# Efficient Numerical Methods for Moving Horizon Estimation

Niels HAVERBEKE

Jury:

Prof. dr. ir. Y. Willems, chair
Prof. dr. ir. B. De Moor, promotor
Prof. dr. M. Diehl, co-promotor
Prof. dr. ir. J. Suykens
Prof. dr. ir. J. Willems
Prof. dr. ir. W. Michiels
Prof. dr. M. Kinnaert (Université Libre de Bruxelles)
Prof. dr. L. Vandenberghe (University of California, Los Angeles)
Prof. dr. M. Verhaegen (Delft University of Technology)

Dissertation presented
in partial fulfillment of the
requirements for the degree
of Doctor in Engineering

September 2011

© 2011 Katholieke Universiteit Leuven
Groep Wetenschap & Technologie
Arenberg Doctoraatsschool
W. de Croylaan 6
B–3001 Heverlee (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of this publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

D/2011/7515/118

ISBN 978-94-6018-417-8

# Voorwoord

Nu ik de laatste hand leg aan wat me de voorbije jaren heeft bezig gehouden, wordt het eens dringend tijd om terug te blikken en diegenen te bedanken die mij geholpen hebben om dit voor mekaar te krijgen.

Zoals de traditie het wil, maar daarom zeker niet minder oprecht, wil ik eerst mijn promotor Prof. Bart De Moor bedanken. Tijdens mijn laatste jaren als burgerlijk ingenieur raakte ik geboeid door zijn lessen regeltechniek en wist ik welke richting ik uit wou gaan. En toen ik vlak voor de laatste examenperiode een interessant doctoraatsproject in Delft zag en zijn mening vroeg, wist hij me te overtuigen om in Leuven te blijven om de MPC-groep te versterken. Ik kreeg de kans om binnen zijn onderzoeksgroep in alle vrijheid mijn doctoraatsonderzoek uit te voeren. Bart, Met jouw niet aflatend enthousiasme weet je al jarenlang mensen te inspireren en te motiveren. Ik herinner mij de ”geeft er een lap op”na het behalen van mijn IWT-beurs, de ”go for it”toen ik de kans kreeg om naar Stanford te gaan, en de keren dat ik na een gesprek met jou weer uit jouw bureau stapte vol nieuwe invalshoeken en helemaal opgepept zoals een bokser die de ring betreedt voor een belangrijke kamp. Ook tijdens de laatste maanden, toen ik tegen beter weten ervoor koos om de combinatie te doen van fulltime werken en een doctoraat afronden, heb je me verschillende keren aangemoedigd. Bedankt daarvoor!

Ook mijn co-promotor, Prof. Moritz Diehl, wil ik graag extra bedanken. Toen ik ongeveer een half jaar aan het doctoreren was, kwam hij als een godsgeschenk op Esat aan om het nieuw opgerichte interdisciplinair center of excellence OPTEC te gaan leiden. Met zijn kennis en ervaring in het gespecialiseerde domein van MHE en optimale controle gaf hij mijn doctoraat een nieuwe wending en een extra boost. Het proefschrift dat hier voor u ligt, draagt daarom zeker het watermerk Prof. Diehl dankzij de continue wetenschappelijke feedback waarop ik steeds kon rekenen. Het doet mij deugd om te zien hoe OPTEC de voorbije jaren stelselmatig werd uitgebouwd tot een kenniscentrum dat nu gekend is tot in alle uithoeken van de wereld. Moritz, Ik bewonder jouw gave om mensen met elkaar in contact te brengen en samenwerkingen te stimuleren. Het was een waar genoegen om met jou te hebben mogen samenwerken.

Daarnaast wil ik ook mijn twee assessoren, Prof. Lieven Vandenberghe en Prof. Jan Swevers, danken voor de waardevolle inbreng van hun expertise. Prof. Vandenberghe

wil ik extra bedanken voor zijn bereidwilligheid om in mijn jury te zetelen en omdat hij een trip naar Boedapest wilde combineren met een tussenstop in België om mijn preliminaire verdediging bij te wonen.

Ook bedank ik de andere leden van mijn examencommissie, voorzitter Prof. Yves Willems, Prof. Wim Michiels, Prof. Jan Willems, Prof. Johan Suykens en de externe juryleden, Prof. Michel Verhaegen (Delft University of Technology) en Prof. Michel Kinnaert (Université Libre de Bruxelles), voor hun bereidwilligheid om in mijn jury te zetelen en voor de nuttige feedback die mijn doctoraatstekst versterkt heeft.

Een speciale dank gaat uit naar Prof. Jan Willems die mij al tijdens mijn eerste weken wist te inspireren en door zijn kritisch wetenschappelijke ingesteldheid een belangrijke invloed gehad heeft op mijn werk en dit op verschillende momenten tijdens mijn doctoraat.

Een speciale dank ook aan Prof. Johan Suykens voor de dynamiek die hij aan SCD geeft en voor de vele informele gesprekken die meer dan eens een nieuwe invalshoek gaven aan de ideeën die ik aan het uitwerken was.

Voorts wil ik Prof. Joos Vandewalle bedanken voor de aangename samenwerking voor het vak systeemidentificatie. Terwijl de dictatische taken meestal een noodzakelijk kwaad zijn tijdens een doctoraatsopleiding, was het voor mij een meerwaarde dankzij de vrijheden en verantwoordelijkheden die ik kreeg om het vak te geven. Tegelijkertijd mocht ik van hem elk jaar opnieuw waardering ondervinden voor het geleverde werk.

*I would like to address a special word of gratitude to Prof. Stephen Boyd for inviting me to Stanford University. It was a very rewarding experience both professionally and personally to work in such an inspiring environment, and in a group of outstanding researchers who are endulged in your spirit of innovation. Even though you are enormously occupied, you regularly reserved time for me for both informal chats and scientific advice. You managed to overwhelm me with a tsunami of new ideas, which all proved to work when I worked them out afterwards!*

Het IWT ben ik dankbaar voor het gestelde vertrouwen en de financiele ondersteuning van mijn onderzoek, en het FWO Vlaanderen voor de financiele ondersteuning van mijn onderzoeksverblijf in Stanford.

Naast wetenschappelijke ontwikkeling gaat een doctoraat ook over persoonlijke ontwikkeling. Het is een continue zoektocht, een reis met ups en downs. Op mijn reis heb ik echter mogen rekenen op de aanwezigheid van mijn collega's, vrienden en familie tijdens de meest droevige maar ook de meest euforische momenten.
*An office is one's home when at work. To the (ex-)colleagues at Esat and OPTEC, thank you for providing such an enjoyable workplace and for the many inspiring discussions.* Tom, onze samenwerking in het glycemie-onderzoek heeft mijn doctoraat verrijkt. Niet alleen door de uitdaging van het onderwerp op zich, maar ook door de vele boeiende discussies en de resultaten die we bereikt hebben. Ik wens je alle succes

met de klinische studies en duim samen met jou dat die droom van MPC-geregelde glycemie normalisatie binnenkort een realiteit wordt in elk ziekenhuis. Daarnaast wil ik jou heel erg bedanken voor alle steun en vriendschap. Raf, partner-in-crime bij de vele fratsen die we uithaalden, jij was altijd degene die leven in de brouwerij bracht! Maar je was ook steeds bereid om naar mij te luisteren, voor zever maar ook voor serieuze praat. Lieven, onze eilandprofessor, jouw goede raad was meer dan eens nodig om het jonge geweld in goede banen te leiden.
Iemand die ik zeker ook extra wil bedanken, is Bert Pluymers, die door zijn begeleiding tijdens onze masterthesis maar zeker ook tijdens mijn eerste doctoraatsmaanden, een belangrijke rol gespeeld heeft in de beginfase van mijn onderzoek.
Toni, op en naast Esat, kon ik steeds op jou rekenen. Merci voor de koffiepauzes en discussies, maar ook voor de pokeravonden, voetbalmatchkes en vooral ook voor de onvergetelijke trips naar Griekenland, Korea en China.
Peter, bedankt om de Overpelt-Leuven *connection* zoveel korter te maken door de aangename babbelrijke autoritjes.
Verder bedank ik mijn andere collega's Philippe, Kris, Jos, Pieter, Kim, Tillmann, Mauricio, Fabian, Carlos, Ernesto, Marco, Maarten, Nico, Pascalis, Joachim, Boris, Quoc, Carlo, Joel, Ioanna, Max, Julian. En natuurlijk de trippers van het eerste uur: Steven G., Bart V., Jeroen, Erik, Marcello, Samuel, Kristiaan, Xander, Steven V., Bert C., Nathalie, Ruth, Wout, Thomas, Karen, Tim, Olivier en alle andere SCD-collega's.

Bedankt aan Ilse, John, Ida, Mimi, Jacqueline en Eliane voor de administratieve ondersteuning. Speciale dank aan Ilse voor haar sociale toets, haar inspanningen om de sfeer in de groep te bevorderen en voor de steun die ik zelf heb gekregen.

Kristof, Maarten en Michael, mijn niet-SCD maten die toch in hetzelfde schuitje zaten, merci voor de koffiemomenten en voor de legendarische poker- en voetbalavonden.

*I also want to thank my friends abroad for making my stay in Stanford unforgettable and for the support in hard times. Special thanks to some very good friends whom I could always count on: Uygar, Kwangmooh, Byron, Kaan and Yang. Also thanks to Argyris, Jacob, Sid, Almir and Joelle. Thanks for the ski-experience and the Bites and Nexus moments. Finally, I wish to address my house lord, Charro. Thank you for treating me as family and for making my stay pleasant. You were always in for a chat and a smile even though you had sorrows of your own.*

Mijn collega's bij Flanders' DRIVE wil ik bedanken voor hun interesse in mijn doctoraat en om regelmatig te vragen naar de status van afronding. Door er regelmatig aan herinnerd te worden, kon ik mezelf telkens motiveren om ook na een drukke werkdag beetje bij beetje de doctoraatstekst af te ronden. Directeur Renilde Craps was zeker een van hen die deze vooruitgang steeds levendig wist te houden door regelmatig te polsen naar de datum van verdediging en door haar interesse.

Tot slot komt de belangrijkste dankbetuiging. Zonder mijn familie en schoonfamilie zou ik nooit staan waar ik nu sta. Het is moeilijk uit te leggen aan een buitenstaander, maar een doctoraat is niet te vergelijken met een gewone job. Je neemt je onderzoek namelijk overal mee. Dat vraagt heel wat begrip van de mensen om je heen. Daarom dank ik mijn familie en schoonfamilie voor hun interesse en aanmoedigingen, ook al was het voor hen waarschijnlijk vaak onduidelijk wat ik daar allemaal aan het *uitvinden* was in Leuven. Mijn lieve ouders ben ik eeuwig dankbaar voor de vele kansen die ze mij gegeven hebben. Ma, Pa, bedankt om steeds voor me klaar te staan met raad en daad. De wekelijkse verhuis van en naar Leuven met kleren, *bakskes* met vitaminen, maar ook zetels, tv, pc en meer van dat, was voor jullie nooit een probleem! Lieve Ma, ik weet dat je vandaag ongelooflijk fier op me zou geweest zijn. Het doet me veel pijn dat je dit niet meer zelf kan meemaken. Maar dit doctoraat draag ik op aan jou! Geert, *big brother*, samen hebben we al heel wat watertjes doorzwommen. Bedankt voor jouw steun tijdens ups en downs. Gerdje, bedankt voor de altijd gezellige en rijkelijk gevulde tafelmomenten. Rita en Martin, jullie stonden altijd klaar voor mij, met liefde, zorg en goede raad. Rita, bedankt voor de nodige *sjot onder mijn gat* op regelmatige tijden wanneer ik er weer eens tegenaan moest gaan. En bedankt om je steeds te ontfermen over mijn *DHL pakske* als ik het zelf weer eens veel te druk had.

Er wordt weleens gezegd dat doctoraten 's nachts geschreven worden. Anneleen weet dat dit ook 's avonds en in het weekend gebeurt. Daarom, aan mijn allerliefste vrouwtje, een dikke merci voor jouw constante steun en aanmoedigingen. Bedankt om je geduld nooit te verliezen als er weer eens *een algoritme voorbij kwam* terwijl je tegen mij aan het vertellen was over je werkdag. Ik besef maar al te goed dat ik dit niet voor mekaar had gekregen zonder jouw enthousiasme, geduld en vertrouwen. Ik kijk er naar uit om samen aan een nieuw hoofdstuk te beginnen … En die verloren avonden en weekends die zullen we nog wel dubbel en dik inhalen. Beloofd!

Ongetwijfeld ben ik, mezelf kennende, nog heel wat mensen vergeten te vermelden, waarvoor alvast mijn excuses. Ik zou immers iedereen die geholpen heeft bij het tot stand komen van dit doctoraat, mij op welke manier dan ook heeft bijgestaan of mij het leven aangenamer heeft gemaakt, oprecht willen bedanken. En, aan mijn ex-collega's die nu nog volop werken aan hun doctoraat, zou ik willen zeggen: *Never give up! The best is yet to come.*

Niels Haverbeke 16 september 2011

# Abstract

In model-based predictive control strategies, accurate estimates of the current state and model parameters are required in order to predict the future system behavior for a given control realization. One particularly powerful approach for constrained nonlinear state estimation is Moving Horizon Estimation (MHE). In MHE past measurements are reconciled with the model response by optimizing states and parameters over a finite past horizon. The basic strategy is to use a moving window of data such that the size of the estimation problem is bounded by looking at only a subset of the available data and summarizing older data in one initial condition term. This also establishes an exponential forgetting of past data which is useful for time-varying dynamics.

Compared to other state estimation approaches, MHE offers many advantages following from its formulation as a dynamic optimization problem. Inequality constraints on the variables (states, parameters, disturbances) can be included in a natural way and the nonlinear model equation is directly imposed over the horizon length. Empirical studies show that MHE can outperform other estimation approaches in terms of accuracy and robustness. In addition to these well-known advantages, the framework of MHE allows for formulations different from the traditional (weighted) least-squares formulation.

The greatest impediment to a widespread acceptance of MHE for real-time applications is still its associated computational complexity. Despite tremendous advances in numerical computing and Moore's law, optimization-based estimation algorithms are still primarily applied to slow processes. In this work, we present fast structure-exploiting algorithms which use robust and efficient numerical methods and we demonstrate the increased performance and flexibility of nonlinear constrained MHE.

MHE problems are typically solved by general purpose (sparse) optimization algorithms. Thereby, the symmetry and structure inherent in the problems are not fully exploited. In addition, the arrival cost is typically updated by running a (Extended) Kalman filter recursion in parallel while the final estimate covariance is computed from the derivative information. In this thesis, Riccati based methods are derived which effectively exploit the inherent symmetry and structure and yield the arrival cost update and final estimate covariance as a natural outcome of the solution process. The primary emphasis is on the robustness of the methods which is achieved by

orthogonal transformations.
When constraints are imposed, the resulting quadratic programming (QP) problems can be solved by active-set or interior-point methods. We derive modified Riccati recursions for interior-point MHE and show that square-root recursions are recommended in this context because of the numerical conditioning. We develop an active-set method which uses the unconstrained solution obtained from Riccati recursions and employs a Schur complement technique to project onto the reduced space of active constraints. The method involves non-negativity constrained QPs for which a gradient projection method is proposed. We implement the algorithms in efficient C code and demonstrate that MHE is applicable to fast systems.
These QP methods are at the core of solution methods for general convex and nonlinear MHE as is demonstrated. Convex formulations are investigated for robustness to outliers and abrupt parameter changes. Furthermore, the methods are embedded in a Sequential Quadratic Programming strategy for nonlinear MHE. One application has been of particular interest during this doctoral research: estimation and predictive control of blood-glucose at the Intensive Care Unit (ICU). For this application reliability and robustness of the estimates as well as of the numerical implementations are crucial. We evaluate an MHE based MPC control strategy and show its potential for this application.

# Korte inhoud

Modelgebaseerde predictieve regelstrategieën vereisen accurate schattingen van de modelparameters en de huidige toestand om het toekomstige systeemgedrag te kunnen voorspellen. Toestandsschatting met schuivend tijdsvenster of moving horizon estimation (MHE) is een krachtige techniek voor niet-lineaire schatting met ongelijkheidsbeperkingen. Het idee in MHE is om de metingen met het model te combineren door middel van optimalisatie over een eindig venster. Door slechts een eindig aantal meetpunten te beschouwen en een schuivend tijdsvenster te gebruiken blijft het optimalisatieprobleem berekenbaar. Zodoende wordt er ook automatisch een laag gewicht toegekend aan oude metingen die weinig informatie geven over de huidige toestand en parameters.

In vergelijking met andere methoden voor toestandsschatting, heeft MHE een aantal voordelen die volgen uit de formulering als dynamisch optimalisatieprobleem. Ongelijkheidsbeperkingen kunnen eenvoudig opgelegd worden en de niet-lineaire modelvergelijkingen worden rechtstreeks in rekening gebracht over het volledige tijdsvenster. Empirische studies tonen aan dat MHE accuratere en robuustere schattingen oplevert dan andere toestandsschattingsmethoden. Naast deze gekende voordelen, laat MHE andere formuleringen toe dan de traditionele kleinste kwadraten formulering.

Een grootschalige doorbraak van MHE voor real-time toepassingen wordt enkel verhinderd door de geassocieerde rekencomplexiteit. In elke tijdstap dient namelijk een optimalisatieprobleem opgelost te worden. Ondanks de enorme vooruitgang op het vlak van numerieke methoden en rekenkracht (de wet van Moore), worden optimalisatiegebaseerde methoden voor toestandsschatting nog voornamelijk toegepast op relatief trage processen. Deze thesis tracht aan te tonen dat MHE toepasbaar is op snelle processen door gebruik te maken van structuurbenuttende robuuste algoritmen en bovendien aanleiding geeft tot een hogere performantie en flexibiliteit.

MHE optimalisatieproblemen worden typisch met standaard optimalisatiealgoritmen opgelost. Daardoor kan de structuur en symmetrie die inherent is aan deze problemen niet ten volle benut worden. Bovendien wordt de zogenaamde *arrival cost* typisch berekend door een (Extended) Kalman filter recursie in parallel uit te voeren en wordt de covariantie van de laatste schatting berekend door een statistische analyse.

In deze thesis worden Riccati gebaseerde methoden ontwikkeld, welke de structuur en symmetrie uitnutten en tezelfdertijd de *arrival cost update* en covariantie van de

laatste schatting opleveren als bijproduct van het oplossingsproces. De ontwikkelde methoden maken gebruik van orthogonale transformaties voor maximale robuustheid. Wanneer ongelijkheidsbeperkingen aanwezig zijn, kunnen *active-set* ofwel *interior-point* methoden aangewend worden. In deze thesis worden aangepaste Riccati recursies afgeleid voor *interior-point* MHE methoden en wordt aangetoond dat *square-root* methoden een duidelijk voordeel bieden wat betreft numerieke conditionering. Daarnaast wordt een *active-set* methode voorgesteld welke de MHE oplossing zonder beperkingen als startpunt neemt en een beperkt aantal active-set iteraties nodig heeft voor convergentie. In elke iteratie wordt geprojecteerd op de gereduceerde ruimte van actieve beperkingen door middel van een Schur complement techniek. Voor de resulterende gereduceerde QPs met positiviteitsbeperkingen wordt een gradiënt projectiemethode voorgesteld. Efficiente implementaties van de algoritmen in C tonen de toepasbaarheid van MHE op snelle systemen aan.

De ontwikkelde methoden voor kwadratische programmering liggen aan de basis van algemene convexe of niet-lineaire MHE methoden. In deze thesis worden convexe formuleringen onderzocht voor robuustheid tegen abnormale meetwaarden en abrupte parametervariaties. Tot slot worden de methoden gebruikt in an SQP strategie voor niet-lineaire MHE.

Een bepaalde toepassing is van bijzonder belang geweest tijdens dit doctoraat: real-time schatting en predictieve regeling van de bloedglucose bij patiënten op intensieve zorgen. Voor deze toepassing zijn robuustheid en betrouwbaarheid van zowel de schattingen als de numerieke implementaties cruciaal. Een MHE gebaseerde MPC strategie wordt geëvalueerd en voorgesteld voor toepassing in de nabije toekomst.

# Contents

**Contents** ix

**List of Notations and Acronyms** xv

**1 Introduction** 1
- 1.1 History . . . 1
- 1.2 Dynamical system models . . . 4
- 1.3 State estimation . . . 6
  - 1.3.1 Filtering, prediction and smoothing . . . 7
- 1.4 Moving horizon estimation . . . 8
  - 1.4.1 Least-squares batch state estimation . . . 8
  - 1.4.2 General batch state estimation . . . 9
  - 1.4.3 Constrained state estimation . . . 10
  - 1.4.4 Moving horizon approximation . . . 11
  - 1.4.5 Simultaneous state and parameter estimation . . . 13
- 1.5 Kalman filter . . . 14
  - 1.5.1 Nonlinear extensions of the Kalman filter . . . 14
- 1.6 A note on the deterministic versus stochastic interpretation of state estimation . . . 15
- 1.7 Model predictive control . . . 16
- 1.8 Similarities and differences between control and estimation . . . 18
- 1.9 Motivation . . . 20
- 1.10 Structure of the thesis . . . 22
- 1.11 Specific contributions . . . 22

**2 Computational framework** 27
- 2.1 Stability framework for MHE . . . 27
  - 2.1.1 Linear time-invariant systems . . . 30
- 2.2 Linear algebra . . . 31
- 2.3 Convex optimization . . . 36
  - 2.3.1 Definitions . . . 37
  - 2.3.2 Duality and Karush-Kuhn-Tucker conditions . . . 37

2.3.3 Linear least-squares . . . 40
2.3.4 Linear programming . . . 42
2.3.5 Quadratic programming . . . 42
2.3.6 Second-order cone programming . . . 42
2.4 Nonlinear optimization . . . 43
2.4.1 Newton's method for unconstrained optimization . . . 43
2.4.2 Sequential Quadratic Programming (SQP) . . . 45
2.4.3 Interior-point methods . . . 47
2.5 Algorithm complexity and memory storage . . . 47
2.5.1 Flop count . . . 47
2.5.2 Memory allocation . . . 48
2.6 Brief introduction to the Kalman filter . . . 48
2.6.1 The covariance Kalman filter (CKF) . . . 49
2.6.2 The Kalman filter as a feedback system . . . 50
2.6.3 The time-invariant and steady-state Kalman filter . . . 50
2.6.4 The square-root covariance Kalman filter (SRCF) . . . 51
2.6.5 The information filter (IF) . . . 54
2.7 Brief introduction to Kalman smoothing . . . 58
2.7.1 Fixed-point smoothing . . . 58
2.7.2 Improvement due to smoothing . . . 59
2.7.3 Fixed-lag smoothing . . . 60
2.7.4 Fixed-interval smoothing . . . 61

**3 Efficient methods for unconstrained MHE 65**
3.1 Introduction . . . 65
3.2 Structure-exploiting algorithms for unconstrained MHE . . . 68
3.2.1 Normal Riccati based solution method . . . 69
3.2.2 Square-root Riccati based solution method . . . 75
3.2.3 Structured *QR* factorizations . . . 82
3.3 Proof of equivalence of unconstrained MHE and the Kalman filter/smoother 82
3.4 Analogy with Riccati methods for MPC . . . 87
3.5 Numerical examples . . . 88
3.6 Conclusions . . . 89

**4 Interior-point methods for MHE 95**
4.1 Introduction . . . 95
4.2 Overview of interior-point methods for quadratic programming . . . 96
4.2.1 Primal barrier method . . . 96
4.2.2 Primal-dual interior-point methods . . . 98
4.3 Structure-exploiting interior-point methods for MHE . . . 98
4.3.1 Computing the Newton step . . . 99
4.4 Numerical examples . . . 109

4.4.1 Constrained linear system . . . 109
4.4.2 Waste water treatment process . . . 111
4.4.3 A hot-starting strategy . . . 113
4.4.4 Numerical conditioning . . . 120
4.5 Conclusions . . . 121

**5 Active-set methods for MHE 123**
5.1 Introduction . . . 123
5.2 Overview of active-set methods for quadratic programming . . . 125
5.2.1 Solving equality constrained QPs . . . 126
5.2.2 Primal active-set methods . . . 127
5.2.3 Dual active-set methods . . . 128
5.2.4 qpOASES - an online active-set strategy . . . 129
5.3 A Schur-complement active-set method for MHE . . . 129
5.3.1 Outline of the active-set method . . . 130
5.3.2 MHE solution using Riccati recursions . . . 137
5.3.3 Forward and backward vector solves . . . 137
5.3.4 Gradient projection method . . . 138
5.3.5 Updating/downdating Cholesky factorizations . . . 142
5.3.6 Computational burden . . . 147
5.4 Numerical examples . . . 147
5.4.1 Waste water treatment process . . . 147
5.5 Conclusions . . . 148

**6 Convex MHE formulations 151**
6.1 Robust estimation using Huber penalty function . . . 151
6.1.1 Robust moving horizon estimation . . . 153
6.1.2 The Huber penalty function . . . 154
6.1.3 The multivariate Huber penalty function . . . 156
6.1.4 Selecting the tuning parameter . . . 157
6.1.5 Huber penalty MHE . . . 158
6.1.6 Smooth hybrid $\ell_1$ - $\ell_2$ MHE . . . 159
6.1.7 $L1$ norm MHE . . . 159
6.1.8 Numerical example . . . 160
6.2 Joint input/parameter and state estimation . . . 165
6.2.1 Cardinality problems . . . 165
6.2.2 Joint estimation with piecewise changing parameters . . . 167
6.2.3 Riccati based solution . . . 168
6.2.4 Polishing . . . 169
6.2.5 Numerical example . . . 169
6.3 Conclusions . . . 170

**7 Nonlinear MHE algorithms and application to estimation and control of blood-glucose in the Intensive Care** **171**
7.1 Introduction . . . 171
7.2 Brief overview of recursive nonlinear estimation methods . . . 172
7.2.1 The Extended Kalman Filter (EKF) . . . 172
7.2.2 The Unscented Kalman Filter (UKF) . . . 173
7.3 Introduction to nonlinear MHE using multiple shoothing and SQP . . 174
7.3.1 Discretization . . . 175
7.3.2 Constrained Gauss-Newton Sequential Quadratic Programming (CGN-SQP) . . . 176
7.3.3 Arrival cost updates . . . 181
7.4 Application of MHE based NMPC to normalize glycemia of critically ill patients . . . 182
7.4.1 Tight glycemic control in the Intensive Care Unit . . . 183
7.4.2 ICU Dataset . . . 183
7.4.3 ICU Minimal Model (ICU-MM) . . . 184
7.4.4 Smoothing of discontinuities . . . 186
7.4.5 Closed-loop nonlinear control system set-up . . . 186
7.4.6 Target calculation . . . 187
7.4.7 Model predictive control . . . 188
7.4.8 ICU-MM parameters . . . 189
7.4.9 Results and discussion . . . 190
7.5 Conclusions . . . 192

**8 General conclusions and outlook** **195**
8.1 Conclusions . . . 195
8.2 Future work . . . 197
8.2.1 Sum-of-norms regularization . . . 197
8.2.2 Emerging applications: fast and large-scale systems . . . 197
8.2.3 Decentralized and distributed MHE . . . 198
8.2.4 Interaction between MHE and MPC . . . 199

**A Simplification of Riccati methods for case $H \equiv 0$** **201**
A.1 Normal Riccati based method . . . 202
A.2 Square-root Riccati based method . . . 204

**B QR factorization methods** **207**
B.1 Dense QR methods . . . 207
B.1.1 Householder QR methods . . . 207
B.1.2 Givens QR methods . . . 209
B.1.3 Numerical results . . . 211

B.2 Structured QR methods . . . . . 212
B.2.1 Structured Householder QR method . . . . . 213
B.2.2 Structured Givens QR method . . . . . 214
B.2.3 Numerical results . . . . . 214

**C Robust measures** **219**

**Bibliography** **223**

**List of Publications** **239**

**Curriculum Vitae** **242**

# List of Notations and Acronyms

## Notations

### Variables

| | |
|---|---|
| $\alpha, \beta, \gamma \in \mathbb{R}$ | Greek symbols denote scalar variables |
| $a, b, c \in \mathbb{R}^n$ | Lower case roman symbols denote scalar or vector variables |
| $A, B, C \in \mathbb{R}^{m \times n}$ | Upper case roman symbols denote matrix variables |
| $A(i, j)$ | Element at the $i^{\text{th}}$ row and $j^{\text{th}}$ column of matrix $A$ |
| $A(i, :)$ | $i^{\text{th}}$ row of matrix $A$ |
| $A(:, j)$ | $j^{\text{th}}$ column of matrix $A$ |
| $A(i : j, k : l)$ | Submatrix spanning rows $i$ through $j$ and columns $k$ through $l$ of matrix $A$ |
| $[a; b; c]$ | Stacked vectors: $[a^{\mathrm{T}}\ b^{\mathrm{T}}\ c^{\mathrm{T}}]^{\mathrm{T}}$ |
| $y_{[0:N]}$ | Stacked vectors: $[y_0^{\mathrm{T}} \dots y_N^{\mathrm{T}}]^{\mathrm{T}}$ |

### Sets

| | |
|---|---|
| $\mathbb{R}$, $\mathbb{R}_0$ | Set of real numbers and nonzero real numbers |
| $\mathbb{Z}$, $\mathbb{Z}^+$, $\mathbb{Z}_0^+$ | Set of integers, positive integers and strictly positive integers respectively |
| $\mathbb{S}^n$ | Set of symmetric $n \times n$ matrices |
| $\mathbb{S}_+^n$ | Set of symmetric positive semidefinite $n \times n$ matrices |
| $\mathbb{S}_{++}^n$ | Set of symmetric positive definite $n \times n$ matrices |

### Matrix operations

| | |
|---|---|
| $A^{\mathrm{T}}$ | Transpose of matrix $A$ |
| $\mathrm{Tr}(A)$ | Trace of a matrix i.e. sum of its diagonal elements |
| $\mathrm{rows}(A)$ | number of rows in matrix $A$ |
| $\mathrm{cols}(A)$ | number of columns in matrix $A$ |

## Norms

| | |
|---|---|
| $\|x\|_1$ | 1-norm of a vector |
| $\|x\|_2$ | 2-norm of a vector: $\sqrt{x^{\mathrm{T}}x}$ |
| $\|x\|_Q$ | Weighted 2-norm of a vector: $\sqrt{x^{\mathrm{T}}Qx}$ with $Q \in \mathbb{S}^n_{++}$ |
| $\|x\|_p$ | p-norm of a vector: $(\sum_{i=1}^n \|x_i\|^p)^{1/p}$ |

## Optimization

| | |
|---|---|
| $\min_x$ | Function minimization over $x$, optimal function value is returned |
| $\arg\min_x$ | Function minimization over $x$, optimal value of $x$ is returned |
| s.t. | Subject to constraints |

## Acronyms

| | |
|---|---|
| ASM | Active-Set Method |
| CGN | Constrained Gauss-Newton |
| EKF | Extended Kalman Filter |
| ICU | Intensive Care Unit |
| ICU-MM | Intensive Care Unit Minimal Model |
| IP | Interior-Point |
| IPM | Interior-Point Method |
| KKT | Karush-Kuhn-Tucker |
| LP | Linear Program |
| LQR | Linear Quadratic Regulator |
| LTI | Linear Time-Invariant |
| LTV | Linear Time-Varying |
| MHE | Moving Horizon Estimation |
| MPC | Model Predictive Control |
| MSE | Mean Squared Error |
| NLP | Nonlinear Program |
| NLS | Nonlinear Least-Squares |
| QP | Quadratic Program |
| SCP | Sequential Convex Programming |
| SOCP | Second-order Cone Program |
| SQP | Sequential Quadratic Programming |
| UKF | Unscented Kalman Filter |

CHAPTER 1

# Introduction

Physical systems are designed to perform specific functions. Knowing the system state is necessary to solve many control problems. In most practical cases, however, the physical state of the system cannot be determined by direct observation. Instead, the state needs to be estimated from noisy output measurements and a process model usually obtained from physical insight.

## 1.1. History

State estimation has had a long and remarkable history in the natural sciences and engineering and was influenced by some of the most prodigious scientists as Gauss, Legendre and Maxwell. The first method for forming an optimal estimate from noisy measurements was the *method of least-squares*. It was developed during the Age of Exploration, late eighteenth century, when scientists sought for solutions to the challenges of navigating the oceans. The accurate description of the behavior of celestial bodies was key to determining the postion of the ships, enabling them to sail in open seas. Carl Friedrich Gauss is commonly credited with developing the fundamentals of the method of least-squares in 1795, although Legendre independently developed the method around the same time and was the first to publish it. Interestingly, Gauss used his method to solve a specific problem, namely determining the orbit of the newly discovered planet Ceres, which is still part of our Solar System but is now categorized as a dwarf planet. The Italian astronomer Piazzi discovered *the last missing planet* Ceres in 1801. He was only able, however, to observe the planet for forty-one days after which it moved behind the sun. This launched a scientific challenge of determining the orbit of Ceres using Piazzis observations such that astronomers would be able to locate the planet when it reappeared. The problem of determining Ceres' orbit was

more complex than any other previous problem in astronomy. For the discovery of Uranus astronomers had relied on the assumption of a circular orbit, which was nearly correct, but the orbit of Ceres was elleptic with unknown eccentricity. Apparently, the orbit could not be determined from the data using known methods. Under certain hypotheses which Gauss adopted from Kepler the orbit was determined by a set of five parameters. No dynamics of the object itself were needed as long as the object remained in this orbit, *i.e.* unperturbed by large planets. Gauss solved the nonlinear least-squares problem by hand calculation and using only three out of the twenty-two observations made by Piazzi over forty-one days. It required over one hundred hours of calculation. His approach was to determine a rough approximation followed by iterative refinements, allowing the estimated orbit to fit the observations smoothly. Using Gauss' predictions astronomers found Ceres again after it reemerged from behind the sun.

In one of his remarks Gauss anticipated the maximum likelihood method, which was only introduced in 1912 by Robert Fisher. It is interesting that Gauss rejected the probabilistic maximum likelihood method in favor of the deterministic method of minimizing the sum of squared residuals [166]. Inspired by the maximum likelihood method Kolmogorov and Wiener independently developed around 1941 a least-mean-square estimation method using autocorrelation functions. This Wiener-Kolmogorov filter has two important drawbacks. It is only applicable to stationary processes and it requires the solution of an integral equation known as the Wiener-Hopf equation.

The breakthrough in estimation theory was achieved by Rudolph Kalman with the development of his famous filter. The Kalman filtering algorithm was originally published by Kalman [112] in 1960 for discrete-time systems and by Kalman and Bucy [113] in 1961 for continuous-time systems. It has been the subject of many research works following its initial publication and has been covered in numerous textbooks. The Kalman filter provides a solution which is far superior to the Wiener-Kolmogorov filter [196] due to its recursive nature and effective use of the Riccati equation. The key idea which led Kalman to derive his filter was equating expectation with projection [83]. The Kalman filter is applicable to non-stationary systems. In fact the first application of the Kalman filter was a nonlinear one – trajectory estimation for the Apollo project, a planned mission to the moon and back which resulted in one of the greatest achievements of mankind with the moon landing of Apollo 11 in 1969 (Figure 1.1).

To date the Kalman filter has found widespread application in diverse areas including space- and aircraft navigation, GPS, automotive, mechatronics, oil refining and chemical process industry, (nuclear) power industry, communication networks, economics, computer vision applications, oceanography, weather and air quality forecasting, human gait analysis, fluid dynamics. The impact of the Kalman filter cannot be overestimated. Its popularity can be attributed to the fact that it is both theoretically attractive

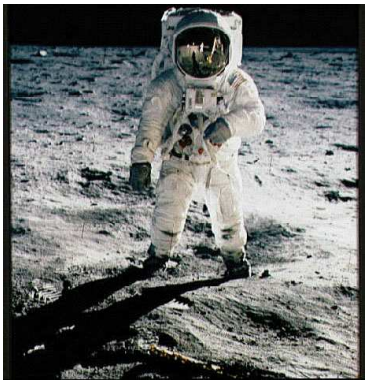


**Figure 1.1.** The Kalman filter was used for trajectory estimation in the Apollo space program. These missions led to the first manned moon landing in 1969, one of the greatest achievements of the 20th century.

– because of all possible linear unbiased state estimators it is the one that is optimal in the sense of minimum variance estimation error and in addition it is asymptotically stable – and at the same time yields a very simple yet powerful practical implementation. Historical surveys on the development of Kalman filtering can be found in [166] and [111].

In 1963 Bryson and Frazier [30] first showed the connection between Kalman filtering and optimization. Early formulations of linear unconstrained MHE, sometimes referred to as limited memory filters, were presented in [106], [162] and [175]. Given the computational limits of the 1960's it is not surprising that recursive solution methods were proposed for these formulations. For general nonlinear dynamical models exact recursive solutions are impossible to compute in finite time as the problem becomes infinite dimensional as shown by Kushner in 1964 [120]. Therefore approximations must be made, which led to several nonlinear filters. The Kalman filter was extended to nonlinear models by linearizing through a first order Taylor series around the current estimate [34]. Nonlinear unconstrained MHE was first proposed by Jang and coworkers [105] in 1986. The formulation, however, ignored disturbances. Their work was extended in the following years by Tjoa and Biegler [177], Liebman and coworkers [125] and Muske and coworkers [134]. Further investigations in the following years have led to a deeper understanding in the optimality and stability properties resulting in effective and stable MHE formulations [59, 146, 155]. Stability of linear constrained MHE was addressed by Findeisen [59], Rao et al [148], Alessandri et al [4] and for nonlinear systems by Alamir et al [1], Rao et al [149], Alessandri et al [5] and Zavala et al [201, 202]. MHE for hybrid systems was investigated in [63].

Since MHE is an optimization-based state estimation method, it strongly depends on

the underlying numerical optimization schemes. An overview of numerical aspects and techniques for MPC and MHE is given by Diehl et al [44]. Albuquerque and Biegler [3] first proposed a structure-exploiting algorithm for MHE which scales linearly in the horizon length. Riccati based methods for MHE problems have been proposed, *e.g.* by Tenny et al [173], Jorgensen et al [108], Haverbeke et al [94] and Zavala et al [199, 202].

On the other side, researchers have proposed explicit MHE methods, see *e.g.* [36, 169], which aim to move the computations offline. These methods typically involve the solution of a multiparametric (quadratic) program over the variable space and a tabulation over all possible regions which is consulted online. The drawback of this approach is an exponential growth of the number of regions hence of the look-up-tables when either the number of variables or the estimation horizon are increased.

Algorithms for efficient nonlinear MHE have also been investigated. Zavala et al [202] proposed an algorithm based on NLP sensitivity and collocation. This was extended to a fast but approximate algorithm for MHE [200]. An MHE scheme, inspired by the multiple shooting real-time iteration scheme for NMPC proposed by Diehl [42, 43], was presented by Kraus et al [118] and Kühl et al [119].

In addition to these theoretical and numerical advances, the superiority of MHE over traditional recursive estimation methods such as the Extended Kalman Filter (EKF) has been demonstrated empirically by Haseltine et al [89].

## 1.2. Dynamical system models

Many phenomena in nature can be described by dynamical models. The central idea is to model the natural process by relations between quantities and their rates of change, *e.g.* relying on laws of nature, thermodynamics, mechanics or electricity. This leads immediately to differential equations. Furthermore, any differential equation of arbitrary order can be transformed into a coupled set of first-order differential equations. A *state-space* representation is a dynamical model where vectors of inputs, outputs and states are related by first-order differential or difference equations. The states are the smallest possible subset of system variables that can represent the entire *state* of the system at any given time. The state-space representation provides a convenient and compact way to model and analyze systems with multiple inputs and outputs. Throughout this thesis we will consider linear and nonlinear state-space models in continuous or discrete time.

## Continuous-time models

Continuous-time systems can often be described by ordinary differential equations (ODEs) as follows

$$\dot{x}(t) = f(t,x(t),u(t),p(t),w(t)), \tag{1.1}$$

where $f$ is a nonlinear function, the time $t$ is the dependent variable, $x(t) \in \mathbf{R}^{n_x}$ denotes the state vector, $u(t) \in \mathbf{R}^{n_u}$ are given inputs, $p(t) \in \mathbf{R}^{n_p}$ is the set of parameters and $w(t) \in \mathbf{R}^{n_w}$ is a vector of state or process disturbances. The output equation may be given by the following expression

$$y(t) = h(t,x(t),u(t)) + v(t), \tag{1.2}$$

where $h$ is a nonlinear function and where $y(t), v(t) \in \mathbf{R}^{n_y}$ are respectively the output and the output disturbances or measurement errors.

If the model is affine in the variables $x$ and $w$, then the following linear continuous-time model is obtained

$$\dot{x}(t) = \bar{f}(t) + A(t)x(t) + G(t)w(t), \tag{1.3}$$
$$y(t) = \bar{h}(t) + C(t)x(t) + v(t). \tag{1.4}$$

Here the (time-varying) system matrices $A(t) \in \mathbf{R}^{n_x \times n_x}$, $G(t) \in \mathbf{R}^{n_x \times n_w}$, $C(t) \in \mathbf{R}^{n_y \times n_x}$ and the offsets $\bar{f}(t) \in \mathbf{R}^{n_x}$, $\bar{h}(t) \in \mathbf{R}^{n_y}$ are assumed to be known.

## Discrete-time models

Often the output is only measured at discrete sampling instants The state equation may also be discretized in advance, leading to the following nonlinear discrete-time model

$$x_{k+1} = f_k(x_k, u_k, p_k, w_k), \tag{1.5}$$
$$y_k = h_k(x_k, u_k) + v_k. \tag{1.6}$$

where $k$ denotes discrete time, $x_k \in \mathbf{R}^{n_x}$ is the state, $w_k \in \mathbf{R}^{n_w}$ is the state or process disturbance, $v_k \in \mathbf{R}^{n_y}$ is the output disturbance or measurement error and $y_k \in \mathbf{R}^{n_y}$ is the observed output.

If the model is affine in $x$ and $w$, then the following linear discrete-time description is obtained

$$x_{k+1} = f_k + A_k x_k + G_k w_k, \tag{1.7}$$
$$y_k = h_k + C_k x_k + v_k. \tag{1.8}$$

Here the (time-varying) system matrices $A_k \in \mathbf{R}^{n_x \times n_x}$, $G_k \in \mathbf{R}^{n_x \times n_w}$, $C_k \in \mathbf{R}^{n_y \times n_x}$ and the offsets $f_k \in \mathbf{R}^{n_x}$, $h_k \in \mathbf{R}^{n_y}$ are assumed to be known.

## 1.3. State estimation

This section defines the state estimation problem and its different facets. The reason of existence of state estimation is given by the following motivations and by the fact that state-space models have become the standard for advanced feedback control.

### Infer states from outputs

Typically not all states are measured either because it is too costly or simply because it is impossible. For example the average molecular weight of a polymer in a chemical reactor cannot be measured directly, but it can be computed based on viscosity measurements. On the other hand, concentrations of chemical components are often determined from temperature measurements, which are much easier to measure. Often the outputs are a subset of states, but in general this state-output relation may be more complex, *i.e.* given by the nonlinear mapping (1.5). Fortunately, states and outputs are all interconnected by the model equations. Therefore, it is well possible to estimate a large number of states (and parameters) from observations of a single output.

In a disturbance-free setting ($w_k, v_k = 0, \forall k$) the main challenge of state estimation is to retrieve the states that have generated the observed outputs, given the dynamical model. This is an inverse problem which is readily solved in the linear case but which requires the solution of a nonlinear combinatorial problem in case constraints are present and the state and/or output equations are nonlinear. When the model is nonlinear the inverse mapping may not even be one-to-one, hence multiple state sequences could have generated the observed output sequence.

Another goal of a deterministic state estimator is to recover from a wrong initial guess. Combining the model predictions with the information comprised in the output measurements allows for asymptotic convergence to the true state sequence.

### Retrieve states from noisy measurements

Disturbances enter the system at two places. Process disturbances, denoted by $w$, account for modelling errors as well as for process variations. Output disturbances $v$ account for modelling errors and for (random) sensor errors. Any mathematical model is a simplification of the processes occurring in reality and our only hope is that the model captures the most important dynamics. Hence the challenge of state estimation

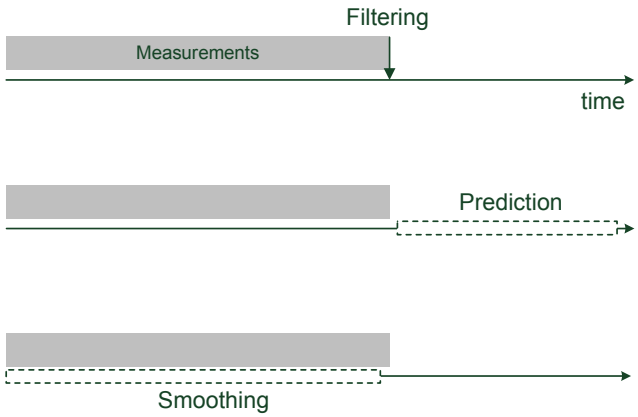



**Figure 1.2.** Illustration of filtering, prediction and smoothing. Using all available measurements up until the current time instant the goals are respectively to estimate the current state, future states or past states.

is to find good state estimates in the face of noisy measurements and process disturbances. Of course the performance of state estimation is limited by the system set-up, more precisely by the quality of the model and by the amount and quality of data (information richness).

### 1.3.1. Filtering, prediction and smoothing

The state estimation problem comes in different forms depending on the span of available measurements used for computing a certain state estimate. This is illustrated in Figure 1.2 and formalized by the following definition.

**Definition 1** (State estimation, filtering, prediction and smoothing)**.** *Given a sequence of output measurements $Y_l = \{y_0, \ldots, y_l\}$ generated by a process defined by one of the models of Section 1.2, the state estimation problem consists in computing an estimate of the state $x_k$ based on $Y_l$. If $k = l$ the estimation problem is called a filtering problem and the estimate $x_k$ a filtered state estimate. If $k > l$ the problem (state estimate) is a prediction problem (predicted estimate) and if $k < l$ it is a smoothing problem (smoothed estimate).*

As time evolves and more measurements are collected, the estimates of a state at a certain point in the past (a process called *fixed-point smoothing*) will become increasingly accurate as one might expect. There is, however, a lower bound to this accuracy which is defined by the amount of noise present in the system and by the characteristics of the model. Indeed, the more unstable the open-loop system is the harder it is

to control it, because more energy needs to be be applied to avoid the process from drifting away, but the easier it is to estimate the states because small perturbations in the state estimate lead to widely diverging trajectories.

On the other hand the accuracy of predictions tends to decrease the further ahead one wants to predict. The evolution of the prediction errors is governed by the open-loop system dynamics since the feedback mechanism that is inherent to the Kalman filter and to MHE (see Chapter 3) is broken.

As will become clear, in MHE a number of smoothed estimates and a filtered estimate are obtained in each time step. Subsequently, when combined with a predictive controller, the filtered (current) state estimate and the model are used to predict the future state trajectory.

## 1.4. Moving horizon estimation

### 1.4.1. Least-squares batch state estimation

In the previous section the goal of state estimation was defined as *finding the state sequence that is most likely given a sequence of observations and a system model.* Now, let us specify, what exactly is meant by *most likely*.

