procedure (Faisca et al., 2008; Kouramas et al., 2011) involves expanding the term $V_{i+1}(x_i)$ and including all terms related to future control inputs in the parameter vector, $\theta$, as shown in (2.23).

$$V_i(\theta) = \min_{u_{i-1}} \|x_N\|_p^2 + \|u_{N-1}\|_R^2 + \sum_{j=i}^{N-1} \|x_j\|_Q^2 + \|u_{j-1}\|_R^2$$

s. t. $x_k = Ax_{k-1} + Bu_{k-1}$, $k = i, \ldots, N$

$$x_{\text{min}} \leq x_i \leq x_{\text{max}}$$

$$u_{\text{min}} \leq u_{i-1} \leq u_{\text{max}}$$

$$\theta = \begin{bmatrix} x_{i-1} & u_i & u_{i+1} & \ldots & u_{N-1} \end{bmatrix}^T$$

(2.23)

The idea behind this reformulation is to form a recursion in which the optimal control inputs obtained in previous iterations are stored and incorporated when the solution of the current iteration is available. By considering the entire objective function of (2.21) in the formulation, a convex problem is obtained, and the need for global optimisation techniques is avoided.

To illustrate the procedure, consider iteration $i = N$ of (2.23) and the associated multi-parametric quadratic problem (2.24).

$$V_N(\theta) = \min_{u_{N-1}} \frac{1}{2} u_{N-1}^T H u_{N-1} + \theta^T F u_{N-1}$$

s. t. $x_{\text{min}} \leq A\theta + Bu_{N-1} \leq x_{\text{max}}$

$$u_{\text{min}} \leq u_{N-1} \leq u_{\text{max}}$$

$$H = 2(B^T PB + R), \quad F = 2A^T PB$$

$$\theta = x_{N-1}$$

(2.24)

The solution of (2.24) is the piece-wise affine function $u_{N-1} = f(\theta)$, defined over the map of $n_N$ critical regions, $\text{CR}_N$. The solution of iteration $i = N$ is stored and the algorithm proceeds to iteration $i = N - 1$, at which stage the multi-parametric quadratic problem to be solved is given by (2.25).
\[ V_{N-1}(\theta) = \min_{U_{N-2}} \frac{1}{2} U^T H U + \theta^T F U \]

s.t. \[ x_{\min} \leq \begin{bmatrix} A & 0 & B \end{bmatrix} \theta + B U_{N-2} \leq x_{\max} \]

\[ u_{\min} \leq u_{N-2} \leq u_{\max} \]

\[ H = 2(\hat{B}^T \hat{Q} \hat{B} + \hat{R}), \quad F = 2 \hat{A}^T \hat{Q} \hat{B} \]

\[ \hat{A} = \begin{bmatrix} A & 0 & B \\ A^2 & B \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} B \\ AB \end{bmatrix} \]

\[ \hat{Q} = \begin{bmatrix} Q & 0 & Q \\ 0 & Q & P \end{bmatrix} \]

\[ \theta = \begin{bmatrix} x_{N-2} & u_{N-1} \end{bmatrix}^T \]

(2.25)

The solution of (2.25) is given by the piece-wise affine function \( u_{N-2} = f(\theta) \), defined over the map of \( n_{N-1} \) critical regions, \( \text{CR}_{N-1} \). In order to fully solve iteration \( i = N-1 \) of the recursive procedure, it is necessary to obtain the optimal solution \( u_{N-2} = f(x_{N-2}) \). This is achieved by combining the solutions of (2.24) and (2.25).

For two critical regions taken from the sets \( \text{CR}_{N-1} \) and \( \text{CR}_N \), the corresponding optimal solutions are given by (2.26) and (2.27), respectively.

\[
\begin{align*}
  u_{N-2} &= C_1 x_{N-2} + C_2 u_{N-1} + C_3 \\
u_{N-1} &= C_4 x_{N-1} + C_5
\end{align*}
\]

(2.26)

(2.27)

The matrices \( C_1, C_2, C_3, C_4, \) and \( C_5 \), in (2.26) and (2.27) are linear coefficient of appropriate dimensions. The relation between \( x_{N-2} \) and \( x_{N-1} \) is given by the dynamic equation (2.28).

\[ x_{N-1} = A x_{N-2} + B u_{N-2} \]

(2.28)

Equations (2.26), (2.27), and (2.28) form a linear system of 3 equations and 4 variables. The solution of iteration \( i = N-1 \) corresponds to the function \( u_{N-2} = f(x_{N-2}) \) obtained by solving the system of equations. As suggested in (Kouramas et al., 2011), this operation may be performed using orthogonal projection methods or Fourier-Motzkin elimination (Schrijver, 1998).

To fully characterise the parameter space at iteration \( i = N-1 \), the procedure of combining solutions of the two stages is repeated for the \( n_N \times n_{N-1} \) combinations of critical regions. Some of the combinations will result in empty regions that may be detected by performing feasibility tests over the intersection of the two critical regions being considered.

Note that, given the convexity of \( V_N(\theta) \) and \( V_{N-1}(\theta) \), the feasible intersection of critical regions is unique, and no overlaps result in the final map of critical regions at each iteration.
In order to obtain the solution of (2.21), the recursive procedure exemplified above is repeated for \( i = N, N-1, \ldots, 1 \). The final solution of iteration \( i = 1 \) corresponds to the solution of (2.21).

The procedure presented in this section is summarised in Algorithm 1.

**Algorithm 1** Explicit model predictive control by multi-parametric programming and dynamic programming (Kouramas et al., 2011).

```plaintext
1: Solve stage \( N \) of (2.23) and obtain the solution \( u_{N-1} = f(x_{N-1}) \) defined over \( CR_N \).
2: for \( i \leftarrow N - 1, \ldots, 1 \) do
3: \( CR_{temp} \leftarrow \emptyset \)
4: Solve stage \( i \) of (2.23) and obtain solution \( u_{i-1} = f(x_{i-1}, u_i, \ldots, u_{N-1}) \) defined over \( CR_i \).
5: for \( j \leftarrow 1, \ldots, \#CR_i \) do
6: for \( k \leftarrow 1, \ldots, \#CR_{i+1} \) do
7: Test feasibility of \( CR_i^j \cap CR_{i+1}^k \).
8: if Intersection is feasible then
9: Compute stage solution by replacing \( u_{i+1} \) in \( u_i \).
10: \( CR_{temp} \leftarrow CR_{temp} \cup (CR_i^j \cap CR_{i+1}^k) \)
11: end if
12: end for
13: end for
14: \( CR_i \leftarrow CR_{temp} \)
15: end for
16: return List of stage solutions and corresponding maps of critical regions.
```

**Remark 1.** As shown in Theorem 1 of (Kouramas et al., 2011), the solution obtained using Algorithm 1 is the optimal solution of (2.17) and therefore no approximation has been introduced by the use of the approach based on multi-parametric programming and dynamic programming.

The performance of Algorithm 1 has been tested in (Kouramas et al., 2011) using a benchmark multi-parametric quadratic programming example. However, little information is provided that allows a thorough comparison between this algorithm and the conventional approach to explicit model predictive control presented in §2.2. Furthermore, the computational times provided refer only to the solution of multi-parametric quadratic programming problems at each stage, and do not take into account the time required to perform the algebraic operations involved in merging the solutions of consecutive iterations, which may be significant.

In the following section, a numerical example is solved using both approaches to explicit model predictive control and the total computational times are compared for different values of the output horizon, \( N \).
2.4 Illustrative example

The example considered in this section is of an explicit model predictive control problem with a quadratic cost function often discussed in the literature (Bemporad et al., 2002). The problem formulation is shown in (2.29).

\[
U(\theta) = \arg\min_{u_0, \ldots, u_{N-1}} \|x_N\|^2_P + \|u_{N-1}\|^2_R + \sum_{i=1}^{N-1} \|x_i\|^2_Q + \|u_{i-1}\|^2_R \\
\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k, \quad k = 0, \ldots, N-1 \\
\quad x_{\text{min}} \leq x_k \leq x_{\text{max}}, \quad k = 1, \ldots, N \\
\quad u_{\text{min}} \leq u_k \leq u_{\text{max}}, \quad k = 0, \ldots, N-1 \\
A = \begin{bmatrix} 1 & 0.05 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0025 \\ 0.05 \end{bmatrix} \\
Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 28.293 & 19.305 \\ 19.305 & 27.31 \end{bmatrix}, \quad R = 1 \\
x_{\text{max}} = \begin{bmatrix} \text{inf} \\ 0.5 \end{bmatrix}, \quad x_{\text{min}} = \begin{bmatrix} -\text{inf} \\ -0.5 \end{bmatrix}, \quad u_{\text{max}} = 1, \quad u_{\text{min}} = -1 \\
\theta = x_0 \in \Theta
\]

The solution of (2.29) is given by the explicit control law, \(u_0, u_1, \ldots, u_{N-1} = f(x_0)\), and the corresponding map of critical regions. The methods presented in §2.2 and §2.3 are used to obtain the solution, and the corresponding associated computational times are compared.

2.4.1 Solution using explicit model predictive control

The solution of (2.29) was obtained for values of \(N\) in the range \(N = 2, \ldots, 25\) by directly reformulating the problem as a multi-parametric quadratic problem of the form (2.18). The results shown in this section were obtained using the Pop Matlab toolbox (ParOS, 2004).

Figure 2.2 shows the map of critical regions obtained for \(N = 2\), in which \(\theta_{0,1}\) and \(\theta_{0,2}\) correspond to the two components of the vector \(\theta_0\). The expressions for a selection of these regions, and the corresponding optimal solutions, are presented in Table B1 of Appendix B.

While the map of critical regions in Figure 2.2 is relatively simple, the number of regions obtained for higher values of \(N\) rapidly increases, as hinted by the estimates (2.19) and (2.20).

This increase is evident in Figure 2.3, which presents the map of critical regions for \(N = 25\). A sample of the expressions for the critical regions and corresponding optimal solutions is presented in Table B.2 of Appendix B.

The first elements of the optimal sequences of inputs for each region in Figure 2.3 are shown in Figure 2.4. The purpose of this figure is to give insights on regions in the
parameter space which have similar optimal solutions; this information may be used in developing approximate solution methods to improve the efficiency of Algorithm 1.

For $N > 25$, the solution of (2.29) could not be computed using this method, since the solver ran out of memory.

### 2.4.2 Solution using dynamic programming and explicit model predictive control

To test the performance of Algorithm 1, a software implementation was developed in Matlab.

As expected, the results obtained using the approach based on multi-parametric programming and dynamic programming replicate the results shown in Figure 2.2 and Figure 2.3 (see Remark 1).

To test the closed-loop performance of the controllers designed, the system was given several disturbed initial conditions, within the bounds of the feasible region of Figure 2.2 and Figure 2.3. The convergence of each disturbed initial state to the set-point, $x = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, is shown in Figure 2.5, which refers to the explicit model predictive controller designed for $N = 25$.

For one of the simulations presented in Figure 2.5, the state and input trajectories are presented in Figure 2.6 and Figure 2.7. It may be observed that the bounds of each variable are respected. The system state converges to the set-point relatively slowly, but the rate of convergence could be improved by tuning the parameters $Q$ and $R$ in (2.29).
Chapter 2. Multi-parametric programming and MP-MPC

Figure 2.3: Map of critical regions for the solution of (2.29) with $N = 25$.

Figure 2.4: Contour plot showing how the first element of the solution of (2.29) with $N = 25$ is distributed in the parameter space.
2.5. Concluding remarks

The computational times required to solve problem (2.29) obtained using the approach of §2.2 (MP-QP) and the approach based on multi-parametric programming and dynamic programming (Dynamic MP-QP) have been compared for \( N = 2, \ldots, 30 \), and the results are plotted in Figure 2.8.

It is noticeable from figure Figure 2.8 that the dynamic programming approach is more efficient in computing the explicit solution of problem (2.29). However, it should be noted that the computational time is in the same order of magnitude as in the conventional explicit model predictive control approach and, since the controller only needs to be designed once, such gains in computational time have little practical importance.

More importantly, using the approach based on dynamic programming, it was possible to derive controllers for the full range of control horizon plotted in Figure 2.8.

Further numerical tests showed that it was possible to solve (2.29), using the dynamic programming based approach, with control horizons up to \( N = 48 \). Compared to the maximum control horizon of \( N = 25 \) reported in §2.4.1, this is an improvement that enables the use of explicit model predictive control for a wider range of applications.

2.5 Concluding remarks

This chapter presented the fundamental theory of multi-parametric programming and explicit model predictive control. The optimality conditions associated with a multi-parametric programming problem were derived for a general case, and it was shown how the explicit optimal solution may be obtained as a piece-wise affine function of
Chapter 2. Multi-parametric programming and MP-MPC

Figure 2.6: Temporal trajectories of the two components of the system state, $x_1$ and $x_2$.

Figure 2.7: Temporal trajectory of the control input of the system, $u$. 
2.5. Concluding remarks

Figure 2.8: Comparison of computational times required for the solution of problem (2.29) using $\text{mp-QP}$, and dynamic programming based $\text{MP-QP}$.

the parameters. These concepts were also presented for the special case of linear and quadratic multi-parametric programming, which find particular relevance in the context of explicit model predictive control.

A conventional approach to explicit model predictive control was presented, consisting of a direct reformulation as a multi-parametric quadratic programming problem, along with an algorithm that uses multi-parametric programming and dynamic programming to design explicit controllers.

These two approaches have been compared using a numerical example often found in the literature, and the results evidenced that the approach based on multi-parametric programming and dynamic programming is more efficient and applicable to a wider range of control horizon values.
Chapter 3

Constrained dynamic programming of mixed-integer linear problems by multi-parametric programming

This chapter presents a novel algorithm for constrained dynamic programming problems involving mixed-integer linear formulations.

The background on dynamic programming for mixed-integer problems is presented in §3.1. It is shown how the corresponding recursion may be formulated as a multi-parametric mixed-integer problem, when the system is described by linear dynamics and linear constraints, and a solver suitable for multi-parametric mixed integer linear problems is described.

The material in §3.1 serves as the basis for the algorithm for constrained dynamic programming of mixed-integer linear systems presented in §3.2.

A numerical example in which the proposed algorithm is applied to an inventory scheduling problem is presented in §3.3.

Qualitative considerations regarding the complexity of this algorithm, compared to conventional multi-parametric programming, are also presented in §3.3, and concluding remarks are summarised in §3.4.

---

1The material presented in this chapter has been submitted for publication (Rivotti and Pistikopoulos, 2013).
3.1 Constrained dynamic programming and multi-parametric programming

Dynamic programming is a technique used to efficiently solve problems involving constrained multi-stage decision processes. Such problems may have a variety of different formulations, but, in general, involve a stage-additive formulation such as (3.1) (Bertsekas, 1995).

\[
z_k(s) = \min_{x,y} f_N(x_{N-1}, y_{N-1}, s_N) + \sum_{i=k}^{N-1} f_i(x_{i-1}, y_{i-1}, s_i) \\
\text{s.t. } g_j(x_{j-1}, y_{j-1}, s_{j-1}) \leq 0, \quad j = k, \ldots, N \\
h_j(x_{j-1}, y_{j-1}, s_{j-1}) = 0, \quad j = k, \ldots, N
\]  

(3.1)

Problem (3.1) comprises \( N - k \) decision steps, each involving continuous, \( x_i \in \mathbb{R}^{n_i} \), and discrete, \( y_i \in \{0,1\}^{m_i} \), decision variables that influence the state \( s_i \in \mathbb{R}^{n_s} \). The formulation is subject to a set of inequality constraints, \( g_j(x_j, y_j, s_j) \leq 0 \), equality constraints, \( h_j(x_j, y_j, s_j) = 0 \), and a stage cost, \( f_i(x_i, y_i, s_i) \).

The sequential structure of (3.1) may be explored by applying the optimality principle, proposed by Bellman (Bellman, 1957). In general terms, this principle states that, given an optimal path, \( V_i \), from \( i \) to \( N \), the optimal path from \( k \) to \( N \), that passes by \( i \), also contains the path \( V_i \). When applied to (3.1), the optimality principle results in recursion (3.2).

\[
z_i(s_{i-1}) = \min_{x_{i-1}, y_{i-1}} f_i(x_{i-1}, y_{i-1}, s_i) + z_{i+1}(s_{i+1}) \\
\text{s.t. } g_i(x_{i-1}, y_{i-1}, s_{i-1}) \leq 0 \\
h_i(x_{i-1}, y_{i-1}, s_{i-1}) = 0
\]  

(3.2)

Note that problem (3.2) is a sub-problem of (3.1), with decision variables, and constraints, pertaining only to stage \( i \). Bellman (Bellman, 1957) demonstrated that, by recursively solving (3.2) for \( i = N, \ldots, k \), the globally optimal solution of (3.1) is obtained.

The problem addressed in this chapter is a special case of (3.1) and (3.2) where the objective function and constraints are linear, as given by (3.3).

\[
z_i(s_{i-1}) = \min_{x_{i-1}, y_{i-1}} Cx_{i-1} + Dy_{i-1} + z_{i+1}(s_{i+1}) \\
\text{s.t. } Ax_{i-1} + Ey_{i-1} \leq b \\
Lx_{i-1} + Ky_{i-1} = c \\
y_{i-1} \in \{0,1\}^{m_i}
\]  

(3.3)

The matrices \( A, C, D, E, L, K \), and vectors \( b, c \) in (3.3) are linear coefficients of appropriate dimensions.
Note that although \( y_i \) in (3.3) is a binary vector, the formulation may be used to describe an integer vector, \( y_i \in (I \in \mathbb{Z}^{n_d}) \), by introducing \( n_d \) binary variables, \( d_i \in \{0,1\}^{n_d} \), such that \( y_i = \sum_{j=1}^{n_d} I_j d_{i,j} \) and \( \sum_{j=1}^{n_d} d_{i,j} = 1 \).

The method for constrained dynamic programming by multi-parametric programming proposed by Faísca et al. (Faísca et al., 2008), for multi-parametric linear/quadratic problems, involves recursion (3.4), formulated for stage \( i \) of a process with \( N \) stages.

\[
\begin{align*}
    z_i(\theta_i) &= \min_{x_{i-1}} f_i(x_{i-1}, \theta_i) + z_{i+1}(\theta_{i+1}) \\
    &\text{s. t. } g_i(x_{i-1}, \theta_i) \leq 0 \\
    &\quad h_i(x_{i-1}, \theta_i) = 0 \\
    &\quad \theta_i = [s_{i-1}, x_i, \ldots, x_{N-1}] \\
\end{align*}
\] 

(3.4)

By considering a parameter vector, \( \theta_i \), that includes both the state vector, \( s_{i-1} \), and the future decisions, \( x_i, \ldots, x_{N-1} \), a convex objective function is obtained, and problem (3.4) may be solved without the need for global optimisation techniques.

Following the same methodology for a mixed-integer linear problem, such as (3.3), a series of multi-parametric mixed-integer linear problems is obtained, of the general form (3.5).

\[
\begin{align*}
    z_i(\theta_i) &= \min_{x_{i-1}, y_{i-1}} C x_{i-1} + D y_{i-1} \\
    &\text{s. t. } A x_{i-1} + E y_{i-1} \leq b + F \theta_i \\
    &\quad L x_{i-1} + K y_{i-1} = c + Q \theta_i \\
    &\quad \theta_i \in \Theta_i, y_{i-1} \in \{0,1\}^{n_d} \\
\end{align*}
\] 

(3.5)

Note that the vector of parameters \( \theta_i \) in (3.4) is augmented in (3.5) to include the discrete optimisation variables, \( y_{i+1}, \ldots, y_N \).

To obtain the solution of problem (3.5), several solving algorithms exist in the literature (Acevedo and Pistikopoulos, 1997; Pertsinidis et al., 1998; Dua and Pistikopoulos, 2000; Li and Ierapetritou, 2007a; Wittmann-Hohlbein and Pistikopoulos, 2012a).

The algorithm proposed by Dua et al. (Dua and Pistikopoulos, 2000) is based on the decomposition of problem (3.5) into a MP-LP subproblem, and an associated deterministic MILP problem.

The MP-LP subproblem associated with (3.5) is obtained by fixing \( y_i = \bar{y}_i \), where \( \bar{y}_i \) is a feasible solution of (3.5). The algorithm for solving the MP-LP subproblem involves analysing the optimality conditions in the neighbourhood of an optimal solution, for a perturbation in the parameter vector, as outlined in §2.1 (Theorem 1 and Corollary 1).

The optimal value of the MP-LP subproblem, \( z^*_i(\theta_i) \), provides an upper bound to the overall solution of (3.5). To obtain a lower bound, the deterministic MILP problem (3.6) is solved for each region, CR\(_i\), of the map of critical regions where the solution of (3.5) is defined.
Chapter 3. Dynamic programming of mixed-integer linear problems

\[ z_i(\theta_i) = \min_{x_i, y_i, \theta_i} \{ Cx_i + Dy_i \} \]

s.t. \[ Ax_i + Ey_i \leq b + F\theta_i \]
\[ Lx_i + Ky_i = c + Q\theta_i \]
\[ (\square) \sum_{j \in J_i^k} y_i^j - \sum_{j \in L_i^k} y_i^j \leq |J_i^k| - 1, \quad k = 1, \ldots, K_i \]  \hspace{1cm} (3.6)
\[ (\triangle) \quad Cx_i + Dy_i \leq z_i^*(\theta_i) \]
\[ \theta_i \in \text{CR}_i \]

The \( K_i \) integer combinations already explored in \( \text{CR}_i \) are divided into sets \( J_i^k = \{ j \mid y_i^j = 1 \} \), and \( L_i^k = \{ j \mid y_i^j = 0 \} \), and excluded from the solution by introducing the integer cuts \( \square \). The inequality \( \triangle \) guarantees that the optimal value of (3.6) is not higher than the current upper bound, \( z_i^*(\theta_i) \).

