Linköping studies in science and technology. Thesis.
No. 1727
Licentiate's Thesis

On Structure Exploiting Numerical Algorithms for Model Predictive Control

Isak Nielsen

Division of Automatic Control
Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden
<http://www.control.isy.liu.se>
isani82@isy.liu.se

Linköping 2015

This is a Swedish Licentiate's Thesis.

Swedish postgraduate education leads to a Doctor's degree and/or a Licentiate's degree.

A Doctor's Degree comprises 240 ECTS credits (4 years of full-time studies).

A Licentiate's degree comprises 120 ECTS credits,
of which at least 60 ECTS credits constitute a Licentiate's thesis.

Linköping studies in science and technology. Thesis.

No. 1727

**On Structure Exploiting
Numerical Algorithms
for Model Predictive Control**

Isak Nielsen

isani82@isy.liu.se

www.control.isy.liu.se

Department of Electrical Engineering

Linköping University

SE-581 83 Linköping

Sweden

ISBN 978-91-7685-965-0

ISSN 0280-7971

Copyright © 2015 Isak Nielsen

Printed by LiU-Tryck, Linköping, Sweden 2015

Tillägnas min underbara familj

Abstract

One of the most common advanced control strategies used in industry today is Model Predictive Control (MPC), and some reasons for its success are that it can handle multivariable systems and constraints on states and control inputs in a structured way. At each time-step in the MPC control loop the control input is computed by solving a constrained finite-time optimal control (CFTOC) problem on-line. There exist several optimization methods to solve the CFTOC problem, where two common types are interior-point (IP) methods and active-set (AS) methods. In both these types of methods, the main computational effort is known to be the computation of the search directions, which boils down to solving a sequence of Newton-system-like equations. These systems of equations correspond to unconstrained finite-time optimal control (UFTOC) problems. Hence, high-performance IP and AS methods for CFTOC problems rely on efficient algorithms for solving the UFTOC problems.

The solution to a UFTOC problem is computed by solving the corresponding Karush-Kuhn-Tucker (KKT) system, which is often done using generic sparsity exploiting algorithms or Riccati recursions. When an AS method is used to compute the solution to the CFTOC problem, the system of equations that is solved to obtain the solution to a UFTOC problem is only changed by a low-rank modification of the system of equations in the previous iteration. This structured change is often exploited in AS methods to improve performance in terms of computation time. Traditionally, this has not been possible to exploit when Riccati recursions are used to solve the UFTOC problems, but in this thesis, an algorithm for performing low-rank modifications of the Riccati recursion is presented.

In recent years, parallel hardware has become more commonly available, and the use of parallel algorithms for solving the CFTOC problem and the underlying UFTOC problem has increased. Some existing parallel algorithms for computing the solution to this type of problems obtain the solution iteratively, and these methods may require many iterations to converge. Some other parallel algorithms compute the solution directly (non-iteratively) by solving parts of the system of equations in parallel, followed by a serial solution of a dense system of equations without the sparse structure of the MPC problem. In this thesis, two parallel algorithms that compute the solution directly (non-iteratively) in parallel are presented. These algorithms can be used in both IP and AS methods, and they exploit the sparse structure of the MPC problem such that no dense system of equations needs to be solved serially. Furthermore, one of the proposed parallel algorithms exploits the special structure of the MPC problem even in the parallel computations, which improves performance in terms of computation time even more. By using these algorithms, it is possible to obtain logarithmic complexity growth in the prediction horizon length.

Populärvetenskaplig sammanfattning

Modellprediktiv reglering (eng. *Model Predictive Control (MPC)*) är en reglerstrategi som kan användas för att styra system med flera styrsignaler och/eller utsignaler samt ta hänsyn till exempelvis begränsningar i styrdon. Ett exempel på ett sådant system är antisladdsystemet i en modern bil, där bromsarna kan styras automatiskt av exempelvis en MPC-regulator för att motverka att bilen sladdar. Den grundläggande principen med MPC-regulatorn är att styrsignalen beräknas på ett optimalt sätt genom att lösa ett optimeringsproblem. Detta optimeringsproblem består av en kostnadsfunktion som ska minimeras, en modell av hur systemet beter sig samt de begränsningar som finns på styrsignaler och/eller utsignaler. Detta optimeringsproblem måste lösas varje gång en styrsignal ska beräknas, och således är det viktigt att det finns effektiva metoder för att lösa denna typ av problem. Exempel på sådana metoder som är vanligt förekommande är så kallade inre punkt-metoder (eng. *interior-point (IP)*) och aktivmängd-metoder (eng. *active-set (AS)*). I dessa två metoder så löser man optimeringsproblemet genom att lösa ett antal förenklade delproblem. Då MPC-regulatorer ska användas för att styra processer som kräver att styrsignalen beräknas snabbt behövs effektiva algoritmer för att lösa dessa förenklade delproblem tidseffektivt, och det är sådana algoritmer denna avhandling fokuserar på.

De förenklade delproblemen löses genom att beräkna lösningen till ett linjärt ekvationssystem som kallas Karush-Kuhn-Tucker (KKT)-systemet. Detta ekvationssystem kan man lösa med exempelvis generella metoder eller med så kallade Riccatirekursioner som utnyttjar strukturen i problemet. När man använder en AS-metod för att lösa MPC-problemet så görs endast små strukturerade ändringar av KKT-systemet mellan varje förenklat delproblem, men detta har tidigare inte utnyttjats i Riccatirekursionen. I denna avhandling presenteras ett sätt att utnyttja detta genom att bara göra små ändringar av Riccatirekursionen för att på så vis minska beräkningstiden för att lösa delproblemet.

Nuförtiden är det vanligt att lösningen till optimeringsproblemet beräknas med hårdvara som klarar av parallella beräkningar, och därför har behovet av parallella algoritmer för att lösa MPC-problem ökat. Att man kan göra beräkningar parallellt innebär att flera delar av problemet kan beräknas samtidigt, och på så sätt minskar den totala beräkningstiden för att lösa optimeringsproblemet. I denna avhandling presenteras två olika parallella algoritmer som kan användas i både IP- och AS-metoder. Båda beräknar lösningen till de mindre delproblemen parallellt med ett förutbestämt antal steg, till skillnad från många andra parallella algoritmer där ett okänt (ofta stort) antal steg krävs för att beräkna lösningen. Båda algoritmerna utnyttjar och bevarar strukturen i MPC-problemet, men den senaste algoritmen kan utnyttja strukturen mer effektivt och kan i många fall lösa problemet snabbare.

Acknowledgments

I would like to take the opportunity to express my deepest gratitude to my co-supervisor Dr. Daniel Axehill. Your guidance and support over the past three years have been invaluable to me. Thank you for giving me feedback and inspiration that enables me to improve my research. Since I started here we have had many interesting discussions, and I really appreciate your enthusiasm that always motivates me. I am also very grateful for the valuable feedback and support I have gotten from my supervisor Prof. Anders Hansson.

Also, thank you Prof. Svante Gunnarsson and Prof. Fredrik Gustafsson for inviting me to start as a PhD student at the Automatic Control group. I would also like to thank Prof. Svante Gunnarsson for being an excellent head of the Division of Automatic Control, and Ninna Stensgård for helping me with administrative things and for making everything run smoothly in the group.

This thesis has been greatly improved by excellent comments from Dr. Sina Khoshfetrat Pakazad, Lic. Daniel Simon, Lic. Johan Dahlin, Lic. Jonas Linder, M. Sc. Hanna Nyqvist and fil. kand. Linnea Ernebro. I am sincerely grateful that you spent some of your valuable time on proof-reading this thesis! Also, thank you Dr. Gustaf Hendeby and Dr. Henrik Tidefelt for providing the L^AT_EX-class that has been used to write this thesis.

During my time as a PhD student at the Division of Automatic Control I have had the pleasure of making a lot of new friends. Unfortunately, I can only mention some of you here. Thank you Jonas Linder for being a really good roommate and friend. Also, thank you Johan Dahlin, Hanna Nyqvist, Niclas Evestedt, Clas Veibäck, Sina Khoshfetrat Pakazad, André Carvalho Bittencourt, Karl Granström, Daniel Simon, Emre Özkan, Tohid Ardeshiri, Manon Kok, Ylva Jung, Patrik Axelsson and Niklas Wahlström for being good friends and for sharing fun skiing and canoeing trips, parties and after parties, wine tasting and safari trips and much more. And thanks everyone else at the Automatic Control group for making this a stimulating and friendly working environment!

I would also like to thank all my friends outside the university for all fun stuff we have done together, and for all your support. Even though we don't see each other as often as before it still means a lot to me to have you as friends!

Financial support from the Swedish Research Council (VR) and CENIT (Center for Industrial Information Technology) are hereby gratefully acknowledged.

Finally, my wonderful parents Kaj and Anneli, my siblings Hugo, Julia and Axel and my lovely girlfriend Linnea: Your love and support mean everything to me, and I wouldn't be where I am today if it weren't for all of you! Thank you for always being there for me when I need it the most, and thank you for being the amazing persons you are. I love you all!

Linköping, September Anno Domini 2015
Isak Nielsen

Contents

Notation	xiii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Contributions	2
1.3 Thesis Outline	3
2 Optimization	5
2.1 Basic Concepts	6
2.2 Convex Optimization	7
2.3 Lagrange Duality	7
2.3.1 Weak and strong duality	8
2.4 Optimality Conditions	9
2.5 Quadratic Programming	10
2.6 Active-set Methods	10
2.6.1 Adding constraints	12
2.6.2 Removing constraints	13
2.6.3 Basic primal active-set method	13
2.7 Interior-point Methods	14
2.7.1 Basic primal interior-point method	15
3 Model Predictive Control	19
3.1 Defining the Model Predictive Control Problem	21
3.2 Solving the Model Predictive Control Problem	22
3.2.1 Interior-point methods	22
3.2.2 Active-set methods	25
3.3 Newton Step Computation	27
3.4 Solving the Karush-Kuhn-Tucker System using the Riccati Recur- sion	29
3.4.1 Derivation of the Riccati recursion	29
3.4.2 Handling singular Karush-Kuhn-Tucker systems	32
3.5 Computing Eliminated Dual Variables	33
3.A Model Predictive Control Formulations	36

4	Low-rank Modifications of Riccati Factorizations	37
4.1	Problem Formulation	38
4.2	Low-rank Modification of the Riccati Factorization	38
4.2.1	Removing control input constraints from the working set	39
4.2.2	Adding control input constraints to the working set	41
4.2.3	The impact on subsequent time-steps	43
4.2.4	Algorithms for modifying the Riccati factorization	45
4.3	Extension to General Constraints	46
4.3.1	Primal and dual problem	46
4.3.2	Methods to handle general constraints	48
4.4	Numerical Results	52
4.4.1	Modification of Riccati factorization	52
4.A	Proof of Lemma 4.1	56
4.B	Derivation of the Dual Problem	57
4.B.1	Construct the dual problem	57
4.B.2	Relation between primal and dual variables	61
5	Parallel Newton Step Computation	65
5.1	Problem Decomposition	67
5.1.1	Solution of the subproblems	70
5.1.2	Solution of a primal degenerate subproblem	72
5.2	Parallel Computation of Newton Step	75
5.2.1	Algorithms for parallel Newton step computation	77
5.3	Numerical Results	79
5.A	Proof of Theorem 5.4	82
5.B	Proof of Lemma 5.8	85
5.C	Proof of Theorem 5.10	86
6	Parallel Riccati Recursion	87
6.1	Problem Decomposition and Reduction	88
6.1.1	Splitting into independent parts	89
6.1.2	Eliminate local variables in a subproblem	89
6.1.3	Constructing the master problem	95
6.2	Computing the Riccati Recursion in Parallel	98
6.2.1	Parallel computation of the Riccati recursion	98
6.2.2	Parallel Riccati recursion algorithms	99
6.3	Numerical Results	101
6.4	Comparison with Chapter 5	105
6.A	Proof of Lemma 6.2	107
7	Conclusions and Further Work	109
7.1	Conclusions	109
7.2	Further Work	110
A	Linear Algebra	113
	Bibliography	117

Notation

NOTATION

Notation	Meaning
$\mathbb{R}$	Set of real numbers
$\mathbb{Z}$	Set of integers
$\mathbb{Z}_{i,j}$	Set of integers $\{i, i + 1, \dots, j - 1, j\}$
$\mathbb{S}^n$	Set of symmetric real matrices with n columns
$\mathbb{S}_+^n$	Set of positive semi-definite matrices with n columns
$\mathbb{S}_{++}^n$	Set of positive definite matrices with n columns
$\mathcal{R}(A)$	Range of a matrix A
$\mathcal{N}(A)$	Nullspace of a matrix A
$C^\perp$	The orthogonal complement of the set C
C^c	Complement of the set C
$A \geq 0$	The matrix A is positive semidefinite
$A > 0$	The matrix A is positive definite
$x \geq y$	Each component in x is larger than or equal to the corresponding component in y
$x > y$	Each component in x is strictly larger than the corresponding component in y
$[a, b]$	Interval of real numbers $z \in \mathbb{R}$ such that $a \leq z \leq b$
(a, b)	Interval of real numbers $z \in \mathbb{R}$ such that $a < z < b$
relint C	Relative interior of the set C
dom f	Domain of the function $f : \mathbb{R}^n \rightarrow \mathbb{R}$
rank A	The rank of a matrix A
∇f	Gradient of the function $f : \mathbb{R}^n \rightarrow \mathbb{R}$
$\nabla^2 f$	Hessian of the function $f : \mathbb{R}^n \rightarrow \mathbb{R}$
$\nabla_x f$	Gradient of the function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ with respect to the variable x
$\nabla_x^2 f$	Hessian of the function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ with respect to the variable x
x	Column vector of stacked vector components, i.e., $x = [x_0^T, \dots, x_N^T]^T$
$I(I_n)$	Identity matrix (of order n)
$x \triangleq y$	Definition of the variable x as y
$\forall x$	For all x
$\mathcal{O}(g(x))$	Complexity growth dominated by $g(x)$ when $x \rightarrow \infty$

ABBREVIATIONS

Abbreviation	Meaning
ADMM	Alternating Direction Method of Multipliers
AS	Active-Set
CFTOC	Constrained Finite-Time Optimal Control
CPU	Central Processing Unit
FPGA	Field Programmable Gate Array
GPU	Graphics Processing Unit
IP	Interior-Point
LICQ	Linear Independence Constraint Qualification
KKT	Karush-Kuhn-Tucker
LQ	Linear-Quadratic
MHE	Moving Horizon Estimation
MPC	Model Predictive Control
QP	Quadratic Program
UFTOC	Unconstrained Finite-Time Optimal Control

1

Introduction

Controlling systems and processes to behave in a desired way is one of the main goals in automatic control. It has become an important part of today's modern society and is found in a wide range of different technologies. Some examples of applications where automatic control is used today are, *e.g.*, process industries, such as pulp factories, control systems in fighter aircrafts, automotive applications, such as autonomous vehicles, cruise controllers and anti-break systems in cars, and many more. In all these examples, some kind of controller is used to achieve the desired behaviour of the system, and this controller can be of different types and complexities depending on for example the control objectives and the system dynamics.

1.1 Background and Motivation

Model Predictive Control (MPC) is a control scheme that combines optimization and automatic control. Essentially, an MPC controller consists of a dynamic model of the system, constraints on states and control inputs and an objective function. The control input is computed by predicting the behaviour of the system using the dynamic model, and minimizing (or maximizing) the objective function while satisfying the constraints. The possibility to easily handle multivariable dynamic models and constraints on states and control inputs has made MPC one of the most widely spread and commonly used advanced control strategies in industry (Maciejowski, 2002).

Traditionally MPC has been used mainly in petrochemical process plants where the sample time of the controller was long enough to compute a solution to the optimization problem. However, as computational hardware and algorithms for

solving the optimization problem in the MPC controller have developed, it has been possible to apply MPC to other processes that require much shorter sampling times and/or to more complex systems. Since the optimization problem is usually solved on-line, the usefulness of MPC heavily relies on efficient optimization routines for solving the MPC optimization problem.

The optimization problem can be of different types depending on, for example, the dynamical model and the constraints, and different methods can be used to compute the solution. Common types of methods are for instance interior-point (IP) and active-set (AS) methods, and in both these types of methods the main computational effort is spent while solving a sequence of subproblems. These subproblems can be solved using, for example, generic solvers or Riccati recursions, and the performance in terms of computation time of these types of methods is dependent on efficient algorithms for solving the subproblems.

The focus in this thesis is to present newly developed algorithms which can be used as subroutines in for example IP and AS solvers to compute the solution to the subproblems when applied to the optimization problem in MPC.

1.2 Contributions

The contributions in this thesis are presented in chapters 4, 5 and 6, and are based on published as well as unpublished material.

In Chapter 4, theory and algorithms for utilizing low-rank modifications of the Riccati recursion are presented. This chapter contains unpublished material, and it includes and extends the results presented in the publication

I. Nielsen, D. Ankelhed, and D. Axehill. Low-rank modification of Riccati factorizations with applications to model predictive control. In *Proceedings of the 52nd IEEE Conference on Decision and Control*, pages 3684–3690, Firenze, Italy, December 2013

The second contribution is on theory and algorithms for computing the Newton step for MPC problems in parallel and is presented in Chapter 5. This material is an edited version of the publication

I. Nielsen and D. Axehill. An $O(\log N)$ parallel algorithm for Newton step computation in model predictive control. In *Proceedings of the 19th IFAC World Congress*, pages 10505–10511, Cape Town, South Africa, August 2014

The editing has been made to improve the presentation of the results, and no additional technical results have been added.

The third contribution is presented in Chapter 6. This contribution is on theory and algorithms for computing the Riccati recursion in parallel in time, and will be presented in the publication

I. Nielsen and D. Axehill. A parallel structure-exploiting factorization algorithm with applications to model predictive control. Accepted for publication at the 54th IEEE Conference on Decision and Control, December 2015

1.3 Thesis Outline

The thesis is organized as follows. In Chapter 2, basic concepts and notation in optimization are introduced, and some common methods to solve convex optimization problems are outlined. In Chapter 3, the MPC problem is discussed and different methods to solve the underlying optimization problem are presented. The main purpose with this chapter is to provide a foundation for the contributions that are presented in chapters 4, 5 and 6.

In Chapter 4, it is shown how to utilize low-rank modifications of the Riccati recursion when it is used to compute Newton steps in an AS solver for MPC problems. The chapter is concluded with numerical results for the proposed algorithm. Chapter 5 contains theory, algorithms and numerical results for computing the Newton step directly in parallel. The algorithms in this chapter can be applied when solving MPC problems using, for example, IP or AS methods. In Chapter 6, theory and algorithms for computing the Riccati recursion directly in parallel are presented, and numerical results for a MATLAB implementation and an ANSI-C implementation of the proposed algorithm are shown.

The thesis is concluded in Chapter 7 and this chapter also contains ideas for further research on the presented topics. In addition, Appendix A contains basic results from linear algebra that is used in the thesis.

2

Optimization

Mathematical optimization, or simply *optimization*, is a framework where the best possible solution to a problem is sought for. Which solution that is regarded to be the best is dependent on the corresponding *optimization problem*. The optimization problem consists of an *objective function* and *constraints* that define the *feasible set* of the problem, *i.e.*, which solutions that are allowed. The best solution is the element that is in the feasible set, *i.e.*, satisfies all constraints, of the problem and minimizes, or maximizes depending on the problem definition, the objective function. Depending on the optimization problem being solved there can exist a unique solution, several solutions or no solution.

Three examples of optimization problems are driving a car between two cities as fast (i), as cheap (ii) or by driving as short distance as possible (iii), while keeping the speed limit and driving on the road. The objective function in the first problem is the total time consumed when travelling between the two cities, and in the second problem, it is the total cost for fuel, wear, etc. In the third problem, the objective function is the total distance travelled, and in all three problems the constraints are, *e.g.*, to keep the speed limits, to stay on the road and physical constraints such as the maximum power given by the car engine. By using these three objective functions three different optimization problems are obtained, all having (possibly) different optimal (or best) solutions. As mentioned earlier, each of the problems might have several solutions, a unique solution or no solution. For example, the problem of driving as short distance as possible has a unique solution if there is only one road that gives the shortest distance. If there are several roads between the cities of equal minimal distance, then there exist several solutions where each is equally good. If there are no roads at all between the two cities the problem does not have any solution.

The purpose of this chapter is to introduce the basic concepts of optimization and also to give a brief survey of some common families of optimization methods that are used to compute the optimal solution to certain types of optimization problems. The chapter and the introduced notation are inspired by Boyd and Vandenberghe (2004), which is an extensive reference on optimization.

2.1 Basic Concepts

Consider an optimization problem given in the form

$$\begin{aligned} & \underset{x}{\text{minimize}} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,m} \\ & && h_i(x) = 0, \quad i \in \mathbb{Z}_{1,p}, \end{aligned} \tag{2.1}$$

where $f_0(x)$ is the *objective function*, $f_i(x)$ are the *inequality constraint functions*, $h_i(x)$ are the *equality constraint functions* and $x \in \mathbb{R}^n$ is the *optimization variable*. Here, $\mathbb{Z}_{i,j}$ is used to denote the set of integers $\{i, i+1, \dots, j\}$. The *domain* $\mathcal{D} \subseteq \mathbb{R}^n$ of the optimization problem (2.1) is the set of all points where the objective and constraint functions are defined, i.e.,

$$\mathcal{D} \triangleq \bigcap_{i=0}^m \text{dom } f_i \cap \bigcap_{i=1}^p \text{dom } h_i, \tag{2.2}$$

and a point $x \in \mathcal{D}$ is said to be *feasible* if it satisfies the inequality and equality constraints in (2.1). If there exists at least one feasible $x \in \mathcal{D}$ the optimization problem (2.1) is said to be *feasible*, and *infeasible* otherwise. The set of all feasible points is denoted the *feasible set*.

The *optimal value* p^* is the infimal value of the objective function evaluated at all feasible points and is defined as

$$p^* \triangleq \inf \{f_0(x) \mid f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,m}, \quad h_i(x) = 0, \quad i \in \mathbb{Z}_{1,p}\}, \tag{2.3}$$

where $p^* = \infty$ if the optimization problem is infeasible, and $p^* = -\infty$ if the optimization problem is *unbounded from below*. A feasible point $x^* \in \mathcal{D}$ with $f_0(x^*) = p^*$ is called an *optimal point* or an *optimal solution* to (2.1).

An optimization problem

$$\begin{aligned} & \underset{\tilde{x}}{\text{minimize}} && \tilde{f}_0(\tilde{x}) \\ & \text{subject to} && \tilde{f}_i(\tilde{x}) \leq 0, \quad i \in \mathbb{Z}_{1,\tilde{m}} \\ & && \tilde{h}_i(\tilde{x}) = 0, \quad i \in \mathbb{Z}_{1,\tilde{p}}, \end{aligned} \tag{2.4}$$

is said to be equivalent to (2.1) if x^* can be trivially computed from the solution $\tilde{x}^*$ and vice versa. Equivalent problems can be used to compute the solution to the original problem by solving an equivalent problem in a simpler form.

2.2 Convex Optimization

Convex optimization problems is a class of optimization problems that attain some special properties which have made it useful in many different areas including, e.g., economics, mechanics, control engineering and circuit design. Before the definition of a convex optimization problem can be presented, some auxiliary definitions are needed. These are given in definitions 2.1, 2.2 and 2.3, and their properties are thoroughly discussed in standard literature such as Boyd and Vandenberghe (2004) and are only repeated here for the sake of completeness.

Definition 2.1 (Convex set). A set C is *convex* if for any two points $x, y \in C$ and any $\theta \in [0, 1]$ it holds that

$$\theta x + (1 - \theta)y \in C. \quad (2.5)$$

Definition 2.2 (Convex function). A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is *convex* if $\text{dom } f$ is a convex set and if for all points $x, y \in \text{dom } f$ and any $\theta \in [0, 1]$ it holds that

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y), \quad (2.6)$$

and *strictly convex* if strict inequality holds for $x \neq y$ and $\theta \in (0, 1)$.

Definition 2.3 (Concave function). A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is *concave* if $-f$ is convex and *strictly concave* if $-f$ is strictly convex.

Now consider the optimization problem

$$\begin{aligned} & \underset{x}{\text{minimize}} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,m} \\ & && a_i^T x = b_i, \quad i \in \mathbb{Z}_{1,p}, \end{aligned} \quad (2.7)$$

where the functions f_i for $i \in \mathbb{Z}_{0,m}$ are all convex. Then (2.7) is a *convex optimization problem*. Note that this definition of convex optimization problems requires that the equality constraints are affine functions of x .

2.3 Lagrange Duality

In Lagrange duality the constraints in the standard problem (2.1), with non-empty domain $\mathcal{D}$, are taken into account by augmenting the objective function using weighted sums of the constraint functions (Boyd and Vandenberghe, 2004). The weights are called Lagrange multipliers and are here denoted λ_i for the inequality constraints and ν_i for the equality constraints, giving the *Lagrangian*

$$L(x, \lambda, \nu) \triangleq f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x). \quad (2.8)$$

By minimizing the Lagrangian with respect to the variable x over the domain $\mathcal{D}$ the *Lagrange dual function* $g(\lambda, \nu)$ is obtained, i.e.,

$$g(\lambda, \nu) \triangleq \inf_{x \in \mathcal{D}} \left(f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x) \right). \quad (2.9)$$

This is a concave function that satisfies the relation

$$g(\lambda, \nu) \leq p^* \text{ if } \lambda \geq 0, \quad (2.10)$$

and takes on the value $-\infty$ if the Lagrangian is unbounded in x (Boyd and Vandenberghe, 2004).

Remark 2.4. The Lagrange dual function $g(\lambda, \nu)$ in (2.9) is concave even when the problem (2.1) is not convex.

The Lagrange dual function gives a lower bound on the optimal value p^* of (2.1) if $\lambda \geq 0$, and maximizing (2.9) gives the best possible lower bound on p^* that can be obtained from the Lagrange dual function. Finding this lower bound is equivalent to solving the optimization problem

$$\begin{aligned} & \underset{\lambda, \nu}{\text{maximize}} && g(\lambda, \nu) \\ & \text{subject to} && \lambda \geq 0, \end{aligned} \quad (2.11)$$

which is called the *Lagrange dual problem*, or simply the *dual problem*, of (2.1) and λ and ν are called the *dual variables*. Since the objective function $g(\lambda, \nu)$ to be maximized is concave and the constraints in (2.11) are convex, the dual problem (2.11) is a convex optimization problem (Boyd and Vandenberghe, 2004). Note that it is a convex problem even when the primal problem (2.1) is not.

2.3.1 Weak and strong duality

Let the optimal value of the dual problem (2.11) be denoted d^* . Then, from (2.10), it is clear that

$$d^* \leq p^*. \quad (2.12)$$

The property (2.12) is referred to as *weak duality* and holds even if the primal problem (2.1) is not convex and/or d^* or p^* are infinite. The *duality gap* is defined as the difference $p^* - d^*$ and is always non-negative.

Under certain conditions, the inequality (2.12) holds with equality, i.e., $d^* = p^*$, and the duality gap is zero. This property is referred to as *strong duality* and does not hold in general for an optimization problem. Conditions that certify that strong duality holds are called *constraint qualifications*. One such constraint qualification is given by Slater's theorem which states that strong duality holds for a convex optimization problem (2.7) given that Slater's condition holds. This condition states that there must exist a *strictly feasible* point for strong duality to hold. A weaker, refined, version of Slater's condition can be used to certify strong duality when some (or all) of the inequality constraint functions are affine. These conditions are given in definitions 2.5 and 2.6.

Definition 2.5 (Slater's condition). There exists an $x \in \text{relint } \mathcal{D}$ such that

$$f_i(x) < 0, \quad i \in \mathbb{Z}_{1,m}, \quad a_i^T x = b_i, \quad i \in \mathbb{Z}_{1,p}. \quad (2.13)$$

Definition 2.6 (Slater's refined condition). There exists an $x \in \text{relint } \mathcal{D}$ such that

$$f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,k}, \quad f_i(x) < 0, \quad i \in \mathbb{Z}_{k+1,m}, \quad a_i^T x = b_i, \quad i \in \mathbb{Z}_{1,p}, \quad (2.14)$$

where f_i for $i \in \mathbb{Z}_{1,k}$ are all affine functions.

Strong duality implies that if there exists a solution x^* to (2.7) with optimal value p^* , then there exists optimal dual variables λ^* and ν^* such that $g(\lambda^*, \nu^*) = d^* = p^*$.

2.4 Optimality Conditions

Consider an optimization problem given by (2.1) where the functions f_i for $i \in \mathbb{Z}_{0,m}$ and h_i for $i \in \mathbb{Z}_{1,p}$ are all differentiable and strong duality holds. Then the optimal primal and dual solutions to this optimization problem satisfy the so called Karush-Kuhn-Tucker (KKT) conditions (Boyd and Vandenberghe, 2004; Nocedal and Wright, 2006).

Definition 2.7 (Karush-Kuhn-Tucker conditions). The conditions

$$\nabla f_0(x) + \sum_{i=1}^m \lambda_i \nabla f_i(x) + \sum_{i=1}^p \nu_i \nabla h_i(x) = 0, \quad (2.15a)$$

$$f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,m}, \quad (2.15b)$$

$$h_i(x) = 0, \quad i \in \mathbb{Z}_{1,p}, \quad (2.15c)$$

$$\lambda_i \geq 0, \quad i \in \mathbb{Z}_{1,m}, \quad (2.15d)$$

$$\lambda_i f_i(x) = 0, \quad i \in \mathbb{Z}_{1,m}, \quad (2.15e)$$

are called the *Karush-Kuhn-Tucker conditions*.

The KKT conditions are necessary for optimality of the solution to (2.1). If the problem is convex the conditions are also sufficient for optimality, which is summarized in Theorem 2.8.

Theorem 2.8. Consider a convex optimization problem in the form in (2.7) that satisfies Slater's condition and has differentiable f_i for $i \in \mathbb{Z}_{0,m}$ and h_i for $i \in \mathbb{Z}_{1,p}$. Then the KKT conditions in Definition 2.7 are necessary and sufficient conditions for optimality, and any primal and dual pair $\tilde{x}$ and $(\tilde{\lambda}, \tilde{\nu})$ that satisfies the KKT conditions (2.15) are primal and dual optimal.

Proof: See Section 5.5 in Boyd and Vandenberghe (2004). □

2.5 Quadratic Programming

One important class of convex optimization problems are *quadratic programs* (QPs). Many problems can be directly formulated as QPs, but QP problems also arise as subproblems in methods for solving general constrained optimization problems (Nocedal and Wright, 2006). A QP is a convex optimization problem in the form (2.7) with affine constraint functions and a quadratic cost function, i.e.,

$$\begin{aligned} & \underset{x}{\text{minimize}} && \frac{1}{2}x^T Qx + l^T x + c \\ & \text{subject to} && A_I x \leq b_I \\ & && A_E x = b_E, \end{aligned} \tag{2.16}$$

where $x \in \mathbb{R}^n$, $Q \in \mathbb{S}_+^n$, $l \in \mathbb{R}^n$, $c \in \mathbb{R}$, $A_I \in \mathbb{R}^{m \times n}$, $b_I \in \mathbb{R}^m$, $A_E \in \mathbb{R}^{p \times n}$ and $b_E \in \mathbb{R}^p$. Since the inequality constraints in (2.16) are affine in the optimization variable x , Slater's refined condition is satisfied and, hence, Theorem 2.8 states that the KKT conditions in Definition 2.7 are necessary and sufficient for optimality. The KKT conditions for a QP problem reduce to

$$Qx + l + A_I^T \lambda + A_E^T v = 0, \tag{2.17a}$$

$$A_I x \leq b_I, \tag{2.17b}$$

$$A_E x = b_E, \tag{2.17c}$$

$$\lambda \geq 0, \tag{2.17d}$$

$$\lambda_i (a_i^T x - b_i) = 0, \quad i \in \mathbb{Z}_{1,m}, \tag{2.17e}$$

and, hence, any $\tilde{x}$ and $(\tilde{\lambda}, \tilde{v})$ that satisfy (2.17) are primal and dual global optimal solutions to (2.16). Here the notation a_i^T and b_i are used to denote the i :th row of A_I and b_I , respectively, i.e.,

$$\begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix} \triangleq A_I, \quad \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \triangleq b_I. \tag{2.18}$$

The QP problem (2.16) can be solved using any method for convex optimization. In this thesis the focus is on numerical algorithms that can be used as subroutines in optimization methods where the step direction is computed as a Newton step. Examples of such optimization methods can be AS methods and IP methods.

2.6 Active-set Methods

Active-set methods is a class of methods that has been widely used since the 1970s (Nocedal and Wright, 2006). The basic idea in this type of method is to find the set of equality constraints and inequality constraints in (2.16) that hold with equality at the optimal solution. To simplify notation, the set of all equality constraints and the set of all inequality constraints for a QP problem in the form (2.16) are introduced in Definition 2.9.

Definition 2.9. Let the sets $\mathcal{E}$ and $\mathcal{I}$ be defined as the indices of all the equality constraints and inequality constraints, respectively, in the problem (2.16). _____

The set of all indices of equality constraints and inequality constraints that hold with equality, or are *active*, is called the *active set* and is defined in Definition 2.10.

Definition 2.10 (Active set). The *active set* at any feasible x in (2.16) is denoted $\mathcal{A}(x)$ and consists of all indices of constraints in (2.16) that hold with equality at x , i.e.,

$$\mathcal{A}(x) \triangleq \mathcal{E} \cup \{i \in \mathcal{I} \mid a_i^T x = b_i\}. \quad (2.19)$$

Remark 2.11. When a constraint is said to be in the active set, it should be interpreted as its corresponding index is in the active set. _____

Next, the linear independence constraint qualification is defined and it plays an important role in optimization methods and will be used later on.

Definition 2.12 (Linear independence constraint qualifications). For the constraints with indices in the active set $\mathcal{A}(x)$ the *linear independence constraint qualification* (LICQ) holds if the constraint gradients are linearly independent. When LICQ is violated it is referred to as *primal degeneracy*. _____

If the optimal active set $\mathcal{A}(x^*)$ is known a priori, the optimal solution can be computed by solving the equality constrained QP problem

$$\begin{aligned} & \underset{x}{\text{minimize}} && \frac{1}{2} x^T Q x + l^T x + c \\ & \text{subject to} && a_i^T x = b_i, \quad i \in \mathcal{A}(x^*). \end{aligned} \quad (2.20)$$

However, the optimal active set is seldom known prior to the solution has been computed, and the fundamental idea with an AS solver is to iteratively search for this optimal active set by adding and removing constraint indices from the so called *working set*. The working set at AS iteration j is denoted $\mathcal{W}_j$ and consists of the indices of all equality constraints and the subset of the indices of the inequality constraints that are forced to hold with equality. Similarly let $\mathcal{W}_j^c$ be the complement of the working set $\mathcal{W}_j$ containing all the inequality constraints that are temporarily disregarded. A requirement on $\mathcal{W}_j$ is that LICQ holds for the constraints in this set (Nocedal and Wright, 2006).

Let x_j denote the j :th iterate in the AS solver. The step to the next iteration x_{j+1} is computed by solving an equality constrained QP where the constraints in $\mathcal{W}_j$ are forced to hold with equality and the remaining constraints are temporarily disregarded, i.e.,

$$\begin{aligned} & \underset{\hat{x}_{j+1}}{\text{minimize}} && \frac{1}{2} \hat{x}_{j+1}^T Q \hat{x}_{j+1} + l^T \hat{x}_{j+1} + c \\ & \text{subject to} && a_i^T \hat{x}_{j+1} = b_i, \quad i \in \mathcal{W}_j. \end{aligned} \quad (2.21)$$

When $\hat{x}_{j+1}$ is computed, there are two different outcomes which requires different actions to be taken. If $\hat{x}_{j+1} = x_j$, then this point minimizes the objective function in the current working set $\mathcal{W}_j$. Depending on the dual variables λ_i^* for $i \in \mathcal{W}_j$, the iterate $\hat{x}_{j+1}$ is either an optimal point of the original problem, or at least one constraint has to be removed from the working set. The other outcome is when $\hat{x}_{j+1} \neq x_j$, and in that case a step from x_j towards $\hat{x}_{j+1}$ along $p_j \triangleq \hat{x}_{j+1} - x_j$ is taken. These different outcomes will be described below.

2.6.1 Adding constraints

For an iterate where $\hat{x}_{j+1} \neq x_j$, the new iterate is computed by taking a step along p_j such that

$$x_{j+1} = x_j + \alpha_j p_j, \quad \alpha_j \in [0, 1], \quad p_j \triangleq \hat{x}_{j+1} - x_j. \quad (2.22)$$

If the point $x_j + p_j$ is feasible with respect to all constraints in (2.16), then $\alpha_j = 1$ is used to update the new iterate in (2.22) as $x_{j+1} = x_j + p_j = \hat{x}_{j+1}$. If $x_j + p_j$ is not feasible, then one or more constraints are violated when moving from x_j to $x_j + p_j$. These violated constraints are called *blocking constraints* and the step-length parameter α_j is chosen such that all blocking constraints (and hence all constraints) remain satisfied.

All constraints $i \in \mathcal{W}_j$ are satisfied since they are included as equality constraints in (2.21). Hence, the blocking constraints must be among the constraints $i \in \mathcal{W}_j^c$ (i.e. $i \notin \mathcal{W}_j$) and α_j can be computed by determining which constraint $i \in \mathcal{W}_j^c$ that is closest to x_j along the direction given by p_j . Inserting the point (2.22) into each separate constraint $i \in \mathcal{W}_j^c$ gives that the relation

$$a_i^T(x_j + \alpha_j p_j) = a_i^T x_j + \alpha_j a_i^T p_j \leq b_i, \quad (2.23)$$

must hold for all $i \in \mathcal{W}_j^c$. Since x_j is a feasible point it follows that $a_i^T x_j \leq b_i$ and, hence, (2.23) is satisfied for all $\alpha_j \geq 0$ whenever $a_i^T p_j \leq 0$. This can be seen by writing (2.23) as

$$a_i^T x_j \leq b_i \leq b_i - \alpha_j a_i^T p_j, \quad \text{where } a_i^T p_j \leq 0, \quad (2.24)$$

and, hence, constraint i cannot be a blocking constraint. If however $a_i^T p_j > 0$, then (2.23) is satisfied only for certain choices of the step-length parameter α_j , giving the upper limit as

$$a_i^T(x_j + \alpha_j p_j) = a_i^T x_j + \alpha_j a_i^T p_j \leq b_i \iff \alpha_j \leq \frac{b_i - a_i^T x_j}{a_i^T p_j}. \quad (2.25)$$

To maximize the decrease of the objective function, α_j is chosen as large as possible in the set $[0, 1]$ while retaining primal feasibility. Hence, the step-length parameter is chosen as

$$\alpha_j \triangleq \min \left(1, \min_{i \in \mathcal{W}_j^c, a_i^T p_j > 0} \frac{b_i - a_i^T x_j}{a_i^T p_j} \right). \quad (2.26)$$

When there exist one or more blocking constraints a new working set $\mathcal{W}_{j+1}$ is constructed by adding the blocking constraint that corresponds to the minimal step-length parameter in (2.26) to $\mathcal{W}_j$. Let the index of this constraint be i , then the new working set $\mathcal{W}_{j+1}$ is obtained as

$$\mathcal{W}_{j+1} = \mathcal{W}_j \cup \{i\}. \quad (2.27)$$

To find the new step direction after modifying the working set a new equality constrained QP in the form (2.21) is solved, and the procedure described above is repeated until a point $\hat{x}_{j+1} = x_j$ that minimizes the current working set $\mathcal{W}_j$ for some AS iteration j is found. By defining $\lambda_i \triangleq 0$ if $i \in \mathcal{W}_j^c$ and noting that the point x_{j+1} is a feasible point that minimizes the objective function in (2.21) it can be concluded that (2.17a), (2.17b), (2.17c) and (2.17e) in the KKT conditions are satisfied.

2.6.2 Removing constraints

According to the discussion above, (2.17a), (2.17b), (2.17c) and (2.17e) in the KKT conditions are satisfied for the case when $\hat{x}_{j+1} = x_j$. If also

$$\lambda_i \geq 0, \quad i \in \mathcal{W}_j \cap \mathcal{I}, \quad (2.28)$$

then (2.17d) is satisfied and the point $\hat{x}_{j+1}$ satisfies the KKT optimality conditions (2.17) for the QP problem (2.16), and is hence an optimal solution to (2.16). If however $\lambda_r < 0$ for some $r \in \mathcal{W}_j \cap \mathcal{I}$, then the KKT condition (2.17d) is violated and the point $\hat{x}_{j+1}$ cannot be optimal.

When $\lambda_r < 0$ for some $r \in \mathcal{W}_j \cap \mathcal{I}$ it may be possible to decrease the value of the objective function by dropping one of these constraints, *i.e.*, removing it from the working set. Any of the constraints corresponding to a negative dual variable can be removed, but a common choice is to remove the most negative one. However, this does not guarantee that the decrease in the objective function is larger than if another constraint is removed (Nocedal and Wright, 2006). Since constraint r has been removed from the working set, $\mathcal{W}_j$ is modified as

$$\mathcal{W}_{j+1} = \mathcal{W}_j \setminus \{r\}, \quad (2.29)$$

and a new equality constrained QP in the form given by (2.21) with equality constraints given by the indices in $\mathcal{W}_{j+1}$ has to be solved.

2.6.3 Basic primal active-set method

The procedure which is presented in this section describes the basic components of a traditional AS method, and is summarized in Algorithm 1. Inspiration for this algorithm is taken from Algorithm 16.3 in Nocedal and Wright (2006). In this algorithm, only one constraint is added or removed at each AS iteration. However, there exist methods that changes the working set by adding or removing several constraints at each iteration, see for instance Axehill (2008).

At Line 1 in Algorithm 1, a feasible starting point x_0 for the solver need to be found. Computing this feasible starting point for a general QP can be done using

Algorithm 1 Basic active-set solver for convex QPs (Nocedal and Wright, 2006)

```

1: Compute a feasible starting point  $x_0$ 
2: Set  $\mathcal{W}_0$  to be a subset of the active constraints at  $x_0$ 
3: for  $j = 0, 1, 2, \dots$  do
4:   Compute  $\hat{x}_{j+1}$  by solving (2.21)
5:   if  $\hat{x}_{j+1} = x_j$  then
6:     Compute the Lagrange multipliers  $\hat{\lambda}_i$  that satisfy (2.17)
7:     if  $\hat{\lambda}_i \geq 0$  for all  $i \in \mathcal{W}_j \cap \mathcal{I}$  then
8:        $x^* := x_j$ 
9:       STOP
10:    else
11:       $r := \operatorname{argmin}_{i \in \mathcal{W}_j \cap \mathcal{I}} \hat{\lambda}_i$ 
12:      Update iterate and working set:  $x_{j+1} := x_j$ ,  $\mathcal{W}_{j+1} := \mathcal{W}_j \setminus \{r\}$ 
13:    end if
14:  else
15:    Compute  $p_j := \hat{x}_{j+1} - x_j$  and step-length parameter  $\alpha_j$  from (2.26)
16:    Update iterate:  $x_{j+1} := x_j + \alpha_j p_j$ 
17:    if  $\alpha_j < 1$  (blocking constraints) then
18:      Set  $i$  to the blocking constraint giving  $\alpha_j$ 
19:      Update working set:  $\mathcal{W}_{j+1} := \mathcal{W}_j \cup \{i\}$ 
20:    else
21:       $\mathcal{W}_{j+1} := \mathcal{W}_j$ 
22:    end if
23:  end if
24: end for

```

a so called *phase I algorithm* and can take up to one-third to one-half of the total solution time (Goldfarb and Idnani, 1983). At Line 2 in Algorithm 1, the initial working set is a subset of the active constraints at x_0 , and the iteration sequence to the solution is different for different $\mathcal{W}_0$ (Nocedal and Wright, 2006).