The objective is defined as

$$\mathcal{J}(T,\mathbf{x},\mathbf{w},\mathbf{v}) = \mathcal{J}_{\text{ic}}(x_0) + \mathcal{J}_{\text{proc}}(T,\mathbf{w}) + \mathcal{J}_{\text{out}}(T,\mathbf{v}), \tag{1.9}$$

where $T$ is the batch size (number of data points) and where $\mathbf{x} \in \mathbf{R}^{(T+1)n_x}$, $\mathbf{w} \in \mathbf{R}^{Tn_w}$ and $\mathbf{v} \in \mathbf{R}^{(T+1)n_y}$ denote the stacked vectors of states, process disturbances and output disturbances respectively.

The first term, $\mathcal{J}_{\text{ic}}$, is the cost associated with the initial condition. Usually, it is assumed that some prior information is available in the form of an initial state estimate $\hat{x}_0$ and a corresponding covariance matrix $P_0$, which allows the following definition

$$\mathcal{J}_{\text{ic}}(x_0) = \|x_0 - \hat{x}_0\|^2_{P_0^{-1}}. \tag{1.10}$$

Hence $P_0$ determines the weight that is given to the initial guess $\hat{x}_0$ relative to the other terms in the objective. If we have high (low) confidence in the estimate $\hat{x}_0$ then the cost of choosing $x_0$ far away from $\hat{x}_0$ is large (small).

The second term, $\mathcal{J}_{\text{proc}}$, is a penalization of the state or process disturbances.

$$\mathcal{J}_{\text{proc}}(T,\mathbf{w}) = \sum_{k=0}^{T-1} \|w_k\|^2_{Q_k^{-1}}. \tag{1.11}$$

Here $Q_k^{-1}$ provides a measure of confidence in the model.

The third term, $\mathcal{J}_{\text{out}}$, is a penalization of the output disturbances or measurement errors

$$\mathcal{J}_{\text{out}}(T,\mathbf{v}) = \sum_{k=0}^{T} \|v_k\|^2_{R_k^{-1}}. \tag{1.12}$$

$R_k^{-1}$ provides a measure of confidence in the measurement data.

The minimization is subject to one of the system models described in Section 1.2. For nonlinear discrete-time systems this yields the following nonlinear least-squares batch estimation problem

$$\begin{aligned} \min_{\mathbf{x},\mathbf{w},\mathbf{v}} \quad & \|x_0 - \hat{x}_0\|^2_{P_0^{-1}} + \textstyle\sum_{k=0}^{T-1} \|w_k\|^2_{Q_k^{-1}} + \sum_{k=0}^{T} \|v_k\|^2_{R_k^{-1}} \\ \text{s.t.} \quad & \begin{aligned} x_{k+1} &= f_k(x_k, u_k, p_k, w_k), \quad k = 0,\ldots,T-1, \\ y_k &= h_k(x_k, u_k) + v_k, \quad k = 0,\ldots,T, \end{aligned} \end{aligned} \tag{1.13}$$

with $\mathbf{x} = \{x_0,\ldots,x_T\}$ the unknown state sequence, $\mathbf{w} = \{w_0,\ldots,w_{T-1}\}$ the unknown process disturbances and $\mathbf{v} = \{v_0,\ldots,v_T\}$ the unknown output disturbances. The matrices $P_0$, $Q_k$ and $R_k$ are tuning parameters for reconciling the model with the measurements and the initial guess.

### 1.4.2. General batch state estimation

Other relevant state estimation problems can be formulated by altering the definitions of the objective terms. For example, instead of $\ell_2$ norms, one could work with $\ell_1$ norms or Huber penalty functions in order to robustify the estimation problem with respect to outliers in the measurements or with respect to parameter jumps. If the objective is composed of convex functions and if the model is linear, *i.e.* $f_k$ and $h_k$ are affine, the problem (1.13) is convex. Convex MHE formulations are discussed in Chapter 6.

In order to generalize the batch estimation problem the following modified definitions of the objective terms are proposed. The initial condition term is given by

$$\mathcal{J}_{\text{ic}}(x_0) = \rho(S_0^{-T}(x_0 - \hat{x}_0)). \tag{1.14}$$

where $\rho(\cdot)$ is an arbitrary penalty function and where $S_0$ is a weighting matrix. In case $\rho(\cdot)$ is the squared $\ell_2$ norm and $S_0$ is a (upper triangular) Cholesky factor of $P_0$, i.e. $P_0 = S_0^T S_0$, the newly defined initial condition (1.14) is equivalent to (1.10).

The process disturbance term is redefined as

$$\mathcal{J}_{\text{proc}}(T,\mathbf{w}) = \sum_{k=0}^{T-1} \rho(W_k^{-T} w_k), \tag{1.15}$$

where $\rho(\cdot)$ is again an arbitrary penalty function not necessarily identical for every $k$ and where $W_k$ is a weighting matrix. Again, the new definition coincides with the old definition (1.11) if the squared $\ell_2$ norm is chosen and if $W_k$ is a Cholesky factor of $Q_k$, i.e., $Q_k = W_k^T W_k$.

Finally, the output disturbance term becomes

$$\mathcal{J}_{\text{out}}(T,\mathbf{v}) = \sum_{k=0}^{T} \rho(V_k^{-T} v_k). \tag{1.16}$$

where for $\rho(\cdot)$ and $V_k$ the same statements hold as with the previous term, *i.e.* equivalence to (1.12) for $\rho(\cdot) = \|\cdot\|_2^2$ and $R_k = V_k^T V_k$.

### 1.4.3. Constrained state estimation

Because the batch state estimation problem is formulated as an optimization problem, inequality constraints on the optimization variables can easily be imposed. This is useful from an engineering viewpoint since in practice additional information about the process is often available in the form of constraints, *e.g.* quantities such as temperature, pressure, mass, postion, speed, acceleration, concentrations are often restricted to a certain range either by definition (*e.g.* nonnegativity) or by physical or practical limitations (*e.g.* for safety reasons). Incorporating this prior knowledge into the estimation problem typically improves the performance and convergence of the estimator [89]. In particular, when the system model is nonlinear the optimization problem is in general nonconvex with several local minima. In such a case non-physical optima may be excluded by restricting the search space yielding a dramatic increase in estimation performance, as shown by Haseltine and Rawlings [89]. Constraints may also be used to simplify the model. Explicitly enforcing constraints in the model, if at all possible, can introduce discontinuities which causes numerical difficulties when the model is used for estimation or control.

In this thesis the following state-disturbance path inequality constraints are considered

$$\begin{array}{lll} g_k(x_k, w_k) & \leq 0 & (\text{discrete} - \text{time}) \\ g(t, x(t), w(t)) & \leq 0 & (\text{continuous} - \text{time}) \end{array} \tag{1.17}$$

where $g$ is an arbitrary (nonlinear) function. After linearization (and discretization) these inequalities are reduced to the following mixed linear inequality constraints

$$D_k x_k + E_k w_k \leq d_k, \tag{1.18}$$

where $D_k \in \mathbf{R}^{r\times n_x}$, $E_k \in \mathbf{R}^{r\times n_w}$ are known matrices and $d_k \in\in \mathbf{R}^r$ is a known vector. As a special case, bound inequality constraints are considered

$$\begin{array}{lll} x_{\min} & \leq x_k \leq & x_{\max}, \\ w_{\min} & \leq w_k \leq & w_{\max}, \\ v_{\min} & \leq v_k \leq & v_{\max}. \end{array} \tag{1.19}$$

### 1.4.4. Moving horizon approximation

The batch state estimator described above cannot be applied to online estimation problems in general because the problem grows unbounded with increasing horizon. In order to bound the problem, people have proposed a moving horizon strategy [59, 148, 149, 155] relying on the ideas of dynamic programming.

Consider the objective function of the least-squares batch estimation problem (1.13) and let us rearrange it as follows

$$\begin{aligned} \mathcal{J}(T,\mathbf{x},\mathbf{w},\mathbf{v}) \quad = \quad & \|x_0 - \hat{x}_0\|^2_{P_0^{-1}} + \sum_{k=0}^{T-N-1} \|w_k\|^2_{Q_k^{-1}} + \|v_k\|^2_{R_k^{-1}} \\ & + \sum_{k=T-N}^{T-1} \|w_k\|^2_{Q_k^{-1}} + \sum_{k=T-N}^{T} \|v_k\|^2_{R_k^{-1}}. \end{aligned} \tag{1.20}$$

Because of the Markov property which arises from the state-space description, the last two terms depend only on the state $x_{T-N}$ and the model and measurements in the interval $[T-N,T]$. Therefore, by Bellman's *principle of optimality* [12], the least-squares batch estimation problem can be replaced by the following equivalent fixed-size estimation problem

$$\begin{array}{lll} \min_{\mathbf{x},\mathbf{w},\mathbf{v}} & \mathcal{Z}_{T-N}(x_{T-N}) + \sum_{k=T-N}^{T-1} \|w_k\|^2_{Q_k^{-1}} + \sum_{k=T-N}^{T} \|v_k\|^2_{R_k^{-1}} & \\ & x_{k+1} = f_k(x_k,u_k,p_k,w_k), & k = T-N,\ldots,T-1, \\ & y_k = h_k(x_k,u_k) + v_k, & k = T-N,\ldots,T, \\ \text{s.t.} & 0 \geq g_k(x_k,w_k) & k = T-N,\ldots,T-1, \\ & 0 \geq g_N(x_N), & \end{array} \tag{1.21}$$

complemented with the requirement that $x_{T-N}$ is reachable. Here $N$ is the horizon length and $\mathcal{Z}_{T-N}(x_{T-N})$ is the *arrival cost* which compactly summarizes past information. The basic strategy of moving horizon estimation is to solve a growing horizon

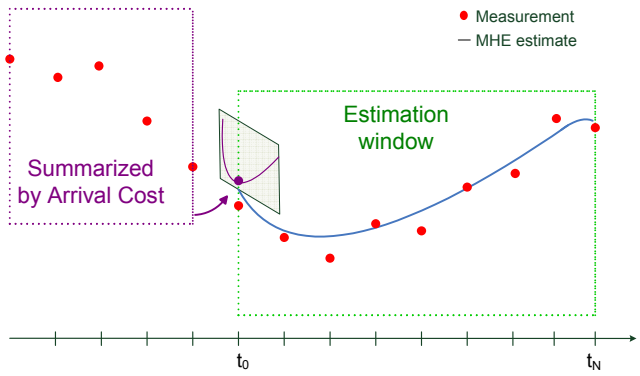



**Figure 1.3.** Illustration of the Moving Horizon Estimation approach: optimize over a finite window trading off measurement disturbance (**data** accuracy) and process disturbance (**model** accuracy) with an additional *arrival cost* which summarizes the data outside the window. When a new measurement comes in, the window is shifted and the *arrival cost* is updated.

batch estimation problem until $T = N$ and afterwards solve a fixed-size estimation problem on a moving horizon. At every iteration the oldest measurement is discarded and the new measurement is added. This is visualized in Figure 1.3.

Arrival cost is a fundamental concept in MHE as it allows to transform a problem which grows unbounded into an equivalent fixed-size problem [146]. In general, however, an analytical expression for the arrival cost does not exist and it should be approximated. We therefore replace the first term in the objective with an approximate arrival cost $\hat{Z}_{T-N}(x_{T-N})$. Rao et al [146, 148, 149] derived conditions for the approximate arrival cost to guarantee stability of MHE (see Chapter 2).

One strategy for computing an approximate arrival cost is to use a first-order Taylor expansion around the trajectory of past estimates. This is equivalent to applying an EKF recursion for the covariance update. In this case, the arrival cost is approximated as

$$\hat{Z}_{T-N}(x_{T-N}) = \|x_{T-N} - \hat{x}_{T-N}\|^2_{P^{-1}_{T-N}} \tag{1.22}$$

where $\hat{x}_{T-N}$ is the MHE estimate obtained at time $k = T-N$ and where $P_{T-N}$ is the covariance propagated by a Kalman filter recursion. This arrival cost approximation has several advantages. For linear unconstrained systems with quadratic objectives this arrival cost is exact since in this case the Kalman filter provides a recursive solution to the problem of estimating the current state (see Chapter 3 for a proof of equivalence). Furthermore, Rao et al [146, 148] have proved that this arrival cost approximation

yields a stable estimator for constrained linear problems. When the model is nonlinear, however, the EKF covariance update does not guarantee stability and additional measures are needed for guaranteed stability.

Other arrival costs have been proposed in the literature. Rawlings and Rajamani propose to approximate the arrival cost using a particle filter [153].

In Chapter 7 so-called *smoothed updates* are discussed. These updates generally show good performance while preserving equivalence in the unconstrained linear case.

The most important advantage of using a larger window size is that this mitigates problems due to poor initialization or poor arrival cost approximation [89]. Furthermore, by casting the estimation problem as an optimization problem MHE inherits the favorable properties of batch estimation being the flexibility of problem formulation, direct handling of constraints and ability to deal with nonlinear system models. The price to pay is an increased complexity as it is required to solve an optimization problem in every time step. However, as shown in this thesis, the computational cost can be made comparable to recursive methods by efficient structure-exploiting numerical methods.

### 1.4.5. Simultaneous state and parameter estimation

Often parameters are unknown in addition to the states and disturbances. The problem is typically tackled by imposing a model on the parameter variations and treating them as states. If no explicit model for the parameter variations is available the following model can be used

$$\begin{array}{lll} p_{k+1} & = p_k + \xi_k & (\text{discrete} - \text{time}) \\ \dot{p} & = \xi(t) & (\text{continuous} - \text{time}) \end{array} \tag{1.23}$$

where $\xi$ are additional disturbances which may be penalized in the objective by an arbitrary penalty function. Usually, parameters are modelled as constant (but unknown), *i.e.* $p_{k+1} = p_k$ or $\dot{p} = 0$, which makes sense for MHE when short horizons are considered since states typically vary much faster than parameters. In the case of batch estimation or MHE with large horizons, it is no longer justifiable to model parameters as constants as the process behavior can substantially vary over time. Then model (1.23) is usually imposed with a squared $\ell_2$ norm penalty on $\xi$, which is called a *random walk* model.

Often parameters enter the system in a highly nonlinear way making the simultaneous estimation of states and parameters a difficult problem. Even when a linear model is considered and one desires to estimate both the states and the system matrices the problem is already quite nonlinear. For such highly nonlinear estimation problems, optimization-based estimators such as MHE usually outperform recursive estimators.

## 1.5. Kalman filter

The Kalman filter yields a recursive solution to the unconstrained linear least-squares batch estimation problem (1.13). As noted before, such a recursive solution is only possible for very specific cases: unconstrained optimal control problems for linear systems with quadratic objective can be solved recursively by dynamic programming, leading to the well-known Kalman filter for estimation and the Linear Quadratic Regulator (LQR) for control. The Riccati equation is central to this recursive solution provided by the Kalman filter and LQR, and also to other main problems in the field such as $\mathcal{H}_infty$ control and the theory of dissipative systems and LMI's [195]. The Kalman filter is briefly reviewed in Chapter 2.

There is a strong connection between MHE and Kalman filtering. When constraints are inactive the linear least-squares batch estimator and the Kalman filter are equivalent. Even when a finite horizon is used (MHE), the estimates coincide with the Kalman filter estimates, hence are optimal in least-squares sense, if the arrival cost is updated using Kalman filter recursions. This equivalence between the Kalman filter and weighted least squares estimation is classical and has been treated in several papers and textbooks, *e.g.* [170], [195], [111, §10.6], [190, Ch. 1], [71]. A proof of equivalence between MHE and Kalman filter for linear unconstrained systems derived from the optimality conditions is given in Chapter 3.

### 1.5.1. Nonlinear extensions of the Kalman filter

Extensions of the Kalman filter have been developed for nonlinear systems. The popular Extended Kalman Filter (EKF) for example linearizes in each time step around the current estimate through a first order Taylor-series approximation. Although the EKF has been successfully applied in numerous applications, there have been several reports of poor estimation performance and even filter divergence (see Chapter 3 for a discussion).

There are a number of variations on the EKF. Higher order Taylor series expansions can be used in the filter equations [120]; when two terms of the expansion are used, the resulting EKF is called a second-order filter. Other algorithms use more linearization iterations in every time-step to improve the approximation accuracy; these filters are termed iterated EKF. Any one of these algorithms may be superior to standard EKF in a particular application, but there are no real guidelines nor theoretical proofs [6].

In the Unscented Kalman Filter (UKF) or Sigma Point Kalman filter, the probability density is approximated by a nonlinear transformation of a random variable, the *unscented transform* (UT), which is more accurate than the first-order Taylor series approximation in the EKF. The approximation utilizes a set of sample points, which

guarantees accuracy with the posterior mean and covariance to the second order for any nonlinearity.

## 1.6. A note on the deterministic versus stochastic interpretation of state estimation

In classical text books the Kalman filter is often derived from a stochastic viewpoint by making assumptions on the characteristics of the *noises* disturbing the system. However, the Kalman filter can perfectly be derived from a purely deterministic least squares formulation and this avoids unnecessary stochastic modeling assumptions which are often difficult to attach physical meaning to. The deterministic interpretation of the Kalman filter has been given in several works, see [170, 196] and the references therein. Willems [196] gives a very comprehensive self-contained treatment of Kalman filtering in its various existence forms from a deterministic perspective.

As indicated by Willems [196], the disturbances $\mathbf{w}$ and $\mathbf{v}$ should be interpreted as unknown inputs, which together with the (unknown) initial state $x_0$ determine the observations $\mathbf{y}$. Then the goal of state estimation is to find among all $(x_0, \mathbf{w}, \mathbf{v})$ which yield $\mathbf{y}$ the one that is *most likely* in the sense that it minimizes a specified objective, *i.e.* the least squares norm or square root of (1.9)-(1.12). By substituting these *optimal* disturbances in the system dynamics, an estimate of any related system variable can be obtained.

In the context of MHE, in which the estimation problem is explicitly formulated as an optimization problem, the deterministic interpretation is the only reasonable one since the fundaments of the probabilistic assumptions are contested when constrainst come into play. Moreover, for nonlinear MHE, the probabilistic approach naturally leads to stochastic differential equations which would unnecessarily complicate the numerical algorithms. Robertson et al [155] showed that bound constraints on the disturbances $w$ and $v$ may be interpreted as truncated normal distributions. But, as Rao [146] states, state constraints cannot (easily) be interpreted stochastically as they may correlate the disturbances and lead to acausality. In contrast, the deterministic intrepretation is perfectly satisfactory for MHE.

Note that, despite this plea for the deterministic approach, for ease of reference and because the terminology is so much established, we stick to the term covariance matrix although we could equally well speak about the inverse weighting, information or confidence matrix.

## 1.7. Model predictive control

The development of MHE, although proposed much earlier in many different forms throughout the literature, was pushed in the nineties motivated by the success of Model Predictive Control (MPC), its counterpart for control. Model Predictive Control (MPC) has gained widespread interest in both academia (see textbooks [31, 121, 128, 156]) and industry (see [145] for a survey) over the past decades. In a wide range of industries it has become the method of choice for advanced process control.

The ultimate goal in optimal control is to find a feedback law that minimizes a certain control objective over an infinite horizon, starting from the current state $x_0$ and subject to a process model (as described in Section 1.2) and constraints. Typically, but not necessarily, the objective is quadratic

$$J(\mathbf{x},\mathbf{u}) = \sum_{k=0}^{\infty} \|x_k\|_{Q_k}^2 + \|u_k\|_{R_k}^2.$$

where the weightings $Q_k$ and $R_k$ are tuning parameters. The optimal solution can be obtained from the solution of an infinite dimensional partial differential equation, called Hamilton-Jacobi-Bellman (HJB) equation. In general, a closed-form expression for the solution of the HJB equation does not exist. One exception is linear unconstrained systems with quadratic objectives. In this case, the solution follows from a matrix equation, *i.e.* a Riccati equation, and the resulting feedback controller is called LQR.

Another class of solution methods is based on *Pontryagin's Maximum Principle* [143] and proceed by maximizing the Hamiltonian matrix. Pontryagin's maximum principle is closely related to the HJB equation and provides conditions that an optimal trajectory must satisfy. However, while the HJB equation provides sufficient conditions for optimality, the minimum principle provides only necessary conditions. The maximum principle typically leads to an intricate multi-point boundary value problem.

Alternatively, and similarly to the MHE case, the infinite-horizon control problem can be replaced by an equivalent finite-horizon problem, due to the Markov property of the state-space model.

$$\begin{array}{lll} \min_{\mathbf{x},\mathbf{u}} & \sum_{k=0}^{N-1} \|x_k\|_{Q_k}^2 + \|u_k\|_{R_k}^2 + \mathcal{V}(x_N) & \\ & x_0 = \bar{x}_0, & \\ & x_{k+1} = f_k(x_k,u_k,p_k), & k=0,\ldots,N-1, \\ \text{s.t.} & 0 \geq g_k(x_k,u_k) & k=0,\ldots,N-1, \\ & 0 \geq g_N(x_N), & \end{array} \tag{1.24}$$

where $\mathcal{V}(x_N)$ is the terminal cost or end cost and $\bar{x}_0$ is the fixed initial state. The minimization is with regards to the state and control sequences $\{x_0,\ldots,x_N\}$ and $\{u_0,\ldots,u_{N-1}\}$ respectively with $u_k \in \mathbf{R}^{n_u}$. Constraints can be imposed on states and controls.

Since a closed-form expression of the terminal cost rarely exists, it should be approximated $F(x_N) = \hat{\mathcal{V}}(x_N)$. Mayne et al [131] derived conditions for stability of the MPC approximation. One popular strategy for approximation of the terminal cost is to assume that after the horizon the system can be controlled using LQR. In this case the (approximate) terminal cost is

$$\hat{\mathcal{V}}(x_N) = \|x_N\|^2_{P_N}.$$

where $P_N$ is the solution to the corresponding LQR discrete-algebraic Riccati equation. This approach, sometimes called dual-mode MPC, guarantees asymptotic stability for linear systems in the absence of disturbances [131].

The strategy of MPC is to solve the open-loop fixed-size optimization problem (*i.e.* (1.24) with approximated terminal cost), apply only the first element of the optimal input sequence to the process, obtain a new state estimate and repeat the procedure.

The unique combination of several important features distinguishes MPC from other control methods. First, analogous to MHE, it is possible to incorporate constraints and impose multivariate nonlinear models in a natural way. Constraints are even more relevant for the control problem than for the estimation problem, because safety limitations, environmental regulations and economic objectives force companies to operate their processes at the constraints. Second, the extensive research on MPC has led to formulations with guaranteed stability [131]. Finally, the ability to control processes proactively is a key feature of MPC. When disturbances are known in advance (*e.g.* grade changes in chemical processes), significant performance gains can be obtained in comparison with pure feedback control by incorporating these future disturbances into the control problem. A common motivation for the importance of this feature is by the example of driving a car; in the event of an upcoming turn one already takes this information into account by slowing down and changing to the outer lane in order to follow an efficient path.

In order to fully exploit the potential of MPC it is required that the underlying model and its parameters are constantly updated to take disturbances and plant-model mismatch into account. The performance of the closed loop system is directly influenced by the quality of the estimates. The combination of MHE and MPC yields a powerful and versatile strategy for advanced process control. States and parameters are adapted based on incoming measurements leading to improved prediction accuracies in turn leading to improved control performance. In addition, empirical studies [117] show that the MPC problem becomes easier to solve when estimates are more accurate because the predicted behavior resembles the true plant behavior more closely.

## 1.8. Similarities and differences between control and estimation

Optimal control and optimal estimation are closely related mathematical problems. For linear time-invariant systems without inequality constraints, there exists a *separation principle* which states that state estimator and controller can be designed separately. If they are both stable then the closed-loop system is also stable. If they are both optimal (*i.e.* Kalman filter and LQR) then the closed-loop system is also optimal. This combination of steady-state Kalman filter and steady-state LQR is called *Linear Quadratic Gaussian (LQG) compensation*.

In the Kalman filter, covariance matrices are propagated by a matrix Riccati recursion. For control, the Linear Quadratic Regulator (LQR) leads to a similar Riccati recursion and both recursions can easily be related using a conversion table for the matrices involved. Remarkably, the Riccati recursion for LQR runs backwards while the Kalman filter Riccati recursion runs forward, and therefore, this duality is only interesting for linear time-invariant models, since in the time-varying case the LQR is impractical as it involves an infinite backward matrix recursion. This so-called *duality* relation between Kalman filter and LQR was noted in the seminal papers of Kalman [112, 113].

Interesting similarities can also be discovered between MHE (1.21) and MPC (1.24) from their respective formulations. The MHE problem approximates the batch estimation problem by adding a weighting on the initial state (arrival cost or cost-to-arrive) while the MPC problem approximates the infinite optimal control problem by adding a weighting on the final state (terminal cost or cost-to-go). Conditions to ensure stability are represented by a dual set of inequalities for the arrival cost and the terminal cost, see [146]. Furthermore, in the MHE problem $w_k$ are the control variables similar to $u_k$ in the MPC problem. These observations suggest a duality between both problems. However, as pointed out by Todorov [178], it is not directly clear from the conversion tables of the Riccati recursion or from the similarity of the formulations in which sense estimation and control are dual problems. In order to make it clear, we will show that the unconstrained batch estimation problem can be rewritten into a form which can be interpreted as a control problem. Thereto, consider the following simple discrete linear time-varying (LTV) model (compare to the more general discrete LTV model (1.7))

$$\begin{aligned} x_{k+1} &= A_k x_k + w_k, \\ y_k &= C_k x_k + v_k. \end{aligned}$$

If, furthermore, the disturbance variables are eliminated, the estimation problem can be written as

$$\min_{\mathbf{x}} \quad \|x_0 - \hat{x}_0\|^2_{P_0^{-1}} + \sum_{k=0}^{T-1} \|x_{k+1} - A_k x_k\|^2_{Q_k^{-1}} + \sum_{k=0}^{T} \|y_k - C_k x_k\|^2_{R_k^{-1}}. \qquad (1.25)$$

This problem is equivalent to the following one

$$\min_{\mathbf{x},\mathbf{u}} \quad \sum_{k=0}^{T-1} \|u_k\|^2_{\bar{R}_k} + \sum_{k=0}^{T} \left( \|x_k\|^2_{\bar{Q}_k} + 2x_k^{\mathrm{T}} \bar{q}_k \right), \tag{1.26}$$

with

$$\begin{array}{lll} \bar{R}_k = Q_k^{-1} & & \\ \bar{Q}_0 = P_0^{-1} + C_0^{\mathrm{T}} R_0^{-1} C_0 & \bar{q}_0 = -\hat{x}_0 - C_0 k^{\mathrm{T}} R_0^{-1} y_0 & \\ \bar{Q}_k = C_k^{\mathrm{T}} R_k^{-1} C_k & \bar{q}_k = -C_k^{\mathrm{T}} R_k^{-1} y_k & 0 < k \leq N, \end{array}$$

and where the process disturbances have been replaced by controls, i.e. $u_k = w_k$. By this reformulation, we can see that the unconstrained batch estimation problem can be interpreted as reference-tracking optimal control problem with a reference trajectory specified by the observations and with a free intial state.

This free initial state vector $x_0$ is the most important difference with MPC. These extra degrees of freedom allow us to fit an observed output sequence according to a specified objective. Therefore the estimation problem is often referred to as an *inverse problem*. It must be noted that the addition of an initial condition typically increases the numerical conditioning as the extra degrees of freedom may result in an infinite number of solutions to the estimation problem.

Duality relations between MHE and MPC, are further complicated due to the presence of constraints and possibly nonlinear dynamics. It was shown by Goodwin, De Doná and coworkers [78, 79, 133] that the dual of the linear constrained MHE problem is a reverse-time nonlinear unconstrained control problem involving projected variables, a special instance of an MPC problem, and that there is no duality gap. Although this result is highly interesting from a theoretical view, there is no direct practical value to it since the dual problem is not easier to solve than the primal problem.

Note that the notion of duality in system theory is more vague than Lagrangian duality in optimization. Duality in system theory, as we showed in this section, means for example that a specific estimation problem can be rewritten and interpreted as a specific control problem. Both problems of course yield the same solution(s). Lagrangian duality, on the other hand, implies that the primal optimization problem has a corresponding dual problem where the Lagrange multipliers are the variables and the primal variables are the Lagrange multipliers. If there is no duality gap, the solutions to both problems are exactly the same. In some cases this duality relation can be exploited *e.g.* if the dual problem is easier to solve than the primal problem. See Chapter 2 for more details.

Figure 1.4 illustrates the various relations between MHE, the Kalman filter, LQR and MPC.

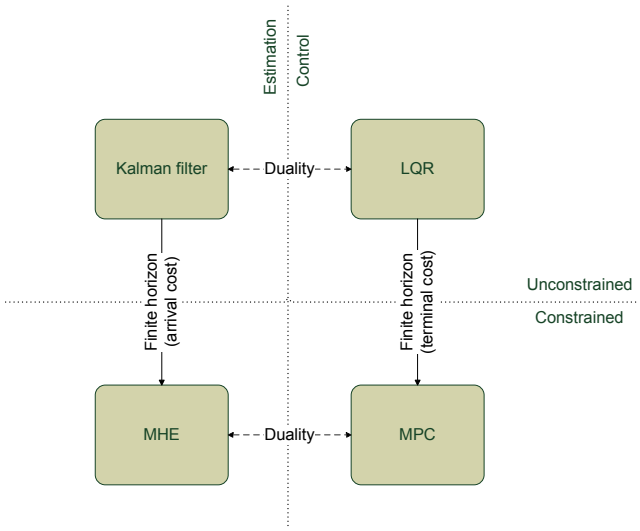



**Figure 1.4.** Relation of Moving Horizon Estimation (MHE) to Kalman filter, Linear Quadratic Regulator (LQR) and Model Predictive Control (MPC).

## 1.9. Motivation

Several factors justify the increasing research interest in real-time optimization-based estimation methods. First, the recursive solution provided by the Kalman filter is only possible when two very specific conditions are fulfilled: a quadratic objective and a linear system model. Whenever these conditions are not met, the problem is to be solved by optimization or by suboptimal recursive methods. Second, the enormous advances in computer technology , in numerical linear algebra and optimization of the last decades now make optimization-based estimation such as MHE a viable alternative to recursive estimation methods, *i.e.* derived from Kalman filtering. Third, due to its formulation as an optimization problem, inequality constraints can be incorprated in a natural way and the nonlinear model is direclty imposed over the entire window. It also adds flexibility for instance in formulating other useful estimation problems different from the traditional least-squares formulation. Finally, using a window (horizon) larger than one provides a higher degree of robustness in the presence of disturbances and allows to overcome problems of divergence commonly encountered with recursive methods such as the Extended Kalman Filter [89].

The development of MHE, although proposed much earlier in many different forms throughout the literature, was pushed in the nineties motivated by the success of MPC, an optimization-based control technique. MHE is often motivated as the dual problem of MPC, although this duality is not directly useful besides for its didactical and

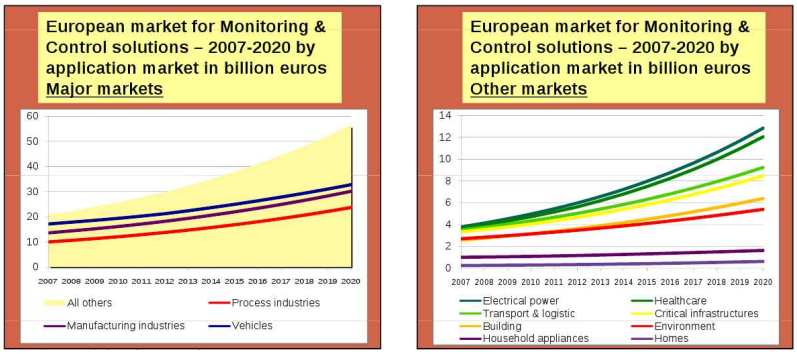



**Figure 1.5.** Predicted growth in major markets (left) and new markets (right) for monitoring (state estimation) and control. Graphs are taken from a report of a 2008 workshop by the European Commission DG Information Society and Media, available for download at [52].

theoretical importance (see Section 1.8).

Similar to the least-squares method and the Kalman filter, the research on MHE and MPC has been driven by applications. The number of applications has increased rapidly over the past decades (see [145]) and prospections are that the market for monitoring and control will continue to grow steadily over the next years [52] (see Figure 1.5). Typical applications are in petrochemicals, pulp and paper, food processing, metallurgy, pharmaceuticals or other areas with slow sampling rates. Examples of MHE applications in these areas can be found in the references [2, 45, 46, 64, 67, 95, 118, 130, 157, 201].

In recent years, a shift has started to occur towards applications with high sampling rates made possible by the development of dedicated algorithms for MPC and MHE. The work presented in this thesis can be situated along these research lines and proposes several dedicated algorithms for fast real-time MHE. The thesis demonstrates that MHE *is* applicable to systems with fast sampling times and forms an attractive alternative to recursive estimation methods because of its increased flexibility.

For linear systems, moving horizon estimation problems can be formulated as sparse convex quadratic programs (QPs) with equality and inequality constraints. Also for nonlinear MHE problems, sparse convex QP subproblems are obtained for instance in every iteration of an SQP approach.

Instead of eliminating the states to obtain a smaller but dense convex QP, it is advantageous to keep the original sparse and highly structured problem and to solve the corresponding system of optimality conditions by exploiting the structure. It is shown how this system of optimality conditions can be decomposed and solved by Riccati based methods. These solution methods fully exploit the inherent symmetry

and structure and furthermore provide improved robustness by using orthogonal transformations. It is argued in this thesis that for MHE problems, the solution method of choice should be Riccati based. This is in contrast to MPC where, depending on the system dimensions and the horizon length, the most effective solution strategy is either by Riccati methods (or sparse solvers) or by *condensing*, *i.e.* solving the reduced system obtained after variable elimination. In MHE, Riccati methods are favorable even for small horizons, because the dimension of the *controls* or process disturbances is typically comparable to the state dimension. Furthermore, the arrival cost update and final estimate covariance are obtained as a natural outcome of the solution process. In contrast, if the problems are solved using a general purpose (sparse) solver, the arrival cost is computed by running a Kalman filter in parallel while the final state covariance is computed by a local first order analysis.

When constraints are imposed, the resulting quadratic programming (QP) problems can be solved by active-set or interior-point methods. Interior point methods preserve the block diagonal structure of the KKT system and hence can be solved with modified Riccati recursions. These modifications are investigated in the context of MHE in this thesis. It is shown that square-root versions are especially useful for interior-point methods due to their increased numerical robustness. Active-set methods typically do not preserve the structure. To circumvent this problem, a Schur-complement active-set method is proposed which uses the unconstrained MHE solution as a starting point and proceeds by projecting onto the set of working constraints.

In order to guarantee robustness of MHE algorithms, robust penalty functions are investigated in the framework of MHE. It is shown in this dissertation that the use of convex penalty functions such as the $\ell_1$ norm or the Huber penalty function can dramatically improve the robustness of the algorithms while efficient structure-exploiting algorithms can still be derived yielding a computational performance comparable to the standard formulation. Robust norms can also be used for penalization of the parameter changes. Such a formulation allows fast detection of within-horizon parameter jumps, which is especially useful for moderate to large horizon lengths, where the standard MHE formulation would smooth out these parameter switches.

## 1.10. Structure of the thesis

The general structure of this doctoral dissertation is depicted in Figure 1.6.

## 1.11. Specific contributions

This thesis aims to bridge the gap between classical recursive estimation methods and optimization-based estimation methods. By introducing ideas and techniques from

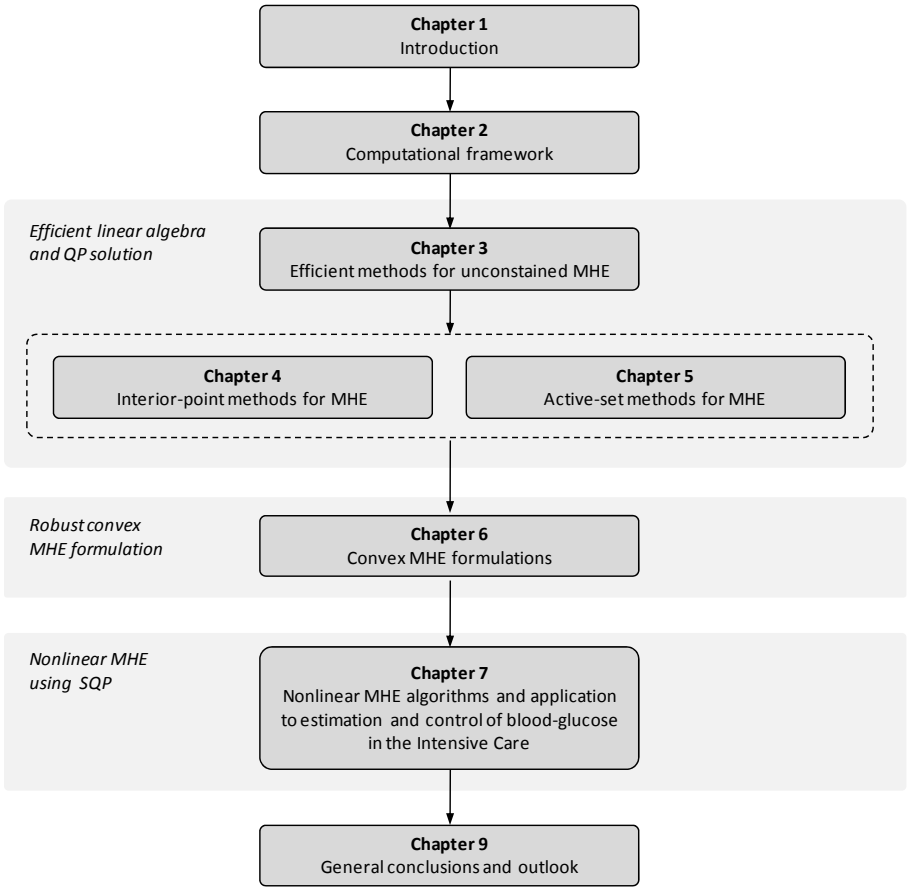



**Figure 1.6.** Structure of the thesis.

the well-developed fields of numerical linear algebra and optimal estimation into the emerging field of MHE, efficient solution methods for the resulting optimization problems are derived yielding fast and reliable algorithms.

In **Chapter 2** the mathematical foundation for both practical and theoretical study of MHE is provided. Stability of the constrained estimator is addressed and the most important concepts from linear algebra and optimization are introduced. A brief introduction to Kalman filtering and Kalman smoothing is given with a special focus on the different implementation forms.

**Chapter 3** addresses unconstrained MHE. It is shown that the decomposition of the KKT system by $LU$ or $LDL^{\mathrm{T}}$ factorizations leads to Riccati based solution methods. The methods fully exploit the structure inherent in the MHE problem. The square-root

version further exploits the symmetry in the system and employs orthogonalization methods yielding increased numerical robustness. Structured QR methods are proposed to reduce the computation times of the square-root Riccati based MHE method. A proof of equivalence of unconstrained MHE with the Kalman filter/smoother is given and the analogy with Riccati methods for MPC is discussed.

*Publications related to this chapter: [73, 90].*

**Chapter 4** extends the Riccati based methods of Chapter 3 to constrained MHE problems using a primal barrier interior-point method. It is shown how the barrier terms enter in the measurement updates and can be interpreted as *perfect measurements*. Several types of constraints are considered in the tailored methods. A hot-starting strategy is proposed and the benefit of a good initialization procedure is demonstrated. Finally, it is demonstrated that square-root recursions are extremely useful in an interior-point method since the condition number of the matrices in the factorization typically grows lograthmically for actively constrained components.

*Publications related to this chapter: [90].*

Active-set methods for the solution of constrained MHE problems are considered in **Chapter 5**. In particular, a Schur complement active-set method is proposed. The presented method starts from the unconstrained MHE solution, *i.e.* without inequality constraints, which can be computed using the Riccati based methods of Chapter 3 and solves a number of non-negativity constrained QPs in the reduced space of active constraints. A gradient projection method using projected Newton steps is proposed to solve the reduced QPs and Cholesky updates and downdates are employed to update factorizations at both outer and inner active set changes. By using square-root Riccati approach, the method involves just a small number of partial forward solves and backward solves. It is demonstrated that the method typically converges in only a few iterations.

*Publications related to this chapter: [93].*

**Chapter 6** addresses general convex MHE formulations. The focus is on two types of robust convex MHE formulations which are particularly useful in practical applications. First, robustness with regards to occasional outliers is investigated by means of Huber penalty MHE and $\ell_1$ penalty MHE. The former is shown to have excellent performance in terms of outlier rejection and estimation accuracy. Second, the joint estimation of states and parameters or inputs is considered. The resuling MHE problem is formulated as a convex cardinality problem yielding robustness with respect to rapid parameter changes, *i.e.* jumps or break points. It is shown that this leads to an MHE problem with $\ell_1$ penalty on the parameter variation and a small number of subsequent corrections to the $\ell_1$ norm MHE problem. Significant improvements in estimation performance are obtained using this procedure and a polishing step.

*Publications related to this chapter: [91, 92].*

In **Chapter 7** the application of MHE to nonlinear constrained problems is considered. This is done in a direct multiple shooting Gauss-Newton framework. The Gauss-Newton SQP iterations yield quadratic subproblems which can be solved efficiently using the methods presented in the previous chapters. The Chapter discusses the application of MHE and MPC to the normalization of blood-glucose of patients in the intensive care unit. The problem, model and data resulted from a collaborative research with the ICU of the University Hospital K.U.Leuven. An existing patient model is used for model-based estimation and control. The idea is to monitor and regulate the blood-glucose level to a normoglycemic range using measurements of glycemia and by advising an appropriate insulin dosing. The patient model is further complemented with a known disturbance of administered carbohydrate calories flow. It is demonstrated by numerical simulations that a strategy of model-based estimation and predictive control is able to provide an adequate regulation of blood-glucose taking into account model imperfections and large intra and inter patient variability.

*Publications related to this chapter: [44, 94, 185].*

CHAPTER 2

# Computational framework

*The goal of this chapter is to provide the mathematical foundation for both the theoretical and practical aspects of MHE. A framework for the stability properties of the moving horizon approximation is introduced. Furthermore, the most important concepts from numerical linear algebra and optimization which constitute the basis of future chapters are briefly reviewed.*

## 2.1. Stability framework for MHE

One important desirable property of any estimator is asymptotic stability. For MHE this framework emerged from the theoretical work that had been done earlier for MPC and is based on Lyapunov theory. For details we refer to [59, 146, 152].

Consider again the MHE problem with approximate arrival cost $\hat{Z}_{T-N}(\cdot)$

$$\begin{aligned} \min_{\mathbf{x},\mathbf{w},\mathbf{v}} \quad & \hat{Z}_{T-N}(x_{T-N}) + \sum_{k=T-N}^{T-1} \|w_k\|^2_{Q_k^{-1}} + \sum_{k=T-N}^{T} \|v_k\|^2_{R_k^{-1}} \\ \text{s.t.} \quad & \begin{aligned} x_{k+1} &= f_k(x_k, u_k, p_k, w_k), && k = T-N, \ldots, T-1, \\ y_k &= h_k(x_k, u_k) + v_k, && k = T-N, \ldots, T, \\ 0 &\geq g_k(x_k, w_k) && k = T-N, \ldots, T-1, \\ 0 &\geq g_N(x_N), \end{aligned} \end{aligned} \tag{2.1}$$

and define the stage cost as $L_k(w_k, v_k) = \sum_{k=T-N}^{T-1} \|w_k\|^2_{Q_k^{-1}} + \sum_{k=T-N}^{T} \|v_k\|^2_{R_k^{-1}}$.

**Definition 2 (Lower semi-continuity).** *A function $\sigma(\cdot)$ is lower semi-continuous at $x_0$ if for every $\varepsilon > 0$ there exists a neighborhood $\mathcal{U}$ of $x_0$ such that*

$$\sigma(x) \geq \sigma(x_0) - \varepsilon, \quad \forall x \in \mathcal{U}. \tag{2.2}$$

**Definition 3 (Lipschitz continuity).** *A function $\sigma(\cdot)$ is (globally) Lipschitz continuous if there exists a constant $K$ such that for all $x_1, x_2 \in \operatorname{dom} \sigma$,*

$$\|\sigma(x_2) - \sigma(x_1)\| \leq K\|x_2 - x_1\| \tag{2.3}$$

If a function is twice differentiable then it is Lipschitz continuous.

**Definition 4 ($\mathcal{K}$ function).** *A function $\sigma(\cdot)$ is a $\mathcal{K}$ function if it is continuous, strictly increasing and $\sigma(0) = 0$.*

Consider the nonlinear discrete-time system model of (2.1). Then the following definition can be given.

**Definition 5 (Uniform observability).** *A system is uniformly observable if there exists a positive integer $N_0$ and a $\mathcal{K}$ function $\phi(\cdot)$ such that for any two states $x_1$ and $x_2$*

$$\phi(\|x_1 - x_2\|) \leq \sum_{j=0}^{N_0-1} \|y(k+j;x_1,k) - y(k+j;x_2,k)\|, \quad \forall k \geq 0,$$

*where $y(k+j;x_1,k)$ denotes the output obtained at discrete time step $k+j$ when the model is initialized with $x_1$ at time step $k$.*

This uniform observability condition states that if the output residuals are small, then the estimation error is also small.

For a (discrete-time) linear system, the definition of uniform observability condition is given as follows.

**Definition 6 (Uniform observability for linear systems).** *A linear system is uniformly observable if the observability Grammian given by*

$$V_k = \sum_{j=0}^{N-1} (C_{k+j}A_{k+j}^j)^{\mathrm{T}} C_{k+j}A_{k+j}^j$$

*is positive definite for all $k \geq 0$ and $N \geq n_x$.*

In order to derive conditions for existence of a solution to the MHE problem, we introduce the following assumptions (see [149]).

**Assumption 1.** *The functions $f_k(\cdot)$ and $h_k(\cdot)$ are Lipschitz continuous for all $k \geq 0$.*

**Assumption 2.** *The stage cost $L_k(\cdot)$ is a continuous function for all $k \geq 0$ and the arrival cost $\hat{Z}_{T-N}(\cdot)$ is a continuous function.*

**Assumption 3.** *There exist $\mathcal{K}$ functions $\eta(\cdot)$ and $\gamma(\cdot)$ such that*

$$\eta(\|(w,v)\|) \leq \quad L_k(w,v) \quad \leq \gamma(\|(w,v)\|) \tag{2.4}$$
$$\eta(\|x-\hat{x}_0\|) \leq \quad \hat{Z}_k(x) \quad \leq \gamma(\|x-\hat{x}_0\|) \tag{2.5}$$

*for all $w$, $v$, $x$ and $\hat{x}_0$ satisfying the constraints and for all $k \geq 0$.*

Furthermore, it is necessary to impose similar conditions on the arrival cost. Let $\hat{\phi}_T$ be the objective function value at the solution for problem (2.1), then we require [149]

**Condition 1.** *There exists a $\mathcal{K}$ function $\overline{\gamma}(\cdot)$ such that*

$$0 \leq \quad \hat{Z}_T(z) - \hat{\phi}_T \quad \leq \overline{\gamma}(\|z-\hat{x}_T\|) \tag{2.6}$$

*for all $z$ satisfying the state constraints and $T \geq 0$.*

**Theorem 2.1.1 (Existence of a solution to MHE problem (2.1)).** *If the following conditions are fulfilled*

- *assumptions 1 - 3 hold*
- *the arrival cost satisfies condition 1*
- *the system is uniformly observable and $N \geq N_0$*

*then a solution to problem 2.1 exists for all $\hat{x}_0$ satisfying the state constraints and for all $T \geq 0$.*

*Proof.* See [149]. □

To guarantee stability of MHE, the arrival cost should satisfy the following condition [149]

**Condition 2.** *Let $\mathcal{R}_\tau^N$ denote the set of reachable states defined as*

$$\mathcal{R}_\tau^N = \{x(\tau; z, \tau-N, \{w\})\} \tag{2.7}$$

*where $x(\tau; z, \tau-N, \{w\})$ represents the state at time $\tau$ when the model is applied forward from initial condition $z$ at time $\tau-N$ using the disturbance sequence $w$.*

- *For $\tau \leq N$ the approximate arrival cost satisfies $\hat{Z}_\tau(\cdot) \leq Z_\tau(\cdot)$.*
- *For $\tau > N$ the approximate arrival cost satisfies*

$$\hat{Z}_\tau(p) \leq \hat{\phi}_{\tau-1} \tag{2.8}$$

*and $p$ equals the final state of the solution $\hat{\phi}_{\tau-1}$.*

**Theorem 2.1.2** (**Asymptotic stability of MHE**). *If the following conditions are fulfilled*

- *assumptions 1 - 3 hold*
- *the arrival cost satisfies conditions 1 and 2*
- *the system is uniformly observable and $N \geq N_0$*

*then for all $\hat{x}_0$ satisfying the state constraints, MHE is asymptotically stable for the system model of (2.1).*

*Proof.* See [149]. □

### 2.1.1. Linear time-invariant systems

For linear time-invariant (LTI) systems, *i.e.* system model (1.7) with $A_k \equiv A$, $G_k \equiv G$ and $C_k \equiv C$ for all $k \geq 0$, properties such as observability and asymptotic stability can be checked more easily.