If (3.6) is infeasible, \( z_i^*(\theta_i) \) becomes the solution of (3.5) for \( \theta_i \in \text{CR}_i \); otherwise, problem (3.5) is re-solved, with \( \bar{y} \) set to the newly found integer solution. The resulting optimal value is then compared to \( z_i^*(\theta_i) \), using the method presented by Acevedo et al. (Acevedo and Pistikopoulos, 1997), to obtain an update of the current upper bound. The algorithm continues to iterate between the \text{MP-LP} and \text{MILP} subproblems, until all \text{MILP} subproblems are infeasible.

The final solution of (3.5), corresponds to the optimal piece-wise affine functions, \( x_i(\theta_i) \) and \( y_i(\theta_i) \), as well as the map of critical regions for which these relations are valid. It is possible for the number of critical regions obtained to be significantly large, thus making it computationally expensive to retrieve the corresponding optimal solutions. However, methods have been developed (Tøndel et al., 2003a) to efficiently compute the optimal solutions, by generating a binary search tree.

Note that instead of re-formulating the dynamic programming recursion, (3.3), the original dynamic programming problem (3.1) may be formulated and solved as a single multi-parametric mixed-integer problem, such as (3.5), of larger dimensionality. However, one of the main limitations of multi-parametric mixed-integer programming remains the exponential complexity of the algorithm (Dua et al., 2002), observed for high-dimensional problems. By re-reformulating the dynamic programming recursion (3.3), a series of lower dimensional multi-parametric mixed-integer linear sub-problems is obtained, that may be efficiently solved. The example in §3.3 presents a qualitative complexity analysis that highlights the corresponding improvement in performance.

3.2 Algorithm for constrained dynamic programming of mixed-integer linear problems

Consider a 2 stage decision process, schematically represented in Figure 3.1.
3.2. Algorithm for dynamic programming of mixed-integer linear problems

![Diagram](image)

Figure 3.1: Schematic depiction of process with 2 stages.

In order to solve recursion (3.5), the procedure should start at stage $i = 2$. At this stage, problem (3.5) involves the decision vector $X_1 = [x_1 \ y_1]^T$ and the parameter vector $\theta_2 = s_1$.

The solution of the sub-problem is obtained by using the algorithm for multi-parametric mixed integer programming described in §3.1. As guaranteed by Theorem 1, and respective Corollary, the solution of (3.3) is a piece-wise affine function of the form $X_1 = f(\theta_2)$, defined over a set of critical regions CR$_2$.

At stage $i = 1$, the sub-problem to be solved involves the decision vector $X_0 = [x_0 \ y_0]^T$ and the parameter vector $\theta_1 = [s_0 \ X_1]^T$. The corresponding optimal solution is the piece-wise affine function $X_0 = f(\theta_1)$, defined over the set CR$_1$.

To obtain the solution of stage $i = 1$, it is necessary to obtain the solution $X_0 = f(s_0)$, by combining the optimal solutions of the two stages.

For two critical regions in the sets CR$_1$ and CR$_2$, the respective optimal solutions are of the form (3.7) and (3.8).

Stage 1: $X_0 = A_1 s_0 + B_1 X_1 + C_1 \quad (3.7)$

Stage 2: $X_1 = A_2 s_1 + C_2 \quad (3.8)$

The solution of stage 1, $X_0 = f(s_0)$ is obtained by replacing (3.8) in (3.7) and noting that $s_1$ is related to $s_0$ through the dynamic model of the system.

Considering that CR$_1$ and CR$_2$ contain $n_1$ and $n_2$ regions, respectively, this procedure is repeated for the $n_1 \times n_2$ possible combinations of critical regions. In order to detect possible empty regions, a feasibility test is performed over the union of the two critical regions being considered.

Note that, given the convexity of $z_1(\theta_1)$ and $z_2(\theta_2)$, the feasible intersection of critical regions is unique, and no overlaps result in the final map of critical regions at each stage.

The procedure described for this two stage problem can be generalised for a problem with $N$ stages.

The proposed algorithm for constrained dynamic programming of mixed-integer linear problems by multi-parametric programming is summarised in the following steps.
Chapter 3. Dynamic programming of mixed-integer linear problems

Algorithm 2 Dynamic programming of mixed-integer linear problems by multi-parametric programming

1: Reformulate dynamic programming recursion (3.2) as a multi-parametric mixed-integer linear problem of the form (3.5).
2: Solve stage \( N \) of (3.5) and obtain solution (3.8) defined over \( \text{CR}_N \subseteq \mathbb{R}^{nN} \).
3: \textbf{for} \( i \leftarrow N - 1, \ldots, 1 \) \textbf{do}
4: \hspace{1em} \text{CR}_\text{temp} \leftarrow \emptyset
5: \hspace{1em} Solve stage \( i \) of (3.5) and obtain solution (3.7) defined over \( \text{CR}_i \subseteq \mathbb{R}^{ni} \)
6: \hspace{1em} \textbf{for} \( j \leftarrow 1, \ldots, n_i \) \textbf{do}
7: \hspace{2em} \textbf{for} \( k \leftarrow 1, \ldots, n_{i+1} \) \textbf{do}
8: \hspace{3em} Test feasibility of \( \text{CR}_i \cap \text{CR}_{i+1} \).
9: \hspace{3em} \textbf{if} Intersection is feasible \textbf{then}
10: \hspace{4em} Compute stage solution by replacing (3.8) in (3.7).
11: \hspace{4em} \text{CR}_\text{temp} \leftarrow \text{CR}_\text{temp} \cup (\text{CR}_i \cap \text{CR}_{i+1})
12: \hspace{3em} \textbf{end if}
13: \hspace{2em} \textbf{end for}
14: \hspace{1em} \textbf{end for}
15: \hspace{1em} \text{CR}_i \leftarrow \text{CR}_\text{temp}
16: \textbf{end for}
17: \textbf{return} List of stage solutions and corresponding maps of critical regions.

3.3 Illustrative example - An inventory scheduling problem

The example that follows is adapted from a process scheduling problem (Bellman and Dreyfus, 1962), in which an optimal stock policy is determined by balancing the cost of changing the level of stock in consecutive periods, and the cost of maintaining a stock level above the minimum required.

The total cost, \( C \), incurred over \( N \) periods, is given by (3.9).

\[
C(x_1, x_2, \ldots, x_N) = \sum_{k=1}^{N} [\phi_k(x_k - r_k) + \psi_k(x_k - x_{k-1})] \quad (3.9)
\]

In (3.9), \( x_i \) and \( r_i \) are integer variables representing the stock level, and minimum stock required, at period \( i \), respectively. There are several possibilities for the penalty functions \( \phi(x_k - r_k) \) and \( \psi(x_k - x_{k-1}) \), but for simplicity these are assumed to be of the form (3.10) and (3.11).

\[
\phi(x_k - r_k) = x_k - r_k \quad (3.10)
\]
\[
\psi(x_k - x_{k-1}) = \max(x_k - x_{k-1}, 0) \quad (3.11)
\]
3.3. Illustrative example - An inventory scheduling problem

It may be noted that the two objectives, (3.10) and (3.11), are conflicting; an optimal stock policy should not involve simply keeping the stock at the minimum required, because this would result in penalties due to the associated stock variations.

The optimal stock policy, \( x^*(r) \), may be determined by solving problem (3.12).

\[
x^*(r, x_0) = \arg\min_{x_1, x_2, \ldots, x_N} \sum_{k=1}^{N} (x_k - r_k) + a \max(x_k - x_{k-1}, 0)
\]

\[
\text{s. t. } x_k \geq r_k, k = 1, \ldots, N
\]

(3.12)

The use of the non-smooth term, \( \max(x_k - x_{k-1}, 0) \), in (3.12) may be prevented by introducing \( N \) auxiliary variables, \( z_k \), and \( 2N \) auxiliary constraints, and re-writing (3.12) as the equivalent problem (3.13).

\[
x^*(r, x_0) = \arg\min_{x_1, x_2, \ldots, x_N} \sum_{k=1}^{N} (x_k - r_k) + az_k
\]

\[
\text{s. t. } x_k \geq r_k, k = 1, \ldots, N
\]

(3.13)

Defining \( f_R \) as (3.14), the recursion associated with (3.13) may be written, for \( i = 1, \ldots, N - 1 \), as (3.15).

\[
f_R = \min_{x_k, \ldots, x_N} \sum_{k=R}^{N} (x_k - r_k) + az_k
\]

\[
\text{s. t. } x_k \geq r_k, k = R, \ldots, N
\]

(3.14)

\[
f_i = \min_{x_i} [(x_i - r_i) + az_i + f_{i+1}]
\]

\[
\text{s. t. } x_i \geq r_i
\]

(3.15)

As opposed to the results presented in (Bellman and Dreyfus, 1962), here the vector of requirements, \( r_i \), is not assumed to be fixed. The assumption is avoided by introducing \( N \) parameters in problem (3.12), and defining, for each, a range of variation around the nominal value, \( r_0^0 \), as shown in (3.16).

\[
r^0 - \delta_r \leq r \leq r^0 + \delta_r
\]

(3.16)
Chapter 3. Dynamic programming of mixed-integer linear problems

Defining $X \triangleq [x_i, \ldots, x_N, z_i, \ldots, z_N]^T$, and a vector, $\theta = [x_{i-1}, r_i, \ldots, r_N]$, the original problem (3.13) may be re-written as a multi-parametric mixed-integer linear problem of the form (3.17).

$$X^*(\theta) = \arg\min_X \left[ \tilde{1}^N \ a^N \right] X + \left[ -\tilde{1}^N \right] \theta$$

s.t.\[
\begin{bmatrix}
- I^N & 0 \cdot I^N \\
0 \cdot I^N & - I^N \\
[1]_N^{-1} \left[ - I^N \right] & - I^N \\
\end{bmatrix} X \leq \begin{bmatrix} 0^N \left[ - \tilde{1}^T \right] \\
0^N \left[ - \tilde{0}^T \right] \\
[1, 0]_N^{-1} \left[ - \tilde{0}^T \right]
\end{bmatrix} + \begin{bmatrix}
- \tilde{1}^T \\
0^N \left[ - \tilde{0}^T \right] \\
[1, 0]_N^{-1} \left[ - \tilde{0}^T \right]
\end{bmatrix} \theta
\]  

(3.17)

In (3.17), $I^N$ is the $N \times N$ identity matrix. For a scalar constant $c_\gamma \bar{c}_i^n \triangleq c \sum_{i=1}^N e_i$ and $\bar{c}_i^n \in \mathbb{R}^{N \times N}$ is a matrix containing the vector $\bar{c}$ in the $i^{th}$ diagonal below the main diagonal and all remaining elements equal to zero.

The solution of (3.17) may be obtained using the multi-parametric mixed-integer solver, presented in §3.1. However, as noted in §3.1, the computational effort required to solve (3.17) becomes prohibitive as $N$ increases, due to the associated increase in the number of optimisation variables, and constraints. Figure 3.3 shows the time required to compute the solution of (3.17) for $a = 2, r^0 = [13, 7, 1, 8, 14]^T$, and $N = 2, \ldots, 5$. The solver ran out of memory for $N > 5$ and therefore the solution could not be computed.

The multi-parametric mixed-integer linear problem (3.17) may be decomposed into a series of sub-problems by recasting the dynamic programming recursion (3.15) as a multi-parametric mixed-integer recursion, to be solved at each stage $i = 1, \ldots, N$. Considering the optimisation vector, $X_i = [x_i, z_i, \ldots, z_N]^T$, and the vector of parameters, $\theta_i = [x_{i-1}, x_{i+1}, \ldots, x_N, r_i, \ldots, r_N]^T$, the multi-parametric recursion is formulated as (3.18).\footnote{Total CPU times refer to an Intel®Core™Quad Q9400 @ 2.66GHz processor, 4GB RAM}
3.3. Illustrative example - An inventory scheduling problem

\[ X_i^* = \arg\min_{X_i} \left[ \left[ 1 \ a^{N-i+1} \right] X + \left[ 0 \quad 1^{N-i} \quad -1^{N-i+1} \right] \theta \right] \]

s.t.

\[
\begin{bmatrix}
-1 & 0^{N-i+1} \\
0^{N-i+1} & -1^{N-i+1} \\
1 & 0^{N-i+1}
\end{bmatrix}
\begin{bmatrix}
X_i \\
0^{N-i+1} \\
0^{N-i+1}
\end{bmatrix}
\leq
\begin{bmatrix}
0 \\
0^{N-i+1} \\
0^{N-i+1}
\end{bmatrix}
\]  

(3.18)

The recursive problem (3.18) may be solved using the algorithm presented in §3.2. The solution of (3.18) at each stage, \( X_i = f(\theta_i) \), and the corresponding map of critical regions provides insights into the structure of the original problem, such as, for example, how the optimal solution at one period is influenced by decisions at later periods. This allows the possibility of simulating different scenarios without the need to re-compute the solution of the problem.

To illustrate the use of the algorithm for mixed-integer dynamic programming, presented in §3.2, problem (3.18) was solved for the values of \( a, r_0, \) and \( N \) given in (3.19).

\[
a = 2 \\
r_0 = \begin{bmatrix} 0^{N-i} & 1^{N} \end{bmatrix} \cdot \begin{bmatrix} 12 & 11 & 13 & 7 & 1 & 8 & 14 \end{bmatrix}
\]

\( N = 2, \ldots , 7 \)

Figure 3.2 illustrates, as an example, the 56 critical regions obtained for the sub-problem (3.18), for \( i = 5 \) (\( N = 6 \)), after replacing the solution of iteration \( i = 6 \). In contrast, the final solution \( X = f(x_0, r_1, \ldots , r_6) \) comprises 1752 critical regions.

Some specimens of the corresponding optimal solutions, \( x_5 = f(x_0, r_5, r_6) \), and the final solution, \( X = f(x_0, r_1, \ldots , r_6) \), are shown in Tables C.1 and C.2, of Appendix C, respectively. Note, that the results for the last six periods obtained in (Bellman and Dreyfus, 1962), may be replicated by taking \( N = 6, r = r_0, \) and \( x_0 = 12 \).

Remark 1. Efficiency of Algorithm 2.

The computational times required to solve the dynamic programming based algorithm for \( N = 2, \ldots , 7 \) are plotted in Figure 3.3. As expected, for low values of \( N \), the solution of (3.17) is computed faster. However, for higher values, the algorithm based on dynamic programming becomes clearly more efficient, and allows to determine solutions that could not be computed using existing algorithms.
Figure 3.2: Map of critical regions for iteration $i = 5$ of dynamic programming based algorithm with $N = 6$.

Figure 3.3: Comparison of computational times required for the solution of problem (3.17) using MP-MILP and dynamic programming based MP-MILP.
Despite the illustrated benefits, it may be observed from Figure 3.3 that the complexity of the algorithm based on dynamic programming remains exponential. This is due to the increase in the number of parameters that occurs after each iteration of sub-problem (3.18).

To study the profile of the required computational time, the algorithm was divided in three sections: (a) Comparison procedure - corresponding to the algebraic operations, and feasibility tests, required to combine the solutions of successive iterations; (b) Critical region reduction - comprising all operations required to obtain a minimal representation of the critical regions, and to remove any existing overlap; (c) MP-MILP solver - consisting of the solution of MP-MILP sub-problems (3.18) using existing MP-MILP solvers.

Taking these three categories into account, the algorithmic profile was determined for the more representative cases, $N = 4, 5, 6$. The results are shown in Figure 3.4.

As seen in Figure 3.4, the solution of MP-MILP problems contributes the least to the total computational effort. However, the high number of critical regions produced at each stage impacts the auxiliary operations required to combine the solutions of successive iterations, since all combinations of critical regions need to be processed, as described in §3.1.

It should be noted that the computational results here presented refer to a prototype algorithm implemented in Matlab, with no particular emphasis on efficiency. In particular, many operations related to the comparison procedure could be parallelised, which would likely result in significant performance improvements.

The critical region simplification, on the other hand, was implemented using mature routines of the Pop Matlab toolbox (ParOS, 2004), and, therefore, only minimal
Chapter 3. Dynamic programming of mixed-integer linear problems

Performance improvements may be expected from code changes.

The possibility of using different algorithms for the solutions of the MP-MILP sub-problems will be the subject of future research. A recent algorithm for multi-parametric programming (Gupta et al., 2011) proposed a combinatorial method for exploring all candidate active sets, which avoids the geometrical operations typically involved in exploring the parameter space. Further results, over an improved algorithm based on the combinatorial approach (Feller et al., 2012), showed that significant performance enhancements may be obtained, for problems with a moderate amount of constraints, and high-dimensional parameter vectors. Based on these results, it is expected that such algorithms will be well suited for the sub-problems that result from the multi-parametric dynamic programming approach. Since the combinatorial method also avoids overlaps, the computations required for the critical region simplification are also expected to be reduced, further contributing to a decrease in the total computational time.

Remark 2. Problems with quadratic objective function and linear constraints.

An area of ongoing research is the extension of Algorithm 2 to multi-stage mixed-integer problems with a quadratic objective function and linear constraints.

The main challenge is the fact that mixed-integer multi-parametric quadratic solvers (Dua et al., 2002) store the explicit solution as an envelope of optimal solutions, corresponding to different feasible integer solutions. This procedure, necessary to avoid non-convex comparison of optimal solutions, implies that, at each stage of recursion (3.3), there are more critical regions stored than those required to identify the optimal solution. Since Algorithm 2 proceeds iteratively, the computational requirements are expected to increase with the size of problem in a manner more pronounced than Figure 3.3.

Remark 3. Uncertainty in the model.

In Chapter 5 it is shown how different modelling approaches may be used to take into account uncertainties in the dynamical model of the system. The method presented in this chapter and in Chapter 4 may be combined with a robustification step to form the basis of a framework for robust dynamic programming of mixed-integer linear systems, as shown in Chapter 5.

3.4 Concluding remarks

This chapter presented an algorithm for constrained dynamic programming of problems involving mixed-integer linear formulations. By re-formulating the dynamic programming recursion as a series of mixed-integer multi-parametric programming sub-problems, the optimal solution of the original problem is obtained without the need for global optimisation techniques. The proposed algorithm was applied to a multi-stage mixed-integer linear problem that arises in the context of inventory scheduling. A qualitative complexity analysis highlights the advantages of using this method, and outlines directions for future work, regarding the efficiency of the algorithm.
Chapter 4

Explicit model predictive control for hybrid systems

As mentioned in Chapter 1, one important research direction in model predictive control, and explicit model predictive control, consists of the design of controllers for systems with formulation that involve integer variables (Morari and Lee, 1999; Pistikopoulos, 2009).

In this chapter, based on the ideas from Chapter 3, a novel algorithm for explicit hybrid model predictive control is presented, using multi-parametric programming and dynamic programming techniques.

Considerations related to the modelling of systems with formulations that include integer variables are presented in §4.1. The class of piece-wise affine problems is defined, and a general framework for the conversion of models based on logical propositions as mixed-integer linear problems is presented.

In §4.2, it is shown how hybrid model predictive control problems may be reformulated as multi-parametric mixed-integer programming problems for which the explicit solution is obtained as a function of the initial state of the system.

The algorithm for explicit model predictive control of hybrid systems based on multi-parametric programming and dynamic programming is presented in §4.3.

In §4.4, a numerical example illustrates how the proposed algorithm may be used to derive an explicit model predictive controller for a piece-wise affine system. Qualitative considerations regarding the complexity of the algorithm are also presented.

---

1The material in this chapter is being prepared for submission as a journal article. (Rivotti and Pistikopoulos, 2013)
4.1 Modelling and optimisation of hybrid systems

Hybrid systems are an important topic in optimisation and control theory, and several classes of models are of a hybrid nature. In this section, the discussion is focused on piece-wise affine systems, a particular class of hybrid systems.

Piece-wise affine systems (Sontag, 1981; van Bokhoven, 1981; Johansson and Rantzer, 1998; Bemporad et al., 2000; Liberzon, 2003) are an important modelling tool and may be used, for example, to describe systems with nonlinear dynamics (Sontag, 1981).

A piece-wise affine state-space model is defined as shown in (4.1).

\[
\begin{align*}
x_{k+1} &= A_i x_k + B_i u_k + c_i, \quad \forall (x_k, u_k) \in P_i \\
y_k &= C_i x_k + D_i u_k
\end{align*}
\]  

(4.1)

In (4.1), \( P_i \) is a polyhedral partition of the space defined by the state and input of the system, and given by (4.2). The union of the polyhedra, \( P = \bigcup_{i=1,...,p} P_i \), is assumed to contain the origin.

\[
P_i = \{ (x_k, u_k) \mid F_i x_k + G_i u_k \leq b_i, \quad i = 1, \ldots, p \}
\]  

(4.2)

Different values of the state-space matrices, \( A_i \in \mathbb{R}^{n \times n}, B_i \in \mathbb{R}^{n \times m}, C_i \in \mathbb{R}^{ny \times n}, \) and \( D_i \in \mathbb{R}^{ny \times m} \) are triggered by the state of the system, \( x_k \in \mathbb{R}^n \), and input of the system, \( u_k \in \mathbb{R}^m \), reaching the corresponding partitions, \( P_i \).

One important property commonly required for the study of piece-wise affine systems is for (4.1) to be a well-posed system, in the sense of Definition 1.

**Definition 1.** Well-posed piece-wise affine system (Bemporad et al., 2000).

A piece-wise affine system, such as (4.1), is considered well-posed in \( P \) if, for all pairs \( (x_k, u_k) \in P \), the pair \( (x_{k+1}, u_{k+1}) \) exists and is uniquely determined.