2.7 Interior-point Methods

In this section a brief introduction of IP methods for convex optimization is given. Consider a convex optimization problem in the form (2.7), i.e.,

$$\begin{aligned}
 & \underset{x}{\text{minimize}} && f_0(x) \\
 & \text{subject to} && f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,m} \\
 & && Ax = b,
 \end{aligned} \tag{2.30}$$

with twice continuously differentiable functions f_i for $i \in \mathbb{Z}_{0,m}$ and $A \in \mathbb{R}^{p \times n}$ with $\text{rank } A = p < n$. Assume that the problem is solvable and strictly feasible. The problem (2.30) can be solved using an IP method, where a sequence of equality

constrained convex optimization problems

$$\begin{aligned} & \underset{x}{\text{minimize}} && t f_0(x) + \phi(x) \\ & \text{subject to} && Ax = b, \end{aligned} \quad (2.31)$$

are solved for increasing $t > 0$ (Boyd and Vandenberghe, 2004). Here, $\phi(x)$ is the *logarithmic barrier function* given by

$$\phi(x) \triangleq - \sum_{i=1}^m \log(-f_i(x)), \quad (2.32)$$

which is used to approximate the indicator function for each of the inequality constraints. Let $x^*(t)$ be the optimal solution to (2.31) for a $t > 0$. Then $x^*(t)$ is called a *central point* and the set of points $x^*(t)$ for $t > 0$ is called the *central path* associated with problem (2.30). Every central point is strictly feasible and for each t , the corresponding central point $x^*(t)$ and the associated dual feasible pair $(\lambda^*(t), v^*(t))$ satisfy the perturbed KKT conditions

$$\nabla f_0(x) + \sum_{i=1}^m \lambda_i \nabla f_i(x) + A^T v = 0, \quad (2.33a)$$

$$f_i(x) \leq 0, \quad i \in \mathbb{Z}_{1,m}, \quad (2.33b)$$

$$Ax = b, \quad i \in \mathbb{Z}_{1,p}, \quad (2.33c)$$

$$\lambda_i \geq 0, \quad i \in \mathbb{Z}_{1,m}, \quad (2.33d)$$

$$-\lambda_i f_i(x) = \frac{1}{t}, \quad i \in \mathbb{Z}_{1,m}. \quad (2.33e)$$

Furthermore, the duality gap associated with $x^*(t)$ and $(\lambda^*(t), v^*(t))$ is given by

$$f_0(x^*(t)) - g(\lambda^*(t), v^*(t)) = m/t, \quad (2.34)$$

which gives the relation

$$f_0(x^*(t)) - p^* \leq m/t, \quad (2.35)$$

and hence $x^*(t)$ is no more than m/t -suboptimal (Boyd and Vandenberghe, 2004). Note that the perturbed KKT conditions (2.33) approach the KKT conditions (2.15) in Definition 2.7 as $t \rightarrow \infty$.

Remark 2.13. The problem (2.31) can be solved by any method for linearly constrained convex optimization (Boyd and Vandenberghe, 2004). However, in this thesis Newton methods are used.

2.7.1 Basic primal interior-point method

In Algorithm 2, a simple basic primal IP method called the *barrier method* is introduced. This algorithm is based on Algorithm 11.1 in Boyd and Vandenberghe (2004), and it computes a sequence of central points for increasing t until an ϵ -suboptimal solution, i.e., a solution for $t > m/\epsilon$, is found. Computation of these central points is where most of the computational effort in IP methods is spent.

Algorithm 2 The Barrier Method (Boyd and Vandenberghe, 2004)

```

1: Initialize  $x, t := t^{(0)} > 0, \mu > 1$ , tolerance  $\epsilon > 0$ 
2: loop
3:   Centering step: Compute  $x^*(t)$  by solving (2.31), starting at  $x$ 
4:   Update:  $x := x^*(t)$ 
5:   if Stopping criterion:  $m/t < \epsilon$  then
6:     STOP
7:   else
8:     Increase  $t$ :  $t := \mu t$ 
9:   end if
10: end loop

```

The centering step at Line 3 in Algorithm 2 consists of solving a problem in the form (2.31), which can be done by computing the solution to the perturbed KKT system (2.33) (Boyd and Vandenberghe, 2004). For a primal IP method, this is done by eliminating the dual variable λ_i in the perturbed KKT system (2.33). The elimination is done using (2.33e) to compute λ_i as

$$\lambda_i = \frac{1}{-t f_i(x)}, \quad (2.36)$$

which inserted in (2.33a) gives the modified KKT system

$$\nabla f_0(x) + \sum_{i=1}^m \frac{1}{-t f_i(x)} \nabla f_i(x) + A^T v = 0, \quad (2.37a)$$

$$Ax = b, \quad (2.37b)$$

which are also the KKT conditions for (2.31). These can be solved using Newton's method starting at a feasible x . See for example Boyd and Vandenberghe (2004); Nocedal and Wright (2006) for a description of Newton's method. In Boyd and Vandenberghe (2004), it is shown that the Newton steps when solving the centering problem (2.31) can be interpreted as Newton steps for solving the modified KKT system in a particular way. Furthermore, it is shown that the Newton step Δx_{nt} and the corresponding dual variable v_{nt} are given by the unique solution to the linear equations

$$\begin{bmatrix} t \nabla^2 f_0(x) + \nabla^2 \phi(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{nt} \\ v_{nt} \end{bmatrix} = - \begin{bmatrix} t \nabla f_0(x) + \nabla \phi(x) \\ 0 \end{bmatrix}, \quad (2.38)$$

where $v = (1/t)v_{nt}$. Hence, solving the centering step (2.31) requires the solution to a sequence of linear systems of equations in the form (2.38). Solving these systems of equations fast is crucial to the performance of an IP method.

Remark 2.14. The KKT matrix in (2.38) must be non-singular for Δx_{nt} to be a proper Newton step (Boyd and Vandenberghe, 2004). In this reference, some conditions that certify non-singularity of the KKT matrix are presented.

The system of equations (2.38) that defines a Newton step to the centering problem (2.31) can be interpreted as the optimality conditions for the equality constrained QP problem that is obtained by inserting the second-order Taylor expansions of $f_0(x)$ and $\phi(x)$ around x into (2.31). The Taylor expansions are given by

$$f_0(x + \Delta x) \approx f_0(x) + \nabla f_0(x)^T \Delta x + \frac{1}{2} \Delta x^T \nabla^2 f_0(x) \Delta x, \quad (2.39)$$

$$\phi(x + \Delta x) \approx \phi(x) + \nabla \phi(x)^T \Delta x + \frac{1}{2} \Delta x^T \nabla^2 \phi(x) \Delta x, \quad (2.40)$$

and inserting these into the centering problem (2.31) gives

$$\begin{aligned} \underset{\Delta x}{\text{minimize}} \quad & \frac{1}{2} \Delta x^T \left(t \nabla^2 f_0(x) + \nabla^2 \phi(x) \right) \Delta x + \left(t \nabla f_0(x)^T + \nabla \phi(x)^T \right) \Delta x + \\ & t f_0(x) + \phi(x) \\ \text{subject to} \quad & A \Delta x = 0. \end{aligned} \quad (2.41)$$

The equality constraints in (2.41) are obtained by inserting $x + \Delta x$ into the equality constraints in the centering problem (2.31), i.e.,

$$A(x + \Delta x) = Ax + A\Delta x = b \stackrel{Ax=b}{\iff} A\Delta x = 0. \quad (2.42)$$

It is clear that the KKT optimality conditions for (2.41) is given by the system of equations in (2.38). Hence, the equality constrained QP problem (2.41) can be solved directly by solving the systems of equations (2.38), and the centering problem (2.31) can be solved by computing the solution to a sequence of equality constrained QP problems in the form (2.41).

3

Model Predictive Control

In this chapter basic notations and concepts for MPC are introduced and some common methods for solving MPC problems are presented. These notations and concepts form a foundation for the theory presented in chapters 4, 5 and 6.

MPC is a control strategy where the control input that is applied to the controlled plant is computed by solving an optimization problem. It has become one of the most widely used advanced control strategies in industry today, and some important reasons for its success are that it can handle multivariable systems and constraints on control inputs and state variables in a structured way (Maciejowski, 2002). The solution to the optimization problem can be computed on-line or off-line. One technique for solving the optimization problem off-line is explicit MPC, where the state-space is divided into polyhedral regions and an affine state feedback is computed off-line for each region, see for example Bemporad et al. (2002). Given the current state during run-time, the correct feedback can be applied by determining which region the current state belongs to. However, the complexity and memory requirements of computing and storing the regions and control laws might be intractable already for small-sized problems (Grancharova et al., 2003). Hence, explicit MPC is used mostly for problems of small dimensions.

In this thesis, methods for computing the solution to the optimization problem on-line are considered. The optimization problem that is the core of the MPC problem is a constrained finite-time optimal control (CFTOC) problem. The MPC problem and the corresponding CFTOC problem can be of various types depending on, e.g., which application, controlled system and problem formulation that is used. Some common types are linear MPC, nonlinear MPC and hybrid MPC, and in most cases the effort spent when solving the corresponding CFTOC problem boils down to solving Newton-system-like equations that correspond to un-

constrained finite-time optimal control (UFTOC) problems. Hence, much focus has been spent on solving this type of problems efficiently when it has the special form from MPC, see, e.g., Jonson (1983); Rao et al. (1998); Hansson (2000); Vandenberghe et al. (2002); Axehill and Hansson (2006); Axehill et al. (2010); Axehill (2008); Axehill and Hansson (2008); Diehl et al. (2009); Nielsen et al. (2013); Nielsen and Axehill (2014).

The basic idea in MPC is to use a dynamic model of the controlled system to predict the behaviour over a prediction horizon, and optimizing the control signal over this prediction horizon to minimize a performance criterion. In this thesis discrete-time dynamic models are considered. The prediction is performed over a prediction horizon N steps into the future, and longer prediction horizons generally generates more computationally demanding optimization problems but better control performance. The MPC controller operates using a receding horizon strategy, meaning that the control signal at time t_0 is computed using the predictions up to $t_0 + N$, and the control input at time t_1 is computed using the predictions up to time $t_1 + N$, etc. This can be seen in Figure 3.1, where the control signal at time $t = 20$ is about to be computed. Here a prediction horizon of $N = 20$ is used, and the predicted control input and output variable are marked in red. The blue dashed line symbolizes that the predicted output and the real output might be different. At each step in the MPC control loop the measured (or estimated) current state is used as initial value and hence the MPC controller operates in closed loop. The basic steps in an MPC control loop are summarized in Algorithm 3.

Figure 3.1: Here, the control signal at time $t = 20$ is computed for an MPC problem with $N = 20$. The predicted output and control inputs are marked in red. The blue dashed line indicates that the predicted output and the real output might be different.

Algorithm 3 Basic MPC operation

-
- 1: **repeat**
 - 2: Measure (or estimate) current state at time t
 - 3: Solve the optimization problem over the prediction horizon t to $t + N$ to compute the control inputs
 - 4: Apply the first computed control input at time t
 - 5: **until** Break by user
-

3.1 Defining the Model Predictive Control Problem

Linear MPC problems are the most common type of MPC problems and will be used to present the algorithms in this thesis. Note, however, that similar linear algebra that is used to solve linear MPC problems can also be used in subroutines for solving, e.g., nonlinear and hybrid MPC problems (Axehill, 2008). In the linear MPC problem used in this thesis a quadratic cost function is minimized over a prediction horizon N subject to equality and inequality constraints on states and control inputs. The corresponding CFTOC problem for the linear MPC problem is

$$\begin{aligned}
 &\underset{x^p, u^p}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{x,t}^p & Q_{xu,t}^p \\ (Q_{xu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} x_t^p \\ u_t^p \end{bmatrix} + \begin{bmatrix} l_{x,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} x_t^p \\ u_t^p \end{bmatrix} + c_t^p \right) + \\
 &&& \frac{1}{2} x_N^p{}^T Q_{x,N}^p x_N^p + l_{x,N}^p{}^T x_N^p + c_N^p \\
 &\text{subject to} && x_0^p = \bar{x} \\
 &&& x_{t+1}^p = A_t^p x_t^p + B_t^p u_t^p + a_t^p, \quad t \in \mathbb{Z}_{0,N-1} \\
 &&& H_{x,t}^p x_t^p + H_{u,t}^p u_t^p + h_t^p \leq 0, \quad t \in \mathbb{Z}_{0,N-1} \\
 &&& H_{x,N}^p x_N^p + h_N^p \leq 0,
 \end{aligned} \tag{3.1}$$

where $x_t^p \in \mathbb{R}^{n_x}$ are the states, $u_t^p \in \mathbb{R}^{n_{u,t}}$ are the control inputs, and $H_{x,t}^p \in \mathbb{R}^{n_{c,t} \times n_x}$, $H_{u,t}^p \in \mathbb{R}^{n_{c,t} \times n_{u,t}}$ and $h_t^p \in \mathbb{R}^{n_{c,t}}$ describe the inequality constraints. The superscript “ p ” denotes a primal problem, and the choice of this notation will later become clear. In this thesis, variables in sans-serif such as

$$x^p \triangleq \begin{bmatrix} x_0^p \\ \vdots \\ x_N^p \end{bmatrix}, \tag{3.2}$$

denotes a vector consisting of stacked vector components. The equality constraints in the CFTOC problem (3.1) are the discrete-time dynamics constraints of the controlled system.

Assumption 3.1.

$$Q_t^p \triangleq \begin{bmatrix} Q_{x,t}^p & Q_{xu,t}^p \\ (Q_{xu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \in \mathbb{S}_+^{n_x + n_{u,t}}, \quad t \in \mathbb{Z}_{0,N-1}, \quad Q_{x,N}^p \in \mathbb{S}_+^{n_x}. \tag{3.3}$$

Assumption 3.2.

$$Q_{u,t}^p \in \mathbb{S}_{++}^{n_{u,t}}, \quad t \in \mathbb{Z}_{0,N-1}. \quad (3.4)$$

If the objective function in the CFTOC problem (3.1) satisfies assumptions 3.1 and 3.2, it is a convex QP problem in the form (2.16) with a unique solution. In the following, assumptions 3.1 and 3.2 hold unless stated otherwise.

The CFTOC problem (3.1) can be cast in a more compact form using similar notation as in (2.16), giving

$$\begin{aligned} \underset{x^p, u^p}{\text{minimize}} \quad & \frac{1}{2} \begin{bmatrix} x^p \\ u^p \end{bmatrix}^T \begin{bmatrix} Q_x^p & Q_{xu}^p \\ (Q_{xu}^p)^T & Q_u^p \end{bmatrix} \begin{bmatrix} x^p \\ u^p \end{bmatrix} + \begin{bmatrix} l_x^p \\ l_u^p \end{bmatrix}^T \begin{bmatrix} x^p \\ u^p \end{bmatrix} + c^p \\ \text{subject to} \quad & A^p x^p + B^p u^p + a^p = 0 \\ & H_x^p x^p + H_u^p u^p + h^p \leq 0, \end{aligned} \quad (3.5)$$

where $Q_x^p, Q_{xu}^p, Q_u^p, l_x^p, l_u^p, c^p, A^p, B^p, a^p, H_x^p, H_u^p$ and h^p are defined in Appendix 3.A, and x^p and u^p are the stacked x_t^p and u_t^p as before.

3.2 Solving the Model Predictive Control Problem

When Assumption 3.1 holds the CFTOC problem (3.1) is a convex QP. Hence, any method for solving this type of problem can be applied to (3.1) to find the optimal solution. In this section IP and AS methods that compute the search directions using the Newton method are presented. The section should not be considered a survey on QP methods for MPC, but rather a motivation for the results presented in chapters 4, 5 and 6.

3.2.1 Interior-point methods

The CFTOC problem in MPC can be solved using IP methods that exploit the structure in the CFTOC problem, see, e.g., Wright (1993); Rao et al. (1998); Hansson (2000); Axehill et al. (2007). In Section 2.7 a basic IP method to solve a convex optimization problem was introduced, and here it will be shown that the main computational effort when using this method to solve the CFTOC problem (3.1) is spent when solving a sequence of UFTOC problems.

In Section 2.7 it was argued that a convex optimization problem is solved by solving a sequence of centering problems in the form (2.31), where each centering problem is solved by computing a sequence of Newton steps corresponding to equality constrained QPs in the form (2.41). For the CFTOC problem (3.1) each of these equality constrained QPs can be interpreted as a second-order Taylor expansion of the centering problem (2.31) around a feasible point $(\bar{x}^p, \bar{u}^p)$. By using the compact form of the CFTOC problem as in (3.5), the Hessian and gradient of the

objective function f_0 are

$$s\nabla^2 f_0(x^p, u^p) = s \begin{bmatrix} Q_x^p & Q_{xu}^p \\ (Q_{xu}^p)^T & Q_u^p \end{bmatrix}, \quad s\nabla f_0(x^p, u^p) = s \begin{bmatrix} Q_x^p & Q_{xu}^p \\ (Q_{xu}^p)^T & Q_u^p \end{bmatrix} \begin{bmatrix} x^p \\ u^p \end{bmatrix} + s \begin{bmatrix} l_x^p \\ l_u^p \end{bmatrix}, \quad (3.6)$$

where s is used here instead of t in (2.31) and (2.38) to avoid confusion with the time index t in (3.1). To compute the gradient and Hessian of the logarithmic barrier function in (2.32), i.e., $\nabla\phi(\bar{x}^p, \bar{u}^p)$ and $\nabla^2\phi(\bar{x}^p, \bar{u}^p)$, it is noted from (2.32) that

$$\phi(x, u) = - \sum_{t=0}^{N-1} \sum_{i=1}^{n_{c,t}} \log(-H_{x,i,t}^p x_t - H_{u,i,t}^p u_t - h_{i,t}^p) - \sum_{i=1}^{n_{c,N}} \log(-H_{x,i,N}^p x_N - h_{i,N}^p), \quad (3.7)$$

and hence $\nabla\phi(x, u)$ and $\nabla^2\phi(x, u)$ will have the structure

$$\nabla\phi(x, u) = \begin{bmatrix} \nabla_{x_0}\phi(x_0, u_0) \\ \vdots \\ \nabla_{x_N}\phi(x_N) \\ \nabla_{u_0}\phi(x_0, u_0) \\ \vdots \\ \nabla_{u_{N-1}}\phi(x_{N-1}, u_{N-1}) \end{bmatrix}, \quad \nabla^2\phi(x, u) = \begin{bmatrix} \nabla_x^2\phi(x, u) & \nabla_{xu}^2\phi(x, u) \\ \nabla_{ux}^2\phi(x, u) & \nabla_u^2\phi(x, u) \end{bmatrix}. \quad (3.8)$$

The blocks in the Hessian $\nabla^2\phi(x, u)$ are block diagonal and are defined as

$$\nabla_x^2\phi(x, u) \triangleq \begin{bmatrix} \nabla_{x_0}^2\phi(x_0, u_0) & & \\ & \ddots & \\ & & \nabla_{x_N}^2\phi(x_N) \end{bmatrix}, \quad (3.9)$$

$$\nabla_u^2\phi(x, u) \triangleq \begin{bmatrix} \nabla_{u_0}^2\phi(x_0, u_0) & & \\ & \ddots & \\ & & \nabla_{u_{N-1}}^2\phi(x_{N-1}, u_{N-1}) \end{bmatrix}, \quad (3.10)$$

$$\nabla_{xu}^2\phi(x, u) \triangleq \begin{bmatrix} \nabla_{x_0 u_0}^2\phi(x_0, u_0) & & \\ & \ddots & \\ & & \nabla_{x_{N-1} u_{N-1}}^2\phi(x_{N-1}, u_{N-1}) \\ 0 & \dots & 0 \end{bmatrix}, \quad (3.11)$$

$$\nabla_{ux}^2\phi(x, u) \triangleq \left(\nabla_{xu}^2\phi(x, u) \right)^T. \quad (3.12)$$

Here the notation $\nabla_x^2\phi(x, u)$, $\nabla_u^2\phi(x, u)$ and $\nabla_{xu}^2\phi(x, u)$ denotes the parts in the block partitioning of the Hessian $\nabla^2\phi(x, u)$, i.e.,

$$\nabla^2\phi(x, u) = \begin{bmatrix} \nabla_x^2\phi(x, u) & \nabla_{xu}^2\phi(x, u) \\ \nabla_{ux}^2\phi(x, u) & \nabla_u^2\phi(x, u) \end{bmatrix}. \quad (3.13)$$

Hence, $\nabla\phi(\bar{x}^p, \bar{u}^p)$ and $\nabla^2\phi(\bar{x}^p, \bar{u}^p)$ are computed by inserting $\bar{x}^p$ and $\bar{u}^p$ into (3.8).

The equality constraints in the problem (2.41) can be formed by inserting the point $\bar{x}^p + \Delta x$ and $\bar{u}^p + \Delta u$ into the dynamics equations in (3.5), i.e.,

$$\begin{aligned} A^p(\bar{x}^p + \Delta x) + B^p(\bar{u}^p + \Delta u) + a^p &= 0 \iff \\ A^p\bar{x}^p + B^p\bar{u}^p + a^p + A^p\Delta x + B^p\Delta u &= 0 \iff A^p\Delta x + B^p\Delta u = 0. \end{aligned} \quad (3.14)$$

The last step holds since $\bar{x}^p$ and $\bar{u}^p$ are primal feasible and hence satisfy

$$A^p\bar{x}^p + B^p\bar{u}^p + a^p = 0. \quad (3.15)$$

By using the expressions for the gradients and Hessians in (3.6) and (3.8), the objective function of the approximated optimization problem (2.41) can be written

$$\begin{aligned} &\frac{1}{2} \begin{bmatrix} \Delta x \\ \Delta u \end{bmatrix}^T \left(s \begin{bmatrix} Q_x^p & Q_{xu}^p \\ (Q_{xu}^p)^T & Q_u^p \end{bmatrix} + \begin{bmatrix} \nabla_x^2 \phi(\bar{x}^p, \bar{u}^p) & \nabla_{xu}^2 \phi(\bar{x}^p, \bar{u}^p) \\ \nabla_{ux}^2 \phi(\bar{x}^p, \bar{u}^p) & \nabla_u^2 \phi(\bar{x}^p, \bar{u}^p) \end{bmatrix} \right) \begin{bmatrix} \Delta x \\ \Delta u \end{bmatrix} + \\ &\left(s \begin{bmatrix} Q_x^p & Q_{xu}^p \\ (Q_{xu}^p)^T & Q_u^p \end{bmatrix} \begin{bmatrix} \bar{x}^p \\ \bar{u}^p \end{bmatrix} + s \begin{bmatrix} l_x^p \\ l_u^p \end{bmatrix} + \begin{bmatrix} \nabla_x \phi(\bar{x}^p, \bar{u}^p) \\ \nabla_u \phi(\bar{x}^p, \bar{u}^p) \end{bmatrix} \right)^T \begin{bmatrix} \Delta x \\ \Delta u \end{bmatrix} + s f_0(\bar{x}^p, \bar{u}^p) + \phi(\bar{x}^p, \bar{u}^p). \end{aligned} \quad (3.16)$$

If the structure in the matrices in (3.16) is considered, it is clear that the quadratic part of the objective function is sparse with the same sparsity pattern as for the quadratic part in the CFTOC problem (3.5). Also the linear term has similar structure as in (3.5). Hence, by defining the following variables

$$Q_{x,t} \triangleq sQ_{x,t}^p + \nabla_{x_t}^2 \phi(\bar{x}_t^p, \bar{u}_t^p), \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.17)$$

$$Q_{x,N} \triangleq sQ_{x,N}^p + \nabla_{x_N}^2 \phi(\bar{x}_N^p), \quad (3.18)$$

$$Q_{u,t} \triangleq sQ_{u,t}^p + \nabla_{u_t}^2 \phi(\bar{x}_t^p, \bar{u}_t^p), \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.19)$$

$$Q_{xu,t} \triangleq sQ_{xu,t}^p + \nabla_{x_t u_t}^2 \phi(\bar{x}_t^p, \bar{u}_t^p), \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.20)$$

$$l_{x,t} \triangleq sQ_{x,t}^p \bar{x}_t^p + sQ_{xu,t}^p \bar{u}_t^p + sl_{x,t}^p + \nabla_{x_t} \phi(\bar{x}_t^p, \bar{u}_t^p), \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.21)$$

$$l_{x,N} \triangleq sQ_{x,N}^p \bar{x}_N^p + sl_{x,N}^p + \nabla_{x_N} \phi(\bar{x}_N^p), \quad (3.22)$$

$$l_{u,t} \triangleq sQ_{u,t}^p \bar{u}_t^p + s(Q_{xu,t}^p)^T \bar{x}_t^p + sl_{u,t}^p + \nabla_{u_t} \phi(\bar{x}_t^p, \bar{u}_t^p), \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.23)$$

$$c_t \triangleq s \begin{bmatrix} \bar{x}_t^p \\ \bar{u}_t^p \end{bmatrix}^T \begin{bmatrix} Q_{x,t}^p & Q_{xu,t}^p \\ (Q_{xu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} \bar{x}_t^p \\ \bar{u}_t^p \end{bmatrix} + s \begin{bmatrix} l_{x,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} \bar{x}_t^p \\ \bar{u}_t^p \end{bmatrix} + sc_t^p + \phi(\bar{x}_t^p, \bar{u}_t^p), \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.24)$$

$$c_N \triangleq s\bar{x}_N^p{}^T Q_{x,N}^p \bar{x}_N^p + sl_{x,N}^p{}^T \bar{x}_N^p + sc_N^p + \phi(\bar{x}_N^p), \quad (3.25)$$

and by using the fact that the dynamics equations (3.14) are equivalent to

$$\Delta x_0 = 0, \quad (3.26)$$

$$\Delta x_{t+1} = A_t^p \Delta x_t + B_t^p \Delta u_t, \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.27)$$

it is seen that the centering step (2.31) when solving the CFTOC problem (3.1) can be solved by computing a sequence of Newton step search directions correspond-

ing to problems in the form of UFTOC problems, given by

$$\begin{aligned}
 & \underset{x,u}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} \Delta x_t \\ \Delta u_t \end{bmatrix}^T \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} \begin{bmatrix} \Delta x_t \\ \Delta u_t \end{bmatrix} + \begin{bmatrix} l_{x,t} \\ l_{u,t} \end{bmatrix}^T \begin{bmatrix} \Delta x_t \\ \Delta u_t \end{bmatrix} + c_t \right) + \\
 & && \frac{1}{2} \Delta x_N^T Q_{x,N} \Delta x_N + l_{x,N}^T \Delta x_N + c_N \\
 & \text{subject to} && \Delta x_0 = 0 \\
 & && \Delta x_{t+1} = A_t^p \Delta x_t + B_t^p \Delta u_t, \quad t \in \mathbb{Z}_{0,N-1}.
 \end{aligned} \tag{3.28}$$

Remark 3.3. In this section the centering step and the corresponding Newton steps for a basic barrier method was presented. However, similar systems of equations also arise when using primal-dual IP methods (Axehill et al., 2007). Hence, the main computational effort in most IP solvers is spent on solving sequences of Newton steps in the form of UFTOC problems as in (3.28).

3.2.2 Active-set methods

In this section it will be shown how the search directions that are required by an AS solver, like for example the one in Algorithm 1, can be computed when solving a CFTOC problem in the form in (3.1). Here, CFTOC problems with upper and lower bounds on the control inputs will be considered. Hence, the CFTOC problem is in the form

$$\begin{aligned}
 & \underset{x^p, u^p}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{x,t}^p & Q_{xu,t}^p \\ (Q_{xu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} x_t^p \\ u_t^p \end{bmatrix} + \begin{bmatrix} l_{x,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} x_t^p \\ u_t^p \end{bmatrix} + c_t^p \right) + \\
 & && \frac{1}{2} x_N^p{}^T Q_{x,N}^p x_N^p + l_{x,N}^p{}^T x_N^p + c_N^p \\
 & \text{subject to} && x_0^p = \bar{x} \\
 & && x_{t+1}^p = A_t^p x_t^p + B_t^p u_t^p + a_t^p, \quad t \in \mathbb{Z}_{0,N-1} \\
 & && u_{\min,t}^p \leq u_t^p \leq u_{\max,t}^p, \quad t \in \mathbb{Z}_{0,N-1}.
 \end{aligned} \tag{3.29}$$

Assumptions 3.1 and 3.2 still hold for (3.29).

Let the working set $\mathcal{W}_j$ and its complement $\mathcal{W}_j^c$ at AS iteration j be introduced as in Section 2.6. In the problem (3.29) only control input constraints are used, and hence the working set describes the set of control inputs that are temporarily fixed to a constant value (either the maximum or minimum limit). Adding a constraint to the working set, i.e., forcing a constraint to hold with equality, thus corresponds to removing that control input as an optimization variable from the optimization problem (3.29). Similarly, by disregarding an inequality constraint the corresponding control input becomes unconstrained and is introduced as an optimization variable in the optimization problem. This can be formalized by introducing w_t^p as the free part of the control inputs and v_t^p as the fixed part, i.e.,

$$w_t^p \triangleq u_t^p(\mathcal{W}_j^c), \quad v_t^p \triangleq u_t^p(\mathcal{W}_j), \quad u_t^p = \Pi \begin{bmatrix} w_t^p \\ v_t^p \end{bmatrix}, \tag{3.30}$$

where Π is a permutation matrix satisfying $\Pi^T \Pi = I$. Here the notation $u_t^p(\mathcal{W}_j)$ is used to denote the elements in u_t^p that are affected by constraints with indices in $\mathcal{W}_j$. Using this notation, B_t^p , $Q_{u,t}^p$, $Q_{xu,t}^p$ and $l_{u,t}^p$ can be partitioned into parts corresponding to w_t^p and v_t^p , respectively, giving

$$B_t^p = \begin{bmatrix} B_{w,t}^p & B_{v,t}^p \end{bmatrix} \Pi^T, \quad Q_{xu,t}^p = \begin{bmatrix} Q_{xw,t}^p & Q_{xv,t}^p \end{bmatrix} \Pi^T, \quad (3.31)$$

$$Q_{u,t}^p = \Pi \begin{bmatrix} Q_{w,t}^p & Q_{wv,t}^p \\ (Q_{xw,t}^p)^T & Q_{v,t}^p \end{bmatrix} \Pi^T, \quad l_{u,t}^p = \begin{bmatrix} l_{w,t}^p \\ l_{v,t}^p \end{bmatrix} \Pi^T. \quad (3.32)$$

The variables v_t^p can be eliminated from the optimization problem by inserting the fixed values given by the corresponding inequality constraints in the working set $\mathcal{W}_j$. Inserting this partitioning of u_t^p and B_t^p into the dynamics constraints in (3.29) gives

$$x_{t+1}^p = A_t^p x_t^p + B_t^p u_t^p + a_t^p = A_t^p x_t^p + B_{w,t}^p w_t^p + \underbrace{B_{v,t}^p v_t^p + a_t^p}_{\triangleq a_{v,t}^p} = A_t^p x_t^p + B_{w,t}^p w_t^p + a_{v,t}^p. \quad (3.33)$$

Similarly, by using the partitioning of the control input and the involved matrices in the objective function in the CFTOC problem (3.29) gives the expression

$$\begin{aligned} & \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t^p \\ w_t^p \end{bmatrix}^T \begin{bmatrix} Q_{x,t}^p & Q_{xw,t}^p \\ (Q_{xw,t}^p)^T & Q_{w,t}^p \end{bmatrix} \begin{bmatrix} x_t^p \\ w_t^p \end{bmatrix} + \begin{bmatrix} l_{x,t}^p \\ l_{w,t}^p \end{bmatrix}^T \begin{bmatrix} x_t^p \\ w_t^p \end{bmatrix} + c_t \right) + \\ & \frac{1}{2} x_N^p{}^T Q_{x,N}^p x_N^p + l_{x,N}^p{}^T x_N^p + c_N^p, \end{aligned} \quad (3.34)$$

where $l_{x,t}$, $l_{w,t}$ and c_t are defined as

$$l_{x,t} \triangleq l_{x,t}^p + Q_{xv,t}^p v_t^p, \quad (3.35)$$

$$l_{w,t} \triangleq l_{w,t}^p + Q_{wv,t}^p v_t^p, \quad (3.36)$$

$$c_t \triangleq c_t^p + \frac{1}{2} v_t^p{}^T Q_{v,t}^p v_t^p + l_{v,t}^p{}^T v_t^p. \quad (3.37)$$

Hence, by using this partition of the control input and the involved matrices, the UFTOC problem that is solved at AS iteration j to compute the Newton step can be written in the form

$$\begin{aligned} & \underset{x^p, w^p}{\text{minimize}} \quad \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t^p \\ w_t^p \end{bmatrix}^T \begin{bmatrix} Q_{x,t}^p & Q_{xw,t}^p \\ (Q_{xw,t}^p)^T & Q_{w,t}^p \end{bmatrix} \begin{bmatrix} x_t^p \\ w_t^p \end{bmatrix} + \begin{bmatrix} l_{x,t}^p \\ l_{w,t}^p \end{bmatrix}^T \begin{bmatrix} x_t^p \\ w_t^p \end{bmatrix} + c_t \right) + \\ & \quad \frac{1}{2} x_N^p{}^T Q_{x,N}^p x_N^p + l_{x,N}^p{}^T x_N^p + c_N^p \\ & \text{subject to} \quad x_0^p = \bar{x} \\ & \quad x_{t+1}^p = A_t^p x_t^p + B_{w,t}^p w_t^p + a_{v,t}^p, \quad t \in \mathbb{Z}_{0,N-1}. \end{aligned} \quad (3.38)$$

3.3 Newton Step Computation

As was indicated in the previous section, the Newton step computation plays a fundamental role when using both AS and IP methods to solve an CFTOC problem in the form (3.1). Both types of methods require a sequence of Newton steps to be computed in order to solve the optimization problem (3.1), and this is where the most computational effort is used (Rao et al., 1998; Vandenberghe et al., 2002; Nocedal and Wright, 2006; Axehill et al., 2007). Hence, efficient numerical algorithms that compute the solution to the corresponding UFTOC problems are crucial to achieve high performance in both AS and IP methods.

The principal form of the optimization problem in both (3.28) and (3.38) can be interpreted as a UFTOC problem in the form

$$\begin{aligned}
 & \underset{x,u}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^T \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \begin{bmatrix} l_{x,t} \\ l_{u,t} \end{bmatrix}^T \begin{bmatrix} x_t \\ u_t \end{bmatrix} + c_t \right) + \\
 & && \frac{1}{2} x_N^T Q_{x,N} x_N + l_{x,N}^T x_N + c_N \\
 & \text{subject to} && x_0 = \bar{x} \\
 & && x_{t+1} = A_t x_t + B_t u_t + a_t, \quad t \in \mathbb{Z}_{0,N-1},
 \end{aligned} \tag{3.39}$$

where $x_t \in \mathbb{R}^{n_x}$ are the states, $u_t \in \mathbb{R}^{n_{u,t}}$ the control inputs, and the equality constraints are the dynamics constraints of a linear dynamical system. Furthermore, let assumptions 3.4 and 3.5 hold

Assumption 3.4. $Q_t \triangleq \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} \in \mathbb{S}_+^{n_x+n_{u,t}}, \quad t \in \mathbb{Z}_{0,N-1}, \quad Q_{x,N} \in \mathbb{S}_+^{n_x}.$

Assumption 3.5. $Q_{u,t} \in \mathbb{S}_{++}^{n_{u,t}}, \quad t \in \mathbb{Z}_{0,N-1}.$

Remark 3.6. For notational brevity the time dependence on the control input dimension is explicitly given only when it is of importance for the results. Hence n_u will be used instead of $n_{u,t}$ in the rest of this thesis unless stated otherwise.

Remark 3.7. As already mentioned, in both IP and AS methods for QP problems the solution to the original CFTOC problem is obtained by solving a sequence of UFTOC problems in the form in (3.39). The number of problems in this sequence is independent of how these UFTOC problems are solved and the main computation time is consumed when solving this sequence of UFTOC problems. Hence, the overall relative performance gain in terms of computation time for solving the entire sequence of problems in order to solve the CFTOC problem is roughly the same as the relative performance gain in terms of computation time obtained when solving a single UFTOC problem.

In Definition 3.8 the notation $\leftrightarrow$ is introduced to indicate the connection between an equality or inequality constraint and the corresponding dual variable in a convex optimization problem. This notation will be used both for scalar and vector valued constraints.

When assumptions 3.4 and 3.5 hold, the KKT coefficient matrix is non-singular and there exists a unique solution to (3.44). There exist a vast number of methods to solve the linear systems of equations in (3.44). The naive way to solve (3.44) would be to compute the inverse of the KKT coefficient matrix, but since the inverse is not explicitly needed, some factorization method is used in practise. A good reference on numerical algorithms for solving linear system of equations is Golub and Van Loan (1996). Using a factorization method that does not exploit the special structure in the KKT coefficient matrix, such as the LDL factorization, to compute the solution to (3.44) requires $\mathcal{O}(N^3)$ complexity. When a sequence of UFTOC problems in the form in (3.39) are solved, the solution to the KKT systems in the form in (3.44) can be solved by updating the factorization instead of re-computing it from scratch, which requires $\mathcal{O}(N^2)$ complexity. However, the KKT coefficient matrix in (3.44) has a special almost block-diagonal structure, and some methods that exploit the sparse structure require only $\mathcal{O}(N)$ operations to solve (3.44). One such method is the Riccati recursion and it will be used in this thesis to exploit the structure in the KKT coefficient matrix to compute the solution efficiently.

Remark 3.9. Algorithms for solving problems similar to the UFTOC problem (3.39) are also important subroutines when solving Moving Horizon Estimation (MHE) problems (Rao et al., 1998; Rao, 2000; Jørgensen, 2004). Hence, the algorithms presented in this thesis for solving UFTOC problems in the form in (3.39) and the corresponding KKT system in (3.44) can be useful also in MHE applications.

3.4 Solving the Karush-Kuhn-Tucker System using the Riccati Recursion

The solution to the UFTOC problem (3.39) is computed by solving the KKT system (3.44). The special structure of the UFTOC problem that is considered in this work corresponds to an almost block diagonal KKT coefficient matrix that can be solved very efficiently using a Riccati recursion (Jonson, 1983; Rao et al., 1998; Vandenberghe et al., 2002; Axehill, 2008). The Riccati recursion factors the KKT coefficient matrix and the factorization is followed by backward and forward substitutions for solving the system of equations. Using the Riccati recursion to compute the solution to (3.39) requires $\mathcal{O}(N)$ computational complexity.

3.4.1 Derivation of the Riccati recursion

In this section the derivation of the Riccati recursion that is used to solve (3.44) is presented. The derivation is similar to what is presented in Axehill (2005), but is here given for a UFTOC problem in the form in (3.39). The results that are presented in chapters 4 and 6 will be based on the Riccati recursion, and some of the details in the derivation of the Riccati recursion will be used in these chapters.

By inspecting the system of equations (3.44) it can be seen that the KKT system consists of N blocks of equations in the form

$$\begin{bmatrix} -I & Q_{x,t} & Q_{xu,t} \\ 0 & Q_{xu,t}^T & Q_{u,t} \\ 0 & A_t & B_t \end{bmatrix} \begin{bmatrix} \lambda_t \\ x_t \\ u_t \end{bmatrix} + \begin{bmatrix} A_t^T & 0 \\ B_t^T & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} \lambda_{t+1} \\ x_{t+1} \end{bmatrix} = \begin{bmatrix} -l_{x,t} \\ -l_{u,t} \\ -a_t \end{bmatrix}, \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.45)$$

and the two boundary equations

$$x_0 = \bar{x}, \quad (3.46)$$

$$-\lambda_N + Q_{x,N} x_N = -l_{x,N}. \quad (3.47)$$

An induction argument will now be used to show that there exists a $P_t \in \mathbb{S}_+^{n_x}$ and $\Psi_t \in \mathbb{R}^{n_x}$ for $t \in \mathbb{Z}_{0,N}$ such that

$$P_t x_t - \lambda_t = \Psi_t. \quad (3.48)$$

Furthermore, it will be shown how the property (3.48) can be used to solve the system of equations (3.44).