In the sequel, let $\Lambda(A)$ denote the set of eigenvalues of the matrix $A$.

**Definition 7** (**Asymptotic stability**). *Consider the following disturbance-free and offset-free LTI system*

$$x_{k+1} = Ax_k, \tag{2.9}$$
$$y_k = Cx_k. \tag{2.10}$$

*The system is asymptotically stable, i.e. $x_k \to 0$, if and only if for all $\lambda \in \Lambda(A)$ holds $|\lambda| < 1$.*

If the observability matrix of order $N$ is defined as

$$O_N := \begin{bmatrix} C \\ CA \\ \vdots \\ CA^N \end{bmatrix},$$

then the following theorem holds.

**Theorem 2.1.3** (Uniform observability for linear time-invariant systems)**.** *A linear time-invariant system is uniformly observable if and only if* rank $(O_{n_x-1}) = n_x$.

*Proof.* see [14, 71]. □

Observability can also be formulated directly in terms of the pair $\{A,C\}$.

**Definition 8** ([14, 71] Uniform observability for linear time-invariant systems in terms of $\{A,C\}$)**.** *Let $\lambda \in \Lambda(A)$, then $\lambda$ is said to be an observable mode of $\{A,C\}$ if*

$$\text{rank}\,\left(\begin{bmatrix}\lambda I - A\\ C\end{bmatrix}\right) = n_x,$$

*otherwise, $\lambda$ is said to be an unobservable mode of $\{A,C\}$.*

Consider the following state equation with controls $u_k \in \mathbf{R}^{n_u}$ and control system matrix $B \in \mathbf{R}^{n_x \times n_u}$

$$x_{k+1} \quad = \quad Ax_k + Bu_k. \tag{2.11}$$

**Definition 9** ([14, 71] Detectability for linear time-invariant systems)**.** *The pair $\{A,C\}$ is detectable if all $\lambda \in \Lambda(A)$ with $|\lambda| \geq 1$ are observable modes of $\{A,C\}$.*

In words this definition states that if a system is observable or stable, then it is also detectable; if a system is unobservable or unstable, then it is detectable if its unobservable modes are stable.

The concepts of controllability and stabilizability are dual to observability and detectability, respectively. Algebraically this implies:

- $[A,C]$ is observable iff $[A^{\mathrm{T}},C^{\mathrm{T}}]$ is controllable,
- $[A,C]$ is detectable iff $[A^{\mathrm{T}},C^{\mathrm{T}}]$ is stabilizable.

## 2.2. Linear algebra

The basic idea behind methods for solving a system of linear equations is to transform the problem into one that is easy to solve [68, 203]. This is exactly what people do when they are asked to solve a set of linear equations by hand. One systematically eliminates variables, *i.e.* eliminate $x_1$ from all equations except the first by subtracting multiples of the first equation and so on, a procedure known as *row reduction* or *Gaussian elimination*.

Consider an $n \times n$ system of linear equations written in matrix form as $Mz = w$. Write $M = \begin{pmatrix} a & b^{\mathrm{T}} \\ c & D \end{pmatrix}$, where $b, c \in \mathbf{R}^{n-1}$, $D \in \mathbf{R}^{(n-1)\times(n-1)}$, and $a \in \mathbf{R}_0$. After applying the elimination process to the first column the following equivalent system is obtained

$$\begin{pmatrix} a & b^{\mathrm{T}} \\ 0 & D - ca^{-1}b^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} x_1 \\ x_{2:n} \end{pmatrix} = \begin{pmatrix} w_1 \\ w_{2:n} - ca^{-1}w_1 \end{pmatrix}. \tag{2.12}$$

Next the same process is applied to the $(n-1) \times (n-1)$ submatrix $D - ca^{-1}b^{\mathrm{T}}$ and so on, until $M$ is transformed into an upper triangular matrix. This system can be solved easily, *i.e.* the value of $x_1$ is immediately obtained as $w_1/a$, next $x_2$ is obtained after substituion of $x_1$ and so on.

### Block elimination, Schur complement and the matrix inversion lemma

The Gaussian elimination idea can be extended to linear systems $Mz = w$ with block-partitioning

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}. \tag{2.13}$$

Suppose $A$ is square and nonsingular and partition $z = \begin{pmatrix} x \\ y \end{pmatrix}$ and $w = \begin{pmatrix} u \\ v \end{pmatrix}$ conformally with $M$.

After eliminating the vector $x$ the problem reduces to solving a smaller linear system

$$\left(D - CA^{-1}B\right) y = v - CA^{-1}u. \tag{2.14}$$

**Definition 10** (**Schur complement**). *Consider a matrix $M$ with block partitioning as given by equation (2.13) and assume $A$ is square and nonsingular. Then the matrix $S_A = D - CA^{-1}B$ is called the Schur complement of $A$ in $M$. Likewise, the matrix $S_D = A - BD^{-1}C$ is called the Schur complement of $D$ in $M$.*

With this definition equation (2.14) can be written as

$$y = S_A^{-1}\left(v - CA^{-1}u\right). \tag{2.15}$$

Substituting this into the first block equation and collecting terms yields

$$x = \left(A^{-1} + A^{-1}BS_A^{-1}CA^{-1}\right) u - A^{-1}BS_A^{-1}v. \tag{2.16}$$

Hence, this derivation leads us to the following expression for the inverse of $M$

$$M^{-1} = \begin{pmatrix} A^{-1} + A^{-1}BS_A^{-1}CA^{-1} & -A^{-1}BS_A^{-1} \\ -S_A^{-1}CA^{-1} & S_A^{-1} \end{pmatrix}. \tag{2.17}$$

Similarly, if $D$ is nonsingular one could start by eliminating $y$ to obtain

$$x = S_D^{-1}\left(u - BD^{-1}v\right). \tag{2.18}$$

where $S_D$ is the Schur complement of $D$ in $M$. Substituting this into the second block equation and collecting terms yields

$$y = \left(D^{-1} + D^{-1}CS_D^{-1}BD^{-1}\right)v - D^{-1}CS_D^{-1}u. \tag{2.19}$$

Putting everything together leads us to the expression

$$M^{-1} = \begin{pmatrix} S_D^{-1} & -S_D^{-1}BD^{-1} \\ -D^{-1}CS_D^{-1} & D^{-1} + D^{-1}CS_D^{-1}BD^{-1} \end{pmatrix}. \tag{2.20}$$

Equating the elements in the equivalent representations (2.20) and (2.17) gives immediately the following lemma known as the ***Matrix Inversion Lemma***.

**Lemma 2.2.1.** *Let $A$, $D$ and $D - CA^{-1}B$ be nonsingular square matrices. Then*

$$\left(A - BD^{-1}C\right)^{-1} = A^{-1} + A^{-1}B\left(D - CA^{-1}B\right)^{-1}CA^{-1}. \tag{2.21}$$

*and*

$$\left(A - BD^{-1}C\right)^{-1}BD^{-1} = A^{-1}B\left(D - CA^{-1}B\right)^{-1}. \tag{2.22}$$

### Symmetry and positive definiteness

When a matrix $M$ is symmetric or symmetric and positive (semi)definite, this can be exploited in block elimination and variants of the Schur complement and the matrix inversion lemma can be formulated. The following theorem (see [203]) will be useful.

**Theorem 2.2.2.** *Let $M$ be a symmetric matrix partitioned as*

$$M = \begin{pmatrix} A & B \\ B^{\mathrm{T}} & D \end{pmatrix},$$

*in which $A$ is square and nonsingular. Then*

- $M \succ 0$ *iff both* $A \succ 0$ *and* $S_A = D - B^{\mathrm{T}}A^{-1}B \succ 0$.
- $M \succeq 0$ *iff both* $A \succ 0$ *and* $S_A = D - B^{\mathrm{T}}A^{-1}B \succeq 0$.

*Consequently if* $M \succeq 0$ *and* $A \succ 0$*, then* $D \succeq S_A$ *and*

$$\det(D) \geq \det(S_A) = \det(M)/\det(D) \geq 0.$$

### Generalized inverse and generalized Schur complement

In the definition (10) of the Schur complement $A$ was assumed to be square and nonsingular. By introducing a generalized inverse this definition can be extended allowing $A$ to be an arbitrary matrix.

**Definition 11 (Generalized inverse).** *A generalized inverse for a given* $m \times n$ *matrix* $M$ *is an* $n \times m$ *matrix denoted by* $M^-$ *(not necessarily unique) such that* $MM^-M = M$.

If $M$ is square and nonsingular, its only generalized inverse is the ordinary inverse.

The best known generalized inverse is the *Moore-Penrose generalized inverse* or ***pseudoinverse*** denoted as $M^\dagger$, which is a unique matrix satisfying

$$MM^\dagger M = M, \quad M^\dagger MM^\dagger = M^\dagger, \quad (M^\dagger M)^{\mathrm{T}} = M^\dagger M. \tag{2.23}$$

If

$$M = U \begin{pmatrix} \Sigma & 0 \\ 0 & 0 \end{pmatrix} V^{\mathrm{T}}, \tag{2.24}$$

is a singular value decomposition of $M$ in which $\Sigma$ has positive diagonal elements and $U$ and $V$ are orthogonal, then

$$M^\dagger = V \begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^{\mathrm{T}}. \tag{2.25}$$

Now, the pseudoinverse is used to define the generalized Schur complement.

**Definition 12 (Generalized Schur complement).** *The generalized Schur complement of* $A$ *in* $M$ *with* $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ *is defined as* $S_A = D - CA^\dagger B$.

## Matrix factorizations

Gaussian elimination describes a process of transforming a matrix into upper triangular form by reducing it column by column. Alternatively, a matrix can be expressed as a *factorization* or *decomposition*, *i.e.* a product of matrices with special properties. The *singular value decomposition* ($M = U\Sigma V^{\mathrm{T}}$) for instance expresses a matrix as a product of two orthogonal matrices ($U$ and $V$) and a diagonal matrix ($\Sigma$). The factorization associated with Gaussian elimination is the $LU$ ***factorization***

$$M = PLU$$

where $P$ is a permutation matrix, $L$ is lower triangular and $U$ is unit upper triangular (upper triangular containing only ones on the diagonal). Every nonsingular matrix $M \in \mathbf{R}^{n\times n}$ can be factorized in this form. The $LU$ factorization is the standard approach for solving a general linear system [29]. When the matrix $M$ is sparse the $LU$ factorization usually includes both row and column pivoting, *i.e.* $M = P_1LUP_2$ with $P_1$ and $P_2$ permutation matrices. Also when $M$ contains very large elements including row and column pivoting typically improves numerical stability.

Suppose $M$ is nonsingular symmetric and suppose it can be factorized as $M = LU$ without row interchanges. Then, by the uniqueness of the $LU$ factorization, $M$ can be written as

$$M = LDL^{\mathrm{T}}$$

with $D = \mathrm{diag}(u_{ii})$. In order to preserve symmetry interchanges must be applied to both rows and columns yielding a factorization

$$M = PLDL^{\mathrm{T}}P^{\mathrm{T}}. \tag{2.26}$$

This factorization is known as the $LDL^{\mathrm{T}}$ ***factorization*** or ***indefinite Cholesky decomposition***. Unfortunately, in contrast to the $LU$ factorization existence is not guaranteed for the $LDL^{\mathrm{T}}$ factorization. For example [68] the matrix $M = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ cannot be factorized by (2.26).

When $M$ is symmetric positive definite the $LDL^{\mathrm{T}}$ factorization always exists and all elements of $D$ are strictly positive. This allows us to write

$$M = LDL^{\mathrm{T}} = LD^{\frac{1}{2}}D^{\frac{1}{2}}L^{\mathrm{T}} = \bar{L}\bar{L}^{\mathrm{T}} = \bar{R}^{\mathrm{T}}\bar{R}, \tag{2.27}$$

which is known as ***Cholesky factorization***. The lower (upper) triangular matrix $\bar{L}$ ($\bar{R}$) is uniquely determined by $M$ and is called the *Cholesky factor* of M.

Every nonsingular matrix $M \in \mathbf{R}^{n\times n}$ can be factored as

$$M = QR$$

where $Q$ is an orthogonal matrix ($QQ^{\mathrm{T}} = Q^{\mathrm{T}}Q = I$, hence $Q^{-1} = Q^{\mathrm{T}}$) and $R$ is upper triangular. This is the $QR$ ***factorization*** and will be used frequently throughout this doctoral thesis. Note that $Q$ does not need to be computed explicitly to solve the linear system, only matrix-vector products $Q^{\mathrm{T}}w$ are needed which saves storage space and computation time. It is important to note that orthogonal transformations preserve Euclidean length of each column and therefore no error build-up occurs (in contrast to Gaussian eleminiation for example).

It can be shown that the $QR$ method when applied to symmetric matrices yields an eigenvalue decomposition, *i.e.* $AQ = Q\Lambda$ or $R = Q\Lambda$ with $\Lambda$ the diagonal matrix of eigenvalues.

The $QR$ factorization is most often applied to linear least-squares problems (overdetermined systems) and is also useful for underdetermined linear systems [29, 68].

A numerically robust way of achieving triangular form involves orthogonal matrices. Such factorization methods are referred to as *orthogonal triangularization* methods. The most popular techniques for orthogonal triangularization uses Householder or Givens transformations.

## 2.3. Convex optimization

Convexity is a fundamental concept in optimization [29, 136]. The class of convex programming problems includes linear least-squares, linear programming and quadratic programming problems, all of which have by now a fairly complete theory and can be solved numerically efficiently [29]. In this section a brief overview is given of the theory and numerical solution methods for some convex programming problems that are used in this thesis.

In the sequel the following standard form of nonlinear optimization problem (Nonlinear Program or NLP) is used

$$\begin{array}{lll} \min_x & f_0(x) & \\ \text{s.t.} & f_i(x) \leq 0, \quad i = 1,\ldots,m, & \\ & h_i(x) = 0, \quad i = 1,\ldots,p, & \end{array} \tag{2.28}$$

with $x \in \mathbf{R}^n$. The domain $\mathcal{D} = \bigcap_{i=0}^{m} \mathrm{dom}(f_i) \cap \bigcap_{i=1}^{p} \mathrm{dom}(h_i)$ is assumed to be nonempty.

### 2.3.1. Definitions

**Definition 13 (Convex set).** *A set $\mathcal{S} \subseteq \mathbf{R}^n$ is convex iff for any two points $x_1, x_2 \in \mathcal{S}$ all convex combinations of these points also lie within the set $\mathcal{S}$:*

$$(1-\theta)x_1 + \theta x_2 \in \mathcal{S}, \qquad \forall \theta \in [0,1], \forall x_1, x_2 \in \mathcal{S}.$$

**Definition 14 (Affine function).** *A function $f : \mathbf{R}^n \to \mathbf{R}$ is affine if it is the sum of a linear function and a constant, i.e. :*

$$f(x) = a^{\mathrm{T}}x + b, \qquad a \in \mathbf{R}^n, b \in \mathbf{R}.$$

It is worth clarifying that in this thesis, as it is frequent in control and estimation theory, the term linear function is sometimes used for functions which are actually affine.

**Definition 15 (Convex function).** *A function $f : \mathbf{R}^n \to \mathbf{R}$ is convex iff its domain* $\mathrm{dom}(f)$ *is convex and for every $x_1, x_2 \in \mathrm{dom}(f)$ and $\theta \in [0,1]$ following inequality holds*

$$f((1-\theta)x_1 + \theta x_2) \leq (1-\theta) f(x_1) + \theta f(x_2). \tag{2.29}$$

**Definition 16 (Strictly convex function).** *A function $f : \mathbf{R}^n \to \mathbf{R}$ is strictly convex iff its domain* $\mathrm{dom}(f)$ *is convex and strict inequality holds in (2.29) for $x1 \neq x2$ and $\theta \in (0,1)$.*

**Definition 17 (Convex optimization problem).** *Problem (2.28) is a convex optimization problem if the objective and inequality constraint functions, $f_0, \ldots, f_m$ are convex and the equality constraint functions are affine, i.e. $h_i(x) := a_i^{\mathrm{T}}x - b_i$ with $a_i \in \mathbf{R}^n, b_i \in \mathbf{R}$ for $i = 1, \ldots, p$.*

The following theorem explains the importance of convexity.

**Theorem 2.3.1** (see [29, 136])**.** *If problem (2.28) is a convex optimization problem then any local solution is also a global solution and the solution set is itself convex. Moreover, if the objective $f_0$ is strictly convex, the solution is unique if it exists.*

### 2.3.2. Duality and Karush-Kuhn-Tucker conditions

The *Lagrangian* associated with the standard optimization problem (2.28) is defined as

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

The vectors $\lambda$ and $\nu$ are called *Lagrange multipliers* associated with the inequality and equality constraints respectively.

The *Lagrange dual function* $g : \mathbf{R}^m \times \mathbf{R}^p \to \mathbf{R}$ is defined as the minimum value of the Lagrangian over $x$: for $\lambda \in \mathbf{R}^m$, $\nu \in \mathbf{R}^p$,

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

Note that the dual function will take on the value $-\infty$ when the Lagrangian is unbounded below. The dual function is concave even when the problem (2.28) is not convex.

**Proposition 2.3.2.** *The dual function yields a lower bound on the optimal value $f_0(x^\star)$ of the problem (2.28), i.e.*

$$g(\lambda, \nu) \leq f_0(x^\star), \quad \forall\, \lambda \geq 0, \nu. \tag{2.30}$$

See [29] for a proof.

For any primal feasible $x$ and for any $\lambda \geq 0$ and $\nu$ the difference $f_0(x) - g(\lambda, \nu)$ is non-negative and is called the *duality gap*. The optimal lower bound can be found by maximizing the dual function with respect to the Lagrange multipliers which leads to the following optimization problem

$$\begin{array}{ll} \max & g(\lambda, \nu) \\ \text{s.t.} & \lambda \geq 0. \end{array} \tag{2.31}$$

This problem is called the *Lagrange dual problem* associated with the problem (2.28). The original problem (2.28) is called the *primal problem*.

The dual problem (2.31) is a convex optimization problem, since the objective to be maximized is concave and the constraint is convex. This is the case whether or not the primal problem is convex.

When the equality holds for expression (2.30) at the solution, *i.e.* $g(\lambda^\star, \nu^\star) = f_0(x^\star)$, the optimal duality gap is zero, and it is said that *strong duality* holds. Otherwise, one speaks of *weak duality*.

An important property is that for feasible convex primal problems strong duality holds if the constraints are linear and $\mathrm{dom}(f_0) = \mathbf{R}^n$, see [29] for a proof. As a result strong duality holds for the QPs considered in Chapters 3 to 4.

Under the assumption of strong duality, it follows from the definition of the dual function that

$$f_0(x^\star) = g(\lambda^\star, \nu^\star) \leq \mathcal{L}(x^\star, \lambda^\star, \nu^\star). \tag{2.32}$$

Writing out the Lagrangian yields

$$f_0(x^\star) \leq f_0(x^\star) + \sum_{i=1}^{m} \lambda_i^\star f_i(x^\star), \tag{2.33}$$

where the term of the equality constraints vanishes because $h(x^\star) = 0$. Thus it follows that

$$\sum_{i=1}^{m} \lambda_i^\star f_i(x^\star) \geq 0. \tag{2.34}$$

But since $\lambda^\star \geq 0$ and $f_i(x^\star) \leq 0$, it also holds that

$$\lambda_i^\star f_i(x^\star) \leq 0, \quad i = 1, \ldots, m. \tag{2.35}$$

Combining the two expressions (2.34) and (2.35) leads to the conclusion that

$$\lambda_i^\star f_i(x^\star) = 0 \quad i = 1, \ldots, m. \tag{2.36}$$

This is called the *complementary slackness* condition. This condition states that for convex problems for which strong duality holds every active inequality has a corresponding strictly positive Lagrange multiplier while every inactive inequality must have a zero Lagrange multiplier.

Now, if all functions in problem (2.28) are assumed to be differentiable, then it follows that the gradient of $\mathcal{L}(x, \lambda^\star, \nu^\star)$ vanishes at $x = x^\star$. This means that for any optimization problem with differential objective and constraint functions for which strong duality holds, the triplet $(x, \lambda, \nu)$ can only be (primal and dual) optimal if the following set of conditions, known as the **Karush-Kuhn-Tucker (KKT) optimality conditions**, are satisfied

$$\begin{aligned}
\nabla f_0(x^\star) + \textstyle\sum_{i=1}^{m} \lambda_i^\star \nabla f_i(x^\star) + \sum_{i=1}^{p} \nu_i^\star \nabla h_i(x^\star) &= 0, \\
f_i(x^\star) &\leq 0, \quad i = 1, \ldots, m \\
h_i(x^\star) &= 0, \quad i = 1, \ldots, p \\
\lambda_i^\star &\geq 0, \quad i = 1, \ldots, m \\
\lambda_i^\star f_i(x^\star) &= 0, \quad i = 1, \ldots, m
\end{aligned} \tag{2.37}$$

These KKT conditions are the first order necessary conditions for optimality for constrained optimization, hence are the equivalent of $\nabla f(x^\star) = 0$ in unconstrained optimization. For convex optimization problems the KKT conditions are also sufficient first order optimality conditions. As a consequence the KKT conditions are necessary and sufficient optimality conditions for the QPs considered in Chapters 3 to 4.

### 2.3.3. Linear least-squares

The linear least-squares problem (LS) can be formulated as

$$\min_x \|Ax-b\|_2^2, \quad A \in \mathbf{R}^{m\times n}, b \in \mathbf{R}^m. \tag{2.38}$$

It can be reduced to solving a set of linear equations

$$(A^{\mathrm{T}}A)x^* = A^{\mathrm{T}}b, \tag{2.39}$$

or in other words yields an analytical solution given by

$$x^* = A^{\dagger}b, \tag{2.40}$$

where $A^{\dagger} = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}$ is the pseudoinverse of $A$.

#### LS and the Schur complement

Consider a block partitioned matrix $M = \begin{pmatrix} Q & S \\ S^{\mathrm{T}} & R \end{pmatrix} \succeq 0$. Then, the least-squares problem over some of the variables has an analytic expression as follows

$$g(x) = \min_y \begin{pmatrix} x \\ y \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} Q & S \\ S^{\mathrm{T}} & R \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = x^{\mathrm{T}}\bar{Q}x \tag{2.41}$$

where $\bar{Q} = Q - SR^{\dagger}S^{\mathrm{T}}$ is the Schur complement of $R$ in $M$.

#### Numerical solution methods for LS

The classical method for solving the least-squares problem (2.38), which can be traced back to the work of Gauss, proceeds by forming the normal equations [22]

$$A^{\mathrm{T}}Ax = A^{\mathrm{T}}b \tag{2.42}$$

and solving them through a Cholesky decomposition of the symmetric positive definite matrix $M = A^{\mathrm{T}}A$, *i.e.* $M = R^{\mathrm{T}}R$ with $R$ upper triangular. In statistical applications this is known as the *square-root method*. The normal equations approach can however give rise to numerical difficulties. First, some significant digits might get lost during explicit formation of $M = A^{\mathrm{T}}A$ and $M$ may not be positive definite or even nonsingular due to round-off. Second, the normal equations approach introduces errors larger than

those inherent in the problem due to the fact that its solution depends upon the square of the condition number of $A$. Methods based on orthogonal triangularization which work directly on $A$ such as the $QR$ factorization on the other hand have shown to possess very good stability properties.

For $A \in \mathbf{R}^{m\times n}$ with $m \geq n$ there exists a $QR$ factorization such that

$$Q^{\mathrm{T}}A = \begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{2.43}$$

where $Q \in \mathbf{R}^{m\times m}$ is orthogonal and $R \in \mathbf{R}^{n\times n}$ is upper triangular with nonnegative diagonal elements. See [22] for a proof. If $A$ has full rank ($\mathrm{rank}(A) = n$) then $R$ has strictly positive diagonal elements, hence is nonsingular. Then one can write

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1 R. \tag{2.44}$$

Here, $Q_1$ and $R$ are uniquely determined. The matrix $Q_2$ is in general not uniquely determined.

The following theorem shows how the $QR$ factorization may be used to solve the linear least-squares problem

**Theorem 2.3.3** (see [68])**.** *Let $A \in \mathbf{R}^{m\times n}$, $m \geq n$ and $b \in \mathbf{R}^m$ be given. Assume that* $\mathrm{rank}(A) = n$ *and that a QR factorization is computed such that*

$$Q^{\mathrm{T}}A = \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad Q^{\mathrm{T}}b = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}. \tag{2.45}$$

*Then the least-squares solution $x^*$ and the corresponding residual $r = b - Ax^*$ satisfy*

$$Rx^* = c_1, \qquad \|r\|_2 = \|c_2\|_2. \tag{2.46}$$

*Proof.* Since $Q$ is orthogonal the objective can be expressed as

$$\|Ax - b\|_2^2 = \|Q^{\mathrm{T}}(Ax - b)\|_2^2 = \|Rx - c_1\|_2^2 + \|c_2\|_2^2. \tag{2.47}$$

And because $R$ is nonsingular by the full rank condition on $A$ the minimum is attained at $x^* = R^{-1}c_1$ and its residual equals the norm of $c_2$. □

### 2.3.4. Linear programming

A Linear Program (LP), another important class of convex programming problems, can be written as

$$\begin{array}{ll} \min_x & c^{\mathrm{T}}x \\ \text{s.t.} & Ax \leq b, \\ & Cx = d, \end{array} \tag{2.48}$$

with $c \in \mathbf{R}^n$, $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$, $C \in \mathbf{R}^{p\times n}$, $d \in \mathbf{R}^p$.

### 2.3.5. Quadratic programming

A convex optimization problem is called a Quadratic Program (QP) if the objective funtion is convex and the constraint functions are affine. It can be written as

$$\begin{array}{ll} \min_x & \frac{1}{2}x^{\mathrm{T}}Hx + g^{\mathrm{T}}x \\ \text{s.t.} & Ax \leq b, \\ & Cx = d, \end{array} \tag{2.49}$$

where $H \in \mathbf{R}^{n\times n}$ is symmetric positive semidefinite, $g \in \mathbf{R}^n$, $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$, $C \in \mathbf{R}^{p\times n}$, $d \in \mathbf{R}^p$.

When linear constraints are added to the least-squares problem (2.38) the solution can no longer be computed analytically, but a QP must be solved.

### 2.3.6. Second-order cone programming

A constraint of the form

$$\|Ax + b\|_2 \leq c^{\mathrm{T}}x + d, \tag{2.50}$$

is a *second-order cone constraint*. A convex optimization problem with such a constraint is called a Second-order Cone Program (SOCP) and can be written as

$$\begin{array}{ll} \min_x & \frac{1}{2}f^{\mathrm{T}}x \\ \text{s.t.} & \|Ax + b\|_2 \leq c^{\mathrm{T}}x + d, \\ & Fx = g, \end{array} \tag{2.51}$$

where $f, c \in \mathbf{R}^n$, $d \in \mathbf{R}$, $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$, $F \in \mathbf{R}^{p\times n}$, $g \in \mathbf{R}^p$. Second-order Cone Programs are more general than (and include) Quadratic Programs.

## 2.4. Nonlinear optimization

If the objective function and/or some of the equality or inequality constraints are nonlinear, the mathematical program is called a nonlinear programming problem. Even when the objective and constraint functions are smooth, the general nonlinear optimization problem can be very difficult to solve. Indeed, for convex problems any local solution is also a global solution (Theorem 2.3.1), however, for non-convex problems the KKT conditions can only characterize local optimality. Global optimization is used for problems with a small number of variables, where computing time is not critical, and the value of finding the true global solution is very high. Therefore, only local optimization methods are considered in this thesis. This should not temper our ambition of finding *good solutions* though, since in moving horizon estimation (and control) a good initial guess is often available and if inequalities can be imposed, the search region can often be dramatically reduced such that *bad* local optima are excluded. Furthermore, with the help of efficient numerical techniques for dynamic programming such as multiple shooting convergence to good local optima, in many cases even the global optimum, is typically obtained.

Nonlinear optimization algorithms are iterative processes which differ by the strategy to move from one iterate to the next. The methods typically involve trade-offs between speed of convergence and computer storage, between robustness and computational speed, and so on. In this thesis so-called Newton-type methods, which make use of first and/or second order derivatives, are employed.

### 2.4.1. Newton's method for unconstrained optimization

The generic unconstrained nonlinear problem is

$$\min_x \quad f_0(x), \tag{2.52}$$

with $x \in \mathbf{R}^n$. Assume $f_0$ is twice continuously differentiable.

This nonlinear problem is solved in (exact) Newton's method by minimizing in every iteration a quadratic approximation, *i.e.* the next iterate $x^{k+1} = x^k + p^k$ is found by minimizing

$$\begin{aligned} m_k(x^k + p) &= f_0(x^k) + \nabla f_0(x^k)^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} \nabla^2 f_0(x^k) p & (2.53) \\ &\approxeq f_0(x^k + p) & (2.54) \end{aligned}$$

with respect to $p$. Here $\nabla f_0(x^k)$ and $\nabla^2 f_0(x^k)$ are the gradient and Hessian respectively of $f_0$ evaluated at $x$. This minimum is directly found by setting the gradient to

zero, yielding

$$p^k = -(\nabla^2 f_0(x^k))^{-1} \nabla f_0(x^k), \tag{2.55}$$

Newton's method converges quadratically close to the solution.

The steepest descent method is a Newton-type method that does not take curvature information into account but instead takes steps $p^k = -\nabla f(x^k)$. It is computationally cheaper but converges only linearly.

In order to enforce convergence of Newton's method from an arbitrary starting point, a suitable globalization strategy is necessary. There are two families of globalization strategies, line-search and trust-region, see [33, 136] for in-depth treatment of the topic.

### Nonlinear Least-Squares (NLS)

In nonlinear least-squares the following special objective consisting of a sum of squared nonlinear functions is assumed

$$f_0(x) = \frac{1}{2}\|r(x)\|_2^2, \tag{2.56}$$

with $r : \mathbf{R}^n \to \mathbf{R}^q$.

The Jacobian of $r$, the $q \times n$ matrix of first partial derivatives, is defined as

$$J(x) = \left[\frac{\partial r_j}{\partial x_i}\right]_{\substack{j=1,\ldots,q \\ i=1,\ldots,n}}$$

With this definition the gradient and Hessian of the objective (2.56) can be written as

$$\begin{array}{ll} \nabla f_0(x) & = J(x)^{\mathrm{T}} r(x), \\ \nabla^2 f_0(x) & = J(x)^{\mathrm{T}} J(x) + \sum_{j=1}^{q} r_j(x) \nabla^2 r_j(x). \end{array} \tag{2.57}$$

Near the solution the first term is typically dominant because either the residuals are small or $\nabla^2 r_j(x)$ are small (near-linearity of the model close the solution). Therefore, the following Hessian approximation is typically used

$$W_k = J_k^{\mathrm{T}} J_k, \tag{2.58}$$

with $J_k := J(x^k)$ This iterative procedure for solving the NLS problem is known as *Gauss-Newton method*. In every iteration a linear least-squares subproblem is solved

$$\begin{aligned} p_{\mathrm{GN}}^k &= \arg\min_p \frac{1}{2}\|r_k + J_k^{\mathrm{T}} p\|_2^2, && (2.59)\\ &= -\left(J_k^{\mathrm{T}} J_k\right)^{-1} J_k^{\mathrm{T}} r_k, && (2.60)\\ &= -J_k^{\dagger} r_k, && (2.61) \end{aligned}$$

where $r_k := r(x^k)$ and $J_k^{\dagger}$ is a pseudo-inverse. Reliable and efficient orthogonal triangularization methods can be applied to these LS subproblems as discussed before, see also [22, 136].

In order to enforce convergence from far-off starting points a suitable globalization strategy is necessary. One particularly popular and effective globalization strategy is the *Levenberg-Marquardt method*. Large steps are penalized by this method by adding a regularization term.

$$\begin{aligned} p_{\mathrm{LM}}^k &= \arg\min_p \frac{1}{2}\|r_k + J_k^{\mathrm{T}} p\|_2^2 + \frac{\alpha_k}{2}\|p\|_2^2, && (2.62)\\ &= -\left(J_k^{\mathrm{T}} J_k + \alpha_k I\right)^{-1} J_k^{\mathrm{T}} r_k, && (2.63) \end{aligned}$$

where $I$ is the identity matrix and $\alpha_k$ is the so-called *damping parameter*. For $\alpha_k$ small the method resembles Gauss-Newton while for $\alpha_k$ large the behavior of steepest descent is obtained.

Now, let us consider constrained nonlinear optimization. The standard form was given before (problem 2.28) and is repeated here for convenience.

$$\begin{array}{lll} \min_x & f_0(x) & \\ \text{s.t.} & f_i(x) \le 0, & i = 1,\ldots,m, \\ & h_i(x) = 0, & i = 1,\ldots,p, \end{array} \qquad (2.64)$$

with $x \in \mathbf{R}^n$. The domain $\mathcal{D} = \bigcap_{i=0}^m \mathrm{dom}(f_i) \cap \bigcap_{i=1}^p \mathrm{dom}(h_i)$ is assumed to be nonempty.

There are two big families of Newton-type algortihms for constrained nonlinear problems, Sequential Quadratic Programming (SQP) type methods and Interior-Point (IP) methods, that differ mainly by the way the complementary slackness condition is handled.

### 2.4.2. Sequential Quadratic Programming (SQP)

One of the most effective methods for nonlinear optimization generates steps by solving quadratic subproblems. As such, this SQP approach, applies in every iteration a

Newton step to the KKT system of the nonlinear programming problem which leads to a fast rate of convergence. It has been shown that SQP solvers require the fewest number of function evaluations to solve NLPs [19, 159]. By linearizing in every iteration all nonlinear functions occurring in the KKT conditions around the current iterate, one arives at the following QP

$$\begin{array}{ll} \min_x & \frac{1}{2}(x-x^k)^{\mathrm{T}}\nabla_x^2\mathcal{L}(x^k,\lambda^k,\nu^k)(x-x^k)+\nabla f_0(x^k)^{\mathrm{T}}x \\ \text{s.t.} & f_i(x^k)+\nabla f_i(x^k)^{\mathrm{T}}(x-x^k)\leq 0, \\ & h_i(x^k)+\nabla h_i(x^k)^{\mathrm{T}}(x-x^k)=0, \end{array} \tag{2.65}$$

where $\nabla f_0(x^k)$ denotes the gradient of the objective evaluated at the current iterate $x^k$ and where $\nabla_x^2\mathcal{L}(x^k,\lambda^k,\nu^k)$ is the Hessian or matrix of second order derivatives of the Lagrangian evaluated at the current iterate of primal and dual variables. If the Hessian is positive semi-definite, this QP is convex.

Sequential Quadratic Programming can be implemented both in a line search or a trust-region framework, see [33, 136]. SQP methods show their strength when applied to problems with significant nonlinearities [136].

### Full quasi-Newton methods

In quasi-Newton methods, also known as variable metric methods, approximations $W_k$ of the Hessian matrix $\nabla_x^2\mathcal{L}(x^k,\lambda^k,\nu^k)$ are maintained. In each iteration a new Hessian approximation $W_{k+1}$ from the previous by a low-rank update. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) update is the most widely used and has proven to be very successful. It is given by

$$W_{k+1} \quad = \quad W_k+\frac{y_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}}s_k}-\frac{W_k s_k s_k^{\mathrm{T}} W_k}{s_k^{\mathrm{T}}W_k s_k}. \tag{2.66}$$

If the process is started with a positive definite approximation $W_0$ then all subsequent Hessian approximations are positive definite. Damped BFGS updates and other variants have been proposed to deal with the problem of non-positive definite true Hessians, see *e.g.* [136].

### Constrained Gauss-Newton (CGN) method

Another particularly successful SQP variant is the Constrained Gauss-Newton (or Generalized Gauss-Newton) method. It uses the Gauss-Newton Hessian approximation (2.58). The constrained Gauss-Newton method has only linear convergence but often with a surprisingly fast contraction rate. The contraction rate is fast when the

residual norm $\|r(x^\star)\|$ is small or the problem functions $f_0, f_i, h_i$ have small second derivatives. It has been developed and extensively investigated by Bock and coworkers, see e.g. [23, 160]. The CGN method is central in Chapter 7 of this dissertation.

### 2.4.3. Interior-point methods

In interior-point methods, in contrast to SQP methods, the solution of the KKT system is attempted by replacing the non-smooth complementary slackness condition $\lambda_i f_i(x) = 0$ by a smooth nonlinear approximation $\lambda_i f_i(x) = \kappa$, with so-called barrier parameter $\kappa$. This amounts to replacing the inequalities with a logarithmic barrier term in the objective weighted with the barrier parameter. Instead of the original problem (2.28) a sequence of problems of the following form

$$\begin{array}{ll} \min_x & f_0(x) - \kappa \sum_{i=1}^{m} \log(-f_i(x)) \\ \text{s.t.} & h_i(x) = 0, \quad i = 1, \ldots, p, \end{array} \tag{2.67}$$

is solved using Newton's method for decreasing $\kappa$. After a limited number of Newton iterations a quite accurate solution of the original Nonlinear Program (NLP) is obtained. By this approach the iterates are forced to remain in the interior of the set described by the inequality constraints, and convergence to the true solution is achieved by gradually reducing the barrier parameter. We refer to the excellent textbook [198] for details. A widely used implementation of nonlinear Interior Point methods is the open source code IPOPT [191].

## 2.5. Algorithm complexity and memory storage

### 2.5.1. Flop count

The complexity of a numerical algorithm is typically expressed by the number of *floating point operations* or *flops* it requires. In this thesis we follow the definition used by Boyd and Vandenberghe [29], Golub and Van Loan [77] and by many other authors of a flop being one arithmetic operation, that is, one addition, subtraction, multiplication or division of two floating-point numbers[1].

Of course the computation time of an algorithm depends on many other factors such as processor speed, choice of compiler, data motion, memory hierarchy and cache boundaries [29, 77], yet the number of flops gives a good indication of the computation time as a function of the problem dimensions. Only the leading terms are typically taken into account.

[1] Some authors define a flop as a multiplication followed by an addition, also known as a DAXPY (double-precision real Alpha times X Plus Y), since this corresponds to one instruction on many processors and is a basic operation in the well-known packages BLAS/LAPACK and LINPACK. Their flop counts are a factor 2 smaller.

#### Vector operations

Consider two vectors $x, y \in \mathbf{R}^n$.

- Addition of $x$ and $y$ requires $n$ flops.
- Multiplication of any of the two vectors with a scalar also requires $n$ flops.
- Computation of the inner product $x^{\mathrm{T}}y$ requires $n$ multiplications and $n-1$ additions, or $2n-1 \approx 2n$ flops.

#### Matrix-vector multiplication

A matrix-vector multiplication $y = Ax$ with $A \in \mathbf{R}^{m\times n}$ requires $m(2n-1) \approx 2mn$ flops.

#### Matrix-matrix multiplication

The product of two matrices $C = AB$ with $A \in \mathbf{R}^{m\times n}$ and $B \in \mathbf{R}^{n\times p}$ requires $mn(2p-1) \approx 2mnp$ flops.

### 2.5.2. Memory allocation

The memory that needs to be allocated for an algorithm is determined by the nature of the data, *i.e.* structure or sparsity, and by the implementation of the algorithm, *i.e.* structure/sparsity exploitation, data overwrite.

In general, storing a vector or matrix requires memory allocation for at least the number of non-zero elements. A dense $m \times n$ matrix requires $mn$ memory entries. If the matrix, however, contains only $N < mn$ non-zero elements it can be stored using only $N$ entries. A triangular or symmetric $n \times n$ matrix requires $\frac{1}{2}n(n+1)$ entries, while a diagonal $n \times n$ matrix - of course - requires $n$ entries. Other structured matrices such as Toeplitz, Hessenberg, Sylvester or banded matrices can be stored efficiently as well.

## 2.6. Brief introduction to the Kalman filter

In his seminal paper [112] Kalman derived his filter using a geometric approach, the orthogonal projection theory. The filter may also be derived using a rigourous probabilistic setup (see *e.g.* [6, 107]) or using maximum likelihood statistics (see *e.g.* [107, 174]), which is the most straightforward and popular way of deriving the filter formulas but has the drawback that several stochastic assumptions are needed, see

the discussion in Section 1.6. The filter equations may be derivation using a purely deterministic least-squares approach, see *e.g.* [71, 190, 196].

In this section derivations are omitted and instead the focus is on the different implementation forms of the Kalman filter and their relations.

### 2.6.1. The covariance Kalman filter (CKF)

For compactness we shall write

$$\begin{aligned} \hat{x}_k &= \hat{x}_{k|k-1}, \quad P_k = P_{k|k-1} \\ \hat{x}_{k+} &= \hat{x}_{k|k}, \quad P_{k+} = P_{k|k} \end{aligned}$$

unless amplification is necessary for clarity. The Kalman filter scheme consists of two distinct stages: a measurement update or correction stage, and a time update or prediction stage.

In the measurement update stage the *a priori* state estimate $\hat{x}_k$ and its associated covariance matrix $P_k$ are combined with measurement information to give an improved estimate and covariance matrix.

**Measurement update**

$$\begin{aligned} R_k^{\mathrm{e}} &= C_k P_k C_k^{\mathrm{T}} + R_k && (2.68) \\ P_{k+} &= P_k - P_k C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} C_k P_k && (2.69) \\ K_k &= P_k C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} && (2.70) \\ \tilde{y}_k &= y_k - h_k - C_k \hat{x}_k && (2.71) \\ \hat{x}_{k+} &= \hat{x}_k + K_k \tilde{y}_k && (2.72) \end{aligned}$$

Here $K_k$ is the *Kalman gain*, $\tilde{y}_k$ is called the *innovation* (*i.e.* , the deviation of predicted output from observed output) and $R_k^{\mathrm{e}}$ is the innovation covariance. The *a posteriori* estimate $\hat{x}_{k+}$ with its associated covariance $P_{k+}$ is now the best linear estimate of $x_k$ using all data up to the current time.

In the time update stage the model is used to propagate the state estimate and covariance matrix one time step forward.

**Time update**

$$\begin{aligned} P_{k+1} &= A_k P_{k+} A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} && (2.73) \\ \hat{x}_{k+1} &= f_k + A_k \hat{x}_{k+} && (2.74) \end{aligned}$$

Provided the initial estimate is a true minimum mean-square-error estimate of the state $x_0$, all subsequent estimates computed by the Kalman recursions are best linear estimates in mean-square sense for the given data. If in addition the disturbances $w$ and $v$ are Gaussian, then the state estimates delivered by the Kalman filter are *optimal* in mean-square sense.

Note that the error covariance and consequently the Kalman gain are independent of the data. Hence, the matrix and vector recursions can be computed in parallel.

Time update and measurement update may be combined yielding

$$\begin{aligned} P_{k+1} &= A_k P_k A_k^{\mathrm{T}} - A_k P_k C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} C_k P_k A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} && (2.75) \\ \bar{K}_k &= A_k P_k C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} && (2.76) \\ \hat{x}_{k+1} &= f_k + A_k \hat{x}_k + \bar{K}_k \tilde{y}_k && (2.77) \end{aligned}$$

Notice that the gain matrix $\bar{K}_k = A_k K_k$. Matrix equation (2.75) yields a recursion for $P_k$ and is termed *Riccati difference equation*.

### 2.6.2. The Kalman filter as a feedback system

Equation (2.77) provides insight into the mechanics of the Kalman filter. It can be viewed as a feedback system where the innovations through a gain matrix $\bar{K}_k$ are injected into the process. This approach was first considered by Luenberger [127] in a deterministic setting. Any gain which stabilizes the error dynamics given by

$$e_{k+1} = (A_k - K_k C_k) e_k \tag{2.78}$$

yields a stable observer[2]. It suffices to choose a gain such that all eigenvalues of $A_k - K_k C_k$ are smaller than one (*e.g.* by *pole placement*). The Kalman filter delivers an optimal gain under the assumed conditions.

### 2.6.3. The time-invariant and steady-state Kalman filter

For linear time-invariant (LTI) systems the Kalman filter recursions as given before can be applied with constant system matrices, however, dedicated algorithms can be derived in order to speed up computations.

[2] When the system is assumed to be disturbance-free (deterministic) a state estimator is called *observer*, when disturbances are considered the term *filter* is preferred.

Consider a recursive estimator for the disturbance-free and offset-free Linear Time-Invariant (LTI) system (2.9) given by the recursion

$$\hat{x}_{k+1} = A\hat{x}_k + K\tilde{y}_k,$$

where $K$ is a gain matrix (not necessarily the Kalman gain) and $\tilde{y}_k = y_k - C\hat{x}_k$ is the innovation vector. Then the estimation error $e_k = x_k - \hat{x}_k$ obeys the following recursion

$$e_{k+1} = (A - KC)e_k.$$

Consequently, the estimation error converges asymptotically to zero if the gain matrix is chosen such that $|\lambda| < 1$ for all $\lambda \in \Lambda(A - KC)$.

**Theorem 2.6.1** (Exponential convergence of the error covariance)**.** *If $\{A, C\}$ is observable and $\{A, W^{\mathrm{T}}\}$ with $Q = W^{\mathrm{T}}W$ is controllable, then the sequence $P_k$ converges exponentially to a limit $P_\infty$ and consequently the Kalman gain converges to a limit $K_\infty$.*

The convergence rate depends on the process and output covariances and on the system dynamics, *i.e.* the eigenvalues of the system matrix $A$. In the limit the (combined) Kalman filter recursion becomes a discrete-time algebraic Riccati equation (DARE):

$$\begin{aligned} P_\infty &= AP_\infty A^{\mathrm{T}} - AP_\infty C^{\mathrm{T}}[R + CP_\infty C^{\mathrm{T}}]^{-1}CP_\infty A^{\mathrm{T}} + GQG^{\mathrm{T}} \\ K_\infty &= AP_\infty C^{\mathrm{T}}[R + CP_\infty C^{\mathrm{T}}]^{-1}. \end{aligned} \tag{2.79}$$

Using the steady-state Kalman filter instead of the (time-invariant) Kalman recursions can yield significant computational savings but involves loss of precision. Whether this trade-off if acceptable depends on the particular application.