As remarked in (Bemporad et al., 2000), for (4.1) to be well-posed it is necessary for the definition of the partitions (4.2) to include both strict and non-strict inequalities.

For purposes such as dynamic simulations or model predictive control, the use of hybrid systems may result in unexpected behaviour that does not occur for continuous-time systems, due to phenomena such as Zeno behaviour (Johansson et al., 1999; Zhang et al., 2001), which consists of the system taking infinite discrete transitions in a finite time.

Due to these particular phenomena, it is important to perform verification tests, referred to as reachability analysis, that guarantee safety properties for all initial states and possible sequences of inputs (Alur et al., 1995; Tomlin et al., 2003). For an overview of computational tools available for the verification of hybrid systems see (Silva et al., 2001).

For numerical applications involving optimisation problems, such as model predictive control, it becomes necessary to reformulate problems of a hybrid nature, such as (4.1), in a form suitable for the use of optimisation methods.
The piece-wise affine system (4.1) may be written as a set of logical propositions by introducing a set of binary variables, \( \delta_k \in \{0, 1\}^p \), as shown in (4.3).

\[
\delta_k = 1 \Leftrightarrow \begin{cases} 
  x_{k+1} = A_i x_k + B_i u_k + c_i, & i = 1, \ldots, p \\
  (x_k, u_k) \in P_i 
\end{cases}
\]

(4.3)

The re-writing of (4.3) as a set of linear equalities and inequalities involves introducing an additional set of continuous-time auxiliary variables, \( z_k \in \mathbb{R}^{n \times p} \), as shown in (4.4) (Williams, 1999; Bemporad and Morari, 1999a).

\[
x_{k+1} = \sum_{i=1}^{p} z_{k,i} \\
\text{s. t. } m \delta_{k,i} \leq z_{k,i} \leq M \delta_{k,i}, \quad i = 1, \ldots, p \\
z_{k,i} \leq A_i x_k + B_i u_k + c_i - M(1 - \delta_{k,i}), \quad i = 1, \ldots, p \\
z_{k,i} \geq A_i x_k + B_i u_k + c_i - m(1 - \delta_{k,i}), \quad i = 1, \ldots, p \\
F_i x_k + G_i u_k \leq b_i + L_i(1 - \delta_{k,i}), \quad i = 1, \ldots, p \\
\sum_{i=1}^{p} \delta_{k,i} = 1
\]

In (4.4), \( M = \max_{i=1,\ldots,p} \left\{ \max_{(x,u) \in P} A_i x + B_i u \right\} \), \( m = \min_{i=1,\ldots,p} \left\{ \max_{(x,u) \in P} A_i x + B_i u \right\} \), and \( L_i = \max_{(x,u) \in P} F_i x + G_i u - b_i \).

The reformulation shown above for the specific case of piece-wise affine functions has been generalised for the conversion of any logical proposition into a set of linear equalities and inequalities. Using the Mixed Logical Dynamical framework (MLD), such propositions may be re-written in the general form (4.5) (Bemporad and Morari, 1999a).

\[
x_{k+1} = A x_k + B_1 u_k + B_2 \delta_k + B_3 z_k \\
y_k = C x_k + D_1 u_k + D_2 \delta_k + D_3 z_k \\
\text{s. t. } E_2 \delta_k + E_3 z_k \leq E_1 u_k + E_4 x_k + E_5
\]

(4.5)

Furthermore, it has been shown that any mixed logical dynamical system of the form (4.5) may be equivalently written as a piece-wise affine system (Bemporad et al., 2000; Bemporad, 2004).

### 4.2 Hybrid explicit model predictive control

In this section, it is shown how an explicit model predictive controller with linear objective function is obtained for a piece-wise affine system of the form (4.1). The problem of designing model predictive controllers based on the 1-norm or \( \infty \)-norm is described in (Zadeh and Whalen, 1962; Bemporad et al., 2002b).

For the piece-wise affine system (4.1), such problem is written in the form (4.6).
Chapter 4. Explicit model predictive control for hybrid systems

\[ U(x_0) = \arg\min_{u_0, \ldots, u_{N-1}} \|x_N\|_P^1 + \|u_{N-1}\|_R^1 + \sum_{k=1}^{N-1} \|x_k\|_Q^1 + \|u_{k-1}\|_R^1 \]

\[ \text{s.t. } \quad x_{k+1} = A_i x_k + B_i u_k + c_i, \forall (x_k, u_k) \in \mathcal{P}_i, \quad k = 0, \ldots, N - 1 \]  (4.6)

\[ x_{\min} \leq x_k \leq x_{\max}, \quad k = 1, \ldots, N \]

\[ u_{\min} \leq u_k \leq u_{\max}, \quad k = 0, \ldots, N - 1 \]

The piece-wise affine system is assumed to be well posed in \( \mathcal{P} = \bigcup_{i=1}^{p} \mathcal{P}_i \).

As mentioned in §4.1, a piece-wise affine system is equivalently described by a set of linear inequalities and equalities. By introducing auxiliary binary variables, \( \delta \), auxiliary continuous variables, \( z \), and replacing (4.4) in (4.6), the model predictive formulation is re-written in the form (4.7).

The non-smooth terms in (4.7), \( \|a\|_A^1 \), resulting from basing the formulation on the 1-norm are relaxed using the conventional linear programming technique (Kelley, 1958) which introduces additional auxiliary continuous variables, \( e^a \), and constraints of the form \( -e^a \leq Aa \leq e^a \).

Problem (4.7) may then be reformulated as a multi-parametric mixed-integer linear problem of the form (4.8), by considering the initial state of the system, \( x_0 \) as the vector of parameters, \( \theta \), and defining the vectors of continuous decision variables, \( X \), and binary decision variables, \( Y \).
4.2. Hybrid explicit model predictive control

\[ U(x_0) = \arg\min_{u_0, \ldots, u_{N-1}} \|x_N\|_P + \|u_{N-1}\|_R + \sum_{k=1}^{N-1} \|x_k\|_Q + \|u_{k-1}\|_R \]

\[ \text{s.t. } x_{k+1} = \sum_{i=1}^p z_{k,i} \]
\[ x_{\text{min}} \leq x_{k+1} \leq x_{\text{max}} \]
\[ u_{\text{min}} \leq u_k \leq u_{\text{max}} \]
\[ m\delta_{k,i} \leq z_{k,i} \leq M\delta_{k,i} \]
\[ z_{k,i} \leq A_i x_k + B_i u_k + c_i - M(1 - \delta_{k,i}) \]
\[ z_{k,i} \geq A_i x_k + B_i u_k + c_i - m(1 - \delta_{k,i}) \]
\[ F_i x_k + G_i u_k \leq b_i + L_i (1 - \delta_{k,i}) \]
\[ \sum_{i=1}^p \delta_{k,i} = 1 \quad k = 0, \ldots, N - 1, \quad i = 1, \ldots, p \]

(4.7)

\[ U(\theta) = \arg\min_{X, Y} \epsilon_X^N + \epsilon_u^{N-1} + \sum_{k=1}^{N-1} \epsilon_X^k + \epsilon_u^{k-1} \]

\[ \text{s.t. } x_{k+1} = \sum_{i=1}^p z_{k,i} \]
\[ x_{\text{min}} \leq x_{k+1} \leq x_{\text{max}} \]
\[ u_{\text{min}} \leq u_k \leq u_{\text{max}} \]
\[ -\epsilon_X^k \leq Q x_{k+1} \leq \epsilon_X^k \]
\[ -\epsilon_X^N \leq P x_N \leq \epsilon_X^N \]
\[ -\epsilon_u^k \leq R u_k \leq \epsilon_u^k \]
\[ m\delta_{k,i} \leq z_{k,i} \leq M\delta_{k,i} \]
\[ z_{k,i} \leq A_i x_k + B_i u_k + c_i - M(1 - \delta_{k,i}) \]
\[ z_{k,i} \geq A_i x_k + B_i u_k + c_i - m(1 - \delta_{k,i}) \]
\[ F_i x_k + G_i u_k \leq b_i + L_i (1 - \delta_{k,i}) \]
\[ 1 \leq \sum_{i=1}^p \delta_{k,i} \leq 1 \quad k = 0, \ldots, N - 1, \quad i = 1, \ldots, p \]

(4.8)

\[ X = [u_0, \ldots, u_{N-1}, z_0, \ldots, z_{N-1}, \epsilon_X^1, \ldots, \epsilon_X^{N-1}, \epsilon_X^0, \epsilon_u^1, \ldots, \epsilon_u^{N-1}]^T \]
\[ Y = [\delta_0, \ldots, \delta_{N-1}]^T \]
\[ \theta = x_0 \]

Problem (4.8) is a multi-parametric mixed-integer linear programming problem of the general form (4.9), that may be solved using the algorithmic implementations described in §3.1.
\[
U(\theta) = \arg\min_{X,Y} CX + DY \\
\text{s.t. } AX + EY \leq b + F\theta \\
LX + KY = c + Q\theta \\
\theta \in \Theta, Y \in \{0,1\}^{N-p}
\]

In (4.9), \(C, D, A, E, F, L, K, Q, b, \) and \(c\) are linear coefficients of appropriate dimensions. The solution of (4.9) comprises the sequence of optimal control inputs, \(X(\theta)\), the associated sequence of switching between partitions, \(Y(\theta)\), and the map of critical regions in the parameter space where these relations are valid.

### 4.3 Explicit hybrid model predictive control by dynamic programming

In this section, a novel algorithm for explicit model predictive control of hybrid systems is proposed. The proposed method is based on the developments in constrained dynamic programming of hybrid linear systems presented in Chapter 3 and on multi-parametric mixed-integer linear programming.

The objective is to obtain the explicit solution of a problem of the form (4.10), involving piece-wise affine system dynamics and a linear objective function.

\[
U(x_0) = \arg\min_{u_0,\ldots,u_{N-1}} \|x_N\|_P + \|u_{N-1}\|_R + \sum_{i=1}^{N-1} \|x_i\|_Q + \|u_{i-1}\|_R \\
\text{s.t. } x_{k+1} = A_i x_k + B_i u_k + c_i, \forall (x_k,u_k) \in P_i, \quad k = 0,\ldots,N-1 \\
x_{\min} \leq x_k \leq x_{\max}, \quad k = 1,\ldots,N \\
u_{\min} \leq u_k \leq u_{\max}, \quad k = 0,\ldots,N-1
\]

(4.10)

Although it is possible to reformulate (4.10) as a multi-parametric mixed integer linear problem, as shown in §4.2, such approach does not take into account the sequential structure of the problem (Bertsekas, 1995). By considering the principle of optimality of Bellman (Bellman, 1957), (4.10) may be written in the recursive form (4.11), which highlights the structure of the problem.

\[
V_j(x_{j-1}) = \min_{u_{j-1}} \begin{cases}  
\|x_j\|_Q + \|u_{j-1}\|_R + V_{j+1}(x_j) & \text{if } j < N \\
\|x_N\|_P + \|u_{j-1}\|_R & \text{if } j = N 
\end{cases} \\
\text{s.t. } x_j = A_i x_{j-1} + B_i u_{j-1} + c_i, \forall (x_{j-1},u_{j-1}) \in P_i \\
x_{\min} \leq x_j \leq x_{\max} \\
u_{\min} \leq u_{j-1} \leq u_{\max}
\]

(4.11)
Comparing (4.11) with (4.10), it is noticeable that the dimensionality of the optimisation vector and number of constraints is reduced by disassembling the original formulation into a sequence of recursive sub-problems.

The procedure for obtaining the explicit solution of (4.11) involves reformulating the problem as a multi-parametric mixed-integer linear problem. For this purpose, the proposed method follows the lines of (Faisca et al., 2008), in which the future cost $V_{j+1}(x_j)$ is expanded and (4.11) is reformulated as a multi-parametric programming problem, by including the terms related to future control inputs in the vector of parameters. The resulting problem is of the form (4.12).

$$V_j(\theta) = \min_{u_{j-1}} \|x_N\|_p^1 + \|u_{N-1}\|_R^1 + \sum_{k=j}^{N-1} \|x_k\|_Q^1 + \|u_{k-1}\|_R^1$$

s. t.  
$$x_k = A_ix_k + B_iu_{k-1} + c_i, \forall (x_{k-1}, u_{k-1}) \in P_i, \quad k = j, \ldots, N$$
$$x_{min} \leq x_j \leq x_{max}$$
$$u_{min} \leq u_{j-1} \leq u_{max}$$
$$\theta = [x_{j-1} \quad u_j \quad u_{j+1} \quad \ldots \quad u_{N-1}]^T$$

(4.12)

The use of the original objective function of (4.10) in (4.12) results in a convex problem which may be solved without the need for global optimisation techniques.

**Remark 1.** It should be noted that if the piece-wise affine term $V_{j+1}(x_j)$ had been introduced in (4.12), instead of its expanded form, the problem could be formulated as multiple multi-parametric mixed-integer linear problems, to be solved over the different partitions in which $V_{j+1}(x_j)$ is defined (Borrelli et al., 2005), without the need for global optimisation. However, this choice would limit future extensions of the algorithm to model predictive control problems with quadratic objective function. This limitation is here avoided by considering a convex objective function.

At each iteration, recursion (4.12) is solved and the values of future control inputs are replaced by the solutions obtained in previous iterations.

To illustrate the procedure described in this section, the sub-problem of (4.12) for iteration $j = N$ is shown in (4.13).

$$V_N(\theta) = \min_{u_{N-1}} \|x_N\|_p^1 + \|u_{N-1}\|_R^1$$

s. t.  
$$x_N = A_ix_{N-1} + B_iu_{N-1} + c_i, \forall (x_{N-1}, u_{N-1}) \in P_i$$
$$x_{min} \leq x_N \leq x_{max}$$
$$u_{min} \leq u_{N-1} \leq u_{max}$$
$$\theta = x_{N-1}$$

(4.13)

The conversion of (4.13) to a multi-parametric mixed-integer linear problem is carried out using the method described in §4.1 and §4.2. The resulting problem is of the form (4.14).
\[ V_N(\theta) = \min_{U_{N-1}} \epsilon_N^x + \epsilon_N^u \]
\[ \text{s.t.} \quad x_N = \sum_{i=1}^{p} z_{N-1,i} \]
\[ x_{\min} \leq x_N \leq x_{\min} \]
\[ u_{\min} \leq u_{N-1} \leq u_{\max} \]
\[ -\epsilon_N^x \leq P x_N \leq \epsilon_N^x \]
\[ -\epsilon_N^u \leq R u_{N-1} \leq \epsilon_N^u \]
\[ m\delta_{N-1,i} \leq z_{N-1} \leq M\delta_{N-1,i}, \quad i = 1, \ldots, p \]
\[ z_{N-1} \leq A_i x_{N-1} + B_i u_{N-1} + c_i - M(1 - \delta_{N-1,i}), \quad i = 1, \ldots, p \]
\[ z_{N-1} \geq A_i x_{N-1} + B_i u_{N-1} + c_i - m(1 - \delta_{N-1,i}), \quad i = 1, \ldots, p \]
\[ F_i x_{N-1} + G_i u_{N-1} \leq b_i + L_i (1 - \delta_{N-1,i}), \quad i = 1, \ldots, p \]
\[ 1 \leq \sum_{i=1}^{p} \delta_{N-1,i} \leq 1 \]
\[ \theta = x_{N-1} \]
\[ U_{N-1} = \begin{bmatrix} u_{N-1} & \delta_{N-1} & z_{N-1}^x & \epsilon_N^u & \epsilon_N^u \end{bmatrix}^T \]

The solution of (4.14) is the piece-wise affine function \( U_{N-1} = f(\theta) \), defined over the map of \( n_N \) critical regions, CR\( _N \).

Having computed the solution of iteration \( j = N \), the algorithm proceeds backwards to iteration \( j = N - 1 \), for which the recursive problem (4.12) to be solved is given by (4.15).

\[ V_{N-1}(\theta) = \min_{U_{N-2}} \|x_{N-1}\|_Q^\frac{1}{4} + \|x_N\|_P^\frac{1}{4} + \|u_{N-2}\|_R^\frac{1}{4} + \|u_{N-1}\|_R^\frac{1}{4} \]
\[ \text{s.t.} \quad x_N = A_i x_{N-1} + B_i u_{N-1} + c_i, \quad (x_N, u_{N-1}) \in P_i \]
\[ x_{N-1} = A_i x_{N-2} + B_i u_{N-2} + c_i, \quad (x_{N-2}, u_{N-2}) \in P_i \]
\[ x_{\min} \leq x_{N-1} \leq x_{\max} \]
\[ u_{\min} \leq u_{N-2} \leq u_{\max} \]
\[ \theta = \begin{bmatrix} x_{N-1} & u_{N-1} \end{bmatrix} \]

The reformulation of (4.15) as a multi-parametric mixed integer linear problem is analogous to the reformulation that results in (4.14) and involves introducing the auxiliary variables \( \delta_{N-2}, z_{N-2}, \delta_{N-1}, z_{N-1}, \epsilon_{N-1}^x, \epsilon_N^x, \epsilon_N^u, \epsilon_N^u \).

The solution of (4.15) is given by the piece-wise affine function \( U_{N-2} = f(\theta) \), defined over the map of \( n_{N-1} \) critical regions, CR\( _{N-1} \). The optimal values obtained for the auxiliary variables may be discarded, except for \( z_{N-2}(\theta) \), which is required to compute the complete solution of iteration \( j = N - 1 \).
4.3. Explicit hybrid model predictive control by dynamic programming

The complete solution of iteration $j = N - 1$ consists of the optimal function $u_{N-2} = f(x_{N-2})$, which is obtained by combining the solutions of (4.13) and (4.15).

The optimal solutions of (4.15) and (4.13) for two critical regions in the respective sets, $CR_{N-1}$ and $CR_N$, are affine functions of the form (4.16) and (4.17), respectively.

$$ u_{N-2} = C_1x_{N-2} + C_2u_{N-1} + C_3 \quad (4.16) $$

$$ u_{N-1} = C_4x_{N-1} + C_5 \quad (4.17) $$

The matrices $C_1$, $C_2$, $C_3$, $C_4$, and $C_5$, in (4.16) and (4.17) are linear coefficients of appropriate dimensions.

To obtain the solution $u_{N-2} = f(x_{N-2})$, (4.17) is replaced in (4.16), and $x_{N-1}$ is eliminated by recalling the relation $x_{N-1} = \sum_{i=1}^{p} z_{N-2,i}(\theta)$. As suggested in (Kouramas et al., 2011), the elimination procedure may be efficiently performed using orthogonal projection methods or Fourier-Motzkin elimination (Schrijver, 1998).

The complete map of critical regions corresponding to the solution $u_{N-2} = f(x_{N-2})$ is obtained by performing the elimination procedure for the $n_N \times n_{N-1}$ possible combinations of critical regions of the two stages. This will result in the presence of empty regions in the parameter space that may be detected by performing feasibility tests over the union of the two critical regions being considered.

Note that, given the convexity of $V_N(\theta)$ in (4.13) and $V_{N-1}(\theta)$ in (4.15), the feasible intersection of critical regions is unique, and no overlaps result in the final map of critical regions at each iteration.

The solution of (4.10) is obtained by performing the steps described above for all iterations $j = N, N - 1, \ldots, 1$. The final solution of iteration $j = 1$ corresponds to the optimal solution of (4.10).

The procedure described in this section is summarised in Algorithm 3.
Algorithm 3 Explicit model predictive control of hybrid systems by multi-parametric programming and dynamic programming.

2. Solve stage $N$ of (4.12) and obtain the solution $u_{N-1} = f(x_{N-1})$ defined over $\text{CR}_N$.
3. for $j \leftarrow N - 1, \ldots, 1$ do
   4. $\text{CR}_{\text{temp}} \leftarrow \emptyset$
   6. Solve stage $j$ of (4.12) and obtain solution $\begin{bmatrix} u_{j-1} & z_{j-1} \end{bmatrix}^T = f(x_{j-1}, u_j, \ldots, u_{N-1})$ defined over $\text{CR}_j$.
   7. for $i \leftarrow 1, \ldots, \#\text{CR}_j$ do
      8. for $k \leftarrow 1, \ldots, \#\text{CR}_{j+1}$ do
         9. Test feasibility of $\text{CR}_j^i \cap \text{CR}_{j+1}^k$.
         10. if Intersection is feasible then
             11. Compute stage solution by replacing $u_{j+1}$ in $u_j$.
             12. $\text{CR}_{\text{temp}} \leftarrow \text{CR}_{\text{temp}} \cup (\text{CR}_j^i \cap \text{CR}_{j+1}^k)$
          end if
      end for
   end for
   16. $\text{CR}_j \leftarrow \text{CR}_{\text{temp}}$
18. return List of stage solutions and corresponding maps of critical regions.

Remark 2. The solution produced by Algorithm 3 is the optimal solution of (4.6) and no approximation has been introduced by the use of the approach based on multi-parametric programming and dynamic programming. This results from the convexity of (4.12) and the principle of optimality (Bellman, 1957).

Remark 3. Efficiency of Algorithm 3.
The number of binary nodes involved in the solution of a hybrid explicit model predictive control problem of a piece-wise affine system defined over $p$ partitions using the approach presented in §4.2 is given by (4.18).

$$\#_{\delta} = p^N \quad (4.18)$$

Using Algorithm 3, the number of binary nodes is given by (4.19).

$$\#_{\delta} = \sum_{i=1}^{N} p^{N-i+1} \quad (4.19)$$

It is clear that, even though the dimensionality of the sub-problems (4.12) is reduced, the solution will be arrived at in a less efficient way, due to the higher number of binary nodes involved in Algorithm 3.

As remarked in (Bertsimas and Weismantel, 2005), using exact dynamic programming for large scale optimisation problems involving integer variables is not practical for
most applications, due to the complex nature of the implicit enumeration procedure. Algorithm 3 is presented to illustrate the methodology based on dynamic programming and multi-parametric programming, and will be the subject of future research, possibly involving approximate dynamic programming methods.