First, note that the choice $P_N = Q_{x,N}$ and $\Psi_N = -l_{x,N}$ in (3.48) satisfies (3.47). Now assume that (3.48) holds for some arbitrary $t+1$ with $P_{t+1} \in \mathbb{S}_+^{n_x}$ and $t \in \mathbb{Z}_{0,N-1}$. Inserting $\lambda_{t+1} = P_{t+1} x_{t+1} - \Psi_{t+1}$ into (3.45) gives

$$\begin{bmatrix} -I & Q_{x,t} & Q_{xu,t} \\ 0 & Q_{xu,t}^T & Q_{u,t} \\ 0 & A_t & B_t \end{bmatrix} \begin{bmatrix} \lambda_t \\ x_t \\ u_t \end{bmatrix} + \begin{bmatrix} A_t^T P_{t+1} \\ B_t^T P_{t+1} \\ -I \end{bmatrix} x_{t+1} - \begin{bmatrix} A_t^T \\ B_t^T \\ 0 \end{bmatrix} \Psi_{t+1} = \begin{bmatrix} -l_{x,t} \\ -l_{u,t} \\ -a_t \end{bmatrix}. \quad (3.49)$$

By eliminating the dynamics constraints in the last block equation in (3.49) this system of equations can equivalently be written

$$\begin{bmatrix} -I & Q_{x,t} + A_t^T P_{t+1} A_t & Q_{xu,t} + A_t^T P_{t+1} B_t \\ 0 & Q_{xu,t}^T + B_t^T P_{t+1} A_t & Q_{u,t} + B_t^T P_{t+1} B_t \end{bmatrix} \begin{bmatrix} \lambda_t \\ x_t \\ u_t \end{bmatrix} + \begin{bmatrix} A_t^T \\ B_t^T \end{bmatrix} (P_{t+1} a_t - \Psi_{t+1}) = \begin{bmatrix} -l_{x,t} \\ -l_{u,t} \end{bmatrix}. \quad (3.50)$$

Now introduce the auxiliary variables F_{t+1} , H_{t+1} and G_{t+1} as

$$F_{t+1} \triangleq Q_{x,t} + A_t^T P_{t+1} A_t \in \mathbb{S}_+^{n_x}, \quad (3.51a)$$

$$H_{t+1} \triangleq Q_{xu,t} + A_t^T P_{t+1} B_t, \quad (3.51b)$$

$$G_{t+1} \triangleq Q_{u,t} + B_t^T P_{t+1} B_t \in \mathbb{S}_{++}^{n_u}. \quad (3.51c)$$

Then, the equations in (3.50) can equivalently be written

$$\begin{bmatrix} -I & F_{t+1} & H_{t+1} \\ 0 & H_{t+1}^T & G_{t+1} \end{bmatrix} \begin{bmatrix} \lambda_t \\ x_t \\ u_t \end{bmatrix} + \begin{bmatrix} A_t^T \\ B_t^T \end{bmatrix} (P_{t+1} a_t - \Psi_{t+1}) = \begin{bmatrix} -l_{x,t} \\ -l_{u,t} \end{bmatrix}. \quad (3.52)$$

Since $Q_{u,t} \in \mathbb{S}_{++}^{n_u}$ also $G_{t+1} \in \mathbb{S}_{++}^{n_u}$ and it is possible to compute the control signal u_t from the lower block equation in (3.52) as

$$u_t = -G_{t+1}^{-1} H_{t+1}^T x_t - G_{t+1}^{-1} (l_{u,t} + B_t^T P_{t+1} a_t - B_t^T \Psi_{t+1}), \quad (3.53)$$

which inserted in the upper block equation in (3.52) gives

$$\begin{aligned} & -\lambda_t + F_{t+1}x_t - H_{t+1}G_{t+1}^{-1}H_{t+1}^T x_t - H_{t+1}G_{t+1}^{-1}(l_{u,t} + B_t^T P_{t+1}a_t - B_t^T \Psi_{t+1}) + \\ & A_t^T P_{t+1}a_t - A_t^T \Psi_{t+1} = -l_{x,t} \iff -\lambda_t + (F_{t+1} - H_{t+1}G_{t+1}^{-1}H_{t+1}^T)x_t = \\ & -l_{x,t} + H_{t+1}G_{t+1}^{-1}(l_{u,t} + B_t^T P_{t+1}a_t - B_t^T \Psi_{t+1}) - A_t^T P_{t+1}a_t + A_t^T \Psi_{t+1}. \end{aligned} \quad (3.54)$$

By identifying the terms in (3.54) with the ones in (3.48) it is clear that (3.54) can be written

$$-\lambda_t + P_t x_t = \Psi_t, \quad (3.55)$$

with

$$P_t \triangleq F_{t+1} - H_{t+1}G_{t+1}^{-1}H_{t+1}^T, \quad (3.56)$$

$$\Psi_t \triangleq -l_{x,t} + (A_t^T - H_{t+1}G_{t+1}^{-1}B_t^T)\Psi_{t+1} + H_{t+1}G_{t+1}^{-1}(l_{u,t} + B_t^T P_{t+1}a_t) - A_t^T P_{t+1}a_t. \quad (3.57)$$

Furthermore, since

$$M_{t+1} \triangleq \begin{bmatrix} F_{t+1} & H_{t+1} \\ H_{t+1}^T & G_{t+1} \end{bmatrix} = \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} + \begin{bmatrix} A_t^T \\ B_t^T \end{bmatrix} P_{t+1} \begin{bmatrix} A_t & B_t \end{bmatrix} \in \mathbb{S}_+^{n_x+n_{u,t}}, \quad (3.58)$$

is positive semidefinite by construction, it follows from Theorem A.8 that the Schur complement of G_{t+1} in M_{t+1} (see Definition A.4) is also symmetric positive semidefinite, and P_t thus satisfies

$$P_t = M_{t+1}/G_{t+1} = F_{t+1} - H_{t+1}G_{t+1}^{-1}H_{t+1}^T \in \mathbb{S}_+^{n_x}. \quad (3.59)$$

Hence, if there exists a $P_{t+1} \in \mathbb{S}_+^{n_x}$ and $\Psi_{t+1} \in \mathbb{R}^{n_x}$ such $P_{t+1}x_{t+1} - \lambda_{t+1} = \Psi_{t+1}$ for some $t \in \mathbb{Z}_{0,N-1}$, there also exists a $P_t \in \mathbb{S}_+^{n_x}$ and $\Psi_t \in \mathbb{R}^{n_x}$ such that $P_t x_t - \lambda_t = \Psi_t$. This holds for an arbitrary $t \in \mathbb{Z}_{0,N-1}$.

Remark 3.10. The matrix P_t in (3.56) is unique, which can be derived from the uniqueness of F_{t+1} , H_{t+1} and G_{t+1} . Furthermore it can be seen, by using the definitions of F_{t+1} , H_{t+1} and G_{t+1} in (3.51a), (3.51b) and (3.51c), respectively, that (3.56) is the discrete-time Riccati recursion

$$P_t = Q_{x,t} + A_t^T P_{t+1} A_t - (Q_{xu,t} + A_t^T P_{t+1} B_t)(Q_{u,t} + B_t^T P_{t+1} B_t)^{-1}(Q_{xu,t}^T + B_t^T P_{t+1} A_t). \quad (3.60)$$

By defining the auxiliary variables K_{t+1} and k_{t+1} as the solutions to the systems of equations

$$G_{t+1}K_{t+1} \triangleq -H_{t+1}^T, \quad (3.61)$$

$$G_{t+1}k_{t+1} \triangleq -(l_{u,t} + B_t^T P_{t+1}a_t - B_t^T \Psi_{t+1}), \quad (3.62)$$

P_t can be expressed as

$$P_t = F_{t+1} - H_{t+1}G_{t+1}^{-1}H_{t+1}^T = F_{t+1} - K_{t+1}^T G_{t+1} K_{t+1}. \quad (3.63)$$

Hence, when P_t , Ψ_t , K_{t+1} and k_{t+1} have been computed using (3.56), (3.57), (3.61) and (3.62), respectively, the solution to the KKT system (3.44) is given by

$$x_0 = \bar{x}, \quad (3.64a)$$

$$u_t = k_{t+1} + K_{t+1}x_t, \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.64b)$$

$$x_{t+1} = A_t x_t + B_t u_t + a_t, \quad t \in \mathbb{Z}_{0,N-1}, \quad (3.64c)$$

$$\lambda_t = P_t x_t - \Psi_t, \quad t \in \mathbb{Z}_{0,N}. \quad (3.64d)$$

This is summarized in algorithms 4, 5 and 6. In Algorithm 4 the KKT coefficient matrix is factored using the Riccati recursion. Algorithm 5 is a backward recursion where the variables k_{t+1} and Ψ_t are computed. Finally, to compute the solution, the forward recursion in Algorithm 6 is used. At Line 6 in Algorithm 4 G_{t+1} is factored using a suitable factorization like for example the Cholesky factorization. This factorization is used to solve the system of equations at Line 7 in Algorithm 4 and at Line 3 in Algorithm 5. Hence, the matrix G_{t+1} is never explicitly inverted and the factorization is reused during several operations. In Algorithm 5 the backward recursion to compute the constant term in the objective function when using the control law (3.64b) is included.

Algorithm 4 Factorization (Riccati recursion)

```

1:  $P_N := Q_{x,N}$ 
2: for  $t = N - 1, \dots, 0$  do
3:    $F_{t+1} := Q_{x,t} + A_t^T P_{t+1} A_t$ 
4:    $G_{t+1} := Q_{u,t} + B_t^T P_{t+1} B_t$ 
5:    $H_{t+1} := Q_{xu,t} + A_t^T P_{t+1} B_t$ 
6:   Compute and store a factorization of  $G_{t+1}$ 
7:   Compute a solution  $K_{t+1}$  to
      $G_{t+1} K_{t+1} = -H_{t+1}^T$ 
8:    $P_t := F_{t+1} - K_{t+1}^T G_{t+1} K_{t+1}$ 
9: end for
```

Algorithm 5 Backward recursion

```

1:  $\Psi_N := -l_{x,N}$ ,  $\bar{c}_N := c_N$ 
2: for  $t = N - 1, \dots, 0$  do
3:   Compute a solution  $k_{t+1}$  to
      $G_{t+1} k_{t+1} = B_t^T \Psi_{t+1} - l_{u,t} - B_t^T P_{t+1} a_t$ 
4:    $\Psi_t := A_t^T \Psi_{t+1} - H_{t+1} k_{t+1} - l_{x,t} - A_t^T P_{t+1} a_t$ 
5:    $\bar{c}_t := \bar{c}_{t+1} + c_t + \frac{1}{2} a_t^T P_{t+1} a_t - \Psi_{t+1}^T a_t - \frac{1}{2} k_{t+1}^T G_{t+1} k_{t+1}$ 
6: end for
```

3.4.2 Handling singular Karush-Kuhn-Tucker systems

Some methods for solving CFTOC problems require the solution of a UFTOC problem in the form in (3.39), but where Assumption 3.5 *does not* hold. Examples of such methods are dual AS methods and dual gradient projection methods as

Algorithm 6 Forward recursion

```

1:  $x_0 := \bar{x}$ 
2: for  $t = 0, \dots, N - 1$  do
3:    $u_t := k_{t+1} + K_{t+1}x_t$ 
4:    $x_{t+1} := A_t x_t + B_t u_t + a_t$ 
5:    $\lambda_t := P_t x_t - \Psi_t$ 
6: end for
7:  $\lambda_N := P_N x_N - \Psi_N$ 

```

the one presented in Axehill (2008); Axehill and Hansson (2008). In such problems the matrices G_{t+1} in the Riccati recursion might be singular, and hence the feedback gain matrices K_{t+1} might be non-unique. However, P_t is still unique, see Axehill (2008). If G_{t+1} is singular, then there might not exist a solution to the system of equations at Line 7 in Algorithm 4, *i.e.*,

$$G_{t+1}K_{t+1} = -H_{t+1}^T, \quad (3.65)$$

or if a solution exists it is not unique. However, Theorem A.8 states that the system of equations in (3.65) is solvable since

$$M_{t+1} = \begin{bmatrix} F_{t+1} & H_{t+1} \\ H_{t+1}^T & G_{t+1} \end{bmatrix} \in \mathbb{S}_+^{n_x + n_{u_t}}, \quad (3.66)$$

and hence $H_{t+1}^T \in \mathcal{R}(G_{t+1})$. Furthermore, by using the basic property (A.1) of the pseudo-inverse of G_{t+1} , *i.e.*,

$$G_{t+1} = G_{t+1} G_{t+1}^\dagger G_{t+1}, \quad (3.67)$$

together with (3.61) and (3.63) it can be seen that P_t can be written as

$$P_t = F_{t+1} - K_{t+1}^T G_{t+1} K_{t+1} = F_{t+1} - K_{t+1}^T G_{t+1} G_{t+1}^\dagger G_{t+1} K_{t+1} = F_{t+1} - H_{t+1} G_{t+1}^\dagger H_{t+1}^T. \quad (3.68)$$

The expression in (3.68) can be recognized as the generalized Schur complement of G_{t+1} in M_{t+1} (see Definition A.3), *i.e.*,

$$P_t = M_{t+1} / {}^t G_{t+1} \in \mathbb{S}_+^{n_x}, \quad (3.69)$$

where Theorem A.8 has been used to guarantee positive semidefiniteness of P_t . Furthermore, since F_{t+1} , H_{t+1} and $G_{t+1}^\dagger$ are unique, it follows that also P_t is unique. Hence, P_t is unique although G_{t+1} is singular and K_{t+1} is non-unique.

3.5 Computing Eliminated Dual Variables

In Section 3.2.2 it was shown how the Newton step can be computed when using an AS method to solve the CFTOC problem (3.29). The UFTOC problem that is solved as a subproblem in the AS method is obtained by eliminating the variables v_t^p that are constrained to fixed values by the inequality constraints in the working set. In an AS method the dual variables corresponding to the inequality

constraints play an important role, see Section 2.6. In this section it will be shown how to compute these when computing the Newton step as in Section 3.2.2. For notational brevity, the superscript “ p ” is omitted.

The inequality constraints $u_{\min,t} \leq u_t \leq u_{\max,t}$ in (3.29) can equivalently be written

$$\begin{bmatrix} I \\ -I \end{bmatrix} u_t \leq \begin{bmatrix} u_{\max,t} \\ -u_{\min,t} \end{bmatrix}. \quad (3.70)$$

Let the dual variables $\mu_t \in \mathbb{R}^{2n_u}$ corresponding to these constraints be introduced as

$$\mu_t \leftrightarrow \begin{bmatrix} I \\ -I \end{bmatrix} u_t - \begin{bmatrix} u_{\max,t} \\ -u_{\min,t} \end{bmatrix} \leq 0. \quad (3.71)$$

Now, using a similar partition and notation as in Section 3.2.2, let $\mu_{v,t}$ be the dual variables corresponding to the constraints in the working set $\mathcal{W}_j$ and let $\mu_{w,t}$ be the dual variables corresponding to the constraints that are temporarily disregarded at AS iteration j . The dual variables $\mu_{w,t}$ are defined to be zero, i.e., $\mu_{w,t} \triangleq 0$ (Nocedal and Wright, 2006). In order to compute $\mu_{v,t}$ the KKT system for the equality constrained QP problem (3.38) before eliminating the fixed v_t is studied. Let $\bar{v}_t$ denote the fixed values of v_t , i.e., $v_t = \bar{v}_t$.

Using the partition of the control input in Section 3.2.2 and the definition of μ_t and $\bar{v}_t$ in this section, it can be seen that a single block of equations in the KKT system for the equality constrained QP problem has the structure

$$\begin{bmatrix} \ddots & & & & & & & & \\ -I & Q_{x,t} & Q_{xw,t} & Q_{xv,t} & 0 & A_t^T & 0 & \dots & \\ 0 & Q_{xw,t}^T & Q_{w,t} & Q_{wv,t} & 0 & B_{w,t}^T & 0 & & \\ 0 & Q_{xv,t}^T & Q_{wv,t}^T & Q_{v,t} & -I & B_{v,t}^T & 0 & & \\ 0 & 0 & 0 & -I & 0 & 0 & 0 & & \\ 0 & A_t & B_{w,t} & B_{v,t} & 0 & 0 & -I & & \\ & & & & -I & Q_{x,t+1} & & & \\ & & & & & & \ddots & & \end{bmatrix} \begin{bmatrix} \vdots \\ \lambda_t \\ x_t \\ w_t \\ v_t \\ \mu_{v,t} \\ \lambda_{t+1} \\ x_{t+1} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ -l_{x,t} \\ -l_{w,t} \\ -l_{v,t} \\ -\bar{v}_t \\ -a_t \\ \vdots \end{bmatrix}. \quad (3.72)$$

The fourth block row in the KKT system (3.72) was used in Section 3.2.2 to eliminate the fixed part of the control signal. From the third block row in (3.72) it follows that $\mu_{v,t}$ can be computed from

$$-\mu_{v,t} + Q_{xv,t}^T x_t + Q_{wv,t}^T w_t + Q_{v,t} v_t + B_{v,t}^T \lambda_{t+1} = -l_{v,t}, \quad (3.73)$$

which is summarized in Algorithm 7. Note that by eliminating v_t and $\mu_{v,t}$ from (3.72) it reduces to the KKT system for the UFTOC problem (3.38).

Algorithm 7 Forward recursion - eliminated dual variables

```

1: for  $t = 0, \dots, N - 1$  do
2:    $\mu_{w,t} := 0$ 
3:    $\mu_{v,t} := l_{v,t} + Q_{xv,t}^T x_t + Q_{wv,t}^T w_t + Q_{v,t} v_t + B_{v,t}^T \lambda_{t+1}$ 
4: end for

```

Appendix

In this appendix the matrices, vectors and constants used in problem (3.5) and in (3.43) are defined.

3.A Model Predictive Control Formulations

The matrices and vectors of stacked elements in the problem (3.5) are defined as

$$Q_x^p \triangleq \begin{bmatrix} Q_{x,0}^p & & \\ & \ddots & \\ & & Q_{x,N}^p \end{bmatrix}, \quad Q_{xu}^p \triangleq \begin{bmatrix} Q_{xu,0}^p & & \\ & \ddots & \\ 0 & \dots & Q_{xu,N-1}^p \\ & & & 0 \end{bmatrix}, \quad Q_u^p \triangleq \begin{bmatrix} Q_{u,0}^p & & \\ & \ddots & \\ & & Q_{u,N-1}^p \end{bmatrix}, \quad (3.74)$$

$$A^p \triangleq \begin{bmatrix} -I & & & \\ A_0^p & -I & & \\ & A_1^p & -I & \\ & & \ddots & \\ & & & A_{N-1}^p & -I \end{bmatrix}, \quad B^p \triangleq \begin{bmatrix} 0 & \dots & 0 \\ B_0^p & & \\ & B_1^p & \\ & & \ddots \\ & & & B_{N-1}^p \end{bmatrix}, \quad a^p \triangleq \begin{bmatrix} \bar{x}^p \\ a_0^p \\ a_1^p \\ \vdots \\ a_{N-1}^p \end{bmatrix}, \quad (3.75)$$

$$H_x \triangleq \begin{bmatrix} H_{x,0} & & \\ & \ddots & \\ & & H_{x,N} \end{bmatrix}, \quad H_u \triangleq \begin{bmatrix} H_{u,0} & & \\ & \ddots & \\ 0 & \dots & H_{u,N-1} \\ & & & 0 \end{bmatrix}, \quad h \triangleq \begin{bmatrix} h_0 \\ \vdots \\ h_N \end{bmatrix}, \quad (3.76)$$

$$l_x^p \triangleq \begin{bmatrix} l_{x,0}^p \\ \vdots \\ l_{x,N}^p \end{bmatrix}, \quad l_u^p \triangleq \begin{bmatrix} l_{u,0}^p \\ \vdots \\ l_{u,N-1}^p \end{bmatrix}, \quad x^p \triangleq \begin{bmatrix} x_0^p \\ \vdots \\ x_N^p \end{bmatrix}, \quad u^p \triangleq \begin{bmatrix} u_0^p \\ \vdots \\ u_{N-1}^p \end{bmatrix}, \quad c^p \triangleq \sum_{t=0}^N c_t^p. \quad (3.77)$$

For the UTOC problem (3.39) the matrices and vectors in (3.43) are defined analogously, but dropping the superscript "p".

4

Low-rank Modifications of Riccati Factorizations

In Chapter 3 the MPC problem and its corresponding CFTOC problem were introduced, and it was argued that this CFTOC problem is solved by computing the solutions to a sequence of UFTOC problems. The UFTOC problems can be solved using for example Riccati recursions, and some examples of methods for speeding up the optimization using this approach are found in, e.g., Jonson (1983); Rao et al. (1998); Hansson (2000); Åkerblad and Hansson (2004); Vandenberghe et al. (2002); Axehill and Hansson (2006); Axehill et al. (2010); Axehill (2008); Axehill and Hansson (2008); Diehl et al. (2009); Nielsen et al. (2013). In this chapter it will be shown how to exploit low-rank modifications of the Riccati recursion when solving the UFTOC problems in an AS method.

The use of the Riccati recursion in AS methods was introduced in Jonson (1983). In this reference a Riccati recursion is used to factor one part of the KKT matrix. For the other part standard low-rank modifications of factorizations are used on a dense system of equations of the size of the number of active inequality constraints. The computational complexity of this algorithm grows quadratically in the number of active inequality constraints. An alternative sparse non-Riccati factorization is used in Kirches et al. (2011), where the factorization is updated after changes in the AS iterations. In Nielsen et al. (2013), an algorithm for performing low-rank modifications of the Riccati factorization by exploiting the structured changes in an AS iteration is presented. In this reference CFTOC problems with non-singular input signal weight matrix and simple upper and lower bounds on the control signal are studied.

In AS methods it is often crucial to be able to update the factorization of the KKT matrix instead of re-factorizing it, and for this reason the Riccati factorization has been considered more useful in IP methods than in AS methods (Kirches et al.,

2010). However, recently it was shown in Nielsen et al. (2013) that it is also possible to update the Riccati factorization when it is used in an AS method. In this chapter, this approach is extended to cover more general CFTOC problems, where the corresponding UFTOC problem in the AS iterations might have a singular KKT matrix. This generalization can be important when using for example dual AS solvers similar to the one used in Axehill (2008). Furthermore, in Nielsen et al. (2013) only upper and lower bounds on the control inputs are considered. Here it will be shown how problems with both state and control signal constraints can be handled and how low-rank modifications of the Riccati factorization can improve performance for such problems.

4.1 Problem Formulation

For now, consider a CFTOC problem in the form (3.29) where Assumption 3.2 does not hold, i.e., a problem where the inequality constraints are given by upper and lower bounds on the control inputs and that might have a singular quadratic weight on the control inputs in the objective function. Note that this is a more general problem description than was made in Section 3.2.2. As was shown in Section 3.2.2 the computation of the search direction in an AS methods corresponds to solving a UFTOC problem in the form (3.38). This chapter will focus on how to solve this problem and how to exploit the structure in the problem after changing the working set in an AS solver. The partitioning of the control input u_t^p in (3.29) is done as in (3.30). For notational brevity, the superscript p in (3.38) is however omitted. It then follows that the UFTOC problem is given by

$$\begin{aligned}
 & \underset{x,w}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t \\ w_t \end{bmatrix}^T \begin{bmatrix} Q_{x,t} & Q_{xw,t} \\ Q_{xw,t}^T & Q_{w,t} \end{bmatrix} \begin{bmatrix} x_t \\ w_t \end{bmatrix} + \begin{bmatrix} l_{x,t} \\ l_{w,t} \end{bmatrix}^T \begin{bmatrix} x_t \\ w_t \end{bmatrix} + c_t \right) + \\
 & && \frac{1}{2} x_N^T Q_{x,N} x_N + l_{x,N} x_N + c_N \\
 & \text{subject to} && x_0 = \bar{x} \\
 & && x_{t+1} = A_t x_t + B_{w,t} w_t + a_{v,t}, \quad t \in \mathbb{Z}_{0,N-1},
 \end{aligned} \tag{4.1}$$

where the problem matrices, vectors and constants are defined as in Section 3.2.2.

4.2 Low-rank Modification of the Riccati Factorization

A standard way to improve the performance of an AS solver is by modifying the factorization of the KKT coefficient matrix instead of re-computing it between AS iterations (Nocedal and Wright, 2006). However, it has not previously been done while simultaneously exploiting the problem structure using Riccati recursions in such a way that the computational complexity grows linearly in the prediction horizon and number of active constraints. In this section it will be shown how to modify the Riccati factorization efficiently for the general case when $Q_{w,t} \in \mathbb{S}_+^{n_w}$ in the UFTOC problem (4.1).

It was shown in Section 3.4.2 that P_t in Line 8 in Algorithm 4 can be calculated as

$$P_t = M_{t+1}^\dagger G_{t+1} = F_{t+1} - H_{t+1} G_{t+1}^\dagger H_{t+1}^T = F_{t+1} - K_{t+1}^T G_{t+1} K_{t+1}, \quad (4.2)$$

when G_{t+1} is positive semidefinite and with M_{t+1} defined as in (3.58). The relation (4.2) is a key property when deriving the expressions for the low-rank modifications in this chapter.

Furthermore, the quotient formula for generalized Schur complements will be used in the derivations of the low-rank modifications of the Riccati factorization and is given in Lemma 4.1.

Lemma 4.1. *Let the positive semidefinite matrices $\mathcal{M} \geq 0$ and $\tilde{\mathcal{M}} \geq 0$ be partitioned into blocks as*

$$\mathcal{M} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12}^T & M_{22} & M_{23} \\ M_{13}^T & M_{23}^T & M_{33} \end{bmatrix}, \quad \tilde{\mathcal{M}} = \begin{bmatrix} M_{22} & M_{23} \\ M_{23}^T & M_{33} \end{bmatrix}. \quad (4.3)$$

Then

$$\begin{aligned} (\mathcal{M}^\dagger M_{33})^\dagger (\tilde{\mathcal{M}}^\dagger M_{33}) &= \mathcal{M}^\dagger \tilde{\mathcal{M}} = M_{11} - M_{12} M_{22}^\dagger M_{12}^T - \\ & (M_{13} - M_{12} M_{22}^\dagger M_{23}) (M_{33} - M_{23}^T M_{22}^\dagger M_{23})^\dagger (M_{13}^T - M_{23}^T M_{22}^\dagger M_{12}^T), \end{aligned} \quad (4.4)$$

with

$$M_{13}^T - M_{23}^T M_{22}^\dagger M_{12}^T \in \mathcal{R}(M_{33} - M_{23}^T M_{22}^\dagger M_{23}) \quad \text{and} \quad M_{33} - M_{23}^T M_{22}^\dagger M_{23} \geq 0. \quad (4.5)$$

Proof: The proof of Lemma 4.1 is given in Appendix 4.A. □

For a detailed description of (generalized) Schur complements see, e.g., Carlson et al. (1974); Boyd and Vandenberghe (2004); Zhang (2005). It will now be shown how to perform low-rank modifications of the Riccati factorization after k constraints have been added or removed from the working set of an AS solver. A tilde will be used to denote that a matrix has been modified, i.e., the modified version of X will be denoted $\tilde{X}$.

4.2.1 Removing control input constraints from the working set

Removing k control input constraints from the working set, i.e., temporarily disregarding these constraints that previously were forced to hold, affects the matrices in the factorization given by Algorithm 4 in the same way as adding k new control inputs. Assume without loss of generality that the k released constraints are all affecting the control input at time t and that the new control inputs are appended at the end of w_t . Then the matrices $B_{w,t}$, $Q_{w,t}$ and $Q_{xw,t}$ are modified such that

$$\tilde{B}_{w,t} \triangleq \begin{bmatrix} B_{w,t} & b \end{bmatrix}, \quad \tilde{Q}_{w,t} \triangleq \begin{bmatrix} Q_{w,t} & q_w \\ q_w^T & q_w \end{bmatrix}, \quad \tilde{Q}_{xw,t} \triangleq \begin{bmatrix} Q_{xw,t} & q_{xw} \end{bmatrix}, \quad (4.6)$$

giving $\tilde{B}_{w,t} \in \mathbb{R}^{n_x \times (n_w+k)}$, $\tilde{Q}_{w,t} \in \mathbb{S}_+^{n_w+k}$ and $\tilde{Q}_{xw,t} \in \mathbb{R}^{n_x \times (n_w+k)}$. This implies that the modified $\tilde{G}_{t+1} \in \mathbb{S}_+^{n_w+k}$ and $\tilde{H}_{t+1} \in \mathbb{R}^{n_x \times (n_w+k)}$ are computed as

$$\tilde{G}_{t+1} = \tilde{Q}_{w,t} + \tilde{B}_{w,t}^T P_{t+1} \tilde{B}_{w,t} = \begin{bmatrix} G_{t+1} & q_w + B_{w,t}^T P_{t+1} b \\ q_w^T + b^T P_{t+1} B_{w,t} & q_w^0 + b^T P_{t+1} b \end{bmatrix}, \quad (4.7)$$

$$\tilde{H}_{t+1} = \tilde{Q}_{xw,t} + A_t^T P_{t+1} \tilde{B}_{w,t} = \begin{bmatrix} H_{t+1} & q_{xw} + A_t^T P_{t+1} b \end{bmatrix}. \quad (4.8)$$

By defining the variables

$$g \triangleq q_w + B_{w,t}^T P_{t+1} b, \quad (4.9a)$$

$$g^0 \triangleq q_w^0 + b^T P_{t+1} b, \quad (4.9b)$$

$$h \triangleq q_{xw} + A_t^T P_{t+1} b, \quad (4.9c)$$

$\tilde{G}_{t+1}$ and $\tilde{H}_{t+1}$ in (4.7) and (4.8), respectively, can compactly be re-written as

$$\tilde{G}_{t+1} = \begin{bmatrix} G_{t+1} & g \\ g^T & g^0 \end{bmatrix}, \quad (4.10)$$

$$\tilde{H}_{t+1} = \begin{bmatrix} H_{t+1} & h \end{bmatrix}. \quad (4.11)$$

In analogy with (3.58) and (4.2), $\tilde{P}_t$ can be calculated as $\tilde{P}_t = \tilde{M}_{t+1} / {}^\dagger \tilde{G}_{t+1}$, where $\tilde{M}_{t+1}$ is the modified version of M_{t+1} , i.e.,

$$\tilde{M}_{t+1} \triangleq \begin{bmatrix} F_{t+1} & H_{t+1} & h \\ H_{t+1}^T & G_{t+1} & g \\ h^T & g^T & g^0 \end{bmatrix}. \quad (4.12)$$

Using Lemma 4.1 with $M_{11} = F_{t+1}$, $M_{12} = H_{t+1}$, $M_{13} = h$, $M_{22} = G_{t+1}$, $M_{23} = g$ and $M_{33} = g^0$, the modified $\tilde{P}_t$ can be calculated as

$$\tilde{P}_t = \underbrace{F_{t+1} - H_{t+1} G_{t+1}^\dagger H_{t+1}^T}_{P_t} - \underbrace{(h - H_{t+1} G_{t+1}^\dagger g)}_{\triangleq V_t^-} \underbrace{(g^0 - g^T G_{t+1}^\dagger g)}_{\triangleq C_t^-} \underbrace{(h^T - g^T G_{t+1}^\dagger H_{t+1}^T)}_{V_t^{-T}}, \quad (4.13)$$

where it follows from the lemma that $C_t^- \in \mathbb{S}_+^k$ and $V_t^- \in \mathbb{R}^{n_x \times k}$ with $V_t^{-T} \in \mathcal{R}(C_t^-)$. Here $\mathcal{R}(C_t^-)$ denotes the range space of C_t^- . Hence, removing k control input constraints from the working set results in the modification

$$\tilde{P}_t = P_t - V_t^- C_t^{-\dagger} V_t^{-T}, \quad (4.14)$$

which is here denoted a *downdate* of P_t of at most rank- k . The superscript “ $-$ ” is used to denote matrices which are the result after a downdate (removing constraints from the working set).

The modifications of G_{t+1} and H_{t+1} will influence K_{t+1} and the modified version $\tilde{K}_{t+1} \in \mathbb{R}^{(n_w+k) \times n_x}$ can be computed by finding a solution to the system of equations given by

$$\tilde{G}_{t+1} \tilde{K}_{t+1} = -\tilde{H}_{t+1}^T \iff \begin{bmatrix} G_{t+1} & g \\ g^T & g^0 \end{bmatrix} \begin{bmatrix} K^0 \\ k^0 \end{bmatrix} = -\begin{bmatrix} H_{t+1}^T \\ h^T \end{bmatrix}. \quad (4.15)$$

Remark 4.2. The factorization of $\tilde{G}_{t+1}$ is computed by updating the factorization of G_{t+1} . Hence, a solution to (4.15) can be computed without having to re-factorize $\tilde{G}_{t+1}$, which requires less computations than re-solving (4.15) from scratch.

For the case when $\tilde{G}_{t+1} \in \mathbb{S}_{++}^{n_u}$ the result in Theorem A.6 can be used to compute the solution to (4.15). Using the definitions of C_t^- and V_t^- in (4.13), the solution to (4.15) when $\tilde{G}_{t+1}$ is invertible can be calculated as

$$\tilde{K}_{t+1} = \begin{bmatrix} K^0 \\ k^0 \end{bmatrix} = \begin{bmatrix} K_{t+1} + G_{t+1}^{-1} g (C_t^-)^{-1} V_t^{-T} \\ -(C_t^-)^{-1} V_t^{-T} \end{bmatrix}, \quad (4.16)$$

where $K^0 \in \mathbb{R}^{n_w \times n_x}$ is computed as a modified version of K_{t+1} and $k^0 \in \mathbb{R}^{k \times n_x}$ is the feedback for the k added control signals.

Remark 4.3. When removing k constraints from the working set also k components in v_t in (3.30) (dropping the superscript " p ") are removed and are instead included in $\tilde{w}_t$ as described in this section. Hence, the corresponding matrices and vectors $B_{v,t}$, $a_{v,t}$, $Q_{xv,t}$, $Q_{v,t}$, $Q_{wv,t}$, $l_{w,t}$ and $l_{v,t}$ in (3.31), (3.32) and (3.33) are also affected accordingly (i.e. the corresponding k columns b in $B_{v,t}$ are included in $\tilde{B}_{w,t}$ etc.). These changes are only affecting the right-hand-side of the system of equations in the KKT system in (3.44), and hence they are not affecting the matrices in the factorization in Algorithm 4.

4.2.2 Adding control input constraints to the working set

When k control input constraints are added to the working set at AS iteration j , it corresponds to removing these control inputs from the problem. The impact from this modification on P_t is similar to when constraints are removed. Assume, without loss of generality, that the k control input constraints are all affecting the control input at time t and that the control inputs are removed from the k last entries of w_t . The matrices $B_{w,t}$, $Q_{w,t}$ and $Q_{xw,t}$ are then modified as

$$B_{w,t} = \begin{bmatrix} \tilde{B}_{w,t} & b \end{bmatrix}, \quad Q_{w,t} = \begin{bmatrix} \tilde{Q}_{w,t} & q_w \\ q_w^T & q_w^0 \end{bmatrix}, \quad Q_{xw,t} = \begin{bmatrix} \tilde{Q}_{xw,t} & q_{xw} \end{bmatrix}, \quad (4.17)$$

i.e., the modified matrices are the (1,1)-block in the partition in (4.17). Analogously, the modified versions $\tilde{G}_{t+1} \in \mathbb{S}_{+}^{n_w-k}$ and $\tilde{H}_{t+1} \in \mathbb{R}^{n_x \times (n_w-k)}$ are computed as the (1,1)-blocks in G_{t+1} and H_{t+1} , i.e., the partitioning is,

$$G_{t+1} = \begin{bmatrix} \tilde{G}_{t+1} & g \\ g^T & g^0 \end{bmatrix}, \quad H_{t+1} = \begin{bmatrix} \tilde{H}_{t+1} & h \end{bmatrix}. \quad (4.18)$$

According to (4.2), P_t can be written as $P_t = M_{t+1} / {}^t G_{t+1}$, where M_{t+1} is now

$$M_{t+1} = \begin{bmatrix} F_{t+1} & \tilde{H}_{t+1} & h \\ \tilde{H}_{t+1}^T & \tilde{G}_{t+1} & g \\ h^T & g^T & g^0 \end{bmatrix}. \quad (4.19)$$

Hence, by using Lemma 4.1, P_t can be computed by

$$P_t = \underbrace{F_{t+1} - \tilde{H}_{t+1} \tilde{G}_{t+1}^+ \tilde{H}_{t+1}^T}_{\tilde{P}_t} - \underbrace{(h - \tilde{H}_{t+1} \tilde{G}_{t+1}^+ g)}_{\triangleq V_t^+} \underbrace{(g^0 - g^T \tilde{G}_{t+1}^+ g)}_{\triangleq C_t^+} \underbrace{(h^T - g^T \tilde{G}_{t+1}^+ \tilde{H}_{t+1}^T)}_{V_t^{+T}}, \quad (4.20)$$

where it follows from the lemma that $C_t^+ \in \mathbb{S}_+^k$ and $V_t^+ \in \mathbb{R}^{n_x \times k}$ with $V_t^{+T} \in \mathcal{R}(C_t^+)$. The result (4.20) is similar to (4.13), and it is equivalent to

$$\tilde{P}_t = P_t + V_t^+ C_t^{+ \dagger} V_t^{+T}, \quad (4.21)$$

which is here denoted an *update* of P_t of at most rank- k . Here, the superscript “ $+$ ” is used to denote matrices after an update (after adding constraints to the working set).

The modified K_{t+1} can be computed by solving the system of equations

$$\tilde{G}_{t+1} \tilde{K}_{t+1} = -\tilde{H}_{t+1}^T, \quad (4.22)$$

where, as before, the factorization of $\tilde{G}_{t+1}$ is obtained by updating the factorization of G_{t+1} .

For the case when $G_{t+1} \in \mathbb{S}_{++}^{n_w}$ a similar expression to (4.16) can be obtained by solving the system of equations

$$\underbrace{\begin{bmatrix} \tilde{G}_t & g \\ g^T & g^0 \end{bmatrix}}_{K_{t+1}} \begin{bmatrix} K^0 \\ k^0 \end{bmatrix} = - \begin{bmatrix} \tilde{H}_{t+1}^T \\ h^T \end{bmatrix}, \quad (4.23)$$

using the inverse in Theorem A.6, giving

$$\begin{bmatrix} K^0 \\ k^0 \end{bmatrix} = \begin{bmatrix} -\tilde{G}_{t+1}^{-1} \tilde{H}_{t+1}^T + \tilde{G}_{t+1}^{-1} g (C_t^+)^{-1} V_t^{+T} \\ -(C_t^+)^{-1} V_t^{+T} \end{bmatrix}. \quad (4.24)$$

By identifying the changes in the first block row of (4.24) it can be seen that $\tilde{K}_{t+1}$ can be computed as

$$\tilde{K}_{t+1} = K^0 - \tilde{G}_{t+1}^{-1} g (C_t^+)^{-1} V_t^{+T}, \quad (4.25)$$

where K^0 are the first $n_w - k$ rows in K_{t+1} and $\tilde{K}_{t+1} = -\tilde{G}_{t+1}^{-1} \tilde{H}_{t+1}^T$. Hence, $\tilde{K}_{t+1}$ is computed as a modified version of a part of K_{t+1} .

Remark 4.4. When adding k constraints to the working set, also k components in v_t in (3.30) are added (dropping the superscript “ p ”) and they are removed from w_t . Hence, $B_{v,t}$, $a_{v,t}$, $Q_{xv,t}$, $Q_{v,t}$, $Q_{wv,t}$, $l_{w,t}$ and $l_{v,t}$ in (3.31), (3.32) and (3.33) are also affected accordingly (i.e. the corresponding k columns b in $B_{w,t}$ are included in $\tilde{B}_{v,t}$ etc.). These changes are only affecting the right-hand-side of the system of equations in the KKT system in (3.44), and hence they are not affecting the matrices in the factorization in Algorithm 4.

4.2.3 The impact on subsequent time-steps

According to sections 4.2.1 and 4.2.2, adding or removing k control input constraints to/from the working set at time t_m induces a modification of at most rank- k of P_{t_m} . It will now be shown that these modifications will affect the matrices in the Riccati factorization in a structured way also for time-steps $t \in \mathbb{Z}_{0,t_m-1}$. Specifically, it will be shown that the modifications in these subsequent time-steps will have at most rank- k , which is an important property.

By inserting (4.14) in the definition of F_{t+1} , H_{t+1} and G_{t+1} in (3.51a), (3.51b) and (3.51c), respectively, straightforward calculations for the rank- k downdate give

$$\begin{aligned}\tilde{F}_{t+1} &= Q_{x,t} + A_t^T \tilde{P}_{t+1} A_t = Q_{x,t} + A_t^T P_{t+1} A_t - A_t^T V_{t+1}^- C_{t+1}^{-\dagger} V_{t+1}^{-T} A_t^T \iff \\ \tilde{F}_{t+1} &= F_{t+1} - A_t^T V_{t+1}^- C_{t+1}^{-\dagger} V_{t+1}^{-T} A_t,\end{aligned}\quad (4.26)$$

$$\begin{aligned}\tilde{G}_{t+1} &= Q_{w,t} + B_{w,t}^T \tilde{P}_{t+1} B_{w,t} = Q_{w,t} + B_{w,t}^T P_{t+1} B_{w,t} - B_{w,t}^T V_{t+1}^- C_{t+1}^{-\dagger} V_{t+1}^{-T} B_{w,t} \iff \\ \tilde{G}_{t+1} &= G_{t+1} - B_{w,t}^T V_{t+1}^- C_{t+1}^{-\dagger} V_{t+1}^{-T} B_{w,t},\end{aligned}\quad (4.27)$$

$$\begin{aligned}\tilde{H}_{t+1} &= Q_{xw,t} + A_t^T \tilde{P}_{t+1} B_{w,t} = Q_{xw,t} + A_t^T P_{t+1} B_{w,t} - A_t^T V_{t+1}^- C_{t+1}^{-\dagger} V_{t+1}^{-T} B_{w,t} \iff \\ \tilde{H}_{t+1} &= H_{t+1} - A_t^T V_{t+1}^- C_{t+1}^{-\dagger} V_{t+1}^{-T} B_{w,t}.\end{aligned}\quad (4.28)$$

This can be written in matrix form as

$$\tilde{M}_{t+1} = \begin{bmatrix} \tilde{F}_{t+1} & \tilde{H}_{t+1} \\ \tilde{H}_{t+1}^T & \tilde{G}_{t+1} \end{bmatrix} = \begin{bmatrix} F_{t+1} & H_{t+1} \\ H_{t+1}^T & G_{t+1} \end{bmatrix} - \begin{bmatrix} A_t^T V_{t+1}^- \\ B_{w,t}^T V_{t+1}^- \end{bmatrix} C_{t+1}^{-\dagger} \begin{bmatrix} A_t^T V_{t+1}^- \\ B_{w,t}^T V_{t+1}^- \end{bmatrix}^T \in \mathbb{S}_+^{n_x+n_w}, \quad (4.29)$$

which is positive semidefinite by construction. Now, define the auxiliary matrices $\hat{M}_{t+1}$ and $\bar{M}_{t+1}$ as

$$\hat{M}_{t+1} \triangleq \begin{bmatrix} F_{t+1} & H_{t+1} & A_t^T V_{t+1}^- \\ H_{t+1}^T & G_{t+1} & B_{w,t}^T V_{t+1}^- \\ V_{t+1}^{-T} A_t & V_{t+1}^{-T} B_{w,t} & C_{t+1}^- \end{bmatrix}, \quad \bar{M}_{t+1} \triangleq \begin{bmatrix} G_{t+1} & B_{w,t}^T V_{t+1}^- \\ V_{t+1}^{-T} B_{w,t} & C_{t+1}^- \end{bmatrix}. \quad (4.30)$$

Using Theorem A.8 it can be shown that $\hat{M}_{t+1} \in \mathbb{S}_+^{n_x+n_w+k}$ since

$$\tilde{M}_{t+1} = \hat{M}_{t+1} / C_{t+1}^- \in \mathbb{S}_+^{n_x+n_w}, \quad (4.31a)$$

$$C_{t+1}^- \in \mathbb{S}_+^k, \quad (4.31b)$$

$$\begin{bmatrix} V_{t+1}^{-T} A_t & V_{t+1}^{-T} B_{w,t} \end{bmatrix} \in \mathcal{R}(C_{t+1}^-), \quad (4.31c)$$

where (4.31c) follows from the property $V_{t+1}^{-T} \in \mathcal{R}(C_{t+1}^-)$. Since $\hat{M}_{t+1}$ is positive semidefinite it follows that also $\bar{M}_{t+1} \in \mathbb{S}_+^{n_w+k}$. Hence, by using (4.31a) and identifying that $\tilde{G}_{t+1}$ in (4.27) can be computed as the generalized Schur complement of C_{t+1}^- in $\bar{M}_{t+1}$, i.e., $\tilde{G}_{t+1} = \bar{M}_{t+1} / C_{t+1}^-$, the quotient formula in Lemma 4.1 can

be used to compute $\tilde{P}_t$ as

$$\begin{aligned} \tilde{P}_t = \tilde{M}_{t+1} / \tilde{G}_{t+1}^\dagger = \hat{M}_{t+1} / \hat{M}_{t+1}^\dagger &= \underbrace{F_{t+1} - H_{t+1} G_{t+1}^\dagger H_{t+1}^T}_{P_t} - \underbrace{(A_t^T - H_{t+1} G_{t+1}^\dagger B_{w,t}^T) V_{t+1}^-}_{\triangleq V_t^-} \\ &\quad \left(\underbrace{C_{t+1}^- - V_{t+1}^{-T} B_{w,t} G_{t+1}^\dagger B_{w,t}^T V_{t+1}^-}_{\triangleq C_t^-} \right)^\dagger \underbrace{V_{t+1}^{-T} (A_t^T - H_{t+1} G_{t+1}^\dagger B_{w,t}^T)^T}_{V_t^{-T}}, \end{aligned} \quad (4.32)$$

where it follows from the lemma that $C_t^- \in \mathbb{S}_+^k$, $V_t^- \in \mathbb{R}^{n_x \times k}$ and $V_t^{-T} \in \mathcal{R}(C_t^-)$. Thus, a downdate of at most rank- k of P_{t+1} will result in a downdate of most rank- k also in P_t in the form

$$\tilde{P}_t = P_t - V_t^- C_t^{-\dagger} V_t^{-T}. \quad (4.33)$$

Similarly, the effect of an update of at most rank- k of P_{t+1} can be analyzed using analogous calculations as for the downdate. The update of at most rank- k $\tilde{P}_{t+1} = P_{t+1} + V_{t+1}^+ C_{t+1}^{+\dagger} V_{t+1}^{+T}$ results in the modification

$$\tilde{P}_t = P_t + V_t^+ C_t^{+\dagger} V_t^{+T}, \quad (4.34)$$

with

$$C_t^+ \triangleq C_{t+1}^+ - V_{t+1}^{+T} B_{w,t} \tilde{G}_{t+1}^\dagger B_{w,t}^T V_{t+1}^+ \in \mathbb{S}_+^k, \quad (4.35a)$$

$$V_t^+ \triangleq (A_t^T - \tilde{H}_{t+1} \tilde{G}_{t+1}^\dagger B_{w,t}^T) V_{t+1}^+ \in \mathbb{R}^{n_x \times k}, \quad (4.35b)$$

$$V_t^{+T} \in \mathcal{R}(C_{t+1}^+). \quad (4.35c)$$

Hence, also an update of at most rank- k of P_{t+1} results in an update of at most rank- k of P_t . This important property is summarized in Theorem 4.5.