The DARE (2.79) can be solved by spectral decomposition of the Hamilton matrix or by iterative methods such as step-doubling or the Newton-Kleinman algorithm [6, 116].

### 2.6.4. The square-root covariance Kalman filter (SRCF)

A well-known problem with the normal Kalman filter recursions is that they can result in a covariance matrix which fails to be symmetric and/or positive definite [6, 71]. This might happen if some measurements are very accurate, which causes numerical ill-conditioning. To cope with this difficulty Potter and Stern [144] introduced the idea of expressing the Kalman filter recursions in terms of a square-root, more precisely a Cholesky factor of the covariance matrix. By propagating such a Cholesky factor, the computed covariance matrix remains symmetric and positive definite at all times. Moreover, the numerical conditioning is generally much better since the condition

number of the Cholesky factor is the square root of the condition number of its corresponding covariance matrix. This means that the precision is effectively doubled. Finally, these square-root recursions are numerically more robust due to the use of orthogonal transformations such as *QR* factorizations [189].

Suppose a square-root or more precisely a (upper) triangular Cholesky factor $S_k$ of the state covariance matrix at time $k$ is given such that $P_k = S_k^{\mathrm{T}} S_k$. Assume that also square roots of the covariance matrices $Q_k$ and $R_k$ are given such that $Q_k = W_k^{\mathrm{T}} W_k$ and $R_k = V_k^{\mathrm{T}} V_k$. Then, the **measurement update**, as described in [6], amounts to finding an orthogonal transformation such that

$$T \begin{pmatrix} V_k & 0 \\ S_k C_k^{\mathrm{T}} & S_k \end{pmatrix} = \begin{pmatrix} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k \\ 0 & S_{k+} \end{pmatrix}, \tag{2.80}$$

where $\tilde{K}_k^{\mathrm{T}} = P_k C_k^{\mathrm{T}} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-1/2}$ and with $P_{k+} = S_{k+}^{\mathrm{T}} S_{k+}$. Here $T$ is any orthogonal matrix making the right hand side triangular. Hence, a *QR* factorization gives us both $T$ and the right hand side.

Subsequently the state estimate is updated as

$$\hat{x}_{k+} = \hat{x}_k - \tilde{K}_k^{\mathrm{T}} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-\mathrm{T}/2} \tilde{y}_k \tag{2.81}$$

Another form of the measurement update suggests computing the following *QR* factorization

$$\begin{pmatrix} I \\ V_k^{-\mathrm{T}} C_k S_k^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} Q & \tilde{Q} \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{2.82}$$

and subsequently updating

$$S_{k+} = R^{-\mathrm{T}} S_k, \tag{2.83}$$

$$\hat{x}_{k+} = \hat{x}_k + S_{k+}^{\mathrm{T}} Q^{\mathrm{T}} \begin{pmatrix} 0 \\ V_k^{-\mathrm{T}} \tilde{y}_k \end{pmatrix}. \tag{2.84}$$

This form requires invertibility of the measurement weighting matrix $R_k$ (hence of $V_k$). The formulation can easily be verified as follows

$$P_{k+} = \left(P_k^{-1} + C_k^{\mathrm{T}} R_k^{-1} C_k\right)^{-1} = \left[S_k^{-1} S_k^{-\mathrm{T}} + C_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} C_k\right]^{-1}, \tag{2.85}$$

$$= \left[S_k^{-1} \left(I + S_k C_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} C_k S_k^{\mathrm{T}}\right) S_k^{-\mathrm{T}}\right]^{-1}, \tag{2.86}$$

$$= \left[S_k^{-1} \left(R^{\mathrm{T}} Q^{\mathrm{T}} Q R\right) S_k^{-\mathrm{T}}\right]^{-1}, \tag{2.87}$$

$$= S_k^{\mathrm{T}} R^{-1} R^{-\mathrm{T}} S_k, \tag{2.88}$$

$$= S_{k+}^{\mathrm{T}} S_{k+}. \tag{2.89}$$

And for the state update

$$\begin{aligned}\hat{x}_{k+} &= \hat{x}_k + P_k C_k^{\mathrm{T}} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-1} \tilde{y}_k, && (2.90)\\ &= \hat{x}_k + S_k^{\mathrm{T}} R^{-1} R^{-\mathrm{T}} S_k C_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} \tilde{y}_k, && (2.91)\\ &= \hat{x}_k + S_k^{\mathrm{T}} R^{-1} Q^{\mathrm{T}} \begin{pmatrix} 0 \\ V_k^{-\mathrm{T}} \tilde{y}_k \end{pmatrix}, && (2.92)\end{aligned}$$

where the second step comes from the Matrix Inversion Lemma and where for the last step we need to prove that

$$R^{-\mathrm{T}} S_k C_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} = Q^{\mathrm{T}} \begin{pmatrix} 0 \\ V_k^{-\mathrm{T}} \end{pmatrix}. \quad (2.93)$$

Left multiplying both sides with $R^{\mathrm{T}}$ gives

$$S_k C_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} = R^{\mathrm{T}} Q^{\mathrm{T}} \begin{pmatrix} 0 \\ V_k^{-\mathrm{T}} \end{pmatrix}. \quad (2.94)$$

and using the definitions from the *QR* factorization (2.82) yields

$$S_k C_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} = \begin{pmatrix} I & S_k C_k^{\mathrm{T}} V_k^{-1} \end{pmatrix} \begin{pmatrix} 0 \\ V_k^{-\mathrm{T}} \end{pmatrix}, \quad (2.95)$$

which concludes the proof.

The **time update** is found by computing the following *QR* factorization

$$\begin{pmatrix} S_{k+} A_k^{\mathrm{T}} \\ W_k G_k^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} \hat{Q} & \bar{Q} \end{pmatrix} \begin{pmatrix} S_{k+1} \\ 0 \end{pmatrix}, \quad (2.96)$$

and the state update remains as in the CKF given by

$$\hat{x}_{k+1} = f_k + A_k \hat{x}_{k+}. \quad (2.97)$$

Note that the following modified combined update equation (compare with (2.75))

$$P_{k+1} = (A_k - \bar{K}_k C_k) P_k (A_k - \bar{K}_k C_k)^{\mathrm{T}} + \bar{K}_k R_k \bar{K}_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} \quad (2.98)$$

tends to promote nonnegative definite covariance matrices. However, if $P_k$ fails to be nonnegative for some reason (*i.e.* numerical approximation errors), all subsequent covariance matrices will fail to be nonnegative as well. In addition, this modification lacks the good numerical conditioning and robustness of the SRCF implementation.

### 2.6.5. The information filter (IF)

Instead of propagating the covariance matrix and the current state estimate the information filter propagates the inverse covariance matrix or *information matrix* and the *information vector*. The information filter is especially useful when the initial state is completely unknown (*i.e.* $P_0 = \infty$), since the covariance Kalman filter would fail in this case. On the other hand, the IF fails when covariance matrices become singular. If the output dimension is large compared with the state dimension then the information filter is more efficient than the covariance filter, which explains its popularity as an observer in large sensor networks. A drawback of the information filter is the loss of physical interpretation of state vector components and covariances [83].

The following information vectors are introduced

$$\hat{a}_k = P_k^{-1}\hat{x}_k, \quad \hat{a}_{k+} = P_{k+}^{-1}\hat{x}_{k+}.$$

Application of the Matrix Inversion Lemma to (2.69) and (2.72) yields the following expressions for the measurement update

$$P_{k+}^{-1} = P_k^{-1} + C_k^{\mathrm{T}} R_k^{-1} C_k \tag{2.99}$$
$$\hat{a}_{k+} = \hat{a}_k + C_k^{\mathrm{T}} R_k^{-1}(y_k - h_k) \tag{2.100}$$

Note that the measurement update is simpler in the information filter. The time update equations on the other hand are slightly more complicated. Assume $A_k$ is invertible and define

$$M_k = A_k^{-\mathrm{T}} P_{k+}^{-1} A_k^{-1}.$$

Then applying the Matrix Inversion Lemma to (2.73) yields

$$P_{k+1}^{-1} = \left(M_k^{-1} + G_k Q_k G_k^{\mathrm{T}}\right)^{-1}, \tag{2.101}$$
$$= \left(I - N_k G_k^{\mathrm{T}}\right) M_k, \tag{2.102}$$

with $N_k = M_k G_k \left(G_k^{\mathrm{T}} M_k G_k + Q_k^{-1}\right)^{-1}$. And furthermore

$$\hat{a}_{k+1} = \left(I - N_k G_k^{\mathrm{T}}\right) M_k \left(f_k + A_k \hat{x}_{k+}\right) \tag{2.103}$$
$$= \left(I - N_k G_k^{\mathrm{T}}\right) \left(M_k f_k + A_k^{-\mathrm{T}} \hat{a}_{k+}\right) \tag{2.104}$$

Note that compared to the covariance Kalman filter this formulation requires inverses of $A_k$ and $Q_k$ which involves extra computations and causes numerical problems in case these matrices are (near) singular.

If $G_k = I$ for all $k$, the expressions can be simplified and the requirement of non-singular $A_k$ can be dropped, since in this case applying the Matrix Inversion Lemma to (2.73) with $G_k = I$ yields

$$P_{k+1}^{-1} = Q_k^{-1} - Q_k^{-1} A_k \left( A_k^{\mathrm{T}} Q_k^{-1} A_k + P_{k+}^{-1} \right)^{-1} A_k^{\mathrm{T}} Q_k^{-1}, \tag{2.105}$$

$$\hat{a}_{k+1} = P_{k+1}^{-1} \left( f_k + A_k P_{k+} \hat{x}_{k+} \right). \tag{2.106}$$

The problem with this formulation, apart from the requirement $G_k = I$, is that the information vector time update can only be formulated in terms of information vectors if invertibility of $P_{k+}^{-1}$ is assumed which is in contradiction with one of the main reasons of existence of the information filter, the possibility of infinite $P_{k+}$.

Another possibility which allows $G_k \neq I$ and avoids the assumption of invertible $A_k$ was given by Mutambara [135]. He proposes to perform the inversion of $P_{k+1}$ explicitly instead of applying the Matrix Inversion Lemma. However, in this case invertibility of both $P_{k+}^{-1}$ and $P_{k+1}^{-1}$ is required which undermines the spirit of information filtering. The time update in this formulation is given by

$$P_{k+1}^{-1} = \left( A_k P_{k+} A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} \right)^{-1} \tag{2.107}$$

$$\hat{a}_{k+1} = P_{k+1}^{-1} ( f_k + A_k P_{k+} \hat{a}_{k+} ). \tag{2.108}$$

Comparing the information filter to the covariance Kalman filter, it is seen that the measurement update is simpler whereas the time update is more complex. Furthermore, the information filter needs inverses of $G_k^{\mathrm{T}} M_k G_k + Q_k^{-1}$ while the covariance Kalman filter needs inverses of $C_k P_k C_k^{\mathrm{T}} + R_k$. Hence if $n_w \ll n_y$ the IF will be more efficient, if the reverse is true the CKF is favorable [6].

Interestingly there exists a *duality* between the CKF and the IF [6, 71]. Table 2.1 contains the duality relations to convert between both formulations.

| CKF time update | IF meas. update |
|---|---|
| $P_{k+1}$ | $P_{k+}^{-1}$ |
| $A_k$ | $C_k^{\mathrm{T}}$ |
| $P_{k+}$ | $R_k^{-1}$ |
| $G_k Q_k G_k^{\mathrm{T}}$ | $P_k^{-1}$ |
| **CKF meas. update** | **IF time update** |
| $P_{k+}$ | $P_{k+1}^{-1}$ |
| $P_k$ | $M_k$ |
| $C_k^{\mathrm{T}}$ | $G_k$ |
| $R_k$ | $Q_k^{-1}$ |

**Table 2.1.** Duality relations between the covariance Kalman filter (CKF) and the information filter (IF).

### The square-root information filter (SRIF)

In analogy to the covariance Kalman filter, a square-root form of the information filter can be derived [6]. Let us define the following (square-root) innovation vectors

$$\hat{b}_k = S_k^{-\mathrm{T}}\hat{x}_k, \quad \hat{b}_{k+} = S_{k+}^{-\mathrm{T}}\hat{x}_{k+}.$$

where $S_k$ denotes a square-root Cholesky factor of $P_k$ as before. Then the measurement update is given by the following $QR$ factorization

$$\begin{pmatrix} S_k^{-\mathrm{T}} \\ V_k^{-\mathrm{T}} C_k \end{pmatrix} = (\hat{Q} \quad \bar{Q}) \begin{pmatrix} S_{k+}^{-\mathrm{T}} \\ 0 \end{pmatrix}, \tag{2.109}$$

and the (square-root) information vector is updated as follows

$$\begin{pmatrix} \hat{b}_{k+} \\ \star \end{pmatrix} = \hat{Q}^{\mathrm{T}} \begin{pmatrix} \hat{b}_k \\ V_k^{-\mathrm{T}}(y_k - h_k) \end{pmatrix}, \tag{2.110}$$

where $\star$ denotes entries which are not important for our discussion. The time update is defined (analogously to (2.80)) by the following orthogonal transformation

$$\bar{T} \begin{pmatrix} W_k^{-\mathrm{T}} & 0 \\ S_{k+}^{-\mathrm{T}} A_k^{-1} G_k & S_{k+}^{-\mathrm{T}} A_k^{-1} \end{pmatrix} = \begin{pmatrix} \left(Q_k^{-1} + G_k^{\mathrm{T}} M_k G_k\right)^{1/2} & \tilde{B}_k^{\mathrm{T}} \\ 0 & S_{k+1}^{-\mathrm{T}} \end{pmatrix}, \tag{2.111}$$

$$\begin{pmatrix} \star \\ \hat{b}_{k+1} \end{pmatrix} = \bar{T} \begin{pmatrix} 0 \\ \hat{b}_{k+} + S_{k+}^{-1} A_k^{-1} f_k \end{pmatrix}. \tag{2.112}$$

with $\tilde{B}_k = M_k G_k \left(Q_k^{-1} + G_k^{\mathrm{T}} M_k G_k\right)^{-\mathrm{T}/2}$ and where $\bar{T}$ is any orthogonal matrix making the right hand side triangular.

Table 2.2 gives an overview of the different implementation forms of the Kalman filter.

<table>
<tr><th>Form</th><th>Normal</th><th>Square-root</th></tr>
<tr><td>Covariance</td><td>

**Measurement update**

$$\begin{aligned} P_{k+} &= P_k - P_k C_k^{\mathrm{T}} \left(C_k P_k C_k^{\mathrm{T}} + R_k\right)^{-1} C_k P_k \\ K_k &= P_k C_k^{\mathrm{T}} \left(C_k P_k C_k^{\mathrm{T}} + R_k\right)^{-1} \\ \tilde{y}_k &= y_k - h_k - C_k \hat{x}_k \\ \hat{x}_{k+} &= \hat{x}_k + K_k \tilde{y}_k \end{aligned}$$

**Time update**

$$\begin{aligned} P_{k+1} &= A_k P_{k+} A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} \\ \hat{x}_{k+1} &= f_k + A_k \hat{x}_{k+} \end{aligned}$$

</td><td>

**Measurement update**

$$\begin{aligned} T \begin{pmatrix} V_k & 0 \\ S_k C_k^{\mathrm{T}} & S_k \end{pmatrix} &= \begin{pmatrix} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k \\ 0 & S_{k+} \end{pmatrix} \\ \hat{x}_{k+} &= \hat{x}_k + \tilde{K}_k \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-\mathrm{T}/2} \tilde{y}_k \end{aligned}$$

**Time update**

$$\begin{aligned} \begin{pmatrix} S_{k+} A_k^{\mathrm{T}} \\ W_k G_k^{\mathrm{T}} \end{pmatrix} &= \begin{pmatrix} \hat{Q} & \bar{Q} \end{pmatrix} \begin{pmatrix} S_{k+1} \\ 0 \end{pmatrix} \\ \hat{x}_{k+1} &= f_k + A_k \hat{x}_{k+} \end{aligned}$$

</td></tr>
<tr><td>Information</td><td>

**Measurement update**

$$\begin{aligned} P_{k+}^{-1} &= P_k^{-1} + C_k^{\mathrm{T}} R_k^{-1} C_k \\ \hat{a}_{k+} &= \hat{a}_k + C_k^{\mathrm{T}} R_k^{-1} (y_k - h_k) \end{aligned}$$

**Time update**

$$\begin{aligned} P_{k+1}^{-1} &= \left(I - N_k G_k^{\mathrm{T}}\right) M_k \\ \hat{a}_{k+1} &= \left(I - N_k G_k^{\mathrm{T}}\right) \left(M_k f_k + A_k^{-\mathrm{T}} \hat{a}_{k+}\right) \end{aligned}$$

with

$$\begin{aligned} M_k &= A_k^{-\mathrm{T}} P_{k+}^{-1} A_k^{-1} \\ N_k &= M_k G_k \left(G_k^{\mathrm{T}} M_k G_k + Q_k^{-1}\right)^{-1} \end{aligned}$$

</td><td>

**Measurement update**

$$\begin{aligned} \begin{pmatrix} S_k^{-\mathrm{T}} \\ V_k^{-\mathrm{T}} C_k \end{pmatrix} &= \begin{pmatrix} \hat{Q} & \bar{Q} \end{pmatrix} \begin{pmatrix} S_{k+}^{-\mathrm{T}} \\ 0 \end{pmatrix} \\ \begin{pmatrix} \hat{b}_{k+} \\ \star \end{pmatrix} &= \hat{Q}^{\mathrm{T}} \begin{pmatrix} \hat{b}_k \\ V_k^{-\mathrm{T}} (y_k - h_k) \end{pmatrix} \end{aligned}$$

**Time update**

$$\begin{aligned} \bar{T} \begin{pmatrix} W_k & 0 \\ S_{k+}^{-\mathrm{T}} A_k^{-1} G_k & S_{k+}^{-\mathrm{T}} A_k^{-1} \end{pmatrix} &= \begin{pmatrix} \left(Q_k + G_k^{\mathrm{T}} A_k G_k\right)^{\mathrm{T}/2} & \tilde{B}_k^{\mathrm{T}} \\ 0 & S_{k+1}^{-\mathrm{T}} \end{pmatrix} \\ \begin{pmatrix} * \\ \hat{b}_{k+1} \end{pmatrix} &= \bar{T} \begin{pmatrix} 0 \\ \hat{b}_{k+} + S_{k+}^{-1} A_k^{-1} f_k \end{pmatrix} \end{aligned}$$

with $\tilde{B}_k = M_k^{\mathrm{T}} G_k \left(Q_k + G_k^{\mathrm{T}} A_k G_k\right)^{-\mathrm{T}/2}$

</td></tr>
</table>

**Table 2.2.** Different implementation forms of the Kalman filter.

## 2.7. Brief introduction to Kalman smoothing

In the setting of filtering, the Kalman filter provides the best estimate based on past data. If *future* data are also available, they can be used to further improve the estimate. In the following discussion, $k$ denotes the current time instant.

In one scenario we may be interested in obtaining an estimate of the state at a fixed time $j < k$ given measurements up to time $k$. As more and more measurements are collected, the estimate $\hat{x}_j$ can be systematically refined. Consider for example a satellite taking pictures at a fixed rate [164]. As the satellite continues to orbit, additional range measurements can be used to improve the picture taken at time $j$. This situation is called *fixed-point smoothing*.

In another situation it might be useful to estimate the state at time $k-N$ where $N$ represents a fixed lag and where the index $k$ is continually changing. In other words, at every time step a state is estimated using a fixed number of $N$ *future* measurements. In our satellite example this situation occurs when $N$ time steps are needed before the picture is transmitted and processed. This case is known as *fixed-lag smoothing*.

Finally, in *fixed-interval smoothing* state estimates in a some interval, *e.g.* $[0,\ldots,N]$ are computed based on all measurements in this interval. This situation occurs if a sequence of satellite pictures is available for post-processing and the goal is to obtain a time history of optimal estimates given all collected data. Fixed-interval smoothing has significant resemblance with MHE as discussed in the following sections. It is possible to employ a single smoothing scheme based on fixed-interval smoothing to solve all three classes of smoothing problems.

The improvement due to smoothing is monotone increasing as more measurements become available. The maximum improvement possible is governed by the system dynamics and the signal-to-noise ratio and can vary from zero to one hundred percent.

Traditionally Kalman smoothing has been primarily used for offline estimation (fixed-interval smoothing) or for online estimation problems which can tolerate some delay (fixed-lag smoothing).

### 2.7.1. Fixed-point smoothing

The fixed-point smoother equations are summarized below. **Initialization**

$$\hat{x}_{j|j} = \hat{x}_j \tag{2.113}$$
$$P_{j|j} = P_j \tag{2.114}$$
$$\Pi_{j|j} = P_j \tag{2.115}$$
$$\tag{2.116}$$

**For** $k = j, j+1, \ldots$

$$
\begin{aligned}
R_k^{\mathrm{e}} &= C_k P_k C_k^{\mathrm{T}} + R_k && (2.117)\\
P_{k+} &= P_k - P_k C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} C_k P_k && (2.118)\\
K_k &= P_k C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} && (2.119)\\
\tilde{y}_k &= y_k - h_k - C_k \hat{x}_k && (2.120)\\
\hat{x}_{k+} &= \hat{x}_k + K_k \tilde{y}_k && (2.121)\\
P_{k+1} &= A_k P_{k+} A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}} && (2.122)\\
\hat{x}_{k+1} &= f_k + A_k \hat{x}_{k+} && (2.123)\\
K_{j|k} &= \Pi_{j|k} C_k^{\mathrm{T}} [R_k^{\mathrm{e}}]^{-1} && (2.124)\\
\hat{x}_{j|k+1} &= \hat{x}_{j|k} + K_{j|k} \tilde{y}_k && (2.125)\\
\Pi_{j|k+1} &= \Pi_{j|k} \left( A_k^{\mathrm{T}} - C_k^{\mathrm{T}} K_{j|k} \right)^{\mathrm{T}} && (2.126)\\
P_{j|k+1} &= P_{j|k} - \Pi_{j|k} C_k^{\mathrm{T}} K_{j|k}^{\mathrm{T}}. && (2.127)
\end{aligned}
$$

### 2.7.2. Improvement due to smoothing

From equations (2.124) and (2.127) it can be seen that

$$
\begin{aligned}
P_{j|k} &= P_{j|k-1} - \Pi_{j|k} C_k^{\mathrm{T}} \left( C_k P_k C_k^{\mathrm{T}} + R_k \right)^{-1} C_k \Pi_{j|k} && (2.128)\\
&= P_{j|j-1} - \sum_{l=j}^{k} \Pi_{j|l} C_l^{\mathrm{T}} \left( C_l P_l C_l^{\mathrm{T}} + R_l \right)^{-1} C_l \Pi_{j|l} && (2.129)
\end{aligned}
$$

Or, equivalently, the improvement due to smoothing can be written as

$$
P_{j|j-1} - P_{j|k} = \sum_{l=j}^{k} \Pi_{j|l} C_l^{\mathrm{T}} \left( C_k P_k C_k^{\mathrm{T}} + R_k \right)^{-1} C_l \Pi_{j|l} \qquad (2.130)
$$

Usually, the relative improvement is expressed as a percentage as follows

$$
\frac{\mathrm{tr}\left[ P_{j|j-1} - P_{j|k} \right]}{\mathrm{tr}\left[ P_{j|j-1} \right]} \times 100\%
$$

Now consider the time-invariant case and suppose the filter has reached steady-state, *i.e.* $\lim_{k\to\infty} P_k = \bar{P}$. Then, from the initialization, eq. (2.115) and recursion eq. (2.126),

$\Pi_{j|k}$ can be written as

$$\Pi_{j|k} = \bar{P}(A-KC)^{k-j+1}, \tag{2.131}$$
$$= \bar{P}\Phi^{k-j+1}. \tag{2.132}$$

Plugging this into expression (2.130) evaluated at steady-state yields the maximum possible improvement due to smoothing:

$$\bar{P} - P_{j|\infty} = \bar{P}\left[\sum_{l=j}^{k}\left(\Phi^{\mathrm{T}}\right)^{l-j} C_l^{\mathrm{T}}\left(C_k P_k C_k^{\mathrm{T}} + R_k\right)^{-1} C_l \Phi^{l-j}\right]\bar{P} \tag{2.133}$$

From this it can be seen that the improvement due to smoothing increases proportianally to the signal-to-noise ratio, *i.e.* $R \to 0$. For estimation problems with a high signal-to-noise ratio the improvement of smoothing can be significant, even close to 100 percent [6, 164]. The improvement increases monotonically with a rate governed by the eigenvalues of the filter dynamics $\Phi = A - KC$. For the time-invariant case, a rule of thumb states that practically all possible improvement due to smoothing is obtained after two or three times the dominant time constant (spectral radius). For the time-varying case, the eigenvalues of the filter dynamics govern the rate of change of improvement with lag, in this case no general conclusion nor quantification can be made since the dynamics are changing.

### 2.7.3. Fixed-lag smoothing

Let us define $\hat{x}_{k,l}$ as the state estimate $\hat{x}_{k-l+1}$ propagated $l$ time steps forward with an identity transition matrix and no disturbance. Or written mathematically

$$\hat{x}_{k,1} = \hat{x}_k, \quad \hat{x}_{k,2} = \hat{x}_{k-1}, \quad \ldots \quad \hat{x}_{k,l} = \hat{x}_{k-l+1}.$$

With this, the augmented system can be defined as

$$\begin{bmatrix} x_{k+1} \\ x_{k+1,1} \\ \vdots \\ x_{k+1,N+1} \end{bmatrix} = \begin{bmatrix} A_k & 0 & \cdots & 0 \\ I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & I & 0 \end{bmatrix} \begin{bmatrix} x_k \\ x_{k,1} \\ \vdots \\ x_{k,N+1} \end{bmatrix} + \begin{bmatrix} I \\ 0 \\ \vdots \\ 0 \end{bmatrix} f_k + \begin{bmatrix} G_k \\ 0 \\ \vdots \\ 0 \end{bmatrix} w_k, \tag{2.134}$$

$$y_k = h_k + \begin{bmatrix} C_k & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} x_k \\ x_{k,1} \\ \vdots \\ x_{k,N+1} \end{bmatrix} + v_k. \tag{2.135}$$

Or, written in terms of Kalman gains

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hat{x}_{k+1,1} \\ \vdots \\ \hat{x}_{k+1,N+1} \end{bmatrix} = \begin{bmatrix} \hat{x}_{k|k-1} \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cdots & 0 \\ I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & I & 0 \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \hat{x}_{k,1} \\ \vdots \\ \hat{x}_{k,N+1} \end{bmatrix} + \begin{bmatrix} K_k \\ K_{k,1} \\ \vdots \\ K_{k,N+1} \end{bmatrix} \tilde{y}_k,$$

where $\hat{x}_{k+1,1}$ is obtained from a normal Kalman filter recursion. By applying the Kalman filter equations to this augmented model, smoothed state estimates for all lags $l = 1,\ldots,N$ are obtained. In fixed-lag smoothing, however, one is only interested in the state with lag $N+1$. For this reason, the fixed-lag smoother for time-varying models and significant lags is computationally involved compared to the Kalman filter. For the time-invariant steady-state case, however, the fixed-lag smoother is very practical and reduces to

$$\hat{x}_{k-l|k} = \hat{x}_{k-l|k-1} + K_{k,l+1}\tilde{y}_k, \tag{2.136}$$

$$K_{k,l+1} = \bar{P}(A-KC)^{k-j+1} C_k^{\mathrm{T}} \left(C_k P_k C_k^{\mathrm{T}} + R_k\right)^{-1}. \tag{2.137}$$

## 2.7.4. Fixed-interval smoothing

### Forward-backward smoothing

The forward-backward algorithm, derived by Fraser and Potter [62], computes the smoothed estimates as a linear combination of two optimal filters.

In a **first step** the standard Kalman filter is applied. In a **second step**, a backward filter is applied. Since forward and backward estimates must be independent and no further information is available, the backward filter needs to be initialized as $P_{b,N} = \infty$. In order to make this recursion computationally feasible, this initialization implies the backward filter needs to be run in information form. The backward information filter is initialized as follows

$$Y_N = P_N^{-1} = 0 \tag{2.138}$$

$$\hat{a}_N = P_N^{-1}\hat{x}_N = 0, \tag{2.139}$$

and subsequently, in every time step, the following recursions are performed

$$Y_{k+} = Y_k + C_k^{\mathrm{T}} R_k^{-1} C_k \tag{2.140}$$

$$\hat{a}_{k+} = \hat{a}_k + C_k^{\mathrm{T}} R_k^{-1}(y_k - h_k) \tag{2.141}$$

$$K_{b,k-1} = Y_{k+}\left(Y_{k+} + Q_{k-1}^{-1}\right)^{-1} \tag{2.142}$$

$$Y_{k-1} = A_{k-1}^{\mathrm{T}}(I - K_{b,k-1})Y_{k+}A_{k-1} \tag{2.143}$$

$$\hat{a}_{k-1} = A_{k-1}^{\mathrm{T}}(I - K_{b,k-1})(\hat{a}_{k+} - Y_{k+}f_k) \tag{2.144}$$

In a **third step**, the smoothed estimates and covariances are computed from the forward and backward filtered estimates as follows

$$\begin{aligned} K_k^s &= P_{k+}Y_k\left(I+P_{k+}Y_k\right)^{-1} && (2.145)\\ P_k^s &= \left(I-K_k^s\right)P_{k+} && (2.146)\\ \hat{x}_k^s &= \left(I-K_k^s\right)\hat{x}_{k+}+P_k^s\hat{a}_k && (2.147) \end{aligned}$$

The optimal weighting between forward and backward estimates is also known as *Millman's theorem* [35]. The derivation of the forward-backward smoother is straightforward but the algorithm is computationally expensive. The following algorithm, best known as Rauch-Tung-Striebel (RTS) smoothing, is computationally more efficient.

### Two-pass smoothing or Rauch-Tung-Striebel (RTS) smoothing

The Rauch-Tung-Striebel algorithm [151] employs the regular Kalman filter in a forward pass whereafter the backward pass applies a correction using only the data provided by the forward pass.

Analogous to the forward-backward smoother, the **first step** of RTS smoothing involves a forward run of the standard Kalman filter.

In a **second step**, the smoothed estimates are computed as a correction to the filtered estimates obtained by the forward run.

$$\begin{aligned} P_N^s &= P_{N+} && (2.148)\\ \hat{x}_N^s &= \hat{x}_{N+}, && (2.149) \end{aligned}$$

and subsequently, in every time step, the following recursions are performed

$$\begin{aligned} K_k^s &= P_{k+}A_k^{\mathrm{T}}P_{k+1}^{-1} && (2.150)\\ P_k^s &= P_{k+}-K_k^s\left(P_{k+1}-P_{k+1}^s\right)\left(K_k^s\right)^{\mathrm{T}} && (2.151)\\ \hat{x}_k^s &= \hat{x}_{k+}+K_k^s\left(\hat{x}_{k+1}^s-\hat{x}_{k+1}\right). && (2.152) \end{aligned}$$

We note that $K_k^s = P_k(A_k - K_kC_k)^{\mathrm{T}}P_{k+1}^{-1}$.

In order to prove these recursions, observe from eq. (2.125) that

$$\begin{aligned} \hat{x}_{k|N} &= \hat{x}_{k|N-1}+K_{k|N}\tilde{y}_k && (2.153)\\ &= \hat{x}_{k|k}+\sum_{l=k+1}^{N}K_{k|l}\tilde{y}_l. && (2.154) \end{aligned}$$

and similarly

$$\hat{x}_{k+1|N} = \hat{x}_{k+1|k} + \sum_{l=k+1}^{N} K_{k+1|l}\tilde{y}_l. \tag{2.155}$$

Furthermore, it can be shown that $K_{k|l} = K_k^s K_{k+1|l}$. Using this information, allows us to write

$$\hat{x}_k^s = \hat{x}_{k|N} = \hat{x}_{k|k} + K_k^s \sum_{l=k+1}^{N} K_{k+1|l}\tilde{y}_l, \tag{2.156}$$

$$= \hat{x}_{k|k} + K_k^s \left(\hat{x}_{k+1|N} - \hat{x}_{k+1|k}\right), \tag{2.157}$$

q.e.d.

The equation for $P_k^s = P_{k|N}$ can be obtained after a similar calculation.

Note that the smoothed covariances are not needed to compute the smoothed estimates. Also note that invertibility of the covariances computed by the forward pass is required.

Similarly to the filtering problem, steady-state smoothing algorithms can be derived for LTI systems to save computational effort. An early survey of smoothing algorithms is given in [132].

CHAPTER 3

# Efficient methods for unconstrained MHE

*In this chapter efficient methods for the solution of MHE problems are discussed for the case when no inequalities are active. Normal and square-root Riccati based solution strategies are derived from decompositions of the KKT system. These solution methods provide insights into the mechanics of MHE and yield an update formula for the arrival cost. Equivalence of unconstrained MHE and Kalman filtering/smoothing is shown and the analogy with MPC is discussed. Structured QR factorization methods are proposed to speed up the computations in the square-root method. The performance and robustness of these solution methods are compared for optimized C implementations and illustrated by virtue of numerical examples.*

## 3.1. Introduction

In this chapter efficient solution methods for unconstrained MHE are discussed. It is well known that a rearrangement of the variables reveals the typical block diagonal structure of the Karush-Kuhn-Tucker (KKT) system of optimal control problems. This special structure can be exploited using either Riccati recursions or sparse decomposition methods. As shown in this chapter, Riccati methods solve an optimal control problem by: (1) factorizing the KKT system using a Riccati recursion, and (2) applying the factors through a forward and a backward vector solve. Riccati methods for optimal control have been investigated for example by Glad and Jonson[75], Rao

et al [150] and Vandenberghe et al [188]. Wang et al [192] amongst others proposed a sparse Cholesky decomposition. Normal Riccati based methods for MHE problems have been proposed before, *e.g.* by Tenny et al [173] and Jorgensen et al [108].

The most important difference between both is that the MHE problem has a free initial state vector $x_0$. These extra degrees of freedom allow us to fit an observed output sequence according to a specified objective. It must be noted that the addition of an initial condition typically increases the numerical conditioning as the extra degrees of freedom may result in an infinite number of solutions to the estimation problem.

As we discussed in Section 1.8 and show in Section 3.4 there is a strong analogy between MPC and MHE. Both are similar in nature, yielding similar KKT systems. The MPC algorithms can be applied with some modifications to MHE problems (and vice versa); more specifically using conversion tables for the Riccati and vector recursions, adaptation of the right hand side (*e.g.* to include the observations) and a modification for the initial state. However, while the Riccati approach for MPC is merely an efficient way of solving the KKT system by exploiting the Gauss-Markov structure, it is much more valuable in the MHE context. In MPC, the terminal weight if calculted by LQR is typically constant because the infinite trajectory beyond the horizon remains infinite. In contrast, in MHE the arrival cost is updated in every time step by one combined Kalman filter update step which is exactly the first step in the solution method if a Riccati approach is used. Furthermore, the quality of the final state estimate, which is the variable of interest in MHE, is important and the Riccati recursion computes a covariance matrix representing second order information for the problem discarding constraints, which is a usual confidence measure in state or parameter estimation and is typically obtained in standard methods by a local first order (statistical) analysis around the estimate. This solution method and the natural outcomes are depicted schematically in Figure 3.1. Furthermore, note that the weighting matrices in the MHE problem are inverses and therefore it can be that the problem may not even be formulated if a singular initial covariance (or infinite initial weighting matrix) is supplied while a (unique) solution is easily found using a Kalman filter recursion. On the other hand, the initial covariance can be chosen infinite if no initial guess is available and in this case information filter recursions can be applied. The applicability of the numerical algorithms is improved by designing them with the particularities of the problem in mind. In this chapter we derive normal and square-root Riccati methods for unconstrained MHE from respectively $LU$ and $LDL^{\mathrm{T}}$ decomposition of the KKT system. The equations may also be derived using the equivalence between weighted least squares (MHE) and the Kalman filter or fixed interval smoother, as presented in several text books, *e.g.* [111, 190].

The complexity of the Riccati based solutions is dominated by the cost of factorization and is $O(N(n_x + n_w)^3)$. The same complexity is achieved with a sparse $LDL^{\mathrm{T}}$ factorization method applied to the KKT system, but in this case the arrival cost up-

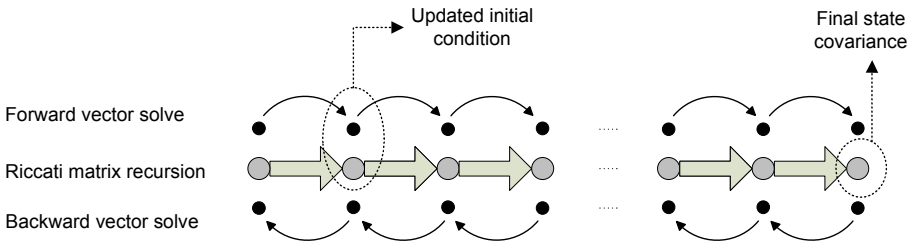



**Figure 3.1.** Schematic representation of the MHE solution strategy using Riccati and vector recursions. The updated initial condition (arrival cost update) and the final state covariance are obtained as a by-product of the solution strategy.

date and final state covariance are not obtained from the solution process. Hence, the cost of the Riccati based solution methods grows only linearly with the horizon length $N$ and, as with any Kalman filter strategy the scales cubically with the system dimensions. In contrast, a dense solver applied to the full KKT system has complexity $O(N^3(n_x+n_w)^3)$.

Another approach for solving optimal control problems is known as *condensing* and was first proposed by Bock and Plitt [25]. For MHE this approach arises from the observation that the state equation $x_{k+1} = f_k + A_k x_k + G_k w_k$ allows reconstruction of the complete state sequence $\{x_0,\dots,x_N\}$ from the knowledge of $x_0$ and $\{w_0,\dots,w_{N-1}\}$ only. Hence, the states $x_1,\dots,x_N$ can be eliminated yielding a reduced optimization problem. The cost of this condensing approach is again dominated by the cost of factorizing the reduced system and is roughly $O(N^3{n_w}^3)$. Alternatively, if $n_x \approx n_w$ which is typical in MHE problems, one could eleminate the disturbances $\{w_0,\dots,w_{N-1}\}$ and obtain a reduced system with a cost of factorization of $O(N^3{n_x}^3)$. This approach can be categorized as a reduced Hessian QP method, where the effort of finding an orthogonal basis for the constraint matrix is avoided by variable elimination.

For control problems, the control dimension is typically much smaller than the state dimension. Therefore, for MPC, condensing is often favorable even though it has some extra overhead due to the computation of the reduced system (block elimination) which is often not mentioned. For MHE problems, where the dimension of the *controls* or process disturbances is typically equal or comparable to the state dimension $n_w \approx n_x$, the Riccati approach is practically always favorable for reasonable horizons. Comparing the order estimates, hence neglecting the overhead cost and smaller terms, of both approaches we see that for $n_w = n_x$ the Riccati approach is already favorable for $N > 2$ for any system dimension. For $n_w = \frac{n_x}{2}$ the breakpoint occurs at $N \approx 5$.

Due to the cubic scaling we note that for large systems, *i.e.* state dimensions of 1000 or more, the Riccati approach or any (sparse) direct factorization method will be com-

putationally expensive. Therefore, for large-scale applications, other methods such as conjugate gradient may be more suitable for solving the KKT system.

We conclude that the Riccati based methods are most suitable for MHE problems with small to moderate system order and large horizons.

## 3.2. Structure-exploiting algorithms for unconstrained MHE

Consider the following general linear unconstrained MHE problem

$$\begin{aligned} \min_{\mathbf{x},\mathbf{w}} \quad & \|\bar{x}_0 + x_0 - \hat{x}_0\|^2_{P_0^{-1}} + \sum_{k=0}^{N-1} \|\bar{w}_k + w_k\|^2_{Q_k^{-1}} + \|h_k + C_k x_k + H_k w_k\|^2_{R_k^{-1}} \\ & + \|h_N + C_N x_N\|^2_{R_N^{-1}} \\ \text{s.t.} \quad & x_{k+1} = f_k + A_k x_k + G_k w_k, \quad k = 0, \ldots, N-1, \end{aligned} \tag{3.1}$$

with $\mathbf{x} = \{x_0, \ldots, x_N\}$ the unknown state sequence and $\mathbf{w} = \{w_0, \ldots, w_{N-1}\}$ the unknown process disturbances.

This equality constrained QP is obtained after linearization at $\bar{\mathbf{x}} = \{\bar{x}_0, \ldots, \bar{x}_N\}$ and $\bar{\mathbf{w}} = \{\bar{w}_0, \ldots, \bar{w}_{N-1}\}$ of the unconstrained nonlinear MHE problem (1.13) in a Gauss-Newton framework. More precisely, the nonlinear functions $f$ and $h$ are approximated by first order Taylor expansions

$$\begin{aligned} f_k(\bar{x}_k, \bar{w}_k) \quad & \approx f_k + A_k \Delta x_k + G_k \Delta w_k, \quad k = 0, \ldots, N-1, \\ y_k - h_k(\bar{x}_k, \bar{w}_k) \quad & \approx h_k + C_k \Delta x_k + H_k \Delta w_k, \quad k = 0, \ldots, N-1, \\ y_N - h_N(\bar{x}_N) \quad & \approx h_N + C_N \Delta x_N. \end{aligned} \tag{3.2}$$

For notational convenience we will from now on rename the system dimensions as $n = n_x$, $m = n_w$ and $p = n_y$.

Let us write the KKT system as $M\xi = r$, with

$$M = \begin{bmatrix} \Phi_0 & \Gamma_0^{\mathrm{T}} & 0 & & & & \\ \Gamma_0 & 0 & -\Upsilon^{\mathrm{T}} & & & & \\ 0 & -\Upsilon & \Phi_1 & & & & \\ & & & \ddots & & & \\ & & & & \Phi_{N-1} & \Gamma_{N-1}^{\mathrm{T}} & 0 \\ & & & & \Gamma_{N-1} & 0 & -I \\ & & & & 0 & -I & \Phi_N \end{bmatrix}, \quad \xi = \begin{bmatrix} z_0 \\ \lambda_0 \\ \vdots \\ z_{N-1} \\ \lambda_{N-1} \\ z_N \end{bmatrix}, \quad r = \begin{bmatrix} r_{d,0} \\ r_{p,0} \\ \vdots \\ r_{d,N-1} \\ r_{p,N-1} \\ r_{d,N} \end{bmatrix}, \tag{3.3}$$

where $r_{d,k}$ (resp. $r_{d,N}$) denotes the dual residual associated with $z_k = (x_k^{\mathrm{T}}, w_k^{\mathrm{T}})^{\mathrm{T}}$ (resp. $z_N = x_N$ ), and $r_{p,k}$ denotes the primal residual associated with $\lambda_k$ and where we defined

$$\Phi_0 = \begin{bmatrix} P_0^{-1} + C_0^{\mathrm{T}} R_0^{-1} C_0 & C_0^{\mathrm{T}} R_0^{-1} H_0 \\ H_0^{\mathrm{T}} R_0^{-1} C_0 & Q_0^{-1} + H_0^{\mathrm{T}} R_0^{-1} H_0 \end{bmatrix}, \quad r_{d,0} = \begin{bmatrix} P_0^{-1}(\hat{x}_0 - \bar{x}_0) - C_0^{\mathrm{T}} R_0^{-1} h_0 \\ -Q_0^{-1}\bar{w}_0 - H_0^{\mathrm{T}} R_0^{-1} h_0 \end{bmatrix},$$

$$\Phi_k = \begin{bmatrix} C_k^{\mathrm{T}} R_k^{-1} C_k & C_k^{\mathrm{T}} R_k^{-1} H_k \\ H_k^{\mathrm{T}} R_k^{-1} C_k & Q_k^{-1} + H_k^{\mathrm{T}} R_k^{-1} H_k \end{bmatrix}, \qquad r_{d,k} = \begin{bmatrix} -C_k^{\mathrm{T}} R_k^{-1} h_k \\ -Q_k^{-1}\bar{w}_k - H_k^{\mathrm{T}} R_k^{-1} h_k \end{bmatrix}, \qquad 1 \le k \le N-1$$

$$\Phi_N = \begin{bmatrix} C_N^{\mathrm{T}} R_N^{-1} C_N \end{bmatrix}, \qquad r_{d,N} = \begin{bmatrix} -C_N^{\mathrm{T}} R_N^{-1} h_N \end{bmatrix},$$

$$\Gamma_k = \begin{bmatrix} A_k & G_k \end{bmatrix}, \qquad r_{p,k} = \begin{bmatrix} -f_k \end{bmatrix}, \qquad 0 \le k \le N-1$$

$$\Upsilon = \begin{bmatrix} I_n \\ 0 \end{bmatrix}$$

## 3.2.1. Normal Riccati based solution method

**Lemma 3.2.1.** *The KKT matrix M (3.3) can be factorized by an LU decomposition $M = LU$ with*

$$L = \begin{bmatrix} \Sigma_{0+}^{-1} & & & & & & \\ \Gamma_0 & -P_1 & & & & & \\ 0 & -\Upsilon & \Sigma_{1+}^{-1} & & & & \\ & & & \ddots & & & \\ & & & & \Sigma_{N-1+}^{-1} & & \\ & & & & \Gamma_{N-1} & -P_N & \\ & & & & 0 & -I_n & \Sigma_{N+}^{-1} \end{bmatrix},$$

$$U = \begin{bmatrix} I_{n+m} & \Sigma_{0+}\Gamma_0^{\mathrm{T}} & & & & & \\ & I_n & P_1^{-1}\Upsilon^{\mathrm{T}} & & & & \\ & & I_{n+m} & \Sigma_{1+}\Gamma_1^{\mathrm{T}} & & & \\ & & & \ddots & & & \\ & & & & I_{n+m} & \Sigma_{N-1+}\Gamma_{N-1}^{\mathrm{T}} & \\ & & & & & I_n & P_N^{-1} \\ & & & & & & I_n \end{bmatrix},$$

*which can be recursively computed by Algorithm 1.*

**Algorithm 1.** *[Riccati recursion]*

1. *Initialization:* $P_0$

2. *For* $k = 0, \ldots, N-1$*:*

   a) *Measurement update step*

   *Let* $\Sigma_k = \begin{bmatrix} P_k & \\ & Q_k \end{bmatrix}$ *and* $D_k = \begin{bmatrix} C_k & H_k \end{bmatrix}$

   *Calculate:*

$$\Sigma_{k+} = \left(\Sigma_k^{-1} + D_k^{\mathrm{T}} R_k^{-1} D_k\right)^{-1} = \Sigma_k - \Sigma_k D_k^{\mathrm{T}} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1} D_k \Sigma_k$$

   b) *Model forwarding step*

   *Let* $\Gamma_k = \begin{bmatrix} A_k & G_k \end{bmatrix}$

   *Calculate:*

$$P_{k+1} = \Gamma_k \Sigma_{k+} \Gamma_k^{\mathrm{T}}$$

   *endfor.*

3. *Final time step:*

   a) *Measurement update step*

   *Let* $\Sigma_N = P_N$ *and* $D_N = C_N$

   *Calculate:*

$$\Sigma_{N+} = \left(\Sigma_N^{-1} + D_N^{\mathrm{T}} R_N^{-1} D_N\right)^{-1} = \Sigma_N - \Sigma_N D_N^{\mathrm{T}} \left(R_N + D_N \Sigma_N D_N^{\mathrm{T}}\right)^{-1} D_N \Sigma_N$$

A proof of Lemma 3.2.1 is now given.