4.4 Illustrative example

This section addresses the problem of designing an explicit controller, with a cost function based on the $1$-norm, for a system of two states and one control input described by piece-wise affine dynamics. The control problem, given by the formulation (4.20), is here solved using the conventional explicit model predictive control approach, described in §4.2, and the approach based on dynamic programming and multi-parametric programming, presented in §4.3.

\[
U(\theta) = \arg\min_{u_0, \ldots, u_{N-1}} \|x_N\|^1_p + \|u_{N-1}\|^1_R + \sum_{i=1}^{N-1} \|x_i\|^1_Q + \|u_{i-1}\|^1_R
\]

s. t. \[
x_{k+1} = \begin{cases} 
0.8 \begin{bmatrix} \cos(\frac{\pi}{3}) & -\sin(\frac{\pi}{3}) \\ \sin(\frac{\pi}{3}) & \cos(\frac{\pi}{3}) \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_k & \text{if } [1 \ 0]^T x_k \geq 0 \\
0.8 \begin{bmatrix} \cos(-\frac{\pi}{3}) & -\sin(-\frac{\pi}{3}) \\ \sin(-\frac{\pi}{3}) & \cos(-\frac{\pi}{3}) \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_k & \text{if } [1 \ 0]^T x_k < 0 
\end{cases}
\]

\[
k = 0, \ldots, N-1
\]

\[
\begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq x_k \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \quad k = 1, \ldots, N-1
\]

\[
\begin{bmatrix} -1 \end{bmatrix} \leq u_k \leq \begin{bmatrix} 1 \end{bmatrix}, \quad k = 1, \ldots, N
\]

\[
Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1
\]

\[
\theta = x_0 \in \Theta
\]

(4.20)

As mentioned in §4.1, the piece-wise affine dynamics in (4.20) may be replaced by an equivalent set of linear equalities and inequalities of the form (4.4).

The non-smooth terms in the cost function of (4.20) are handled by introducing auxiliary variables, $\epsilon^x$ and $\epsilon^u$, and inequalities such that the cost function is written as (4.21).
\[
U(\theta) = \arg\min_{u_0,...,u_{N-1}} \|x_N\|_P^1 + \|u_{N-1}\|_R^1 + \sum_{i=1}^{N-1} \|x_i\|_Q^1 + \|u_{i-1}\|_R^1
\]
\[
\Leftrightarrow U(\theta) = \arg\min_{u_0,...,u_{N-1}} \epsilon_{x_N}^x + \sum_{i=1}^{N-1} \epsilon_{i}^x + \epsilon_{i-1}^u
\]
\[
\text{s. t. } \begin{align*}
-\epsilon_{k}^x &\leq Qx_k \leq \epsilon_{k}, & k = 1,\ldots,N-1 \\
-\epsilon_{N}^x &\leq Px_N \leq \epsilon_{N}^x \\
-\epsilon_{k}^u &\leq Ru_k \leq \epsilon_{k}^u, & k = 0,\ldots,N-1
\end{align*}
\]

Replacing (4.21) in (4.20), and using the MLD framework described in §4.1, problem (4.20) is re-written as the equivalent multi-parametric mixed-integer linear programming problem (4.22).
\[ U(\theta) = \arg\min_Z \sum_{i=0}^{N-1} e_i^x + e_i^u \]
\[ \text{s.t.} \quad -e_k^x \leq Q x_k \leq e_k^x, \quad k = 1, \ldots, N - 1 \]
\[ -e_N^x \leq P x_N \leq e_N^x \]
\[ -e_k^u \leq R u_k \leq e_k^u, \quad k = 0, \ldots, N - 1 \]
\[ x_{k+1} = \begin{bmatrix} I^2 & I^2 \end{bmatrix} z_k, \quad k = 1, \ldots, N \]
\[ x_{\min} \leq x_k \leq x_{\max}, \quad k = 1, \ldots, N \]
\[ u_{\min} \leq u_k \leq u_{\max}, \quad k = 0, \ldots, N - 1 \]
\[ Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1 \]
\[ \theta = x_0 \in \Theta \]

\[
\begin{array}{cccccccc}
10.1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 10.1 \\
0 & 10.1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-12.028 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-12.028 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
-12.028 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-12.028 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
12.028 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 12.028 \\
12.028 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 12.028 \\
12.028 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 12.028 \\
12.028 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 12.028 \\
\end{array}
\]

In (4.22), \( Z \triangleq \begin{bmatrix} u_0, z_0, d_0, \ldots, u_{N-1}, z_{N-1}, d_{N-1}, e_N^x, \ldots, e_N^x, e_{N-1}^u, \ldots, e_{N-1}^u \end{bmatrix}^T \) is the optimisation vector and \( I^2 \) represents the \( 2 \times 2 \) identity matrix.

### 4.4.1 Solution using explicit model predictive control

Problem (4.20) may be directly solved using the POP Matlab toolbox (ParOS, 2004) or MPT Matlab toolbox (Kvasnica et al., 2004), which contain an implementation of the method described in 4.4.2.

The map of critical regions for the solution of (4.20) with \( N = 2 \) is presented in Figure 4.1. Some examples of the critical regions and corresponding optimal solutions, \( U(\theta) \), are shown in Table D.1 of Appendix D.
When confronted with the explicit model predictive control example in §2.4 (Figure 2.2), which is of equivalent size but does not include integer variables, it is noticeable that the number of critical regions is similar. There is, however, a more significant discrepancy in the corresponding computational times, which are of 0.1s for the solution of (2.29) and 2.7s for the solution of (4.20).

The increase in computational complexity becomes more evident for larger values of the control horizon. For \( N = 5 \), the computational time required to solve (4.20) is of 433.3s, as opposed to 0.7s for the solution of (2.29). The map of critical regions for the solution of (4.20) with \( N = 5 \) is presented in Figure 4.2. Examples of the expression of the critical regions and corresponding optimal solutions, \( U(\theta) \) are shown in Table D.2 of Appendix D.

### 4.4.2 Solution using dynamic programming and explicit model predictive control

As remarked in §4.3, no approximation is introduced by using the approach based on dynamic programming and multi-parametric programming, and therefore the results shown in §4.4.1 may be replicated using Algorithm 3.

The state trajectories obtained for simulations using the controller designed for \( N = 5 \) with different initial conditions are presented in Figure 4.3. It may be observed that all initial conditions lead to trajectories that converge in the set-point.

For one of the simulations in Figure 4.3, the temporal trajectories of the two components of the state vector are presented in Figure 4.4. The corresponding temporal...
4.4. Illustrative example

Figure 4.2: Map of critical regions for the solution of (4.20) with $N = 5$.

Figure 4.3: State-space trajectories for different initial conditions (■) converging to the set-point (◊).
Figure 4.4: Temporal trajectories of the two components of the system state, $x_1$ and $x_2$.

Figure 4.5: Temporal trajectory of the control input of the system, $u$.

Trajectories of the input variable are presented in Figure 4.5.

For the trajectories shown in Figure 4.4 and Figure 4.5, the switching between the two affine dynamics of (4.20), triggered by the state trajectory, is illustrated in Figure 4.6.

The computational time required to solve problem (4.20) using conventional explicit model predictive control (MP-MILP) and dynamic programming and multi-parametric programming (Dynamic MP-MILP) is presented in Figure 4.7.
To obtain the computational times associated with the conventional explicit model predictive control approach, the one-shoot algorithm of the MPT Matlab toolbox was used. This toolbox also implements an efficient dynamic programming solution based on (Borrelli et al., 2005). However, comparison with this algorithm is not meaningful because, as remarked in §4.3, the objective is to propose an approach that has the potential of being extended for the case of explicit model predictive control problems with quadratic cost function.

As expected, the computational performance of Algorithm 3 is inferior to the conventional explicit model predictive control approach. This fact is attributed to the high number of binary nodes visited during the procedure, given by (4.19).

### 4.5 Concluding remarks

This chapter introduced an algorithm for explicit model predictive control of hybrid systems with linear cost function. The method is based on multi-parametric mixed-integer programming and on the developments in constrained dynamic programming of hybrid linear systems presented in Chapter 3.

The numerical examples in this chapter illustrate the increased complexity inherent to explicit model predictive control problems of hybrid systems, which motivates the pursuit of more efficient solution techniques.

Future research will be focused on improving the efficiency of the algorithm, by exploring approximate dynamic programming solutions, and on extending the approach to the problem of hybrid explicit model predictive control with quadratic cost function.
Figure 4.7: Comparison of computational times required for the solution of problem (4.22) using MP-MILP and dynamic programming based MP-MILP.
This chapter presents a method for deriving robust explicit model predictive controllers for hybrid systems, based on the developments proposed in Chapter 4. The method is used to derive explicit controllers with linear cost function which guarantee feasible operation for the worst-case realisation of the uncertainty in the model.

The approaches most commonly used to model uncertainty in the context of robust explicit model predictive control are presented in §5.1, for the particular case of systems described by piece-wise affine dynamics.

The methodology for designing robust explicit model predictive controllers, based on dynamic programming and multi-parametric programming, is presented in §5.2.

In §5.3, a numerical example is presented that illustrates the need to consider model uncertainty in the control design and demonstrates the closed-loop performance of the controllers designed with the proposed method.
5.1 Uncertainty description for piece-wise affine systems

Several different methods have been used in the literature to model uncertainty in a dynamical system (Bemporad and Morari, 1999b). In the context of robust explicit model predictive control, most approaches focus on uncertainty in the form of additive disturbances or polytopic uncertainty (see Table 1.5 of §1.3.3).

The modelling approaches presented in this section rely on reformulating the state-space model of a generic piece-wise affine system of the form (5.1).

\[ x_{k+1} = A_i x_k + B_i u_k + c_i, \quad \forall (x_k, u_k) \in \mathcal{P}_i \]  \tag{5.1}

In (5.1), \( A_i \in \mathbb{R}^{n \times n} \), \( B_i \in \mathbb{R}^{n \times m} \), and \( c_i \in \mathbb{R}^n \) are uncertain coefficients for which the nominal values are given by \( A_i^0 \), \( B_i^0 \), and \( c_i^0 \). The space defined by the state of the system, \( x_k \in \mathbb{R}^n \), and input of the system, \( u_k \in \mathbb{R}^m \), is divided into \( p \) polyhedral partitions, \( \mathcal{P}_i \), defined as (5.2). It is assumed that the union of partitions, \( \mathcal{P} = \bigcup_{i=1}^{p} \mathcal{P}_i \), contains the origin and that (5.1) is a well-posed piece-wise affine system (as defined in §4.1).

\[ \mathcal{P}_i = \{(x_k, u_k) \mid F_i x_k + G_i u_k \leq b_i, \quad i = 1, \ldots, p\} \]  \tag{5.2}

In the case of additive disturbances (Weinmann, 1991) an additional term, \( w \in \mathbb{R}^n \), is introduced in the piece-wise affine formulation (5.1), as shown in (5.3).

\[ x_{k+1} = A_i x_k + B_i u_k + w_k + c_i, \quad \forall (x_k, u_k) \in \mathcal{P}_i \]  \tag{5.3}

The additive term in (5.3) is usually assumed to be bounded in a known set \( W \), such that \( w_k \in W \).

A different representation of uncertainty, referred to as polytopic uncertainty (Kothare et al., 1994), consists of replacing \( A_i \) and \( B_i \) in (5.1) with the convex hull defined by \( s \) extreme values of the matrices, as shown in (5.4).

\[ [A_i \ B_i] \in \text{Co} \left\{ [A_{i,1} \ B_{i,1}], \ldots, [A_{i,s} \ B_{i,s}] \right\}, \quad i = 1, \ldots, p \]  \tag{5.4}

An alternative way of representing polytopic uncertainty is by considering the matrices \( A_i \) and \( B_i \) as linear combinations of the \( s \) extreme values, as shown in (5.5).

\[ A_i = A_i^0 + \sum_{j=1}^{s} \lambda_{ij}^A A_{ij}, \quad i = 1, \ldots, p \]
\[ B_i = B_i^0 + \sum_{j=1}^{s} \lambda_{ij}^B B_{ij}, \quad i = 1, \ldots, p \]
\[ \sum_{j=1}^{s} \lambda_{ij}^A = 1, \quad \lambda_{ij}^A \geq 0, \quad i = 1, \ldots, p, \quad j = 1, \ldots, s \]
\[ \sum_{j=1}^{s} \lambda_{ij}^B = 1, \quad \lambda_{ij}^B \geq 0, \quad i = 1, \ldots, p, \quad j = 1, \ldots, s \]  \tag{5.5}

The method for explicit robust model predictive control presented in this chapter considers a particular case of (5.4) and (5.5), in which \( A_i \) and \( B_i \) are given by (5.6).
A_i = A_i^0 + \Delta A_i, \quad i = 1, \ldots, p
B_i = B_i^0 + \Delta B_i, \quad i = 1, \ldots, p

(5.6)

The uncertainty matrices, \( \Delta A_i \) and \( \Delta B_i \), in (5.6) are bounded by a percentage \( \gamma \) of the absolute value of the absolute value of the corresponding nominal values, \( A_i^0 \) and \( B_i^0 \). These bounds are expressed by the element-wise inequalities (5.7), where \( (a_i^0)_{j,k} \), \( (b_i^0)_{j,k} \), \( (\delta a_i)_{j,k} \), and \( (\delta b_i)_{j,k} \), represent the element in line \( j \) and column \( k \) of matrices \( A_i^0 \), \( B_i^0 \), \( \Delta A_i \), and \( \Delta B_i \), respectively. For simplicity, the same value of \( \gamma \) is used to describe the uncertainty in \( A_i \) and in \( B_i \).

\[- \gamma |(a_i^0)_{j,k}| \leq (\delta a_i)_{j,k} \leq \gamma |(a_i^0)_{j,k}|, \quad i = 1, \ldots, p; \quad j = 1, \ldots, n; \quad k = 1, \ldots, n
- \gamma |(b_i^0)_{j,k}| \leq (\delta b_i)_{j,k} \leq \gamma |(b_i^0)_{j,k}|, \quad i = 1, \ldots, p; \quad j = 1, \ldots, n; \quad k = 1, \ldots, m\]

(5.7)

5.2 Robust explicit model predictive control for hybrid systems

This section shows how the methodology for designing explicit model predictive controllers for hybrid system, presented in Chapter 4, may be extended to the problem of designing a controller which is immunised against uncertainty in the model.

The explicit model predictive control problem to be solved is of the form (5.8).

\[
U(x_0) = \arg\min_{u_0, \ldots, u_{N-1}} \left\| x_N \right\|_p^1 + \left\| u_{N-1} \right\|_R^1 + \sum_{k=1}^{N-1} \left\| x_k \right\|_Q^1 + \left\| u_k \right\|_R^1
\]

s. t. \( x_{k+1} = A_i x_k + B_i u_k, \forall (x_k, u_k) \in P_i \)

\[
\begin{align*}
x_{\text{min}} & \leq x_{k+1} \leq x_{\text{max}} \\
u_{\text{min}} & \leq u_k \leq u_{\text{max}}, \quad k = 0, \ldots, N-1, \quad i = 1, \ldots, p
\end{align*}
\]

(5.8)

In (5.8), the matrices \( A_i \) and \( B_i \) are considered to be affected by an uncertain variation of the corresponding nominal values, \( A_i^0 \) and \( B_i^0 \), as shown in (5.6). The value of \( \gamma \) is assumed to be fixed, and therefore the bounds on the uncertain variations, \( \Delta A_i \) and \( \Delta B_i \), are known and given by (5.7). The objective is to design a controller that guarantees feasible operation and satisfies the problem constraints for all values of \( A_i \) and \( B_i \) in the range (5.7).

Remark 1. The methodology presented in this section may be easily extended to include the variation \( \gamma \) as a parameter in the solution of (5.8). One of the practical purposes of this extension would be to decrease the conservativeness of the control actions in regions of the state space for which the model is known to be more accurate.

In §4.3, it was shown how the sequential structure of (5.8) may be explored by using the principle of optimality (Bellman, 1957) to obtain the equivalent recursive representation (5.9).
By defining a parameter vector consisting of the initial state of each stage and control inputs of future stages, the recursion associated with (5.8) may be reformulated as a multi-parametric mixed-integer problem of the form (5.9).

$$V_j(\theta) = \min_{u_{j-1}} \|x_N\|_P^1 + \|u_{N-1}\|_R^1 + \sum_{k=j}^{N-1} \|x_j\|_Q^1 + \|u_{j-1}\|_R^1$$

s.t. $x_j = A_i x_{j-1} + B_i u_{j-1} + c_i, \forall (x_{j-1}, u_{j-1}) \in P_i$

$$x_{min} \leq x_j \leq x_{max}$$

$$u_{min} \leq u_{j-1} \leq u_{max}$$

$$\theta = [x_{j-1}, u_j, u_{j+1}, \ldots, u_{N-1}]^T$$

(5.9)

The solution of recursion (5.9) is obtained by solving each iteration, proceeding backwards, for $j = N, \ldots, 1$. The choice of objective function and vector of parameters in (5.9) leads to a convex problem and avoids the need for global optimisation techniques that usually arises in this type of recursive problem. (Faísca et al., 2008; Kouramas et al., 2012).

The problem to be solved in the first iteration of (5.9), $j = N$, is given by (5.10).

$$V_N(\theta) = \min_{u_{N-1}} \|x_N\|_P^1 + \|u_{N-1}\|_R^1$$

s.t. $x_N = A_i x_{N-1} + B_i u_{N-1} + c_i, \forall (x_{N-1}, u_{N-1}) \in P_i$

$$x_{min} \leq x_N \leq x_{max}$$

$$u_{min} \leq u_{N-1} \leq u_{max}$$

$$\theta = x_{N-1}$$

(5.10)

Following the procedure described in §4.1 and §4.3, (5.10) may be re-written as a multi-parametric mixed-integer linear problem of the form (5.11).
\[ V_N(\theta) = \min_{U_{N-1}} \epsilon_N^x + \epsilon_{N-1}^u \]  
\( \text{s.t. } x_N = \sum_{i=1}^{p} z_{N-1,i} \)
\( x_{\min} \leq x_N \leq x_{\max} \)
\( u_{\min} \leq u_{N-1} \leq u_{\max} \)
\( -\epsilon_N^x \leq Pu_N \leq \epsilon_N^x \)
\( -\epsilon_{N-1}^u \leq Ru_{N-1} \leq \epsilon_{N-1}^u \)
\( m \delta_{N-1,i} \leq z_{N-1,i} \leq M \delta_{N-1,i}, \quad i = 1, \ldots, p \)
\( z_{N-1,i} \leq A_i x_{N-1} + B_i u_{N-1} + c_i - M(1 - \delta_{N-1,i}), \quad i = 1, \ldots, p \)
\( z_{N-1,i} \geq A_i x_{N-1} + B_i u_{N-1} + c_i - m(1 - \delta_{N-1,i}), \quad i = 1, \ldots, p \)
\( f_i x_{N-1} + g_i u_{N-1} \leq b_i + l_i (1 - \delta_{N-1,i}), \quad i = 1, \ldots, p \)
\( \sum_{i=1}^{p} \delta_{N-1,i} \leq 1 \)
\( \theta = x_{N-1} \)
\( U_{N-1} = \begin{bmatrix} u_{N-1} & \delta_{N-1} & z_{N-1} & \epsilon_N^x & \epsilon_{N-1}^u \end{bmatrix}^T \)

Taking into account the uncertainty description given by (5.6) and (5.7), it may be noted that the objective function (5.11) and constraints (5.12)-(5.20) are affected by different realisations of the uncertain matrices \( A_i \) and \( B_i \) in the range (5.7).

Following the lines of (Pistikopoulos et al., 2007b; Kouramas et al., 2012), the constraint (5.13) is immunised against the worst-case realisation of the uncertainty, while the nominal values \( A_i^0 \) and \( B_i^0 \) are used for the terms related to the objective function (5.12).

A piece-wise affine representation of (5.13) is given by (5.24).

\[ x_{\min} \leq A_i x_{N-1} + B_i u_{N-1} \leq x_{\max}, \quad i = 1, \ldots, p \]

Taking advantage of the binary variable \( \delta_{N-1,i} \), (5.24) may be re-written in the form (5.25).

\[ \delta_{N-1,i} = 1 \iff \begin{cases} x_{\min} \leq A_i x_{N-1} + B_i u_{N-1} \leq x_{\max}, \\ (x_{N-1}, u_{N-1}) \in P_i \end{cases}, \quad i = 1, \ldots, p \]
\( \sum_{i=1}^{p} \delta_{N-1,i} = 1 \)

The matrices \( A_i \) and \( B_i \) in (5.25) are replaced by the expression (5.6), which describes the type of uncertainty being considered, resulting in (5.26).
\[
\delta_{N-1,i} = 1 \iff \begin{cases} 
  x_{\text{min}} \leq A_i^0 x_{N-1} + B_i^0 u_{N-1} + \Delta A_i x_{N-1} + \Delta B_i u_{N-1} \leq x_{\text{max}} \\
  (x_{N-1}, u_{N-1}) \in \mathcal{P}_i, \quad i = 1, \ldots, p
\end{cases}
\]

\[
\sum_{i=1}^{p} \delta_{N-1,i} = 1
\]

Since the objective function terms were assumed to have the nominal values, the robustification suggested in (Ben-Tal and Nemirovski, 2000) may be applied.

This step consists of choosing the values of the terms \(\Delta A_i x_{N-1}\) and \(\Delta B_i u_{N-1}\) corresponding to the worst-case realisation of the uncertainty, so that the constraints in (5.26) are satisfied for all values in the range (5.7).

For simplicity, the robustification step is shown only for the constraint corresponding to the upper bound of (5.26).