Theorem 4.5. *Consider a modification of at most rank- k of P_{t_m} in Algorithm 4 at a single time instant $t_m \in \mathbb{Z}_{1,N}$ in either of the forms*

$$\begin{cases} \tilde{P}_{t_m} = P_{t_m} - V_{t_m}^- C_{t_m}^{-\dagger} V_{t_m}^{-T} & (\text{downdate}) \\ \tilde{P}_{t_m} = P_{t_m} + V_{t_m}^+ C_{t_m}^{+\dagger} V_{t_m}^{+T} & (\text{update}) \end{cases}, \quad (4.36)$$

where $C_{t_m}^-, C_{t_m}^+ \in \mathbb{S}_+^k$, $V_{t_m}^-, V_{t_m}^+ \in \mathbb{R}^{n_x \times k}$ and $V_{t_m}^{-T} \in \mathcal{R}(C_{t_m}^-)$, $V_{t_m}^{+T} \in \mathcal{R}(C_{t_m}^+)$, respectively. Then it holds for all $t \in \mathbb{Z}_{0,t_m-1}$ that P_t is modified as

$$\begin{cases} \tilde{P}_t = P_t - V_t^- C_t^{-\dagger} V_t^{-T} & (\text{downdate}) \\ \tilde{P}_t = P_t + V_t^+ C_t^{+\dagger} V_t^{+T} & (\text{update}) \end{cases}, \quad (4.37)$$

with $C_t^-, C_t^+ \in \mathbb{S}_+^k$, $V_t^-, V_t^+ \in \mathbb{R}^{n_x \times k}$ and $V_t^{-T} \in \mathcal{R}(C_t^-)$, $V_t^{+T} \in \mathcal{R}(C_t^+)$, respectively.

Proof: From (4.32) and (4.34) it is clear that a modification of at most rank- k of P_{t_m} results in a corresponding modification of at most rank- k of P_{t_m-1} . Further-

more, the derivation for the subsequent iterations giving (4.32) and (4.34) are made for an arbitrary $t \in \mathbb{Z}_{0,t_m-1}$. Hence, the proof follows by induction. $\square$

Remark 4.6. In a classic AS solver like for example the one in Algorithm 1 only one constraint is removed at each iteration, meaning that $k = 1$ and C_t^- and C_t^+ are scalar. $\square$

The modified $\tilde{K}_{t+1}$ can be computed by solving the system of equations

$$\tilde{G}_{t+1} \tilde{K}_{t+1} = -\tilde{H}_{t+1}^T. \quad (4.38)$$

For the case $\tilde{G}_{t+1} \in \mathbb{S}_{++}^{n_w}$ it is possible to use the Sherman-Morrison-Woodbury formula to simplify computations, see, e.g., Golub and Van Loan (1996).

4.2.4 Algorithms for modifying the Riccati factorization

In this section, the theory derived in Section 4.2.1-4.2.3 is summarized in Algorithm 8. The algorithm assigns $\alpha = 1$ for an update and $\alpha = -1$ for a downdate, and computes the modified matrices at time t_m . When $t < t_m$, Theorem 4.5 and the theory in Section 4.2.3 is used to modify the factorization for these subsequent time-steps. At lines 3 and 8 in Algorithm 8, the factorization should not be re-computed from scratch but instead standard methods for up- and downdating for example Cholesky factorizations should be used. By modifying the Riccati factorization instead of re-computing it, the computational complexity is reduced from roughly $\mathcal{O}(N(n_u^3 + n_x^3 + n_u^2 n_x + n_x^2 n_u))$ to $\mathcal{O}(t_m(n_u^2 n_x + n_u^2 + k n_u n_x + k n_x^2))$ when using the Cholesky factorization at Line 6 in Algorithm 4. The complexity is further reduced to approximately $\mathcal{O}(t_m(n_u^2 + k n_u^2 + k n_u n_x + k n_x^2))$ if Line 9 in Algorithm 8 is computed using the Sherman-Morrison-Woodbury formula, see, e.g., Golub and Van Loan (1996). The exact expression for the complexity of the algorithm is dependent on, e.g., the choice of factorization and updating methods in algorithms 4 and 8.

Remark 4.7. In Algorithm 8 the superscript “-” or “+” for C_t and V_t are not used. Instead, the parameter α determines how C_t and V_t are computed. $\square$

To compute the solution of the modified UFTOC problem after the modification of the factorization in Algorithm 4, the backward and forward recursions in algorithms 5 and 6 need to be re-computed using the modified matrices. Furthermore, Algorithm 7 needs to be re-computed to obtain the dual variables corresponding to the eliminated constraints. However, since the factorization is only affected for time-instances $t \in \mathbb{Z}_{0,t_m}$, the backward recursion in Algorithm 5 is only re-computed for these time indices using the modified matrices $\tilde{B}_{w,t_m}$, $\tilde{G}_t$, $\tilde{H}_t$ and $\tilde{P}_t$, which improves performance even more.

Algorithm 8 Low-rank modifications of Riccati factorization

```

1: Assign  $\alpha := 1$  (update) or  $\alpha := -1$  (downdate)
2: Calculate  $g, g^0, h, V_{t_m}, C_{t_m}, k^0$  and  $K^0$ 
3: Modify the factorization of  $G_{t_m+1}$  and compute  $\tilde{H}_{t_m+1}$  and  $\tilde{K}_{t_m+1}$ 
4:  $\tilde{P}_{t_m} := P_{t_m} + \alpha V_{t_m} C_{t_m}^\dagger V_{t_m}^T$ 
5: for  $t = t_m - 1, \dots, 0$  do
6:    $\tilde{F}_{t+1} := F_{t+1} + \alpha A_t^T V_{t+1} C_{t+1}^\dagger V_{t+1}^T A_t$ 
7:    $\tilde{H}_{t+1} := H_{t+1} + \alpha A_t^T V_{t+1} C_{t+1}^\dagger V_{t+1}^T B_{w,t}$ 
8:   Modify factorization for
      $\tilde{G}_{t+1} = G_{t+1} + \alpha B_{w,t}^T V_{t+1} C_{t+1}^\dagger V_{t+1}^T B_{w,t}$ 
9:   Compute a solution  $\tilde{K}_{t+1}$  to
      $\tilde{G}_{t+1} \tilde{K}_{t+1} = -\tilde{H}_{t+1}^T$ 
10:  if  $\alpha = 1$  then
11:     $C_t := C_{t+1} - V_{t+1}^T B_{w,t} \tilde{G}_{t+1}^\dagger B_{w,t}^T V_{t+1}$ 
12:     $V_t := (A_t - B_{w,t} \tilde{G}_{t+1}^\dagger \tilde{H}_{t+1}^T)^T V_{t+1}$ 
13:  else
14:     $C_t := C_{t+1} - V_{t+1}^T B_{w,t} G_{t+1}^\dagger B_{w,t}^T V_{t+1}$ 
15:     $V_t := (A_t - B_{w,t} G_{t+1}^\dagger H_{t+1}^T)^T V_{t+1}$ 
16:  end if
17:   $\tilde{P}_t := P_t + \alpha V_t C_t^\dagger V_t^T$ 
18: end for

```

4.3 Extension to General Constraints

The CFTOC problem that is the core of many MPC problems in industry is often of a more general type than (3.29) and also includes constraints on the states and the possibility to control only certain states or a combination of states (Maciejowski, 2002). In this section, it is shown how to exploit the low-rank modifications of the Riccati factorization in order to improve performance when solving more general problems than (3.29).

4.3.1 Primal and dual problem

In this section a primal CFTOC problem of a more general type than (3.1) and a corresponding dual problem is presented. This problem is chosen since it can be shown that for a certain choice of dual variables the dual problem is also in the form of a CFTOC problem, see Axehill (2005) and Appendix 4.B. This property will be exploited in this section. A superscript “ p ” is used to denote variables in the primal problem, and a superscript “ d ” is used to denote variables in the dual problem.

Primal problem

Consider an MPC problem and its corresponding CFTOC problem with states $x_t^p \in \mathbb{R}^{n_x}$, controlled variables $z_t^p \in \mathbb{R}^{n_z}$ and control inputs $u_t^p \in \mathbb{R}^{n_u}$, and with

inequality constraints that involve both states and control inputs. This CFTOC problem is in a general form that covers many linear MPC applications and is given by the following optimization problem

$$\begin{aligned}
& \underset{x^p, z^p, u^p}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + \begin{bmatrix} l_{z,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + c_t^p \right) + \\
& && \frac{1}{2} z_N^p{}^T Q_{z,N}^p z_N^p + l_{z,N}^p{}^T z_N^p + c_N^p \\
& \text{subject to} && x_0^p = \bar{x} \\
& && x_{t+1}^p = A_t^p x_t^p + B_t^p u_t^p + a_t^p, \quad t \in \mathbb{Z}_{0,N-1} \\
& && z_t^p = M_t^p x_t^p, \quad t \in \mathbb{Z}_{0,N} \\
& && H_{x,t}^p x_t^p + H_{u,t}^p u_t^p + h_t^p \leq 0, \quad t \in \mathbb{Z}_{0,N-1} \\
& && H_{x,N}^p x_N^p + h_N^p \leq 0,
\end{aligned} \tag{4.39}$$

where $H_{x,t}^p \in \mathbb{R}^{n_{c,t} \times n_x}$, $H_{u,t}^p \in \mathbb{R}^{n_{c,t} \times n_u}$ and $h_t^p \in \mathbb{R}^{n_{c,t}}$. Here, $n_{c,t}$ is the number of inequality constraints at time t . Furthermore, let Assumption 4.8 hold for the problem in (4.39).

Assumption 4.8.

$$\begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \in \mathbb{S}_{++}^{n_z+n_u}, \quad t \in \mathbb{Z}_{0,N-1}, \quad Q_{z,N}^p \in \mathbb{S}_{++}^{n_z}. \tag{4.40}$$

The CFTOC problem (4.39) can be solved by using for example a primal AS solver, but the theory presented in this chapter is not directly applicable in that case. Hence, utilizing low-rank modifications of the Riccati factorization instead of re-computing it cannot be exploited to increase performance. However, the primal problem in (4.39) can also be solved using a dual AS solver as in Goldfarb and Idrani (1983); Axehill and Hansson (2006), or using a dual gradient projection method as in Axehill (2008). In these types of solvers it is sometimes possible to exploit low-rank modifications of the Riccati factorization to speed up computations.

Dual problem

In Axehill (2005), an optimization problem which is equivalent to the (Lagrange) dual problem of a slightly less general problem than (4.39) is introduced. The corresponding primal problem does not have any linear or constant terms in the objective function and the cross-terms between z_t^p and u_t^p are zero. Furthermore, it is shown that also the dual problem can be interpreted as a CFTOC problem using a suitable variable transformation. This dual problem is used in the solvers in Axehill and Hansson (2006); Axehill (2008); Axehill and Hansson (2008).

In this thesis a dual problem to (4.39) is derived using analogous calculations as in Axehill (2005). The details of the derivations are presented in Appendix 4.B.

The dual problem can be interpreted as a CFTOC problem with states $x_\tau^d \in \mathbb{R}^{n_x}$, control inputs $u_\tau^d \in \mathbb{R}^{n_z+n_c, N-\tau}$ and prediction horizon $N^d = N + 1$, i.e.,

$$\begin{aligned}
 & \underset{x^d, u^d}{\text{minimize}} && \sum_{\tau=0}^{N^d-1} \left(\frac{1}{2} \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix}^T \begin{bmatrix} Q_{x,\tau}^d & Q_{xu,\tau}^d \\ (Q_{xu,\tau}^d)^T & Q_{u,\tau}^d \end{bmatrix} \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix} + \begin{bmatrix} l_{x,\tau}^d \\ l_{u,\tau}^d \end{bmatrix}^T \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix} + c_\tau^d \right) + \\
 & && (l_{x,N^d}^d)^T x_{N^d}^d \\
 & \text{subject to} && x_0^d = 0 \\
 & && x_{\tau+1}^d = A_\tau^d x_\tau^d + B_\tau^d u_\tau^d, \quad \tau \in \mathbb{Z}_{0,N^d-1} \\
 & && \begin{bmatrix} 0 & -I_{n_c, N-\tau} \end{bmatrix} u_\tau^d \leq 0, \quad \tau \in \mathbb{Z}_{0,N^d-1},
 \end{aligned} \tag{4.41}$$

where $\tau = N - t$ and

$$\begin{bmatrix} Q_{x,\tau}^d & Q_{xu,\tau}^d \\ (Q_{xu,\tau}^d)^T & Q_{u,\tau}^d \end{bmatrix} \in \mathbb{S}_+^{n_x+n_z+n_c, N-\tau}, \quad Q_{x,N^d}^d \in \mathbb{S}_+^{n_x}. \tag{4.42}$$

The state variables, control inputs and all matrices and vectors in the dual problem are defined in Appendix 4.B.

The solution to the primal CFTOC problem (4.39) can be derived from the solution to the dual CFTOC problem (4.41). The relations between the solutions to the primal and dual problems are given by (4.76) and (4.80) in Appendix 4.B, and are repeated here

$$x_t^p = -\lambda_{N+1-t}, \quad t \in \mathbb{Z}_{0,N}, \tag{4.43a}$$

$$\begin{aligned}
 u_t^p = & -(\bar{Q}_{zu,t})^T l_{z,t}^p - \bar{Q}_{u,t} l_{u,t}^p - \bar{Q}_{u,t} (B_{u,t}^p)^T x_{N-t}^d + \\
 & \left[(\bar{Q}_{zu,t})^T \quad -\bar{Q}_{u,t} (H_{u,t}^p)^T \right] u_{N-t}^d, \quad t \in \mathbb{Z}_{0,N-1},
 \end{aligned} \tag{4.43b}$$

where λ_τ are the dual variables corresponding to the equality constraints in the dual problem and are defined in (4.77a), and $\bar{Q}_{zu,t}$ and $\bar{Q}_{u,t}$ are defined in (4.66). Furthermore, the dual variables μ_τ corresponding to the inequality constraints in the dual problem are defined in (4.77b). From the relation (4.85), i.e.,

$$H_{x,t}^p x_t^p + H_{u,t}^p u_t^p + h_t = -\mu_{N-t}, \tag{4.44}$$

it is clear that these dual variables can be interpreted as the slack variables for the inequality constraints in the primal CFTOC problem (4.39).

4.3.2 Methods to handle general constraints

The primal problem (4.39) can be solved by using for example a dual AS type solver as proposed in Axehill and Hansson (2006), or a dual gradient projection method as in Axehill (2008); Axehill and Hansson (2008). In these types of methods, the primal problem is solved by constructing the corresponding dual problem and applying primal methods to solve the dual problem. The solver iterates until dual optimality is obtained, while maintaining dual feasibility in all itera-

tions. Since Slater's condition in Definition 2.6 is satisfied for (4.39), strong duality holds. Hence, if the dual problem is bounded from above, the dual optimal point for (4.41) corresponds to a primal feasible and optimal point for (4.39).

Remark 4.9. The primal method that is used to solve the dual problem must be able to handle the positive semi-definite case (Goldfarb and Idnani, 1983). In Axehill and Hansson (2006); Axehill (2008); Axehill and Hansson (2008), this is done by imposing extra assumptions on the inequality constraints in (4.39).

In both the previously mentioned dual AS type solver and the dual gradient projection method the solution to the primal CFTOC problem (4.39) is computed by solving a dual problem in the form (4.41). This is done by iteratively changing the working set to find the optimal active set as in a primal AS method. The dual CFTOC problem (4.41) is in the same form as the CFTOC problem (3.29). Hence, it is solved by computing a sequence of Newton steps corresponding to UFTOC problems in the form in (4.1). Since these solvers exploit the structure in (4.1) by using a Riccati recursion to compute the Newton steps, the theory presented in this chapter can be used to perform low-rank modifications of the Riccati factorization. A principal outline of a dual AS type solver is given in Algorithm 9.

Algorithm 9 Outline of dual AS type solver

- 1: Construct the dual problem (4.41) from the primal problem (4.39)
 - 2: Find a dual feasible point (the origin is a trivial choice for a dual AS solver and no phase I algorithm is needed) and an initial working set
 - 3: **repeat**
 - 4: Compute the Newton step by solving a subproblem in the form (4.1)
 - 5: Perform the Newton step in the dual variables x^d and u^d and update the working set if needed
 - 6: **until** Dual optimal point x^{d*} and u^{d*} is found
 - 7: Transform the dual optimal point to the primal optimal point according to (4.43a) and (4.43b)
-

However, when a dual solver is used to solve (4.39), primal feasibility is obtained only at the optimum (Goldfarb and Idnani, 1983; Bartlett and Biegler, 2006). In a real-time MPC control loop this might be problematic due to real-time computation time constraints. If the solver used to compute the optimal solution to the primal CFTOC problem is forced to terminate before an optimum is found, the control input that is returned by the solver is not necessarily primal feasible. Hence, either a control signal which might lead to violation of constraints is used as input to the plant, or the control signal has to be computed in some alternative way. An approach to address this problem and still be able to exploit the low-rank modifications of the Riccati recursion presented here is to use a primal solver which maintains primal feasibility, but computing the Newton step by solving a dual UFTOC problem. This procedure is described for a primal CFTOC problem in the form (4.39), and is briefly outlined in Algorithm 10 at the end of this section.

When a primal AS method like for example the one given in Algorithm 1 is applied to solve the CFTOC problem (4.39), an optimization problem in the form

$$\begin{aligned}
 & \underset{x^p, z^p, u^p}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + \begin{bmatrix} l_{z,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + c_t^p \right) + \\
 & && \frac{1}{2} z_N^p{}^T Q_{z,N}^p z_N^p + l_{z,N}^p{}^T z_N^p + c_N^p \\
 & \text{subject to} && x_0^p = \bar{x} \\
 & && x_{t+1}^p = A_t^p x_t^p + B_t^p u_t^p + a_t^p, \quad t \in \mathbb{Z}_{0,N-1} \\
 & && z_t^p = M_t^p x_t^p, \quad t \in \mathbb{Z}_{0,N} \\
 & && H_{x,i,t}^p x_t^p + H_{u,i,t}^p u_t^p + h_{i,t}^p = 0, \quad \forall (i, t) \in \mathcal{W}_j \\
 & && H_{x,i,N}^p x_N^p + h_{i,N}^p = 0, \quad \forall (i, N) \in \mathcal{W}_j,
 \end{aligned} \tag{4.45}$$

is solved at each AS iteration j . Here, $\mathcal{W}_j$ denotes the working set and

$$H_{x,t}^p = \begin{bmatrix} H_{x,1,t}^p \\ \vdots \\ H_{x,n_c,t}^p \end{bmatrix}, \quad H_{u,t}^p = \begin{bmatrix} H_{u,1,t}^p \\ \vdots \\ H_{u,n_c,t}^p \end{bmatrix}, \quad h_t^p = \begin{bmatrix} h_{1,t}^p \\ \vdots \\ h_{n_c,t}^p \end{bmatrix}. \tag{4.46}$$

Furthermore, the notation $(i, t) \in \mathcal{W}_j$ is introduced to denote that the inequality constraint

$$H_{x,i,t}^p x_t^p + H_{u,i,t}^p u_t^p + h_{i,t}^p \leq 0, \tag{4.47}$$

is in the working set and is hence forced to hold with equality. In principal, it is possible to solve (4.45) by solving the dual UFTOC problem and then computing the corresponding primal solution from the solution to the dual problem. The dual UFTOC problem is constructed from the problem in (4.41) by letting some of the inequality constraints in (4.41) hold with equality, and temporarily disregarding the rest. For all inequality constraints (i, t) that are forced to hold with equality in (4.45), the corresponding dual variable $\gamma_{i,t}$ (defined in (4.59d) and (4.59e)) is an unconstrained optimization variable. Hence, by using the definition of u_τ^d in (4.72) in Appendix 4.B it follows that $u_{n_z+i,N-t}^d$ is also an unconstrained optimization variable in the dual problem. Similarly, if a primal inequality constraint (i, t) is temporarily disregarded, i.e., it is not in the working set, the corresponding dual variable is defined to be $\gamma_{i,t} = 0$ (Nocedal and Wright, 2006). It then follows that the corresponding u_τ^d is constrained to zero, i.e., $u_{n_z+i,N-t}^d = 0$. To summarize; if the primal constraint (i, t) is in the working set, then the corresponding dual control input is unconstrained in the dual problem, and if the primal constraint (i, t) is not in the working set, the corresponding dual control input is constrained to zero.

Using the reasoning above, the dual problem of the problem in (4.45) is given by

$$\begin{aligned}
 & \underset{x^d, u^d}{\text{minimize}} && \sum_{\tau=0}^{N^d-1} \left(\frac{1}{2} \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix}^T \begin{bmatrix} Q_{x,\tau}^d & Q_{xu,\tau}^d \\ (Q_{xu,\tau}^d)^T & Q_{u,\tau}^d \end{bmatrix} \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix} + \begin{bmatrix} l_{x,\tau}^d \\ l_{u,\tau}^d \end{bmatrix}^T \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix} + c_\tau^d \right) + \\
 & && \left(l_{x,N^d}^d \right)^T x_{N^d}^d \\
 & \text{subject to} && x_0^d = 0 \\
 & && x_{\tau+1}^d = A_\tau^d x_\tau^d + B_\tau^d u_\tau^d, \quad \tau \in \mathbb{Z}_{0,N^d-1} \\
 & && -u_{n_z+i,\tau}^d = 0, \quad \forall (i, N-\tau) \in \mathcal{W}_j^c.
 \end{aligned} \tag{4.48}$$

The primal solver requests a solution to (4.45) which is computed via (4.48). The solution to (4.45) can easily be recovered from the solution to (4.48). This process is transparent to the primal solver. By eliminating the constrained variables $u_{n_z+i,\tau}^d$ in (4.48) this problem can be interpreted as a UTOC problem in the same form as (4.1). Hence (4.48) can be solved using a Riccati recursion after eliminating the constrained control inputs. The dual solution is transformed to the primal variables using (4.43a) and (4.43b), and the Newton step can be performed in the primal variables in the AS solver, maintaining primal feasibility.

Remark 4.10. Performing the Newton step in the primal AS solver does not depend on how the step direction is computed. Hence, the logics involved in performing the step and possibly adding or removing constraints from the working set are the same regardless of how the Newton step direction is computed.

Furthermore, the dual problem to the primal problem (4.45) is constructed such that adding a primal constraint to the working set corresponds to removing a dual constraint, and vice versa. Hence, the dual problem is changed as in Section 3.2.2 when primal inequality constraints are added or removed from the working set. As a result, the low-rank modification algorithms presented in this chapter can be applied to increase performance when computing the solution to the dual UTOC problem. This solver is outlined in Algorithm 10.

Algorithm 10 Outline of primal AS solver using dual Newton step computation

- 1: Construct the dual problem (4.41) from the primal problem (4.39)
 - 2: Find a primal feasible point (for example by using a Phase I approach) and an initial working set
 - 3: **repeat**
 - 4: Form the dual UTOC problem (4.48)
 - 5: Solve the problem (4.48)
 - 6: Compute the variables x_{j+1}^p and u_{j+1}^p using (4.43a) and (4.43b) and, if needed, α_{j+1} , β_{j+1} and γ_{j+1} from (4.71) and (4.72) in Appendix 4.B
 - 7: Perform the Newton step in the primal variables x^p and u^p and change the working set if necessary
 - 8: **until** Primal optimal point x^{p*} and u^{p*} is found
-

Remark 4.11. In Algorithm 10 the dual CFTOC problem (4.41) is constructed only once. By inspecting the structure of the dual UFTOC problem that is solved at each AS iteration, it can be seen that only the equality constraints $u_{n_z+i,\tau}^d = 0$ are changed between different iterations. Hence, at Line 4 in Algorithm 10 only computationally cheap operations are needed.

4.4 Numerical Results

The proposed algorithms for solving the KKT system of (4.1) by modifying the Riccati factorization have been implemented in MATLAB, and a comparison with a MATLAB implementation of a standard Riccati recursion has been performed. The focus of the comparison is the relative performance gain in terms of computation time obtained by modifying the Riccati factorization instead of re-computing it. The algorithms are compared by solving random UFTOC problems of varying dimensions and the computation times are averaged over fifteen problems of the same dimensions.

All simulations were performed on an Intel®Xeon® E5-2650 CPU running Linux (version 2.6.32-504.16.2.el6.x86_64) and MATLAB (version 8.0.0.783, R2012b).

Remark 4.12. Although no numerical issues have been observed in the double-precision implementation of the algorithms in this chapter, it might occur in implementations using lower-precision arithmetics. For such applications, a square-root implementation of the algorithms can be useful to increase numerical stability (Kailath, 1980).

4.4.1 Modification of Riccati factorization

Most of the main operations have been implemented in m-code and the number of computational threads is limited to one to get a comparison of the computational effort that is required by modifying and re-computing the Riccati recursion, respectively. However, standard basic MATLAB operations such as matrix multiplications and additions have been used. To get a completely valid comparison of the two approaches, all operations should be implemented in either m-code or in highly efficient linear algebra packages such as BLAS. The comparison in this section thus shows less performance gains than what is possible since the dominating n_x^3 -complexities are efficiently implemented internally in MATLAB using optimized linear algebra routines from commercial software, compared to the n_u^3 -complexities that are implemented by the author in m-code.

The prediction horizon has been chosen as $N = 100$ for all UFTOC problems, and one constraint has been removed at different time steps $t_m \in \{1, 25, 50, 75, 100\}$, i.e., ranging from modifying only one step of the factorization to modifying the full factorization (giving the worst-case scenario). The problem dimensions n_x and n_u have been chosen in the interval $[10, 200]$, and the relative computation times averaged over 15 problems of the same size when $t_m = 1$ and $t_m = 100$ are presented in Figure 4.1. In the figures 4.2 and 4.3 the results when n_x is fixed to $n_x = 20$ and n_u is fixed to $n_u = 20$, respectively, are presented for different

Figure 4.1: Comparison of the computation time that is required to compute the Riccati recursion using the standard algorithms in Section 3.4 and the modifying algorithms presented in Chapter 4. The computation times are compared when removing a constraint from the working set at $t_m = 100 = N$ (worst-case) and $t_m = 1$.

values of t_m . From these figures, it is clear that modifying the Riccati recursion instead of re-computing it can significantly reduce the required computation time for solving the UFTOC problem. This is true especially for larger problem sizes and/or when a small part of the factorization is affected by the change of the working set.

Figure 4.2: Comparison of the computation time that is required to compute the Riccati recursion using the standard algorithms in Section 3.4 and the modifying algorithms presented in Chapter 4. The computation times are compared when removing a constraint from the working set at $t_m = 100 = N$ (worst-case) and $t_m = 1$ for $n_x = 20$.

Figure 4.3: Comparison of the computation time that is required to compute the Riccati recursion using the standard algorithms in Section 3.4 and the modifying algorithms presented in Chapter 4. The computation times are compared when removing a constraint from the working set at $t_m = 100 = N$ (worst-case) and $t_m = 1$ for $n_u = 20$.

Appendix

In this appendix the proof of Lemma 4.1 is given. Furthermore, the derivation of the dual CFTOC problem (4.41) and the definition of the dual variables are presented here.

4.A Proof of Lemma 4.1

Proof: Consider the positive semidefinite matrices $\mathcal{M} \geq 0$ and $\bar{\mathcal{M}} \geq 0$ which are partitioned as

$$\mathcal{M} = \begin{bmatrix} M_{11} & \hat{\mathcal{M}} \\ \hat{\mathcal{M}}^T & \bar{\mathcal{M}} \end{bmatrix}, \quad \bar{\mathcal{M}} = \begin{bmatrix} M_{22} & M_{23} \\ M_{23}^T & M_{33} \end{bmatrix}, \quad \text{where } \hat{\mathcal{M}} = \begin{bmatrix} M_{12} & M_{13} \end{bmatrix}. \quad (4.49)$$

From Theorem A.8 on page 115 in Appendix A and $\bar{\mathcal{M}} \geq 0$ it is clear that

$$\bar{\mathcal{M}}/^\dagger M_{22} = M_{33} - M_{23}^T M_{22}^\dagger M_{23} \geq 0. \quad (4.50)$$

Furthermore, since $\mathcal{M} \geq 0$ it follows from Theorem A.8 on page 115 that $\hat{\mathcal{M}}^T \in \mathcal{R}(\bar{\mathcal{M}})$, and from Theorem A.11 on page 115 it follows that that

$$\left(M_{13}^T - M_{23}^T M_{22}^\dagger M_{12}^T \right) \in \mathcal{R} \left(M_{33} - M_{23}^T M_{22}^\dagger M_{23} \right). \quad (4.51)$$

By defining the perturbation matrix

$$\Pi \triangleq \begin{bmatrix} 0 & 0 & I \\ 0 & I & 0 \\ I & 0 & 0 \end{bmatrix}, \quad (4.52)$$

the matrix A used in Theorem 4 in Carlson et al. (1974) can be written

$$A = \Pi \mathcal{M} \Pi^T. \quad (4.53)$$

Hence, the relation

$$\left(\mathcal{M}/^\dagger M_{33} \right)^\dagger \left(\bar{\mathcal{M}}/^\dagger M_{33} \right) = \mathcal{M}/^\dagger \bar{\mathcal{M}}, \quad (4.54)$$

follows directly from Theorem 4 in Carlson et al. (1974).

Furthermore, since $\mathcal{M} \geq 0$ there exists a matrix P such that

$$\bar{\mathcal{M}}P = \hat{\mathcal{M}}^T, \quad (4.55)$$

and the matrix $\mathcal{M}$ can thus be written

$$\mathcal{M} = \begin{bmatrix} M_{11} & P^T \bar{\mathcal{M}} \\ \bar{\mathcal{M}}P & \bar{\mathcal{M}} \end{bmatrix}. \quad (4.56)$$

By using this representation the Schur complement $\mathcal{M}/^\dagger \bar{\mathcal{M}}$ can be written

$$\mathcal{M}/^\dagger \bar{\mathcal{M}} = M_{11} - P^T \bar{\mathcal{M}} \bar{\mathcal{M}}^\dagger \bar{\mathcal{M}} P = M_{11} - P^T \bar{\mathcal{M}} P. \quad (4.57)$$

Here, the property (A.1) on page 113 of the pseudo inverse has been used.

From Theorem A.11 on page 115, the solution to (4.55) can be computed. By inserting this solution into (4.57) and using standard matrix computations gives

$$\begin{aligned} \mathcal{M}/^\dagger \bar{\mathcal{M}} &= M_{11} - P^T \bar{\mathcal{M}} P = M_{11} - M_{12} M_{22}^\dagger M_{12}^T - \\ &\quad (M_{13} - M_{12} M_{22}^\dagger M_{23}) (M_{33} - M_{23}^T M_{22}^\dagger M_{23})^\dagger (M_{13}^T - M_{23}^T M_{22}^\dagger M_{12}^T), \end{aligned} \quad (4.58)$$

which completes the proof. □

4.B Derivation of the Dual Problem

The derivation of the dual CFTOC problem (4.41) will be similar to the derivation of the dual problem in Axehill (2005). However, the slightly more general CFTOC problem (4.39) is considered in this section.

4.B.1 Construct the dual problem

Consider a primal CFTOC problem given in the form in (4.39) where Assumption 4.8 hold. Introduce the dual variables α_t , β_t and γ_t corresponding to the constraints in (4.39) as

$$\alpha_0 \leftrightarrow -x_0^p + \bar{x} = 0, \quad (4.59a)$$

$$\alpha_{t+1} \leftrightarrow -x_{t+1}^p + A_t^p x_t^p + B_t^p u_t^p + a_t^p = 0, \quad t \in \mathbb{Z}_{0,N-1}, \quad (4.59b)$$

$$\beta_t \leftrightarrow -z_t^p + M_t^p x_t^p = 0, \quad t \in \mathbb{Z}_{0,N}, \quad (4.59c)$$

$$\gamma_t \leftrightarrow H_{x,t}^p x_t^p + H_{u,t}^p u_t^p + h_t^p \leq 0, \quad t \in \mathbb{Z}_{0,N-1}, \quad (4.59d)$$

$$\gamma_N \leftrightarrow H_{x,N}^p x_N^p + h_N^p \leq 0, \quad (4.59e)$$

where the symbol $\leftrightarrow$ is introduced in Definition 3.8.

The first step to derive the dual problem is to use the definitions of these dual variables to form the Lagrangian of the primal CFTOC problem. By using the definition of the Lagrangian in (2.8) and the dual variables defined in (4.59), the

Lagrangian is given by

$$\begin{aligned}
L_P(x^P, u^P, z^P, \alpha, \beta, \gamma) = & \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + \begin{bmatrix} l_{z,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + c_t^p \right) + \\
& \frac{1}{2} z_N^p{}^T Q_{z,N}^p z_N^p + l_{z,N}^p{}^T z_N^p + c_N^p + \sum_{t=0}^{N-1} \alpha_{t+1}^T (-x_{t+1}^p + A_t^p x_t^p + B_t^p u_t^p + a_t^p) + \\
& \alpha_0^T (-x_0^p + \bar{x}) + \sum_{t=0}^N \beta_t^T (-z_t^p + M_t^p x_t^p) + \sum_{t=0}^{N-1} \gamma_t^T (H_{x,t}^p x_t^p + H_{u,t}^p u_t^p + h_t^p) + \\
& \gamma_N^T (H_{x,t}^p x_t^p + h_t^p), \tag{4.60}
\end{aligned}$$

with the corresponding Lagrange dual function

$$g(\alpha, \beta, \gamma) = \inf_{x^P, u^P, z^P} L_P(x^P, u^P, z^P, \alpha, \beta, \gamma). \tag{4.61}$$

To simplify the minimization over the primal variables x^P, u^P, z^P in (4.61), the terms in the Lagrangian (4.60) are re-ordered, resulting in

$$\begin{aligned}
L_P(x^P, u^P, z^P, \alpha, \beta, \gamma) = & \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + \begin{bmatrix} l_{z,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + c_t^p \right) + \\
& \frac{1}{2} z_N^p{}^T Q_{z,N}^p z_N^p + l_{z,N}^p{}^T z_N^p + c_N^p + \sum_{t=0}^{N-1} (\alpha_{t+1}^T B_t^p + \gamma_t^T H_{u,t}^p) u_t^p - \sum_{t=0}^N \beta_t^T z_t^p + \\
& \sum_{t=0}^{N-1} (-\alpha_t^T + \alpha_{t+1}^T A_t^p + \beta_t^T M_t^p + \gamma_t^T H_{x,t}^p) x_t^p + (-\alpha_N^T + \beta_N^T M_N^p + \gamma_N^T H_{x,t}^p) x_N^p + \\
& \alpha_0^T \bar{x} + \sum_{t=0}^{N-1} \alpha_{t+1}^T a_t^p + \sum_{t=0}^{N-1} \gamma_t^T h_t^p + \gamma_N^T h_t^p. \tag{4.62}
\end{aligned}$$

When Assumption 4.8 hold it is clear that (4.62) is strictly convex in the primal variables u^P and z^P , and linear in the primal variable x^P . Hence, if (4.62) is minimized with respect to all primal variables, the optimal value is not bounded below unless the coefficient in front of x^P is zero. As a result, explicit equality constraints that exclude all the values of the dual variables such that $g(\alpha, \beta, \gamma) = -\infty$ can be formed as

$$-\alpha_t + (A_t^p)^T \alpha_{t+1} + (M_t^p)^T \beta_t + (H_{x,t}^p)^T \gamma_t = 0, \tag{4.63a}$$

$$-\alpha_N + (M_N^p)^T \beta_N + (H_{x,t}^p)^T \gamma_N = 0. \tag{4.63b}$$

These can be identified as the transpose of the coefficient in front of x_t^p and x_N^p in (4.62), respectively. By inserting (4.63a) and (4.63b) into (4.62), and re-

arranging some terms, the Lagrangian can be written

$$\begin{aligned}
 L_P(u^p, z^p, \alpha, \beta, \gamma) = & \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix}^T \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix} \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + \right. \\
 & \left. \begin{bmatrix} l_{z,t}^p - \beta_t \\ l_{u,t}^p + (B_t^p)^T \alpha_{t+1} + (H_{u,t}^p)^T \gamma_t \end{bmatrix}^T \begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} + c_t^p + (a_t^p)^T \alpha_{t+1} + (h_t^p)^T \gamma_t \right) + \\
 & \frac{1}{2} z_N^p{}^T Q_{z,N}^p z_N^p + (l_{z,N}^p - \beta_N)^T z_N^p + c_N^p + (h_N^p)^T \gamma_N + \bar{x}^T \alpha_0,
 \end{aligned} \quad (4.64)$$

which is a strictly convex quadratic function of z^p and u^p . From basic optimization results it follows that the minimum of (4.64) with respect to z^p and u^p is obtained at

$$\begin{bmatrix} z_t^p \\ u_t^p \end{bmatrix} = - \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix}^{-1} \begin{bmatrix} l_{z,t}^p - \beta_t \\ l_{u,t}^p + (B_t^p)^T \alpha_{t+1} + (H_{u,t}^p)^T \gamma_t \end{bmatrix}, \quad (4.65a)$$

$$z_N^p = - (Q_{z,N}^p)^{-1} (l_{z,N}^p - \beta_N). \quad (4.65b)$$

Inserting this solution in the Lagrangian (4.64) and introducing the notation

$$\begin{bmatrix} \bar{Q}_{z,t} & \bar{Q}_{zu,t} \\ (\bar{Q}_{zu,t})^T & \bar{Q}_{u,t} \end{bmatrix} \triangleq \begin{bmatrix} Q_{z,t}^p & Q_{zu,t}^p \\ (Q_{zu,t}^p)^T & Q_{u,t}^p \end{bmatrix}^{-1}, \quad (4.66)$$

to simplify notation gives (after standard, but tedious, matrix algebra and rearrangements) the Lagrange dual function

$$\begin{aligned}
 g(\alpha, \beta, \gamma) = & - \left(\sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} \alpha_{t+1} \\ \beta_t \\ \gamma_t \end{bmatrix}^T \begin{bmatrix} B_t^p \bar{Q}_{u,t} (B_t^p)^T & -B_t^p (\bar{Q}_{zu,t})^T & B_t^p \bar{Q}_{u,t} (H_{u,t}^p)^T \\ -\bar{Q}_{zu,t} (B_t^p)^T & \bar{Q}_{z,t} & -\bar{Q}_{zu,t} (H_{u,t}^p)^T \\ H_{u,t}^p \bar{Q}_{u,t} (B_t^p)^T & -H_{u,t}^p (\bar{Q}_{zu,t})^T & H_{u,t}^p \bar{Q}_{u,t} (H_{u,t}^p)^T \end{bmatrix} \begin{bmatrix} \alpha_{t+1} \\ \beta_t \\ \gamma_t \end{bmatrix} + \right. \\
 & \left. \begin{bmatrix} -a_t^p + B_t^p (\bar{Q}_{zu,t})^T l_{z,t}^p + B_t^p \bar{Q}_{u,t} l_{u,t}^p \\ -(\bar{Q}_{z,t} l_{z,t}^p + (\bar{Q}_{zu,t})^T l_{u,t}^p) \\ -h_t^p + H_{u,t}^p (\bar{Q}_{zu,t})^T l_{z,t}^p + H_{u,t}^p \bar{Q}_{u,t} l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} \alpha_{t+1} \\ \beta_t \\ \gamma_t \end{bmatrix} + \begin{bmatrix} l_{z,t}^p \\ l_{u,t}^p \end{bmatrix}^T \begin{bmatrix} \bar{Q}_{z,t} & \bar{Q}_{zu,t} \\ (\bar{Q}_{zu,t})^T & \bar{Q}_{u,t} \end{bmatrix} \begin{bmatrix} l_{z,t}^p \\ l_{u,t}^p \end{bmatrix} - c_t^p \right) \\
 & + \frac{1}{2} \begin{bmatrix} \beta_N \\ \gamma_N \end{bmatrix}^T \begin{bmatrix} (Q_{z,N}^p)^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \beta_N \\ \gamma_N \end{bmatrix} + \begin{bmatrix} -(Q_{z,N}^p)^{-1} l_{z,N}^p \\ -h_N^p \end{bmatrix}^T \begin{bmatrix} \beta_N \\ \gamma_N \end{bmatrix} + \\
 & \left. \frac{1}{2} l_{z,N}^p{}^T (Q_{z,N}^p)^{-1} l_{z,N}^p - c_N^p - \bar{x}^T \alpha_0 \right),
 \end{aligned} \quad (4.67)$$

which is a concave function. Hence, the dual problem to (4.39) can be written as

$$\begin{aligned}
 & \underset{\alpha, \beta, \gamma}{\text{maximize}} && g(\alpha, \beta, \gamma) \\
 & \text{subject to} && -\alpha_t + (A_t^p)^T \alpha_{t+1} + (M_t^p)^T \beta_t + (H_{x,t}^p)^T \gamma_t = 0, \quad t \in \mathbb{Z}_{0,N-1} \\
 & && -\alpha_N + (M_N^p)^T \beta_N + (H_{x,N}^p)^T \gamma_N = 0 \\
 & && \gamma_t \geq 0, \quad t \in \mathbb{Z}_{0,N},
 \end{aligned} \tag{4.68}$$

or equivalent

$$\begin{aligned}
 & \underset{\alpha, \beta, \gamma}{\text{minimize}} && -g(\alpha, \beta, \gamma) \\
 & \text{subject to} && -\alpha_t + (A_t^p)^T \alpha_{t+1} + (M_t^p)^T \beta_t + (H_{x,t}^p)^T \gamma_t = 0, \quad t \in \mathbb{Z}_{0,N-1} \\
 & && -\alpha_N + (M_N^p)^T \beta_N + (H_{x,N}^p)^T \gamma_N = 0 \\
 & && \gamma_t \geq 0, \quad t \in \mathbb{Z}_{0,N}.
 \end{aligned} \tag{4.69}$$

Here (4.69) is formed by using the fact that maximizing the concave function $g(\alpha, \beta, \gamma)$ subject to the constraints corresponds to minimizing the convex function $-g(\alpha, \beta, \gamma)$ subject to the same constraints (Boyd and Vandenberghe, 2004).