*Proof.* To find out how $\Sigma_{0+}$, $P_1$, $\Sigma_{1+}$ etc. can be computed, it suffices to multiply out $LU$ and equate with $M$. This yields

$$\begin{aligned}
\Sigma_{0+}^{-1} &= \Phi_0 && (3.4)\\
&\vdots \\
-\Upsilon P_k^{-1} \Upsilon^{\mathrm{T}} + \Sigma_{k+}^{-1} &= \Phi_k && (3.5)\\
\Gamma_k \Sigma_{k+} \Gamma_k^{\mathrm{T}} - P_{k+1} &= 0 && (3.6)\\
&\vdots \\
-P_N^{-1} + \Sigma_{N+}^{-1} &= \Phi_N && (3.7)
\end{aligned}$$

Equation (3.6) yields exactly the model forwarding step of the Riccati recursion. Equations (3.4), (3.5) and (3.7) follow directly from the definition of $\Phi_0$, $\Phi_k$ and $\Phi_N$ respectively. □

The block structure of the KKT matrix and its *LU* factorization corresponding to Lemma 3.2.1 are illustrated in Figure 3.2. In order to solve the KKT system the matrices $L$ and $U$ do not need to be constructed, instead the factors $\Sigma_{0+}$, $P_1$, $\Sigma_{1+}$ etc. are computed and directly applied to the residual vector. The solution vector $\xi$ of primal and dual variables can be obtained after a forward solve $L\xi^{'} = r$ followed by a backward solve $U\xi = \xi^{'}$. This is formalized in Theorem 3.2.2. First, the direct calculations are presented by the following algorithms (Algorithms 2 and 3).

**Algorithm 2.** *[Direct forward vector recursion]*

1. *Initialization ($k = 0$):*
$$\begin{aligned} z_0^{'} &= \Sigma_{0+} r_{d,0} \\ \lambda_0^{'} &= P_1^{-1}(-r_{p,0} + \Gamma_0 z_0^{'}) \end{aligned}$$

2. *For $k = 1, \ldots, N-1$:*
$$\begin{aligned} z_k^{'} &= \Sigma_{k+}(\Upsilon \lambda_{k-1}^{'} + r_{d,k}) \\ \lambda_k^{'} &= P_{k+1}^{-1}(-r_{p,k} + \Gamma_k z_k^{'}) \end{aligned}$$

   *endfor.*

3. *Final time step:*
$$z_N^{'} = \Sigma_{N+}(\lambda_{N-1}^{'} + r_{d,N})$$

**Algorithm 3.** *[Direct backward vector recursion]*

1. *Initialization:*
$$z_N = z_N^{'}$$

2. *For $k = N-1, \ldots, 0$:*
$$\begin{aligned} \lambda_k &= \lambda_k^{'} - P_{k+1}^{-1} z_{k+1} \\ z_k &= z_k^{'} - \Sigma_{k+} \Gamma_k^{\mathrm{T}} \lambda_k \end{aligned}$$

   *endfor.*

**Theorem 3.2.2.** (KKT solution using direct *LU* factor-solve method) *The KKT system $M\xi = r$ with $M$,$\xi$ and $r$ as defined before (3.3) can be solved by performing the following steps*

1. *Factorize $M = LU$ according to Lemma 3.2.1 using the Riccati recursion of Algorithm 1,*
2. *Solve $L\xi' = r$ using Algorithm 2,*
3. *Solve $U\xi = \xi'$ using Algorithm 3.*

*Proof.* The proof follows directly from Lemma 3.2.1 and block elimination using the block structure in $L$, $U$ and $r$. □

The forward and backward vector solves proposed in Theorem 3.2.2 requires invertibility of matrices $P_k$ for all $k$ not only in solving the KKT system but also in forming the first dual residual $r_{d,0}$ which contains $P_0^{-1}$. This can be avoided by matrix calculation as formalized in Theorem 3.2.3 and Algorithms 4 and 5.

**Algorithm 4.** *[Forward vector recursion]*

1. *Initialization:*

$$d_0 = \begin{bmatrix} \hat{x}_0 - \bar{x}_0 \\ -\bar{w}_0 \end{bmatrix}$$

2. *For $k = 0, \ldots, N-1$:*

$$\begin{aligned} z_k' &= d_k - \Sigma_k D_k^{\mathrm{T}} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1} (D_k d_k + h_k) \\ \hat{x}_{k+1} &= f_k + \Gamma_k z_k' \\ d_{k+1} &= \begin{bmatrix} \hat{x}_{k+1} \\ -\bar{w}_{k+1} \end{bmatrix} \end{aligned}$$

*If ($k = N-1$) then*

$$d_N = \hat{x}_N$$

*else*

$$d_{k+1} = \begin{bmatrix} \hat{x}_{k+1} \\ -\bar{w}_{k+1} \end{bmatrix}$$

*endif.*

*endfor.*

3. *Final time step:*

$$z_N' = d_N - \Sigma_N D_N^{\mathrm{T}} \left(R_N + D_N \Sigma_N D_N^{\mathrm{T}}\right)^{-1} (D_N d_N + h_N)$$

**Algorithm 5.** *[Backward vector recursion]*

1. *Initialization ($k = N$:*

$$\begin{aligned} z_N &= z_N^{'} \\ \lambda_{N-1} &= C_N^{\mathrm{T}} \left(R_N + D_N \Sigma_N D_N^{\mathrm{T}}\right)^{-1} (D_N d_N + h_N) \end{aligned}$$

2. *For $k = N-1, \ldots, 1$:*

$$\begin{aligned} z_k &= z_k^{'} - \Sigma_{k+} \Gamma_k^{\mathrm{T}} \lambda_k \\ \lambda_{k-1} &= A_k^{\mathrm{T}} \lambda_k + C_k^{\mathrm{T}} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1} \left[D_k (d_k - \Sigma_k \Gamma_k^{\mathrm{T}} \lambda_k) + h_k\right] \end{aligned}$$

   *endfor.*

3. *Final step ($k = 0$):*

$$z_0 = z_0^{'} - \Sigma_{0+} \Gamma_0^{\mathrm{T}} \lambda_0$$

**Theorem 3.2.3.** (KKT solution using modified *LU* factor-solve method) *The KKT system $M\xi = r$ with $M$,$\xi$ and $r$ as defined before (3.3) can be solved by performing the following steps*

1. *Factorize $M = LU$ according to Lemma 3.2.1 using the Riccati recursion of Algorithm 1,*
2. *Solve $L\xi^{'} = r$ using Algorithm 4,*
3. *Solve $U\xi = \xi^{'}$ using Algorithm 5.*

*Proof.* The factorization step is identical to Theorem 3.2.2, hence, it remains to prove that the second and third step (Algorithms 4 and 5) yield an identical optimal vector $\xi$ as in Theorem 3.2.2 (Algorithms 2 and 3).

First, consider the forward vector solve. At time step $k = 0$ we have that

$$\begin{aligned} z_0^{'} &= \Sigma_{0+} r_{d,0} \\ &= \left(\Sigma_0^{-1} + D_0^{\mathrm{T}} R_0^{-1} D_0\right)^{-1} \left(\Sigma_0^{-1} d_0 - D_0^{\mathrm{T}} R_0^{-1} h_0\right) \\ &= d_0 - \Sigma_0 D_0^{\mathrm{T}} \left(R_0 + D_0 \Sigma_0 D_0^{\mathrm{T}}\right)^{-1} (D_0 d_0 + h_0) \end{aligned}$$

with $d_0 = \begin{bmatrix} \hat{x}_0 - \bar{x}_0 \\ -\bar{w}_0 \end{bmatrix}$ and $\Sigma_0$ and $D_0$ as defined in Algorithm 1. For the last step we used the results from Equations (3.24) - (3.27). Next, the variables $\lambda_k^{'}$, which are not calculated in Algorithm 4, can be written as

$$\begin{aligned} \lambda_k^{'} &= P_{k+1}^{-1}(-r_{p,k} + \Gamma_k z_k^{'}) \\ &= P_{k+1}^{-1}(f_k + \Gamma_k z_k^{'}) \\ &= P_{k+1}^{-1}\hat{x}_{k+1} \\ &= P_{k+1}^{-1}\Upsilon^{\mathrm{T}} d_{k+1} \end{aligned}$$

with $d_{k+1} = \begin{bmatrix} \hat{x}_{k+1} \\ -\bar{w}_{k+1} \end{bmatrix}$ and $\Upsilon = \begin{bmatrix} I \\ 0 \end{bmatrix}$.

Then

$$\begin{aligned} z_{k+1}^{'} &= \Sigma_{k+1+}(\Upsilon\lambda_k^{'} + r_{d,k+1}) \\ &= \Sigma_{k+1+}(\Sigma_{k+1}^{-1} d_{k+1} - D_{k+1}^{\mathrm{T}} R_{k+1}^{-1} h_{k+1}) \\ &= d_{k+1} - \Sigma_{k+1} D_{k+1}^{\mathrm{T}} \left(R_{k+1} + D_{k+1}\Sigma_{k+1} D_{k+1}^{\mathrm{T}}\right)^{-1} (D_{k+1} d_{k+1} + h_{k+1}) \end{aligned}$$

which concludes the proof that identical $z_k^{'}$ are obtained in Algorithms 2 and 4.

Second, consider the backward vector solve.

$$\begin{aligned} \lambda_{N-1} &= \lambda_{N-1}^{'} - P_N^{-1} z_N \\ &= C_N^{\mathrm{T}} \left(R_N + C_N P_N C_N^{\mathrm{T}}\right)^{-1} (C_N d_N + h_N) \end{aligned}$$

where we plugged in $\lambda_{N-1}^{'} = P_N^{-1} d_N$ and $z_N = z_N^{'} = d_N - P_N C_N^{\mathrm{T}} \left(R_N + C_N P_N C_N^{\mathrm{T}}\right)^{-1} (C_N d_N + h_N)$. For $k < N$ we plug in $\lambda_{k-1}^{'} = \Sigma_k^{-1}\Upsilon^{\mathrm{T}} d_k$ and $z_k = z_k^{'} - \Sigma_{k+}\Gamma_k^{\mathrm{T}}\lambda_k$ into

$$\begin{aligned} \lambda_{k-1} &= \lambda_{k-1}^{'} - P_k^{-1} z_k \\ &= \Sigma_k^{-1}\Upsilon^{\mathrm{T}} d_k - \Sigma_k^{-1}\Upsilon^{\mathrm{T}} d_k + \Sigma_k^{-1}\Upsilon^{\mathrm{T}}\Sigma_{k+}\Gamma_k^{\mathrm{T}}\lambda_k \\ &\quad + \Sigma_k^{-1}\Upsilon^{\mathrm{T}}\Sigma_k D_k^{\mathrm{T}} \left(R_N + D_N\Sigma_N D_N^{\mathrm{T}}\right)^{-1} (D_N d_N + h_N) \end{aligned}$$

which concludes the proof that identical $z_k$ are obtained in Algorithms 3 and 5. □

**Lemma 3.2.4.** *The MHE problem (3.1) has a unique optimal primal-dual pair* $(\mathbf{z}, \lambda)$ *if the KKT matrix M (3.3) is full rank. Sufficient conditions are*

- $P_0 \succ 0$

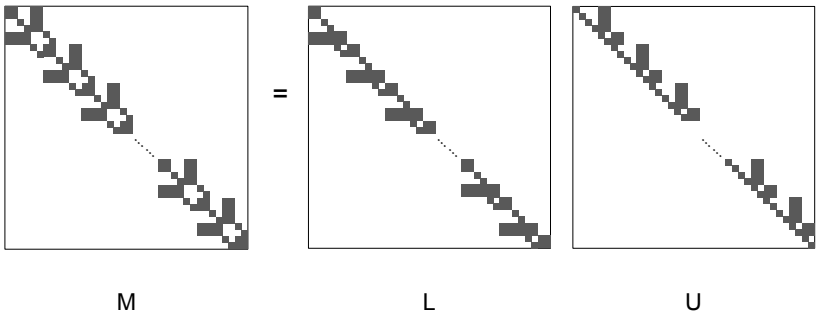



**Figure 3.2.** Visualization of the block structure in the KKT matrix and its *LU* decomposition.

- $Q_k \succ 0, \forall k$
- $R_k \succ 0, \forall k$

*Proof.* From Lemma 3.2.1 it is clear that the KKT matrix is full rank iff the factor $L$ is full rank. This occurs when $\Sigma_{k+}$ for $k \in [0,N]$ and $P_k$ for $k \in [1,N]$ are positive definite. A necessary condition is that $\left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)$ is positive definite for $k \in [0,N]$. This is guaranteed if $R_k \succ 0$ and $\Sigma_k \succ 0\ \forall k$. The latter is fulfilled if $P_0 \succ 0$ and $Q_k \succ 0, \forall k$ (see [107] for a proof). □

**Remark 1.** *In practical implementations the forward Riccati matrix and forward vector recursion are combined in a single loop.*

**Remark 2.** *We opted to present the methods in a general formulation, i.e. with* $H \neq 0$. *If* $H_k = 0$ *for all k, then there is no* mixing *between x and w and the Riccati recursion involves covariances of order* $n \times n$. *The resulting algorithms are similar but simplified. The modified decomposition and algorithms are given in Appendix A.*

### 3.2.2. Square-root Riccati based solution method

Suppose a square-root or more precisely a (upper) triangular Cholesky factor $S_0$ of the state covariance matrix is given such that $P_0 = S_0^{\mathrm{T}} S_0$. Assume that also square roots of the covariance matrices $Q_k$ and $R_k$ are given such that $Q_k = W_k^{\mathrm{T}} W_k$ and $R_k = V_k^{\mathrm{T}} V_k$.

Then the cost function of MHE problem (3.1) can be written as follows

$$\begin{aligned} J_N &= \|S_0^{-\mathrm{T}}(\bar{x}_0 + x_0 - \hat{x}_0)\|_2^2 \\ &\quad + \sum_{k=0}^{N-1} \|W_k^{-\mathrm{T}}(\bar{w}_k + w_k)\|_2^2 + \|V_k^{-\mathrm{T}}(h_k + C_k x_k + H_k w_k)\|_2^2 \\ &\quad + \|V_N^{-\mathrm{T}}(h_N + C_N x_N)\|_2^2. \end{aligned} \tag{3.8}$$

The symmetry which is inherent in the KKT system (3.3) can be exploited. Here we describe a symmetric decomposition and a resulting recursive solution strategy which uses square-roots and orthogonal transformations thereby increasing the robustness of the methods. The block structure of the KKT matrix and its symmetric decomposition are illustrated in Figure 3.3.

**Lemma 3.2.5.** *If the KKT matrix M (3.3) is full rank (see Lemma 3.2.4 for sufficient conditions), then it is symmetric indefinite and can be factorized by an indefinite Cholesky decomposition $M = LDL^T$ with*

$$L = \begin{bmatrix} T_{0+}^{-1} & & & & & & & \\ \Gamma_0 T_{0+}^T & S_1^T & & & & & & \\ 0 & -\Upsilon S_1^{-1} & T_{1+}^{-1} & & & & & \\ & & \ddots & & & & & \\ & & & T_{N-1+}^{-1} & & & & \\ & & & \Gamma_{N-1} T_{N-1+}^T & S_N^T & & & \\ & & & 0 & -\Upsilon S_N^{-1} & T_{N+}^{-1} & & \end{bmatrix},$$

$$D = \begin{bmatrix} I_{n+m} & & & & & & & \\ & -I_n & & & & & & \\ & & I_{n+m} & & & & & \\ & & & \ddots & & & & \\ & & & & I_{n+m} & & & \\ & & & & & -I_n & & \\ & & & & & & I_n & \end{bmatrix},$$

*which can be recursively computed by Algorithm 6.*

**Algorithm 6.** *[Square-root Riccati recursion]*

1. *Initialization:* $S_0$
2. *For* $k = 0, \ldots, N-1$*:*

*a) Measurement update step*

*Let* $T_k = \begin{bmatrix} S_k & \\ & W_k \end{bmatrix}$ *(note that* $\Sigma_k = T_k^{\mathrm{T}} T_k$*)*
*and* $D_k = \begin{bmatrix} C_k & H_k \end{bmatrix}$

*Compute QR-factorization:*

$$\begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix} = \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}$$

*with* $\tilde{R}_k^e = \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1/2}$ *and* $\tilde{K}_k = \Sigma_k D_k^{\mathrm{T}} \tilde{R}_k^e$.

*b) Model forwarding step*

*Let* $\Gamma_k = \begin{bmatrix} A_k & G_k \end{bmatrix}$
*Compute QR-factorization:*

$$T_{k+} \Gamma_k^T = \begin{bmatrix} \hat{Q}_{k+1} & \check{Q}_{k+1} \end{bmatrix} \begin{bmatrix} \hat{S}_{k+1} \\ 0 \end{bmatrix}$$

*endfor.*

*3. Final time step:*

*a) Measurement update step*

*Let* $T_N = S_N$ *(note that* $\Sigma_N = T_N^{\mathrm{T}} T_N$*)*
*and* $D_N = C_N$

*Compute QR-factorization:*

$$\begin{bmatrix} V_N & \\ T_N D_N^{\mathrm{T}} & T_N \end{bmatrix} = \begin{bmatrix} \bar{Q}_N & \tilde{Q}_N \end{bmatrix} \begin{bmatrix} (\tilde{R}_N^e)^{-1} & \tilde{K}_N^{\mathrm{T}} \\ 0 & T_{N+} \end{bmatrix}$$

*with* $\tilde{R}_N^e = \left(R_N + D_N \Sigma_N D_N^{\mathrm{T}}\right)^{-1/2}$ *and* $\tilde{K}_N = \Sigma_N D_N^{\mathrm{T}} \tilde{R}_N^e$.

A proof of Lemma 3.2.5 is now given.

*Proof.* To find out how $T_{0+}$, $S_1$, $T_{1+}$ etc. can be computed, we multiply out $LDL^{\mathrm{T}}$ and

equate with $M$. Then,

$$\begin{aligned} T_{0+}^{-1}T_{0+}^{-\mathrm{T}} &= \Phi_0 && (3.9)\\ &\vdots \\ -\Upsilon S_k^{-1}S_k^{-T}\Upsilon^{\mathrm{T}} + T_{k+}^{-1}T_{k+}^{-T} &= \Phi_k && (3.10)\\ \Gamma_k T_{k+}^{\mathrm{T}} T_{k+}\Gamma_k^{\mathrm{T}} - S_{k+1}^{\mathrm{T}} S_{k+1} &= 0 && (3.11)\\ &\vdots \\ -S_N^{-1}S_N^{-T} + T_{N+}^{-1}T_{N+}^{-\mathrm{T}} &= \Phi_N && (3.12) \end{aligned}$$

First we prove the measurement update step, i.e. Equations (3.9), (3.10) and (3.12). Let us define $D_k = \begin{bmatrix} C_k & H_k \end{bmatrix}$. Note that

$$\begin{aligned} \Phi_0 &= \begin{bmatrix} S_0^{-1}S_0^{-\mathrm{T}} & \\ & W_0^{-1}W_0^{-\mathrm{T}} \end{bmatrix} + D_0^{\mathrm{T}} V_0^{-1} V_0^{-\mathrm{T}} D_0, \\ \Phi_k &= \begin{bmatrix} 0 & \\ & W_k^{-1}W_k^{-\mathrm{T}} \end{bmatrix} + D_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} D_k, \quad 1 \le k \le N-1 \\ \Phi_N &= D_N^{\mathrm{T}} V_N^{-1} V_N^{-\mathrm{T}} D_N. \end{aligned} \tag{3.13}$$

Let us compute the $QR$ factorization

$$\begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix} = \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix} \tag{3.14}$$

with $T_k = \begin{bmatrix} S_k & \\ & W_k \end{bmatrix}$ and $\tilde{K}_k^{\mathrm{T}} = \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-T/2} \Sigma_k D_k$ and where $\bar{Q}_k$ and $\tilde{Q}_k$ are orthogonal matrices. To see that this is a valid choice for $T_{k+}$, invert the triangular matrix on the right-hand-side

$$\begin{bmatrix} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}^{-1} = \begin{bmatrix} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1/2} & -\left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1/2} \tilde{K}_k^{\mathrm{T}} T_{k+}^{-1} \\ 0 & T_{k+}^{-1} \end{bmatrix}$$

which follows from application of the Schur complement (see equations (2.13)-(2.17)). From the $QR$ factorization (3.14) we know that

$$\begin{aligned} \begin{bmatrix} \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}^{-1} \bar{Q}^{\mathrm{T}} &= \begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix}^{-1} && (3.15)\\ &= \begin{bmatrix} V_k^{-1} & \\ D_k^{\mathrm{T}} V_k^{-1} & T_k^{-1} \end{bmatrix} && (3.16) \end{aligned}$$

where we assumed without loss of generality that $\begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix}$ is invertible.

Now, it can be seen that

$$\begin{bmatrix} \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}^{-1} \begin{bmatrix} \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{1/2} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}^{-\mathrm{T}} \\ = \begin{bmatrix} \star & -\left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1/2} \tilde{K}_k^{\mathrm{T}} T_{k+}^{-1} T_{k+}^{-\mathrm{T}} \\ -T_{k+}^{-1} T_{k+}^{-\mathrm{T}} \tilde{K}_k \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-T/2} & T_{k+}^{-1} T_{k+}^{-\mathrm{T}} \end{bmatrix}$$

where $\star = \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1} + \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1/2} \tilde{K}_k^{\mathrm{T}} T_{k+}^{-1} T_{k+}^{-\mathrm{T}} \tilde{K}_k \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-T/2}$.

Moreover, we have

$$\begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix}^{-1} \begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix}^{-\mathrm{T}} = \begin{bmatrix} V_k^{-1} V_k^{-\mathrm{T}} & V_k^{-1} V_k^{-\mathrm{T}} D_k \\ D_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} & D_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} D_k + T_k^{-1} T_k^{-\mathrm{T}} \end{bmatrix}$$

Since both expressions are equal by the orhtogonality of $\bar{Q}$, we obtain by equating the bottom right elements

$$T_{k+}^{-1} T_{k+}^{-\mathrm{T}} \quad = \quad T_k^{-1} T_k^{-\mathrm{T}} + D_k^{\mathrm{T}} V_k^{-1} V_k^{-\mathrm{T}} D_k \tag{3.17}$$

which concludes the proof for Equations (3.9) and (3.10). The proof for Equation (3.12) is similar.

Next, we prove the model forwarding step. From Eq. (3.6) it can be seen that with any orthogonal matrix $\hat{Q}_{k+1}$

$$S_{k+1} \quad = \quad \hat{Q}_{k+1}^T T_{k+} \Gamma_k^T, \quad 0 \le k \le N-1 \tag{3.18}$$

We compute a *QR* factorization

$$T_{k+}\Gamma_k^T = \begin{bmatrix} \hat{Q}_{k+1} & \check{Q}_{k+1} \end{bmatrix} \begin{bmatrix} \hat{S}_{k+1} \\ 0 \end{bmatrix} \tag{3.19}$$

where $\hat{Q}_{k+1}$ and $\check{Q}_{k+1}$ are orthogonal matrices and $\hat{S}_{k+1}$ is upper triangular. □

**Theorem 3.2.6.** (KKT solution using direct $LDL^{\mathrm{T}}$ factor-solve method) *The KKT system $M\xi = r$ with $M$,$\xi$ and $r$ as defined before (3.3) can be solved by performing the following steps*

1. *Factorize $M = LDL^{\mathrm{T}}$ according to Lemma 3.2.5 using the Riccati recursion of Algorithm 6,*
2. *Solve $L\xi' = r$,*

3. *Solve* $L^{\mathrm{T}} D\xi = \xi'$

*Proof.* The proof follows directly from Lemma 3.2.5 and block elimination using the block structure in $L$ and $r$. □

Similarly to the direct solution of Theorem 3.2.2 the forward and backward vector solves according to Theorem 3.2.6 require invertibility of matrices $S_k$ for all $k$ not only in solving the KKT system but also in forming the first dual residual $r_{d,0}$ which contains $S_0^{-1}S_0^{-\mathrm{T}}$. This can be avoided by using the approach of Theorem 3.2.3 but reformulated in terms of square-roots. More specifically, the square-root Riccati recursion of Algorithm 6 is performed, but the forward and backward solves corrspond to the $LU$ factorsization in order to avoid unnecessary matrix inversions. This is proposed in Theorem 3.2.7.

**Theorem 3.2.7.** (KKT solution using modified $LDL^{\mathrm{T}}$ factor-solve method employing square-roots) *The KKT system $M\xi = r$ with $M$,$\xi$ and $r$ as defined before (3.3) can be solved by performing the following steps*

1. *Factorize $M = LDLt$ according to Lemma 3.2.5 using the Riccati recursion of Algorithm 6,*
2. *Forward vector solve using Algorithm 7,*
3. *Backward vector solve using Algorithm 8.*

**Algorithm 7.** *[Square-root forward vector recursion]*

1. *Initialization:*
$$d_0 = \begin{bmatrix} \hat{x}_0 - \bar{x}_0 \\ -\bar{w}_0 \end{bmatrix}$$
2. *For $k = 0, \ldots, N-1$:*
$$\begin{aligned} z_k' &= d_k - \tilde{K}_k (\tilde{R}_k^e)^{\mathrm{T}} (D_k d_k + h_k) \\ \hat{x}_{k+1} &= f_k + \Gamma_k z_k' \\ d_{k+1} &= \begin{bmatrix} \hat{x}_{k+1} \\ -\bar{w}_{k+1} \end{bmatrix} \end{aligned}$$

*If ($k = N-1$) then*
$$d_N = \hat{x}_N$$

*else*
$$d_{k+1} = \begin{bmatrix} \hat{x}_{k+1} \\ -\bar{w}_{k+1} \end{bmatrix}$$

*endif.*

*endfor.*

3. *Final time step:*
$$z_N' = d_N - \tilde{K}_N(\tilde{R}_N^e)^{\mathrm{T}}(D_N d_N + h_N)$$

**Algorithm 8.** *[Square-root backward vector recursion]*

1. *Initialization ($k = N$:*
$$\begin{aligned} z_N &= z_N' \\ \lambda_{N-1} &= C_N^{\mathrm{T}}\tilde{R}_N^e(\tilde{R}_N^e)^{\mathrm{T}}(D_N d_N + h_N) \end{aligned}$$

2. *For $k = N-1,\ldots,1$:*
$$\begin{aligned} z_k &= z_k' - T_{k+}^{\mathrm{T}} T_{k+}\Gamma_k^{\mathrm{T}}\lambda_k \\ \lambda_{k-1} &= A_k^{\mathrm{T}}\lambda_k + C_k^{\mathrm{T}}\tilde{R}_k^e(\tilde{R}_k^e)^{\mathrm{T}}\left[D_k(d_k - T_k^{\mathrm{T}}T_k\Gamma_k^{\mathrm{T}}\lambda_k) + h_k\right] \end{aligned}$$

*endfor.*

3. *Final step ($k = 0$):*
$$z_0 = z_0' - T_{0+}^{\mathrm{T}} T_{0+}\Sigma_{0+}\Gamma_0^{\mathrm{T}}\lambda_0$$

A proof of Theorem 3.2.7 is now given.

*Proof.* The first step follows from Lemma 3.2.5.

For the second step it suffices to prove equivalence of Algorithms 4 and 7, which amounts to verifying that

$$\Sigma_k D_k^{\mathrm{T}}\left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1} = \tilde{K}_k(\tilde{R}_k^e)^{\mathrm{T}}.$$

This follows directly from the definitions of $\tilde{K}_k$ and $\tilde{R}_k^e$, repeated here for convenience

$$\begin{aligned} \tilde{R}_k^e &= \left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1/2} \\ \tilde{K}_k &= \Sigma_k D_k^{\mathrm{T}}\left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1/2} \end{aligned}$$

Similarly, for the third step equivalence of Algorithms 5 and 8 is easily verified by checking that

$$C_k^{\mathrm{T}}\left(R_k + D_k\Sigma_k D_k^{\mathrm{T}}\right)^{-1} = C_k^{\mathrm{T}}(\tilde{R}_k^e)^{\mathrm{T}}\tilde{R}_k^e$$

which follows from the definition of $\tilde{R}_k^e$. □

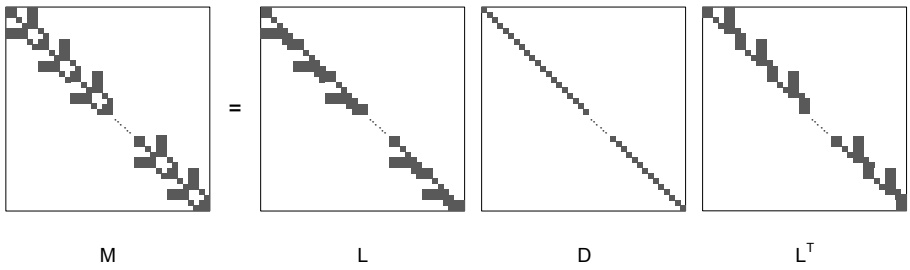



**Figure 3.3.** Visualization of the block structure in the KKT matrix and its indefinite Cholesky decomposition.

**Note that only the *R*-factors of both the measurement update step and the time update step are needed for the forward and backward solves. Hence the algorithms can use *Q*-less *QR* factorizations. See Appendix B allowing a reduction in computation time.**

**Remark 3.** *Also for the square-root Riccati method, the algorithms can be simplified if $H \equiv 0$. The adapted algorithms are given in Appendix A.*

### 3.2.3. Structured *QR* factorizations

The systematic use of orthogonal transformations to factorize arbitrary matrices was initiated by Givens [74] and Householder [98]. In Appendix B algorithms and flop counts are provided for Givens and Householder methods which are still standard practice due to their excellent numerical stability properties [22, 77]. A row-reordering of the matrices involved in the measurement update step, is known to lead to computational savings in the factorizations, see standard textbooks, *i.e.* by Verhaegen and Verduld [190] or Kailath et al [111]. Both standard and row-reordered (structured) versions have been implented in C and numerical simulation results are presented in Appendix B. In the numerical examples presented in Section 3.5 these structured QR methods are used in the square-root MHE algorithms.

## 3.3. Proof of equivalence of unconstrained MHE and the Kalman filter/smoother

The equivalence of the Kalman filter/smoother to a weighted least-squares estimator is well-known and is described in several papers and text books, *e.g.* [170], [195], [111,

§10.6], [190, Ch. 1], [71]. However, because the derivation via the optimality conditions (see also [146]) is crucial in showing that the moving horizon approximation with the arrival cost computed from the forward recursions is exact[1] in the linear unconstrained case and that MHE yields a sequence of smoothed estimates and a filtered estimate, we present the derivation in this section.

Consider the following optimization problem at $N = 0$

$$\begin{array}{ll} \min_{x_0,v_0} & \frac{1}{2}\{(x_0-\hat{x}_0)^{\mathrm{T}}P_0^{-1}(x_0-\hat{x}_0)+v_0^{\mathrm{T}}R_0^{-1}v_0\}, \\ \text{s.t.} & y_0 = h_0+C_0x_0+v_0. \end{array} \tag{3.20}$$

The optimality conditions are

$$\begin{bmatrix} P_0^{-1} & C_0^{\mathrm{T}} & \\ C_0 & & I \\ & I & R_0^{-1} \end{bmatrix} \begin{bmatrix} x_0 \\ \pi_0 \\ v_0 \end{bmatrix} = \begin{bmatrix} P_0^{-1}\hat{x}_0 \\ y_0-h_0 \\ 0 \end{bmatrix}, \tag{3.21}$$

where $\pi_0$ is the vector of Lagrange multipliers corresponding to the equality constraints and $I$ represents the identity matrix of appropriate dimensions.

Block elimination yields

$$\pi_0 = R_0v_0 \tag{3.22}$$
$$v_0 = y_0-h_0-C_0x_0 \tag{3.23}$$
$$x_0 = (P_0^{-1}+C_0^{\mathrm{T}}R_0^{-1}C_0)^{-1}(P_0^{-1}\hat{x}_0+C_0^{\mathrm{T}}R_0^{-1}(y_0-h_0)). \tag{3.24}$$

The following equalities are a direct result of the *Matrix Inversion Lemma*

$$(P_0^{-1}+C_0^TR_0^{-1}C_0)^{-1} = P_0-P_0C_0^T(R_0+C_0P_0C_0^T)^{-1}C_0P_0, \tag{3.25}$$
$$(P_0^{-1}+C_0^TR_0^{-1}C_0)^{-1}C_0^TR_0^{-1} = P_0C_0^T(R_0+C_0P_0C_0^T)^{-1}. \tag{3.26}$$

Using this and some matrix calculation the following expression for $x_0$ can be obtained

$$x_0 = \hat{x}_0+P_0C_0^{\mathrm{T}}(R_0+C_0P_0C_0^{\mathrm{T}})^{-1}(y_0-h_0-C_0\hat{x}_0). \tag{3.27}$$

Note that this exactly corresponds to the Kalman filter measurement update: $x0 = \hat{x}_{0+}$ and $(P_0^{-1}+C_0^TR_0^{-1}C_0)^{-1} = P_{0+}$ (see equations (2.72) and (2.69)).

Next, consider the case $N = 1$. The optimization problem now becomes

$$\begin{array}{ll} \min_{x_0,v_0,w_0,x_1,v_1} & \frac{1}{2}\{(x_0-\hat{x}_0)^TP_0^{-1}(x_0-\hat{x}_0)+w_0^TQ_0^{-1}w_0+v_0^TR_0^{-1}v_0+v_1^TR_1^{-1}v_1\}, \\ & y_0 = h_0+C_0x_0+v_0, \\ \text{s.t.} & y_1 = h_1+C_1x_1+v_1, \\ & x_1 = f_0+A_0x_0+G_0w_0. \end{array} \tag{3.28}$$

[1] Meaning that the results for all three estimators, the batch estimator, MHE and the Kalman filter, coincide.

The recursive solution can again be obtained from the KKT optimality conditions

$$\begin{bmatrix} P_0^{-1} & C_0^{\mathrm{T}} & & & A_0^{\mathrm{T}} & & & \\ C_0 & & I & & & & & \\ & I & R_0^{-1} & & & & & \\ & & & Q_0^{-1} & G_0^{\mathrm{T}} & & & \\ A_0 & & & G_0 & & -I & & \\ & & & & -I & & C_1^{\mathrm{T}} & \\ & & & & & C_1 & & I \\ & & & & & & I & R_1^{-1} \end{bmatrix} \begin{bmatrix} x_0 \\ \pi_0 \\ v_0 \\ w_0 \\ \lambda_0 \\ x_1 \\ \pi_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} P_0^{-1}\hat{x}_0 \\ y_0 - h_0 \\ 0 \\ 0 \\ -f_0 \\ 0 \\ y_1 - h_1 \\ 0 \end{bmatrix}. \tag{3.29}$$

Here $\pi_0$ and $\pi_1$ are the vector of Lagrange multipliers corresponding to the first two equality constraints (measurement equations), while $\lambda_0$ is the vector of Lagrange multipliers corresponding to the last equality constraint (state equation).

Using the insight obtained from the case $N = 0$ the variables $\pi_0$ and $v_0$ can be eliminated, yielding

$$\begin{bmatrix} P_0^{-1} + C_0^{\mathrm{T}} R_0^{-1} C_0 & & A_0^{\mathrm{T}} & & & \\ & Q_0^{-1} & G_0^{\mathrm{T}} & & & \\ A_0 & G_0 & & -I & & \\ & & -I & & C_1^{\mathrm{T}} & \\ & & & C_1 & & I \\ & & & & I & R_1^{-1} \end{bmatrix} \begin{bmatrix} x_0 \\ w_0 \\ \lambda_0 \\ x_1 \\ \pi_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} P_0^{-1}\hat{x}_0 + C_0^{\mathrm{T}} R_0^{-1}(y_0 - h_0) \\ 0 \\ -f_0 \\ 0 \\ y_1 - h_1 \\ 0 \end{bmatrix}. \tag{3.30}$$

Block elimination of the variables $w_0$, $\lambda_0$ and $x_1$ gives

$$\begin{bmatrix} P_0^{-1} + C_0^{\mathrm{T}} R_0^{-1} C_0 & A_0^{\mathrm{T}} C_1^{\mathrm{T}} & \\ C_1 A_0 & -C_1 G_0 Q_0 G_0^{\mathrm{T}} C_1^{\mathrm{T}} & I \\ & I & R_1^{-1} \end{bmatrix} \begin{bmatrix} x_0 \\ \pi_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} P_0^{-1}\hat{x}_0 + C_0^{\mathrm{T}} R_0^{-1}(y_0 - h_0) \\ y_1 - h_1 - C_1 f_0 \\ 0 \end{bmatrix}. \tag{3.31}$$

We know that by construction $x_1 = f_0 + A_0 x_0 + G_0 w_0$, which can be rewritten with the optimal $w_0 = -Q_0 G_0^{\mathrm{T}} C_1^{\mathrm{T}} \pi_1$ as

$$x_1 = f_0 + A_0 x_0 - G_0 Q_0 G_0^{\mathrm{T}} C_1^{\mathrm{T}} \pi_1. \tag{3.32}$$

Furthermore, the following definitions are introduced in accordance with the Kalman filter covariance time update formula (2.69)

$$P_1 = A_0 P_{0+} A_0^{\mathrm{T}} + G_0 Q_0 G_0^{\mathrm{T}}, \tag{3.33}$$

$$\hat{x}_1 = f_0 + A_0 \hat{x}_{0+}, \tag{3.34}$$

with $P_{0+}$ and $x_{0+}$ as defined before.

Then, an equivalent representation of the above set of optimality conditions (3.31) is

$$\begin{bmatrix} P_1^{-1} & C_1^{\mathrm{T}} & \\ C_1 & & I \\ & I & R_1^{-1} \end{bmatrix} \begin{bmatrix} x_1 \\ \pi_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} P_1^{-1}\hat{x}_1 \\ y_1 - h_1 \\ 0 \end{bmatrix}. \tag{3.35}$$

To verify that this is indeed an equivalent representation let us start from the first block equation of (3.31)

$$P_{0+}^{-1}x_0 + A_0^{\mathrm{T}}C_1^{\mathrm{T}}\pi_1 = P_0^{\mathrm{T}}\hat{x}_0 + C_0^{\mathrm{T}}R_0^{-1}(y_0 - h_0).$$

with $P_{0+} = (P_0^{-1} + C_0^{\mathrm{T}}R_0^{-1}C_0)^{-1}$ as before.

Left multiplication of both sides with $A_0P_{0+}$ and addition of $f_0$ gives

$$\begin{aligned} f_0 + A_0x_0 + A_0P_{0+}A_0^{\mathrm{T}}C_1^{\mathrm{T}}\pi_1 \\ = f_0 + A_0P_{0+}[P_0^{-1}\hat{x}_0 + C_0^{\mathrm{T}}R_0^{-1}(y_0 - h_0)]. \end{aligned} \tag{3.36}$$

Adding and subtracting $G_0Q_0G_0^{\mathrm{T}}C_1^{\mathrm{T}}\pi_1$ we obtain for the **left hand side** of eq.(3.36)

$$\begin{aligned} \text{LHS} &= f_0 + A_0x_0 - G_0Q_0G_0^{\mathrm{T}}C_1^{\mathrm{T}}\pi_1 + G_0Q_0G_0^{\mathrm{T}}C_1^{\mathrm{T}}\pi_1 + A_0P_{0+}A_0^{\mathrm{T}}C_1^{\mathrm{T}}\pi_1, \\ &= x_1 + P_1C_1^{\mathrm{T}}\pi_1. \end{aligned}$$

Where we used eq. (3.32) for $x_1$ and the definition of $P_1$.

The **right hand side** of eq.(3.36) can be reformulated using the *Matrix Inversion Lemma*

$$\begin{aligned} \text{RHS} &= f_0 + A_0P_{0+}[P_0^{-1}\hat{x}_0 + C_0^{\mathrm{T}}R_0^{-1}(y_0 - h_0)], \\ &= f_0 + A_0\hat{x}_{0+}, \\ &= \hat{x}_1. \end{aligned}$$

Where we used the definitions of $\hat{x}_{0+}$ (3.27) and $\hat{x}_1$ (3.34).

Finally, left multiplication of both sides with $P_1^{-1}$ gives the first block equation of (3.35).

Block elimination applied to (3.35) leads to the solution (analoguous to (3.27))

$$x_1 = \hat{x}_1 + P_1C_1^T(R + C_1P_1C_1^T)^{-1}(y_1 - h_1 - C_1\hat{x}_1). \tag{3.37}$$

This concludes the proof for $N = 1$.

By comparing the equivalent representations (3.31) and (3.35) it is seen how the information of the first time step is propagated to the next yielding a recursive solution.

The proof for $N > 1$ follows by recursively applying the procedure described above.□

Equivalently, it can be proven that solving the optimization problem (3.28) yields not only a filtered estimate of $x_1$ but also a smoothed estimate for $x_0$. To this end, consider again the optimality conditions for $N = 1$ written compactly as follows

$$\begin{bmatrix} P_{0+}^{-1} & & A_0^{\mathrm{T}} & \\ & Q_0^{-1} & G_0^{\mathrm{T}} & \\ A_0 & G_0 & & -I \\ & & -I & C_1^{\mathrm{T}} R_1^{-1} C_1 \end{bmatrix} \begin{bmatrix} x_0 \\ w_0 \\ \lambda_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} P_{0+}^{-1} x_{0+} \\ 0 \\ -f_0 \\ C_1^{\mathrm{T}} R_1^{-1} (y_1 - h_1) \end{bmatrix}. \tag{3.38}$$

The first block equation gives

$$x_0 = \hat{x}_{0+} - P_{0+} A_0^{\mathrm{T}} \lambda_0. \tag{3.39}$$

The second block equation gives

$$w_0 = -Q_0 G_0^{\mathrm{T}} \lambda_0. \tag{3.40}$$

Plugging this into the third block equation yields

$$\begin{aligned} x_1 &= f_0 + A_0 \hat{x}_{0+} - A_0 P_{0+} A_0^{\mathrm{T}} \lambda_0 - G_0 Q_0 G_0^{\mathrm{T}} \lambda_0, && (3.41) \\ &= f_0 + A_0 \hat{x}_{0+} - P_1 \lambda_0, && (3.42) \\ &= \hat{x}_1 - P_1 \lambda_0. && (3.43) \end{aligned}$$

Now, since the optimal $x_1$ equals the filtered estimate $\hat{x}_{1+}$, it follows that

$$\lambda_0 = P_1^{-1} (\hat{x}_1 - \hat{x}_{1+}). \tag{3.44}$$

Plugging this into eq. (3.39) gives

$$x_0 = \hat{x}_{0+} + P_{0+} A_0^{\mathrm{T}} P_1^{-1} (\hat{x}_{1+} - \hat{x}_1), \tag{3.45}$$

which is exaclty the RTS smoothed estimate (equations (2.150) and (2.152) ). The proof for $N > 1$ follows by recursively applying the same procedure.

**Remark 4.** *Having proven the equivalence between kalman filtering/smoothing and unconstrained MHE it is clear that their estimates are globally optimal in least-squares sense. This again emphasizes the remarkable property of the Kalman filter of being a recursive solution to the optimal estimation problem. This, as mentioned before, is attributed to the two special properties; linear model and quadratic objective.*

## 3.4. Analogy with Riccati methods for MPC

As discussed in Section 1.8, batch estimation or MHE problems are essentially optimal control problems.

Consider the following unconstrained MPC problem.

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{u}} & \|\bar{x}_N + x_N\|^2_{\bar{Q}_N} + \sum_{k=0}^{N-1} \|\bar{x}_k + x_k\|^2_{\bar{Q}_k} + \|\bar{u}_k + u_k\|^2_{\bar{R}_k} \\ \text{s.t.} & x_{k+1} = f_k + A_k x_k + B_k u_k, \quad k = 0,\ldots,N-1, \\ & x_0 = \bar{x}_0, \quad k = 0,\ldots,N-1, \end{array} \tag{3.46}$$

with as optimization variables the state sequence $\mathbf{x} = \{x_0,\ldots,x_N\}$ and the input sequence $\mathbf{u} = \{u_0,\ldots,u_{N-1}\}$. The sequences $\{\bar{x}_0,\ldots,\bar{x}_N\}$ and $\{\bar{u}_0,\ldots,\bar{u}_{N-1}\}$ describe nominal or reference trajectories for the states and inputs respectively.

This optimization problem leads to the following KKT system

$$\begin{bmatrix} 0 & -I & & & & & & \\ -I & \Pi_0 & \Theta_0^{\mathrm{T}} & 0 & & & & \\ & \Theta_0 & 0 & -I & & & & \\ & 0 & -I & \Pi_1 & & & & \\ & & & & \ddots & & & \\ & & & & & \Pi_{N-1} & \Theta_{N-1}^{\mathrm{T}} & 0 \\ & & & & & \Theta_{N-1} & 0 & -I \\ & & & & & 0 & -I & \Pi_N \end{bmatrix} \cdot \begin{bmatrix} \lambda_0 \\ z_0 \\ \lambda_1 \\ \vdots \\ z_{N-1} \\ \lambda_{N-1} \\ z_N \end{bmatrix} = \begin{bmatrix} \bar{x}_0 \\ r_{d,0} \\ r_{p,1} \\ \vdots \\ r_{d,N-1} \\ r_{p,N-1} \\ r_{d,N} \end{bmatrix}. \tag{3.47}$$

where we defined

$$\Pi_k = \begin{bmatrix} \bar{Q}_k & 0 \\ 0 & \bar{R}_k \end{bmatrix}, \quad r_{d,k} = \begin{bmatrix} -\bar{Q}_k \bar{x}_k \\ -\bar{R}_k \bar{u}_k \end{bmatrix}, \quad 0 \le k \le N-1$$

$$\Pi_N = \begin{bmatrix} \bar{Q}_N \end{bmatrix}, \qquad r_{d,N} = \begin{bmatrix} -\bar{Q}_N \bar{x}_N \end{bmatrix},$$

$$\Theta_k = \begin{bmatrix} A_k & B_k \end{bmatrix}, \quad r_{p,k} = \begin{bmatrix} -f_k \end{bmatrix}, \qquad 0 \le k \le N-1$$

Note that in the literature the initial state is often eliminated in the KKT system representation, see *e.g.* [150] or [192], since it is fixed anyway. We preferred to keep the equations for $x_0$ qnd $\lambda_0$ in, because it facilitates the comparison with the KKT systems for MPC (3.47) and MHE (3.3).

The KKT system (3.47) can be factorized using the following (combined) Riccati recursion

$$V_k \quad = \quad A_k^{\mathrm{T}} V_{k+1} A_k - A_k^{\mathrm{T}} V_{k+1} B_k (\bar{R}_k + B_k^{\mathrm{T}} V_{k+1} B_k)^{-1} B_k^{\mathrm{T}} V_{k+1} A_k + \bar{Q}_k \qquad (3.48)$$

which runs backwards starting from the terminal weight $V_N = \bar{Q}_N$. Hence, the MPC KKT system can be solved by a backwards vector and Riccati recursion followed by a forward vector recursion. See the references [150, 188] for more details.