It is straightforward to show (Kouramas et al., 2012) that the worst-case realisations of \(\Delta A_i x_{N-1}\) and \(\Delta B_i u_{N-1}\) that affect the upper bound of (5.26) are given by (5.27).

\[
\Delta A_i x_{N-1} = \gamma \left| A_i^0 \right| |x_{N-1}| \\
\Delta B_i u_{N-1} = \gamma \left| B_i^0 \right| |u_{N-1}|
\]

Replacing (5.27) in (5.26), the robust counterpart (Ben-Tal and Nemirovski, 2000) of the upper bound of (5.26) is given by (5.28).

\[
\delta_{N-1,i} = 1 \iff \begin{cases} 
  A_i^0 x_{N-1} + B_i^0 u_{N-1} + \gamma \left| A_i^0 \right| |x_{N-1}| + \gamma \left| B_i^0 \right| |u_{N-1}| \leq x_{\text{max}} \\
  (x_{N-1}, u_{N-1}) \in \mathcal{P}_i, \quad i = 1, \ldots, p
\end{cases}
\]

\[
\sum_{i=1}^{p} \delta_{N-1,i} = 1
\]

If the constraint in (5.28) is satisfied, it is guaranteed that the upper bound of (5.26) is satisfied for all realisations of the uncertainty in (5.7).

The non-smooth terms in (5.28), \(|x_{N-1}|\) and \(|u_{N-1}|\), are handled by introducing auxiliary variables and linear inequalities, as shown in (5.29).

\[
\delta_{N-1,i} = 1 \iff \begin{cases} 
  A_i^0 x_{N-1} + B_i^0 u_{N-1} + \gamma \left| A_i^0 \right| w_{N-1}^x + \gamma \left| B_i^0 \right| w_{N-1}^u \leq x_{\text{max}} \\
  -w_{N-1}^x \leq x_{N-1} \leq w_{N-1}^x \\
  -w_{N-1}^u \leq u_{N-1} \leq w_{N-1}^u \\
  (x_{N-1}, u_{N-1}) \in \mathcal{P}_i, \quad i = 1, \ldots, p
\end{cases}
\]

\[
\sum_{i=1}^{p} \delta_{N-1,i} = 1
\]

As shown in §4.1, the logical proposition implicit in (5.29) may be equivalently formulated as a set of linear equalities and inequalities. Replacing the resulting set in (5.11)-(5.23), a multi-parametric mixed-integer linear programming problem is obtained, for which the solution may be obtained using the methods described in §3.1.
5.2. Robust explicit model predictive control for hybrid systems

The solution of iteration \( j = N \) corresponds to the piece-wise affine function \( U_{N-1} = f(\theta) \), defined over the map of critical regions \( CR_N \in \mathbb{R}^{nN} \).

Having computed the solution of iteration \( j = N \), the recursive procedure continues to iteration \( j = N - 1 \), for which the sub-problem of (5.9) to be solved is given by (5.30).

\[
V_{N-1}(\theta) = \min_{u_{N-2}} \|x_{N-1}\|_Q^{1} + \|x_{N}\|_P^{1} + \|u_{N-2}\|_R^{1} + \|u_{N-1}\|_R^{1} \\
\text{s.t.} \quad x_N = A_i x_{N-1} + B_i u_{N-1} + c_i, \forall (x_{N-1}, u_{N-1}) \in P_i \\
\quad x_{N-1} = A_i x_{N-2} + B_i u_{N-2} + c_i, \forall (x_{N-2}, u_{N-2}) \in P_i \\
\quad x_{\min} \leq x_{N-1} \leq x_{\max} \\
\quad u_{\min} \leq u_{N-2} \leq u_{\max} \\
\quad \theta = x_{N-2}
\]

(5.30)

The reformulation of (5.30) as a mixed-integer linear programming problem which takes into account the worst-case uncertainty of the system matrices is analogous to the procedure shown for iteration \( j = N \).

The solution of (5.30) is given by the piece-wise affine function \( U_{N-2} = f(\theta) \), defined over the map of critical regions \( CR_{N-1} \in \mathbb{R}^{nN-1} \).

The complete solution of iteration \( j = N - 1 \) consists of the optimal function \( u_{N-2} = f(x_{N-2}) \), which is obtained by combining the solutions of (5.10) and (5.30).

The optimal solutions of (5.30) and (5.10) for two critical regions in the respective sets, \( CR_{N-1} \) and \( CR_N \), are affine functions of the form (5.31) and (5.32), respectively.

\[
u_{N-2} = A_1 x_{N-2} + B_1 u_{N-1} + C_1 \]

(5.31)

\[
u_{N-1} = A_2 x_{N-1} + C_2 \]

(5.32)

The matrices \( A_k, B_k, C_k \) in (5.31) and (5.32) are linear coefficient of appropriate dimensions.

To obtain the solution \( u_{N-2} = f(x_{N-2}) \), (5.32) is replaced in (5.31), and \( x_{N-1} \) is eliminated by recalling the relation \( x_{N-1} = \sum_{i=1}^{p} z_{N-2,i}(\theta) \). As suggested in (Kouramas et al., 2011), the elimination procedure may be efficiently performed using orthogonal projection methods or Fourier-Motzkin elimination (Schrijver, 1998).

The complete map of critical regions corresponding to the solution \( u_{N-2} = f(x_{N-2}) \) is obtained by performing the elimination procedure for the \( n_N \times n_{N-1} \) possible combinations of critical regions of the two stages. This will result in the presence of empty regions in the parameter space that may be detected by performing feasibility tests over the union of the two critical regions being considered.

Note that, given the convexity of \( V_N(\theta) \) in (5.10) and \( V_{N-1}(\theta) \) in (5.30), the feasible intersection of critical regions is unique, and no overlaps result in the final map of critical regions at each iteration.
The solution of (5.8) is obtained by performing the steps described above for all iterations $j = N, N - 1, \ldots, 1$. The final solution of iteration $j = 1$ corresponds to the optimal solution of (5.8).

The procedure described in this section is summarised in Algorithm 4.

**Algorithm 4** Robust explicit model predictive control of hybrid systems by multi-parametric programming and dynamic programming.

1: Reformulate stage $N$ of (5.9) as a multi-parametric mixed-integer linear programming problem.
2: Reformulate the constraints of stage $N$, taking into account the worst-case realisation of the uncertainty, as shown in (5.28).
3: Solve stage $N$ of (5.9) and obtain the solution $u_{N-1} = f(x_{N-1})$ defined over $CR_N$.
4: for $j \leftarrow N - 1, \ldots, 1$ do
5:    $CR_{temp} \leftarrow \emptyset$
6:    Reformulate stage $j$ of (5.9) as a multi-parametric mixed-integer linear problem.
7:    Reformulate the constraints of stage $j - 1$, taking into account the worst-case realisation of the uncertainty, as shown in (5.28).
8:    Solve stage $j$ of (5.9) and obtain solution $\begin{bmatrix} u_{j-1} & z_{j-1} \end{bmatrix}^T = f(x_{j-1}, u_j, \ldots, u_{N-1})$ defined over $CR_j$.
9:    for $i \leftarrow 1, \ldots, \#CR_j$ do
10:       for $k \leftarrow 1, \ldots, \#CR_{j+1}$ do
11:          Test feasibility of $CR_i^j \cap CR_{j+1}^k$.
12:             if Intersection is feasible then
13:                Compute stage solution by replacing $u_{j+1}$ in $u_j$.
14:                $CR_{temp} \leftarrow CR_{temp} \cup (CR_i^j \cap CR_{j+1}^k)$
15:             end if
16:       end for
17:    end for
18:    $CR_j \leftarrow CR_{temp}$
19: end for
20: return List of stage solutions and corresponding maps of critical regions.
5.3. **Illustrative example**

In this section, the problem of designing an explicit controller for a piece-wise affine system, solved in Chapter 4 for the nominal case, is revisited for the case where the model is affected by uncertainty. The problem to be solved, given in (5.33), involves a system with two states and one control input.

It is assumed that the model matrices, $A_i$ and $B_i$, are affected by the same percentage of uncertainty, given by the constant value $\gamma$.

$$
U(\theta) = \arg\min_{u_0, \ldots, u_{N-1}} \|x_N\|_p^1 + \|u_{N-1}\|_R^1 + \sum_{k=1}^{N-1} \|x_k\|_Q^1 + \|u_{k-1}\|_R^1
$$

s.t. $x_{k+1} = \begin{cases} 
0.8A_1x_k + B_1u_k & \text{if } \begin{bmatrix} 1 & 0 \end{bmatrix} x_k \geq 0 \\
0.8A_2x_k + B_2u_k & \text{if } \begin{bmatrix} 1 & 0 \end{bmatrix} x_k < 0 
\end{cases}, \quad k = 0, \ldots, N-1$

$$
\begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq x_k \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \quad k = 1, \ldots, N-1
$$

$$
\begin{bmatrix} -1 \\ \end{bmatrix} \leq u_k \leq \begin{bmatrix} 1 \\ \end{bmatrix}, \quad k = 1, \ldots, N
$$

$$
Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1
$$

(5.33)

$$
A_1 = \begin{bmatrix} \cos(\frac{\pi}{3}) & -\sin(\frac{\pi}{3}) \\ \sin(\frac{\pi}{3}) & \cos(\frac{\pi}{3}) \end{bmatrix} + \Delta A_1, \quad B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \Delta B_1
$$

$$
A_2 = \begin{bmatrix} \cos(-\frac{\pi}{3}) & -\sin(-\frac{\pi}{3}) \\ \sin(-\frac{\pi}{3}) & \cos(-\frac{\pi}{3}) \end{bmatrix} + \Delta A_2, \quad B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \Delta B_2
$$

$$
- \gamma |A_i^0| \leq \Delta A_i \leq \gamma |A_i^0|, \quad - \gamma |B_i^0| \leq \Delta B_i \leq \gamma |B_i^0|, \quad i = 1, 2
$$

$$
\theta = x_0 \in \Theta
$$

To demonstrate the importance of considering the model uncertainty in the design of a controller, the nominal controller derived in §4.4 with $N = 5$ has been used in closed-loop simulations where the model is contaminated with 10% of uncertainty. The simulations performed with different initial states are presented in Figure 5.1.

While some initial conditions in Figure 5.1 result in trajectories that converge to the set-point, it is noticeable that an initial condition near the boundaries of the feasible region results in an unexpected trajectory. This is due to the mismatch between the
model of the real system and the model used for predicting the state trajectory. The result is violation of constraints and infeasible operation of the controller.

The map of critical regions obtained for a controller designed using the method presented in §5.2, with $\gamma = 10\%$, is illustrated in Figure 5.2 and Figure 5.3 for $N = 2$ and $N = 5$, respectively.

When compared to Figure 4.1 and Figure 4.2, it is noticeable that the feasible space in the maps of critical regions of Figure 5.2 and Figure 5.3 is smaller. The design of the controller using a robust formulation puts in evidence regions in the parameter space that would result in infeasible operation if the mismatch between the dynamics of the system and the uncertain model were of a value up to $\gamma$. Therefore, the robust controller provides information that would prevent the operation from being started at a point too close to the boundary on the state variable, such as the initial condition shown in Figure 5.1.

The shrinking of the feasible space of the critical regions shown in Figure 5.2 and Figure 5.3 becomes more evident for higher values of $\gamma$. This is illustrated in Figures 5.4a-5.4c, which show the map of critical regions corresponding to a robust controller with $N = 2$ for $\gamma = 10\%$, $\gamma = 20\%$, and $\gamma = 30\%$.

The performance of the robust controller derived with $N = 5$ has been tested for different initial conditions, using a model contaminated with 10% of uncertainty to simulate the dynamics of the real system. The results of the simulations are shown in Figure 5.5.

For one of the initial conditions depicted in Figure 5.5, the temporal trajectories of the two states of the system, $x_1$ and $x_2$, are shown in Figure 5.6. The corresponding temporal trajectory of the input of the system is shown in Figure 5.7.
Figure 5.2: Map of critical regions for the solution of (5.33) with $N = 2$.

Figure 5.3: Map of critical regions for the solution of (5.33) with $N = 5$. 
Figure 5.4: Map of critical regions for a robust controller with $N = 2$ for different values of $\gamma$: (a) $\gamma = 10\%$; (b) $\gamma = 10\%$; (c) $\gamma = 30\%$.

Figure 5.5: State-space trajectories for different initial conditions (■) converging to the set-point (♦).

The switching between the two affine dynamics of the system, corresponding to the trajectories shown in Figure 5.6 and Figure 5.7 is presented in Figure 5.8.

When confronted with Figure 4.4 and Figure 4.5, which refer to the closed-loop performance of a nominal controller, Figure 5.6 and Figure 5.7 show a different trajectory for the same initial conditions. This is attributed to the different dynamics used for the simulation of the real system and to the compromise in optimality that results from using a robust control approach based on the worst-case realisation of the uncertainty.

The computation time required to derive robust controllers for different values of $N$ and $\gamma$ has been compared to the time required to derive nominal controllers. The results indicate that little overhead is introduced by considering the worst-case realisation of the uncertainty, and the obtained computational times closely follow the trend shown in Figure 4.7 (Dynamic mp-milp).
5.3. **Illustrative example**

Figure 5.6: Temporal trajectories of the two components of the system state, $x_1$ and $x_2$.

Figure 5.7: Temporal trajectory of the control input of the system, $u$. 
This chapter presented a method for designing robust explicit model predictive controllers for hybrid systems.

Based on the developments in explicit model predictive control for hybrid system by dynamic programming, presented in Chapter 4, the control problem is dissembled into a series of multi-parametric mixed-integer linear sub-problems that are solved sequentially. At each stage of the recursion, the constraints of the sub-problem are reformulated to take into account the presence of uncertainty in the dynamical model of the system.

The method was presented for the case of uncertainty being described by a bounded variation of the matrices of the dynamical model of the system and may be extended to other uncertainty descriptions such as polytopic or additive uncertainty.

The proposed approach has been illustrated with a numeric example concerning the design of a robust explicit model predictive controller with linear cost function for a piece-wise affine system.
Chapter 6

Model reduction and explicit nonlinear model predictive control

This chapter presents a method for nonlinear explicit model predictive control that combines model reduction techniques and nonlinear multi-parametric programming. The proposed method enables the full cycle of development of explicit controllers from high-fidelity models presented in §1.3.

The basic concepts of nonlinear multi-parametric programming and nonlinear explicit model predictive control are presented in §6.1. An algorithm for nonlinear explicit model predictive control based on sensitivity analysis is also presented in this section.

In §6.2, a nonlinear model order reduction technique based on balancing of empirical gramians is described in detail. This section also provides an overview of a meta-modelling based model approximation technique.

The combined use of model reduction techniques and nonlinear multi-parametric programming for nonlinear explicit model predictive control is demonstrated for two examples in §6.3. The methodology describes a step-by-step approach to deriving explicit model predictive controllers for a nonlinear system of a distillation column and a train of CSTRs.

1The material presented in this chapter has been published (Rivotti et al., 2011).
6.1 Explicit nonlinear model predictive control

6.1.1 Nonlinear multi-parametric programming and model predictive control

A general formulation of a nonlinear multi-parametric problem, with a set of parameters \( \theta \in \Theta \), has the form of (6.1).

\[
\begin{align*}
    z(\theta) &= \min_{x,y} f(x, y, \theta) \\
    \text{s.t.} & \quad g(x, y, \theta) \leq 0 \\
    & \quad h(x, y, \theta) = 0 \\
    & \quad \theta \in \Theta
\end{align*}
\]  (6.1)

In (6.1), \( z(\theta) \) is the optimal value of the cost function, \( f \), evaluated at the optimal set of decision variables which may be continuous, \( x \), or discrete, \( y \). The problem is subject to a set of inequality and equality parametric constraints, \( g \) and \( h \), respectively, which may be nonlinear.

The solution of problem (6.1) comprises (a) the optimal cost function, \( z^*(\theta) \), and the corresponding optimal decision variables, \( x^*(\theta) \) and \( y^*(\theta) \), and (b) the map of regions in the parameter space (critical regions) for which the optimal functions are valid.

By considering the input variables corresponding to a plant model, as the vector of optimisation variables, and the initial system states as the vector of parameters, a constrained MPC problem may be formulated analogously to (6.1) (Pistikopoulos et al., 2002).

Consider the discrete-time dynamic system with an equilibrium point \( f(0,0) = 0 \) given by (6.2).

\[
\begin{align*}
    x_{k+1} &= f(x_k, u_k) \\
    y_k &= g(x_k, u_k)
\end{align*}
\]  (6.2)

In (6.2), \( f(x_k, u_k) \) describes the evolution of the vector of system states, \( x \), for all instances \( k \geq 0 \). The vectors \( u \) and \( y \) correspond to the control inputs and system outputs, respectively. An optimal control problem for system (6.2) has the form of (6.3).

\[
\begin{align*}
    z(\theta) &= \min_{u} \|x_N\|^b_p + \|u_{N-1}\|^b_R + \sum_{k=1}^{N-1} \|x_k\|^b_Q + \|u_{k-1}\|^b_R \\
    \text{s.t.} & \quad x_{k+1} = f(x_k, u_k), k = 0, 1, \ldots, N - 1 \\
    & \quad y_k = g(x_k, u_k), k = 0, 1, \ldots, N - 1 \\
    & \quad A_i u \leq b_i, i = 1, \ldots, p \\
    & \quad \theta = x_0
\end{align*}
\]  (6.3)
In (6.3), \( Q \succeq 0 \) and \( R \succ 0 \) are cost matrices related to the states and inputs, respectively and \( P \succeq 0 \) corresponds to the terminal cost matrix; \( N \) represents the finite prediction horizon of the problem; the constraints \( Au \leq b \) usually refer to operational or safety restrictions on the control inputs. The choice of the norm \( b = 1 \) or \( b = \infty \) defines a linear cost function, while the choice \( b = 2 \) defines a quadratic cost function. Even though problem (6.3) only involves constraints on the control input, \( u \), the presentation in this chapter could be directly generalised to problems with constraints involving the state, \( x \).

The optimal control problem (6.3) may be directly reformulated similarly to (6.1) as shown in (6.4).

\[
\begin{align*}
z(\theta) &= \min_u f(u, x_0) \\
\text{s.t. } & H_j(u, x_0) = 0, j = 1, \ldots, q \\
& \quad A_i u \leq b_i, i = 1, \ldots, p \\
& \quad \theta = x_0 \in \mathcal{X} \\
\end{align*}
\]

(6.4)

In (6.4), \( \mathcal{X} \) corresponds to a feasible set of initial states, \( x_0 \), for which the explicit solution should be obtained. The procedure for solving problem (6.4) is based on the principles of parametric nonlinear programming for which the main concepts are outlined in §2.1 (Theorem 1 and Corollary 1).

As mentioned in §2.1, the critical region in the parameter space, \( \mathcal{CR} \), where each first-order estimation of the solution of (6.4) is valid, is obtained by enforcing the feasibility and optimality conditions (6.5) (Dua et al., 2002; Bemporad et al., 2002a).

\[
\mathcal{CR} = \left\{ x_0 \mid \hat{A} u(x_0) \leq \hat{b}, H(u(x_0), x_0) = 0, \hat{\lambda}(x) \geq 0, \mathcal{CR}_I \right\}
\]

(6.5)

In (6.5), \( \hat{A} u(x_0) \leq \hat{b} \) corresponds to inactive inequality constraints, \( \hat{\lambda} \) are the Lagrange multipliers corresponding to the active inequality constraints and \( \mathcal{CR}_I \) corresponds to the initial region in the parameter space.

It may be noted that if \( H(u(x_0), x_0) \) in (6.5) is nonlinear, the resulting critical regions are nonlinear, and often non-convex. Given that, for implementation purposes, it is practical to have a set of convex critical regions, the current methods for nonlinear MPMPC focus on approximate algorithms (Johansen, 2002; Sakizlis et al., 2007; Domínguez and Pistikopoulos, 2011).

### 6.1.2 **NLSENS algorithm for nonlinear MPMPC**

The approximate algorithm for nonlinear MPMPC presented by Domínguez et al. (Domínguez et al., 2010) is based on the sensitivity analysis results of Fiacco (Fiacco, 1976).

As mentioned above, if the nonlinear equalities \( H(u(x_0), x_0) \) of problem (6.4) are replaced in the feasibility and optimality conditions (6.5), the resulting critical regions
Chapter 6. Model reduction and explicit nonlinear model predictive control

are nonlinear, and possibly non-convex. The main idea of the algorithm is to replace the nonlinear equalities \( H(u(x_0), x_0) \) by a linearisation around a local solution of the corresponding NLP problem.

A first-order approximation of \( H(u(x_0), x_0) \) around a local NLP solution, \( u_0 \), is given by (6.6).

\[
H(u_0, x_0) + \nabla_u H(u_0, x_0)(u - u_0) = 0
\] (6.6)

Replacing (6.6) in (6.5) gives an approximate representation that will generate a convex critical region, \( CR_0 \), defined as (6.7).

\[
CR_0 = \{ x_0 \mid \Psi x_0 \leq \psi \}
\] (6.7)

The remaining region in the parameter space, \( CR_I - CR_0 \), may be partitioned using the procedure presented in (Dua and Pistikopoulos, 2000). The algorithm is then repeated for every new critical region, until the entire initial parameter space is explored.

The algorithm is summarised below.