It will now be shown that the dual problem (4.69) can be interpreted as a new CFTOC problem in the form in (3.1) by introducing new variables $x_\tau^d \in \mathbb{R}^{n_x}$ and $u_\tau^d \in \mathbb{R}^{n_z+n_{c,N-\tau}}$. The dimensions will be clear from the definitions. Now define $\tau \triangleq N - t$, $N^d \triangleq N + 1$, and let x_τ^d and u_τ^d be defined as

$$x_0^d \triangleq 0, \tag{4.70}$$

$$x_\tau^d \triangleq \alpha_{N+1-\tau}, \quad \tau \in \mathbb{Z}_{1,N+1}, \tag{4.71}$$

$$u_\tau^d \triangleq \begin{bmatrix} \beta_{N-\tau} \\ \gamma_{N-\tau} \end{bmatrix}, \quad \tau \in \mathbb{Z}_{0,N}, \tag{4.72}$$

and define the following matrices, vectors and constants ($\tau \in \mathbb{Z}_{1,N}$ if not stated otherwise)

$$A_0^d \triangleq 0, \quad Q_{x,0}^d \triangleq 0, \quad Q_{xu,0}^d \triangleq 0, \quad l_{x,0}^d \triangleq 0, \tag{4.73a}$$

$$A_\tau^d \triangleq (A_{N-\tau}^p)^T, \tag{4.73b}$$

$$B_\tau^d \triangleq \begin{bmatrix} (M_{N-\tau}^p)^T & (H_{x,N-\tau}^p)^T \end{bmatrix}, \quad \tau \in \mathbb{Z}_{0,N}, \tag{4.73c}$$

$$Q_{x,\tau}^d \triangleq B_{N-\tau}^p \bar{Q}_{u,N-\tau} (B_{N-\tau}^p)^T, \tag{4.73d}$$

$$Q_{u,0}^d \triangleq \begin{bmatrix} (Q_{z,N}^p)^{-1} & 0 \\ 0 & 0 \end{bmatrix}, \tag{4.73e}$$

$$Q_{u,\tau}^d \triangleq \begin{bmatrix} \bar{Q}_{z,N-\tau} & -\bar{Q}_{zu,N-\tau} (H_{u,N-\tau}^p)^T \\ -H_{u,N-\tau}^p (\bar{Q}_{zu,N-\tau})^T & H_{u,N-\tau}^p \bar{Q}_{u,N-\tau} (H_{u,N-\tau}^p)^T \end{bmatrix}, \tag{4.73f}$$

$$Q_{xu,\tau}^d \triangleq \begin{bmatrix} -B_{N-\tau}^p (\bar{Q}_{zu,N-\tau})^T & B_{N-\tau}^p \bar{Q}_{u,N-\tau} (H_{u,N-\tau}^p)^T \end{bmatrix}, \quad (4.73g)$$

$$l_{x,\tau}^d \triangleq -a_{N-\tau}^p + B_{N-\tau}^p (\bar{Q}_{zu,N-\tau})^T l_{z,N-\tau}^p + B_{N-\tau}^p \bar{Q}_{u,N-\tau} l_{u,N-\tau}^p, \quad (4.73h)$$

$$l_{u,0}^d \triangleq \begin{bmatrix} -(Q_{z,N}^p)^{-1} l_{z,N}^p \\ -h_N^p \end{bmatrix}, \quad (4.73i)$$

$$l_{u,\tau}^d \triangleq \begin{bmatrix} -\bar{Q}_{z,N-\tau} l_{z,N-\tau}^p - (\bar{Q}_{zu,N-\tau})^T l_{u,N-\tau}^p \\ -h_{N-\tau}^p + H_{u,N-\tau}^p (\bar{Q}_{zu,N-\tau})^T l_{z,N-\tau}^p + H_{u,N-\tau}^p \bar{Q}_{u,N-\tau} l_{u,N-\tau}^p \end{bmatrix}, \quad (4.73j)$$

$$l_{x,N^d}^d \triangleq -\bar{x}, \quad (4.73k)$$

$$c_0^d \triangleq \frac{1}{2} l_{z,N}^p{}^T (Q_{z,N}^p)^{-1} l_{z,N}^p - c_N^p, \quad (4.73l)$$

$$c_\tau^d \triangleq \frac{1}{2} \begin{bmatrix} l_{z,N-\tau}^p \\ l_{u,N-\tau}^p \end{bmatrix}^T \begin{bmatrix} \bar{Q}_{z,N-\tau} & \bar{Q}_{zu,N-\tau} \\ (\bar{Q}_{zu,N-\tau})^T & \bar{Q}_{u,N-\tau} \end{bmatrix} \begin{bmatrix} l_{z,N-\tau}^p \\ l_{u,N-\tau}^p \end{bmatrix} - c_{N-\tau}^p. \quad (4.73m)$$

Then, the dual problem (4.69) can be equivalently written

$$\begin{aligned} & \underset{x^d, u^d}{\text{minimize}} && \sum_{\tau=0}^{N^d-1} \left(\frac{1}{2} \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix}^T \begin{bmatrix} Q_{x,\tau}^d & Q_{xu,\tau}^d \\ (Q_{xu,\tau}^d)^T & Q_{u,\tau}^d \end{bmatrix} \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix} + \begin{bmatrix} l_{x,\tau}^d \\ l_{u,\tau}^d \end{bmatrix}^T \begin{bmatrix} x_\tau^d \\ u_\tau^d \end{bmatrix} + c_\tau^d \right) + \\ & && \left(l_{x,N^d}^d \right)^T x_{N^d}^d \\ & \text{subject to} && x_0^d = 0 \\ & && x_{\tau+1}^d = A_\tau^d x_\tau^d + B_\tau^d u_\tau^d, \quad \tau \in \mathbb{Z}_{0,N^d-1} \\ & && \begin{bmatrix} 0 & -I_{n_{c,N-\tau}} \end{bmatrix} u_\tau^d \leq 0, \quad \tau \in \mathbb{Z}_{0,N^d-1}. \end{aligned} \quad (4.74)$$

Using these definitions, the dual problem (4.74) can be interpreted as a CFTOC problem in the form in (3.1).

4.B.2 Relation between primal and dual variables

The connection between the primal variables x_t^p and u_t^p , and the solution to the dual problem (4.74) is investigated in this section. The relation between these variables can be used to compute the primal solution from the dual solution. The primal control inputs can be computed using (4.65a) giving

$$u_t^p = -(\bar{Q}_{zu,t})^T l_{z,t}^p + (\bar{Q}_{zu,t})^T \beta_t - \bar{Q}_{u,t} l_{u,t}^p - \bar{Q}_{u,t} (B_t^p)^T \alpha_{t+1} - \bar{Q}_{u,t} (H_{u,t}^p)^T \gamma_t, \quad (4.75)$$

which by using the definition of x_τ^d and u_τ^d in (4.71) and (4.72) is equivalent to

$$u_t^p = -(\bar{Q}_{zu,t})^T l_{z,t}^p - \bar{Q}_{u,t} l_{u,t}^p - \bar{Q}_{u,t} (B_t^p)^T x_{N-t}^d + \left[(\bar{Q}_{zu,t})^T \quad -\bar{Q}_{u,t} (H_{u,t}^p)^T \right] u_{N-t}^d. \quad (4.76)$$

Hence, given the dual state variables x_τ^d and dual control inputs u_τ^d , the primal control input is computed using (4.76).

To compute the primal state variables x_t^p , the optimality conditions (2.17a) for the dual problem (4.74) is used. Let the dual variables to the dual problem be defined as

$$\lambda_{\tau+1} \leftrightarrow -x_{\tau+1}^d + A_\tau^d x_\tau^d + B_\tau^d u_\tau^d = 0, \quad \tau \in \mathbb{Z}_{0,N^d-1}, \quad (4.77a)$$

$$\mu_\tau \leftrightarrow \begin{bmatrix} 0 & -I_{n_{c,N-\tau}} \end{bmatrix} u_\tau^d \leq 0, \quad \tau \in \mathbb{Z}_{0,N^d-1}. \quad (4.77b)$$

Then, by using the definition of $Q_{x,\tau}^d$, $Q_{xu,\tau}^d$, $l_{x,\tau}^d$ and A_τ^d in (4.73d), (4.73g), (4.73h) and (4.73b), respectively, the optimality condition (2.17a) gives (for $\tau \in \mathbb{Z}_{0,N^d-1}$)

$$\begin{aligned} Q_{x,\tau}^d x_\tau^d + Q_{xu,\tau}^d u_\tau^d + l_{x,\tau}^d + (A_\tau^d)^T \lambda_{\tau+1} - \lambda_\tau &= 0 \iff \\ \lambda_\tau &= A_\tau^p \lambda_{\tau+1} + B_\tau^p \bar{Q}_{u,t} (B_\tau^p)^T x_\tau^d - B_\tau^p \left[(\bar{Q}_{zu,t})^T \quad -\bar{Q}_{u,t} (H_{u,t}^p)^T \right] u_\tau^d + \\ B_\tau^p (\bar{Q}_{zu,t})^T l_{z,t}^p + B_\tau^p \bar{Q}_{u,t} l_{u,t}^p - a_t^p &= A_\tau^p \lambda_{\tau+1} - B_\tau^p u_t^p - a_t^p. \end{aligned} \quad (4.78)$$

For $\tau = N^d$ the optimality condition is

$$l_{x,N^d}^d - \lambda_{N^d} = 0 \iff \lambda_{N^d} = l_{x,N^d}^d = -\bar{x}, \quad (4.79)$$

where the definition of l_{x,N^d}^d in (4.73k) has been used. Hence, from (4.78) and (4.79) it can be seen that if the primal variables x_t^p are computed as

$$x_t^p = -\lambda_{N+1-t}, \quad (4.80)$$

they will satisfy the dynamics equations in the primal problem (4.39). By inserting this choice of the primal variables into (4.78) and (4.79) gives

$$-x_{t+1}^p = -A_t^p x_t^p - B_t^p u_t^p - a_t^p, \quad t \in \mathbb{Z}_{0,N-1} \iff x_{t+1}^p = A_t^p x_t^p + B_t^p u_t^p + a_t^p, \quad t \in \mathbb{Z}_{0,N-1}, \quad (4.81)$$

and

$$x_0^p = \bar{x}. \quad (4.82)$$

It can easily be verified that (4.81) and (4.82) are the dynamics constraints in the primal CFTOC problem (4.39).

Furthermore, by using the definition of $Q_{xu,\tau}^d$, $Q_{u,\tau}^d$, $l_{u,\tau}^d$ and B_τ^p in (4.73g), (4.73f), (4.73j) and (4.73c), respectively, in the optimality condition (2.17a) for the dual problem (4.74) also gives

$$\begin{aligned} (Q_{xu,\tau}^d)^T x_\tau^d + Q_{u,\tau}^d u_\tau^d + l_{u,\tau}^d + \begin{bmatrix} 0 \\ -I_{n_{c,N-\tau}} \end{bmatrix} \mu_\tau + (B_\tau^d)^T \lambda_{\tau+1} &= 0 \iff \\ \begin{bmatrix} -\bar{Q}_{zu,t} (B_t^p)^T \\ H_{u,t}^p \bar{Q}_{u,t} (B_t^p)^T \end{bmatrix} x_\tau^d + \begin{bmatrix} \bar{Q}_{z,t} & -\bar{Q}_{zu,t} (H_{u,t}^p)^T \\ -H_{u,t}^p (\bar{Q}_{zu,t})^T & H_{u,t}^p \bar{Q}_{u,t} (H_{u,t}^p)^T \end{bmatrix} u_\tau^d + \\ \begin{bmatrix} -\bar{Q}_{z,t} l_{z,t}^p - (\bar{Q}_{zu,t})^T l_{u,t}^p \\ -h_t^p + H_{u,t}^p (\bar{Q}_{zu,t})^T l_{z,t}^p + H_{u,t}^p \bar{Q}_{u,t} l_{u,t}^p \end{bmatrix} + \begin{bmatrix} 0 \\ -I_{n_{c,t}} \end{bmatrix} \mu_\tau + \begin{bmatrix} M_t^p \\ H_{x,t}^p \end{bmatrix} \lambda_{\tau+1} &= 0. \end{aligned} \quad (4.83)$$

The lower block-equations in (4.83) gives

$$\begin{aligned}
 & H_{u,t}^p \bar{Q}_{u,t} (B_t^p)^T x_\tau^d + H_{u,t}^p \left[-(\bar{Q}_{zu,t})^T \quad \bar{Q}_{u,t} (H_{u,t}^p)^T \right] u_\tau^d + H_{u,t}^p (\bar{Q}_{zu,t})^T l_{z,t}^p + \\
 & H_{u,t}^p \bar{Q}_{u,t} l_{u,t}^p - h_t^p + H_{x,t}^p \lambda_{\tau+1} - \mu_\tau = 0 \iff H_{x,t}^p \lambda_{\tau+1} - h_t^p + \\
 & H_{u,t}^p \left(\left[-(\bar{Q}_{zu,t})^T \quad \bar{Q}_{u,t} (H_{u,t}^p)^T \right] u_\tau^d + (\bar{Q}_{zu,t})^T l_{z,t}^p + \bar{Q}_{u,t} l_{u,t}^p + \bar{Q}_{u,t} (B_t^p)^T x_\tau^d \right) = \mu_\tau.
 \end{aligned} \tag{4.84}$$

By using the expressions for x_t^p and u_t^p in (4.80) and (4.76), respectively, together with $\tau = N - t$ in (4.84) the following relation is obtained

$$-H_{x,t}^p x_t^p - H_{u,t}^p u_t^p - h_t = \mu_{N-t} \iff H_{x,t}^p x_t^p + H_{u,t}^p u_t^p + h_t = -\mu_{N-t}. \tag{4.85}$$

Hence, the dual variables μ_τ corresponding to the inequality constraints in the dual problem (4.74) can be interpreted as the slack variables for the inequality constraints in the primal problem (4.39).

5

Parallel Newton Step Computation

In recent years much effort has been spent on efficient parallel computations for MPC problems (Constantinides, 2009). In Soudbakhsh and Annaswamy (2013) an extended Parallel Cyclic Reduction algorithm is used to reduce the computation to smaller systems of equations that are solved in parallel. The computational complexity of this algorithm is reported to be $\mathcal{O}(\log N)$, where N is the prediction horizon. In Zhu and Laird (2008); Laird et al. (2011); Reuterswärd (2012) a time-splitting approach to split the prediction horizon into blocks is adopted. The subproblems in the blocks are connected through complicating variables and are solved in parallel using Schur complements. The complicating variables are computed via a consensus step where a dense system of equations involving all complicating variables has to be solved sequentially. In O'Donoghue et al. (2013) a splitting method based on Alternating Direction Method of Multipliers (ADMM) is used, where some steps of the algorithm can be computed in parallel. Stathopoulos et al. (2013) develop an iterative three-set splitting QP solver. In this method the prediction horizon is split into smaller subproblems that can be solved in parallel, and a consensus step using ADMM is performed to obtain the final solution.

In Chapter 3 it was concluded that in both AS and IP methods the main computational bottle-neck is computing the Newton step. In this chapter it will be shown how the UFTOC problem that corresponds to the Newton step can be solved in parallel on several computational units. Furthermore, it is shown that a UFTOC problem of prediction horizon N can be reduced in parallel to a new, smaller UFTOC problem in the same form but with prediction horizon $p < N$. In the algorithm proposed here the solution is computed directly without involving any iterative updates of variables. The material that is presented in this chapter is an extended version of the material presented in Nielsen and Axehill (2014). The

editing has been made to improve the presentation of the results and no technical results are added.

The parallel algorithm proposed here is related to nonserial dynamic programming methods that describe the connection between variables in the problem using tree representations, see, e.g., Bertelè and Brioschi (1973); Moallemi (2007); Shcherbina (2007). Nonserial dynamic programming share the basic ideas with serial dynamic programming, see Bertsekas (2000), but can handle more general problem structures. Unlike some nonserial methods in these references, the algorithm presented in this chapter can compute the solution exactly in parallel, and no iterative updates of estimates of the variables are computed. Furthermore, the exact solution can be computed in a finite number of steps, which can be determined a priori to the solution of the problem. The parallel algorithm proposed here and in Nielsen and Axehill (2014) is limited to solving problems in the form (3.39) or a similar form. In Khoshfetrat Pakazad et al. (2015) a more general message-passing algorithm for IP methods has later been presented which extends the results in Nielsen and Axehill (2014). This message-passing algorithm utilizes tree structure representations similar to nonserial dynamic programming and can be used to solve more general problems than (3.39).

The UFTOC problem that will be focused on in this chapter is given in the form in (3.39) with prediction horizon N , and where assumptions 3.4 and 3.5 are assumed to hold. A problem with this structure will be denoted $\mathcal{P}(N)$, i.e.,

$$\begin{aligned}
 & \underset{x,u}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^T \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \begin{bmatrix} l_{x,t} \\ l_{u,t} \end{bmatrix}^T \begin{bmatrix} x_t \\ u_t \end{bmatrix} + c_t \right) + \\
 \mathcal{P}(N) : &&& \frac{1}{2} x_N^T Q_{x,N} x_N + l_{x,N}^T x_N + c_N \\
 & \text{subject to} && x_0 = \bar{x} \\
 &&& x_{t+1} = A_t x_t + B_t u_t + a_t, \quad t \in \mathbb{Z}_{0,N-1}.
 \end{aligned} \tag{5.1}$$

Hence, using this notation, a UFTOC in the principal form (5.1) but with prediction horizon p will be denoted $\mathcal{P}(p)$, etc. This will be used to simplify notation in this chapter and in Chapter 6.

Remark 5.1. The notation $\mathcal{P}(N)$ is used to denote a family of problems where the instances have the *structure* of a UFTOC problem as the one in (5.1). Hence, two different instances, i.e., UFTOC problems with the same structure but different objective functions and constraints, are both denoted $\mathcal{P}(N)$.

The following definition of the product operator for matrices will be used

Definition 5.2. The *product operator* for a sequence of matrices $A_l, A_{l+1}, \dots, A_k$ is defined as

$$\prod_{t=l}^k A_t \triangleq \begin{cases} A_k \cdots A_{l+1} A_l, & l \leq k, \\ I, & l > k. \end{cases} \tag{5.2}$$

5.1 Problem Decomposition

The structure of the UFTOC problem (5.1) can be exploited by splitting it into smaller subproblems that only share a small number of complicating variables. Given the value of the complicating variables, the subproblems can be solved individually. These smaller subproblems are obtained by splitting the prediction horizon in $p + 1$ intervals $i \in \mathbb{Z}_{0,p}$ and introducing local variables

$$\mathbf{x}_i \triangleq \begin{bmatrix} x_{0,i} \\ \vdots \\ x_{N_i,i} \end{bmatrix}, \quad \mathbf{u}_i \triangleq \begin{bmatrix} u_{0,i} \\ \vdots \\ u_{N_i-1,i} \end{bmatrix}, \quad (5.3)$$

and initial constraints $x_{0,i} = \hat{x}_i$ for each subproblem, and terminal constraints $x_{N_i,i} = d_i$ for $i \in \mathbb{Z}_{0,p-1}$. Then, each interval is of length N_i and the relation $x_{N_i,i} = x_{0,i+1}$ for $i \in \mathbb{Z}_{0,p-1}$ holds. Here, d_i is only used as a dummy variable in an intermediate step. The connections between the subproblems, i.e., $x_{N_i,i} = x_{0,i+1}$ are introduced as the coupling constraints $\hat{x}_{i+1} = d_i$ for $i \in \mathbb{Z}_{0,p-1}$. For notational aspects, and without loss of generality, d_i is parametrized as $d_i = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i$, where $\hat{x}_i \in \mathbb{R}^{n_x}$ and $\hat{u}_i \in \mathbb{R}^{n_u}$ with $n_{\hat{u}} \leq n_x$ are the complicating variables. How to choose the parametrization $\hat{A}_i$, $\hat{B}_i$ and $\hat{a}_i$ will soon be shown. Let $x_{t,i}$ and $u_{t,i}$ denote the local states and control inputs in subproblem i . Then, an extended optimization problem that is equivalent to the UFTOC problem (5.1) can be constructed as

$$\begin{aligned} \underset{\hat{x}, \hat{u}, \mathbf{x}_i, \mathbf{u}_i}{\text{minimize}} \quad & \sum_{i=0}^p \sum_{t=0}^{N_i-1} \left(\frac{1}{2} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix}^T \underbrace{\begin{bmatrix} Q_{x,t,i} & Q_{xu,t,i} \\ Q_{xu,t,i}^T & Q_{u,t,i} \end{bmatrix}}_{\triangleq Q_{t,i}} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix} + \underbrace{\begin{bmatrix} l_{x,t,i} \\ l_{u,t,i} \end{bmatrix}^T}_{\triangleq l_{t,i}} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix} + c_{t,i} \right) + \\ & \frac{1}{2} x_{N_p,p}^T Q_{N_p,p} x_{N_p,p} + l_{N_p,p}^T x_{N_p,p} + c_{N_p,p} \\ \text{subject to} \quad & \hat{x}_0 = \bar{x}_0 \\ & \forall i \in \mathbb{Z}_{0,p} \begin{cases} x_{0,i} = \hat{x}_i \\ x_{t+1,i} = A_{t,i} x_{t,i} + B_{t,i} u_{t,i} + a_{t,i}, t \in \mathbb{Z}_{0,N_i-1} \\ x_{N_i,i} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, i \neq p \\ \hat{x}_{i+1} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, i \in \mathbb{Z}_{0,p-1}, \end{cases} \end{aligned} \quad (5.4)$$

where the variables $Q_{t,i}$, $l_{t,i}$, $c_{t,i}$, $A_{t,i}$, $B_{t,i}$ and $a_{t,i}$ are defined conformly. Note that the initial state $\hat{x}_0$ is equal to the initial state of the original problem (5.1). The word "extended" is used here to indicate that the UFTOC problem (5.1) is extended with both extra variables and constraints to form (5.4).

The introduction of the terminal constraints might result in infeasible subproblems. Hence, in order to obtain feasible subproblems in (5.4), the parametrizations of the terminal constraints must be chosen such that it is possible to satisfy these constraints. To do this, the dynamics constraints in subproblem i can be

used to compute the final state in subproblem i given the control signals u_i and the initial state $\hat{x}_i$ as

$$x_{N_i,i} = \mathcal{A}_i \hat{x}_i + \mathcal{S}_i u_i + \mathcal{D}_i a_i, \quad (5.5)$$

where

$$\mathcal{A}_i \triangleq \prod_{t=0}^{N_i-1} A_{t,i}, \quad \mathcal{D}_i \triangleq \left[\prod_{t=1}^{N_i-1} A_{t,i} \quad \cdots \quad A_{N_i-1,i} \quad I \right], \quad (5.6)$$

$$\mathcal{S}_i \triangleq \left[\prod_{t=1}^{N_i-1} A_{t,i} B_{0,i} \quad \cdots \quad A_{N_i-1,i} B_{N_i-2,i} \quad B_{N_i-1,i} \right], \quad (5.7)$$

and a_i and u_i are the stacked $a_{t,i}$ and $u_{t,i}$ for $t \in \mathbb{Z}_{0,N_i-1}$, respectively. The product operator is defined in Definition 5.2. The feasibility of the subproblem can be ensured by a careful selection of the parametrization of the problem. Here, this is performed by requiring that the final state satisfies the terminal constraint $x_{N_i,i} = d_i = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i$, where d_i is in the reachable subspace given by $\mathcal{A}_i$, $\mathcal{S}_i$ and $\mathcal{D}_i a_i$. This can be assured by requiring

$$\hat{A}_i \triangleq \mathcal{A}_i, \quad \hat{B}_i \triangleq \mathcal{T}_i, \quad \hat{a}_i \triangleq \mathcal{D}_i a_i, \quad (5.8)$$

where the columns of $\mathcal{T}_i$ form a basis for the range of $\mathcal{S}_i$.

Remark 5.3. The terminal state $x_{N_i,i}$ in the subproblems $i \in \mathbb{Z}_{0,p-1}$ can be discarded, and the two constraints

$$\begin{cases} x_{N_i,i} = A_{N_i-1,i} x_{N_i-1,i} + B_{N_i-1,i} u_{N_i-1,i} + a_{N_i-1,i}, \\ x_{N_i,i} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, \end{cases} \quad (5.9)$$

can be combined into a single constraint

$$\hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i = A_{N_i-1,i} x_{N_i-1,i} + B_{N_i-1,i} u_{N_i-1,i} + a_{N_i-1,i}, \quad (5.10)$$

and the corresponding parts in the objective function are discarded. However, for notational purposes the terminal state $x_{N_i,i}$ will be used as a local variable in the subproblems.

For $i \in \mathbb{Z}_{0,p-1}$ the individual subproblems in (5.4) are given by

$$\begin{aligned} & \underset{x_i, u_i}{\text{minimize}} && \sum_{t=0}^{N_i-1} \left(\frac{1}{2} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix}^T \begin{bmatrix} Q_{x,t,i} & Q_{xu,t,i} \\ Q_{xu,t,i}^T & Q_{u,t,i} \end{bmatrix} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix} + \begin{bmatrix} l_{x,t,i} \\ l_{u,t,i} \end{bmatrix}^T \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix} + c_{t,i} \right) \\ & \text{subject to} && x_{0,i} = \hat{x}_i \\ & && x_{t+1,i} = A_{t,i} x_{t,i} + B_{t,i} u_{t,i} + a_{t,i}, \quad t \in \mathbb{Z}_{0,N_i-1} \\ & && x_{N_i,i} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i. \end{aligned} \quad (5.11)$$

Here i is the index of the subproblem. The last subproblem $i = p$ is on the same form as in (5.11) but with the additional term

$$\frac{1}{2} x_{N_p,p}^T Q_{N_p,p} x_{N_p,p} + l_{N_p,p}^T x_{N_p,p} + c_{N_p,p}, \quad (5.12)$$

in the objective function, and no terminal constraint on $x_{N_p,p}$, i.e.,

$$\begin{aligned}
 & \underset{x_p, u_p}{\text{minimize}} && \sum_{t=0}^{N_p-1} \left(\frac{1}{2} \begin{bmatrix} x_{t,p} \\ u_{t,p} \end{bmatrix}^T \begin{bmatrix} Q_{x,t,p} & Q_{xu,t,p} \\ Q_{xu,t,p}^T & Q_{u,t,p} \end{bmatrix} \begin{bmatrix} x_{t,p} \\ u_{t,p} \end{bmatrix} + \begin{bmatrix} l_{x,t,p} \\ l_{u,t,p} \end{bmatrix}^T \begin{bmatrix} x_{t,p} \\ u_{t,p} \end{bmatrix} + c_{t,p} \right) + \\
 & && \frac{1}{2} x_{N_p,p}^T Q_{N_p,p} x_{N_p,p} + l_{N_p,p}^T x_{N_p,p} + c_{N_p,p} \\
 & \text{subject to} && x_{0,p} = \hat{x}_p \\
 & && x_{t+1,p} = A_{t,p} x_{t,p} + B_{t,p} u_{t,p} + a_{t,p}, \quad t \in \mathbb{Z}_{0,N_p-1}.
 \end{aligned} \tag{5.13}$$

Temporarily excluding details, each subproblem (5.11) can be solved parametrically and the solution to each subproblem is a function of the complicating variables $\hat{x}_i$ and $\hat{u}_i$, and the parametric solution to subproblem (5.13) is a function of $\hat{x}_p$. Furthermore, the value function $\hat{V}_i(\hat{x}_i, \hat{u}_i)$ for each subproblem is also a function of the complicating variables. By inserting these parametric solutions of all subproblems in (5.4) and using the coupling constraints between the subproblems, the extended problem (5.4) can be reduced to the master problem

$$\begin{aligned}
 & \underset{\bar{x}, \bar{u}}{\text{minimize}} && \sum_{i=0}^{p-1} \left(\frac{1}{2} \begin{bmatrix} \hat{x}_i \\ \hat{u}_i \end{bmatrix}^T \underbrace{\begin{bmatrix} \hat{Q}_{x,i} & \hat{Q}_{xu,i} \\ \hat{Q}_{xu,i}^T & \hat{Q}_{u,i} \end{bmatrix}}_{\triangleq \hat{Q}_i} \begin{bmatrix} \hat{x}_i \\ \hat{u}_i \end{bmatrix} + \underbrace{\begin{bmatrix} \hat{l}_{x,i} \\ \hat{l}_{u,i} \end{bmatrix}^T}_{\triangleq \hat{l}_i} \begin{bmatrix} \hat{x}_i \\ \hat{u}_i \end{bmatrix} + \hat{c}_i \right) + \\
 & \mathcal{P}(p) : && \frac{1}{2} \hat{x}_p^T \hat{Q}_p \hat{x}_p + \hat{l}_p^T \hat{x}_p + \hat{c}_p \\
 & \text{subject to} && \hat{x}_0 = \bar{x}_0 \\
 & && \hat{x}_{i+1} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, \quad i \in \mathbb{Z}_{0,p-1}.
 \end{aligned} \tag{5.14}$$

Here $\hat{A}_i$, $\hat{B}_i$, $\hat{a}_i$, $\hat{Q}_i$, $\hat{l}_i$ and $\hat{c}_i$ are computed in each subproblem i and represent the constraints and the value function in each subproblem, respectively. The dynamics constraints in the master problem are given by the coupling constraints between the subproblems in (5.4). This new UFTOC problem is on the same form as the original UFTOC problem (5.1), but with prediction horizon $p < N$. The reduction of the problem is summarized in Theorem 5.4 and is graphically depicted in Figure 5.1, where the dotted lines represent repetition of the structure. This approach is similar to primal decomposition which is described in Lasdon (1970); Boyd et al. (2008), where the $p + 1$ subproblems share complicating variables $\hat{x}_i$ and $\hat{u}_i$ that are generally computed iteratively. In the work presented in this chapter it is shown that the master problem that is solved to compute the complicating variables is in the form of a UFTOC problem. Hence the complicating variables are determined directly by solving the new, reduced UFTOC problem at the upper level in Figure 5.1 directly (non-iteratively). Inserting the optimal $\hat{x}_i$ and $\hat{u}_i$ into the subproblems given by (5.11) and $\hat{x}_p$ into the problem (5.13) gives the solution to (5.1).

Figure 5.1: The parameters $\hat{x}_i$ and $\hat{u}_i$ in each subproblem i can be interpreted as new state and control variables in the reduced UFTOC problem with prediction horizon p . The value functions $\hat{V}_i(\hat{x}_i, \hat{u}_i)$ are the terms in the new objective function.

Theorem 5.4. Consider an optimization problem $\mathcal{P}(N)$ defined in (5.1). Then $\mathcal{P}(N)$ can be reduced to $\mathcal{P}(p)$ in parallel, where $1 \leq p < N$. Furthermore, the primal and dual optimal solutions x^* , u^* and λ^* to $\mathcal{P}(N)$ can be computed in parallel from the primal and dual solutions $\hat{x}^*$, $\hat{u}^*$ and $\hat{\lambda}^*$ to $\mathcal{P}(p)$.

Proof: For the proof of Theorem 5.4, see Appendix 5.A. □

In the remainder of this section it will be shown how the subproblems (5.11) and (5.13) are solved parametrically and how the matrices in (5.14) are computed.

5.1.1 Solution of the subproblems

In this section, it will be shown that each subproblem $i \in \mathbb{Z}_{0,p}$ can be solved parametrically and that the solution can be expressed as a function of the complicating variables $\hat{x}_i$ and $\hat{u}_i$ for $i \in \mathbb{Z}_{0,p-1}$ and $\hat{x}_p$ for $i = p$.

For now it is assumed that LICQ holds for (5.11). By defining the parameter vector θ_i as

$$\theta_i \triangleq \begin{bmatrix} \hat{x}_i \\ \hat{u}_i \end{bmatrix}, \quad i \in \mathbb{Z}_{0,p-1}, \quad \theta_p \triangleq \hat{x}_p, \quad (5.15)$$

the optimization problem (5.11) can be cast in a more compact form

$$\begin{aligned} & \underset{X_i}{\text{minimize}} && \frac{1}{2} X_i^T Q_i X_i + l_i^T X_i + c_i \\ & \text{subject to} && A_i X_i = b_i + G_i \theta_i, \end{aligned} \quad (5.16)$$

where the optimization variables X_i and the corresponding matrices are defined as

$$X_i \triangleq \begin{bmatrix} x_{0,i} \\ u_{0,i} \\ \vdots \\ u_{N_i-1,i} \\ x_{N_i,i} \end{bmatrix}, \quad Q_i \triangleq \begin{bmatrix} Q_{0,i} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & Q_{N_i-1,i} & 0 \\ 0 & \cdots & 0 & 0 \end{bmatrix}, \quad l_i \triangleq \begin{bmatrix} l_{0,i} \\ \vdots \\ l_{N_i-1,i} \\ 0 \end{bmatrix}, \quad c_i \triangleq \sum_{t=0}^{N_i-1} c_{t,i}, \quad (5.17a)$$

$$A_i \triangleq \begin{bmatrix} -I & 0 & \dots & & \dots & 0 \\ A_{0,i} & B_{0,i} & -I & 0 & \dots & \vdots \\ 0 & 0 & A_{1,i} & \dots & & \\ \vdots & \vdots & & \ddots & & 0 \\ \vdots & \vdots & & & A_{N_i-1,i} & B_{N_i-1,i} & -I \\ 0 & \dots & & \dots & 0 & -I \end{bmatrix}, \quad (5.17b)$$

$$b_i \triangleq \begin{bmatrix} 0 \\ -a_{0,i} \\ \vdots \\ -a_{N_i-1,i} \\ -\hat{a}_i \end{bmatrix}, \quad G_i \triangleq \begin{bmatrix} -I & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ -\hat{A}_i & -\hat{B}_i \end{bmatrix}. \quad (5.17c)$$

Remark 5.5. For a subproblem where LICQ holds, $\hat{A}_i = 0$, $\hat{B}_i = I$ and $\hat{a}_i = 0$ in the parametrization of the terminal constraint are valid choices since S_i has full row rank. _____

The dual variables λ_i corresponding to the equality constraints in (5.11) are introduced as

$$\lambda_i \triangleq \begin{bmatrix} \lambda_{0,i} \\ \lambda_{1,i} \\ \vdots \\ \lambda_{N_i,i} \\ \lambda_{tc,i} \end{bmatrix}, \quad (5.18)$$

where

$$\lambda_{0,i} \leftrightarrow -x_{0,i} + \hat{x}_i = 0, \quad (5.19a)$$

$$\lambda_{t+1,i} \leftrightarrow -x_{t+1,i} + A_{t,i}x_{t,i} + B_{t,i}u_{t,i} + a_{t,i} = 0, \quad t \in \mathbb{Z}_{0,N_i-1}, \quad (5.19b)$$

$$\lambda_{tc,i} \leftrightarrow -x_{N_i,i} + \hat{A}_i\hat{x}_i + \hat{B}_i\hat{u}_i + \hat{a}_i = 0. \quad (5.19c)$$

The symbol $\leftrightarrow$ is introduced in Definition 3.8. The subscript "tc" is an abbreviation of "terminal constraint".

Note that (5.16) is a very simple multiparametric quadratic programming problem with parameters θ_i and only equality constraints. Hence the optimal primal and dual solution to this problem are both affine functions of the parameters θ_i (Tøndel et al., 2003).

Remark 5.6. Since the simple parametric programming problem (5.16) is only subject to equality constraints, it is not *piecewise* affine in the parameters. Hence, the solution to this equality constrained problem can be computed cheaply and it does not suffer from the complexity issues of a general multiparametric programming problem. _____

Since LICQ is assumed to hold, the unique optimal primal and dual solution can be expressed as

$$X_i^*(\theta_i) = K_i^x \theta_i + k_i^x, \quad (5.20)$$

$$\lambda_i^*(\theta_i) = K_i^\lambda \theta_i + k_i^\lambda, \quad (5.21)$$

for some K_i^x , k_i^x , K_i^λ and k_i^λ (Tøndel et al., 2003). Here i denotes the index of the subproblem. The value function of (5.11) expressed in the parameter θ_i is obtained by inserting the parametric primal optimal solution (5.20) into the objective function in (5.16), resulting in

$$\hat{V}_i(\theta_i) = \frac{1}{2} \theta_i^T \hat{Q}_i \theta_i + \hat{l}_i^T \theta_i + \hat{c}_i, \quad (5.22)$$

where

$$\hat{Q}_i \triangleq (K_i^x)^T Q_i K_i^x, \quad (5.23a)$$

$$\hat{l}_i \triangleq (K_i^x)^T l_i + (K_i^x)^T Q_i k_i^x, \quad (5.23b)$$

$$\hat{c}_i \triangleq c_i + \frac{1}{2} (k_i^x)^T Q_i k_i^x + l_i^T k_i^x. \quad (5.23c)$$

The last subproblem given by (5.13) is different from the p first subproblems since there is no terminal constraint on $x_{N_p, p}$. Hence the parametric solution of this subproblem only depends on the initial state $\hat{x}_p$, and λ_p in (5.18) and Q_p , l_p , c_p , A_p , b_p and G_p in (5.17) are modified accordingly. The derivation of the solution is analogous to the one for the subproblems $i \in \mathbb{Z}_{0, p-1}$, but with $\theta_p = \hat{x}_p$. The unique optimal primal and dual solutions to (5.13) are hence given by (5.20) and (5.21), and the value function is given by (5.22) (all with $i = p$).

5.1.2 Solution of a primal degenerate subproblem

The terminal constraint in a subproblem given by (5.11) introduces n_x new constraints, which might result in violation of LICQ for the subproblem even though this is not the case in the original UFTOC problem (5.1). According to Definition 2.12, violation of LICQ is known as primal degeneracy and the dual variables for a primal degenerate problem are non-unique (Tøndel et al., 2003). In this section it will be shown how to choose dual variables in the subproblems that satisfy the optimality conditions of the original UFTOC problem (5.1).

Remark 5.7. If the subproblem violates LICQ when $x_{N_i, i}$ is used as a local variable in (5.11), it violates LICQ even when $x_{N_i, i}$ is excluded as was mentioned in Remark 5.3 and the constraints are combined as in (5.10). _____

The optimal parametric primal solution to a primal degenerate problem in the form (5.16) is given by (5.20) and the optimal dual solution is given by

$$\lambda_i^*(\theta_i) = K_i^\lambda \theta_i + k_i^\lambda + \lambda_i^{\mathcal{N}}, \quad (5.24)$$

where $\lambda_i^{\mathcal{N}} \in \mathcal{N}(A_i^T)$ (Tøndel et al., 2003). Here $\mathcal{N}(A_i^T)$ is the nullspace of A_i^T ,

and the expression for $\mathcal{N}(A_i^T)$ is given by Lemma 5.8.

Lemma 5.8. *The nullspace of A_i^T is given by*

$$\mathcal{N}(A_i^T) \triangleq \{z \mid z = Z_i w_i, \forall w_i \in \mathcal{N}(S_i^T)\}, \quad (5.25)$$

where

$$Z_i \triangleq \begin{bmatrix} -\hat{A}_i^T \\ -\mathcal{D}_i^T \\ I \end{bmatrix}, \quad (5.26)$$

and $\hat{A}_i$ and $\mathcal{D}$ are defined in (5.6) and S_i is the reachability matrix defined in (5.7).

Proof: For the proof of Lemma 5.8, see Appendix 5.B. □

Remark 5.9. Note that Z_i is computed cheaply since the matrices $\hat{A}_i$ and $\mathcal{D}_i$ are already computed. □

The dual variables of the extended problem (5.4) are introduced by (5.19) for each subproblem, and by

$$\hat{\lambda}_0 \leftrightarrow -\hat{x}_0 + \bar{x}_0 = 0, \quad (5.27a)$$

$$\hat{\lambda}_{i+1} \leftrightarrow -\hat{x}_{i+1} + \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i = 0, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.27b)$$

for the coupling constraints that connect the subproblems in (5.4). Note that $\lambda_{tc,i}$ in (5.19c) is the dual variable corresponding to the terminal constraint in each subproblem, whereas $\hat{\lambda}_{i+1}$ in (5.27b) are the dual variables corresponding to the coupling constraints between the subproblems (interpreted as the dynamics constraints in the master UFTOC problem (5.14)). Hence, $\lambda_{tc,i}$ is computed in the subproblem, and $\hat{\lambda}_i$ is computed when (5.14) is solved. This is depicted in Figure 5.2 where the upper level corresponds to the master UFTOC problem (5.14) and the lower level to problem (5.4) (which is equivalent to $\mathcal{P}(N)$).

The purpose in this chapter is to compute a solution to the original UFTOC problem (5.1) by computing solutions to the $p + 1$ subproblems given in the form in (5.11) and (5.13). However, for primal degenerate subproblems the dual solutions are non-unique (Tøndel et al., 2003), and it is necessary to choose dual solutions to the subproblems that satisfy the optimality conditions of the original UFTOC problem (5.1). To do this, the relations between the dual variables of different subproblems are exploited and it is shown that it is possible to select the dual solution such that this is achieved. The relation between the dual variables in different subproblems is given by Theorem 5.10 and Corollary 5.11.

Figure 5.2: The dual variables $\hat{\lambda}_i$ in the reduced problem are related to the dual variables $\lambda_{tc,i}$ and $\lambda_{0,i+1}$ in the subproblems.

Theorem 5.10. Consider an optimization problem in the form in (5.4). Let the dual variables be defined by (5.19) and (5.27). Then, the relations between the optimal dual solutions in different subproblems are given by

$$\lambda_{0,p} = \hat{\lambda}_p, \quad (5.28a)$$

$$\lambda_{0,i} = \hat{\lambda}_i - \hat{A}_i^T (\lambda_{tc,i} + \hat{\lambda}_{i+1}), \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.28b)$$

$$\hat{B}_i^T (\lambda_{tc,i} + \hat{\lambda}_{i+1}) = 0, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.28c)$$

$$\lambda_{N_i,i} = -\lambda_{tc,i}, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.28d)$$

where $\hat{A}_i$ and $\hat{B}_i$ are defined in (5.8).