By comparing (3.48) to the Kalman filter recursion for factorizing the MHE KKT system, see (2.75) or Appendix A, we find the well known *duality* relations (see also the discussion in Section 1.8) which are summarized in Table 3.1.

**Table 3.1.** Conversion table between LQR and Kalman Riccati recursions.

| LQR | $k$ | $V$ | $A$ | $B$ | $Q$ | $\bar{R}$ | $Q_N$ |
|---|---|---|---|---|---|---|---|
| Kalman | $n-k$ | $P$ | $A^{\mathrm{T}}$ | $C^{\mathrm{T}}$ | $GQG^{\mathrm{T}}$ | $R$ | $P_{0+}$ |

## 3.5. Numerical examples

To illustrate the efficiency of the unconstrained MHE algorithms using normal or square-root Riccati recursions and their properties, *i.e.* linear scaling in the horizon length, we have implemented both Riccati based methods in C and applied them to random linear time-varying systems with dimensions $(n_x, n_w, n_y) = (5, 5, 2)$. The square-root version employs structured QR methods (see Appendix B).

Numerical results were obtained for randomly generated matrices sampled from the standard normal distribution[2] and averaged[3] over 100 repetitions. Computing times correspond to an Intel Core2-Duo processor at 2.13 GHz with 2 MB cache and 2 GB RAM, and using the compiler gcc version 4.4.5. The results are presented in Figure 3.4.

The linear scaling property is clear from the growing horizon case, while the computation times are bounded for the moving horizon implementation. For this system with dimensions $(n_x, n_w, n_y) = (5, 5, 2)$ the computation times are all below 0.5 ms for horizon lengths up to 50. For the MHE version with horizon 5 computation times are even below 0.06 ms. The square-root algorithm is slower by about a factor 1.5 compared to the normal algorithm. Both codes were equally optimized, *i.e.* unnecessary calculations are avoided, previously computed quantities are reused, multiplications

[2] Using the random number generator from GNU Scientific Library (GSL)

[3] The median was used rather than the mean in order to remove bursts originating from processes running in the background.

with triangular matrices are executed efficiently, etc. All calculations have been implemented in a self-contained code, which differs from our previous implementation (see [90]) where calls to BLAS and LAPACK were used for matrix-matrix and matrix-vector operations and other linear algebraic computations. We have noticed that our implementations are faster compared to the BLAS/LAPACK calls.

The increased computational complexity of the square-root is completely attributed to the QR factorizations involved, which motivates our efforts to develop structured QR methods. This increased computational burden is compensated by an increased accuracy and stability of the square-root version as demonstrated next. Furthermore, the square-root version is more memory efficient as it works with triangular covariances. This becomes apparent from Figure 3.5 where we ran into a memory problem. Since the normal algorithm is more memory intensive it causes the program to exceed the cache boundaries as the amount of data grows while this effect is delayed with the square-root algorithm.

In order to compare the accuracy of both methods, we applied them to random linear time-varying systems with stable system matrices, *i.e.* $A_k = \frac{c}{\lambda_{\max}(A_k)} A_k$ with $c < 1$, and with $n_w = n_x$, $G_k = I$ and $Q_k = I, \forall k$. For this case the covariance evolves to $P_\infty = I$ or $|S_\infty| = I$ and the error can be readily checked.

Application to random linear time-varying systems with dimensions $(n_x, n_w, n_y) = (2,2,2)$ and stability factor $c = 0.1$ averaged over 100 repetitions yields the results presented in Figure 3.6. Both the error in smoothed state vector $\|x^* - x\|_2$ divided by the horizon length and the error[4] in final covariance matrix $\|P_T - I\|_2$ or $\||S_T| - I\|_2$ with $T$ the horizon length are computed. For this stability factor convergence to the steady state covariance is obtained after only 1 or 2 time steps. It can be seen that the square-root algorithm is consistently more accurate in both the estimates and covariances. Note that the non-smoothness in the error covariance evolution is due to the time-variance of the system.

The difference in accuracy grows dramatically when larger systems are considered. In Figure3.7 the final covariance error is presented for systems with dimensions ($n_x$,$n_w$,$n_y$) = (5,5,5) and ($n_x$,$n_w$,$n_y$) = (10,10,5). The errors of the square-root algorithm are two orders in magnitude smaller compared to those of the normal algorithm.

## 3.6. Conclusions

TIn this chapter the problems of state estimation were presented in an optimization framework and connections with Kalman filtering and MPC were discussed. It was shown how the KKT system can be decomposed by $LU$ and $LDL^{\mathrm{T}}$ factorizations.

[4]The Frobenius norm was used for the covariance matrix error

These decompositions yield normal and square-root Riccati based solution methods for the MHE problem. The methods fully exploit the structure inherent in the MHE problem. The square-root version further exploits the symmetry in the system and employs orthogonalization methods yielding increased numerical robustness. Structured QR methods were proposed for the square-root Riccati based method to reduce its increased computation time.

The methods have been implemented in C in a fully self-contained and optimized code and numerical results were presented illustrating the linear scaling with horizon length. Computation times below 0.5 ms are obtained for order 5 time-varying systems and horizon lengths of up to 50. For the moving horizon version computation times below 0.06 ms are possible. The simulations further demonstrate that the normal Riccati algorithm is faster than the square-root Riccati algorithm by about a factor 1.5. This increased computational burden of the square-root version is compensated by a substantial increase in numerical accuracy and robustness as demonstrated by numerical examples. The developed Riccati based solution methods form the basis for constrained and nonlinear MHE methods presented in the following chapters.

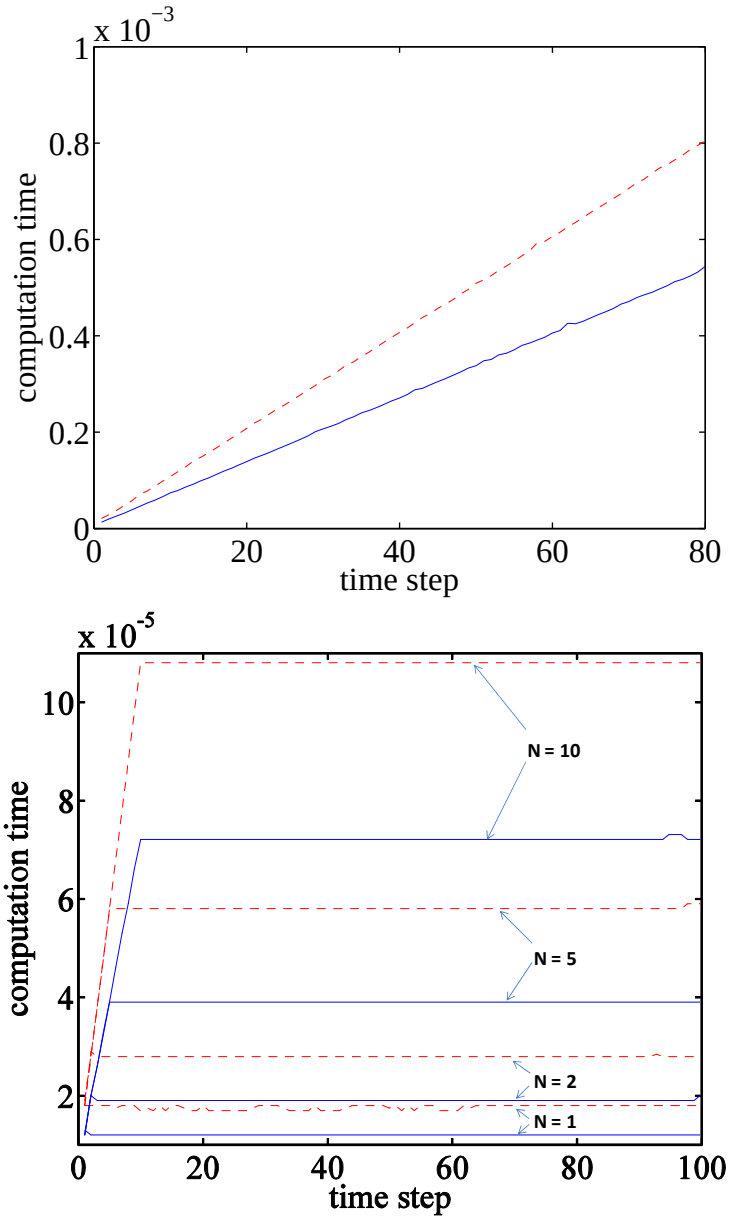



**Figure 3.4.** Computation times in seconds for random linear time-varying systems and estimation with growing horizon (**top**) and with moving horizon of lengths 1, 2, 5 and 10 (**bottom**). The results for the normal Riccati based algorithm are depicted in solid line; the results for the square-root Riccati based algorithm in dashed line.

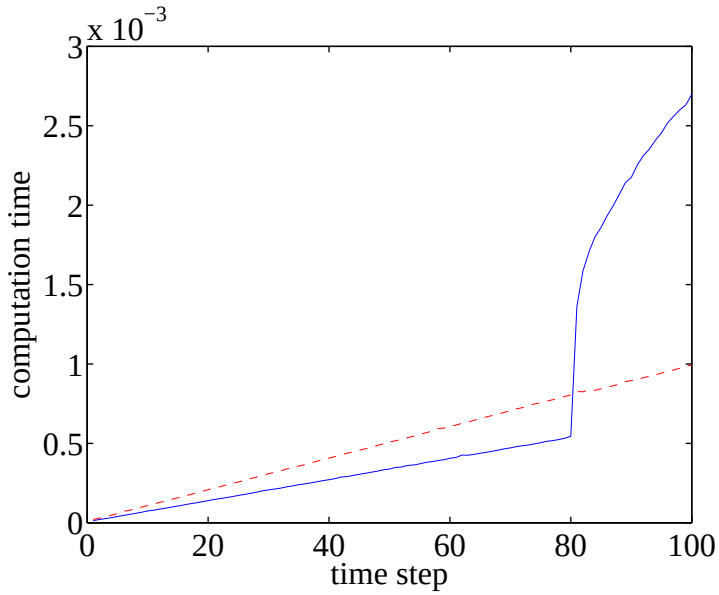



**Figure 3.5.** Computation times in seconds for random linear time-varying systems and estimation with growing horizon. The results for the normal Riccati based algorithm are depicted in solid line; the results for the square-root Riccati based algorithm in dashed line. The normal algorithm is more memory intensive which causes the program to exceed the cache boundaries as the amount of data grows. The square-root algorithm is clearly more memory efficient.

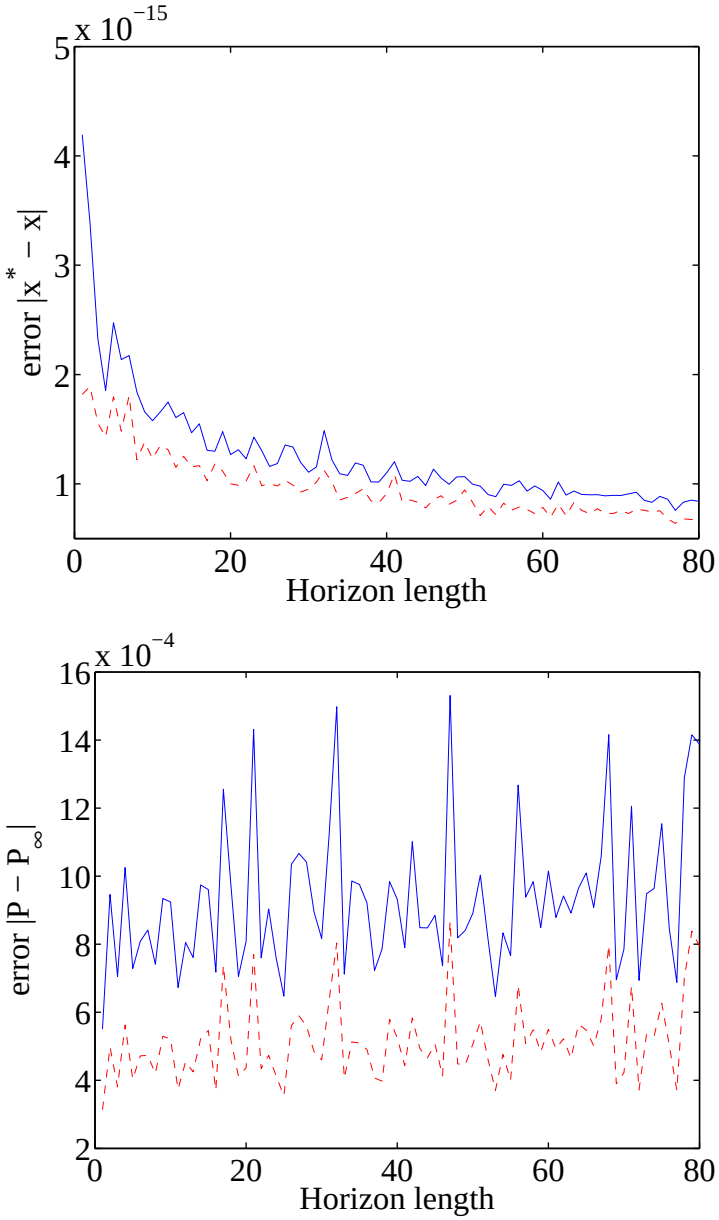



**Figure 3.6.** Errors for the normal Riccati based algorithm (solid line) and square-root Riccati based algorithm (dashed line) for the growing horizon case and system dimensions $(n_x, n_w, n_y) = (2,2,2)$. **Top panel:** error in smoothed state vecor $\|x^* - x\|_2$ divided by $T$. **Bottom panel:** error in final state covariance $\|P_T - I\|_2$ or $\|S_T - I\|_2$ with $T$ the horizon length. It is seen that the square-root algorithm is consistently more accurate.

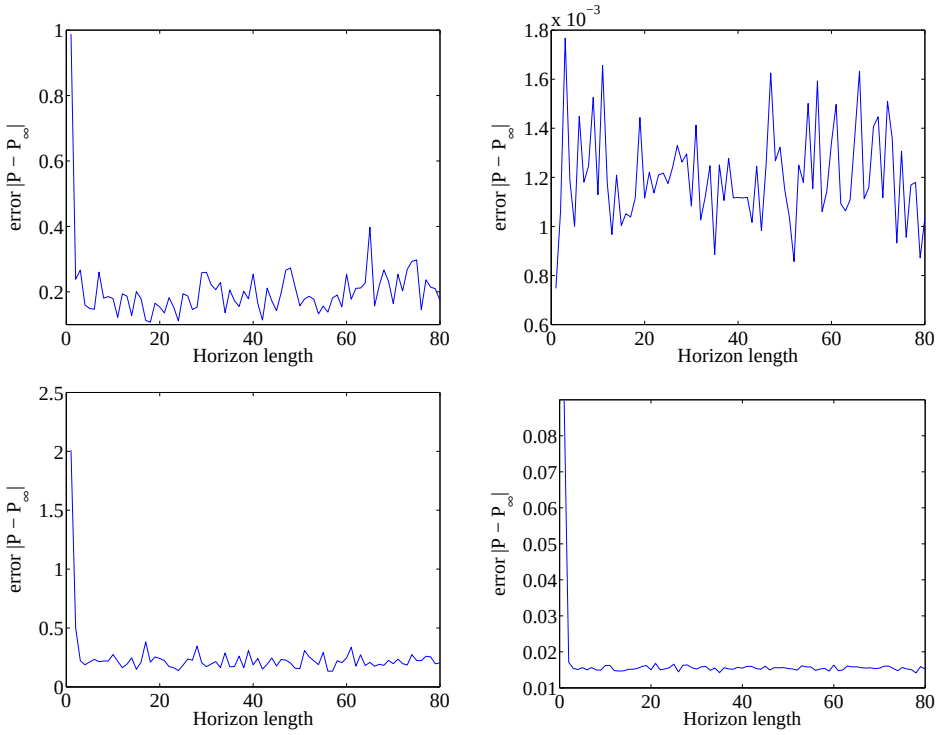



**Figure 3.7.** Error in final state covariance $\|P_T - I\|_2$ or $\||S_T| - I\|_2$ with $T$ the horizon length for the normal Riccati based algorithm and square-root Riccati based algorithm for growing horizon length. **Top row:** system dimensions $(n_x, n_w, n_y) = (5,5,5)$. **Bottom row:** system dimensions $(n_x, n_w, n_y) = (10,10,5)$. **Left column:** normal Riccati based algorithm. **Right column:** square-root Riccati based algorithm. The errors of the square-root algorithm are two orders in magnitude smaller.

CHAPTER 4

# Interior-point methods for MHE

*In this chapter the solution of constrained MHE problems by an infeasible start primal barrier method is addressed. Modifications of the normal and square-root Riccati based methods presented in the previous chapter are proposed yielding MHE methods which scale linearly with both the horizon length and the number of interior-point iterations. Typically a limited number of interior-point iterations is sufficient to achieve convergence as is demonstrated by numerical examples. It is shown that a good initialisation strategy improves the performance of the algorithms. The primal barrier method experiences a logarithmically growing condition number near an active constraint, which motivates the use of square-root methods because of their numerical accuracy.*

## 4.1. Introduction

Riccati based solution methods for optimal control problems have been investigated before in the literature [10, 17, 44, 75, 80, 87, 88, 108, 150, 167, 188, 192]. Almost all of these publications focus on the control problem, i.e. MPC. The state estimation problem, i.e. MHE, is closely related to the control problem and has largely the same optimal control structure. However, the main differences with MPC are that the initial value is free and typically the *control vector*, called state disturbance in MHE, has about the same dimension as the state vector while in the MPC problem the control vector is typically much smaller allowing an effective reduction in problem dimensions by eleminating the state variables.

The interior-point method presented here closely resembles the one for MPC problems presented in [192] in the sense that an infeasible start primal barrier method is used.

However, the method presented in [192] solves the KKT system by a block-Cholesky decomposition which is only applicable to strictly positive definite Hessians. In MHE problems the Hessian is typically positive semidefinite. Furthermore, the algorithms presented here employ Riccati based solution methods for the KKT system along the lines of the methods presented in Chapter 3.

Riccati based methods for MHE problems have been proposed before, *e.g.* by Tenny et al [173] and Jorgensen et al [108]. In these works normal Kalman filter recursions were proposed to factorize the KKT system. As mentioned in previous chapters, a well known problem with the normal Kalman filter recursions is its numerical stability [6, 189]. The recursions depend on the square of the condition number of the matrices involved. When the process is well known, *i.e.* small covariance $Q_k$, it is often observed that rounding errors render the state covariance matrix $P_k$ invalid: negative diagonal entries or otherwise loss of positive semi-definiteness. The stability issue is even more important when these recursions are applied in an interior-point method, since the barrier term can have extremely large impact on the conditioning of the matrices to be propagated, as shown in this chapter.

To avoid these problems, square-root Riccati based methods are proposed in this chapter and are demonstrated to provide reliable estimates in an interior-point context. Extracts of this chapter were published in [90].

## 4.2. Overview of interior-point methods for quadratic programming

The standard convex QP may be expressed as (see (2.49))

$$\begin{array}{ll} \min_z & \frac{1}{2}z^{\mathrm{T}}Hz + g^{\mathrm{T}}z \\ \text{s.t.} & Cz = d, \\ & Pz \leq h, \end{array} \tag{4.1}$$

with Hessian $H \in \mathbf{R}^{n\times n}$ symmetric positive semidefinite, gradient $g \in \mathbf{R}^n$, equality constraints $C \in \mathbf{R}^{m\times n}$, $d \in \mathbf{R}^m$ and inequality constraints $P \in \mathbf{R}^{p\times n}$, $h \in \mathbf{R}^p$.

### 4.2.1. Primal barrier method

We will use a primal barrier method to solve the QP [29, 136]. The inequality constraints in the QP (4.1) are replaced with a barrier term in the objective, to get the approximate problem

$$\begin{array}{ll} \min_z & z^T Hz + g^T z + \kappa\phi(z) \\ \text{s.t.} & Cz = b, \end{array} \tag{4.2}$$

where $\kappa > 0$ is a barrier parameter, and $\phi$ is the log barrier associated with the inequality constraints, defined as

$$\phi(z) = \sum_{i=1}^{p} -\log(h_i - p_i^T z),$$

where $p_i^T$ denotes the $i$th row of $P$. The problem (4.2) is a convex optimization problem with smooth objective and linear equality constraints, and can be solved by Newton's method.

In a basic primal barrier method, a sequence of problems of the form (4.2) is solved, using Newton's method starting from the previously computed point, for a decreasing sequence of values of $\kappa$. As $\kappa$ approaches zero, the solution of (4.2) converges to a solution of the QP (4.1).

Let us introduce a dual variable $\nu \in \mathbf{R}^m$ associated with the equality constraint $Cz = b$. The optimality conditions for (4.2) are then

$$\begin{aligned} r_d &= Hz + g + \kappa P^T d + C^T \nu = 0 \\ r_p &= Cz - b = 0 \end{aligned} \tag{4.3}$$

with $r_d$ the dual and $r_p$ the primal residual and where $d_i = 1/(h_i - p_i^T z)$. The term $\kappa P^T d$ is the gradient of $\kappa\phi(z)$. We also have the implicit constraint $Pz < h$. The stacked vector $r = [r_d^T \; r_p^T]^T$ is called the residual. The optimality conditions for (4.2) can then be expressed as $r = 0$.

The algorithm is initialized with a $z^0$ point that strictly satisfies the inequality constraints but need not satisfy the equality constraints. An arbitrary initial value can be used for $\nu^0$. This is called an infeasible-start Newton method.

An approximate $z$ (with $Pz < h$) and $\nu$ is computed at each step. The optimality conditions (4.16) are linearized and primal and dual steps $\Delta z$, $\Delta\nu$ are calculated for which $z + \Delta z$, $\nu + \Delta\nu$ yield zero residuals in the linearized approximation.

The primal and dual search directions $\Delta z$ and $\Delta\nu$ can be computed by solving the following KKT system

$$\begin{bmatrix} H + \kappa P^T \mathbf{diag}(d)^2 P & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta\nu \end{bmatrix} = - \begin{bmatrix} r_d \\ r_p \end{bmatrix}. \tag{4.4}$$

Here the term $\kappa P^T \mathbf{diag}(d)^2 P$ is the Hessian of the barrier $\kappa\phi(z)$ and

Having obtained this Newton step, a feasibility search and a backtracking line search on the norm of the residual $r$ (see, e.g., [29]) are performed. Finally, the primal and dual variables are updated: $z := z + s\Delta z$ and $\nu := \nu + s\Delta\nu$. This procedure is repeated until the norm of the residual is below an acceptable threshold.

It can be shown that primal feasibility (i.e., $Cz = b$) is achieved in a finite number of steps, assuming the problem (4.2) is strictly feasible. The primal and dual variables $z$ and $v$ will converge to an optimal point. A typical method is to reduce $\kappa$ by a factor $\tau$ each time a solution of (4.2) is computed (within some accuracy). The total number of Newton steps to convergence is bounded and depends on the number of constraints, the initial point $z^0$ (and $v^0$), the initial barrier parameter $\kappa_0$ and the reduction factor $\tau$ [29]. However, it is observed that an accurate solution of the original QP can be obtained with a limited number of Newton steps, far less than the theoretical bound.

### 4.2.2. Primal-dual interior-point methods

In a standard primal-dual interior-point method the primal and dual variables are obtained from the solution of two similar linear systems with different right hand side corresponding to an affine scaling step and a centering step (or centering-corrector if this step is combined *i.e.* with the popular Mehrotra's corrector step), see [150]. The search directions are similar but not identical to those of the primal barrier method. A primal-dual method typically yields more aggressive step sizes (close to the boundaries). In practice the primal-dual methods are often faster and more reliable. It can be shown that the linear systems to be solved for obtaining the search directions are structurally identical to those of the primal barrier method, see [29, 150]. For this reason the interior-point methods for MHE have been developed for the primal barrier method and can be extended to a primal-dual method.

## 4.3. Structure-exploiting interior-point methods for MHE

In this section structure-exploiting primal barrier interior-point methods for solving linear constrained MHE problems are described. Consider the following general linear constrained MHE problem

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{w}} & \|\bar{x}_0 + x_0 - \hat{x}_0\|^2_{P_0^{-1}} + \sum_{k=0}^{N-1} \|\bar{w}_k + w_k\|^2_{Q_k^{-1}} + \|h_k + C_k x_k + H_k w_k\|^2_{R_k^{-1}} + \|h_N + C_N x_N\|^2_{R_N^{-1}} \\ & x_{k+1} = f_k + A_k x_k + G_k w_k, \quad k = 0,\ldots,N-1, \\ \text{s.t.} & T_k^x x_k + T_k^w w_k \leq t_k \qquad k = 0,\ldots,N-1 \\ & T_N^x x_N \leq t_N \end{array} \tag{4.5}$$

with $\mathbf{x} = \{x_0,\ldots,x_N\}$ the unknown state sequence and $\mathbf{w} = \{w_0,\ldots,w_{N-1}\}$ the unknown process disturbances.

Let us first define an overall optimization variable

$$z = (x_0, w_0, x_1 \ldots, w_{N-1}, x_N) \in \mathbf{R}^{(N+1)n+Nm}.$$

Then the QP may be expressed as

$$\begin{array}{ll} \min_z & \frac{1}{2} z^T H z + g^T z \\ \text{s.t.} & Cz = b \\ & Pz \leq h \end{array} \tag{4.6}$$

with

$$\begin{aligned} H &= \begin{bmatrix} S_0^{-1} S_0^{-T} + C_0^T C_0 & C_0^T H_0 & & \\ H_0^T C_0 & I_m + H_0^T H_0 & & \\ & & \ddots & \\ & & & C_N^T C_N \end{bmatrix}, \\ g &= \begin{bmatrix} -S_0^{-1} S_0^{-T} \bar{x} + C_0^T h_0 \\ H_0^T h_0 \\ \vdots \\ H_{N-1}^T h_{N-1} \\ C_N^T h_N \end{bmatrix}, \quad b = - \begin{bmatrix} f_0 \\ \vdots \\ f_{N-1} \end{bmatrix}, \\ C &= \begin{bmatrix} A_0 & G_0 & -I_n & & & \\ & & \ddots & & & \\ & & & A_{N-1} & G_{N-1} & -I_n \end{bmatrix}, \\ P &= \begin{bmatrix} T_0^x & T_0^w & & & & \\ & \ddots & & & & \\ & & T_{N-1}^x & T_{N-1}^w & & \\ & & & & T_N^x \end{bmatrix}, \quad h = \begin{bmatrix} t_0 \\ \vdots \\ t_{N-1} \\ t_N \end{bmatrix}, \end{aligned} \tag{4.7}$$

where $I_n$ denotes the unit matrix of dimension $n$.

The Newton step $\Delta z$ and $\Delta \nu$ can be computed by solving the KKT system (4.4) for this specific problem, which can be done efficiently using Riccati based solution methods along the lines of the previous chapter. This is described in detail in the next section for the different use cases: mixed linear constraints, state constraints, disturbance constraints and bound constraints.

### 4.3.1. Computing the Newton step

By rearranging the KKT system, the block diagonal structure of the KKT matrix is revealed (see Section 3.2). Let us write the KKT system as $M\xi = r$, with

$$M = \begin{bmatrix} \Phi_0 & \Gamma_0^{\mathrm{T}} & 0 & & & & \\ \Gamma_0 & 0 & -\Upsilon^{\mathrm{T}} & & & & \\ 0 & -\Upsilon & \Phi_1 & & & & \\ & & & \ddots & & & \\ & & & & \Phi_{N-1} & \Gamma_{N-1}^{\mathrm{T}} & 0 \\ & & & & \Gamma_{N-1} & 0 & -I \\ & & & & 0 & -I & \Phi_N \end{bmatrix}, \quad \xi = \begin{bmatrix} \Delta z_0 \\ \Delta v_0 \\ \vdots \\ \Delta z_{N-1} \\ \Delta v_{N-1} \\ \Delta z_N \end{bmatrix}, \quad r = \begin{bmatrix} r_{d,0} \\ r_{p,0} \\ \vdots \\ r_{d,N-1} \\ r_{p,N-1} \\ r_{d,N} \end{bmatrix}, \tag{4.8}$$

where $r_{d,k}$ (resp. $r_{d,N}$) denotes the dual residual associated with $\Delta z_k = (\Delta x_k^{\mathrm{T}}, \Delta w_k^{\mathrm{T}})^{\mathrm{T}}$ (resp. $\Delta z_N = \Delta x_N$ ), and $r_{p,k}$ denotes the primal residual associated with $v_k$.

We will now consider different types of constraints and present Riccati based solution methods for the different cases.

### Mixed linear constraints

*Mixed linear constraints* are commonly encountered in optimal control problems and are given by

$$\begin{array}{ll} T_k^x x_k + T_k^w w_k & \leq t_k \quad k = 0, \ldots, N-1 \\ T_N^x x_N & \leq t_N \end{array} \tag{4.9}$$

For this type of constraints the Hessian of the barrier function has block diagonal structure. It can be expressed as

$$P^T \mathbf{diag}(d)^2 P = \begin{bmatrix} M_0^T M_0 & M_0^T L_0 & & \\ L_0^T M_0 & L_0^T L_0 & & \\ & & \ddots & \\ & & & M_N^T M_N \end{bmatrix}, \tag{4.10}$$

with

$$\begin{array}{l} M_k = \mathbf{diag}(1/(t_k - T_k^x x_k - T_k^w w_k)) T_k^x, \quad 0 \leq k \leq N-1 \\ L_k = \mathbf{diag}(1/(t_k - T_k^x x_k - T_k^w w_k)) T_k^w, \quad 0 \leq k \leq N-1 \\ M_N = \mathbf{diag}(1/(t_N - T_N^x x_N)) T_N^x. \end{array} \tag{4.11}$$

The gradient of the barrier function is given by

$$g_p = \begin{bmatrix} M_0^{\mathrm{T}} \\ L_0^{\mathrm{T}} \\ \vdots \\ M_k^{\mathrm{T}} \\ L_k^{\mathrm{T}} \\ \vdots \\ M_N^{\mathrm{T}} \end{bmatrix} \mathbf{1} \tag{4.12}$$

where **1** represents a column vector of all ones with appropriate length.

Plugging this into the KKT system (4.8) yields

$$\begin{aligned}
\Phi_0 &= \begin{bmatrix} P_0^{-1} + C_0^{\mathrm{T}} R_0^{-1} C_0 + \kappa M_0^{\mathrm{T}} M_0 & C_0^{\mathrm{T}} R_0^{-1} H_0 + \kappa M_0^{\mathrm{T}} L_0 \\ H_0^{\mathrm{T}} R_0^{-1} C_0 + \kappa L_0^{\mathrm{T}} M_0 & Q_0^{-1} + H_0^{\mathrm{T}} R_0^{-1} H_0 + \kappa L_0^{\mathrm{T}} L_0 \end{bmatrix} \\
\Phi_k &= \begin{bmatrix} C_k^{\mathrm{T}} R_k^{-1} C_k + \kappa M_k^{\mathrm{T}} M_k & C_k^{\mathrm{T}} R_k^{-1} H_k + \kappa M_k^{\mathrm{T}} L_k \\ H_k^{\mathrm{T}} R_k^{-1} C_k + \kappa L_k^{\mathrm{T}} M_k & Q_k^{-1} + H_k^{\mathrm{T}} R_k^{-1} H_k + \kappa L_k^{\mathrm{T}} L_k \end{bmatrix}, \qquad 1 \le k \le N-1 \\
\Phi_N &= \begin{bmatrix} C_N^{\mathrm{T}} R_N^{-1} C_N + \kappa M_N^{\mathrm{T}} M_N \end{bmatrix} \\
r_{d,0} &= \begin{bmatrix} P_0^{-1}(\hat{x}_0 - \bar{x}_0) - C_0^{\mathrm{T}} R_0^{-1} h_0 - (P_0^{-1} + C_0^{\mathrm{T}} R_0^{-1} C_0) x_0 - A_0^{\mathrm{T}} v_0 - \kappa g_{p,x_0} \\ -Q_0^{-1} \bar{w}_0 - H_0^{\mathrm{T}} R_0^{-1} h_0 - (Q_0^{-1} + H_0^{\mathrm{T}} R_0^{-1} H_0) w_0 - G_0^{\mathrm{T}} v_0 + \kappa g_{p,w_0} \end{bmatrix} \\
r_{d,k} &= \begin{bmatrix} -C_k^{\mathrm{T}} R_k^{-1} h_k - C_k^{\mathrm{T}} R_k^{-1} C_k x_k - (-v_{k-1} + A_k^{\mathrm{T}} v_k) - \kappa g_{p,x_k} \\ -Q_k^{-1} \bar{w}_k - H_k^{\mathrm{T}} R_k^{-1} h_k - (Q_k^{-1} + H_k^{\mathrm{T}} R_k^{-1} H_k) w_k - G_k^{\mathrm{T}} v_k + \kappa g_{p,w_k} \end{bmatrix}, \qquad 1 \le k \le N-1 \\
r_{d,N} &= \begin{bmatrix} -C_N^{\mathrm{T}} R_N^{-1} h_N - C_N^{\mathrm{T}} R_N^{-1} C_N x_N + v_{N-1} - \kappa g_{p,x_N} \end{bmatrix} \\
r_{p,k} &= \begin{bmatrix} f_k + A_k x_k + G_k w_k - x_{k+1} \end{bmatrix}, \qquad 0 \le k \le N-1 \\
\Gamma_k &= \begin{bmatrix} A_k & G_k \end{bmatrix}, \qquad 0 \le k \le N-1 \\
\Upsilon &= \begin{bmatrix} I_n \\ 0 \end{bmatrix}
\end{aligned}$$

where $g_{p,x_k}$ (resp. $g_{p,w_k}$) denotes the barrier gradient associated with $\Delta x_k$ (resp. $\Delta w_k$).

**Riccati based IP method** A Riccati based solution method has been developed for constrained MHE using interior-point methods. The constraints enter the Riccati recursion in the measurement update step, which is intuitive since active constraints can be interpreted as measurements with very large weight (or, equivalently, with very small covariance).

The normal Riccati recursion is modified as follows (compare with Algorithm 1).

Set $\Sigma_k = \begin{bmatrix} P_k & \\ & Q_k \end{bmatrix}$ and $D_k = \begin{bmatrix} C_k & H_k \\ \sqrt{\kappa}M_k & \sqrt{\kappa}L_k \end{bmatrix}$.

Calculate the covariance update:

$$\Sigma_{k+} = \left(\Sigma_k^{-1} + D_k^{\mathrm{T}} R_k^{-1} D_k\right)^{-1} = \Sigma_k - \Sigma_k D_k^{\mathrm{T}} \left( \begin{bmatrix} R_k & \\ & I_{ni_k} \end{bmatrix} + D_k \Sigma_k D_k^{\mathrm{T}} \right)^{-1} D_k \Sigma_k \tag{4.13}$$

with $ni_k$ the number of inequality constraints for time step $k$.

The measurement update of the square-root recursion modifies similarly. Let $T_k = \begin{bmatrix} S_k & \\ & W_k \end{bmatrix}$ (note that $\Sigma_k = T_k^{\mathrm{T}} T_k$)
and $D_k = \begin{bmatrix} C_k & H_k \\ \sqrt{\kappa}M_k & \sqrt{\kappa}L_k \end{bmatrix}$

$$\begin{bmatrix} T_k D_k^{\mathrm{T}} & T_k \\ \begin{bmatrix} V_k & \\ & I_{ni_k} \end{bmatrix} & \end{bmatrix} = \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix} \tag{4.14}$$

with $\tilde{R}_k^e = \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1/2}$ and $\tilde{K}_k = \Sigma_k D_k^{\mathrm{T}} \tilde{R}_k^e$.

**Remark 5.** *It can be seen that in case of mixed linear inequalities, the general form of Riccati based solution is necessary whether H is zero or not. This is one of the motivations for developing the algorithms for this general case in Chapter 3 and throughout this thesis.*

Note that in comparison with the previous chapter, the block rows of the matrix to be factorized have been interchanged since this is how the recursions have been implemented (see Appendix B for a discussion on structured *QR* factorization). These row interchanges do not change the *R* factor.

A solution method for interior-point MHE using square-root recursions is now presented.

**Lemma 4.3.1.** *If the KKT matrix M (4.8) is full rank (see Lemma 3.2.4 for sufficient conditions), then it is symmetric indefinite and can be factorized by an indefinite Cholesky decomposition $M = LDL^T$ with*

$$L = \begin{bmatrix} T_{0+}^{-1} & & & & & & & \\ \Gamma_0 T_{0+}^T & S_1^T & & & & & & \\ 0 & -\Upsilon S_1^{-1} & T_{1+}^{-1} & & & & & \\ & & & \ddots & & & & \\ & & & & T_{N-1+}^{-1} & & & \\ & & & & \Gamma_{N-1} T_{N-1+}^T & S_N^T & & \\ & & & & 0 & -\Upsilon S_N^{-1} & T_{N+}^{-1} \end{bmatrix},$$

$$D = \begin{bmatrix} I_{n+m} & & & & & & \\ & -I_n & & & & & \\ & & I_{n+m} & & & & \\ & & & \ddots & & & \\ & & & & I_{n+m} & & \\ & & & & & -I_n & \\ & & & & & & I_n \end{bmatrix},$$

*which can be recursively computed by Algorithm 9.*

*Proof.* See proof of Lemma 3.2.5. □

**Algorithm 9.** *[Square-root Riccati recursion for interior-point MHE]*

1. *Initialization:* $S_0$
2. *For* $k = 0, \ldots, N-1$*:*

   a) *Measurement update step*

   *Let* $T_k = \begin{bmatrix} S_k & \\ & W_k \end{bmatrix}$ *(note that* $\Sigma_k = T_k^{\mathrm{T}} T_k$*)*

   *and* $D_k = \begin{bmatrix} C_k & H_k \\ \sqrt{\kappa} M_k & \sqrt{\kappa} L_k \end{bmatrix}$

   *Compute QR-factorization:*

$$\begin{bmatrix} T_k D_k^{\mathrm{T}} & T_k \\ \begin{bmatrix} V_k & \\ & I_{ni_k} \end{bmatrix} & \end{bmatrix} = \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}$$

   *with* $\tilde{R}_k^e = \left(R_k + D_k \Sigma_k D_k^{\mathrm{T}}\right)^{-1/2}$ *and* $\tilde{K}_k = \Sigma_k D_k^{\mathrm{T}} \tilde{R}_k^e$.

*b) Model forwarding step*

*Let* $\Gamma_k = \begin{bmatrix} A_k & G_k \end{bmatrix}$

*Compute QR-factorization:*

$$T_{k+}\Gamma_k^T = \begin{bmatrix} \hat{Q}_{k+1} & \check{Q}_{k+1} \end{bmatrix} \begin{bmatrix} \hat{S}_{k+1} \\ 0 \end{bmatrix}$$

*endfor.*

3. *Final time step:*

*a) Measurement update step*

*Let* $T_N = S_N$ *(note that* $\Sigma_N = T_N^{\mathrm{T}} T_N$*)*

*and* $D_N = \begin{bmatrix} C_N \\ \sqrt{\kappa} M_N \end{bmatrix}$

*Compute QR-factorization:*

$$\begin{bmatrix} T_N D_N^{\mathrm{T}} & T_N \\ \begin{bmatrix} V_N & \\ & I_{ni_N} \end{bmatrix} & \end{bmatrix} = \begin{bmatrix} \bar{Q}_N & \tilde{Q}_N \end{bmatrix} \begin{bmatrix} (\tilde{R}_N^e)^{-1} & \tilde{K}_N^{\mathrm{T}} \\ 0 & T_{N+} \end{bmatrix}$$

*with* $\tilde{R}_N^e = \left(R_N + D_N \Sigma_N D_N^{\mathrm{T}}\right)^{-1/2}$ *and* $\tilde{K}_N = \Sigma_N D_N^{\mathrm{T}} \tilde{R}_N^e$.

Note that the dimensions of the matrices resulting from the measurement update become $\tilde{R}_k^e \in \mathbf{R}^{(p+ni_k)\times(p+ni_k)}$ and $\tilde{K}_k \in \mathbf{R}^{(p+ni_k)\times(n+m)}$.

**Theorem 4.3.2.** (KKT solution using modified $LDL^{\mathrm{T}}$ factor-solve method employing square-roots) *The KKT system* $M\xi = r$ *with* $M$,$\xi$ *and* $r$ *as defined before (4.8) can be solved by performing the following steps*

1. *Factorize* $M = LDL^{\mathrm{T}}$ *according to Lemma 4.3.1 using the Riccati recursion of Algorithm 9,*
2. *Forward vector solve using Algorithm 10,*
3. *Backward vector solve using Algorithm 11.*

*Proof.* See proof of Theorem 3.2.7. □

The computation of the newton step in an interior-point method is preceded by a calculation of barrier Hessian and gradient and primal and dual residuals

$$\begin{aligned} r_d &= Hz + g + \kappa P^T d + C^T \nu = 0 \\ r_p &= Cz - b = 0 \end{aligned} \tag{4.15}$$

Hence the term $C^T v$ with

$$C = \begin{bmatrix} A_0 & G_0 & -I_n & & & \\ & & \ddots & & & \\ & & & A_{N-1} & G_{N-1} & -I_n \end{bmatrix} \tag{4.16}$$

has been computed and can be used in the following recursions. For notational convenience we write $c_k$ as the component of $C^T v$ corresponding to time step $k$, that is

$$\begin{aligned} c_0 &= \Gamma_0^{\mathrm{T}} v_0 \\ c_k &= \Gamma_k^{\mathrm{T}} v_k - \begin{bmatrix} v_{k-1} \\ 0 \end{bmatrix} \\ c_N &= -v_N \end{aligned} \tag{4.17}$$

The following square-root forward and backward vector recursions can be derived for the IP MHE case

**Algorithm 10.** *[Square-root forward vector recursion for interior-point MHE]*

1. *Initialization:*

$$d_0 = \begin{bmatrix} \hat{x}_0 - \bar{x}_0 \\ -\bar{w}_0 \end{bmatrix} - z_0$$

2. *For $k = 0, \ldots, N-1$:*

$$\begin{aligned} g_k &= \begin{bmatrix} h_k + C_k x_k + H_k w_k \\ \sqrt{\kappa} e_{ni_k} \end{bmatrix} \\ \Delta z_k^{'} &= d_k - \tilde{K}_k (\tilde{R}_k^e)^{\mathrm{T}} (D_k d_k + g_k) - T_{k+}^{\mathrm{T}} T_{k+} c_k \\ \hat{x}_{k+1} &= f_k + \Gamma_k (\Delta z_k^{'} + z_k) \\ d_{k+1} &= \begin{bmatrix} \hat{x}_{k+1} \\ -\bar{w}_{k+1} \end{bmatrix} - z_k \end{aligned}$$

   *endfor.*

3. *Final time step:*

$$\begin{aligned} g_N &= \begin{bmatrix} h_N + C_N x_N \\ \sqrt{\kappa} e_{ni_N} \end{bmatrix} \\ \Delta z_N^{'} &= d_N - \tilde{K}_N (\tilde{R}_N^e)^{\mathrm{T}} (D_N d_N + g_N) - T_{N+}^{\mathrm{T}} T_{N+} c_N \end{aligned}$$

**Algorithm 11.** *[Square-root backward vector recursion for interior-point MHE]*

1. *Initialization ($k = N$):*

$$\begin{aligned} \Delta z_N &= \Delta z_N' \\ \eta_N &= c_N \\ \Delta v_{N-1} &= \eta_N + D_N^{\mathrm{T}} \tilde{R}_N^e (\tilde{R}_N^e)^{\mathrm{T}} \left[ D_N (d_N - T_N^{\mathrm{T}} T_N \eta_N) + h_N \right] \end{aligned}$$

2. *For $k = N-1, \ldots, 1$:*

$$\begin{aligned} \Delta z_k &= \Delta z_k' - T_{k+}^{\mathrm{T}} T_{k+} \Gamma_k^{\mathrm{T}} \lambda_k \\ \eta_k &= c_k + A_k^{\mathrm{T}} \Delta v_k \\ \Delta v_{k-1} &= \eta_k + D_k^{\mathrm{T}} \tilde{R}_k^e (\tilde{R}_k^e)^{\mathrm{T}} \left[ D_k (d_k - T_k^{\mathrm{T}} T_k \eta_k) + h_k \right] \end{aligned}$$

   *endfor.*

3. *Final step ($k = 0$):*

$$\Delta z_0 = \Delta z_0' - T_{0+}^{\mathrm{T}} T_{0+} \Gamma_0^{\mathrm{T}} \Delta v_0$$

with $\Delta z_k = \begin{bmatrix} \Delta x_k \\ \Delta w_k \end{bmatrix}$.

**Note that only the $R$ factors of the $QR$ factorizations in measurement and time update steps are required, hence $Q$-less $QR$ factorizations (see Appendix B) are used to improve computation speeds.**

### Separate state and disturbance constraints

In case the inequality constraints are not *mixed* and $H \equiv 0$ the updates of states and disturbances can be decoupled. General (linear) state constraints are given by

$$T_k^x x_k \leq t_k, \tag{4.18}$$

with $T_k^x \in \mathbf{R}^{ni_k^x \times n}$ and $t_k \in \mathbf{R}^{ni_k^x}$.

General (linear) disturbance constraints are given by

$$T_k^w w_k \leq s_k, \tag{4.19}$$

with $T_k^w \in \mathbf{R}^{ni_k^w \times n}$ and $s_k \in \mathbf{R}^{ni_k^w}$.

For this type of constraints the Hessian of the barrier function has block diagonal structure. It can be expressed as

$$P^T \,\mathbf{diag}(d)^2 P = \begin{bmatrix} M_0^T M_0 & & & \\ & L_0^T L_0 & & \\ & & \ddots & \\ & & & M_N^T M_N \end{bmatrix}. \tag{4.20}$$

with

$$\begin{aligned} M_k &= \mathbf{diag}(1/(t_k - T_k^x x_k))T_k^x, \quad 0 \le k \le N \\ L_k &= \mathbf{diag}(1/(s_k - T_k^w w_k))T_k^w, \quad 0 \le k \le N-1. \end{aligned} \tag{4.21}$$

The gradient of the barrier function is given by (4.12).

If state constraints are present and $H \equiv 0$ the measurement update step of the square-root algorithm involves covariances of order $n \times n$ and reduces to

$$\begin{aligned} D_k &= \begin{bmatrix} C_k \\ \sqrt{\kappa} M_k \end{bmatrix} \\ \begin{bmatrix} S_k D_k^{\mathrm{T}} & S_k \\ \begin{bmatrix} V_k \\ & I_{ni_k^x} \end{bmatrix} & \end{bmatrix} &= \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & S_{k+} \end{bmatrix} \end{aligned}$$

with $\tilde{R}_k^e \in \mathbf{R}^{(p+ni_k^x)\times(p+ni_k^x)}$ and $\tilde{K}_k \in \mathbf{R}^{(p+ni_k^x)\times n}$. The forward and backward vector recursions are modified accordingly.

In case no state constrains are present $D_k = C_k$ and the algorithms of Appendix A are valid and $\tilde{R}_k^e \in \mathbf{R}^{p\times p}$, $\tilde{K}_k \in \mathbf{R}^{p\times n}$.