**Algorithm 5 Nonlinear sensitivity based algorithm (nlsens).**

1: Define initial region, \( CR_I \), a list of regions to be explored, \( CR \), and a list of optimal critical regions, \( CR^* \).
2: Set \( CR = CR_I \).
3: Select \( x_0^0 \) from \( CR \).
4: Solve NLP at \( x_0^0 \in CR \) and record solution \( u_0 \).
5: Compute first-order approximate solution (6.6) in the neighbourhood of \( x_0^0 \).
6: Replace the nonlinear equalities by the corresponding linearisation (6.6) around \( u_0 \).
7: Obtain the approximate region \( CR_0 \) using (6.7) and add \( CR_0 \) to \( CR^* \).
8: Set \( CR = CR - CR_0 \).
9: Partition \( CR \) using the method in (Dua and Pistikopoulos, 2000) and collect the generated critical regions.
10: Repeat from Step 3 until \( CR = \emptyset \).
11: return Union of all regions in \( CR^* \) along with the corresponding optimal solutions.

A comparison of the closed-loop control performance for this algorithm and other approaches may be found in (Domínguez et al., 2010).

6.2 Background on model reduction

6.2.1 Nonlinear model reduction - balancing of empirical gramians

Model reduction based on balancing of gramians is a well established model order reduction technique for linear systems (Samar et al., 1995; Skogestad and Postlethwaite, 2005).
6.2. Background on model reduction

However, for the case of nonlinear systems, the methodology is not directly applicable. Hahn and Edgar (Hahn and Edgar, 2002) proposed an alternative technique based on the concept of empirical gramians (Moore, 1981; Lall et al., 1999) for nonlinear control-affine systems. These systems are represented by the set of dynamic equations (6.8).

\[
x_{k+1} = f(x_k) + g(x_k)u_k \\
y_k = h(x_k)
\] (6.8)

In (6.8), \(f, g,\) and \(h\) are functions of class \(C^\infty\), \(f(0) = 0\) and \(g(0) = 0\).

Remark 1. The systems considered in the examples of §6.3.1 and §6.3.2 are nonlinear control-affine systems.

Even though the methodology based on empirical gramians could be applied to an arbitrary nonlinear system, Hahn and Edgar point out that it is limited to control-affine systems, because only for these it is possible to calculate impulse response behaviour (Hahn and Edgar, 2002).

The empirical controllability gramian, \(W_c\), and empirical observability gramian, \(W_o\), of a system are determined using the definitions (6.9) and (6.10).

\[
W_c = \sum_{i=1}^{r} \sum_{m=1}^{s} \sum_{l=1}^{p} \frac{1}{rsc_m} \int_0^\infty (x^l_{il} t - x^l_{0i}) (x^l_{il} t - x^l_{0i})^T dt \\
W_o = \sum_{i=1}^{r} \sum_{m=1}^{s} \frac{1}{rsc_m} \int_0^\infty T_1 (y^l_{il} t - y^l_{0i}) (y^l_{il} t - y^l_{0i})^T T_1^T dt
\] (6.9) (6.10)

In (6.9) and (6.10), \(x^l_{il}\) is the system state corresponding to the impulse input \(u_i = c_m T_1 e_1 \delta_t + u_0\), and \(y^l_{il}\) is the output corresponding to the initial condition \(x_0 = c_m T_1 e_1 + x_0\). \(x^l_{0i}\) and \(y^l_{0i}\) refer to the steady-state of the system and corresponding output, respectively.

The balanced form of (6.9) and (6.10) is obtained by finding the transformation matrix, \(T\), for which the relation (6.11) holds.

\[
\tilde{W}_c = \tilde{W}_o = \Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n \end{bmatrix}, \quad \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0
\] (6.11)

In (6.11), \(\sigma_i\) denotes the Hankel singular values.

The empirical gramians (6.11) relate to (6.9) and (6.10) according to the relation (6.12).

\[
\tilde{W}_c = TW_c T^T \\
\tilde{W}_o = (T^{-1})^T W_o T^{-1}
\] (6.12)
A balanced form of the system (6.8) may then be obtained by applying the transformation (6.13).

\[ \bar{x} = Tx \]  

(6.13)

A transformed system of the form (6.14) is then obtained.

\[ \bar{x}_{k+1} = T f(T^{-1}\bar{x}_k) \] 
\[ y_k = h(T^{-1}\bar{x}_k) \]  

(6.14)

(6.15)

From (6.13) it may be noticed that, due to the transformation, the state variables, \( \bar{x} \), in the reduced space do not hold the original physical meaning.

From the balanced gramians of the system, \( \Sigma \), it may be concluded which states, \( \bar{x} \), contribute more to the dynamic behaviour of the system - states corresponding to higher Hankel singular values have a greater contribution. The order of the model may then be reduced using techniques such as truncation or residualisation (Skogestad and Postlethwaite, 2005).

In this work, the order of the nonlinear models was reduced using residualisation, which typically provides better results than reduction by truncation (Hahn and Edgar, 2002). In this technique, the derivatives of the less significant states are set to zero and the system is then described by the dynamic equations of the remaining states.

**Remark 2.** Even though the order of the system is reduced, this does not imply that the complexity of the model is equally reduced. In fact, since the dynamics of the system are projected on only a few states, the resulting model is usually more dense than the original. However, for the implementation of the nonlinear multi-parametric algorithm (Domínguez et al., 2010), it is possible to use such reduced models, as long as the number of reduced states is small enough and the reduced model is convex.

### 6.2.2 Meta-modelling based model approximation

This section presents a brief overview of a recently proposed model approximation method based on meta-modelling (Lambert et al., 2011). Further details are presented in Appendix G. This approach is used in the example of §6.3.1 for comparison purposes.

The approximation method is inspired by the use of high-dimensional model representation (HDMR) (Li et al., 2002). Contrary to the conventional HDMR approach, the control variables are discretized over the time horizon, as a result of which the nonlinear polynomial terms are eliminated and only linear terms are used.

The method is summarized in the following steps.

**Step 1.** Define bounds for the states and control variables.

**Step 2.** Define a control horizon and sampling time.

**Step 3.** Perform multiple simulations exploring the space of controls and initial states.
Step 4. Build affine expressions for the system output, $y$, as a function of the control inputs and initial states, for every time point along the horizon.

The final model of the system comprises the various independent affine expressions determined in Step 4 and is designated as a meta-model. These affine expressions are of the form (6.16).

$$y_j = a_0^j + \sum_{i=1}^{n} a_i^j x_0^i + \sum_{k=1}^{j} \sum_{l=1}^{m} \alpha^j_{ik} u_k$$

(6.16)

The coefficient $a_0^j$ in (6.16) represents the average value of $y_j$. The remaining coefficients, $a_i^j$ and $\alpha^j_{ik}$, are determined by sampling the space of initial states and controls and using Monte Carlo integration schemes. To obtain uniform sampling it is advised to use low discrepancy sequences such as Sobol sequences (Sobol’, 1967).

The main advantages of this method are listed below:

1. The use of the approximations in a receding horizon fashion does not require an arbitrarily large number of model approximations as it was the case in previous applications (Jeffrey et al., 1999).

2. The low discrepancy sampling of the space of initial states and control inputs allows the a posteriori calculation of the average and maximum model mismatch. Statistics on the error allow to obtain an a priori error bound which may then be used for robust control applications.

3. Even though the computation of the model is expensive, it only needs to be carried out once, offline. On the other hand, simulations with the approximate model only require simple algebraic computations.

This approach is used for comparison purposes in the example of a distillation column presented in §6.3.

6.3 Examples

6.3.1 Example 1 - Distillation column

This example considers the design of a controller for a simplified model of a distillation column (Benallou et al., 1986). The motivation for this example is to demonstrate how nonlinear model reduction techniques may be used to overcome the limitations of multi-parametric programming algorithms for systems with high dimensionality. The assumptions in this example do not intend to describe an industrial situation and, at the current state of the art, explicit multi-parametric controllers are not suitable for large scale applications such as industrial distillation columns (Pistikopoulos, 2009).

The system is schematically depicted in Figure 6.1 and the underlying equations presented in Appendix E.1. It may be noted that the system is mostly linear, with nonlinearities arising only from the equilibrium relations (E.6).
Figure 6.1: Schematic representation of the distillation column in example 6.3.1.

The control problem consists of regulating the product purity to a fixed set-point of \( x_1 = 0.935 \), using the reflux ratio (E.9) as the manipulated variable. The system states, \( x_i, i = 1, \ldots, 32 \), are assumed to be measured and no external disturbances are considered. A constraint is imposed on the manipulated variable, which is allowed to vary in the interval \( RR \in [0; 5] \).

Due to the high dimensionality of the model, the MP-NMPC algorithm presented in §6.1.2 cannot be directly applied and a model order reduction step should be included beforehand. For the purposes of this example, reduced order models with one and two states were derived, using the technique presented in §6.2.1.

The discrete-time representation of the reduced system of ODEs was obtained using an implicit Runge-Kutta method (Zavala et al., 2008). For the discretisation, three collocation points were used, and the number of finite elements was set to 9 and 6 for the reduced order controllers with one state and two states, respectively. The number of collocation points and finite elements may be determined performing offline simulations. Even though a larger number of finite elements would lead to a finer approximation, and have impact on the control performance, it is limited by the corresponding increase in computational burden.

The resulting control law consists of an expression for the manipulated variable, \( u \), as an explicit function of the reduced states of the system, \( \theta \). As depicted in Figure 6.2 and Figure 6.3, the control law is affine in each of the critical regions identified by
6.3. Examples

the \texttt{mp-nmpc} algorithm. Some examples of the optimal solutions and corresponding critical regions are presented in Appendix F.1.

The algorithm resulted in 11 and 49 critical regions, for the controllers with one state and two states, respectively. The map of critical regions for the reduced order controller with two states is depicted in Figure 6.4. It should be noted that the reduction scheme projects the system dynamics into a different space and therefore the state variables of the reduced order model, \( \theta \), do not hold the same physical meaning as in the original model.

Also presented in Figure 6.4 are the state trajectories for several disturbances with initial conditions lying in different critical regions of the state space. It may be observed that all trajectories converge to the desired set-point.

To assess the quality of the \texttt{mp-nmpc} algorithm approximation, the closed-loop performance of the reduced controller with two states was compared against a \texttt{nmpc} controller based on the same reduced model. Figure 6.5 shows how the two controllers perform in rejecting a disturbance of \(-5\%\). It may be observed that the explicit multi-parametric controller very closely approximates the performance of the \texttt{nmpc} controller based on the same reduced model. However, it should be noted that the computational time required to compute each control action is significantly lower for the explicit multi-parametric controller. While the \texttt{nmpc} based on the reduced model with two states took an average of 10.4s to compute each control action, the explicit multi-parametric controller based on the same model took less than 0.001s².

\footnote{Computational times refer to an Intel\textsuperscript{{\textregistered}} Core\textsuperscript{{\textsuperscript{T}}M} Quad Q9400 @ 2.66GHz processor, 4GB RAM.}

Figure 6.2: Control inputs as a function of the states for the controllers based on the reduced model with one state.
Figure 6.3: Control inputs as a function of the states for the controllers based on the reduced model with two states.

Figure 6.4: Critical regions and system trajectory for different disturbances. ■ - Initial point ◇ - Set-point.
6.3. Examples

For the disturbance of $-5\%$, the performance of the reduced order controllers with one state and two states was compared. The output trajectories, shown in Figure 6.6, indicate that the original system dynamics are mainly projected in the first state of the reduced model. Therefore, no significant improvement is obtained by increasing the order of the reduced model from one to 2, in accordance with results presented in (Hahn and Edgar, 2002).

The performance of the explicit multi-parametric controller based on the reduced model with one state was also assessed against (a) a NMPC controller based on the original full order model and (b) a controller designed using the meta-modelling based approach presented in §6.2.2. The NMPC online optimisation problem was solved at each time instant using the dynamic optimisation tools in the gPROMS package.

The results of the comparison, presented in Figure 6.7, show that the reduced order controller (nlsens) and the controller based on the meta-modelling approach closely (hdmr) approximate the performance of the full order controller.

A small offset ($\sim0.2\%$), observed to be a persistent deviation from the desired set-point, is present for the reduced order controller with one state. It will be part of future work to eliminate this offset by incorporating integral action in the optimisation formulation (6.3).

It should be noted that for the meta-modelling based approach, the problem to be solved is a linear multi-parametric program, which is significantly less computationally intensive than a nonlinear multi-parametric program. However, for this example, there is no perceptible advantage of using the nlsens algorithm. This observation may however be specific to this problem, since the nonlinearities in the original model are relatively
Figure 6.6: Closed-loop performance for the reduced order controllers with one state and two states.

Figure 6.7: Closed-loop controller performance for disturbance rejection.

mild, as explained above.

6.3.2 Example 1 - Train of cstr

This example concerns a nonlinear model of a train of two Continuous Stirred-Tank Reactors (cstr) where a generic irreversible reaction $A \rightarrow B$ takes place (Hahn and
6.3. Examples

Edgar, 2002). The system is schematically depicted in Figure 6.8 and the underlying equations presented in Appendix E.2. As opposed to the example in §6.3.1, the nonlinearities present in this model are more pronounced, especially due to the exponential terms in the energy balances (E.15) and (E.12).

The system comprises six states, corresponding to the temperature, volume, and concentration of $A$ in each reactor. The volume and temperature of the second reactor are observed variables and correspond to the outputs of the system.

The control problem consists of regulating the system outputs to a fixed set-point of $V_2 = 100$ and $C_2 = 463.13$. Two control inputs, $u$, are available, which allow manipulating the heat $Q$ supplied to the first reactor, according to (E.18), and the outlet flow $q_2$, according to (E.17). As in the example of §6.3.1, all states are assumed to be measured and no external disturbances are considered. A constraint is imposed on both manipulated variables, which are allowed to vary in the interval $u \in [0.5, 1.1]$.

Even though the model has a relatively small dimensionality, a reduced model with two states was derived in order to test the combined usage of nonlinear model order reduction and nonlinear MP-MPC in the presence of this type of nonlinearities. The model order reduction was achieved using the technique presented in §6.2.1. The controllability and observability gramian matrices, as well as the corresponding balanced form, are presented in Appendix F.2.

The discrete-time representation of the reduced system of ODES was obtained using an implicit Runge-Kutta method (Zavala et al., 2008). For the discretisation, three collocation points were used, and the number of finite elements was set to 10.

The resulting control law consists of an expression for the manipulated variables, $u$, as an explicit function of the reduced states of the system, $\theta$. Figure 6.9 presents the second component of the affine map of control actions in each of the critical regions identified by the MP-NMPC algorithm. Some examples of the optimal solutions and corresponding critical regions are presented in Appendix F.2.

The algorithm identified 26 critical regions, presented in Figure 6.10. It should be noted that the reduction scheme projects the system dynamics into a different space...
and therefore the state variables of the reduced order model, $\theta$, do not hold the same physical meaning as in the original model.

Also presented in Figure 6.10 are the state trajectories for several disturbances with initial conditions lying in different critical regions of the state space. It may be observed that all trajectories converge to the desired set-point.

To assess the quality of the MP-NMPC algorithm approximation, the closed-loop...
6.4. Concluding remarks

This chapter demonstrated the combined use of nonlinear model order reduction techniques and nonlinear multi-parametric control for the design and implementation of fast responding explicit multi-parametric controllers for nonlinear systems. It was shown that

The performance of the reduced order controller was compared against an online NMPC controller based on the same reduced model. Figure 6.11 and Figure 6.12 show how the two controllers perform in rejecting a disturbance of +5% from the steady-state of the system. It may be observed, in accordance with the results from the example in §6.3.1, that the explicit multi-parametric controller very closely approximates the performance of the NMPC controller based on the same reduced model. The time required to compute the control actions was decreased from an average of 8.3s, for the online NMPC controller, to less than 0.001s, for the explicit multi-parametric controller.3

The performance of the explicit multi-parametric controller based on the reduced order model was also assessed against an online NMPC controller based on the original full order model. The NMPC online optimisation problem was solved at each time instant using the dynamic optimisation tools in the gPROMS package.

The results of the comparison, presented in Figure 6.13 and Figure 6.14, show that the reduced order explicit controller closely approximates the performance of the full order controller. In contrast to the example in §6.3.1, no significant offset was detected for any of the system outputs.

Figure 6.11: Output trajectories of the volume in the second reactor for a disturbance of +5%
Figure 6.12: Output trajectories of the temperature in the second reactor for a disturbance of +5%.

Figure 6.13: Closed-loop controller performance for disturbance rejection.
the multi-parametric algorithm provides a very close approximation for the corresponding online control problem, while significantly reducing the required computational time. The explicit multi-parametric controller also showed a good closed-loop response, when compared to a full order online controller, based on the original model.
Chapter 7

Conclusions

The work presented in this thesis concerns different aspects of the theory of multi-parametric programming, dynamic programming, and explicit model predictive control. In this chapter, the proposed developments are summarised and the significance of the key contributions is highlighted. Based on the theory and results discussed throughout Chapters 2-6, future research directions and opportunities are also suggested.

7.1 Thesis summary

The fundamental theory of multi-parametric programming and its relation to model predictive control is presented in detail in Chapter 2. The presentation is initially based on a generic multi-parametric programming problem for which the optimality conditions are derived. It is shown that, under certain assumptions, the solution of a generic multi-parametric programming problem may be computed as a piece-wise affine function of the parameter vector in the neighbourhood of a local solution. For the particular case of multi-parametric linear and quadratic problems, it is shown that such neighbourhood may be determined as a convex set in the parameter space by enforcing feasibility and optimality conditions.

The relation between multi-parametric programming and explicit model predictive control is explained in §2.2. It is shown that a model predictive control problem with linear or quadratic objective function may be directly reformulated as a multi-parametric linear or quadratic problem, by taking the initial state of the system as the vector of parameters. A more efficient approach for explicit model predictive control is described in §2.3, based on multi-parametric programming and dynamic programming. The method described takes advantage of the sequential structure of model predictive control formulations to disassemble the problem into a set of recursive sub-problems that may be efficiently solved. The two methods of deriving explicit model predictive controllers are compared in §2.4 and the results obtained provide evidence of the computational benefits of using the approach based on multi-parametric programming and dynamic programming.
The computational benefits highlighted in Chapter 2 motivate the development and extension of the method based on multi-parametric programming and dynamic programming for the case of hybrid systems. This is the main subject of the developments proposed in Chapters 3-5.

In Chapter 3, the ideas of Chapter 2 are extended for the general problem of constrained dynamic programming of hybrid linear systems. The general class of constrained dynamic programming problems that involve mixed-integer linear formulations is shown to be equivalently described by a recursive problem, as a result of the principle of optimality. For such recursion, a reformulation based on multi-parametric mixed-integer linear programming is proposed, resulting in a convex problem that may be solved without the need for global optimisation methods that normally arises when solving constrained dynamic programming problems. The findings in this chapter are presented as an algorithm and illustrated for a constrained dynamic problem of a mixed-integer nature that arises in the context of inventory scheduling. This example highlights the computational benefits and added flexibility of the proposed algorithm as compared to conventional approaches.

The algorithm proposed in Chapter 3 is applicable to a wide-range of problems that have an inherent sequential structure. Given that model predictive control problems exhibit such special structure, the algorithm proposed in Chapter 3 is used as the basis for the developments presented in Chapter 4, concerning explicit model predictive control for hybrid linear systems. Chapter 4 begins with a mathematical presentation of the modelling of hybrid systems, with particular emphasis on the case of piece-wise affine systems. It is shown that by describing such systems as a set of logical propositions, it is possible to obtain an equivalent set of mixed-integer linear constraints that may be used in optimisation and control problems.

The formulation of a hybrid model predictive control problem with linear cost function for a piece-wise affine system is defined in §4.2. Using the principles described in §2.2, the state of the art approach is described, in which the hybrid model predictive control problem is reformulated as a mixed-integer linear programming problem, by taking the initial state of the system as the vector of parameters. A novel approach, based on the developments in Chapter 3, is presented in §4.3. The method is illustrated for a numerical example in which an explicit model predictive controller with linear cost function is designed for a piece-wise affine system. The chapter concludes with a discussion of the efficiency of the algorithm, where the limitations related to the complexity of the algorithm are identified and used to motivate future research directions.

The design of explicit model predictive controllers in Chapter 4 was based on the assumption that the model of the system was not affected by uncertainty. A more general approach is proposed in Chapter 5 where the problem of robust explicit model predictive control for hybrid systems is addressed. The method is based on multi-parametric programming and dynamic programming techniques for hybrid explicit model predictive control presented in Chapter 4.

Different approaches to modelling of uncertainties are presented in §5.1. The ap-
Main contributions

The algorithms and numerical studies presented in this thesis provide contributions that extend the state of the art in multi-parametric programming and explicit model predictive control. The key contributions are listed below.

- Numerical study of explicit model predictive control by constrained dynamic programming (Chapter 2).
  Despite the qualitative analysis on the complexity of the algorithm (Kouramas et al., 2011), the results currently available are vague and not compared to other methods. The discussion in this thesis shows by means of numerical examples that the algorithm is an improvement over the current state of the art, but also points out its shortcomings and the limit on the size of problem that may be expected to be solved.

- An algorithm for constrained dynamic programming of mixed-integer linear problems (Chapter 3).
  Constrained dynamic programming is an important and challenging problem that
has not benefited from significant developments in the literature. The algorithm presented in this thesis provides a valuable alternative that has the potential to be extended to the currently unsolved problem of constrained dynamic programming of mixed-integer quadratic problems.

- An algorithm for explicit model predictive control of hybrid systems by dynamic programming (Chapter 4).
  The developments in constrained dynamic programming presented in this thesis have important implications in the context of explicit model predictive control of hybrid systems. The proposed algorithm has the potential to be extended for the case of hybrid explicit model predictive control with quadratic cost function while avoiding the need for global optimisation that usually arises for these problems.

- An algorithm for robust explicit model predictive control of hybrid systems (Chapter 5).
  The literature on robust explicit model predictive control for hybrid systems is limited to very few publications, despite the importance and relevance of the problem. In this thesis, the problem of considering model uncertainty while designing explicit controllers for hybrid systems is handled in a systematic way, using multi-parametric programming and dynamic programming.