Proof: For the proof of Theorem 5.10, see Appendix 5.C. □

Corollary 5.11. Let the assumptions in Theorem 5.10 be satisfied, and let LICQ hold for all subproblems $i \in \mathbb{Z}_{0,p}$. Then the optimal dual variables in the subproblems are unique and the relations between the dual solutions in the subproblems are given by

$$\lambda_{0,i} = \hat{\lambda}_i, \quad i \in \mathbb{Z}_{0,p}, \quad (5.29a)$$

$$\lambda_{tc,i} = -\hat{\lambda}_{i+1} = -\lambda_{0,i+1}, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.29b)$$

$$\lambda_{N_i,i} = -\lambda_{tc,i} = \lambda_{0,i+1}, \quad i \in \mathbb{Z}_{0,p-1}. \quad (5.29c)$$

Proof: Let LICQ hold for all subproblems $i \in \mathbb{Z}_{0,p}$ in (5.4). Then $\mathcal{N}(A_i^T) = \emptyset$, $i \in \mathbb{Z}_{0,p}$ and the dual solutions are unique. Furthermore,

$$\text{rank}(\mathcal{S}_i) = n_x \iff \mathcal{T}_i = \hat{B}_i \in \mathbb{R}^{n_x \times n_x} \text{ non-singular} \stackrel{(5.28c)}{\implies} \lambda_{tc,i} = -\hat{\lambda}_{i+1}. \quad (5.30)$$

Inserting (5.30) into (5.28b) and using (5.28a) gives $\lambda_{0,i} = \hat{\lambda}_i$, $i \in \mathbb{Z}_{0,p}$, which by also inserting (5.30) into (5.28d) gives (5.29c). □

Lemma 5.8 is used to choose the nullspace element $\lambda_i^{\mathcal{N}}$ in (5.24) to obtain the correct dual solution for subproblem i . According to the lemma, $\lambda_i^{\mathcal{N}}$ can be com-

puted as $\lambda_i^{\mathcal{N}} = Z_i w_i$, $w_i \in \mathcal{N}(\mathcal{S}_i^T)$, giving the optimal dual variable for subproblem i as

$$\lambda_i^*(\theta_i, w_i) = K_i^\lambda \theta_i + k_i^\lambda + Z_i w_i, \quad w_i \in \mathcal{N}(\mathcal{S}_i^T). \quad (5.31)$$

Let $\xi_i = K_i^\lambda \theta_i + k_i^\lambda$ be the dual solution when the minimum norm nullspace element is selected, and let $\hat{\lambda}_i$ be given by the dual solution to problem (5.14). Then it follows from Theorem 5.10 that

$$\xi_{0,i} = \hat{\lambda}_i - \hat{A}_i^T (\xi_{tc,i} + \hat{\lambda}_{i+1}), \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.32a)$$

$$\hat{B}_i^T (\xi_{tc,i} + \hat{\lambda}_{i+1}) = 0, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.32b)$$

$$\xi_{N,i} = -\xi_{tc,i}, \quad i \in \mathbb{Z}_{0,p-1}. \quad (5.32c)$$

To obtain a dual solution λ_i which satisfies the optimality conditions for the original problem (5.1), the freedom in the choice of the dual variables from (5.31) is exploited, i.e., the dual variable is chosen as

$$\lambda_i = \xi_i + Z_i w_i. \quad (5.33)$$

Theorem 5.10 certifies that the relation $\lambda_{N,i} = \hat{\lambda}_{i+1} = \lambda_{0,i+1}$ holds if $\lambda_{tc,i} = -\hat{\lambda}_{i+1}$ holds. This is the same relation as in the non-degenerate case and, hence, satisfies the optimality conditions of (5.1). The last block in (5.26) and (5.33) give $\lambda_{tc,i} = \xi_{tc,i} + w_i$ and hence, to obtain $\lambda_{tc,i} = -\hat{\lambda}_{i+1}$, w_i is chosen as

$$w_i = -(\xi_{tc,i} + \hat{\lambda}_{i+1}) \in \mathcal{N}(\mathcal{S}_i^T) \Rightarrow \lambda_{tc,i} = -\hat{\lambda}_{i+1}. \quad (5.34)$$

Note that (5.32b) gives that $w_i = -(\xi_{tc,i} + \hat{\lambda}_{i+1}) \in \mathcal{N}(\hat{B}_i^T) = \mathcal{N}(\mathcal{S}_i^T)$. By using this choice of w_i in the optimal dual solution (5.31) together with (5.26), (5.32a) and (5.32c) the following hold

$$\lambda_{0,i} = \xi_{0,i} + \lambda_{0,i}^{\mathcal{N}} \stackrel{(5.26)}{=} \xi_{0,i} - \hat{A}_i^T w_i \stackrel{(5.32a)}{=} \hat{\lambda}_i, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.35)$$

$$\lambda_{N,i} = \xi_{N,i} + \lambda_{N,i}^{\mathcal{N}} \stackrel{(5.26) \& (5.32c)}{=} -\xi_{tc,i} - w_i \stackrel{(5.34)}{=} \hat{\lambda}_{i+1}, \quad i \in \mathbb{Z}_{0,p-1}. \quad (5.36)$$

Hence, the dual solution of subproblem i satisfies the optimality conditions for the UFTOC problem (5.1) if it is computed as

$$\lambda_i^*(\theta_i, \hat{\lambda}_{i+1}) = K_i^\lambda \theta_i + k_i^\lambda - Z_i(\xi_{tc,i} + \hat{\lambda}_{i+1}) = \xi_i - Z_i(\xi_{tc,i} + \hat{\lambda}_{i+1}). \quad (5.37)$$

The dual solution to the original problem can be retrieved from (5.37) for $i \in \mathbb{Z}_{0,p-1}$ and (5.21) for $i = p$ (for which LICQ always holds).

5.2 Parallel Computation of Newton Step

In Section 5.1 it was shown that a UFTOC problem $\mathcal{P}(N)$ can be solved by splitting it into subproblems and reducing it to a master UFTOC problem $\mathcal{P}(p)$ with $p < N$, and then solve this master problem $\mathcal{P}(p)$ to determine the optimal parameters of the subproblems. The master problem $\mathcal{P}(p)$ can be solved using any suitable method for equality constrained QPs. However, a key feature here is that Theo-

Figure 5.3: The tree structure obtained when the UFTOC problems are reduced in several steps. Each level in the tree forms a UFTOC problem that is again split into several smaller problems. $\mathcal{P}_i^k(N_i^k)$ denotes a subproblem in the form in (5.11) with prediction horizon N_i^k .

rem 5.4 guarantees that the master problem is a UFTOC problem in the form (5.1) with prediction horizon p . Hence, instead of solving $\mathcal{P}(p)$ completely, it can be reduced again to a UFTOC problem with even shorter prediction horizon. In this section it will be shown how Theorem 5.4 can be used repeatedly to obtain a problem structure that can be solved in parallel.

Each reduction has the principal structure as is shown in Figure 5.1, and repeatedly reducing the problems thus creates the tree structure that can be seen in Figure 5.3. Let the UFTOC problem at level k have prediction horizon p_{k-1} and be denoted $\mathcal{P}(p_{k-1})$, and let $\mathcal{P}_i^k(N_i^k)$ be the i :th subproblem in the form (5.11) (or (5.13)) with prediction horizon N_i^k at level k , where $i \in \mathbb{Z}_{0,p_k}$. Then each level in the tree is a UFTOC problem, and the nodes are its corresponding subproblems. The top node $\mathcal{P}_0^m(N_0^m)$ is not reduced further, but instead solved completely, and the leaves at the bottom of the tree are the subproblems of the original UFTOC problem $\mathcal{P}(N)$.

The problem $\mathcal{P}(p_{k-1})$ is reduced to its master problem $\mathcal{P}(p_k)$ by solving all subproblems $\mathcal{P}_i^k(N_i^k)$, $i \in \mathbb{Z}_{0,p_k}$ parametrically according to Section 5.1. Since all of the subproblems $\mathcal{P}_i^k(N_i^k)$ are independent and only communicates with its parent in the tree in Figure 5.3, this can be done in parallel. While repeatedly reducing the UFTOC problems, i.e., moving up in the tree in Figure 5.3, the parametric solution that is communicated to its parent consists of the value of the objective function of each subproblem as a function of the complicating variables, and the dynamics equations from the initial state to the terminal state.

The reduction of the UFTOC problem is continued until the top problem, which has a pre-determined minimum prediction horizon, is obtained. The size of this problem is pre-defined by the user, and hence also the maximum number of levels in the tree is known a priori to the solution. When the top node has been solved, the optimal parameter values are propagated down in the tree to its children. Hence, the original problem $\mathcal{P}(N)$ is solved by first reducing the problem in m steps until $\mathcal{P}(p_{m-1})$ is obtained, and thereafter propagating the solution of $\mathcal{P}(p_{m-1})$ down to level $k = 0$. Since information is exchanged between parents

and children only, the optimal solution to each $\mathcal{P}_i^k(N_i^k)$ can be computed individually from the other subproblems at each level k , meaning that this can be done completely in parallel for each level. The solution to $\mathcal{P}(N)$ can thus be computed directly in parallel without involving any iterative step. Note however that m a priori known and user defined reduction steps and m propagating steps have to be performed for each time the UFTOC problem is solved.

Remark 5.12. At each level k in the tree in Figure 5.3 the complicating variables for level $k-1$ are computed. Hence, the consensus step to decide the complicating variables at each level is done in one iteration, and it is not necessary to iterate to get consensus between the subproblems as in many other parallel methods. _____

The theory presented in this chapter is summarized in algorithms 11 and 12. The algorithms can be used to compute the Newton step, which is defined by the solution to (5.1), in parallel. The computations can be performed using several computational units, and the level of parallelism can be tuned to fit the hardware, e.g., the number of processing units, memory capacity, bus speed and more. The level of parallelism is decided by adjusting the number of subproblems at each level in the tree in Figure 5.3. Similar ideas as is used in partial condensing, introduced in Axehill (2015), can be used to decide the sizes of the subproblems.

5.2.1 Algorithms for parallel Newton step computation

The algorithm for solving the UFTOC problem $\mathcal{P}(N)$ in parallel is based on two major steps; reduce the UFTOC problem in several steps according to the tree structure in Figure 5.3, followed by a propagation of the solution from the top level downwards in the tree to the bottom level. In both steps standard parallel numerical linear algebra can be used to further parallelize for example matrix multiplications, backward and forward substitutions and factorizations. Here, the focus is on parallelization using the inherent structure of the UFTOC problem (5.1), and the scope will be limited to this in the thesis.

The first step is summarized in Algorithm 11. Since all subproblems at the same level are independent of each other, the parfor-loop on lines 6-10 in Algorithm 11 can be performed in parallel on different computational units. Let $p_{\max}$ be the maximum number of subproblems at any level in the tree. Then, if there are $p_{\max}$ computational units available, all subproblems $\mathcal{P}_i^k(N_i^k)$ at level k can be solved simultaneously. The size of the subproblems can be adjusted such that the maximum number of subproblems at the bottom level is equal to the number of available computational units.

The second step is to propagate the solution from the top problem down in the tree until the subproblems at the bottom level are solved. The propagation of the solution is summarized in Algorithm 12. Since all subproblems in the tree only use information from their parents, the parfor-loop at lines 4-13 in Algorithm 12 can be computed in parallel. As for the first step, if there is one computational unit for each subproblem, all problems at each level in the tree can be solved simultaneously.

Algorithm 11 Parallel reduction of UFTOC problem

```

1: Set the maximum level number  $m$ 
2: Set the number of subproblems  $p_k + 1$  for each level  $k \in \mathbb{Z}_{0,m-1}$ 
3: for  $k = 0, \dots, m - 1$  do
4:   Split the prediction horizon into  $p_k + 1$  segments  $0, \dots, N_0^k$  up to  $0, \dots, N_{p_k}^k$ 
5:   Create subproblems  $i = 0, \dots, p_k$  for each time block
6:   parfor  $i = 0, \dots, p_k$  do
7:     Solve subproblem  $i$  parametrically and store  $K_i^x, k_i^x, K_i^\lambda$  and  $k_i^\lambda$ 
8:     Compute  $\hat{A}_i, \hat{B}_i, \hat{a}_i, \hat{Q}_i, \hat{l}_i$  and  $\hat{c}_i$  for the next level
9:     Compute and store  $Z_i$ 
10:  end parfor
11: end for

```

Algorithm 12 Parallel propagation of solution

```

1: Get maximum level number  $m$  from Algorithm 11
2: Solve the top problem  $\mathcal{P}_0^m(N_0^m)$  with  $x_0^m := \bar{x}$ 
3: for  $k = m - 1, \dots, 0$  do
4:   parfor  $i = 0, \dots, p_k$  do
5:     Compute primal solution given by (5.20)
6:     Compute dual solution given by (5.21)
7:     if Primal degenerate subproblem
8:       Select the dual solution according to (5.37)
9:     end if
10:    if  $k > 0$ 
11:      Send parameters  $\theta_{t,i}^{k-1}$  to each children at level  $k - 1$ 
12:    end if
13:  end parfor
14: end for

```

So far no assumptions on the length of the prediction horizon of each subproblem has been made. If, however, the length of each subproblem is fixed to N_s , and the prediction horizon of the original problem is chosen as $N = N_s^{m+1}$ for simplicity, then the tree will get $m + 1$ levels. Furthermore, assume that N_s^m processors are available. Then, since $m = \log_{N_s}(N) - 1$, the computational complexity grows logarithmically in the prediction horizon, i.e. as $\mathcal{O}(\log N)$. The optimal length N_s of the subproblems can be adjusted to fit the hardware which the algorithms are implemented on. Depending on for example the number of computational units, the available memory and the communication overhead between processors, the size of N_s can be chosen differently. How to optimally adjust the size of the subproblems is not investigated here, but it could possibly be done by for example benchmarking the algorithm using the desired hardware and mapping the performance versus different sizes of the subproblems.

Remark 5.13. Since the tree structure in Figure 5.3 is pre-determined by the user, the major parts of lines 4 and 5 in Algorithm 11 can be done prior to the solution. _____

Remark 5.14. The sizes of different subproblems in Figure 5.3 can be different, both within the same level and between levels. Hence, depending on for example the communication layout on the computational cluster, different sizes of the subproblems can be used to reduce the total computation time. _____

Remark 5.15. If the parallel algorithm is used to compute UFTOC problems in an AS method, the eliminated dual variables must be computed as in Section 3.5 using Algorithm 7. Given the primal and dual solution to the UFTOC problem, it can be seen that this algorithm can be computed in parallel for the independent intervals $i \in \mathbb{Z}_{0,p}$. _____

5.3 Numerical Results

The proposed algorithm for computing the Newton step in parallel has been implemented in MATLAB and used to solve random stable UFTOC problems in the form (5.1). The algorithm has been implemented serially, and the parallel computation times are simulated by summing over the maximum solution time at each level in the tree. Hence, memory and communication delays have not been addressed here. This approximation is investigated and commented in more detail in Chapter 6. In the implemented algorithm K_i^x , k_i^x , K_i^λ and k_i^λ are computed using the methods proposed in Tøndel et al. (2003). The implementation faced some numerical issues when unstable LTI systems were used. Hence, stable systems have been used to evaluate performance.

Remark 5.16. It is possible to use prestabilization to obtain stable LTI systems even when the original LTI system is unstable. _____

Remark 5.17. Here the structure in the subproblems is not exploited when computing K_i^x , k_i^x , K_i^λ and k_i^λ , which can be expected to increase performance even more. An alternative algorithm that exploits also the structure in the subproblems is presented in Chapter 6. _____

The numerical results for the algorithm when solving Newton steps for problems with $n_x = 15$, $n_u = 10$ and $N_s = 2$ are seen in Figure 5.4. The computation times are averaged over fifteen problems of the same size. Here, the proposed algorithm has been compared to a well known state-of-the-art serial algorithm based on the Riccati recursion presented in Section 3.4, which has $\mathcal{O}(N)$ complexity growth. From the figure, the linear complexity of the Riccati based algorithm is evident. It is not obvious from this plot that the complexity grows logarithmically for the implementation of the proposed parallel algorithm. However, it can be observed that the computation time required by the parallel algorithm is significantly less and the growth of the computational complexity is much lower.

The simulations were performed on an Intel Core i7-3517U CPU @ 1.9GHz running Windows 7 (version 6.1, build 7601: Service Pack 1) and MATLAB (8.0.0.783, R2012b).

Figure 5.4: Average computation times for the parallel solution of $\mathcal{P}(N)$. It is compared to a serial Riccati algorithm with $\mathcal{O}(N)$ complexity. Here, systems of dimension $n_x = 15$ and $n_u = 10$ have been used.

Appendix

This appendix contains the proofs of some of the theorems and lemmas presented in this chapter. First, the equations in the KKT system for the UFTOC problem (5.1) and the extended UFTOC problem (5.4) are presented. These equations are used in several of the proofs in this appendix.

The original UFTOC problem is given by (5.1). Let the dual variables λ_t , corresponding to the dynamics constraints, be defined as

$$\lambda_0 \leftrightarrow -x_0 + \bar{x} = 0, \quad (5.38a)$$

$$\lambda_t \leftrightarrow -x_{t+1} + A_t x_t + B_t u_t + a_t = 0, \quad t \in \mathbb{Z}_{0,N-1}. \quad (5.38b)$$

By writing down the equations of the KKT system it is seen that the following equations hold for an optimal solution to (5.1)

$$Q_{x,t} x_t + Q_{xu,t} u_t + l_{x,t} - \lambda_t + A_t^T \lambda_{t+1} = 0, \quad t \in \mathbb{Z}_{0,N-1}, \quad (5.39a)$$

$$Q_{xu,t}^T x_t + Q_{u,t} u_t + l_{u,t} + B_t^T \lambda_{t+1} = 0, \quad t \in \mathbb{Z}_{0,N-1}, \quad (5.39b)$$

$$Q_{x,N} x_N + l_{x,N} - \lambda_N = 0, \quad (5.39c)$$

$$x_{t+1} = A_t x_t + B_t u_t + a_t, \quad t \in \mathbb{Z}_{0,N-1}, \quad (5.39d)$$

$$x_0 = \bar{x}_0. \quad (5.39e)$$

The extended problem is given in (5.4) and the complicating variables $\hat{x}_i$ and $\hat{u}_i$ are introduced as optimization variables in this extended problem. Let the dual variables corresponding to constraints in the subproblems $i \in \mathbb{Z}_{0,p}$ be defined by (5.19) and let the dual variables corresponding to the coupling constraints be defined in (5.27). Then, the KKT system for the extended problem consists of the

following equations

$$Q_{x,t,i}x_{t,i} + Q_{xu,t,i}u_{t,i} + l_{x,t,i} - \lambda_{t,i} + A_{t,i}^T \lambda_{t+1,i} = 0, \quad t \in \mathbb{Z}_{0,N_i-1}, \quad i \in \mathbb{Z}_{0,p}, \quad (5.40a)$$

$$Q_{xu,t,i}^T x_{t,i} + Q_{u,t,i}u_{t,i} + l_{u,t,i} + B_{t,i}^T \lambda_{t+1,i} = 0, \quad t \in \mathbb{Z}_{0,N_i-1}, \quad i \in \mathbb{Z}_{0,p}, \quad (5.40b)$$

$$Q_{N,p}x_{N,p} + l_{N,p} - \lambda_{N,p} = 0, \quad (5.40c)$$

$$x_{0,i} = \hat{x}_i, \quad i \in \mathbb{Z}_{0,p}, \quad (5.40d)$$

$$x_{t+1,i} = A_{t,i}x_{t,i} + B_{t,i}u_{t,i} + a_{t,i}, \quad t \in \mathbb{Z}_{0,N_i-1}, \quad i \in \mathbb{Z}_{0,p}, \quad (5.40e)$$

$$x_{N_i,i} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.40f)$$

$$\hat{x}_0 = \bar{x}, \quad (5.40g)$$

$$\hat{x}_{i+1} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.40h)$$

$$\lambda_{0,p} = \hat{\lambda}_p, \quad (5.40i)$$

$$\lambda_{0,i} = \hat{\lambda}_i - \hat{A}_i^T (\lambda_{t_c,i} + \hat{\lambda}_{i+1}), \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.40j)$$

$$\hat{B}_i^T (\lambda_{t_c,i} + \hat{\lambda}_{i+1}) = 0, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.40k)$$

$$\lambda_{N_i,i} = -\lambda_{t_c,i}, \quad i \in \mathbb{Z}_{0,p-1}. \quad (5.40l)$$

5.A Proof of Theorem 5.4

The proof of Theorem 5.4 is based on the derivation of the KKT systems for the UFTOC problem (5.1) and the extended problem (5.4), given in the introduction of this appendix. In the proof it will be shown that the solutions to the subproblems given by (5.20) and (5.37) together satisfy the KKT conditions (5.39) for the original UFTOC problem (5.1).

Proof: The parallel reduction of $\mathcal{P}(N)$ to $\mathcal{P}(p)$ with $p < N$ follows directly from the theory presented in the text in Section 5.1.

Let $\hat{x}^*$, $\hat{u}^*$ and $\hat{\lambda}^*$ be the optimal primal and dual solutions to the master problem $\mathcal{P}(p)$ in (5.14). By using these optimal parameters, the primal and dual optimal solutions X_i^* and λ_i^* to each subproblem $i \in \mathbb{Z}_{0,p-1}$ are given by (5.20) and (5.37), and by (5.20) and (5.21) for $i = p$. Hence $\hat{x}^*$, $\hat{u}^*$, $\hat{\lambda}^*$, X_i^* and λ_i^* satisfy the KKT conditions in (5.40).

For the optimal solution it follows from (5.40d), (5.40f) and (5.40h) that

$$x_{N_i,i}^* \stackrel{(5.40f)}{=} \hat{A}_i \hat{x}_i^* + \hat{B}_i \hat{u}_i^* + \hat{a}_i \stackrel{(5.40h)}{=} \hat{x}_{i+1}^* \stackrel{(5.40d)}{=} x_{0,i+1}^*, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.41)$$

is satisfied and hence the relation

$$x_{N_i,i}^* = x_{0,i+1}^*, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.42)$$

holds between the terminal state in subproblem i and the initial state in subproblem $i + 1$. Hence, the dynamics equations in (5.40e) and (5.40g) for $t \in \mathbb{Z}_{0,N_i-1}$

and $i \in \mathbb{Z}_{0,p}$ can be written

$$x_{0,0}^* = \bar{x}, \quad (5.43a)$$

$$x_{1,0}^* = A_{0,0}x_{0,0}^* + B_{0,0}u_{0,0}^* + a_{0,0}, \quad (5.43b)$$

$$\vdots$$

$$x_{0,i+1}^* = x_{N_i,i}^* = A_{N_i-1,i}x_{N_i-1,i}^* + B_{N_i-1,i}u_{N_i-1,i}^* + a_{N_i-1,i}, \quad (5.43c)$$

$$x_{1,i+1}^* = A_{0,i+1}x_{0,i+1}^* + B_{0,i+1}u_{0,i+1}^* + a_{0,i+1}, \quad (5.43d)$$

$$\vdots$$

$$x_{N_p,p}^* = A_{N_p-1,p}x_{N_p-1,p}^* + B_{N_p-1,p}u_{N_p-1,p}^* + a_{N_p-1,p}. \quad (5.43e)$$

Here the relation (5.42) has been used in (5.43c) for all $i \in \mathbb{Z}_{0,p-1}$. Furthermore, $A_{t,i}$, $B_{t,i}$ and $a_{t,i}$ were implicitly defined as

$$A_{t,i} \triangleq A_{\sum_{j=0}^{i-1} N_j+t}, \quad B_{t,i} \triangleq B_{\sum_{j=0}^{i-1} N_j+t}, \quad a_{t,i} \triangleq a_{\sum_{j=0}^{i-1} N_j+t}, \quad (5.44)$$

when the extended problem (5.4) was constructed. By using these definitions and also defining the variables

$$x^* \triangleq \begin{bmatrix} x_0^* \\ \vdots \\ x_N^* \end{bmatrix} = \begin{bmatrix} x_{0,0}^* \\ \vdots \\ x_{N_i-1,i}^* \\ x_{0,i+1}^* \\ \vdots \\ x_{N_p,p}^* \end{bmatrix}, \quad u^* \triangleq \begin{bmatrix} u_0^* \\ \vdots \\ u_{N-1}^* \end{bmatrix} = \begin{bmatrix} u_{0,0}^* \\ \vdots \\ u_{N_i-1,i}^* \\ u_{0,i+1}^* \\ \vdots \\ u_{N_p-1,p}^* \end{bmatrix}, \quad (5.45)$$

it can be seen that the equations in (5.43) can be written

$$x_0^* = \bar{x}, \quad (5.46)$$

$$x_1^* = A_0x_0^* + B_0u_0^* + a_0, \quad (5.47)$$

$$\vdots$$

$$x_N^* = A_{N-1}x_{N-1}^* + B_{N-1}u_{N-1}^* + a_{N-1}. \quad (5.48)$$

These equations can be identified as the dynamics equations in the original UFTOC problem (5.1), and hence it is clear that x^* and u^* satisfy (5.39d) and (5.39e) in the KKT system for the original UFTOC problem.

Furthermore, the optimal dual solution λ_i^* is computed as in (5.37), and according to the discussion in Section 5.1.2 and specifically (5.34) this solution is chosen such that it satisfies

$$\lambda_{tc,i} = -\hat{\lambda}_{i+1}. \quad (5.49)$$

Hence, using this dual solution (5.40j) and (5.40l) gives

$$\lambda_{N_i,i}^* \stackrel{(5.40l)}{=} -\lambda_{tc,i}^* \stackrel{(5.34)}{=} \hat{\lambda}_{i+1}^* \stackrel{(5.40j)}{=} \lambda_{0,i+1}^*, \quad i \in \mathbb{Z}_{0,p-1}, \quad (5.50)$$

and (5.40i) gives

$$\lambda_{0,p}^* = \hat{\lambda}_p^*. \quad (5.51)$$

By using (5.50) it can be seen that (5.40a) and (5.40b) for $t = N_i - 1$ can be written

$$\begin{aligned} Q_{x,N_i-1,i} x_{N_i-1,i}^* + Q_{xu,N_i-1,i} u_{N_i-1,i}^* + l_{x,N_i-1,i} - \lambda_{N_i-1,i}^* + A_{N_i-1,i}^T \lambda_{N_i,i}^* & \stackrel{\lambda_{N_i,i}^* = \lambda_{0,i+1}^*}{=} \\ Q_{x,N_i-1,i} x_{N_i-1,i}^* + Q_{xu,N_i-1,i} u_{N_i-1,i}^* + l_{x,N_i-1,i} - \lambda_{N_i-1,i}^* + A_{N_i-1,i}^T \lambda_{0,i+1}^* & = 0, i \in \mathbb{Z}_{0,p-1} \end{aligned} \quad (5.52)$$

$$\begin{aligned} Q_{xu,N_i-1,i}^T x_{N_i-1,i}^* + Q_{u,N_i-1,i} u_{N_i-1,i}^* + l_{u,N_i-1,i} + B_{N_i-1,i}^T \lambda_{N_i,i}^* & \stackrel{\lambda_{N_i,i}^* = \lambda_{0,i+1}^*}{=} \\ Q_{xu,N_i-1,i}^T x_{N_i-1,i}^* + Q_{u,N_i-1,i} u_{N_i-1,i}^* + l_{u,N_i-1,i} + B_{N_i-1,i}^T \lambda_{0,i+1}^* & = 0, i \in \mathbb{Z}_{0,p-1}. \end{aligned} \quad (5.53)$$

Hence (5.40a), (5.40b) and (5.40c) in the KKT system for the extended problem can be written

$$Q_{x,0,0} x_{0,0}^* + Q_{xu,0,0} u_{0,0}^* + l_{x,0,0} - \lambda_{0,0}^* + A_{0,0}^T \lambda_{1,0}^* = 0, \quad (5.54a)$$

$$Q_{xu,0,0}^T x_{0,0}^* + Q_{u,0,0} u_{0,0}^* + l_{u,0,0} + B_{0,0}^T \lambda_{1,0}^* = 0, \quad (5.54b)$$

$\vdots$

$$Q_{x,N_i-1,i} x_{N_i-1,i}^* + Q_{xu,N_i-1,i} u_{N_i-1,i}^* + l_{x,N_i-1,i} - \lambda_{N_i-1,i}^* + A_{N_i-1,i}^T \lambda_{0,i+1}^* = 0, \quad (5.54c)$$

$$Q_{xu,N_i-1,i}^T x_{N_i-1,i}^* + Q_{u,N_i-1,i} u_{N_i-1,i}^* + l_{u,N_i-1,i} + B_{N_i-1,i}^T \lambda_{0,i+1}^* = 0, \quad (5.54d)$$

$$Q_{x,0,i+1} x_{0,i+1}^* + Q_{xu,0,i+1} u_{0,i+1}^* + l_{x,0,i+1} - \lambda_{0,i+1}^* + A_{0,i+1}^T \lambda_{1,i+1}^* = 0, \quad (5.54e)$$

$$Q_{xu,0,i+1}^T x_{0,i+1}^* + Q_{u,0,i+1} u_{0,i+1}^* + l_{u,0,i+1} + B_{0,i+1}^T \lambda_{1,i+1}^* = 0, \quad (5.54f)$$

$\vdots$

$$Q_{N_p,p} x_{N_p,p}^* + l_{N_p,p} - \lambda_{N_p,p}^* = 0. \quad (5.54g)$$

Here (5.52) and (5.53) have been used in (5.54c) and (5.54d), respectively, for all $i \in \mathbb{Z}_{0,p-1}$. $Q_{t,i}$ and $l_{x,i}$ were implicitly defined as

$$Q_{t,i} \triangleq Q_{\sum_{j=0}^{i-1} N_j+t}, \quad l_{t,i} \triangleq l_{\sum_{j=0}^{i-1} N_j+t}, \quad (5.55)$$

when the extended problem (5.4) was constructed. By using these definitions and defining the variable

$$\lambda^* \triangleq \begin{bmatrix} \lambda_0^* \\ \vdots \\ \lambda_{N_p,p}^* \end{bmatrix} = \begin{bmatrix} \lambda_{0,0}^* \\ \vdots \\ \lambda_{N_i-1,i}^* \\ \lambda_{0,i+1}^* \\ \vdots \\ \lambda_{N_p,p}^* \end{bmatrix}, \quad (5.56)$$

together with the definitions of x^* and u^* in (5.45) it can be seen that the system of equations (5.54) can be written

$$Q_{x,0}x_0^* + Q_{xu,0}u_0^* + l_{x,0} - \lambda_0^* + A_0^T \lambda_1^* = 0, \quad (5.57a)$$

$$Q_{xu,0}^T x_0^* + Q_{u,0}u_0^* + l_{u,0} + B_0^T \lambda_1^* = 0, \quad (5.57b)$$

$$\vdots$$

$$Q_{x,N}x_N^* + l_{x,N} - \lambda_N^* = 0. \quad (5.57c)$$

These equations can be recognized as (5.39a), (5.39b) and (5.39c) in the KKT system of the original UFTOC problem.

Hence, the primal and dual solutions x^* , u^* and λ^* defined in (5.45) and (5.56), respectively, satisfy the KKT system (5.39), and from Theorem 2.8 it follows that x^* , u^* and λ^* are optimal primal and dual solutions to the UFTOC problem (5.1).

Furthermore, the solution in each subproblem $i \in \mathbb{Z}_{0,p}$ can be computed completely independent of the other subproblems, and hence the solution x^* , u^* and λ^* to (5.1) can be computed in parallel from the solution $\hat{x}^*$, $\hat{u}^*$ and $\hat{\lambda}^*$ to (5.14). This concludes the proof. □

5.B Proof of Lemma 5.8

Proof: The nullspace of A_i^T is given by all $\lambda_i^{\mathcal{N}}$ such that $A_i^T \lambda_i^{\mathcal{N}} = 0$, which by the definition of A_i in (5.17b) can be expressed as

$$-\lambda_{t,i}^{\mathcal{N}} + A_{t,i}^T \lambda_{t+1,i}^{\mathcal{N}} = 0, \quad t \in \mathbb{Z}_{0,N_i-1}, \quad (5.58a)$$

$$B_{t,i}^T \lambda_{t+1,i}^{\mathcal{N}} = 0, \quad t \in \mathbb{Z}_{0,N_i-1}, \quad (5.58b)$$

$$\lambda_{N_i,i}^{\mathcal{N}} = -\lambda_{tc,i}^{\mathcal{N}}. \quad (5.58c)$$

Equation (5.58a) and (5.58c) can be combined into

$$\lambda_i^{\mathcal{N}} = \begin{bmatrix} \lambda_{0,i}^{\mathcal{N}} \\ \vdots \\ \lambda_{N_i,i}^{\mathcal{N}} \\ \lambda_{tc,i}^{\mathcal{N}} \end{bmatrix} = \begin{bmatrix} -(\prod_{t=0}^{N_i-1} A_{t,i})^T \\ \vdots \\ -I \\ I \end{bmatrix} \lambda_{tc,i}^{\mathcal{N}} = \begin{bmatrix} -\hat{A}_i^T \\ -\mathcal{D}_i^T \\ I \end{bmatrix} \lambda_{tc,i}^{\mathcal{N}}, \quad (5.59)$$

where $\mathcal{D}_i$ and $\hat{A}_i$ are defined as in (5.6) and (5.8), respectively. By combining (5.58a) and (5.58b) it can be seen that

$$B_{N_i-1,i}^T \lambda_{N_i,i}^{\mathcal{N}} = 0, \quad B_{N_i-2,i}^T A_{N_i-1,i}^T \lambda_{N_i,i}^{\mathcal{N}} = 0, \quad \dots, \quad B_{0,i}^T \left(\prod_{t=1}^{N_i-1} A_{t,i} \right)^T \lambda_{N_i,i}^{\mathcal{N}} = 0, \quad (5.60)$$

which, using also (5.58c), can be expressed as

$$-\begin{bmatrix} B_{0,i}^T \left(\prod_{t=1}^{N_i-1} A_{t,i} \right)^T \\ \vdots \\ B_{N_i-2,i}^T A_{N_i-1,i}^T \\ B_{N_i-1,i}^T \end{bmatrix} \lambda_{tc,i}^{\mathcal{N}} = -S_i^T \lambda_{tc,i}^{\mathcal{N}} = 0, \quad (5.61)$$

where S_i is defined in (5.7). Hence $\lambda_{tc,i}^{\mathcal{N}}$ has to satisfy $\lambda_{tc,i}^{\mathcal{N}} \in \mathcal{N}(S_i^T)$. For notational convenience, let $w_i \triangleq \lambda_{tc,i}^{\mathcal{N}}$ and define Z_i as

$$Z_i \triangleq \begin{bmatrix} -\hat{A}_i^T \\ -\mathcal{D}_i^T \\ I \end{bmatrix}. \quad (5.62)$$

Then any nullspace element $\lambda_i^{\mathcal{N}}$ can be expressed as

$$\lambda_i^{\mathcal{N}} = Z_i w_i, \quad w_i \in \mathcal{N}(S_i^T), \quad (5.63)$$

which completes the proof. □

5.C Proof of Theorem 5.10

Proof: The equations (5.28) are directly given by (5.40i)-(5.40l) in the KKT conditions for the extended problem, which concludes the proof. □

6

Parallel Riccati Recursion

In this chapter, an algorithm for solving UFTOC problems by computing the Riccati recursion in parallel is presented. This algorithm is similar to the one presented in Chapter 5 in the sense that it can be used to compute an optimal solution to the UFTOC problem in (3.39) in a fixed number of iterations in parallel. Furthermore, both algorithms use a fixed a priori known communication layout between the computational units. However, the algorithm presented here parallelizes the Riccati recursion itself, and the subproblems are computed by solving parts of the Riccati recursion individually. Hence, in this algorithm the structure in the UFTOC problem is exploited even in the individual subproblems. The results in this chapter will later be presented in Nielsen and Axehill (2015).

The proposed algorithm has been implemented in MATLAB as a proof-of-concept that it works, and a comparison of this implementation and the algorithm in Chapter 5 is presented. In the MATLAB implementation the communication overhead is neglected. Furthermore, an ANSI-C implementation of the algorithm presented in this chapter has been evaluated truly in parallel on a computational cluster consisting of several multi-core computers.

The classical serial Riccati recursion that was introduced in Section 3.4 exploits the causality of the problem. For that reason it is not obvious that it can be split and parallelized in time, especially without involving some form of iterative consensus step. In this chapter it is shown that it in fact is possible to compute the Riccati factorization in Algorithm 4 and the backward and forward recursions in algorithms 5 and 6 directly in parallel in time. The main idea is to exploit the problem structure in time and divide the Riccati recursion into several parts. Each part of the Riccati recursion corresponds to the KKT conditions of a smaller UFTOC problem, and it will be shown how to eliminate variables from each of

these parts in parallel in order to obtain a new Riccati recursion but with fewer time-steps. The new Riccati recursion corresponds to the KKT conditions of a master UFTOC problem. Hence, the overall structure where a UFTOC problem $\mathcal{P}(N)$ is reduced to a master UFTOC problem $\mathcal{P}(p)$ is similar to the approach presented in Chapter 5. However, some important differences are, *e.g.*, that the method to eliminate variables in the smaller UFTOC problems are different and utilizes the structure in those problems, and that there are no difficulties with non-unique dual variables as in the algorithm in Chapter 5. The differences and similarities between the algorithms in chapter 5 and 6 are discussed in more detail in Section 6.4. The parallel algorithm in this chapter is related to nonserial dynamic programming over trees as described in the introduction to Chapter 5.

As a reminder, the UFTOC problem (3.39) that corresponds to the Newton step that occurs in many IP and AS solvers is repeated here

$$\begin{aligned}
 & \underset{x,u}{\text{minimize}} && \sum_{t=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^T \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \begin{bmatrix} l_{x,t} \\ l_{u,t} \end{bmatrix}^T \begin{bmatrix} x_t \\ u_t \end{bmatrix} + c_t \right) + \\
 & && \frac{1}{2} x_N^T Q_{x,N} x_N + l_{x,N}^T x_N + c_N \\
 & \text{subject to} && x_0 = \bar{x} \\
 & && x_{t+1} = A_t x_t + B_t u_t + a_t, \quad t \in \mathbb{Z}_{0,N-1}.
 \end{aligned} \tag{6.1}$$

As in Chapter 5, let assumptions 3.4 and 3.5 hold and let $\mathcal{P}(N)$ denote an optimization problem with the structure in (6.1).

6.1 Problem Decomposition and Reduction

The focus in this section will be to describe how to split the computations of the solution to the problem (6.1) into several smaller independent subproblems by dividing the prediction horizon into several intervals (Section 6.1.1), how to eliminate variables in each independent subproblem using Riccati recursions (Section 6.1.2) and how to combine the reduced subproblems to a master problem (Section 6.1.3).

By examining algorithms 4 and 5, it can be seen that given $P_{\bar{t}}$, $\Psi_{\bar{t}}$ and $\bar{c}_{\bar{t}}$ the factorization and backward recursion can be computed for $0 \leq t \leq \bar{t}$. If the factorization and backward recursions are computed, it follows from Algorithm 6 that given $x_{\bar{s}}$ the forward recursion can be computed for $\bar{s} \leq t \leq N$. Hence, if $P_{\bar{t}}$, $\Psi_{\bar{t}}$, $\bar{c}_{\bar{t}}$ and $x_{\bar{s}}$ are known, the Riccati recursion and the primal and dual optimal solutions for the interval $\bar{s} \leq t \leq \bar{t}$ can be computed using information from this interval only. As a consequence, provided that $P_{t_{i+1}}$, $\Psi_{t_{i+1}}$, $\bar{c}_{t_{i+1}}$ and x_{t_i} are known for $i \in \mathbb{Z}_{0,p}$ for some p , it is possible to compute the Riccati recursion and the primal and dual optimal solutions in each interval $t_i \leq t \leq t_{i+1}$ with $i \in \mathbb{Z}_{0,p}$ independently from the other intervals. This property will be exploited in Section 6.1.1.

6.1.1 Splitting into independent parts

To decompose the UFTOC problem (6.1), let the prediction horizon be divided into $p + 1$ intervals, or *batches*, such that x and u are split into

$$\begin{bmatrix} x_0 \\ \vdots \\ x_{t_1} \end{bmatrix}, \dots, \begin{bmatrix} x_{t_p} \\ \vdots \\ x_{t_{p+1}} \end{bmatrix}, \begin{bmatrix} u_0 \\ \vdots \\ u_{t_1-1} \end{bmatrix}, \dots, \begin{bmatrix} u_{t_p} \\ \vdots \\ u_{t_{p+1}-1} \end{bmatrix}, \quad (6.2)$$

where the last state $x_{t_{i+1}}$ in batch i coincides with the first state in batch $i + 1$. Now introduce the batch-wise variables

$$x_i = \begin{bmatrix} x_{0,i} \\ \vdots \\ x_{N_i,i} \end{bmatrix} \triangleq \begin{bmatrix} x_{t_i} \\ \vdots \\ x_{t_{i+1}} \end{bmatrix}, \quad u_i = \begin{bmatrix} u_{0,i} \\ \vdots \\ u_{N_i-1,i} \end{bmatrix} \triangleq \begin{bmatrix} u_{t_i} \\ \vdots \\ u_{t_{i+1}-1} \end{bmatrix}, \quad i \in \mathbb{Z}_{0,p}, \quad (6.3)$$

where N_i is the length of batch i and $t_0 = 0$. Note that $x_{N_i,i} = x_{0,i+1}$.

By following the reasoning in the introduction of this section, it is possible to compute the Riccati recursion and the optimal value in batch i if $\hat{x}_i \triangleq x_{t_i}$, $\hat{P}_{i+1} \triangleq P_{t_{i+1}}$, $\hat{\Psi}_{i+1} \triangleq \Psi_{t_{i+1}}$ and $\hat{c}_{i+1} \triangleq \bar{c}_{t_{i+1}}$ are known. Hence, if these variables are known for all batches $i \in \mathbb{Z}_{0,p}$, the solution to the original UFTOC problem (6.1) can be computed from $p + 1$ independent subproblems in the UFTOC form

$$\begin{aligned} & \underset{x_i, u_i}{\text{minimize}} \quad \sum_{t=0}^{N_i-1} \left(\frac{1}{2} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix}^T \begin{bmatrix} Q_{x,t,i} & Q_{xu,t,i} \\ Q_{xu,t,i}^T & Q_{u,t,i} \end{bmatrix} \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix} + \begin{bmatrix} l_{x,t,i} \\ l_{u,t,i} \end{bmatrix}^T \begin{bmatrix} x_{t,i} \\ u_{t,i} \end{bmatrix} + c_{t,i} \right) + \\ & \quad \frac{1}{2} x_{N_i,i}^T \hat{P}_{i+1} x_{N_i,i} - \hat{\Psi}_{i+1}^T x_{N_i,i} + \hat{c}_{i+1} \\ & \text{subject to} \quad x_{0,i} = \hat{x}_i \\ & \quad x_{t+1,i} = A_{t,i} x_{t,i} + B_{t,i} u_{t,i} + a_{t,i}, \quad t \in \mathbb{Z}_{0,N_i-1}, \end{aligned} \quad (6.4)$$

using $p + 1$ individual Riccati recursions. Here $Q_{x,t,i}$, $Q_{xu,t,i}$, $Q_{u,t,i}$, $l_{x,t,i}$, $l_{u,t,i}$, $c_{t,i}$, $A_{t,i}$, $B_{t,i}$ and $a_{t,i}$ are defined consistently. Note that for the final batch it holds that $\hat{P}_{p+1} = Q_{x,N}$, $\hat{\Psi}_{p+1} = -l_{x,N}$ and $\hat{c}_{p+1} = c_N$ and they are hence known.