If disturbance constraints are present, the disturbance vectors can be recursively updated separately from the state uppdates as follows

$$\begin{bmatrix} W_k\sqrt{\kappa}L_k^{\mathrm{T}} & W_k \\ I_{ni_k^w} & \end{bmatrix} = \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} (\tilde{R}_k^w)^{-1} & (\tilde{K}_k^w)^{\mathrm{T}} \\ 0 & W_{k+} \end{bmatrix}$$

with $\tilde{R}_k^w \in \mathbf{R}^{ni_k^w \times ni_k^w}$, $\tilde{K}_k^w \in \mathbf{R}^{ni_k^w \times m}$ and $W_{k+} \in \mathbf{R}^{m\times m}$.

$$\begin{aligned} d_k^w &= -\bar{w}_0 - w_0 \\ w_{k+} &= d_k^w - \tilde{K}_k^w (\tilde{R}_k^w)^{\mathrm{T}} L_k W_{k+}^{\mathrm{T}} W_{k+} \left(c_k^w + \kappa g_{p,w_k}\right) \end{aligned}$$

where $c_k^w$ is the part of $c_k$ corresponding to $w_k$.

### Bound constraints

In case of bound constraints the barrier gradient is computed as

$$\begin{aligned} d_k^+ &= 1/(x_k^{\max} - x_k) \\ d_k^- &= 1/(x_k^{\min} - x_k) \\ g_{p,x_k} &= d_k^+ - d_k^- \end{aligned} \tag{4.22}$$

Furthermore, the matrices $M_k$ and $L_k$ are sparse. They can be constructed in two ways

$$M_k = \begin{bmatrix} \operatorname{diag}(d_k^+) \\ \operatorname{diag}(d_k^-) \end{bmatrix}$$

or

$$M_k = \operatorname{diag}\left(\sqrt{(d_k^+)^2 + (d_k^-)^2}\right)$$

The latter version is more compact, while the first version is more consistent with the general state constraint case and yields slightly simpler vector updates.

For disturbance constraints, *i.e.* matrices $L_k$, both versions are possible as well. In this case consistency with the general constraint case is preserved for the both version and there is no difference for the vector update.

The sparsity of these barrier term matrices is exploited in the bound constrained versions of the algorithms.

**Fixed initial state** In order to be able to solve general optimal control problems (see Section 3.4), the possibility of a fixed initial state should be handled. When the initial state is exactly known, its covariance becomes singular. A singular $S_0$ does not present any problems to the algorithm for finding the Newton step, in fact any $S_k$ and $S_k+$ can be singular in the algorithm. However, in order to compute the dual residual, which is required by the interior-point method for performing a line search, the objective gradient and hessian are needed and they are infinite in case of singular covariance of the initial state. The problem can be circumvented through a minor modification of the KKT system. The following constraint is added to the MHE problem

$$\Delta x_0 = \bar{x} - x_0.$$

Let us denote by $v_0$ the dual variable associated to this constraint. Then in the KKT matrix an extra row and column are added to the top left, leading the the following modified KKT system:

$$\begin{bmatrix} 0 & I_n & 0 & \dots & 0 \\ I_n & & & & \\ 0 & & & & \\ \vdots & & M & & \\ 0 & & & & \end{bmatrix} \begin{bmatrix} \Delta v_0 \\ \xi \end{bmatrix} = \begin{bmatrix} \bar{x} - x_0 \\ r \end{bmatrix}.$$

The optimal vector $\xi$ is obtained as before by solving the original KKT system with singular $S_0$ and the dual variable $\Delta v_0$ is obtained from the modified KKT system.

## 4.4. Numerical examples

The interior-point method presented in this chapter has been implemented in C. The implementation is an extension of the square-root Riccati based codes for unconstrained MHE presented in Chapter 3 and employ structured QR methods (see Appendix B). All calculations have been implemented in a self-contained code, which differs from our previous implementation (see [90]) where calls to BLAS and LAPACK were used for matrix-matrix and matrix-vector operations and other linear algebraic computations. We have experienced that the self-contained code is faster compared to the BLAS/LAPACK calls.

Computing times correspond to an Intel Core2-Duo processor at 2.13 GHz with 2 MB cache and 2 GB RAM, and using the compiler gcc version 4.4.5.

### 4.4.1. Constrained linear system

Consider the following linear discrete-time system [149]

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} 0.99 & 0.2 \\ -0.1 & 0.3 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w_k, & k = 0,1,\dots \\ y_k &= \begin{bmatrix} 1 & -3 \end{bmatrix} x_k + v_k, & k = 0,1,\dots \end{aligned}$$

We assume $v_k$ to be zero-mean normally distributed random noise with variance 0.01, and $w_k = |z_k|$ with $z_k$ zero-mean normally distributed random noise with unit variance. We also assume $x_0$ to be normally distributed with zero mean and unit covariance. We formulate the constrained estimation problem with $Q = 1$, $R = 0.01$, $P_0 = I$, and $\bar{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

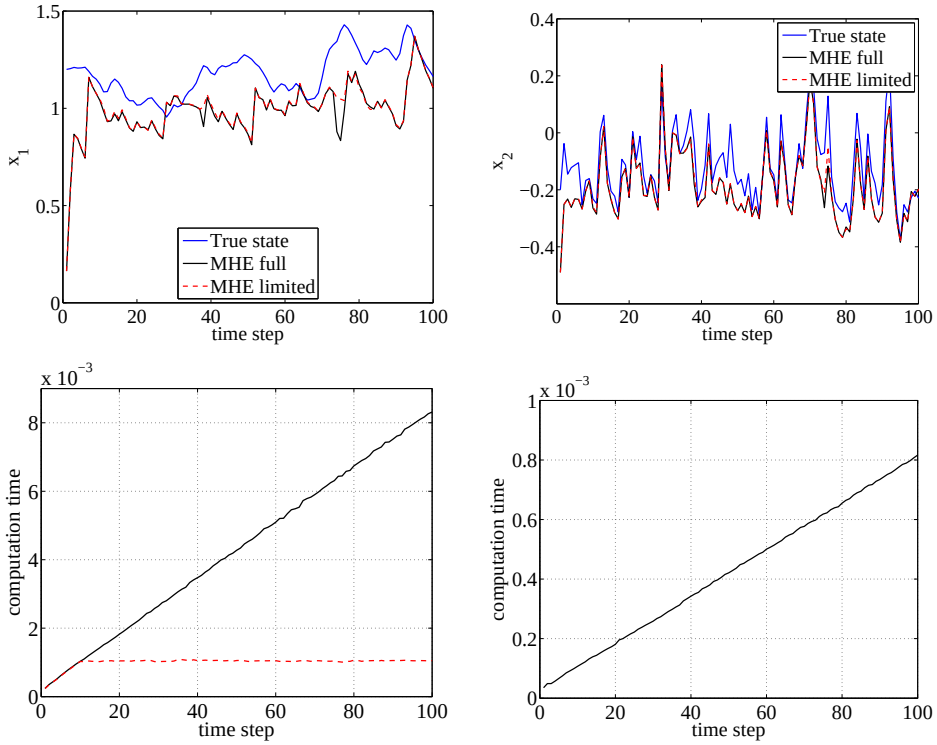



**Figure 4.1. Top left:** First state and MHE estimates for horizon length 10. **Top right:** Second state and MHE estimates for horizon length 10. When the number of interior-point iterations is limited to 10 (dashed red line), the performance is still almost identical to the full convergent estimator (solid black line). True states are depicted in solid blue line. **Bottom left:** Computation times for the algorithm with limited number of iterations (ten) and a growing horizon (solid black line) as well as horizon 10 (dashed red line). **Bottom right:** Computation times per iteration for the growing horizon case.

Simulations were performed for this example. Good performance was obtained with $\kappa_{\text{init}} = 1$ and decreasing factor $\tau = 0.2$. The results are shown in Figure 4.1. The top panels show the true states and MHE estimates for horizon length 10. When the number of interior-point iterations is limited to 10, the performance is still almost identical to the full convergent estimator. In the bottom left panel the computation times are shown for the algorithm with limited number of iterations (ten) and a growing horizon as well as horizon 10. In the bottom right panel the computation times per iteration are shown for the growing horizon case. As expected, the computation times scale linearly with the number of iterations for the interior-point method. From the figures it can be concluded that the moving horizon estimator with horizon length 10 and limited number of iterations can be run at 1 ms per time step or 1 kHz.

### 4.4.2. Waste water treatment process

Consider a waste water treatment process, shown in Figure 4.3. Waste water enters the equalizing tank which is designed to equalize concentration fluctuations of the incoming wastewater and to attenuate the effects of flow surges, *i.e.* due to a batch processing facility, on the treatment tanks. The contents of the equalization tank are continuously stirred.

In the primary sedimentation stage, the sludge settles while grease and oils rise to the surface and are skimmed off. Secondary treatment is designed to substantially degrade the biological content of the sewage which are derived from human waste, food waste, soaps and detergent.

Tertiary treatment removes stubborn contaminants that secondary treatment was not able to clean up. Wastewater effluent becomes even cleaner in this treatment process through the use of stronger and more advanced treatment systems. All tanks are continuously stirred.

Now suppose there is a leak in the process and our goal is to detect its location and magnitude. The example is taken from [146]. The process is described by the following linear model

$$
\begin{aligned}
x_{k+1} = &\begin{bmatrix} 0.89168 & 0 & 0 & 0 & 10 \\ 0.10832 & 0.90518 & 0 & 0.04306 & 0 \\ 0 & 0.09482 & 0.89524 & 0 & 0 \\ 0 & 0 & 0.10476 & 0.89235 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} x_k + \\
&\begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} w_k, \\
y_k = &\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} x_k + v_k,
\end{aligned}
$$

Here the state vector represents $x_k = \begin{bmatrix} m_0 & m_1 & m_2 & m_3 & m_{\text{in}} \end{bmatrix}^{\text{T}}$ with $m_0$ the mass in the equalizing tank, $m_{\text{in}}$ the mass of waste water entering the equalizing tank and $m_i$ the mass in treatment tank $i = 1,2,3$.

Suppose the masses are measured with error covariance

$$R = \text{diag}\begin{pmatrix} 8 & 8 & 8 & 8 & 4 \end{pmatrix}$$

The leak is located in treatment tank 2. The leak is modelled as $w_k = |z_k|$ with $z_k$ a normally distributed random variable with error covariance

$$Q = \text{diag}\begin{pmatrix} 0 & 0 & 5 & 0 & 15 \end{pmatrix}$$

As the location of the leak is unknown, the estimator is designed with

$$Q = \text{diag}\begin{pmatrix} 5 & 5 & 5 & 5 & 15 \end{pmatrix}$$

In order to satisfy the mass balances, the constraints $x_k \geq 0$ and $w_k \geq 0$ are added: tanks have positive mass and mass is only lost through a leak.

Mean losses are represented in Table 4.1. It can be seen that both MHE and the Kalman filter are able to detect the leak in treatment tank 2 and the resulting waste losses entering the equalizing tank as one would expect [146]. When there is no leak, the Kalman filter would predict a net addition of mass to the tank process, which is physically impossible.

The state constraints do not add much information, but the disturbance constraints do. The results are Figures 4.4 to 4.13. As can be seen from the graphs and from the mean squared errors, represented in Table 4.2, MHE is clearly the better estimator.

Computation times are shown in Figure 4.2 for MHE with horizon of 10. For this horizon length high estimation accuracy was obtained and a C code of the interior-point algorithm yields computation times of 1.5 ms. Hence, for this problem size the estimator can be run at almost 1 kHz on a standard computer.

**Table 4.1.** Mean losses - MHE and Kalman filter estimates.

| | mean losses | | | | |
|---|---|---|---|---|---|
| | equal. Tank | Tank 1 | Tank 2 | Tank 3 | Waste |
| Actual | 0 | 0 | 1.7514 | 0 | 3.2396 |
| MHE | 0.2909 | 0.2959 | 1.0198 | 0.3069 | 3.0795 |
| Kalman filter | −0.1818 | −0.0189 | 1.5709 | 0.1533 | 3.0466 |

**Table 4.2.** Mean squared errors for state estimates.

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| MHE | 2.4619 | 1.9164 | 4.8446 | 1.3962 | 0.8280 |
| Kalman filter | 2.8675 | 2.5624 | 5.4899 | 2.1047 | 0.8921 |

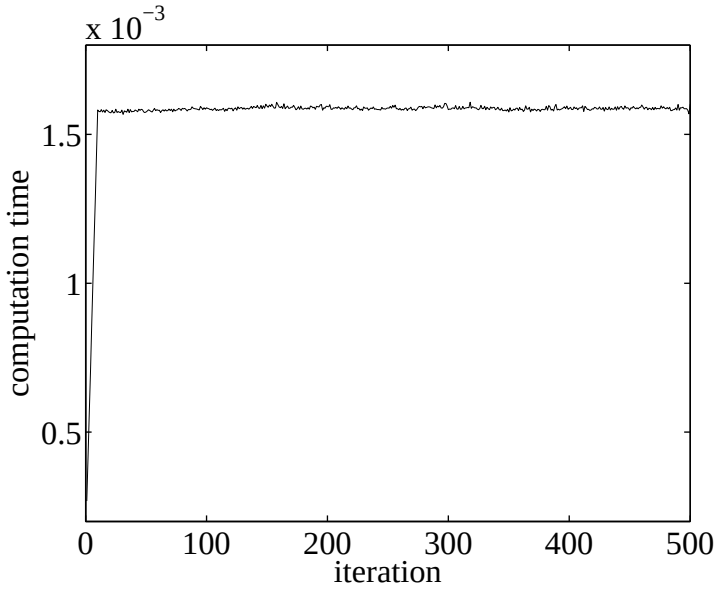



**Figure 4.2.** Computation times in seconds for the waste water treatment problem and MHE with horizon of 10.

### 4.4.3. A hot-starting strategy

In MHE similar optimal control problems are solved repeatedly. Therefore, it is a reasonable assumption that the solution of an MHE problem can be shifted one time step forward to yield a good starting point for the next MHE problem. Unfortunately, as pointed out in [150], in interior-point methods it is better to use a starting point away from the boundary (a strictly feasible point). More specifically we recall that in interior-point methods a sequence of problems of the form (4.2) is solved for decreasing $\kappa$, however if we shift from one MHE problem to the next, we start a new optimization problem, i.e. with $\kappa = \kappa_{\text{init}}$. Hence it turns out that the solution to the previous MHE problem (i.e. with small $\kappa$) is not always a good initialization. The procedure we suggest here, illustrated in Figure 4.4.3, is to solve a sequence of problems for decreasing $\kappa$ until reasonable accuracy is obtained, but use the solution of the problem with $\kappa = \kappa_{\text{init}}$ as an initialization for the next time step. Note that, as pointed out in [192], the particular value of $\kappa_{\text{init}}$ turns out not to matter much; any value over a wide range seems to give good results. In both cases we do not shift the trajectory, but instead omit the first time point and use the model with a zero state disturbance to simulate a new final state, since this typically yields better results than a shift provided that the constraints do not change from one problem to the next. Figure 4.15 shows

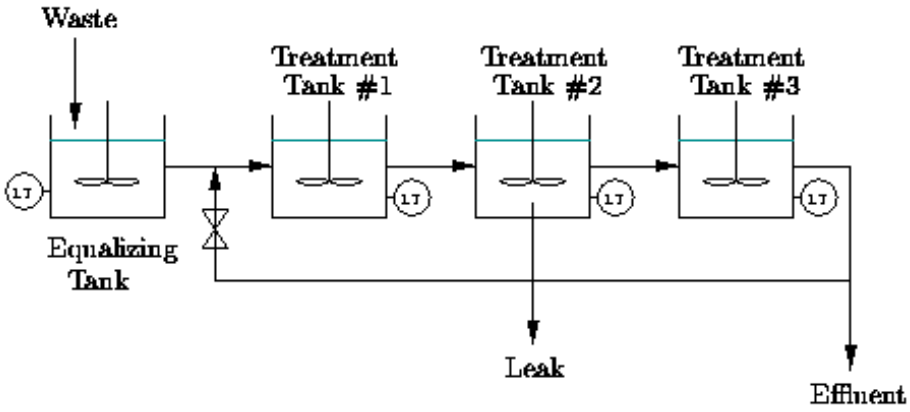





**Figure 4.3.** Detection of a leak in a waste water treatment process (from Rao [146]). **Top:** Illustration of the connected tanks process. **Bottom:** Aerial photograph of a typical waste water treatment plant.

the results for the waste water treatment problem over 500 time steps and MHE with horizon 10. $\kappa_{\text{init}} = 0.01$ and decreasing factor $\tau = 0.9$. When the number of iterations is limited, the hot-starting strategy yields a slight but consitent improvement compared to initialization using the solution of the previous time step. For comparison the results with initialization at zero in every time step, *i.e.* no hot-starting, are depicted as well. From this, it is clear that a smart re-use of information from the previous

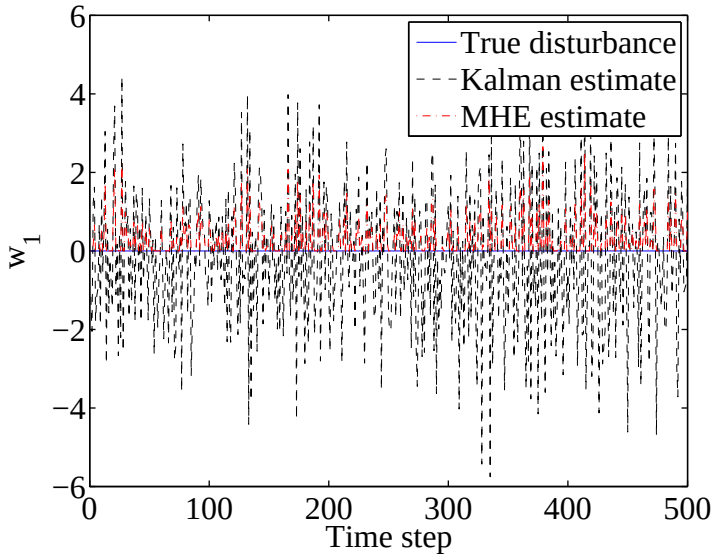



**Figure 4.4.** Disturbance estimates in the equalizing tank.

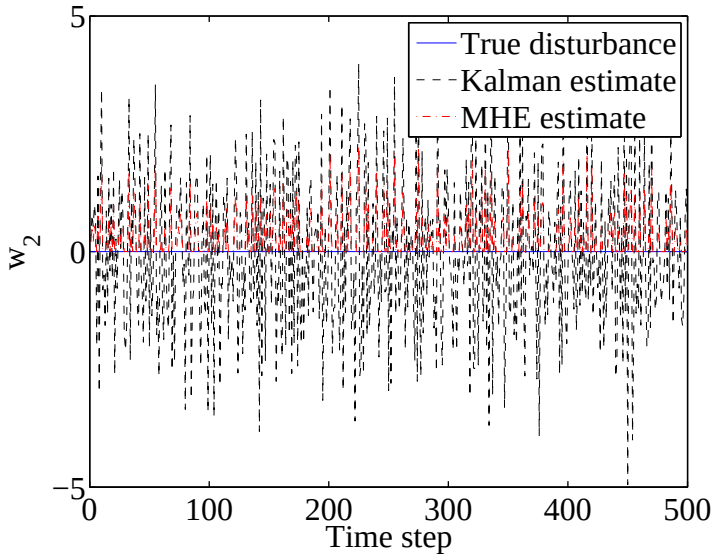



**Figure 4.5.** Disturbance estimates in treatment tank 1.

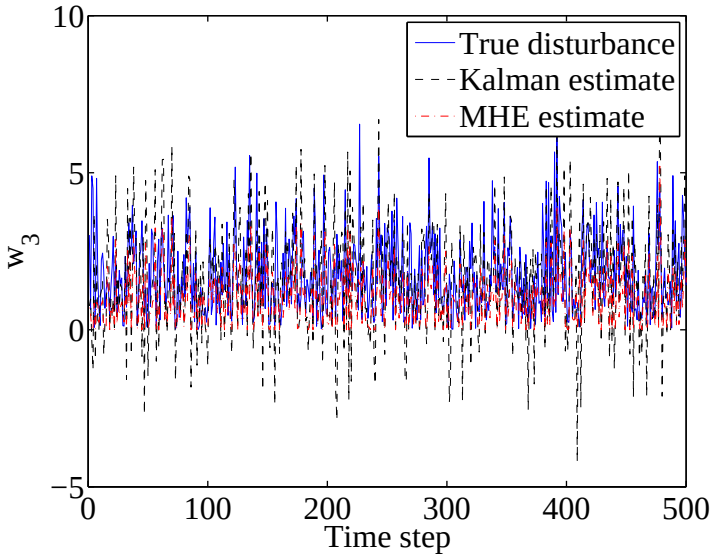



**Figure 4.6.** Disturbance estimates in treatment tank 2.

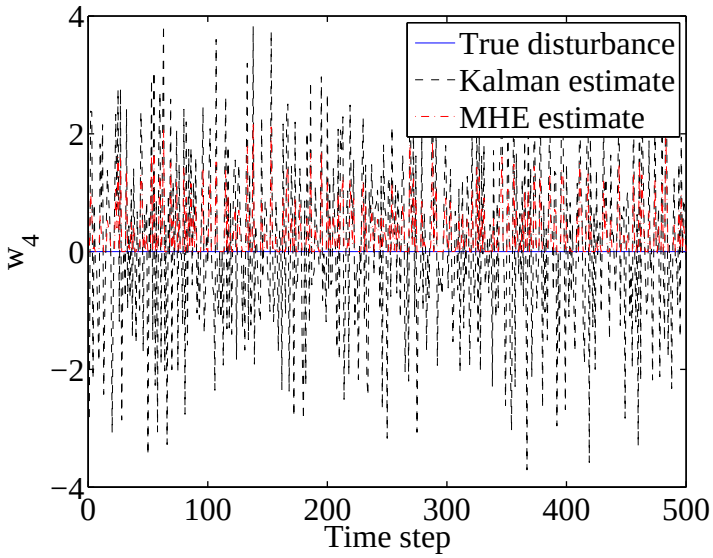



**Figure 4.7.** Disturbance estimates in treatment tank 3.

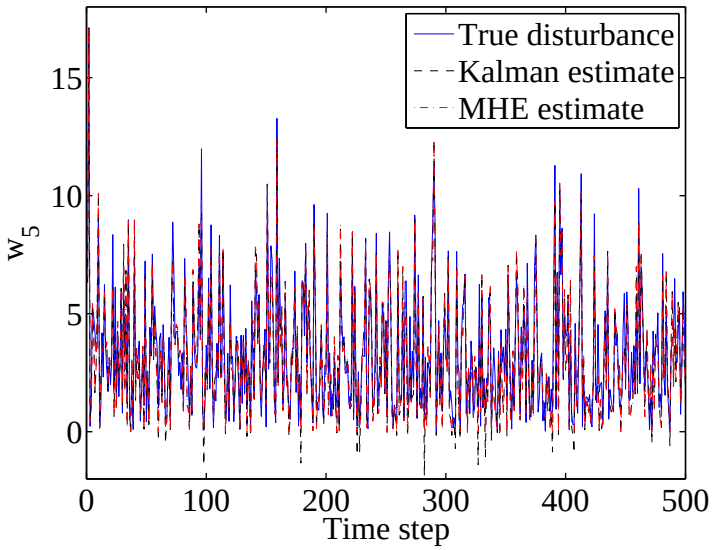



**Figure 4.8.** Disturbance estimates waste entering equalizing tank.

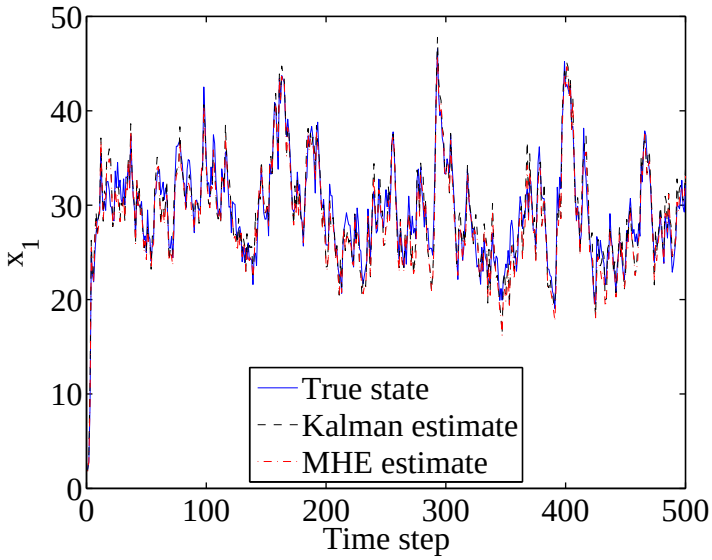



**Figure 4.9.** State estimates in the equalizing tank.

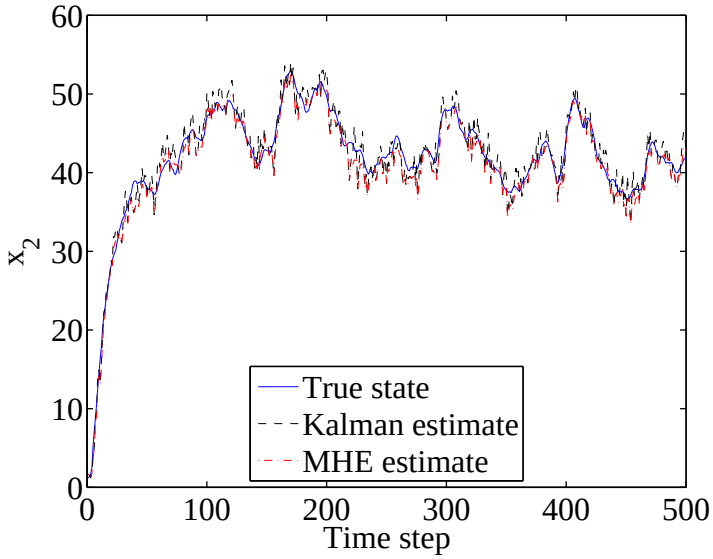



**Figure 4.10.** State estimates in treatment tank 1.

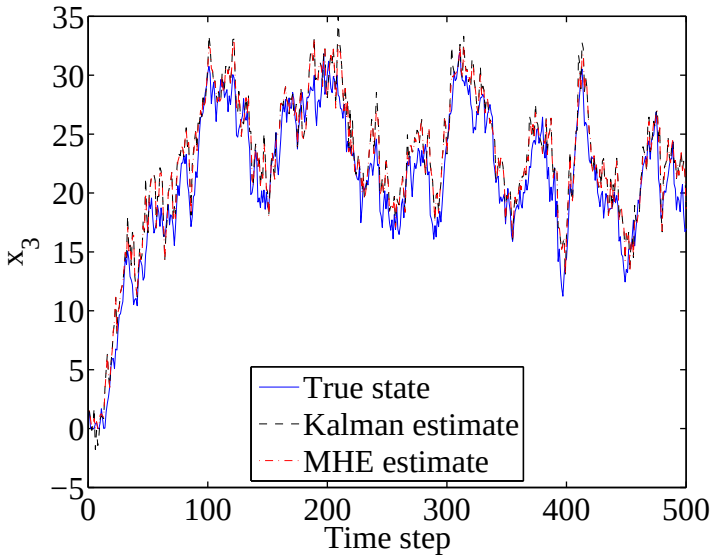



**Figure 4.11.** State estimates in treatment tank 2.

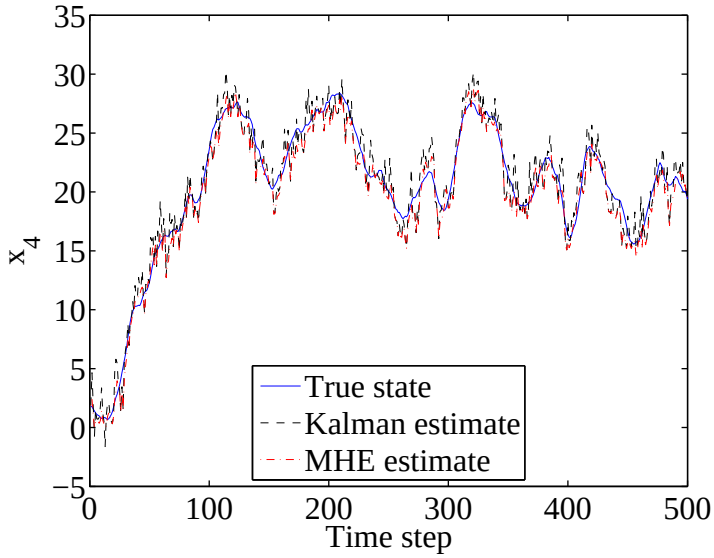



**Figure 4.12.** State estimates in treatment tank 3.

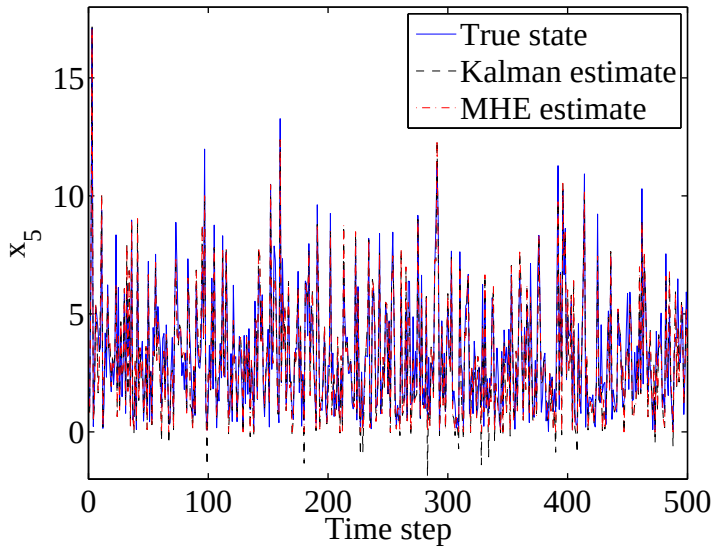



**Figure 4.13.** State estimates waste entering equalizing tank.

time step allows a significant reduction in the number of iterations to convergence, or equivalently, in computation time.

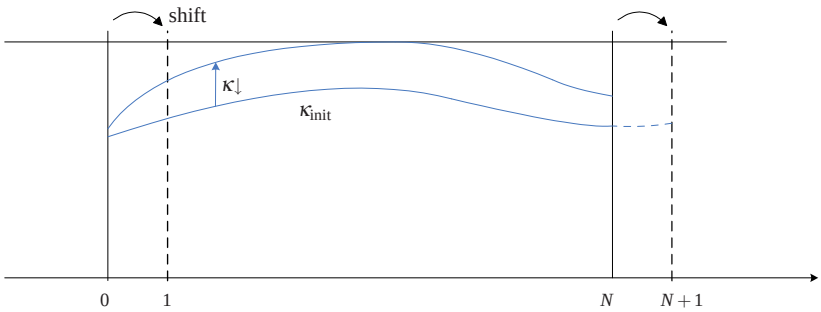



**Figure 4.14.** Illustration of the hot starting procedure. Solve a sequence of problems with decreasing $\kappa$, starting with $\kappa_{\text{init}}$, until reasonable accuracy. Use the solution of the problem with $\kappa_{\text{init}}$ as an initialization for the next time step.

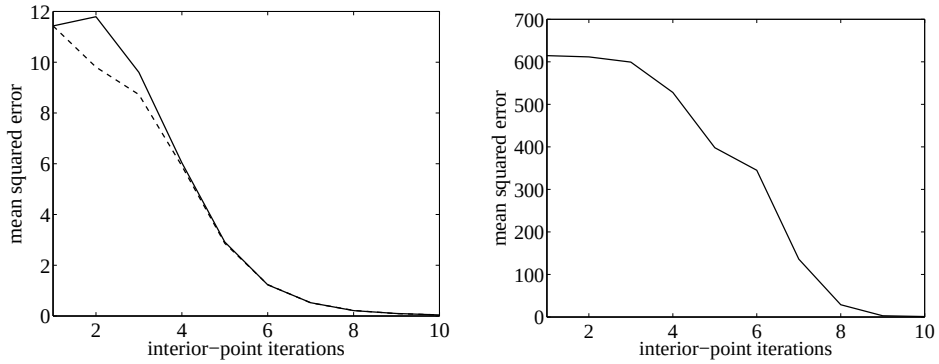



**Figure 4.15. Left:** Mean squared error between the true constrained solution and the solution for a fixed number of iterations initialized by shifting the previous solution (solid line) versus the hot-starting strategy (dashed line). **Right:** Mean squared error for initialization at zero (no shift). The plots were generated for the leak detection problem with 500 time steps and MHE with horizon 10.

### 4.4.4. Numerical conditioning

In order to analyse the numerical conditioning of the square-root and the normal Riccati method, we applied them to the waste water treatment problem for $\kappa_{\text{init}} = 0.01$ and decreasing factor $\tau = 0.9$ and for a fixed $\kappa = 0.01$. The top panels in Figure 4.16 show the condition number of a matrix involved in the measurement update for a state with an actively constrained component. For the normal Riccati method the condition

number of the following matrix is depicted

$$\left(\begin{bmatrix} R_k & \\ & I_{ni_k} \end{bmatrix} + D_k \Sigma_k D_k^{\mathrm{T}}\right) \tag{4.25}$$

which needs to be inverted (see Eq.(4.13)), or in practice is factorized and its factors are applied. For the square-root method, the condition number of

$$\begin{bmatrix} T_k D_k^{\mathrm{T}} & T_k \\ \begin{bmatrix} V_k & \\ & I_{ni_k} \end{bmatrix} & \end{bmatrix}, \tag{4.26}$$

is depicted. Which is the matrix that is factorized in the measurement update of the square-root method (see Eq.(4.14)). The bottom panels in Figure 4.16 show the logarithmic growth of the condition number of the square-root barrier Hessian $M_k$. In case of a decreasing barrier parameter, the effect is slightly amplified. For a decreasing $\kappa$ the algorithm converges in 10 iterations while for a fixed $\kappa$ convergence is achieved in 14 iterations. It can be seen from Figure 4.16 that the condition number for the typcial matrix in the square-root method is the square of the one in the normal method, as expected. This motivates the use of the square-root Riccati method in the context of (primal barrier) interior point methods.

The same problems of numerical conditions are also observed in the method of weighting for constrained least-squares (see [22, 77]). In this case it is recommended to use row and column permutations in the factorization process [22]. For most problems it is sufficient to initially order the rows with potentially large terms first as done in the proposed row-reordering for the measurement update. Although Van Loan [187] shows on certain contrived examples that this is not always sufficient, we have not experienced any problems in our implementations.

## 4.5. Conclusions

In this chapter Riccati based solutions for constrained MHE using a primal barrier interior-point method were presented. It was demonstrated how the barrier terms enter in the measurement update step of the recursion and constraints can be interpreted as *perfect measurements*. The square-root algorithm exploits both the structure and the symmetry in the problems and employs structured QR methods at the core. Several types of constraints were considered: mixed or separate, general or bound constraints. A hot-starting strategy was presented which showed improved performance in the first iterations. The use of square-root Riccati methods within an interior-point method could be strongly motivated by the fact that the condition number of the matrices involved in the factorization typically grow logarithmically. A C implementation

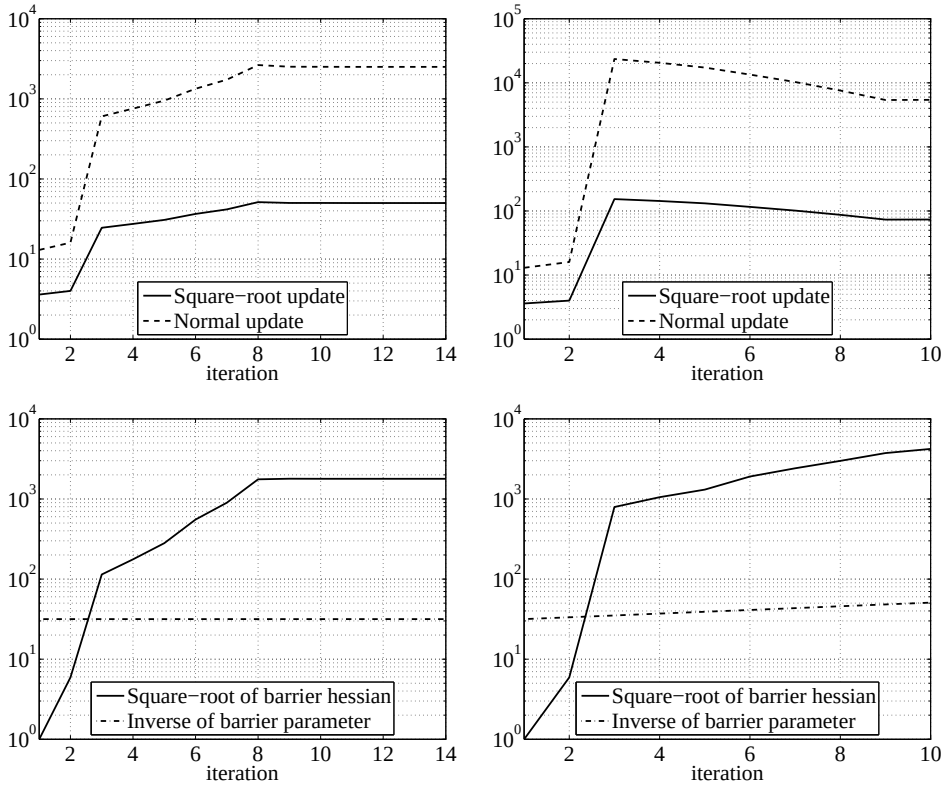



**Figure 4.16.** Logarithmic growth of the numerical conditioning in function of the iteration number for a state with actively constrained components. **Top row:** Evolution condition number of a matrix involved in the measurement update of the square-root (solid) and normal Riccati method (dashed), for a fixed barrier parameter **(left)** and a decreasing barrier parameter **(right)**. **Bottom row:** Inverse barrier parameter (dashed) and evolution of the condition number of the square-root of the barrier hessian (solid), for a fixed barrier parameter **(left)** and a decreasing barrier parameter **(right)**.

demonstrated the performance of the square-root algorithm. It was shown by numerical examples that the method can be run in the milisecond range on a standard computer for moderate horizons.

CHAPTER 5

# Active-set methods for MHE

*In this chapter active-set strategies are proposed for the solution of constrained MHE problems. In particular a Schur complement active-set method is presented. The method uses as the starting point the unconstrained MHE solution which can be computed efficiently using Riccati based algorithms as discussed in Chapter 3 and allows multiple updates to the active set per iteration. These properties are desirable for MHE since only a small number of inequality constraints is expected to be active at the solution. The proposed active-set method involves the solution of reduced non-negativity constrained QPs in the working-set null space. For this, a gradient-projection method using projected Newton steps and Cholesky factorizations is proposed. Cholesky updates are employed in the projected Newton iterations. Once a solution to the reduced QP is found, it is used to update primal and dual variables using a Schur complement technique. Between (outer) active set iterations the reduced Hessian changes by adding some constraints to the working set. These changes are exploited by a Cholesky downdate. Application of a C implementation of the method to several numerical examples shows its excellent performance on typical MHE problems.*

## 5.1. Introduction

Albuquerque and Biegler [3] first proposed a structure-exploiting algorithm for MHE which scales linearly in the horizon length. They applied a type of condensing in which the *controls* and multipliers are eliminated and the reduced QP is formulated and solved in the state space. By construction this approach should have a strong

relation with Riccati recursion, although the proposed algorithm does not explicitly employ a Riccati recursion. Riccati based methods for MHE problems have been proposed before, *e.g.* by Tenny et al [173] and Jorgensen et al [108]. Typically, normal Riccati (Kalman filter) recursions are proposed to solve the KKT system. A structure-exploiting interior-point MHE method using square-root was presented in [90].

Active-set methods for quadratic programming can be classified into *primal* feasible or *dual* feasible methods. In primal feasible methods, a Phase I calculation to find an feasible initial point is typically required. Subsequently, constraints are added or deleted in order to reduce the objective while maintaining primal feasiblity. Upon convergence, dual feasibility is obtained. Dual methods, on the other hand start with a dual feasible point, which usually can be computed cheaply (no Phase I), and maintain dual feasibility during subsequent QP iterations. `QPOPT` [140] is a primal feasible active-set method which uses the null-space approach. It maintains a dense Cholesky factorization of the reduced Hessian. `QPKWIK` [161] is a dual feasible active-set method based on the famous method of Goldfarb and Idnani [76]. Both `QPOPT` and `QPKWIK` are dense methods and both require a positive definite Hessian. Therefore, they are not directly suitable for MHE problems.

A primal Schur complement active-set method, called `SchurQP`, was presented by Gill et al [70]. This method applies the Schur complement for every change in the working set which can be addition or removal of one constraint. A dual Schur complement active-set method, called `QPSchur`, was presented by Bartlett et al [10, 11]. It allows structure exploitation in the KKT system matrices and is applied to MPC problems. The method, however, still requires a positive definite Hessian and proceeds by adding or deleting one constraint at a time as usual in active-set methods.

Axehill and Hansson [7, 8] have presented a dual active-set method for MPC which uses gradient projection. In their algorithm the two step procedure, comprising a Cauchy point calculation and a Newton direction computation, is directly applied to the dual MPC problem. Inequalities are discarded in the Cauchy point calculation and (forward) normal Riccati recursions are used in the projected Newton step. Their method permits multiple active-set changes per iteration, requires few QP iterations and is shown to perform very well on several linear and hybrid MPC examples.

In this chapter, we present a Schur complement active-set method tailored to MHE problems. The method inherits properties from both primal and dual methods. The algorithm is motivated by two observations: (1) the *unconstrained* MHE problem, i.e. discarding inequalities, can be solved very efficiently using Riccati recursions and (2) in MHE only a small number of inequality constraints is expected to be active at the solution. Therefore, the method uses the unconstrained MHE solution as the starting point and solves a number of QPs in the reduced space of working set constraints until the active set is determined. The method allows multiple active-set changes per itera-

tion and typically converges in a few iterations. For the unconstrained MHE solution, a Riccati based method using square-roots is proposed which allows fast computation of the QP matrices. The underlying QPs are non-negativity constrained quadratic problems for which a gradient projection method using projected Newton steps is presented. *Modified* Cholesky factorizations are suggested for solving the QP KKT systems and subsequent changes are exploited by Cholesky downdates and updates at the level of outer and inner active set iterations. The method bears resemblances with the method for MPC by Axehill et al [7, 8], but differs in the following: (1) it is specifically designed for MHE, (2) it uses the unconstrained solution, hence empty initial working set, as a starting point, which can be motivated for MHE, (3) it proceeds by adding sets of constraints per iteration and constraints are only removed upon convergence, inactive constraints are automatically assigned multipliers equal to zero (4) by using Cholesky factorizations for the solution of underlying KKT systems and updating factorizations at both the lower and higher level (inner and outer active set iterations), the efficiency is maximized.

## 5.2. Overview of active-set methods for quadratic programming

In this section, we give a brief introduction to standard active-set methods.

Unlike for interior-point methods, no polynomial bound on the complexity of the active-set method can be given. In fact, active-set methods can display exponential-time behavior on certain problems [115]. However, in practice they can outperform interior-point methods on many problems and the number of iterations required is typically a small polynomial function of the dimension [198]. The key successs factor for active-set methods is a good warm starting strategy, since the number of iterations to reach the optimum is greatly influenced by the initial working set. In a typical active-set method, one constraint is added or deleted from the working set per iteration. Even when there is more than one blocking constraint. For example, if $l$ constraints in the initial working set are inactive at the solution, at least $l$ iterations are required to converge. More than $l$ iterations may be required if constraints are added during iterations but removed later.

The standard convex QP may be expressed as (see (2.49))

$$\begin{array}{ll} \min_z & \frac{1}{2}z^{\mathrm{T}}Hz + g^{\mathrm{T}}z \\ \text{s.t.} & Cz = d, \\ & Pz \leq h, \end{array} \tag{5.1}$$

with Hessian $H \in \mathbf{R}^{n\times n}$ symmetric positive semidefinite, gradient $g \in \mathbf{R}^n$, equality constraints $C \in \mathbf{R}^{m\times n}$, $d \in \mathbf{R}^m$ and inequality constraints $P \in \mathbf{R}^{p\times n}$, $h \in \mathbf{R}^p$.

### 5.2.1. Solving equality constrained QPs

Standard active-set methods proceed by adding or deleting constraints. Therefore, factorization updates for the KKT matrix are typically employed and are crucial to the efficiency of the active-set method. Here, we briefly introduce the three most popular approaches to factorize and update the KKT matrix.

#### The direct factor-solve method

One approach to solving the equality constrained QP is to factorize the full KKT matrix and solve the following KKT system using the factors

$$\begin{bmatrix} H & \bar{C}^{\mathrm{T}} \\ \bar{C} & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ -\lambda \end{bmatrix} = \begin{bmatrix} -g \\ \bar{d} \end{bmatrix}. \tag{5.2}$$

Here $\bar{C}, \bar{d}$ represent the equality constraints $C, d$ augmented with the working set of active inequality constraints $P_i, h_i$ for $i \in \mathcal{W}$. Since the KKT matrix is indefinite, the Cholesky factorization cannot be used. *QR* or *LU* factorization can be used, but in order to exploit the symmetry the $LDL^{\mathrm{T}}$ or *indefinite Cholesky decomposition* is typically used, *i.e.* $M = PLDL^{\mathrm{T}}P^{\mathrm{T}}$. The direct factor-solve method using indefinite Cholesky can be effective on some problems. However, if the KKT system is sparse then this sparsity might be destroyed in the $L$ factor unless the permutation matrix can be chosen by prior knowledge of the KKT structure. In fact the unconstrained MHE problem is an equality constrained QP and the square-root Riccati-based MHE algorithm formalized in Theorem 3.2.6 follows exactly from a direct factor-solve method using prior knowledge of the KKT structure to construct a structure-preserving factorization. When $H$ and $\bar{C}$ are large and sparse, the KKT system may also be solved using a sparse symmetric linear solver such as MA27 [50] or MA57 [51].

#### Range-space method

In the range-space method, the Hessian is used to eliminate the primal direction

$$\bar{C}H^{-1}\bar{C}^{\mathrm{T}}\lambda \quad = \quad \bar{C}H^{-1}g - d. \tag{5.3}$$

This assumes that the Hessian is strictly positive definite. Note that this is not the case for typical MHE problems.

Afterwards, the primal variables are recovered from the dual variables

$$H\Delta z \quad = \quad \bar{C}^{\mathrm{T}}\lambda - g. \tag{5.4}$$

The range-space method works well if the Hessian is easy to invert and the number of equality constraints is small, *i.e.* yielding an effictive data compression by $CH^{-1}C^{\mathrm{T}}$.

### Null-space method

The null-space method allows a positive semi-definite Hessian. It builds on the observation that any vector can be decomposed in two orthogonal parts

$$\Delta z = Y\Delta z_Y + N\Delta z_N, \tag{5.5}$$

where $N \in \mathbf{R}^{n\times(n-m)}$ is the null-space matrix and $Y$ is any matrix such that $\begin{bmatrix} Y & N \end{bmatrix}$ is nonsingular. By substituting this decomposition into the KKT system, the following equations are obtained

$$\bar{C}Y\Delta z_y = d, \tag{5.6}$$
$$HY\Delta z_y + HN\Delta z_N - \bar{C}^{\mathrm{T}}\lambda = -g. \tag{5.7}$$

Multiplying the second equation by $N^{\mathrm{T}}$ yields

$$N^{\mathrm{T}}HN\Delta z_N = -N^{\mathrm{T}}(HY\Delta z_y + g). \tag{5.8}$$

This system with reduced Hessian $N^{\mathrm{T}}HN\Delta z_N \in \mathbf{R}^{(n-m)\times(n-m)}$ is always positive definite and can be solved using Cholesky decomposition. The null-space method works well when the number of degrees of freedom $n-m$ is small.