- A novel method for nonlinear explicit model predictive control (Chapter 6).
  Model reduction techniques that capture the most important dynamics of the model while reducing its dimensionality are a subject of high relevance that has been studied to a considerable extent. However, the studies in the literature do not demonstrate a complete methodology to design explicit controllers based on a high-fidelity model of the system. The method proposed in this thesis combines model approximation techniques and nonlinear multi-parametric programming to derive controllers that are successfully tested in closed-loop simulations against the original high-fidelity model.

### 7.3 Future research directions

The theory and results presented in Chapters 2-6 motivate several directions for future research, which are summarised below.

**Constrained dynamic programming of hybrid systems**

Despite the computational benefits of using the approach proposed for constrained dynamic programming of hybrid systems, the obtained results show that the complexity of the algorithm is exponential. In order to improve the performance, future work will be focused on exploring combinatorial methods for multi-parametric mixed-integer programming (Gupta et al., 2011) which are adequate for problems with small number of constraints and high parameter dimensionality, such as the sub-problems involved in the dynamic programming recursion.
An additional source of complexity is inherent to using exact dynamic programming methods for systems involving integer variables (Bertsimas and Weismantel, 2005). This was evident when the algorithm was applied to the problem of explicit model predictive control of a piece-wise affine system. In this context, future work will involve exploring approximate dynamic programming solutions based on heuristics and greedy algorithms to reduce the number of integer nodes required to solve the problem.

Reducing the complexity will facilitate the extension of the algorithm for constrained dynamic programming of hybrid systems involving quadratic cost functions, which is of significant practical importance.

**Explicit model predictive control for hybrid systems**

The methods currently available for explicit model predictive control of hybrid systems either rely on non-efficient multi-parametric mixed-integer formulations or dynamic programming formulations that are efficient but limited to control problems with linear cost function. Given that for numerical reasons quadratic cost functions are a far more popular choice in the literature, it is of key importance to derive efficient methods for explicit model predictive control of hybrid systems with quadratic cost function. The method presented in this thesis, based on dynamic programming, has the potential to be extended to this type of problem, but there are challenges related to the nature of multi-parametric mixed-integer quadratic programming problems.

At each stage of the dynamic programming recursion, the solution of the multi-parametric mixed-integer quadratic reformulation is given as the envelope of solutions corresponding to different integer nodes (Dua et al., 2002). As a consequence, the computational requirements involved in replacing the solutions of consecutive stages are significantly increased. Additionally, the final solution of the problem will have a high number of critical regions and may not be suitable for efficient online implementation. Developments in this area will either involve efficient point location algorithms for critical regions of high cardinality or approximate methods to reduce the number of critical regions obtained at each stage.

**Robust multi-parametric model predictive control**

The developments in robust explicit model predictive control presented in this thesis were derived under the assumption that the terms in the objective function were not affected by model uncertainty. Even though this assumption is frequently found in the literature, it would be desirable to take into account the effect of the uncertainty in the optimality of the solution. The main challenge in this area is that including the worst-case realisation of uncertainty in the objective function results in a min-max formulation for which nonlinear optimisation techniques are required (Wang and Rawlings, 2004).

Another important consideration related to uncertain systems is that the variability of the model affects the state estimation error and may lead to infeasible operation or
constraint violation. Future research in this topic will be focused on combining the method proposed in this thesis with moving horizon estimation (Voelker et al., 2013).

Nonlinear explicit model predictive control

As highlighted in the framework presented in §1.3, model reduction plays an important role in the design of explicit model predictive controllers for complex systems of high dimensionality. Given the results shown in this thesis, the combined use of model reduction and multi-parametric programming is a promising method for tackling the challenging problem of nonlinear explicit model predictive control. Future work will be focused on exploring different model approximation methods and discretisation schemes that minimise the complexity of the reduced order model. One area of particular interest is to derive bounds for the error resulting from the model reduction procedure. Having the knowledge of such bounds will enable the use of the explicit robust model predictive control methods presented in this thesis.

Extension to continuous-time systems

The algorithms for explicit model predictive control presented in this thesis have been developed for discrete-time systems. The reason for this is that considering the continuous differential equations that describe the system dynamics results in a problem of increased complexity. Additionally, the error resulting from the discretisation of a continuous system is usually acceptable for a wide range of applications. However, for applications where safety is critical, such as biomedical drug delivery applications (Parker et al., 1999; Dua et al., 2004), the error resulting from the discretisation may not be acceptable. In these cases, it is necessary to directly solve continuous-time multi-parametric programming problems for which the current state of the art is limited (Sakizlis et al., 2005). Another motivation for directly tackling the continuous-time problem is that the discretisation procedure usually increases the complexity of the model and might reduce the applicability of the algorithm to problems with smaller dimensions. To achieve this goal, further developments in the theory of dynamic optimisation under uncertainty are necessary.

Applications

The use of multi-parametric programming to address explicit model predictive control is a relatively recent theory; it is understandable that there is a significant drive to push the boundaries of the state of the art as close as possible to the well established field of model predictive control. However, the fast theoretical advances observed in the last decade have not been followed by an equivalent number of technology implementations based on multi-parametric programming and explicit model predictive control. Therefore, there is an increased need to showcase applications that make use of recent developments such as robust explicit model predictive control or explicit nonlinear model predictive control.
One application of particular interest is the field of biomedical drug delivery systems. The recent trend in this field has been towards deriving complex high-fidelity models that accurately capture the dynamics of the human body (Krieger et al., 2013). In addition to the complexity involved, these models are subject to significant variability and require control strategies that take into account the uncertainty to guarantee safety of the patient. In this context, explicit model predictive control provides the advantage of allowing exhaustive computer-based tests prior to the implementation in the real system. For certain applications, such as type 1 diabetes (Parker et al., 1999), portability of the drug delivery device would provide significant benefits; using explicit model predictive control, the device has the enhanced predictive capabilities of model predictive but requires significantly less hardware for implementation.

7.4 Publications from this thesis

The work from this thesis that has been presented in international conferences or published as journal articles is listed below.

Journal articles


- Pedro Rivotti and Efstratios N Pistikopoulos. Explicit model predictive control for hybrid systems by multi-parametric programming and dynamic programming. *Article to be submitted*.

Conference proceedings


- Pedro Rivotti, Martina Wittmann-Hohlbein, and Efstratios N Pistikopoulos. A combined multi-parametric and dynamic programming approach for model predictive control.
control of hybrid linear systems. In 22nd European Symposium on Computer Aided Process Engineering, 2012d

Oral presentations in conferences


Poster presentations in conferences


References


References


References


ParOS. Parametric optimization programming (pop), 2004. URL www.parostech.co.uk.


References


Pedro Rivotti and Efstratios N Pistikopoulos. Explicit model predictive control for hybrid systems by multi-parametric programming and dynamic programming. *Article to be submitted*.


References


References


Appendix A

Simple multi-parametric programming example from §1.1

The following multi-parametric linear programming problem, used to illustrate the concepts presented in §1.1, has been adapted from (Gal and Davis, 1979).

\[
\begin{align*}
  z(\theta_1, \theta_2) &= \max_{x_1, x_2} 3x_1 + 8x_2 \\
  \text{s.t.} & \quad x_1 + x_2 \leq -0.5\theta_1 + 2\theta_2 + 13 \\
                  & \quad 5x_1 - 4x_2 \leq 1.2\theta_2 + 20 \\
                  & \quad -8x_1 + 22x_2 \leq 8\theta_1 - 2.5\theta_1 + 121 \\
                  & \quad 4x_1 + x_2 \geq 0.6\theta_1 + 8 \\
                  & \quad 92x_1 + x_2 \leq 2\theta_1 + 2\theta_2 + 34 \\
                  & \quad x_1 \geq 0, \quad x_2 \geq 0 \\
\end{align*}
\]

(A.1)

In problem (A.1), \(x_1, x_2 \in \mathbb{R}\) are the decision variables, \(z(\theta_1, \theta_2) \in \mathbb{R}\) is the cost function, and \(\theta_1, \theta_2 \in \mathbb{R}\) are the parameters, bounded in the initial region \(\text{CR}_{\text{init}}\).

Problem (A.1) may be solved using any of the multi-parametric programming Matlab toolboxes available (ParOS, 2004; Kvasnica et al., 2004). The optimal solution and corresponding critical regions are given in Table A.1. The map of critical regions in the parameter space is depicted in Figure 1.1b.
Table A.1: Optimal solution and critical regions of multi-parametric linear problem (A.1).

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
</table>
| CR1    | \[0.597\theta_1 - 0.802\theta_2 \leq 1.466\]
|        | \[-0.496\theta_1 - 0.868\theta_2 \leq -3.898\]
|        | \[\theta_1 \leq 10\]  |
|        | \[\theta_2 \leq 10\]  |
|        | \[x_1 = 0.109\theta_1 + 0.190\theta_2 - 0.853\]  |
|        | \[x_2 = 0.403\theta_1 - 0.045\theta_2 + 5.190\]  |
| CR2    | \[0.483\theta_1 - 0.876\theta_2 \leq 0.786\]
|        | \[-0.597\theta_1 - 0.802\theta_2 \leq -1.466\]
|        | \[\theta_1 \leq 10\]  |
|        | \[x_1 = -0.711\theta_1 + 1.290\theta_2 + 1.158\]  |
|        | \[x_2 = 0.105\theta_1 + 0.355\theta_2 + 5.921\]  |
| CR3    | \[0.496\theta_1 + 0.868\theta_2 \leq 3.898\]
|        | \[-0.707\theta_1 - 0.707\theta_2 \leq 0.353\]
|        | \[\theta_1 \leq 4.8\]  |
|        | \[\theta_2 \leq 4.8\]  |
|        | \[x_1 = 0\]  |
|        | \[x_2 = \theta_1 + \theta_2 + 0.500\]  |
| CR4    | \[-0.483\theta_1 + 0.876\theta_2 \leq -0.786\]
|        | \[0.243\theta_1 - 0.970\theta_2 \leq -6.306\]
|        | \[4.8 \leq \theta_1 \leq 10\]  |
|        | \[x_1 = 0\]  |
|        | \[x_2 = -0.250\theta_1 + \theta_2 + 6.500\]  |
Further results for example of §2.4

Table B.1 and Table B.2 present the expressions of the critical regions, and corresponding optimal solutions, for problem (2.29) with $N = 2$ and $N = 25$, respectively.

**Table B.1**: Sample of critical regions and corresponding optimal solutions for (2.29), $N = 2$.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR₁</td>
<td>$\theta_2 \leq 0.4$ $-\theta_2 \leq 0.55$ $0.683\theta_1 + \theta_2 \leq -0.757$ $-\theta_1 \leq 2$ $u_0 = 1$</td>
</tr>
<tr>
<td>CR₄</td>
<td>$-0.004\theta_1 + \theta_2 \leq 0.501$ $-0.004\theta_1 - \theta_2 \leq -0.399$ $0.108\theta_1 - \theta_2 \leq -0.577$ $-\theta_1 \leq 2$ $u_0 = -0.037\theta_1 - 10.001\theta_2 \leq 4.987$</td>
</tr>
<tr>
<td>CR₅</td>
<td>$0.052\theta_1 + \theta_2 \leq 0.537$ $-0.108\theta_1 + \theta_2 \leq 0.577$ $0.052\theta_1 - \theta_2 \leq 0.537$ $0.108\theta_1 - \theta_2 \leq 0.577$ $-0.707\theta_1 - \theta_2 \leq 0.732$ $0.707\theta_1 + \theta_2 \leq 0.732$ $u_0 = -0.965\theta_1 - 1.366\theta_2$</td>
</tr>
<tr>
<td>CR₉</td>
<td>$0.004\theta_1 - \theta_2 \leq 0.501$ $0.004\theta_1 + \theta_2 \leq -0.399$ $-0.108\theta_1 + \theta_2 \leq -0.578$ $-\theta_1 \leq 2$ $u_0 = -0.037\theta_1 - 10.001\theta_2 - 4.987$</td>
</tr>
</tbody>
</table>
Table B.2: Sample of critical regions and corresponding optimal solutions for (2.29), $N = 25$.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR$_7$</td>
<td>$0.129\theta_1 - \theta_2 \leq -0.160$ $0.135\theta_1 + \theta_2 \leq -0.367$ $\theta_1 \geq -2$ $u_0 = 1$</td>
</tr>
<tr>
<td>CR$_{212}$</td>
<td>$-\theta_1 + 0.963\theta_2 \leq 1.251$ $-\theta_1 - 0.452\theta_2 \leq 1.109$ $-0.611\theta_1 - \theta_2 \leq 0.627$ $\theta_1 - 0.845\theta_2 \leq -1.200$ $\theta_1 + 0.452\theta_2 \leq -1.080$ $u_0 = -0.868\theta_1 - 1.420\theta_2 + 0.112$</td>
</tr>
<tr>
<td>CR$_{411}$</td>
<td>$\theta_1 - 0.845\theta_2 \leq 1.200$ $0.640\theta_1 + \theta_2 \leq 0.658$ $-\theta_1 + 0.742\theta_2 \leq -1.157$ $-\theta_1 - 0.478\theta_2 \leq -1.078$ $u_0 = -0.894\theta_1 - 1.398\theta_2 - 0.080$</td>
</tr>
<tr>
<td>CR$_{582}$</td>
<td>$0.013\theta - \theta_2 \leq 0.506$ $\theta_1 + 0.056\theta_2 \leq 0.908$ $-0.026\theta_1 + \theta_2 \leq -0.513$ $-\theta_1 - 0.056\theta_2 \leq -0.883$ $u_0 = -0.090\theta_1 - 6.671\theta_2 - 3.293$</td>
</tr>
<tr>
<td>CR$_{750}$</td>
<td>$0.013\theta - \theta_2 \leq 0.508$ $\theta_1 + 0.056\theta_2 \leq 1.008$ $-0.029\theta_1 + \theta_2 \leq -0.516$ $-\theta_1 - 0.056\theta_2 \leq -0.983$ $u_0 = -0.100\theta_1 - 6.671\theta_2 - 3.283$</td>
</tr>
<tr>
<td>CR$_{932}$</td>
<td>$\theta_1 \leq 1.132$ $-\theta_2 \leq 0.550$ $-0.005\theta_1 + \theta_2 \leq -0.503$ $-\theta_1 \leq -1.107$ $u_0 = -20\theta_2 - 10$</td>
</tr>
<tr>
<td>CR$_{1020}$</td>
<td>$0.006\theta_1 - \theta_1 \leq 0.504$ $\theta_1 + 0.025\theta_2 \leq 1.245$ $-0.018\theta_1 + \theta_2 \leq -0.512$ $-\theta_1 + 0.025\theta_2 \leq -1.220$ $u_0 = -0.062\theta_1 - 10.002\theta_2 - 4.962$</td>
</tr>
</tbody>
</table>
Appendix C

Further results for example of §3.3

Table C.1 shows examples of the critical regions, and corresponding optimal solutions, of (3.18), obtained in iteration $i = 5$ ($N = 6$) after replacing the solution of iteration $i = 6$.

Some example of the final solution of the example in §3.3, for $N = 6$, are presented in Table C.2.

Table C.1: Critical regions and corresponding optimal solutions for example of §3.3, $i = 5, N = 6$.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
</table>
| CR$_1$ | $15.5 \leq x_4 \leq 18$  
          $6 \leq r_5 \leq 10$  
          $15 \leq r_6 \leq 16$  
          $x_5 = 16$  
          $x_6 = 16$ |
| CR$_5$ | $11.5 \leq x_4 \leq 12.5$  
          $6 \leq r_5 \leq 10$  
          $15 \leq r_6 \leq 16$  
          $x_5 = 12$  
          $x_6 = 16$ |
| CR$_{22}$ | $10 \leq x_4 \leq 10.5$  
             $6 \leq r_5 \leq 10$  
             $14 \leq r_6 \leq 15$  
             $x_5 = 10$  
             $x_6 = 15$ |
| CR$_{30}$ | $7 \leq x_4 \leq 8$  
              $14 \leq r_5 \leq 15$  
              $x_5 = 8$  
              $x_6 = 15$ |
| CR$_{30}$ | $0 \leq x_4 \leq 7.5$  
              $6 \leq r_5 \leq 7$  
              $13 \leq r_6 \leq 14$  
              $x_5 = 7$  
              $x_6 = 14$ |
Table C.2: Critical regions and corresponding optimal solutions for final solution of example of §3.3, $N = 6$.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR₁</td>
<td></td>
</tr>
<tr>
<td>$14.5 \leq x₀ \leq 18$</td>
<td>$x₁ = 15$</td>
</tr>
<tr>
<td>$9 \leq r₁ \leq 13$</td>
<td>$x₂ = 15$</td>
</tr>
<tr>
<td>$14 \leq r₂ \leq 15$</td>
<td>$x₃ = 10$</td>
</tr>
<tr>
<td>$8 \leq r₃ \leq 9$</td>
<td>$x₄ = 10$</td>
</tr>
<tr>
<td>$0 \leq r₄ \leq 3$</td>
<td>$x₅ = 10$</td>
</tr>
<tr>
<td>$9 \leq r₅ \leq 10$</td>
<td>$x₆ = 16$</td>
</tr>
<tr>
<td>$15 \leq r₆ \leq 16$</td>
<td></td>
</tr>
<tr>
<td>CR₁₂₄</td>
<td></td>
</tr>
<tr>
<td>$13 \leq x₀ \leq 13.5$</td>
<td>$x₁ = 13$</td>
</tr>
<tr>
<td>$9 \leq r₁ \leq 13$</td>
<td>$x₂ = 15$</td>
</tr>
<tr>
<td>$14 \leq r₂ \leq 15$</td>
<td>$x₃ = 8$</td>
</tr>
<tr>
<td>$5 \leq r₃ \leq 8$</td>
<td>$x₄ = 8$</td>
</tr>
<tr>
<td>$0 \leq r₄ \leq 3$</td>
<td>$x₅ = 9$</td>
</tr>
<tr>
<td>$8 \leq r₅ \leq 9$</td>
<td>$x₆ = 16$</td>
</tr>
<tr>
<td>$15 \leq r₆ \leq 16$</td>
<td></td>
</tr>
<tr>
<td>CR₄₀₃</td>
<td></td>
</tr>
<tr>
<td>$0 \leq x₀ \leq 13$</td>
<td>$x₁ = 11$</td>
</tr>
<tr>
<td>$10 \leq r₁ \leq 11$</td>
<td>$x₂ = 12$</td>
</tr>
<tr>
<td>$11 \leq r₂ \leq 12$</td>
<td>$x₃ = 6$</td>
</tr>
<tr>
<td>$5 \leq r₃ \leq 6$</td>
<td>$x₄ = 6$</td>
</tr>
<tr>
<td>$0 \leq r₄ \leq 3$</td>
<td>$x₅ = 7$</td>
</tr>
<tr>
<td>$6 \leq r₅ \leq 7$</td>
<td>$x₆ = 16$</td>
</tr>
<tr>
<td>$15 \leq r₆ \leq 16$</td>
<td></td>
</tr>
<tr>
<td>CR₇₀₀</td>
<td></td>
</tr>
<tr>
<td>$13 \leq x₀ \leq 13.5$</td>
<td>$x₁ = 13$</td>
</tr>
<tr>
<td>$9 \leq r₁ \leq 13$</td>
<td>$x₂ = 15$</td>
</tr>
<tr>
<td>$14 \leq r₂ \leq 15$</td>
<td>$x₃ = 8$</td>
</tr>
<tr>
<td>$7 \leq r₃ \leq 8$</td>
<td>$x₄ = 7$</td>
</tr>
<tr>
<td>$0 \leq r₄ \leq 3$</td>
<td>$x₅ = 7$</td>
</tr>
<tr>
<td>$6 \leq r₅ \leq 7$</td>
<td>$x₆ = 15$</td>
</tr>
<tr>
<td>$14 \leq r₆ \leq 15$</td>
<td></td>
</tr>
<tr>
<td>CR₁₅₀₅</td>
<td></td>
</tr>
<tr>
<td>$12 \leq x₀ \leq 18$</td>
<td>$x₁ = 12$</td>
</tr>
<tr>
<td>$9 \leq r₁ \leq 11$</td>
<td>$x₂ = 12$</td>
</tr>
<tr>
<td>$11 \leq r₂ \leq 12$</td>
<td>$x₃ = 6$</td>
</tr>
<tr>
<td>$5 \leq r₃ \leq 6$</td>
<td>$x₄ = 6$</td>
</tr>
<tr>
<td>$0 \leq r₄ \leq 3$</td>
<td>$x₅ = 9$</td>
</tr>
<tr>
<td>$8 \leq r₅ \leq 9$</td>
<td>$x₆ = 13$</td>
</tr>
<tr>
<td>$12 \leq r₆ \leq 13$</td>
<td></td>
</tr>
</tbody>
</table>
Appendix D

Further results for example of §4.4

Table D.1 and Table D.2 present the expressions of the critical regions, and corresponding optimal solutions, for problem (4.20) with $N = 2$ and $N = 5$, respectively.