Remark 6.1. The individual subproblems in the form in (6.4) do not have any terminal constraints as in the subproblems in (5.11) in Chapter 5. Here the coupling is instead given by the terminal state cost $\frac{1}{2} x_{N_i,i}^T \hat{P}_{i+1} x_{N_i,i} - \hat{\Psi}_{i+1}^T x_{N_i,i}$.

6.1.2 Eliminate local variables in a subproblem

It will now be shown that even when $\hat{P}_{i+1}$, $\hat{\Psi}_{i+1}$ and $\hat{c}_{i+1}$ are not known, it is possible to work on the individual subproblems to eliminate local variables and reduce their sizes. This can be done separately for the $p + 1$ subproblems, which opens up for a structure that can be solved in parallel. The core idea with this approach is that the unknowns $\hat{P}_{i+1}$ and $\hat{\Psi}_{i+1}$ will indeed influence the solution

of the subproblem, but as soon will be shown the resulting degree of freedom is often very limited compared to the dimension of the full control signal vector u_i . The constant $\hat{c}_{i+1}$ affects the optimal value of the cost function but not the solution. It will be shown in this section that the structured perturbation from $\hat{P}_{i+1}$ and $\hat{\Psi}_{i+1}$ only introduces $n_{\hat{u}} \leq n_x$ degrees of freedom, and hence the subproblem can be reduced to depend only on the initial state $\hat{x}_i$ and the freedom in the structured perturbation, of dimension n_x and $n_{\hat{u}}$ respectively.

Lemma 6.2 gives an expression for the cost-to-go at state $x_{t,i}$ in (6.4) for some t when the control inputs $u_{\tau,i}$ for $\tau \in \mathbb{Z}_{t,N-1}$ are computed in a certain way, and the lemma will be used in this section to simplify some key computations.

Lemma 6.2. *Consider a UFTOC problem in the form in (6.4). Assume that $u_{\tau,i}$ is computed as*

$$u_{\tau,i} = k_{\tau+1,i} + K_{\tau+1,i}x_{\tau,i} + \bar{u}_{\tau,i}, \quad \tau \in \mathbb{Z}_{t,N-1}, \quad (6.5)$$

where $\bar{u}_{\tau,i} \in \mathbb{R}^{n_u}$ is an arbitrary vector and the preliminary feedback matrices $K_{\tau+1,i}$ and $k_{\tau+1,i}$ are computed in algorithms 4 and 5 for fixed $\hat{P}_{i+1}$, $\hat{\Psi}_{i+1}$ and $\hat{c}_{i+1}$, respectively. Then, the cost-to-go at a state $x_{t,i}$ in (6.4) is given by

$$\bar{V}(x_{t,i}, \bar{u}_i) \triangleq \frac{1}{2}x_{t,i}^T P_{t,i} x_{t,i} - \Psi_{t,i}^T x_{t,i} + \bar{c}_{t,i} + \frac{1}{2} \sum_{\tau=t}^{N-1} \bar{u}_{\tau,i}^T G_{\tau+1,i} \bar{u}_{\tau,i}, \quad (6.6)$$

where $P_{t,i}$, $\Psi_{t,i}$, $\bar{c}_{t,i}$ and $G_{\tau+1,i}$ are computed in algorithms 4 and 5.

Proof: For the proof of Lemma 6.2, see Appendix 6.A. □

In the remaining part of this section the subindices i in (6.4) are omitted for notational brevity, meaning that $\hat{\Psi}_{i+1}$ is written $\hat{\Psi}$, etc.

It will now be shown how local variables in a subproblem can be eliminated by exploiting the structure in (6.4). A preliminary feedback given by the Riccati recursion will be used to simplify the reduction of the subproblem. The use of this preliminary feedback is in principle not necessary, but it will later be seen that some computationally demanding key computations can be performed more efficiently by using it. To compute this preliminary feedback, let the UFTOC problem (6.4) be factored and solved for $\hat{P} = 0$, $\hat{\Psi} = 0$ and $\hat{c} = 0$ using the Riccati factorization and backward recursion in algorithms 4 and 5. The resulting optimal control law for $\hat{P} = 0$ and $\hat{\Psi} = 0$ is then $u_{0,t} = k_{0,t+1} + K_{0,t+1}x_t$ for $t \in \mathbb{Z}_{0,N-1}$. The subindex "0" is used to denote variables that correspond to the solution of the UFTOC problem (6.4) when the preliminary $\hat{P} = 0$, $\hat{\Psi} = 0$ and $\hat{c} = 0$ are used. Hence, $k_{0,t+1}$ and $K_{0,t+1}$ define the preliminary feedback and it will be shown that the changes of this preliminary optimal control law are small when $\hat{P} \neq 0$ and $\hat{\Psi} \neq 0$.

Remark 6.3. It is possible to compute the preliminary feedback for other choices than $\hat{P} = 0$ and $\hat{\Psi} = 0$ and still simplify the key computations. However, since the notation then becomes tedious, the choice $\hat{P} = 0$ and $\hat{\Psi} = 0$ is used here for presentation purposes.

It will now be investigated how the control signal u_t and the cost function are affected when $\hat{P} \neq 0$, $\hat{\Psi} \neq 0$ and $\hat{c} \neq 0$. Let the contribution to the control signal u_t from the unknown $\hat{P}$ and $\hat{\Psi}$ be denoted $\bar{u}_t \in \mathbb{R}^{n_u}$. By using the preliminary feedback, the control law can be written

$$u_t = k_{0,t+1} + K_{0,t+1}x_t + \bar{u}_t, \quad t \in \mathbb{Z}_{0,N-1}. \quad (6.7)$$

Note that $\bar{u}_t$ is an arbitrary n_u -vector and, hence, there is no loss of generality in this assumption. From now on, the control law (6.7) is used in the subproblem, and it will be shown that the degree of freedom in $\bar{u}_t$ can be reduced. By defining $\bar{Q}_{\bar{u}}$, A , B and a as

$$\bar{Q}_{\bar{u}} \triangleq \begin{bmatrix} G_{0,1} & & \\ & \ddots & \\ & & G_{0,N} \end{bmatrix}, \quad A \triangleq \begin{bmatrix} I & & \\ & A_0 + B_0 K_{0,1} & \\ & \vdots & \\ & \prod_{t=0}^{N-1} (A_t + B_t K_{0,t+1}) \end{bmatrix}, \quad (6.8a)$$

$$B \triangleq \begin{bmatrix} 0 & 0 & \dots & 0 \\ B_0 & & & \\ (A_1 + B_1 K_{0,2})B_0 & B_1 & & \vdots \\ \vdots & & \ddots & \\ \prod_{t=1}^{N-1} (A_t + B_t K_{0,t+1}) B_0 & \dots & B_{N-1} \end{bmatrix}, \quad (6.8b)$$

$$a \triangleq \begin{bmatrix} 0 \\ a_0 + B_0 k_{0,1} \\ (A_1 + B_1 K_{0,2})(a_0 + B_0 k_{0,1}) + a_1 + B_1 k_{0,2} \\ \vdots \\ \sum_{\tau=0}^{N-1} \prod_{t=\tau+1}^{N-1} (A_t + B_t K_{0,t+1}) (a_\tau + B_\tau k_{0,\tau+1}) \end{bmatrix}, \quad (6.8c)$$

and using (6.7), the states x along the prediction horizon can be expressed as

$$x = Ax_0 + B\bar{u} + a, \quad (6.9)$$

where $\bar{u} \in \mathbb{R}^{N n_u}$. The cost function when the control law (6.7) and terminal state cost $\hat{P} = 0$, $\hat{\Psi} = 0$ and $\hat{c} = 0$ are used is given by the cost-to-go at x_0 given by Lemma 6.2, i.e.,

$$\bar{V}(x_0, \bar{u}) = \frac{1}{2} x_0^T P_{0,0} x_0 - \Psi_{0,0}^T x_0 + \frac{1}{2} \bar{u}^T \bar{Q}_{\bar{u}} \bar{u} + \bar{c}_{0,0}. \quad (6.10)$$

Here $P_{0,0}$, $\Psi_{0,0}$ and $\bar{c}_{0,0}$ are computed by algorithms 4 and 5 with the choice $\hat{P} = 0$, $\hat{\Psi} = 0$ and $\hat{c} = 0$.

Remark 6.4. When using the preliminary feedback to compute the control signal as in (6.7) the cross terms between x_0 and $\bar{u}$ in the cost function in (6.10) are eliminated. Furthermore, there is no linear penalty term for $\bar{u}$ and the quadratic weight matrix $\bar{Q}_{\bar{u}}$ is block diagonal as can be seen in (6.8a). These properties will later be important from a computational point of view.

Let the last block rows in A , a and B be denoted as $\hat{A}$, $\hat{a}$ and S , respectively. The dynamics equations from x_0 to x_N are then given by

$$x_N = \hat{A}x_0 + S\bar{u} + \hat{a}. \quad (6.11)$$

The terminal cost in (6.4), given by $\hat{P}$, $-\hat{\Psi}$ and $\hat{c}$, is (possibly) non-zero. Hence, the total cost in the subproblem (6.4) is obtained by adding the non-zero terminal cost to the cost function $\tilde{V}(x_0, \bar{u})$ in (6.10). Note that the influence of a non-zero $\hat{P}$ and/or $-\hat{\Psi}$ on the states and control signals for $t < N$ is incorporated in $\bar{u}$ by definition. The UFTOC problem (6.4) can then equivalently be written

$$\begin{aligned} & \underset{x_0, \bar{u}, x_N}{\text{minimize}} && \frac{1}{2}x_0^T P_{0,0}x_0 - \Psi_{0,0}^T x_0 + \frac{1}{2}\bar{u}^T \bar{Q}_{\bar{u}}\bar{u} + \bar{c}_{0,0} + \\ & && \frac{1}{2}x_N^T \hat{P}x_N - \hat{\Psi}^T x_N + \hat{c} \\ & \text{subject to} && x_0 = \hat{x} \\ & && x_N = \hat{A}x_0 + S\bar{u} + \hat{a}, \end{aligned} \quad (6.12)$$

where the dynamics equations in (6.11) have been used. This is a UFTOC problem with prediction horizon 1, i.e., one step from the initial to the final state, and Nn_u control signals. This problem is obtained similarly as the subproblems in partial condensing which is introduced in Axehill (2015). The equations that define the factorization of the KKT system of this UFTOC problem are

$$\bar{F} = P_{0,0} + \hat{A}^T \hat{P} \hat{A}, \quad (6.13a)$$

$$\bar{G} = \bar{Q}_{\bar{u}} + S^T \hat{P} S, \quad (6.13b)$$

$$\bar{H} = \hat{A}^T \hat{P} S, \quad (6.13c)$$

$$\bar{G}\bar{K} = -\bar{H}^T, \quad (6.13d)$$

$$\bar{G}\bar{k} = S^T (\hat{\Psi} - \hat{P}\hat{a}), \quad (6.13e)$$

and hence (6.13) can be used to compute the optimal solution of (6.12).

Remark 6.5. The preliminary feedback in (6.7) eliminates the cross terms between x_0 and $\bar{u}$, giving only one term in (6.13c) which simplifies computations. _____

Using (6.13b) and (6.13c), the system of equations (6.13d) can be re-written as

$$(\bar{Q}_{\bar{u}} + S^T \hat{P} S)\bar{K} = -S^T \hat{P} \hat{A}, \quad (6.14)$$

where $\bar{K} \in \mathbb{R}^{Nn_u \times n_x}$. It will now be shown that it is possible to reduce the number of equations by exploiting the structure in (6.14). To do this, let $U_1 \in \mathbb{R}^{Nn_u \times n_1}$ with $n_1 \leq n_x$ be an orthonormal basis for $\mathcal{R}(S^T)$ and let U_2 be an orthonormal basis of $\mathcal{R}(S^T)^\perp$, both given by the singular value decomposition of S^T , i.e.,

$$S^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_S & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U_1 \Sigma_S V_1^T. \quad (6.15)$$

Then, $U \triangleq [U_1 \ U_2]$ is an orthonormal basis for $\mathbb{R}^{Nn_u}$, and by using the identity

$UU^T = I$ and the definitions

$$\Sigma \triangleq \Sigma_S V_1^T \hat{P} V_1 \Sigma_S, \quad \Gamma \triangleq -\Sigma_S V_1^T \hat{P} \hat{A}, \quad (6.16)$$

equation (6.14) can equivalently be written

$$(UU^T \bar{Q}_{\bar{u}} + U_1 \Sigma U_1^T) \bar{K} = U_1 \Gamma. \quad (6.17)$$

By multiplying (6.17) with U_1^T and U_2^T , respectively, from left gives the two systems of equations

$$(U_1^T \bar{Q}_{\bar{u}} + \Sigma U_1^T) \bar{K} = \Gamma, \quad (6.18a)$$

$$U_2^T \bar{Q}_{\bar{u}} \bar{K} = 0 \iff \bar{K} = \bar{Q}_{\bar{u}}^{-1} U_1 \hat{K}, \quad (6.18b)$$

where $\hat{K} \in \mathbb{R}^{n_1 \times n_x}$. Here, $U_1^T U_1 = I$, $U_2^T U_1 = 0$ and $U_2^T U_2 = I$ were used to reduce the size of the system of equations. The matrix $\hat{K}$ is introduced in (6.18b) to parametrize the nullspace of U_2^T as $U_1 \hat{K}$. Hence, the feedback matrix $\bar{K}$ is parametrized as $\bar{K} = \bar{Q}_{\bar{u}}^{-1} U_1 \hat{K}$, and inserting this expression into (6.18a) gives

$$(U_1^T \bar{Q}_{\bar{u}} + \Sigma U_1^T) \bar{Q}_{\bar{u}}^{-1} U_1 \hat{K} = (I + \Sigma U_1^T \bar{Q}_{\bar{u}}^{-1} U_1) \hat{K} = \Gamma. \quad (6.19)$$

This parametrization and reduction of the system of equations is done without any assumptions on $\hat{P}$ and $\hat{\Psi}$ and, hence, the reduction can be performed independently of the unknown $\hat{P}$ and $\hat{\Psi}$.

Multiplying (6.19) with $U_1^T \bar{Q}_{\bar{u}}^{-1} U_1 \in \mathbb{S}_{++}^{n_1}$ from the left gives

$$(U_1^T \bar{Q}_{\bar{u}}^{-1} U_1 + U_1^T \bar{Q}_{\bar{u}}^{-1} U_1 \Sigma U_1^T \bar{Q}_{\bar{u}}^{-1} U_1) \hat{K} = U_1^T \bar{Q}_{\bar{u}}^{-1} U_1 \Gamma. \quad (6.20)$$

From the definition of Σ and Γ together with (6.15), this system of equations can be written

$$(U_1^T \bar{Q}_{\bar{u}}^{-1} U_1 + U_1^T \bar{Q}_{\bar{u}}^{-1} S^T \hat{P} S \bar{Q}_{\bar{u}}^{-1} U_1) \hat{K} = -U_1^T \bar{Q}_{\bar{u}}^{-1} S^T \hat{P} \hat{A}. \quad (6.21)$$

Now, by introducing the variables

$$\hat{Q}_u \triangleq U_1^T \bar{Q}_{\bar{u}}^{-1} U_1, \quad \hat{B} \triangleq S \bar{Q}_{\bar{u}}^{-1} U_1, \quad (6.22)$$

$$\hat{G} \triangleq \hat{Q}_u + \hat{B}^T \hat{P} \hat{B}, \quad \hat{H} \triangleq \hat{A}^T \hat{P} \hat{B}, \quad (6.23)$$

equation (6.21) can be written as

$$\hat{G} \hat{K} = -\hat{H}^T. \quad (6.24)$$

Remark 6.6. Since using the preliminary feedback in (6.7) results in a block diagonal $\bar{Q}_{\bar{u}}$ with the blocks given by $G_{0,t+1}$ for $t \in \mathbb{Z}_{0,N-1}$, computing $\hat{Q}_u$ and $\hat{B}$ in (6.22) can be efficiently done by block-wise computations. Furthermore, the factorizations of $G_{0,t+1}$ from the computation of $K_{0,t+1}$ can be re-used here. _____

Similarly, the structure in (6.13e) can be exploited and this system of equations can be reduced to

$$\hat{G} \hat{k} = \hat{B}^T (\hat{\Psi} - \hat{P} \hat{a}), \quad (6.25)$$

with $\hat{k} \in \mathbb{R}^{n_1}$. Hence, by also defining the variables

$$\hat{F} \triangleq \bar{F}, \quad \hat{Q}_x \triangleq P_{0,0}, \quad \hat{l}_x \triangleq -\Psi_{0,0}, \quad (6.26)$$

the equations (6.13) can now be written as

$$\hat{F} = \hat{Q}_x + \hat{A}^T \hat{P} \hat{A}, \quad (6.27a)$$

$$\hat{G} = \hat{Q}_u + \hat{B}^T \hat{P} \hat{B}, \quad (6.27b)$$

$$\hat{H} = \hat{A}^T \hat{P} \hat{B}, \quad (6.27c)$$

$$\hat{G} \hat{K} = -\hat{H}^T, \quad (6.27d)$$

$$\hat{G} \hat{k} = \hat{B}^T (\hat{\Psi} - \hat{P} \hat{a}), \quad (6.27e)$$

which can be identified as the factorization of the KKT system of a UFTOC problem in the form (6.12) but with control signal $n_{\hat{u}} = n_1 \leq n_x$. Hence (6.27) define the optimal solution to a smaller UFTOC problem in the variables $\hat{x}, x_N \in \mathbb{R}^{n_x}$ and $\hat{u} \in \mathbb{R}^{n_{\hat{u}}}$. This reduction is performed by eliminating local variables in the problem (6.4) and by exploiting the structure in the subproblem. This important result is summarized in Theorem 6.7, where the subindices i in (6.4) are again used for completeness.

Theorem 6.7. *A UFTOC problem given in the form (6.4) with unknown $\hat{P}_{i+1}$, $\hat{\Psi}_{i+1}$ and $\hat{c}_{i+1}$ can be reduced to a UFTOC problem in the form*

$$\begin{aligned} & \underset{x_{0,i}, x_{N,i}, \hat{u}_i}{\text{minimize}} && \frac{1}{2} x_{0,i}^T \hat{Q}_{x,i} x_{0,i} + \frac{1}{2} \hat{u}_i^T \hat{Q}_{u,i} \hat{u}_i + \hat{l}_{x,i}^T x_{0,i} + \hat{c}_i + \\ & && \frac{1}{2} x_{N,i}^T \hat{P}_{i+1} x_{N,i} - \hat{\Psi}_{i+1}^T x_{N,i} + \hat{c}_{i+1} \\ & \text{subject to} && x_{0,i} = \hat{x}_i \\ & && x_{N,i} = \hat{A}_i x_{0,i} + \hat{B}_i \hat{u}_i + \hat{a}_i, \end{aligned} \quad (6.28)$$

where $\hat{x}_i, x_{0,i}, x_{N,i} \in \mathbb{R}^{n_x}$ and $\hat{u}_i \in \mathbb{R}^{n_{\hat{u}}}$, with $n_{\hat{u}} \leq n_x$. $\hat{A}_i$ and $\hat{a}_i$ are defined as the last block rows in A_i and a_i given (6.8a) and (6.8c), respectively, and $\hat{Q}_{x,i}$, $\hat{Q}_{u,i}$, $\hat{l}_{x,i}$ and $\hat{B}_i$ are given by (6.22) and (6.26), and $\hat{c}_i \triangleq \bar{c}_{0,0}$ where $\bar{c}_{0,0}$ is defined in (6.10).

Proof: Theorem 6.7 follows directly from the derivations that are presented in the text in Section 6.1.2. $\square$

To avoid computing the orthonormal basis U_1 and U_2 in practice, a transformation $\hat{K} = T\hat{L}$, where $T \in \mathbb{R}^{n_1 \times n_x}$ has full rank, $U_1 T = S^T$ and $\hat{L} \in \mathbb{R}^{n_x \times n_x}$, can be used. By using this choice of $\hat{K}$ in (6.21) and then multiplying from the left with T^T , the matrices $\hat{Q}_u$, $\hat{B}$ and (6.24) can instead be written

$$\hat{Q}_u = \hat{B} \triangleq S \bar{Q}_u^{-1} S^T, \quad (6.29)$$

$$\hat{G} \hat{L} = -\hat{H}^T, \quad (6.30)$$

where $\hat{G}$ and $\hat{H}$ are defined as in (6.23) but with the new $\hat{Q}_u$ and $\hat{B}$. The UFTOC problem corresponding to (6.27) then obtains an (possibly) increased control sig-

nal dimension $n_{\hat{u}} = n_x \geq n_1$ compared to when $\hat{Q}_u$ and $\hat{B}$ are defined as in (6.22). The advantage by using this choice is that $\hat{Q}_u$ and $\hat{B}$ can be easily computed. Analogously calculations can be made for $\hat{k}$.

Remark 6.8. If S^T is rank deficient, then $U_1 \in \mathbb{R}^{N n_u \times n_1}$ will have $n_1 < n_x$ columns. Hence $\hat{G}$ is singular and $\hat{L}$ non-unique in (6.30). How to cope with this case is described in Section 3.4.2 and Axehill (2008).

For the last subproblem $i = p$, the variables $\hat{P}_{p+1} = Q_{x,N_p,p}$, $\hat{\Psi}_{p+1} = -l_{x,N_p,p}$ and $\hat{c}_{p+1} = c_{N_p,p}$ in (6.4) are known. Hence, in the last subproblem all variables but the initial state can be eliminated, giving the (trivial) optimization problem

$$\begin{aligned} & \underset{x_{0,p}}{\text{minimize}} && x_{0,p}^T \hat{Q}_{x,p} x_{0,p} + \hat{l}_{x,p}^T x_{0,p} + \hat{c}_p \\ & \text{subject to} && x_{0,p} = \hat{x}_p. \end{aligned} \quad (6.31)$$

Here $\hat{Q}_{x,p} \triangleq P_{0,p}$, $\hat{l}_{x,p} \triangleq -\Psi_{0,p}$ and $\hat{c}_p \triangleq \bar{c}_{0,p}$ are given by algorithms 4 and 5.

The formal validity of the reduction of each subproblem $i \in \mathbb{Z}_{0,p-1}$ is given by Theorem 6.7, while the computational procedure is summarized in Algorithm 13. This algorithm is basically a Riccati factorization and backward recursion as in algorithms 4 and 5, respectively. Algorithm 13 is used to eliminate the local variables in each subproblem $i \in \mathbb{Z}_{0,p-1}$. Here $\hat{Q}_{u,i}$ and $\hat{B}_i$ are computed as in (6.29).

6.1.3 Constructing the master problem

According to Theorem 6.7 and Section 6.1.2 all subproblems $i \in \mathbb{Z}_{0,p-1}$ can be reduced to UFTOC problems in the variables $x_{0,i}$, $x_{N_i,i}$ and $\hat{u}_i$ with initial state $\hat{x}_i$, and subproblem $i = p$ depends only on the initial state $\hat{x}_p$. The variable $\hat{u}_i$ represents the unknown part of the control signals $u_{t,i}$ that are due to the initially unknown $\hat{P}_{i+1}$ and $\hat{\Psi}_{i+1}$, and can be interpreted as a new control signal for batch i . By using the definition of the subproblems and $\hat{P}_{i+1} \triangleq P_{t_i+1}$, $\hat{\Psi}_{i+1} \triangleq \Psi_{t_i+1}$, $\hat{c}_{i+1} \triangleq c_{t_i+1}$ together with the relation $x_{N_i,i} = x_{0,i+1} = \hat{x}_{i+1}$ between the subproblems that were introduced in Section 6.1.1, the reduced subproblems $i \in \mathbb{Z}_{0,p}$ can be combined into a master problem. How to do this will now be shown.

From the introduction of Section 6.1 it is clear that the optimal solution in the first interval of the prediction horizon can be computed by solving the first subproblem, i.e., by solving the problem

$$\begin{aligned} & \underset{x_{0,0}, x_{N_0,0}, \hat{u}_0}{\text{minimize}} && \frac{1}{2} x_{0,0}^T \hat{Q}_{x,0} x_{0,0} + \frac{1}{2} \hat{u}_0^T \hat{Q}_{u,0} \hat{u}_0 + \hat{l}_{x,0}^T x_{0,0} + \hat{c}_0 + \\ & && \frac{1}{2} x_{N_0,0}^T \hat{P}_1 x_{N_0,0} - \hat{\Psi}_1^T x_{N_0,0} + \hat{c}_1 \\ & \text{subject to} && x_{0,0} = \hat{x}_0 = \bar{x} \\ & && x_{N_0,0} = \hat{A}_0 x_{0,0} + \hat{B}_0 \hat{u}_0 + \hat{a}_0, \end{aligned} \quad (6.32)$$

which can be done if $\hat{P}_1$, $\hat{\Psi}_1$ and $\hat{c}_1$ are known. However, these variables are not

Algorithm 13 Reduction using Riccati factorization

```

1:  $P_N := 0, \Psi_N := 0, \bar{c}_N := 0$ 
    $\hat{Q}_u := 0, D_N := I, d_N := 0$ 
2: for  $t = N - 1, \dots, 0$  do
3:    $F_{t+1} := Q_{x,t} + A_t^T P_{t+1} A_t$ 
4:    $G_{t+1} := Q_{u,t} + B_t^T P_{t+1} B_t$ 
5:    $H_{t+1} := Q_{xu,t} + A_t^T P_{t+1} B_t$ 
6:   Compute and store a factorization of  $G_{t+1}$ 
7:   Compute a solution  $K_{t+1}$  to
      $G_{t+1} K_{t+1} = -H_{t+1}^T$ 
8:   Compute a solution  $k_{t+1}$  to
      $G_{t+1} k_{t+1} = B_t^T \Psi_{t+1} - l_{u,t} - B_t^T P_{t+1} a_t$ 
9:    $\Psi_t := A_t^T \Psi_{t+1} - H_{t+1} k_{t+1} - l_{x,t} - A_t^T P_{t+1} a_t$ 
10:   $P_t := F_{t+1} - K_{t+1}^T G_{t+1} K_{t+1}$ 
11:  Compute a solution  $L_{t+1}$  to
      $G_{t+1} L_{t+1} = -B_t^T D_{t+1}$ 
12:   $D_t := (A_t^T + K_{t+1}^T B_t^T) D_{t+1}$ 
13:   $d_t := d_{t+1} + D_{t+1}^T (a_t + B_t k_{t+1})$ 
14:   $\hat{Q}_u := \hat{Q}_u + L_{t+1}^T G_{t+1} L_{t+1}$ 
15: end for
16:  $\hat{A} := D_0^T, \hat{B} := \hat{Q}_u, \hat{a} := d_0$ 
    $\hat{Q}_x := P_0, \hat{l}_x := -\Psi_0, \hat{c} := \bar{c}_0$ 

```

known for now, but by using the definition of the local variables in (6.3) it is clear that $x_{N_0,0} = x_{0,1}$. Hence, it is possible to compute the the terminal state cost in subproblem $i = 0$, i.e., the term

$$\frac{1}{2} x_{N_0,0}^T \hat{P}_1 x_{N_0,0} - \hat{\Psi}_1^T x_{N_0,0} + \hat{c}_1, \quad (6.33)$$

in (6.32) by instead computing

$$\frac{1}{2} x_{0,1}^T \hat{P}_1 x_{0,1} - \hat{\Psi}_1^T x_{0,1} + \hat{c}_1. \quad (6.34)$$

Similarly, from the introduction to Section 6.1 it can be seen that this can be done by solving the second subproblem

$$\begin{aligned}
& \underset{x_{0,1}, x_{N_1,1}, \hat{u}_1}{\text{minimize}} && \frac{1}{2} x_{0,1}^T \hat{Q}_1 x_{0,1} + \frac{1}{2} \hat{u}_1^T \hat{Q}_u \hat{u}_1 + \hat{l}_{x,1}^T x_{0,1} + \hat{c}_1 + \\
& && \frac{1}{2} x_{N_1,1}^T \hat{P}_2 x_{N_1,1} - \hat{\Psi}_2^T x_{N_1,1} + \hat{c}_2 \\
& \text{subject to} && x_{0,1} = \hat{x}_1 = x_{N_0,0} \\
& && x_{N_1,1} = \hat{A}_1 x_{0,1} + \hat{B}_1 \hat{u}_1 + \hat{a}_1,
\end{aligned} \quad (6.35)$$

using the optimal $\hat{x}_i = x_{x_{N_0,0}}, \hat{P}_2, \hat{\Psi}_2$ and $\hat{c}_2$. By utilizing these relations between the subproblems, the two subproblems (6.32) and (6.35) can be combined into

one optimization problem given by

$$\begin{aligned}
 & \underset{\hat{x}_0, \hat{u}_0, \hat{x}_1, \hat{u}_1, x_{N_1,1}}{\text{minimize}} && \sum_{i=0}^1 \left(\frac{1}{2} \hat{x}_i^T \hat{Q}_{x,i} \hat{x}_i + \frac{1}{2} \hat{u}_i^T \hat{Q}_{u,i} \hat{u}_i + \hat{l}_{x,i}^T \hat{x}_i + \hat{c}_i \right) + \\
 & && \frac{1}{2} x_{N_1,1}^T \hat{P}_2 x_{N_1,1} - \hat{\Psi}_2^T x_{N_1,1} + \hat{c}_2 \\
 & \text{subject to} && \hat{x}_0 = \bar{x} \\
 & && \hat{x}_1 = \hat{A}_0 \hat{x}_0 + \hat{B}_0 \hat{u}_0 + \hat{a}_0 \\
 & && x_{N_1,1} = \hat{A}_1 \hat{x}_1 + \hat{B}_1 \hat{u}_1 + \hat{a}_1.
 \end{aligned} \tag{6.36}$$

Here the local variables $x_{0,0}$, $x_{N_0,0}$ and $x_{0,1}$ have been trivially eliminated. The problem (6.36) can be solved given that $\hat{P}_2$, $\hat{\Psi}_2$ and $\hat{c}_2$ are known. Hence, by induction, all subproblems $i \in \mathbb{Z}_{0,p}$ can be combined into one master problem given by

$$\begin{aligned}
 & \underset{\bar{x}, \hat{u}}{\text{minimize}} && \sum_{i=0}^{p-1} \left(\frac{1}{2} \begin{bmatrix} \hat{x}_i \\ \hat{u}_i \end{bmatrix}^T \begin{bmatrix} \hat{Q}_{x,t} & 0 \\ 0 & \hat{Q}_{u,t} \end{bmatrix} \begin{bmatrix} \hat{x}_i \\ \hat{u}_i \end{bmatrix} + \hat{l}_{x,i}^T \hat{x}_i + \hat{c}_i \right) + \\
 & && \frac{1}{2} \hat{x}_p^T \hat{Q}_{x,p} \hat{x}_p + \hat{l}_{x,p}^T \hat{x}_p + \hat{c}_p \\
 & \text{subject to} && \hat{x}_0 = \bar{x}_0 \\
 & && \hat{x}_{i+1} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i, \quad i \in \mathbb{Z}_{0,p-1}.
 \end{aligned} \tag{6.37}$$

This problem is a UFTOC problem in the same form as (6.1) but with shorter prediction horizon $p < N$ and control signal dimension given by $n_{\hat{u}}$ in each sample i . The dynamics equations $\hat{x}_{i+1} = \hat{A}_i \hat{x}_i + \hat{B}_i \hat{u}_i + \hat{a}_i$ represent the relation $x_{N_i,i} = x_{0,i+1}$ between subproblem i and $i+1$. Hence, a UFTOC problem $\mathcal{P}(N)$ of prediction horizon length N can be reduced, while exploiting the structure by using Riccati recursions in each subproblem, to a UFTOC problem $\mathcal{P}(p)$ on the same form but with shorter prediction horizon and possibly lower control signal dimension. Figure 6.1 illustrates this reduction procedure, where the notation $\mathcal{P}_i(N_i)$ denotes subproblem i given by (6.4) with prediction horizon N_i .

Remark 6.9. The master problem (6.37) is in a similar form as the master problem in (5.14) in Chapter 5. However, here the use of the preliminary feedback results in no cross terms between $\hat{x}_i$ and $\hat{u}_i$ which is an advantage since it decreases the amount of data that needs to be communicated. _____

Remark 6.10. The columns of $\hat{B}_i$ in (6.22) form a basis for the reachability matrix S_i . However, the reachability matrix is invariant under the variable transformation (6.7). Hence, both $\hat{B}_i$ in (6.22) and $\hat{B}_i$ in Chapter 5 are bases for the same space. _____

Figure 6.1: The original MPC problem $\mathcal{P}(N)$ can be reduced to a smaller problem $\mathcal{P}(p)$ with the same structure but with shorter prediction horizon.

6.2 Computing the Riccati Recursion in Parallel

In Section 6.1 it was shown how to split the original UFTOC problem (6.1) into several individual subproblems, how to reduce these and how to construct a master problem by combining the individual reduced subproblems. Since all the subproblems are independent of each other, the reduction of these according to Section 6.1.2 to obtain the master problem (6.37) can be performed in parallel. To reach consensus between all subproblems and solve the original problem (6.1), the master problem $\mathcal{P}(p)$ can be solved to obtain $\hat{P}_{i+1}$, $\hat{\Psi}_{i+1}$, $\hat{c}_{i+1}$ and the optimal $\hat{x}_i$ for $i \in \mathbb{Z}_{0,p}$. When these variables are known, the independent subproblems are solved in parallel using algorithms 4-6 with the initial $x_{0,i} = \hat{x}_i$, $\hat{P}_{i+1}$, $\hat{\Psi}_{i+1}$ and $\hat{c}_{i+1}$ for $i \in \mathbb{Z}_{0,p}$. In this section it will be shown how to compute these variables in order to solve (6.1) in parallel.

6.2.1 Parallel computation of the Riccati recursion

To compute $\hat{P}_{i+1}$, $\hat{\Psi}_{i+1}$, $\hat{c}_{i+1}$ and $\hat{x}_i$, the master problem (6.37) can be solved serially using the Riccati recursion. However, similarly as in Chapter 5, the master problem can instead itself be reduced in parallel by repeatedly using the theory presented in Section 6.1 until a UFTOC problem with a prediction horizon that is of a pre-determined length is obtained. This top problem is solved, and the solution is propagated down until the subproblems of the original problem (6.1) are solved. This procedure is shown in Figure 6.2, where $\mathcal{P}_i^k(N_i^k)$ denotes subproblem i in the form (6.4) at level k in the tree. The overall procedure is similar to what was done in Chapter 5, but the conceptual difference here is the way it is performed using Riccati recursions, which exploits the structure also in the subproblems.

The parallel Riccati algorithm thus consists of two steps; reducing the original problem $\mathcal{P}(N)$ in parallel in several steps to the top problem $\mathcal{P}(p_{m-1})$, and propagating the solution down in the tree by solving all $\mathcal{P}_i^k(N_i^k)$ at each level in parallel. Since the subproblems at each level can be reduced and solved in parallel, and the information flow is between parent and children in the tree, the Riccati recursion can be computed in parallel using the theory proposed in this chapter.

Figure 6.2: The original UFTOC problem $\mathcal{P}(N)$ can be reduced repeatedly. The Riccati recursion is used to exploit the structure in the subproblems. When the solution to the top problem is computed, it can be propagated back in the tree until the bottom level is solved.

6.2.2 Parallel Riccati recursion algorithms

In this section algorithms for computing the Riccati recursion in parallel are presented. Beyond what is presented here, as observed already in Axehill and Hansson (2012), standard parallel linear algebra can be used for many computations in the serial Riccati recursion in each subproblem to boost performance even further. This has however not been utilized in this work.

In Algorithm 14, the original problem (6.1) is reduced in parallel in several steps to a UFTOC problem with pre-defined prediction horizon p_{m-1} . Assume, for simplicity, that all subproblems are of equal batch length N_s and that $N = N_s^{m+1}$ for some $1 \leq m \in \mathbb{Z}$. Then the reduction can be made in m steps, provided that N_s^m computational units are available. Hence, the reduction algorithm has $\mathcal{O}(\log N)$ complexity growth, which is the same as for the algorithm in Chapter 5.

Algorithm 14 Parallel reduction of UFTOC problem

- 1: Set the maximum level number m
 - 2: Set the number of subproblems $p_k + 1$ for each level $k \in \mathbb{Z}_{0,m}$, with $p_m = 0$
 - 3: **for** $k = 0, \dots, m - 1$ **do**
 - 4: **parfor** $i = 0, \dots, p_k$ **do**
 - 5: Create subproblem $\mathcal{P}_i^k(N_i^k)$
 - 6: Reduce subproblem $\mathcal{P}_i^k(N_i^k)$ using Algorithm 13
 - 7: Send $\hat{A}_i^k, \hat{B}_i^k, \hat{a}_i^k, \hat{Q}_{x,i}^k, \hat{Q}_{u,i}^k, \hat{l}_{x,i}^k$ and $\hat{c}_i^k$ to parent
 - 8: **end parfor**
 - 9: **end for**
-

When the reduction step is complete, each subproblem $i \in \mathbb{Z}_{0,p_{k-1}}$ at level k is solved in Algorithm 15 using algorithms 4-6 with the optimal $\hat{x}_i^{k+1}, \hat{p}_{i+1}^{k+1}, \hat{\Psi}_{i+1}^{k+1}$

and $\hat{c}_{i+1}^{k+1}$ from the respective parent. The algorithm starts by solving the top problem $\mathcal{P}(p_{m-1})$ in Figure 6.2, and this solution is propagated to its children. By solving the subproblems at each level and propagating the solution to the level below in the tree, the subproblems $\mathcal{P}_i^0(N_i^0)$ for $i \in \mathbb{Z}_{0,p_0}$ at the bottom level can finally be solved individually. All subproblems can be solved using only information from their parents, and hence each level in the tree can be solved completely in parallel. The optimal primal solution to the original UFTOC problem (6.1) can be constructed from the solutions to the subproblems using the definition of the local variables in (6.3). The dual variables can be computed from all $P_{t,i}^0$ and $\Psi_{t,i}^0$ from the subproblems at the bottom level using the relation (3.48), and hence there are no complications with non-unique dual variables as in Chapter 5 when using the algorithm presented in this chapter. The propagation of the solution from the top level to the bottom level can be made in $m + 1$ steps provided that N_s^m processing units are available. Since both algorithms 14 and 15 are solved in $\mathcal{O}(\log N)$ complexity, the Riccati recursion and the solution to the UFTOC problem (6.1) can be computed with $\mathcal{O}(\log N)$ complexity growth.

Algorithm 15 Parallel solution of UFTOC problem

- 1: Get the maximum level number m and all p_k :s from Algorithm 14
 - 2: Initialize $\hat{x}_0^m := \bar{x}$
 - 3: **for** $k = m, m - 1, \dots, 0$ **do**
 - 4: **parfor** $i = 0, \dots, p_k$ **do**
 - 5: Solve subproblem $\mathcal{P}_i^k(N_i^k)$ using algorithms 4-6
 - 6: **if** $k > 0$ **then**
 - 7: Send $\hat{P}_{t,i}^k, \hat{\Psi}_{t,i}^k, \hat{c}_{t,i}^k$ and $\hat{x}_{t,i}^k$ to each children at level $k - 1$
 - 8: **end if**
 - 9: **end parfor**
 - 10: **end for**
 - 11: Retrieve the solution of (6.1) from the solutions of $\mathcal{P}_i^0(N_i^0)$ for $i \in \mathbb{Z}_{0,p_0}$
-

Remark 6.11. At Line 5 in Algorithm 15 the dual variables of $\mathcal{P}_i^k(N_i^k)$ (λ_t in Algorithm 6) are only computed if $k = 0$, since at levels $k > 0$ the dual variables are not used. _____

If the algorithms are implemented in double precision arithmetics there are 8 bytes of data per number. Hence, on Line 7 in Algorithm 14 and Line 7 in Algorithm 15 the number of sent bytes is $16n_x^2 + 24n_x + 8$ and $4n_x^2 + 20n_x + 8$, respectively. This can be compared to the algorithms in Chapter 5 where $32n_x^2 + 32n_x + 8$ and $24n_x$ bytes of data are sent. The significant decrease in the amount of communication in the algorithm proposed here is due to exploitation of the structure in the subproblems by using Riccati recursions and the fact that $\hat{B}_i = \hat{Q}_{u,i}$.

Remark 6.12. If the parallel algorithm is used to compute UFTOC problems in an AS method, the eliminated dual variables must be computed as in Section 3.5 using Algorithm 7. Given the primal and dual solutions to the UFTOC problem, it can be seen that this algorithm can be computed in parallel for the independent intervals $i \in \mathbb{Z}_{0,p}$. _____

6.3 Numerical Results

The algorithms presented in Section 6.2.2 have been implemented in MATLAB for development and comparison purposes, and in ANSI-C for evaluation on a computational cluster. In MATLAB parallel executions of the algorithms are simulated by executing them serially using one computational thread, but still using the same information flow as for an actual parallel execution. The total computation time in MATLAB has been estimated by summing over the maximum computation time for each level in the tree, and hence the communication overhead is neglected. The influence of the communication overhead is discussed in the end of this section. The performance of the parallel Riccati algorithm proposed in this chapter is compared to both the serial Riccati recursion, which is considered a state-of-the-art serial method, and the parallel algorithm presented in Chapter 5 (only in MATLAB). The numerical results show that the computation time can be significantly decreased when using the parallel Riccati recursion proposed in this chapter compared to the serial Riccati and the parallel algorithm presented in Chapter 5. In all results presented in this section $N_s = 2$ has been used in all subproblems.

Remark 6.13. It is possible to choose different batch lengths for each subproblem and level in the tree. How to make these choices to minimize computation time is not investigated here. However, similarly to what is described in Axehill (2015), the optimal choice depends on, e.g., the problem and the hardware on which the algorithm is implemented.

In MATLAB the algorithms have been compared when solving problems in the form (6.1) for systems of dimension $n_x = 7$ and $n_u = 5$, see Figure 6.3, and of dimension $n_x = 20$ and $n_u = 20$, see Figure 6.4. From the figures it is clear that the parallel Riccati algorithm outperforms the serial Riccati for $N \gtrsim 18 - 20$ for both problem sizes. The parallel Riccati algorithm is outperforming the parallel algorithm in Chapter 5 for both problem sizes, and for the problem in Figure 6.4 the parallel Riccati algorithm is approximately three times faster than the parallel algorithm in Chapter 5. This significant boost in performance is possible since the algorithm proposed in this chapter utilizes the structure in the subproblems.

In addition to the MATLAB implementation, an ANSI-C implemented algorithm has been run on a computational cluster consisting of nodes with 8-core Intel Xeon E5-2660 @ 2.2 GHz CPUs and the communication is performed over TCP/IP on Gigabit Ethernet between the nodes, and over the TCP/IP loop-back for communication between different cores on the same node. The computations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at NSC. The implementation is rudimentary and especially the communication setup can be improved. However, the implemented algorithm serves as a proof-of-concept that the algorithm works and improves performance in terms of computation times for computations on real parallel hardware, taking communication delays into account. The computation times when solving problems in the form (6.1) for systems of order $n_x = 20$ and $n_u = 20$ are seen

Figure 6.3: Computation times using systems of order $n_x = 7$ and $n_u = 5$. The parallel Riccati outperforms the serial Riccati for $N \gtrsim 18 - 20$, and it is always faster than the parallel algorithm proposed in Chapter 5.

in Figure 6.5, where it is clear that the parallel algorithm solves a problem with $N = 512$ approximately as fast as the serial algorithm solves the same one for $N = 64$, and the break even is at $N \approx 28$. This computational speed-up can be important in problems that require long prediction horizons like, e.g., moving horizon estimation problems (Rao et al., 1998) or high-level planners for autonomous vehicles.

The communication overhead is approximately 20% for this problem size, and it has been observed that communication times are approximately the same regardless of the problem size which indicates that there is a significant latency in the communication channel. Reducing these latencies, by for example a better communication setup and using more suitable hardware for fast inter process communication, can significantly improve performance of the ANSI-C implemented algorithm, especially for small problem dimensions.

Remark 6.14. Experimental results have shown that in communication channels with high latencies the total communication time is affected more by the number of times that the nodes communicate with each other than the amount of data sent at each instant. Hence, for such communication networks, algorithms that send more data fewer times might perform better in terms of computation time than methods that send small amounts of data frequently.

Figure 6.4: Computation times using systems of order $n_x = 20$ and $n_u = 20$. The parallel Riccati outperforms the serial Riccati for $N \gtrsim 18 - 20$, and it is significantly faster than the parallel algorithm in Chapter 5.

Figure 6.5: Computation times for the ANSI-C implementation using systems of order $n_x = 20$ and $n_u = 20$. The communication overhead is approximately 20% of the total computation time for this problem size.

6.4 Comparison with Chapter 5

The basic idea with both the algorithm presented in this chapter and in Chapter 5 is to split the prediction horizon into several smaller parts, and performing computations in each part in parallel to reduce the overall computation time. Hence, the two algorithms share similarities but there are also important differences. In this section a brief comparison of the two algorithms is presented.

In the algorithm in Chapter 5, the original UFTOC problem (5.1) is explicitly extended to the problem (5.4) by introducing the parameters $\hat{x}_i$ and $\hat{u}_i$ as optimization variables together with extra equality constraints. The terminal constraint in each subproblem is parameterized by $\hat{x}_i$ and $\hat{u}_i$, and the parametrization is chosen such that only reachable terminal states are obtained for all values of the parameters. This is done by choosing $\hat{B}_i$ in (5.8) as *any* basis for the reachability matrix (5.7). The value function is then computed by solving the subproblem parametrically using a standard nullspace method as in Tøndel et al. (2003). By combining the value functions from all individual subproblems with the extra equality constraints in the extended problem (5.4), the extended problem is reduced to the master problem (5.14). This master problem is a UFTOC problem in the parameters $\hat{x}_i$ and $\hat{u}_i$. After the master problem has been solved, the optimal $\hat{x}_i^*$ and $\hat{u}_i^*$ are used in the subproblems to compute the optimal solution.

In the algorithm in this chapter no extra constraints are imposed on the terminal states in the subproblems, but instead a variable terminal cost is used to incorporate information from other batches. The states in the subproblem can be eliminated in a similar way as in partial condensing which is introduced in Axehill (2015), and the control signals can be parametrized by the parameter $\hat{u}_i$ by exploiting the structure in the subproblem. This parametrization computes $\hat{B}_i$ in (6.22) as a specific basis for the reachability matrix S_i in this chapter. The value function of a subproblem can be computed as a function of the initial value of each subproblem and the parameter $\hat{u}_i$ by using the Riccati recursion as in Lemma 6.2. When the states and control signals have been eliminated in the subproblems, these reduced problems can be combined into a new UFTOC problem by using the definition of the splitting of the prediction horizon (as described in Section 6.1.3). When the new UFTOC problem has been solved, the optimal $\hat{x}_i^*$, $\hat{P}_i$ and $\hat{\Psi}_i$ are used in the individual subproblems to compute the optimal primal and dual solutions.

Hence, in both algorithms the freedom in the subproblems are parameterized such that only reachable terminal states can be obtained in the master problem, and the value function is computed as a function of the parameters. However, the parametrization is introduced in different ways; in Chapter 5 the freedom is introduced as a freedom in the terminal state in each subproblem, whereas it is described as a freedom in the terminal cost in each subproblem in this chapter. The algorithm in this chapter exploits the structure in the subproblems by eliminating variables using a preliminary feedback that is computed with a Riccati recursion. Furthermore, it exploits the structure also by solving the subproblems

using Riccati recursions. The algorithm in Chapter 5 does not exploit any structure in the subproblem when the parametric solution is computed.

The extra constraint on the terminal state in the subproblems in Chapter 5 might introduce primal degeneracy, which results in non-unique dual variables. This was solved by choosing dual variables that satisfy the optimality condition of the original UFTOC problem. In the algorithm in this section this problem is evaded, and primal degeneracy does not appear in the subproblems. Here, the dual variables are computed using the uniquely defined $P_{t,i}$ and $\Psi_{t,i}$ for each subproblem. Furthermore, the information that is used by the subproblems to compute the solution is different in the two algorithms; in Chapter 5 the optimal parameters are used, and in this chapter the optimal initial parameter $\hat{x}_i^*$, and $\hat{P}_i$ and $\hat{\Psi}_i$ are used to compute the feedback matrices and optimal solution in the subproblems.

The numerical results in Section 6.3 confirms that the MATLAB implementations of the algorithm proposed in this chapter outperforms the algorithm in Chapter 5 in terms of computation time. In a truly parallel implementation the difference would probably be even larger, since more data is communicated in the algorithm in Chapter 5.

Appendix

In this appendix the proof of Lemma 6.2 is given.

6.A Proof of Lemma 6.2

The subindices i are omitted for notational brevity in the proof.

Proof: Assume that (6.6) holds for an arbitrary $t + 1 \in \mathbb{Z}_{1,N-1}$. Then the cost at time t is given by

$$\frac{1}{2} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^T \begin{bmatrix} Q_{x,t} & Q_{xu,t} \\ Q_{xu,t}^T & Q_{u,t} \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \begin{bmatrix} l_{x,t} \\ l_{u,t} \end{bmatrix}^T \begin{bmatrix} x_t \\ u_t \end{bmatrix} + c_t + \bar{V}(x_{t+1}, \bar{u}). \quad (6.38)$$

Inserting the expression of $\bar{V}(x_{t+1}, \bar{u})$ from (6.6) and $x_{t+1} = A_t x_t + B_t u_t + a_t$ into (6.38) gives

$$\frac{1}{2} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^T \begin{bmatrix} F_{t+1} & H_{t+1} \\ H_{t+1}^T & G_{t+1} \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \begin{bmatrix} \bar{l}_{x,t} \\ \bar{l}_{u,t} \end{bmatrix}^T \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \bar{c}_t + \frac{1}{2} \sum_{t=t+1}^{N-1} \bar{u}_t^T G_{t+1} \bar{u}_t, \quad (6.39)$$

where F_{t+1} , H_{t+1} and G_{t+1} are defined by (3.51a), (3.51b) and (3.51c), respectively, and

$$\begin{aligned} \bar{l}_{x,t} &\triangleq l_{x,t} + A_t^T P_{t+1} a_t - A_t^T \Psi_{t+1}, \\ \bar{l}_{u,t} &\triangleq l_{u,t} + B_t^T P_{t+1} a_t - B_t^T \Psi_{t+1}, \\ \bar{c}_t &\triangleq \bar{c}_{t+1} + c_t + \frac{1}{2} a_t^T P_{t+1} a_t - \Psi_{t+1}^T a_t. \end{aligned} \quad (6.40)$$

Finally, using the control law $u_t = k_{t+1} + K_{t+1} x_t + \bar{u}_t$ and the definition of P_t , Ψ_t , k_{t+1} and K_{t+1} from algorithms 4 and 5, together with $\bar{l}_{x,t}$ and $\bar{l}_{u,t}$ gives the cost function

$$\bar{V}(x_t, \bar{u}) = \frac{1}{2} x_t^T P_t x_t - \Psi_t^T x_t + \frac{1}{2} \sum_{t=t}^{N-1} \bar{u}_t^T G_{t+1} \bar{u}_t + \bar{c}_t, \quad (6.41)$$

where

$$\bar{c}_t \triangleq \tilde{c}_t + \frac{1}{2} k_{t+1}^T G_{t+1} k_{t+1} + \bar{l}_{u,t}^T k_{t+1}. \quad (6.42)$$

Note that the linear terms for $\bar{u}_t$ and the cross terms between x_t and $\bar{u}$ in the cost function (6.39) vanishes since $G_{t+1} k_{t+1} = -\bar{l}_{u,t}$ and $G_{t+1} K_{t+1} = -H_{t+1}^T$. Equation (6.6) holds specifically for $t = N - 1$ and hence Lemma 6.2 follows by induction. $\square$

7

Conclusions and Further Work

The aim of this thesis is to present numerical algorithms for important subroutines in solvers for MPC problems. Developing a complete solver is outside the scope of the thesis, and the focus is instead on the linear algebra that is used in the solvers. The performance gains in terms of computation time are evaluated using mostly MATLAB implementations of the proposed algorithms. No comparisons of complete solvers are made due to the limitation of the scope of the thesis.

7.1 Conclusions

In this thesis, different numerical algorithms for computing the Newton steps in both IP and AS methods for MPC are presented. This is often where the most computational effort is spent in these types of methods. It is shown in MATLAB evaluations that the performance in terms of computational complexity and computation time can be significantly reduced in an AS solver by modifying the Riccati factorization between changes to the working set instead of re-computing it. This results in quadratic complexity growth in the problem size instead of cubic, which can be a crucial decrease in complexity in order to solve large MPC problems when the structure is exploited using Riccati recursions.

Furthermore, it is shown how to compute the Newton step in parallel on several computational units by exploiting the structure in the MPC problem in time. Two different approaches are presented, where both exploit the special structure in the MPC problem. The first approach adds additional complicating variables and constraints and divides the MPC problem into several smaller subproblems. These are solved parametrically in parallel to obtain a master MPC problem with shorter prediction horizon. By repeating this procedure, the original problem can

be solved with logarithmic complexity in the prediction horizon length, provided that enough computational units are available. The second approach is similar to the first, but the way the subproblems are solved is different. Here the Riccati recursion is used to solve the MPC problem, and it is shown how to compute the Riccati recursion in parallel by dividing the prediction horizon into several smaller intervals. In each interval most of the variables are eliminated, and the remaining variables are combined into a new MPC problem. As for the first algorithm, this procedure can be repeated to obtain MPC problems with shorter prediction horizons, which results in logarithmic complexity growth in the prediction horizon length, provided that enough computational units are available.

7.2 Further Work

In this thesis, contributions to the area of numerical algorithms for MPC are presented, and some possibilities for future work in this field are presented here. There are interesting and challenging theoretical and practical extensions of the work presented in this thesis. Some of the interesting topics for future work are listed below

- Using the results from Chapter 4 in the dual gradient projection method presented in Axehill (2008).
- Implementing the algorithms presented in Chapter 4 in ANSI-C.
- Develop a square-root implementation of the algorithms in Chapter 4.
- Including the results in chapters 5 and 6 in a complete IP or AS solver to further utilize structure and parallel computations.
- Extending the results to cover also estimation and system identification problems.
- Combining the theory in chapters 4 and 6 to perform modifications of the Riccati factorization in parallel.
- Investigate how the size of the subproblems in chapters 5 and 6 should be chosen to minimize the total computation time.
- Improving the ANSI-C implementation of the parallel Riccati recursion in Chapter 6.
- Reducing the communication latencies in the parallel algorithms by using suitable parallel hardware such as Graphics Processing Units (GPUs) or Field Programmable Gate Arrays (FPGAs).

Appendix

A

Linear Algebra

In this appendix, some linear algebra notation and results that are used throughout the thesis are listed. For further results in linear algebra, the reader is referred to standard literature such as Lancaster and Tismenetsky (1985).

Definition A.1 (Moore-Penrose pseudo-inverse). Consider a matrix $A \in \mathbb{R}^{m \times n}$. Then the matrix $A^\dagger \in \mathbb{R}^{n \times m}$ that satisfies the four equations

$$AA^\dagger A = A, \tag{A.1}$$

$$A^\dagger AA^\dagger = A^\dagger, \tag{A.2}$$

$$(AA^\dagger)^T = AA^\dagger, \tag{A.3}$$

$$(A^\dagger A)^T = A^\dagger A, \tag{A.4}$$

is the *Moore-Penrose pseudo-inverse*, or simply the *pseudo-inverse* of A . Furthermore, $A^\dagger$ is unique. _____

Remark A.2. If A is non-singular, then $A^\dagger = A^{-1}$. _____

Now study a square matrix M which is partitioned into blocks as

$$M \triangleq \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}, \tag{A.5}$$

where M_{11} and M_{22} are square matrices.

Definition A.3 is repeated from Carlson et al. (1974), but with slightly different notation.

Definition A.3 (Generalized Schur complement). The *generalized Schur complement* of M_{22} in M is denoted $M/^\dagger M_{22}$ and is defined as

$$M/^\dagger M_{22} \triangleq M_{11} - M_{12}M_{22}^\dagger M_{21}. \quad (\text{A.6})$$

Similarly, the generalized Schur complement of M_{11} in M is defined as

$$M/^\dagger M_{11} \triangleq M_{22} - M_{21}M_{11}^\dagger M_{12}. \quad (\text{A.7})$$

When M_{22} , or M_{11} , is non-singular, the Schur complement is defined as in Definition A.4.

Definition A.4 (Schur complement). The *Schur complement* of M_{22} in M is denoted M/M_{22} and is defined as

$$M/M_{22} \triangleq M_{11} - M_{12}M_{22}^{-1}M_{21}. \quad (\text{A.8})$$

Similarly, the Schur complement of M_{11} in M is defined as

$$M/M_{11} \triangleq M_{22} - M_{21}M_{11}^{-1}M_{12}. \quad (\text{A.9})$$

Assumption A.5. The matrix M in (A.5) is non-singular.

Theorem A.6. Let Assumption A.5 hold. Then the inverse of M in (A.5) is

$$M^{-1} = \begin{bmatrix} M_{11}^{-1} + M_{11}^{-1}M_{12}(M/M_{11})^{-1}M_{21}M_{11}^{-1} & -M_{11}^{-1}M_{12}(M/M_{11})^{-1} \\ -(M/M_{11})^{-1}M_{21}M_{11}^{-1} & (M/M_{11})^{-1} \end{bmatrix}. \quad (\text{A.10})$$

Proof: Write M as

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} = \begin{bmatrix} I & 0 \\ M_{21}M_{11}^{-1} & I \end{bmatrix} \begin{bmatrix} M_{11} & 0 \\ 0 & M/M_{11} \end{bmatrix} \begin{bmatrix} I & M_{11}^{-1}M_{12} \\ 0 & I \end{bmatrix}, \quad (\text{A.11})$$

where the outer matrices on the right-hand-side are invertible. Taking the inverse of both sides gives

$$\begin{aligned} \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}^{-1} &= \begin{bmatrix} I & -M_{11}^{-1}M_{12} \\ 0 & I \end{bmatrix} \begin{bmatrix} M_{11}^{-1} & 0 \\ 0 & (M/M_{11})^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -M_{21}M_{11}^{-1} & I \end{bmatrix} = \\ &= \begin{bmatrix} M_{11}^{-1} + M_{11}^{-1}M_{12}(M/M_{11})^{-1}M_{21}M_{11}^{-1} & -M_{11}^{-1}M_{12}(M/M_{11})^{-1} \\ -(M/M_{11})^{-1}M_{21}M_{11}^{-1} & (M/M_{11})^{-1} \end{bmatrix}, \end{aligned} \quad (\text{A.12})$$

which completes the proof. $\square$

Assumption A.7. The matrix M is symmetric, giving

$$M_{21} = M_{12}^T, \quad M_{11} = M_{11}^T, \quad M_{22} = M_{22}^T. \quad (\text{A.13})$$

The following theorem is repeated from Albert (1969).

Theorem A.8. Consider a block partitioned matrix M in (A.5) where Assumption A.7 holds. Then the following hold

$$M \geq 0 \iff M_{11} \geq 0, (I - M_{11}M_{11}^\dagger)M_{12} = 0, M/\dagger M_{11} = M_{22} - M_{12}^T M_{11}^\dagger M_{12} \geq 0, \quad (\text{A.14})$$

$$M \geq 0 \iff M_{22} \geq 0, (I - M_{22}M_{22}^\dagger)M_{12}^T = 0, M/\dagger M_{22} = M_{11} - M_{12}M_{22}^\dagger M_{12}^T \geq 0. \quad (\text{A.15})$$

Proof: The proof follows from Theorem 1 in Albert (1969). $\square$

Now consider the system of equations

$$\underbrace{\begin{bmatrix} A_1 & A_2 \\ A_2^T & A_3 \end{bmatrix}}_A \underbrace{\begin{bmatrix} X_1 \\ X_2 \end{bmatrix}}_X = \underbrace{\begin{bmatrix} H_1 \\ H_2 \end{bmatrix}}_H. \quad (\text{A.16})$$

Assumption A.9. The system of equations given in (A.16) is solvable, i.e.,

$$H \in \mathcal{R}(A) \iff AA^\dagger H = H. \quad (\text{A.17})$$

Assumption A.10. $A \geq 0$.

Theorem A.11. Consider the system of equations in (A.16) under assumptions A.9 and A.10. Then one solution X^* is given by

$$\begin{bmatrix} X_1^* \\ X_2^* \end{bmatrix} = \begin{bmatrix} A_1^\dagger + A_1^\dagger A_2 (A/\dagger A_1)^\dagger A_2^T A_1^\dagger & -A_1^\dagger A_2 (A/\dagger A_1)^\dagger \\ -(A/\dagger A_1)^\dagger A_2^T A_1^\dagger & (A/\dagger A_1)^\dagger \end{bmatrix} \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}, \quad (\text{A.18})$$

and the following properties hold

$$(H_1 - A_2 A_3^\dagger H_2) \in \mathcal{R}(A_1 - A_2 A_3^\dagger A_2^T), \quad (\text{A.19})$$

$$(H_2 - A_2^T A_1^\dagger H_1) \in \mathcal{R}(A_3 - A_2^T A_1^\dagger A_2). \quad (\text{A.20})$$

Proof: Factor A as

$$\underbrace{\begin{bmatrix} A_1 & A_2 \\ A_2^T & A_3 \end{bmatrix}}_{\triangleq U} = \underbrace{\begin{bmatrix} I & 0 \\ A_2^T A_1^\dagger & I \end{bmatrix}}_{\triangleq \bar{A}} \underbrace{\begin{bmatrix} A_1 & 0 \\ 0 & A/\dagger A_1 \end{bmatrix}}_{\triangleq \bar{A}} \begin{bmatrix} I & A_1^\dagger A_2 \\ 0 & I \end{bmatrix} = U \bar{A} U^T. \quad (\text{A.21})$$

Note that U is invertible and

$$U^{-1} = \begin{bmatrix} I & 0 \\ -A_2^T A_1^\dagger & I \end{bmatrix}. \quad (\text{A.22})$$

Using this factorization, re-write the system of equations (A.16) as

$$AX = \begin{bmatrix} I & 0 \\ A_2^T A_1^\dagger & I \end{bmatrix} \begin{bmatrix} A_1 & 0 \\ 0 & A/\dagger A_1 \end{bmatrix} \begin{bmatrix} I & A_1^\dagger A_2 \\ 0 & I \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}. \quad (\text{A.23})$$

Since U is invertible, $\mathcal{R}(A) = \mathcal{R}(\tilde{A})$ and the system of equations

$$\begin{bmatrix} A_1 & 0 \\ 0 & A/\dagger A_1 \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}, \quad (\text{A.24})$$

with

$$Z \triangleq U^T X = \begin{bmatrix} X_1 + A_1^\dagger A_2 X_2 \\ X_2 \end{bmatrix}, \quad W \triangleq U^{-1} H = \begin{bmatrix} H_1 \\ -A_2^T A_1^\dagger H_1 + H_2 \end{bmatrix}, \quad (\text{A.25})$$

is solvable. The solution Z^* is

$$\begin{bmatrix} Z_1^* \\ Z_2^* \end{bmatrix} = \begin{bmatrix} A_1^\dagger W_1 + \nu_1 \\ (A/\dagger A_1)^\dagger W_2 + \nu_2 \end{bmatrix}, \quad \nu_1 \in \mathcal{N}(A_1), \quad \nu_2 \in \mathcal{N}(A/\dagger A_1). \quad (\text{A.26})$$

By using the definition of Z and W this gives

$$\begin{bmatrix} X_1^* \\ X_2^* \end{bmatrix} = \begin{bmatrix} A_1^\dagger + A_1^\dagger A_2 (A/\dagger A_1)^\dagger A_2^T A_1^\dagger & -A_1^\dagger A_2 (A/\dagger A_1)^\dagger \\ -(A/\dagger A_1)^\dagger A_2^T A_1^\dagger & (A/\dagger A_1)^\dagger \end{bmatrix} \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} + \begin{bmatrix} I & -A_1^\dagger A_2 \\ 0 & I \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix}. \quad (\text{A.27})$$

Hence, by choosing the solution corresponding to $\nu = 0$ it is clear that one solution to (A.16) is given by (A.18).

Furthermore, (A.20) follows from the fact that (A.24) is solvable. Analogous calculation proves (A.19). This concludes the proof. $\square$

Remark A.12. Note that the solution (A.18) is not in general the same as $X^* = A^\dagger H$. The requirements for them to coincide are listed in for example Burns et al. (1974). $\square$

Bibliography

- M. Åkerblad and A. Hansson. Efficient solution of second order cone program for model predictive control. *International Journal of Control*, 77(1):55–77, 2004. Cited on page 37.
- A. Albert. Conditions for positive and nonnegative definiteness in terms of pseudoinverses. *SIAM Journal on Applied Mathematics*, 17(2):434–440, 1969. doi: 10.1137/0117041. Cited on page 115.
- D. Axehill. *Applications of Integer Quadratic Programming in Control and Communication*. Number 1218 in Linköping Studies in Science and Technology. Thesis. 2005. Cited on pages 29, 46, 47, and 57.
- D. Axehill. *Integer Quadratic Programming for Control and Communication*. PhD thesis, Linköping University, 2008. Cited on pages 13, 20, 21, 29, 33, 37, 38, 47, 48, 49, 95, and 110.
- D. Axehill. Controlling the level of sparsity in MPC. *Systems & Control Letters*, 76:1–7, 2015. Cited on pages 77, 92, 101, and 105.
- D. Axehill and A. Hansson. A mixed integer dual quadratic programming algorithm tailored for MPC. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5693–5698, San Diego, USA, December 2006. Cited on pages 20, 37, 47, 48, and 49.
- D. Axehill and A. Hansson. A dual gradient projection quadratic programming algorithm tailored for model predictive control. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 3057–3064, Cancun, Mexico, 2008. Cited on pages 20, 33, 37, 47, 48, and 49.
- D. Axehill and A. Hansson. Towards parallel implementation of hybrid MPC – a survey and directions for future research. In Rolf Johansson and Anders Rantzer, editors, *Distributed Decision Making and Control*, volume 417 of *Lecture Notes in Control and Information Sciences*, pages 313–338. Springer Verlag, 2012. Cited on page 99.

- D. Axehill, A. Hansson, and L. Vandenberghe. Relaxations applicable to mixed integer predictive control – comparisons and efficient computations. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 4103–4109, New Orleans, USA, 2007. Cited on pages 22, 25, and 27.
- D. Axehill, L. Vandenberghe, and A. Hansson. Convex relaxations for mixed integer predictive control. *Automatica*, 46(9):1540–1545, 2010. Cited on pages 20 and 37.
- R.A. Bartlett and L.T. Biegler. Qpschur: A dual, active-set, schur-complement method for large-scale and structured convex quadratic programming. *Optimization and Engineering*, 7(1):5–32, 2006. ISSN 1389-4420. doi: 10.1007/s11081-006-6588-z. Cited on page 49.
- A. Bemporad, A. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3 – 20, 2002. ISSN 0005-1098. doi: 10.1016/S0005-1098(01)00174-1. Cited on page 19.
- U. Bertelè and F. Brioschi. On non-serial dynamic programming. *Journal of Combinatorial Theory, Series A*, 14(2):137 – 148, 1973. ISSN 0097-3165. doi: 10.1016/0097-3165(73)90016-2. Cited on page 66.
- D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000. ISBN 1886529094. Cited on page 66.
- S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. Cited on pages 6, 7, 8, 9, 15, 16, 39, and 60.
- S. Boyd, L. Xiao, A. Mutapcic, and J. Mattingley. Notes on decomposition methods. Technical report, Stanford University, 2008. URL http://see.stanford.edu/materials/lsocoe364b/\08-decomposition_notes.pdf. Cited on page 69.
- F. Burns, D. Carlson, E. Haynsworth, and T. Markham. Generalized inverse formulas using the schur complement. *SIAM Journal on Applied Mathematics*, 26(2):pp. 254–259, 1974. ISSN 00361399. Cited on page 116.
- D. Carlson, E. Haynsworth, and T. Markham. A generalization of the schur complement by means of the moore-penrose inverse. *SIAM Journal on Applied Mathematics*, 26(1):pp. 169–175, 1974. ISSN 00361399. Cited on pages 39, 56, and 113.
- G.A. Constantinides. Tutorial paper: Parallel architectures for model predictive control. In *Proceedings of the European Control Conference*, pages 138–143, Budapest, Hungary, 2009. Cited on page 65.
- M. Diehl, H.J. Ferreau, and N. Haverbeke. *Nonlinear Model Predictive Control*, chapter Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation, pages 391–417. Springer Berlin / Heidelberg, 2009. Cited on pages 20 and 37.

- D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27(1):1–33, 1983. ISSN 0025-5610. doi: 10.1007/BF02591962. Cited on pages 14, 47, and 49.
- G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996. ISBN 9780801854149. Cited on pages 29 and 45.
- A. Grancharova, T.A. Johansen, J. Kocijan, and D. Vrancic. Design of reduced dimension explicit model predictive controller for a gas-liquid separation plant. In *EUROCON 2003. Computer as a Tool. The IEEE Region 8*, volume 1, pages 391–395 vol.1, Sept 2003. doi: 10.1109/EURCON.2003.1248050. Cited on page 19.
- A. Hansson. A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE Transactions on Automatic Control*, 45(9): 1639–1655, September 2000. Cited on pages 20, 22, and 37.
- H. Jonson. *A Newton method for solving non-linear optimal control problems with general constraints*. PhD thesis, Linköpings Tekniska Högskola, 1983. Cited on pages 20, 29, and 37.
- J.B. Jørgensen. *Moving Horizon Estimation and Control*. PhD thesis, Technical University of Denmark (DTU), 2004. Cited on page 29.
- T. Kailath. *Linear Systems*. Information and System Sciences Series. Prentice-Hall, 1980. ISBN 9780135369616. Cited on page 52.
- S. Khoshfetrat Pakazad, A. Hansson, and M.S. Andersen. Distributed primal-dual interior-point methods for solving loosely coupled problems using message passing. *arXiv:1502.06384v2*, 2015. Cited on page 66.
- C. Kirches, L. Wirsching, S. Sager, and H.G. Bock. Efficient numerics for non-linear model predictive control. In M. Diehl, F. Glineur, E. Jarlebring, and W. Michiels, editors, *Recent Advances in Optimization and its Applications in Engineering*, pages 339–359. Springer, 2010. ISBN 978-3-6421-2597-3. Cited on page 37.
- C. Kirches, H.G. Bock, J.P. Schlöder, and S. Sager. A factorization with update procedures for a KKT matrix arising in direct optimal control. *Mathematical Programming Computation*, 3(4):319–348, 2011. Cited on page 37.
- C.D. Laird, A.V. Wong, and J. Akesson. Parallel solution of large-scale dynamic optimization problems. In *21st European Symposium on Computer Aided Process Engineering*, volume 21, 2011. Cited on page 65.
- P. Lancaster and M. Tismenetsky. *The Theory of Matrices: With Applications*. Academic Press, 1985. ISBN 9780124355606. Cited on page 113.
- L. Lasdon. *Optimization theory for large systems*. DoverPublications. com, 1970. Cited on page 69.

- J.M. Maciejowski. *Predictive control with constraints*. Prentice Hall, 2002. Cited on pages 1, 19, and 46.
- C.C. Moallemi. *A message-passing paradigm for optimization*. PhD thesis, Stanford University, 2007. Cited on page 66.
- I. Nielsen and D. Axehill. An $O(\log N)$ parallel algorithm for Newton step computation in model predictive control. In *Proceedings of the 19th IFAC World Congress*, pages 10505–10511, Cape Town, South Africa, August 2014. Cited on pages 20, 65, and 66.
- I. Nielsen and D. Axehill. A parallel structure-exploiting factorization algorithm with applications to model predictive control. Accepted for publication at the 54th IEEE Conference on Decision and Control, December 2015. Cited on page 87.
- I. Nielsen, D. Ankelhed, and D. Axehill. Low-rank modification of Riccati factorizations with applications to model predictive control. In *Proceedings of the 52nd IEEE Conference on Decision and Control*, pages 3684–3690, Firenze, Italy, December 2013. Cited on pages 20, 37, and 38.
- J. Nocedal and S. Wright. *Numerical Optimization*. Springer-Verlag, 2006. Cited on pages 9, 10, 11, 13, 14, 16, 27, 34, 38, and 50.
- B. O'Donoghue, G. Stathopoulos, and S. Boyd. A splitting method for optimal control. In *IEEE Transactions on Control Systems Technology*, volume 21, pages 2432–2442. IEEE, 2013. Cited on page 65.
- C.V. Rao. *Moving Horizon Strategies for the Constrained Monitoring and Control of Nonlinear Discrete-Time Systems*. PhD thesis, University of Wisconsin-Madison, 2000. Cited on page 29.
- C.V. Rao, S.J. Wright, and J.B. Rawlings. Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications*, 99(3):723–757, December 1998. Cited on pages 20, 22, 27, 29, 37, and 102.
- P. Reuterswärd. *Towards Pseudospectral Control and Estimation*. ISSN 0280–5316. 2012. Licentiate Thesis. Cited on page 65.
- O. Shcherbina. Nonserial dynamic programming and tree decomposition in discrete optimization. In Karl-Heinz Waldmann and Ulrike M. Stocker, editors, *Operations Research Proceedings 2006*, volume 2006 of *Operations Research Proceedings*, pages 155–160. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-69994-1. doi: 10.1007/978-3-540-69995-8_26. Cited on page 66.
- D. Soudbakhsh and A.M. Annaswamy. Parallelized model predictive control. In *Proceedings of the American Control Conference*, pages 1715–1720, Washington, DC, USA, 2013. Cited on page 65.
- G. Stathopoulos, T. Keviczky, and Y. Wang. A hierarchical time-splitting approach for solving finite-time optimal control problems. In *Proceedings of*

- the European Control Conference*, pages 3089–3094, Zurich, Switzerland, July 2013. Cited on page 65.
- P. Tøndel, T.A. Johansen, and A. Bemporad. Further results on multiparametric quadratic programming. In *Proceedings of the 42nd IEEE European Control Conference*, pages 3173–3178, 2003. Cited on pages 71, 72, 73, 79, and 105.
- L. Vandenberghe, S. Boyd, and M. Nouralishahi. Robust linear programming and optimal control. Technical report, Department of Electrical Engineering, University of California Los Angeles, 2002. Cited on pages 20, 27, 29, and 37.
- S.J. Wright. Interior point methods for optimal control of discrete-time systems. *Journal of Optimization Theory and Applications*, 77:161–187, 1993. Cited on page 22.
- F. Zhang. *The Schur complement and its applications*, volume 4. Springer, 2005. Cited on page 39.
- Y. Zhu and C.D. Laird. A parallel algorithm for structured nonlinear programming. In *5th International Conference on Foundations of Computer-Aided Process Operations*, volume 5, pages 345–348, 2008. Cited on page 65.

Licentiate Theses
Division of Automatic Control
Linköping University

- P. Andersson:** Adaptive Forgetting through Multiple Models and Adaptive Control of Car Dynamics. Thesis No. 15, 1983.
- B. Wahlberg:** On Model Simplification in System Identification. Thesis No. 47, 1985.
- A. Isaksson:** Identification of Time Varying Systems and Applications of System Identification to Signal Processing. Thesis No. 75, 1986.
- G. Malmberg:** A Study of Adaptive Control Missiles. Thesis No. 76, 1986.
- S. Gunnarsson:** On the Mean Square Error of Transfer Function Estimates with Applications to Control. Thesis No. 90, 1986.
- M. Viberg:** On the Adaptive Array Problem. Thesis No. 117, 1987.
- K. Ståhl:** On the Frequency Domain Analysis of Nonlinear Systems. Thesis No. 137, 1988.
- A. Skeppstedt:** Construction of Composite Models from Large Data-Sets. Thesis No. 149, 1988.
- P. A. J. Nagy:** MaMiS: A Programming Environment for Numeric/Symbolic Data Processing. Thesis No. 153, 1988.
- K. Forsman:** Applications of Constructive Algebra to Control Problems. Thesis No. 231, 1990.
- I. Klein:** Planning for a Class of Sequential Control Problems. Thesis No. 234, 1990.
- F. Gustafsson:** Optimal Segmentation of Linear Regression Parameters. Thesis No. 246, 1990.
- H. Hjalmarsson:** On Estimation of Model Quality in System Identification. Thesis No. 251, 1990.
- S. Andersson:** Sensor Array Processing; Application to Mobile Communication Systems and Dimension Reduction. Thesis No. 255, 1990.
- K. Wang Chen:** Observability and Invertibility of Nonlinear Systems: A Differential Algebraic Approach. Thesis No. 282, 1991.
- J. Sjöberg:** Regularization Issues in Neural Network Models of Dynamical Systems. Thesis No. 366, 1993.
- P. Pucar:** Segmentation of Laser Range Radar Images Using Hidden Markov Field Models. Thesis No. 403, 1993.
- H. Fortell:** Volterra and Algebraic Approaches to the Zero Dynamics. Thesis No. 438, 1994.
- T. McKelvey:** On State-Space Models in System Identification. Thesis No. 447, 1994.
- T. Andersson:** Concepts and Algorithms for Non-Linear System Identifiability. Thesis No. 448, 1994.
- P. Lindskog:** Algorithms and Tools for System Identification Using Prior Knowledge. Thesis No. 456, 1994.
- J. Plantin:** Algebraic Methods for Verification and Control of Discrete Event Dynamic Systems. Thesis No. 501, 1995.
- J. Gunnarsson:** On Modeling of Discrete Event Dynamic Systems, Using Symbolic Algebraic Methods. Thesis No. 502, 1995.
- A. Ericsson:** Fast Power Control to Counteract Rayleigh Fading in Cellular Radio Systems. Thesis No. 527, 1995.
- M. Jirstrand:** Algebraic Methods for Modeling and Design in Control. Thesis No. 540, 1996.
- K. Edström:** Simulation of Mode Switching Systems Using Switched Bond Graphs. Thesis No. 586, 1996.

J. Palmqvist: On Integrity Monitoring of Integrated Navigation Systems. Thesis No. 600, 1997.

A. Stenman: Just-in-Time Models with Applications to Dynamical Systems. Thesis No. 601, 1997.

M. Andersson: Experimental Design and Updating of Finite Element Models. Thesis No. 611, 1997.

U. Forssell: Properties and Usage of Closed-Loop Identification Methods. Thesis No. 641, 1997.

M. Larsson: On Modeling and Diagnosis of Discrete Event Dynamic systems. Thesis No. 648, 1997.

N. Bergman: Bayesian Inference in Terrain Navigation. Thesis No. 649, 1997.

V. Einarsson: On Verification of Switched Systems Using Abstractions. Thesis No. 705, 1998.

J. Blom, F. Gunnarsson: Power Control in Cellular Radio Systems. Thesis No. 706, 1998.

P. Spångéus: Hybrid Control using LP and LMI methods – Some Applications. Thesis No. 724, 1998.

M. Norrlöf: On Analysis and Implementation of Iterative Learning Control. Thesis No. 727, 1998.

A. Hagenblad: Aspects of the Identification of Wiener Models. Thesis No. 793, 1999.

E. Tjärnström: Quality Estimation of Approximate Models. Thesis No. 810, 2000.

C. Carlsson: Vehicle Size and Orientation Estimation Using Geometric Fitting. Thesis No. 840, 2000.

J. Löfberg: Linear Model Predictive Control: Stability and Robustness. Thesis No. 866, 2001.

O. Härkegård: Flight Control Design Using Backstepping. Thesis No. 875, 2001.

J. Elbornsson: Equalization of Distortion in A/D Converters. Thesis No. 883, 2001.

J. Roll: Robust Verification and Identification of Piecewise Affine Systems. Thesis No. 899, 2001.

I. Lind: Regressor Selection in System Identification using ANOVA. Thesis No. 921, 2001.

R. Karlsson: Simulation Based Methods for Target Tracking. Thesis No. 930, 2002.

P.-J. Nordlund: Sequential Monte Carlo Filters and Integrated Navigation. Thesis No. 945, 2002.

M. Östring: Identification, Diagnosis, and Control of a Flexible Robot Arm. Thesis No. 948, 2002.

C. Olsson: Active Engine Vibration Isolation using Feedback Control. Thesis No. 968, 2002.

J. Jansson: Tracking and Decision Making for Automotive Collision Avoidance. Thesis No. 965, 2002.

N. Persson: Event Based Sampling with Application to Spectral Estimation. Thesis No. 981, 2002.

D. Lindgren: Subspace Selection Techniques for Classification Problems. Thesis No. 995, 2002.

E. Geijer Lundin: Uplink Load in CDMA Cellular Systems. Thesis No. 1045, 2003.

M. Enqvist: Some Results on Linear Models of Nonlinear Systems. Thesis No. 1046, 2003.

T. Schön: On Computational Methods for Nonlinear Estimation. Thesis No. 1047, 2003.

F. Gunnarsson: On Modeling and Control of Network Queue Dynamics. Thesis No. 1048, 2003.

S. Björklund: A Survey and Comparison of Time-Delay Estimation Methods in Linear Systems. Thesis No. 1061, 2003.

M. Gerdin: Parameter Estimation in Linear Descriptor Systems. Thesis No. 1085, 2004.

A. Eidehall: An Automotive Lane Guidance System. Thesis No. 1122, 2004.

E. Wernholt: On Multivariable and Nonlinear Identification of Industrial Robots. Thesis No. 1131, 2004.

J. Gillberg: Methods for Frequency Domain Estimation of Continuous-Time Models. Thesis No. 1133, 2004.

G. Hendeby: Fundamental Estimation and Detection Limits in Linear Non-Gaussian Systems. Thesis No. 1199, 2005.

D. Axehill: Applications of Integer Quadratic Programming in Control and Communication. Thesis No. 1218, 2005.

J. Sjöberg: Some Results On Optimal Control for Nonlinear Descriptor Systems. Thesis No. 1227, 2006.

D. Törnqvist: Statistical Fault Detection with Applications to IMU Disturbances. Thesis No. 1258, 2006.

H. Tidefelt: Structural algorithms and perturbations in differential-algebraic equations. Thesis No. 1318, 2007.

S. Moberg: On Modeling and Control of Flexible Manipulators. Thesis No. 1336, 2007.

J. Wallén: On Kinematic Modelling and Iterative Learning Control of Industrial Robots. Thesis No. 1343, 2008.

J. Harju Johansson: A Structure Utilizing Inexact Primal-Dual Interior-Point Method for Analysis of Linear Differential Inclusions. Thesis No. 1367, 2008.

J. D. Hol: Pose Estimation and Calibration Algorithms for Vision and Inertial Sensors. Thesis No. 1370, 2008.

H. Ohlsson: Regression on Manifolds with Implications for System Identification. Thesis No. 1382, 2008.

D. Ankelhed: On low order controller synthesis using rational constraints. Thesis No. 1398, 2009.

P. Skoglar: Planning Methods for Aerial Exploration and Ground Target Tracking. Thesis No. 1420, 2009.

C. Lundquist: Automotive Sensor Fusion for Situation Awareness. Thesis No. 1422, 2009.

C. Lyzell: Initialization Methods for System Identification. Thesis No. 1426, 2009.

R. Falkeborn: Structure exploitation in semidefinite programming for control. Thesis No. 1430, 2010.

D. Petersson: Nonlinear Optimization Approaches to $\mathcal{H}_2$ -Norm Based LPV Modelling and Control. Thesis No. 1453, 2010.

Z. Sjanic: Navigation and SAR Auto-focusing in a Sensor Fusion Framework. Thesis No. 1464, 2011.

K. Granström: Loop detection and extended target tracking using laser data. Thesis No. 1465, 2011.

J. Callmer: Topics in Localization and Mapping. Thesis No. 1489, 2011.

F. Lindsten: Rao-Blackwellised particle methods for inference and identification. Thesis No. 1480, 2011.

M. Skoglund: Visual Inertial Navigation and Calibration. Thesis No. 1500, 2011.

S. Khoshfetrat Pakazad: Topics in Robustness Analysis. Thesis No. 1512, 2011.

P. Axelsson: On Sensor Fusion Applied to Industrial Manipulators. Thesis No. 1511, 2011.

A. Carvalho Bittencourt: On Modeling and Diagnosis of Friction and Wear in Industrial Robots. Thesis No. 1516, 2012.

P. Rosander: Averaging level control in the presence of frequent inlet flow upsets. Thesis No. 1527, 2012.

N. Wahlström: Localization using Magnetometers and Light Sensors. Thesis No. 1581, 2013.

R. Larsson: System Identification of Flight Mechanical Characteristics. Thesis No. 1599, 2013.

Y. Jung: Estimation of Inverse Models Applied to Power Amplifier Predistortion. Thesis No. 1605, 2013.

M. Sylđatk: On Calibration of Ground Sensor Networks. Thesis No. 1611, 2013.

M. Roth: Kalman Filters for Nonlinear Systems and Heavy-Tailed Noise. Thesis No. 1613, 2013.

D. Simon: Model Predictive Control in Flight Control Design — Stability and Reference Tracking. Thesis No. 1642, 2014.

J. Dahlin: Sequential Monte Carlo for inference in nonlinear state space models. Thesis No. 1652, 2014.

M. Kok: Probabilistic modeling for positioning applications using inertial sensors. Thesis No. 1656, 2014.

J. Linder: Graybox Modelling of Ships Using Indirect Input Measurements. Thesis No. 1681, 2014.

G. Mathai: Direction of Arrival Estimation of Wideband Acoustic Wavefields in a Passive Sensing Environment. Thesis No. 1721, 2015.