Table D.1: Critical regions and corresponding optimal solutions for the solution of (4.20), $N = 2.$

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR$_1$</td>
<td>$u_0 = 0.246\theta_1 - 0.373\theta_2$</td>
</tr>
<tr>
<td>CR$_4$</td>
<td>$u_0 = 0.693\theta_1 - 0.4\theta_2$</td>
</tr>
<tr>
<td>CR$_9$</td>
<td>$u_0 = 1$</td>
</tr>
<tr>
<td>CR$_{14}$</td>
<td>$u_0 = -0.446\theta_1 - 0.027\theta_2$</td>
</tr>
<tr>
<td>CR$_{19}$</td>
<td>$u_0 = -0.246\theta_1 - 0.373\theta_2$</td>
</tr>
</tbody>
</table>
### Table D.2: Critical regions and corresponding optimal solutions for the solution of (4.20), $N = 5.$

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{CR}_{10}$</td>
<td>$0.5\theta_1 + 0.866\theta_2 \leq -3.383$  $u_0 = 0.462\theta_1 - 0.8\theta_2 - 1.563$</td>
</tr>
<tr>
<td></td>
<td>$0.5\theta_1 + 0.866\theta_2 \leq 0$</td>
</tr>
<tr>
<td></td>
<td>$-\theta_2 \leq 0$</td>
</tr>
<tr>
<td></td>
<td>$-0.5\theta_1 + 0.866\theta_2 \leq 1.083$</td>
</tr>
<tr>
<td></td>
<td>$0.5\theta_1 - 0.867\theta_2 \leq 1.353$</td>
</tr>
<tr>
<td>$\text{CR}_{23}$</td>
<td>$\theta_2 \leq 0$  $u_0 = -0.462\theta_1 - 0.8\theta_2$</td>
</tr>
<tr>
<td></td>
<td>$-0.5\theta_1 - 0.866\theta_2 \leq 0$</td>
</tr>
<tr>
<td></td>
<td>$0.5\theta_1 + 0.866\theta_2 \leq 1.083$</td>
</tr>
<tr>
<td>$\text{CR}_{32}$</td>
<td>$-0.5\theta_1 + 0.866\theta_2 \leq -3.045$</td>
</tr>
<tr>
<td>$\text{CR}_{47}$</td>
<td>$0.5\theta_1 - 0.866\theta_2 \leq 3.383$  $u_0 = -0.462\theta_1 - 0.8\theta_2 - 1.563$</td>
</tr>
<tr>
<td></td>
<td>$0.5\theta_1 + 0.866\theta_2 \leq -1.691$</td>
</tr>
<tr>
<td></td>
<td>$-0.5\theta_1 - 0.866\theta_2 \leq 2.774$</td>
</tr>
<tr>
<td></td>
<td>$-0.5\theta_1 + 0.866\theta_2 \leq -3.383$</td>
</tr>
<tr>
<td>$\text{CR}_{51}$</td>
<td>$0.866\theta_1 + 0.5\theta_2 \leq 0$  $u_0 = -0.693\theta_1 - 0.4\theta_2$</td>
</tr>
<tr>
<td></td>
<td>$-0.866\theta_1 - 0.5\theta_2 \leq 1.25$</td>
</tr>
<tr>
<td></td>
<td>$-\theta_2 \leq 10$</td>
</tr>
<tr>
<td>$\text{CR}_{55}$</td>
<td>$0.5\theta_1 - 0.866\theta_2 \leq -1.353$  $u_0 = -0.462\theta_1 - 0.8\theta_2 + 1.25$</td>
</tr>
<tr>
<td></td>
<td>$-\theta_1 \leq 0$</td>
</tr>
<tr>
<td></td>
<td>$0.5\theta_1 + 0.866\theta_2 \leq 2.436$</td>
</tr>
</tbody>
</table>
Appendix E

Equations and parameters for examples of §6.3

E.1 Distillation column with 32 states

The following equations and parameter refer to the distillation column example presented in §6.3.1 and schematically depicted in Figure 6.1.

It is assumed that in each tray, \( i \), equilibrium is established between the vapour composition, \( y_i \), and the liquid composition, \( x_i \). The condenser is considered to be a total condenser, i.e., \( x_1 = y_1 \).

Dynamic equations:

Condenser:

\[
\frac{dx_1}{dt} = \frac{1}{A_{\text{cond}}} V (y_2 - x_1) \quad (E.1)
\]

Trays 2 to 16:

\[
\frac{dx_i}{dt} = \frac{1}{A_{\text{tray}}} [L_1 (x_{i-1} - x_i) - V (y_i - y_{i+1})] \quad (E.2)
\]

Feed tray:

\[
\frac{dx_{17}}{dt} = \frac{1}{A_{\text{tray}}} [Fx_F + L_1 x_{16} - L_2 x_{17} - V (y_{17} - y_{18})] \quad (E.3)
\]

Trays 17 to 31:

\[
\frac{dx_i}{dt} = \frac{1}{A_{\text{tray}}} [L_2 (x_{i-1} - x_i) - V (y_i - y_{i+1})] \quad (E.4)
\]

Reboiler:

\[
\frac{dx_{32}}{dt} = \frac{1}{A_{\text{reb}}} [L_2 x_{31} - (F - D) x_{32} - Vy_{32}] \quad (E.5)
\]
Appendix E. Equations and parameters for examples of §6.3

**Equilibrium relation:**

\[
\alpha = \frac{y_i(1 - x_i)}{x_i(1 - y_i)} \quad (E.6)
\]

**Other equations:**

\[
V = L_1 + D \quad (E.7)
\]

\[
L_2 = F + L_1 \quad (E.8)
\]

\[
RR = \frac{L_1}{D} \quad (E.9)
\]

**Parameters:**

\[
F = 0.4
\]

\[
D = 0.2
\]

\[
A_{\text{cond}} = 0.5
\]

\[
A_{\text{tray}} = 0.25
\]

\[
A_{\text{reb}} = 1.0
\]

\[
\alpha = 1.6
\]

**E.2 cstr train with 6 states**

The following equations and parameter refer to the cstr train example presented in §6.3.2 and schematically depicted in Figure 6.8.

**Dynamic equations**

**Tank 1**

\[
\frac{dV_1}{dt} = q_f - q_1 \quad (E.10)
\]

\[
\frac{dCa_1}{dt} = q_f \frac{Ca_1}{V_1} - k_0 Ca_1 \exp\left(-\frac{E_A}{RT_1}\right) - q_1 \frac{Ca_1}{V_1} - \frac{Ca_1 dV_1}{d t} \quad (E.11)
\]

\[
\frac{dT_1}{dt} = q_f \frac{T_f}{V_1} + \frac{dHk_0}{pCp} Ca_1 \exp\left(-\frac{E_A}{RT_1}\right) - q_1 \frac{T_1}{V_1} + Q - \frac{T_1 dV_1}{d t} \quad (E.12)
\]

**Tank 2**

\[
\frac{dV_2}{dt} = q_1 - q_2 \quad (E.13)
\]

\[
\frac{dCa_2}{dt} = q_f \frac{Ca_1}{V_2} - k_0 Ca_2 \exp\left(-\frac{E_A}{RT_2}\right) - q_2 \frac{Ca_2}{V_2} - \frac{Ca_2 dV_2}{d t} \quad (E.14)
\]

\[
\frac{dT_2}{dt} = q_f \frac{T_1}{V_2} + \frac{dHk_0}{pCp} Ca_2 \exp\left(-\frac{E_A}{RT_2}\right) - q_1 \frac{T_2}{V_2} + Q - \frac{T_2 dV_2}{d t} \quad (E.15)
\]
Other equations

\[ q_1 = c_1 \sqrt{V_1 - V_2} \]  \hspace{1cm} (E.16)
\[ q_2 = c_1 \sqrt{V_2 u_1} \]  \hspace{1cm} (E.17)
\[ Q = -c_2 u_2 \]  \hspace{1cm} (E.18)

Parameters

\[ c_1 = 10 \]
\[ c_2 = 48.1909 \]
\[ qf = 100 \]
\[ Caf = 1 \]
\[ Tf = 350.0 \]
\[ k_0 = 7.2 \times 10^4 \]
\[ \frac{E_A}{R} = 1 \times 10^4 \]
\[ \rho = 1000 \]
\[ Cp = 0.239 \]
\[ dH = 4.78 \times 10^4 \]
Appendix F

Further results for examples of §6.3

F.1 Distillation column with 32 states

Tables F.1 and F.2 present a sample of the critical regions and corresponding optimal solutions for the reduced order explicit controllers with 1 state and 2 states, respectively.

Table F.1: Critical regions for the reduced controllers with 1 state and corresponding optimal solutions.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR₁</td>
<td>$-5.593 \leq \theta \leq -5.286$</td>
</tr>
<tr>
<td>CR₃</td>
<td>$-5.960 \leq \theta \leq -5.808$</td>
</tr>
<tr>
<td>CR₅</td>
<td>$-5.045 \leq \theta \leq -5.048$</td>
</tr>
</tbody>
</table>

F.2 cstr train with 6 states

The controllability and observability gramians, $W_c$ and $W_o$, and the corresponding balanced form, $\Sigma$, are given by the matrices (F.1), (F.2) and (F.3), respectively.

$$W_c = \begin{bmatrix} 0.0820 & -0.1927 & 0.0049 & 0.1724 & -0.4344 & 0.0014 \\ -0.1927 & 18.3590 & -0.8540 & -1.4913 & 43.8790 & -1.0874 \\ 0.0049 & -0.8540 & 0.0400 & 0.0637 & -2.0726 & 0.0529 \\ 0.1724 & -1.4913 & 0.0637 & 0.6358 & -7.1474 & 0.2254 \\ -0.4344 & 43.8790 & -2.0726 & -7.1474 & 179.4900 & -5.4751 \\ 0.0014 & -1.0874 & 0.0529 & 0.2254 & -5.4751 & 0.1908 \end{bmatrix}$$

(F.1)
Table F.2: Example of critical regions for the reduced controllers with 2 state and corresponding optimal solutions.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR&lt;sub&gt;1&lt;/sub&gt;</td>
<td>( u(\theta) = 5.0399\theta_1 + 0.3108\theta_2 + )</td>
</tr>
<tr>
<td>(-\theta_1 - 0.0617\theta_2 \leq 5.3238)</td>
<td>(\theta_1 + 0.0340\theta_2 \leq -4.8435)</td>
</tr>
<tr>
<td>(4.4 \leq \theta_2 \leq 6.6)</td>
<td>(26.831)</td>
</tr>
<tr>
<td>CR&lt;sub&gt;9&lt;/sub&gt;</td>
<td>( u(\theta) = 5 )</td>
</tr>
<tr>
<td>(-\theta_1 + 0.052\theta_2 \leq -4.0766)</td>
<td>(-\theta_1 + 0.0785\theta_2 \leq 4.9492)</td>
</tr>
<tr>
<td>(-\theta_1 - 0.194\theta_2 \leq 4.3365)</td>
<td>(\theta_1 + 0.0376\theta_2 \leq -4.1730)</td>
</tr>
<tr>
<td>(4.4 \leq \theta_2 \leq 6.6)</td>
<td>(14.7306)</td>
</tr>
<tr>
<td>CR&lt;sub&gt;12&lt;/sub&gt;</td>
<td>( u(\theta) = 2.1612\theta_1 - 0.0154\theta_2 + )</td>
</tr>
<tr>
<td>(-\theta_1 + 0.1156\theta_2 \leq 5.2001)</td>
<td>(-\theta_1 - 0.0071\theta_2 \leq -4.5023)</td>
</tr>
<tr>
<td>(\theta_1 - 0.6346\theta_2 \leq 0.4671)</td>
<td>(\theta_1 - 0.6141\theta_2 \leq -8.4067)</td>
</tr>
<tr>
<td>(\theta_2 \leq 6.6)</td>
<td>(14.4268)</td>
</tr>
<tr>
<td>CR&lt;sub&gt;20&lt;/sub&gt;</td>
<td>( u(\theta) = 2.1228\theta_1 + 0.0051\theta_2 + )</td>
</tr>
<tr>
<td>(-\theta_1 + 0.0024\theta_2 \leq -4.4408)</td>
<td>(-\theta_1 + 0.0071\theta_2 \leq 8.3198)</td>
</tr>
<tr>
<td>(\theta_1 - 0.6356\theta_2 \leq 0.4610)</td>
<td>(\theta_2 \leq 6.6)</td>
</tr>
<tr>
<td>CR&lt;sub&gt;36&lt;/sub&gt;</td>
<td>( u(\theta) = 2.1745\theta_1 - 0.0150\theta_2 + )</td>
</tr>
<tr>
<td>(-\theta_1 - 0.0962\theta_2 \leq 4.3982)</td>
<td>(-\theta_1 + 0.0194\theta_2 \leq -4.3375)</td>
</tr>
<tr>
<td>(\theta_1 + 0.6007\theta_2 \leq 8.3198)</td>
<td>(\theta_1 - 0.6356\theta_2 \leq 0.4610)</td>
</tr>
<tr>
<td>CR&lt;sub&gt;49&lt;/sub&gt;</td>
<td>( u(\theta) = 4.7692\theta_1 + 0.2830\theta_2 + )</td>
</tr>
<tr>
<td>(-\theta_1 + 0.1071\theta_2 \leq -4.3500)</td>
<td>(\theta_2 \geq 4.4)</td>
</tr>
<tr>
<td>(\theta_1 + 0.1071\theta_2 \leq -4.3500)</td>
<td>(\theta_2 \geq 4.4)</td>
</tr>
<tr>
<td>(\theta_1 + 0.1071\theta_2 \leq -4.3500)</td>
<td>(\theta_2 \geq 4.4)</td>
</tr>
</tbody>
</table>

\[
W_0 = \begin{bmatrix}
159900.0 & -665.5 & -65597.0 & 32341.0 & -3.6 & 801.2 \\
-665.5 & 10.4 & 1198.7 & -501.1 & 0.4 & 451.7 \\
-65597.0 & 1198.7 & 143790.0 & -60516.0 & 18.2 & 57433.0 \\
32341.0 & -501.1 & -60516.0 & 43051.0 & -8.2 & -28834.0 \\
-3.6 & 0.4 & 18.2 & -8.2 & 0.2 & 9.5 \\
801.2 & 451.7 & 57433.0 & -28834.0 & 9.5 & 248620.0 \\
\end{bmatrix}
\]  

\[
\Sigma = \begin{bmatrix}
222.450 & 0 & 0 & 0 & 0 & 0 \\
0 & 193.890 & 0 & 0 & 0 & 0 \\
0 & 0 & 36.114 & 0 & 0 & 0 \\
0 & 0 & 0 & 9.625 & 0 & 0 \\
0 & 0 & 0 & 0 & 1.541 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.037 \\
\end{bmatrix}
\]

Table F.3 presents a sample of the critical regions and corresponding optimal solutions for the reduced order explicit controller with 2 states.
Table F.3: Example of critical regions for the reduced controllers with 2 state and corresponding optimal solutions.

<table>
<thead>
<tr>
<th>Region</th>
<th>Optimal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR$_1$</td>
<td>( u(\theta) = -0.0489\theta_1 + 0.0166\theta_2 - 1.0015 )</td>
</tr>
<tr>
<td>CR$_2$</td>
<td>( u(\theta) = 1.1 )</td>
</tr>
<tr>
<td>CR$_6$</td>
<td>( u(\theta) = -0.0774\theta_1 - 0.0447\theta_2 - 0.9936 )</td>
</tr>
<tr>
<td>CR$_{16}$</td>
<td>( u(\theta) = -0.0507\theta_1 + 0.0116\theta_2 - 0.9900 )</td>
</tr>
<tr>
<td>CR$_{18}$</td>
<td>( u(\theta) = -0.0650\theta_1 - 0.0181\theta_2 - 1.0052 )</td>
</tr>
<tr>
<td>CR$_{23}$</td>
<td>( u(\theta) = -0.0710\theta_1 - 0.0302\theta_2 - 1.2055 )</td>
</tr>
</tbody>
</table>
Appendix G

Meta-modelling based N-step ahead prediction

Consider a multiple input single output (MISO) continuous dynamical system of the form:

\[
\begin{align*}
\dot{x} &= f(x(t), u(t)) \\
y(t) &= h(x(t), u(t))
\end{align*}
\]

where \(x\) represents the vector of states \(x \in \mathbb{R}^n\), \(y \in \mathbb{R}\) that of the output of the model and \(u \in \mathbb{R}^m\) of control inputs. \(f\) and \(h\) are \(C^2\) vector fields on the space of states and controls. \(f: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n\) with an equilibrium \(f(0,0) = 0\) and \(h: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}\). The aim is the ability to formulate a convex control problem that enables the use of state of the practice linear online and explicit MPC. The reference tracking problem, where the output \(y\) is to be driven to the set-point \(y_{sp}\), over a time horizon \(N\), is formulated as follows.

In a nonlinear MPC implementation, an open loop optimal control problem is formulated considering the state of the system at time \(t_k\) as the initial state:

\[
\begin{align*}
\min_{u} \int_{0}^{T} \Phi(y(t), u(t))dt + \Xi(y(T)) \\
\text{s.t.} \quad x(0) &= x(t_k) \\
\dot{x} &= f(x(t), u(t)) \\
y(t) &= h(x(t), u(t)) \\
u_{\text{min}} \leq u(t) \leq u_{\text{max}} \\
y_{\text{min}} \leq y(t) \leq y_{\text{max}}
\end{align*}
\]

Where \(\Xi: \mathbb{R}^n \mathbb{R}^m \to \mathbb{R}\) is a stage cost, \(\Xi: \mathbb{R}^m \to \mathbb{R}\) is a terminal cost function and \(T\) is the predicted horizon length. The formulation in equation (G.2) is an infinite dimensional control problem which in practice cannot be solved directly. Instead, the following finite dimensional discrete-time approximation is solved (denoting \(N\) as the prediction time horizon for the control problem):
The idea is now to replace the iterative functions \( \tilde{f} \) and \( \tilde{h} \) by a set of \( N \) static mappings \( \{\Psi\}_{j \in [1:N]} \), i.e., algebraic expressions of the form:

\[
\forall j \in [1, N], y_{t+j} = \Psi_j(x_t, u_1, u_2, \ldots, u_j)
\]  

(G.4)

where \( x_t \) is the initial condition as in (G.2). The nonlinear and continuous dynamical system in (G.1) is thus discretized by merely replacing it by a set of algebraic functions. This set of algebraic functions is used as a surrogate model or meta-model. The expressions are linear expressions of the initial states and control parameters expressed as follows.

The affine structure of the meta-model is postulated as an affine algebraic expression of the form:

\[
y_{t+j} = \Psi_j(x_t^1, x_t^2, \ldots, x_t^n, u_1, u_2, \ldots, u_j) \approx \alpha_0^j + \sum_{i=1}^{n} \Psi_i(x_t^i) + \sum_{k=1}^{m} \sum_{l=1}^{j} \Psi^l(u_k^l)
\]  

(G.5)

where \( \alpha_0 \) is the average value for \( y_j \) and the coefficients are calculated via numerical integration as follows.

Consider \( N \) samples of the \( n \)-dimensional vector \( x = (x_t^1, x_t^2, \ldots, x_t^n, u_1, \ldots, u_j) \) randomly generated and uniformly distributed on a hypercube \( I^{n+m} \), where \( x_t^1, \ldots, x_t^n, u_1, \ldots, u_j \) represents the variables \( x_t^1, x_t^2, \ldots, x_t^n, u_1, u_2, \ldots, u_j \) scaled on the interval \([0,1]\). Then \( \alpha^j \) in (G.5) can be approximated by the mean value of \( \Psi_j(x) \) for each sample:

\[
\alpha_0^j = \frac{1}{N} \int_0^1 \Psi_j(x) \, dx \approx \lim_{N \to \infty} \frac{1}{N} \sum_{s=1}^{N} \Psi_j(x^s)
\]  

(G.6)

Similarly, \( \alpha_i \) and \( \alpha^j \) are computed:
\[ a_i = \int_0^1 \Psi_i x_i \Phi_1 x_i' dx_i' \approx \lim_{N \to \infty} \frac{1}{N} \sum_{s=1}^N \Psi_i x_i' \Phi_1 x_i' \tag{G.7} \]

\[ a^k_i = \int_0^1 \Psi_i^k u_i' \Phi_1 u_i' du_i' \approx \lim_{N \to \infty} \frac{1}{N} \sum_{s=1}^N \Psi_i^k u_i' \Phi_1 u_i' \tag{G.8} \]

where \( \Phi_1 \) is the first order scaled Legendre polynomial.

If we note \( x = x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_j \) we use data collected from the simulation for each time point of the time horizon and sampling of the space of all parameters to determine the outputs \( y_{t+j}(x_i) \) for \( i = 1, \ldots, s \) and \( j = 1, \ldots, N \).

This input data is used to train the meta-models \( \Psi(x), j = 1, \ldots, N \).

Finally, the reference tracking problem in (G.3), where the output \( y \) is to be driven to the set-point \( y_{sp} \), over a time horizon \( N \), is explicitly formulated as follows, by organising the mappings \( \{\Psi_j\}_{j\in[1:N]} \) under matrix forms:

\[
\begin{align*}
\min_{\Delta u} y^{*T}_{t+n} P y^{*}_{t+n} + \sum_{j=0}^{N-1} y^{*T}_{t+j} Q y^{*}_{t+j} + \delta u^{T}_{t+j} R \delta u_{t+j} \\
\text{s.t. } y^{*}_{t+j} &= y_{t+j} - y_{sp}, j = 1, \ldots, N \\
y_{t+j} &= A_j x_t + B_j u + C_j, j = 1, \ldots, N \\
u_{t+j} &= u_{t+j-1} + \delta u_{t+j}, j = 1, \ldots, N \\
y_{t+j} &\in Y, j = 0, \ldots, N \\
u_{t+j} &\in U, j = 0, \ldots, N 
\end{align*}
\tag{G.9}
\]

The algorithm is summarized as follows:

**Step 1.** Define bounds for the states and controls.

**Step 2.** Define a control horizon and sampling time.

**Step 3.** Perform multiple simulations exploring the space of controls and initial states.

**Step 4.** Build a \( \Psi_j \) extension for every time point along the time horizon.

The results above can easily translate and be applicable to a multiple input-multiple output dynamical system at no extra cost. This is because most of the computational cost is from simulation and the construction step of approximations from simulation data is not computationally demanding.