## 5.2.2. Primal active-set methods

Primal active-set methods start with finding a feasible inital iterate. If a (good) feasible initial point is not immediately available, it can be computed by solving a feasibility problem (typically an LP), which is termed a *Phase I*. Afterwards, equality constrained QP are solved iteratively in which the original equality constraints are complemented with the inequality constraints in the *working set* (equation (5.2)). If $\Delta z \neq 0$, then the objective function is decreased along this search direction. The goal is to select the largest possible step $\alpha$, $0 < \alpha \leq 1$, that does not violate the constraints.

$$\alpha = \min\left(1, \min_{p_i^{\mathrm{T}}\Delta z > 0} \frac{h_i - p_i^{\mathrm{T}} z^j}{p_i^{\mathrm{T}}\Delta z}\right). \tag{5.9}$$

where $j$ denotes the active set iteration index. A constraint which renders $\alpha < 1$ is called a *blocking constraint*. If there is a blocking constraint, it is added to the working

set. If not, then a full step is taken $\alpha = 1$ and the working set is left unchanged. Next, optimality is checked. If there are negative multipliers, then the constraint with most negative multiplier is removed from the working set and the iteration is repeated. Hence, in every iteration primal feasibility is maintained. Dual feasibility is obtained upon convergence.

There is flexibility in the choice of initial working set and each initial choice leads to a different iteration sequence. If it is possible to obtain a good choice of the initial working set from prior knowledge of the QP, then a substantial reduction in the number of active-set iterations can be achieved.

### 5.2.3. Dual active-set methods

In the dual approach, a dual feasible initial iterate is required which is usually computed cheaply. Hence a Phase I is avoided, which is the main motivation for dual active-set methods. More specifically, the following initialization is dual feasible

$$\begin{aligned} z^0 &= -H^{-1}g \\ y^0 &= 0 \\ \mathcal{W} &= \emptyset \end{aligned}$$

where $y$ represents the multipliers for the inequalities in the working set.

We give a short description of the famous dual active-set method by Goldfarb and Idnani [76], which is applicable to strictly convex QPs. An extension to convex QPs was given by Boland [26]. At every iteration in a dual active-set method, a violated constraint is selected to be added to the working set. Let us denote this constraint as $q$. Then the step directions for the primal and dual variables are determined as follows

$$\Delta z = \left[H^{-1}\bar{C}^{\mathrm{T}}\left(\bar{C}H^{-1}\bar{C}^{\mathrm{T}}\right)^{-1}\bar{C}H^{-1} - H^{-1}\right]p_q^{\mathrm{T}}, \tag{5.10}$$

$$\Delta\lambda = -\left(\bar{C}H^{-1}\bar{C}^{\mathrm{T}}\right)^{-1}\bar{C}H^{-1}p_q^{\mathrm{T}}. \tag{5.11}$$

Note that these step directions are very similar to the range-space approach (equations (5.3) and (5.4)).

The *primal-dual step length* $\tau$ is chosen such that constraint $q$ becomes active (primal feasible), but small enough to maintain dual feasibility.

$$\tau = \min\left\{\tau^{\text{primal}}, \tau^{\text{dual}}\right\}, \tag{5.12}$$

with

$$\tau^{\text{primal}} = \begin{cases} \infty & \text{if} \Delta z = 0 \\ \frac{p_q^T z^j - h_q}{p_q^T \Delta z} & \text{otherwise} \end{cases} \tag{5.13}$$

$$\tau^{\text{dual}} = \min_{i \in \mathcal{W}} \left\{ \frac{-\lambda_{\mathcal{W}}^j}{\Delta \lambda_i} \mid |\Delta \lambda_i > 0 \right\}. \tag{5.14}$$

If the primal step direction $\Delta z$ is non-zero, a primal-dual step is taken. There are two possibilities. If $\tau = \tau^{\text{primal}}$, a full primal step is taken and constraint $q$ is added to the working set. If $\tau = \tau^{\text{dual}}$, only a partial step can be taken, since the blocking constraint determined by (5.14) must be dropped from the working set. If the $\Delta z = 0$, constraint $q$ cannot be be satisfied simultaneously with all other constraints in the working set. Therefore, no primal step is taken. A partial dual step is taken, provided $\tau^{\text{dual}} < \infty$, such that one constraint can be removed from the working set. In case such a constraint cannot be found, the QP is infeasible. Once a full step can be taken, a new violated constraint is selected and added to the working set and the procedure is repeated. If there are no violated constraints, the algorithm successfully terminates.

Summarizing, in a dual active-set method, the KKT matrix is updated as constraints are added or removed from the working set and the dual objective function is iteratively increased while maintaining dual feasibility. The algorithm terminates at a primal feasible point.

### 5.2.4. qpOASES - an online active-set strategy

Ferreau et al [56, 57] have developed an online active-set strategy which is extremely useful for parameterized successive QPs as it employs a homotopy path from one QP to the next thereby allowing warm starting without a phase I. The original method [56] uses a null-space method for solving the equality constrained QPs. An open-source implementation of the method is available under the name qpOASES [55]. The method cannot be strictly classified as either primal or dual.

## 5.3. A Schur-complement active-set method for MHE

The main motivation for the proposed algorithm is that the unconstrained problem, i.e. discarding inequalities, can be solved very efficiently (Chapter 3) and that in MHE

only a small number of inequality constraints is expected to be active at the solution. The latter is due to the stochastic nature of the estimation problem: different observation sequences will yield (slightly) different solutions which prevents that certain inequalities are always active. Constraints are often merely added to avoid divergence or convergence to far-off local optima.

### 5.3.1. Outline of the active-set method

Let us recapture the standard QP formulation (5.1)

$$\begin{array}{ll} \min_z & \frac{1}{2} z^{\mathrm{T}} H z + g^{\mathrm{T}} z \\ \text{s.t.} & Cz = d, \\ & Pz \leq h, \end{array}$$

which represents a linear constrained MHE problem if the optimization variables are defined as

$$z = (x_0, w_0, x_1 \ldots, w_{N-1}, x_N) \in \mathbf{R}^{(N+1)n+Nm}.$$

and the matrices are composed as follows

$$\begin{aligned} H &= \begin{bmatrix} S_0^{-1} S_0^{-T} + C_0^T C_0 & C_0^T H_0 & & \\ H_0^T C_0 & I_m + H_0^T H_0 & & \\ & & \ddots & \\ & & & C_N^T C_N \end{bmatrix}, \\ g &= \begin{bmatrix} -S_0^{-1} S_0^{-T} \bar{x} + C_0^T h_0 \\ H_0^T h_0 \\ \vdots \\ H_{N-1}^T h_{N-1} \\ C_N^T h_N \end{bmatrix}, \quad b = - \begin{bmatrix} f_0 \\ \vdots \\ f_{N-1} \end{bmatrix}, \\ C &= \begin{bmatrix} A_0 & G_0 & -I_n & & & \\ & & & \ddots & & \\ & & & & A_{N-1} & G_{N-1} & -I_n \end{bmatrix}, \\ P &= \begin{bmatrix} T_0^x & T_0^w & & & & \\ & & \ddots & & & \\ & & & T_{N-1}^x & T_{N-1}^w & \\ & & & & & T_N^x \end{bmatrix}, \quad h = \begin{bmatrix} t_0 \\ \vdots \\ t_{N-1} \\ t_N \end{bmatrix}, \end{aligned} \tag{5.15}$$

where $I_n$ denotes the unit matrix of dimension $n$.

Furthermore, let us introduce multipliers $\lambda \in \mathbf{R}^{n_c}$ associated with the equality constraint $Cz = b$ and multipliers $y \in \mathbf{R}^{n_p}$ associated with the inequality constraints $Pz \leq h$.

We denote the $i$-th row of the inequality constraints as $p_i^{\mathrm{T}} z \leq h_i$ and say that this constraint is *active* at $z_0$ if $p_i^{\mathrm{T}} z_0 = h_i$ holds. Active-set algorithms search iteratively for the binding set or set of active constraints at optimality. In every iteration a working set is kept $I \subseteq \{1, \ldots, n_p\}$.

Then the KKT optimality conditions for QP (5.1) are given by (see Chapter 2):

$$\begin{aligned} Hz + g + C^T\lambda + \textstyle\sum_{i\in\mathbf{A}} p_i^T y_i &= 0 \\ Cz - b &= 0, \\ Pz - h &\leq 0, \\ y &\geq 0, \\ y_i(p_i^T z - h_i) &= 0, \quad i \in \mathbf{A} \end{aligned} \tag{5.16}$$

The dual problem of QP (5.1) is

$$\begin{aligned} \max_{z,y} \quad & -\tfrac{1}{2} z^{\mathrm{T}} H z + y^{\mathrm{T}} b \\ \text{s.t.} \quad & Cz = d, \\ & Hz + g = P^{\mathrm{T}} y, \\ & y \geq 0, \end{aligned} \tag{5.17}$$

or, equivalently

$$\begin{aligned} \max_{y\geq 0} \quad \min_{z} \quad & \tfrac{1}{2} z^T H z + g^T z + y^{\mathrm{T}}(p_{\mathbf{A}}^{\mathrm{T}} z - h_{\mathbf{A}}) \\ \text{s.t.} \quad & Cz = b, \end{aligned} \tag{5.18}$$

A solution for the equality constrained problem, discarding inequalities, is found by solving the KKT system

$$K\xi = r, \tag{5.19}$$

with

$$K = \begin{bmatrix} H & C^T \\ C & 0 \end{bmatrix}, \xi = \begin{bmatrix} z \\ \lambda \end{bmatrix}, r = \begin{bmatrix} -g \\ b \end{bmatrix} \tag{5.20}$$

Now, for the definitions (5.15), this corresponds to the *unconstrained MHE problem* and we have presented in Chapter 3 efficient Riccati based methods for this problem. The active-set method we propose here, uses the unconstrained MHE solution as a

starting point. Next, if there are violated inequality constraints, they are added to the working set $I \subseteq \{1,\ldots,n_p\}$. A solution to the equality constrained QP

$$\begin{array}{ll} \min_z & \frac{1}{2}z^T Hz + g^T z \\ \text{s.t.} & Cz = b, \\ & p_I^T z \leq h_I, \end{array} \tag{5.21}$$

can then be obtained by a Schur complement technique.

**Proposition 5.3.1.** *Under the conditions of Lemma 3.2.4, the KKT matrix K is invertible.*

*Proof.* See Lemma 3.2.4. □

Let us define $V^{\mathrm{T}} = \begin{bmatrix} p_I^T & 0 \end{bmatrix}$. Then our aim is to find $(z^{\text{new}}, \lambda^{\text{new}})$ such that the inequality constraints are satisfied, or

$$V^{\mathrm{T}} \begin{bmatrix} z^{\text{new}} \\ \lambda^{\text{new}} \end{bmatrix} \leq h_I \ . \tag{5.22}$$

By projecting onto the space of active constraints the following reduced QP is obtained

$$\begin{array}{ll} \min_y & \frac{1}{2}y^T My + c^T y \\ \text{s.t.} & y \geq 0, \end{array} \tag{5.23}$$

with

$$M = V^T(K^{-1}V), \tag{5.24}$$

$$c = -V^T \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} + h_I. \tag{5.25}$$

Note that $M$ is the Schur complement of $K$ in the KKT matrix $\begin{bmatrix} K & V \\ V^{\mathrm{T}} & 0 \end{bmatrix}$.

**Proposition 5.3.2.** *The reduced Hessian $M = V^T K^{-1} V$ is positive semidefinite.*

*Proof.* First, note that $M$ exists by Proposition 5.3.1. Substitute $V$ and $K$ from their definitions (5.19) resp. (5.22)

$$M = \begin{bmatrix} p_I^T & 0 \end{bmatrix} \begin{bmatrix} H & C^T \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} p_I \\ 0 \end{bmatrix} , \tag{5.26}$$

with H symmetric positive semidefinite. The proof is given in [139]. □

**Corollary 5.3.3.** *As a consequence of Proposition 5.3.2 the reduced QP is convex.*

After a solution of the reduced QP is obtained, the unconstrained MHE solution can be expanded using a Schur complement technique as follows

$$\begin{bmatrix} z^{\text{new}} \\ \lambda^{\text{new}} \end{bmatrix} = \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} - (K^{-1}V)y. \tag{5.27}$$

**Lemma 5.3.4.** *Given a pair $(z,\lambda)$ and a working set defined by matrix $V^{\mathrm{T}} = \begin{bmatrix} p_I^T & 0 \end{bmatrix}$ and vector $h_I$, the solution of the reduced QP (5.23) and Schur complement vector update (5.27) yield a pair $(z^{\text{new}}, \lambda^{\text{new}})$ with $z^{\text{new}}$ satisfying the inequality constraints in the working set and corresponding multipliers $y_I$.*

*Proof.* The KKT conditions of the reduced QP (5.23) are

$$\begin{aligned} My + c - \mu &= 0, \\ y &\geq 0, \\ \mu &\geq 0, \\ y_i \mu_i &= 0. \end{aligned} \tag{5.28}$$

From the first equation we can eliminate $\mu$, leading to

$$\begin{aligned} My + c &\geq 0, \\ y &\geq 0, \ , \\ y_i(My+c)_i &= 0, \end{aligned} \tag{5.29}$$

By substituting $M$ and $c$ from their definition, we get for the first equation

$$\begin{aligned}
&V^T K^{-1} V y - V^T \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} + h_I \geq 0, \\
\Leftrightarrow\ &-V^T \left( \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} - K^{-1} V y \right) \geq -h_I, \\
\Leftrightarrow\ &V^T \begin{bmatrix} z^{\text{new}} \\ \lambda^{\text{new}} \end{bmatrix} \leq h_I, \\
\Leftrightarrow\ &p_I^T z^{\text{new}} \leq h_I,
\end{aligned}$$

which yields the following equivalent set of KKT conditions for the reduced QP

$$\begin{array}{ll}
p_I^T z^{\text{new}} & \leq h_I, \\
y & \geq 0, \\
y_i (p_i^T z^{\text{new}} - h_i) & = 0.
\end{array} \tag{5.30}$$

Hence, the solution of the reduced QP gives us a $z^{\text{new}}$ which satisfies the inequalities in the working set and corresponding multipliers $y_I$. □

Next, the new solution is checked with the inequalities. If there are (other) violated inequalities, they are added to the working set and the procedure is repeated. The procedure is summarized in Algorihtm 12. Convergence of the algorithm is formalized by Theorem 5.3.5. Figure 5.1 represents a flowchart of the proposed algorithm.

It is important to note that the working set is growing and inactive constraints are only removed from the working set upon convergence, simply by checking which constraints have multipliers equal to zero. This strategy prevents cycling.

**Algorithm 12.** *[Active-set Schur-complement method]*

1. *Initialize iteration counter $j = 0$, working set $I = \emptyset$ and compute the unconstrained solution*

$$\begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} = (K^{-1} r)$$

2. *Add the violated inequalities to $I$*

$$I = I \cup \{ i | p_i^T z > h_i \}$$

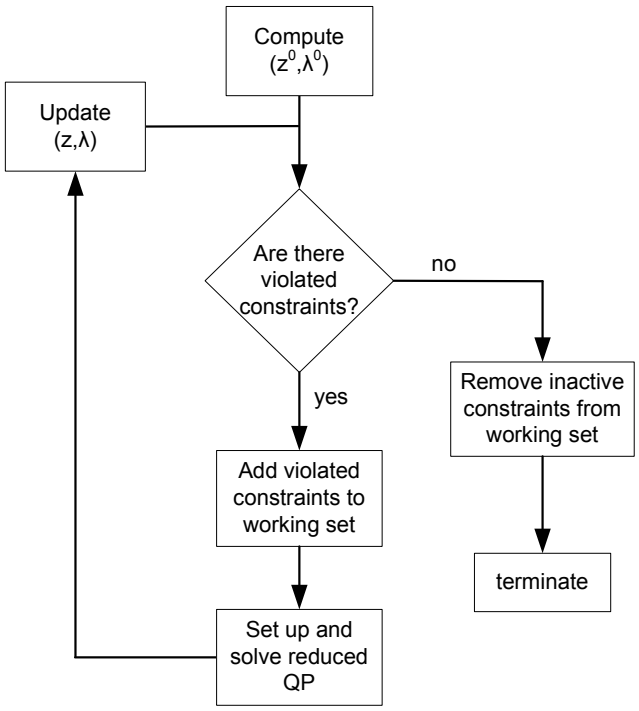



**Figure 5.1.** Flowchart for the Schur-complement active-set method.

*Define* $V^T = \begin{bmatrix} p_I^T & 0 \end{bmatrix}$. *Then we want to find* $(z^{\text{new}}, \lambda^{\text{new}})$ *such that*

$$V^T \begin{bmatrix} z^{\text{new}} \\ \lambda^{\text{new}} \end{bmatrix} \leq h_I$$

*Define*

$$\begin{aligned} M &= V^T (K^{-1} V) \\ c &= -V^T \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} + h_I \end{aligned}$$

3. *Solve the reduced QP*

$$\begin{aligned} \min_y \quad & \tfrac{1}{2} y^T M y + c^T y \\ \text{s.t.} \quad & y \geq 0 \end{aligned}$$

*which yields the multipliers for inequalities (5.22).*

4. *Expand the solution using a Schur complement technique*

$$\begin{bmatrix} z^{\text{new}} \\ \lambda^{\text{new}} \end{bmatrix} = \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} - (K^{-1}V)y$$

5. *Verify that $z^{\text{new}}$ satisfies all other inequalities. If so, go to step 6. If not, increase the iteration counter $j \leftarrow j+1$ and go to step 2.*

6. *Solution obtained. Drop all inactive inequalities, i.e. with multipliers equal to zero. This gives the set of active constraints $\mathbf{A}$, and corresponding components of the constraint matrix and vector $p_{\mathbf{A}}, h_{\mathbf{A}}$.*

**Theorem 5.3.5.** *After termination of the algorithm all KKT conditions of the original QP (5.1) are satisfied.*

*Proof.* To verify this, let us denote the *nonzero* values of $y$ as $y^*$. Then the KKT conditions for QP (5.1) are given as

$$\begin{bmatrix} K & V_{\mathbf{A}} \\ V_{\mathbf{A}}^T & 0 \end{bmatrix} \begin{bmatrix} \xi^* \\ y^* \end{bmatrix} = \begin{bmatrix} r \\ h_{\mathbf{A}} \end{bmatrix} \ , \tag{5.31}$$

or after substituting $K$ and $r$

$$\begin{bmatrix} H & C^{\mathrm{T}} & P_{\mathbf{A}} \\ C & 0 & 0 \\ P_{\mathbf{A}}^{\mathrm{T}} & 0 & 0 \end{bmatrix} \begin{bmatrix} z^* \\ \lambda^* \\ y^* \end{bmatrix} = \begin{bmatrix} -g \\ b \\ h_{\mathbf{A}} \end{bmatrix} \ . \tag{5.32}$$

Which is equivalent to (5.16), q.e.d. □

Furthermore, from the following equation

$$\begin{bmatrix} z^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} z^0 \\ \lambda^0 \end{bmatrix} - K^{-1}V_{\mathbf{A}}y^* \tag{5.33}$$

it is clear how the constrained solution relates to the *unconstrained* solution.

Note the difference with a classical (primal) active-set Schur-complement method which typically starts with a guess for the active set $\tilde{P}, \tilde{h}$ (without solving the unconstrained system), solves a KKT system of the form

$$\begin{bmatrix} K & \tilde{P}^T \\ \tilde{P} & 0 \end{bmatrix} \begin{bmatrix} \xi \\ \tilde{\pi} \end{bmatrix} = \begin{bmatrix} r \\ \tilde{h} \end{bmatrix} \ ,$$

and then iteratively adds violated inequalities, one at a time, and updates the KKT solutions. If there are many active constraints, this approach requires many iterations which can be time consuming. The proposed Schur complement active-set method on the other hand allows larger working set updates in every iteration.

### 5.3.2. MHE solution using Riccati recursions

The unconstrained MHE solution can be obtained efficiently using normal or square-root Riccati recursions as described in detail in Chapter 3. These methods comprise of a forward matrix recursion, which factorizes the KKT matrix, combined with a forward vector solve followed by a backward vector solve. The normal Riccati method essentially factorizes the KKT system as an $LU$ factorization (Lemma 3.2.1), while the square-root Riccati method represents an $LDL^{\mathrm{T}}$ factorization (Lemma 3.2.5). A square-root version using structured $QR$-factorizations was shown to exhibit excellent numerical robustness properties (see Chapter 3). These MHE methods scale linearly with the horizon length.

### 5.3.3. Forward and backward vector solves

In order to calculated the Schur complement matrix $M$ (5.24), we need to solve $K^{-1}V$. Fortunately, from the Riccati based solution of the unconstrained MHE problem we already have a factorization of $K^{-1}$. If the square-root Riccati method is employed it is given as

$$K^{-1} = L^{-\mathrm{T}} D^{-1} L^{-1} \tag{5.34}$$

Recall that $D = \mathbf{blkdiag}(I_{n+m}, -I_n, \ldots, I_{n+m}, -I_n, I_n)$, hence $D^{-1} = D$.

Thus, to compute $V^{\mathrm{T}}(K^{-1}V)$, we need to solve

$$LX = V, \tag{5.35}$$

change the sign of the components in $X$ associated with the multipliers (let us denote it $X_1$), and left-multiply with the original $X^{\mathrm{T}}$.

Equation (5.35) can be solved using direct forward solves employing square-root factors (see Theorem 3.2.6). Furthermore, note that $V$ has special structure. It only contains non-zeros at the locations of the violated constraints. Therefore, only partial forward solves are needed starting at the location of the constraint violation, which further reduces the computation times.

After obtaining the reduced QP solution $y$, the primal and dual variables are updated using equation (5.27). Here, we need to compute $K^{-1}Vy$. This is done by multiplying the already obtained $X_1 = D^{-1}L^{-1}V$ with $y$ which yields a vector. And apply one direct backsolve (see Theorem 3.2.6).

For comparison, the dual active-set algorithm for MPC by Axehill et al [7, 8] which applies gradient projection to the dual MPC problem and employs a (forward) normal Riccati recursion requires one partial forward solve and one backward solve per active-set change. The latter is avoided in our algorithm by the Schur complement expansion.

### 5.3.4. Gradient projection method

Notice the special form of the reduced QP (5.23): only non-negativity constraints and no equalities. Several methods exist for solving these types of optimization problems. We propose a gradient projection method which is designed to make rapid changes to the active set allows to exploit the factorizations already computed in previous iterations at the highest and lowest level by Cholesky updates. The method was also adapted to work on semidefinite matrices.

The gradient projection is described in detail below and was compared with two other QP methods: qpOASES, an open-source C++ implementation of the online active set strategy developed by Ferreau et al [56, 57], and a primal barrier interior-point method. The reduced QP is indeed also an ideal case for a primal barrier interior-point method, since any point in the positive orthant is a valid starting point. Hence, a Phase I can be omitted. With suitable values for the initial barrier parameter and descreasing factor, we observed that the algorithm typically converges after only 10-20 iterations. Unfortunately, factorizations cannot (easily) be re-used in the interior-point method.

The presentation of the gradient projection method is largely a summary of the ones presented in [136] and [7]. The gradient projection method consists of two stages, see Figure 5.2. In the first stage, the Cauchy point $x_c$ is computed by searching along a piecewise linear path starting from the current iterate in the steepest descent direction. The working set is then defined as the set of active constraints at the Cauchy point. By this stage, global convergence of the gradient projection method is guaranteed. However, the convergence rate can be improved by adding a second step in which a QP in the subspace defined by the working set is solved approximately.

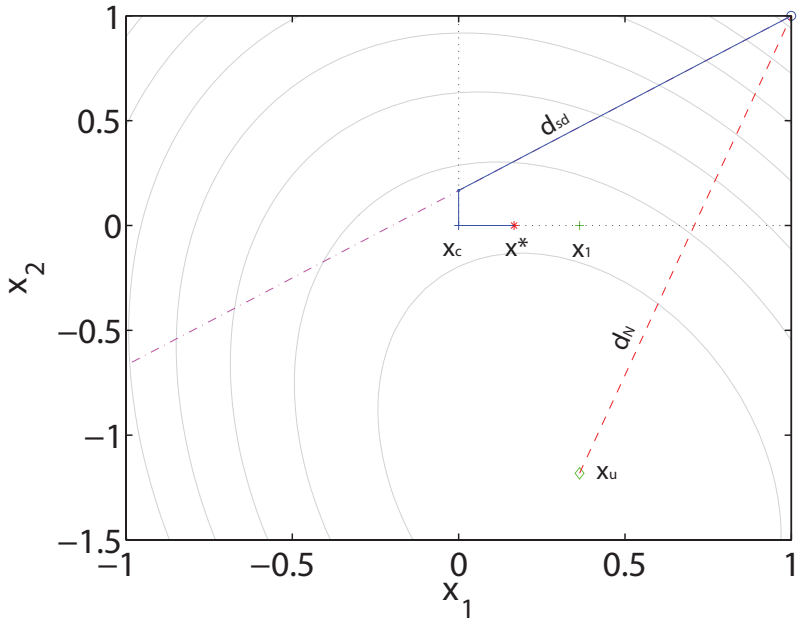



**Figure 5.2.** The figure illustrates that it is not sufficient to project the unconstrained minimizer in order to solve the non-negativity constrained QP. The contour lines of the quadratic objective are plotted in gray. The constrained solution is denoted $x^*$. The unconstrained minimizer is denoted $x_u$ and its projection is denoted $x_1$. The gradient projection method first computes the Cauchy point $x_c$ by searching along a piecewise linear path starting from the current iterate in the steepest descent direction $d_{\text{sd}}$. This step guarantees global convergence. Next, the convergence rate is improved *i.e.* by a projected Newton step. The gradient projection method converges in two iterations for this example.

### Cauchy point computation

For the non-negativity constrained QP (5.23), the piecewise linear path from the current point $x^k$ in the steepest descent direction $p$ is given by

$$x(t) = [x^k + tp]^+, \tag{5.36}$$

where the scalar $t$ parametrizes the path and where the projection onto the positive orthant is defined as

$$[z]_i^+ = \begin{cases} z_i, & z_i \geq 0 \\ 0, & z_i < 0. \end{cases} \tag{5.37}$$

The Cauchy point is then computed as the first local minimizer of a univariate piecewise quadratic function. To find this minimizer, the breakpoints are computed, and

then each of the line segments is examined seperately until a minimizer is found. The breakpoints are given explicitly as [7, 136]

$$\bar{t_i} = \begin{cases} \frac{x_i^k}{p_i}, & p_i < 0 \\ \infty, & p_i \geq 0. \end{cases} \tag{5.38}$$

The components of $x(t)$ can then be expressed as

$$x_i(t) = \begin{cases} x_i^k + t p_i, & t < \bar{t_i} \\ 0, & \text{otherwise.} \end{cases} \tag{5.39}$$

Duplicate values and zeros have to be removed and the remaining values are sorted into an ordered set such that $0 < t_1 < t_2 < \ldots < t_f$. On each interval $[0,t_1]$, $[t_1,t_2]$, $\ldots$,$[t_{f-1},t_f]$, the objective function is quadratic and can be optimized analytically if the upper and lower bounds are temporarily discarded. For the interval $[t_{j-1},t_j]$ it follows that

$$x(t) = x(t_{j-1}) + \Delta t \hat{p}^{j-1}, \quad \Delta t \in [0, t_j - t_{j-1}] \tag{5.40}$$

with

$$\hat{p}_i^{j-1} = \begin{cases} p_i, & t_{j-1} < \bar{t_i} \\ 0, & \text{otherwise.} \end{cases} \tag{5.41}$$

Inserting this into the objective function results on a scalar unconstrained quadratic optimization problem on each interval, which can be solved analytically. After such an unconstrained minimizer for one of these quadratic subproblems has been found, it has to belong to the current interval, *i.e.* be at $t_{j-1} + \Delta t^*, \quad \Delta t^* \in [0, t_j - t_{j-1})$, or be at the boundary $t_{j-1}$. In all other cases, we move on to the next interval and continue the search.

### Projected Newton method

After the Cauchy point has been computed, the working set is defined as the components of $x^c$ which are at the bounds, *i.e.* which are zero. The following QP can be formulated

$$\begin{array}{ll} \min_z & \frac{1}{2}x^{\mathrm{T}}Mx + c^{\mathrm{T}}x \\ \text{s.t.} & x_i = x_i^{\mathrm{c}},\ i \in \mathcal{A}(x^c), \\ & x_i \geq 0,\ i \notin \mathcal{A}(x^c) \end{array} \tag{5.42}$$

Solving this QP exactly may be as hard as solving the original QP. Therefore, it is commonly approximated by an equality constrained QP ignoring the bounds and solved iteratively using *e.g.* a projected conjugate gradient method or a projected Newton method [7, 15].

$$\begin{array}{ll} \min_z & \frac{1}{2}x^{\mathrm{T}}Mx + c^{\mathrm{T}}x \\ \text{s.t.} & x_i = 0,\ i \in \mathcal{W}, \end{array} \tag{5.43}$$

where $\mathcal{W}$ denotes the working set. The projected Newton method has been opted for, motivated by the fact that it yields a Newton step in the subspace defined by the active components and thereby allows a substantial improvement to the convergence rate obtained from the Cauchy steps. Indeed, the gradient method (Cauchy steps) typically converges in a zig-zag path yielding slow convergence. By adding a Newton step with quadratic convergence rate, the performance of the algorithm can be drastically improved. This is illustrated in Figure 5.3 for a problem of order 50 with 31 active constraints and condition number $\kappa(M) = \frac{|\lambda_{\max}(M)|}{|\lambda_{\min}(M)|} = 1e^4$. The two-stage gradient projection method converges in less than 5 iterations.

The KKT conditions for QP (5.43) are given by

$$\begin{array}{rll} m_i^{\mathrm{T}}x & = -c_i, & i \notin \mathcal{W} \\ m_i^{\mathrm{T}}x - \lambda & = -c_i, & i \in \mathcal{W} \\ x_i & = 0 & i \in \mathcal{W} \end{array} \tag{5.44}$$

Hence, a Newton step in the subspace defined by the non-zero components remains to be calculated. This can be done by a Cholesky factor-solve. However, since the matrix $M$ can be positive semidefinite, a modified outer product Cholesy factorization [77, page 148] is developed. It is described in Algorithm 13. Note that when a zero pivot is encountered nothing is to be done. This is verified by [77, Theorem 4.2.6].

**Algorithm 13.** *[Cholesky decomposition for positive semidefinite matrices.]*

***for*** $i = 1$ *to* $n$ ***do***
  ***if*** $L(i,i) > 0$ ***then***
    $L(i,i) = \sqrt{L(i,i)}$
    $L(i+1:n,i) = L(i+1:n,i)/L(i,i)$

```
    for j = i+1 to n do
      L_(j:n, j) = L(j:n, j) − L(j:n, i)L(j, i)
    end for
  end if
end for
```

The projected Newton method proceeds by finding a Newton step in the subspace defined by the working set. If negative components of the minimizer are encoutered, they are added to the working set and a new problem is solved. In contrast to an ordinary active set method, several constraints can be added in every iteration.

**Algorithm 14.** *[Projected Newton algorithm]*

```
I = {i | x_i^c = 0}
J = {1, ..., n}
while p > 0 do
  J = J \ I
  L^r = rows and columns deletion (Algorithm 15) (L, I)
  Solve lower triangular system : L^r x̃ = −c_J
  Solve upper triangular system : (L^r)^T x^r = x̃
  Set p = 0
  for i = 1 to length(J) do
    if |x_i^r| < 1e^−14 then
      I = I ∪ J(i)
    else if x_i^r < 0 then
      I = I ∪ J(i)
      p = p + 1
    else
      x_J = x_i^r
    end if
  end for
end while
x_I = 0
λ_I = M_{I,:} x + c_I
```

### 5.3.5. Updating/downdating Cholesky factorizations

In this work we further improve the performance of the projected Newton method by re-using factorizations. Note that successive Newton step computations involve reduced Hessian obtained by deleting rows and columns associated to the working

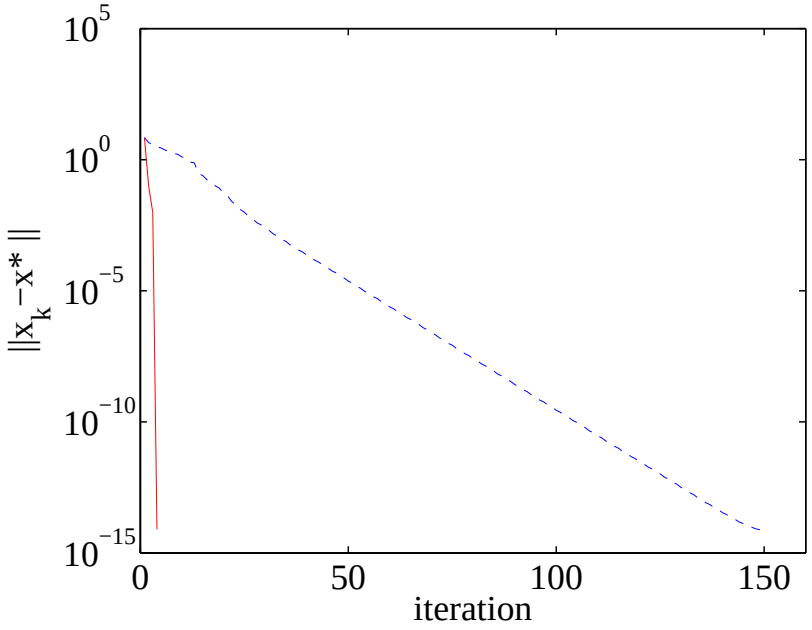



**Figure 5.3.** The convergence improvement of the projected Newton step of the gradient projection method is illustrated. Like in the unconstrained case, the gradient descent method typically converges in a zig-zag path yielding slow convergence (dashed line). By adding a Newton step with quadratic convergence rate, the performance of the algorithm can be drastically improved (solid line). The plot was generated for a non-negativity constrained QP of dimension 50 with 31 active constraints. The method converged in less than 5 iterations.

set. The Cholesky decomposition can be updated to incorporate low rank additions or downdated for low rank subtractions [163].

### Deleting a row and column

Deleting row and column $k$ of a matrix actually means setting all entries to zero except for the diagonal entry. which is set to an arbitrar value [40]. The original factorization can be written as

$$\begin{bmatrix} \mathbf{C}_{11} & \mathbf{c}_{12} & \mathbf{C}_{31}^{\mathrm{T}} \\ \mathbf{c}_{12}^{\mathrm{T}} & c_{22} & \mathbf{c}_{32}^{\mathrm{T}} \\ \mathbf{C}_{31} & \mathbf{c}_{32} & \mathbf{C}_{33} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & & \\ \mathbf{l}_{12}^{\mathrm{T}} & l_{22} & \\ \mathbf{L}_{31} & \mathbf{l}_{32} & \mathbf{L}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathrm{T}} & \mathbf{l}_{12} & \mathbf{L}_{31}^{\mathrm{T}} \\ & l_{22} & \mathbf{l}_{32}^{\mathrm{T}} \\ & & \mathbf{L}_{33}^{\mathrm{T}} \end{bmatrix} \tag{5.45}$$

After deleting row and column $k$, we have

$$\begin{bmatrix} \mathbf{C}_{11} & \mathbf{0} & \mathbf{C}_{31}^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & \rho^2 & \mathbf{0}^{\mathrm{T}} \\ \mathbf{C}_{31} & \mathbf{0} & \mathbf{C}_{33} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & & \\ \mathbf{0}^{\mathrm{T}} & \rho & \\ \mathbf{L}_{31} & \mathbf{0} & \bar{\mathbf{L}}_{33} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathrm{T}} & \mathbf{0} & \mathbf{L}_{31}^{\mathrm{T}} \\ & \rho & \mathbf{0}^{\mathrm{T}} \\ & & \bar{\mathbf{L}}_{33}^{\mathrm{T}} \end{bmatrix} \tag{5.46}$$

Hence, we only need to set row and column $k$ of $\bar{\mathbf{L}}$ to zero, set the diagonal entry to $\rho$ and compute $\bar{\mathbf{L}}_{33}$.

For this term, the original factorization is

$$\mathbf{L}_{33}\mathbf{L}_{33}^{\mathrm{T}} = \mathbf{C}_{33} - \mathbf{L}_{31}\mathbf{L}_{31}^{\mathrm{T}} - \mathbf{l}_{32}\mathbf{l}_{32}^{\mathrm{T}} \tag{5.47}$$

while the new factorization is

$$\bar{\mathbf{L}}_{33}\bar{\mathbf{L}}_{33}^{\mathrm{T}} = \mathbf{C}_{33} - \mathbf{L}_{31}\mathbf{L}_{31}^{\mathrm{T}} \tag{5.48}$$

Combining these equations, we observe that deleting a row and column is equivalent to a rank-1 update

$$\bar{\mathbf{L}}_{33}\bar{\mathbf{L}}_{33}^{\mathrm{T}} = \mathbf{L}_{33}\mathbf{L}_{33}^{\mathrm{T}} + \mathbf{w}\mathbf{w}^{\mathrm{T}} \tag{5.49}$$

with $\mathbf{w} = \mathbf{l}_{32}$.

Several methods for rank modifications of a Cholesky have been proposed in the literature. A review can be found in [69]. Here we disucss one of the most stable techniques, due to Stewart and available as the `dchud` routine in `LINPACK` [48]. See [163] for a more detailed discussion.

The idea is to apply Givens rotations $J_k$ (see Appendix B) to the augmented matrix

$$\mathbf{J}_n \dots \mathbf{J}_1 \begin{bmatrix} \mathbf{L}^{\mathrm{T}} \\ \mathbf{w}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{L}}^{\mathrm{T}} \\ \mathbf{0} \end{bmatrix} \tag{5.50}$$

leading to $\bar{\mathbf{L}}\bar{\mathbf{L}}^{\mathrm{T}} = \mathbf{L}\mathbf{L}^{\mathrm{T}} + \mathbf{w}\mathbf{w}^{\mathrm{T}}$. If a rotation results in $\bar{\mathbf{L}}_{kk} < 0$, we simply flip signs of the Givens rotation factors $c_k := -c_k$ and $s_k := -s_k$. The computation takes roughly $O(n^2)$ flops (compared to $O(n^3)$ for a full factorization).

**Algorithm 15.** *[Row and column deletion]*

$I = \{i | x_i^c = 0\}$
$\mathcal{J} = \{1, \dots, n\}$
***for*** $i = 1$ *to* length$(I)$ ***do***

$w = L(I(i)+1:\text{end}, I(i))$
$L(I(i), 1:I(i)-1) = 0$
$L(I(i)+1:n, I(i)) = 0$
$L(I(i), I(i)) = 1$
$L(I(i)+1:n, I(i)+1:n) = \textbf{rank}-\textbf{1 update}(L(I(i)+1:n, I(i)+1:n), w)$
***end for***

**Algorithm 16.** *[Rank-1 update]*

***for*** $i = 1$ *to* $n$ ***do***
  ***if*** $L(i,i) \neq 0$ ***then***
    ***if*** $|w(i)| > |L(i,i)|$ ***then***
      $t = -L(i,i)/w(i)$
      $s = 1/\sqrt{1+t^2}$
      $c = st$
    ***else***
      $t = -w(i)/L(i,i)$
      $c = 1/\sqrt{1+t^2}$
      $s = ct$
    ***end if***
    $a = cL(i,i) - sw(i)$
    ***if*** $a < 0$ ***then***
      $c = -c$
      $s = -s$
      $a = cL(i,i) - sw(i)$
    ***end if***
    $b = sL(i,i) + cw(i)$
    $L(i,i) = a$
    $w(i) = b$
    ***for*** $j = i+1$ *to* $n$ ***do***
      $a = cL(i,i) - sw(i)$
      $b = sL(i,i) + cw(i)$
      $L(j,i) = a$
      $w(j) = b$
    ***end for***
  ***end if***
***end for***

We apply these updates in the projected Newton method where in every iteration one

or multiple rows are deleted. Afterwards, the updated factors are used to solve

$$\bar{\mathbf{L}}^{\mathrm{T}} x' = -c \tag{5.51}$$
$$\bar{\mathbf{L}} x = -x' \tag{5.52}$$

At the higher level, we apply rank-1 updates to remove the inactive constraints from the active-set upon convergence of the Schur complement active-set method (see Figure 5.1).

### Adding rows and columns

As we will show below, adding a row and column to a matrix corresponds to a negative rank-1 modification or downdate $\bar{\mathbf{L}}\bar{\mathbf{L}}^{\mathrm{T}} = \mathbf{L}\mathbf{L}^{\mathrm{T}} - \mathbf{w}\mathbf{w}^{\mathrm{T}}$. This connection between modification of a matrix and modification of the factorization is somewhat counterintuitive. Removing elements from a matrix correspond to an update of the factorization, while adding elements corresponds to a downdate.

In the context of our Schur complement active-set method, multiple rows and columns are added in every iteration. Hence, we are interested in block addition of the existing Cholesky factor of the reduced Hessian.

Consider an existing factorization

$$\mathbf{C}_{11} = \mathbf{L}_{11}\mathbf{L}_{11}^{\mathrm{T}} \tag{5.53}$$

and suppose it is extended as follows

$$\begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{21}^{\mathrm{T}} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & \\ \bar{\mathbf{L}}_{21} & \bar{\mathbf{L}}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{L}_{11}^{\mathrm{T}} & \bar{\mathbf{L}}_{21}^{\mathrm{T}} \\ & \bar{\mathbf{L}}_{22}^{\mathrm{T}} \end{bmatrix} \tag{5.54}$$
$$= \begin{bmatrix} \mathbf{L}_{11}\mathbf{L}_{11}^{\mathrm{T}} & \mathbf{L}_{11}\bar{\mathbf{L}}_{21}^{\mathrm{T}} \\ \bar{\mathbf{L}}_{21}\mathbf{L}_{11}^{\mathrm{T}} & \bar{\mathbf{L}}_{21}\bar{\mathbf{L}}_{21}^{\mathrm{T}} + \bar{\mathbf{L}}_{22}\bar{\mathbf{L}}_{22}^{\mathrm{T}} \end{bmatrix} \tag{5.55}$$

Hence the new factors can be computed as follows

$$\mathbf{L}_{21} = \mathbf{C}_{21}\mathbf{L}_{11}^{-\mathrm{T}} \tag{5.56}$$

$$\mathbf{L}_{22}\mathbf{L}_{22}^{\mathrm{T}} = \mathbf{C}_{22} - \mathbf{L}_{21}\mathbf{L}_{21}^{\mathrm{T}} \tag{5.57}$$

The latter is indeed a downdate. It can be calculated using hyperbolic rotations potentially in combination with Givens rotations for numerical stability, we refer to

[21] for a detailed treatment. In our case, we do not have a factorization of $\mathbf{C}_{22} = \mathbf{L}_{22}\mathbf{L}_{22}^{\mathrm{T}}$. Therefore, it makes more sense to compute the Cholesky factorization of $\mathbf{C}_{22} - \mathbf{L}_{21}\mathbf{L}_{21}^{\mathrm{T}}$. Since the subtraction can give negative diagonal entries, a check is made before executing the factorization.

### 5.3.6. Computational burden

Let $n_A$ denote the number of inequalities in the active set and $n_{\mathrm{it}}$ the number of (outer) active-set iterations. Then the computational burden is summarized in Table 5.1. Here we subdivide the work needed for solving the unconstrained problem and the work needed for the constrained solution, i.e. the iterations of the active-set method.

**Table 5.1.** Overview of the computational burden subdivided into the unconstrained problem (unc) and the constrained problem (con). The operations are: a Riccati recursion, a forward vector solve (fsolve), a partial forward vector solve (partial fsolve),a backward vector solve (bsolve) and solving a reduced QP (rQP).

| | Riccati | fsolve | partial fsolve | bsolve | rQP |
|---|---|---|---|---|---|
| unc | 1 | 1 | 0 | 1 | 0 |
| con | 0 | 0 | $n_{\mathrm{A}}$ | $n_{\mathrm{it}}$ | $n_{\mathrm{it}}$ |
| total | 1 | 1 | $n_{\mathrm{A}}$ | $n_{\mathrm{it}}+1$ | $n_{\mathrm{it}}$ |

## 5.4. Numerical examples

### 5.4.1. Waste water treatment process

Consider again the waste water treatment problem presented in Section 4.4.2. The state and disturbance estimates are identical (up to numerical accuracy) to those obtained with the interior-point MHE method and were shown and discussed in Chapter 4. Hence, we will only compare the performance of both algorithms and discuss the working of the Schur complement active-set method. The computation times are shown in Figure 5.4. It can be seen that the algorithm needs only 0.2 ms if an horizon of 10 is used. Hence, the algorithm is about a factor 2 faster than a comparable interior-point method, see Section 4.4.2.

Figure 5.5 shows that the algorithm typically needs only 2 or 3 active-set iterations. Once the number of active constraints stabilizes, also the number of constraints in the final working set stabilizes around the same number.

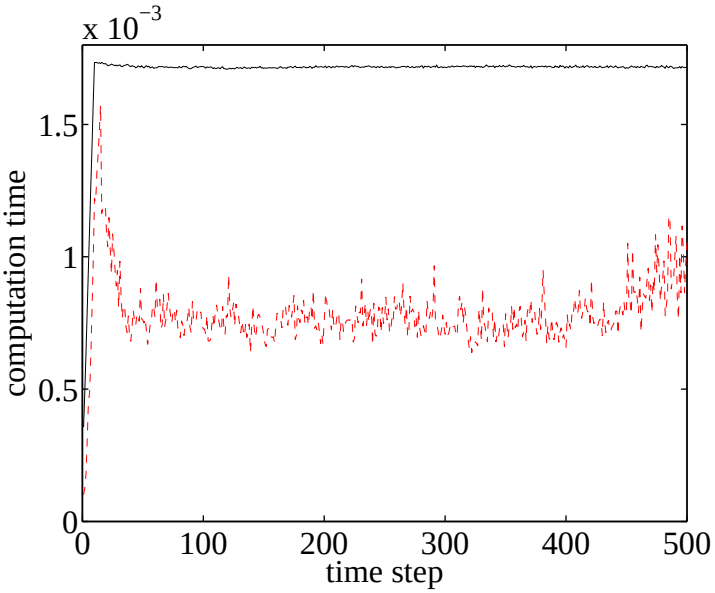



**Figure 5.4.** Computation times in seconds for the waste water treatment application and MHE with horizon 10. The Schur complement active-set MHE method (dashed line) is about a factor 2 faster than the interior point method using ten iterations (solid line).

## 5.5. Conclusions

In this chapter a Schur complement active-set method was presented. It uses as the starting point the unconstrained MHE solution which can be computed efficiently using a square-root Riccati based algorithm. By projecting onto the reduced space of active constraints a reduced non-negativity constrained QP is obtained. For this, a gradient-projection method using projected Newton steps and Cholesky factorizations is proposed. Cholesky updates are employed in the projected Newton iterations. Once a solution to the reduced QP is found, it is used to update primal and dual variables using a Schur complement technique. The method allows multiple updates to the active set per iteration. Between (outer) active set iterations the reduced Hessian changes by adding some constraints to the working set. These changes are exploited by a Cholesky downdate. The performance of the algorithm was demonstrated by application of a C implementation to some numerical examples.

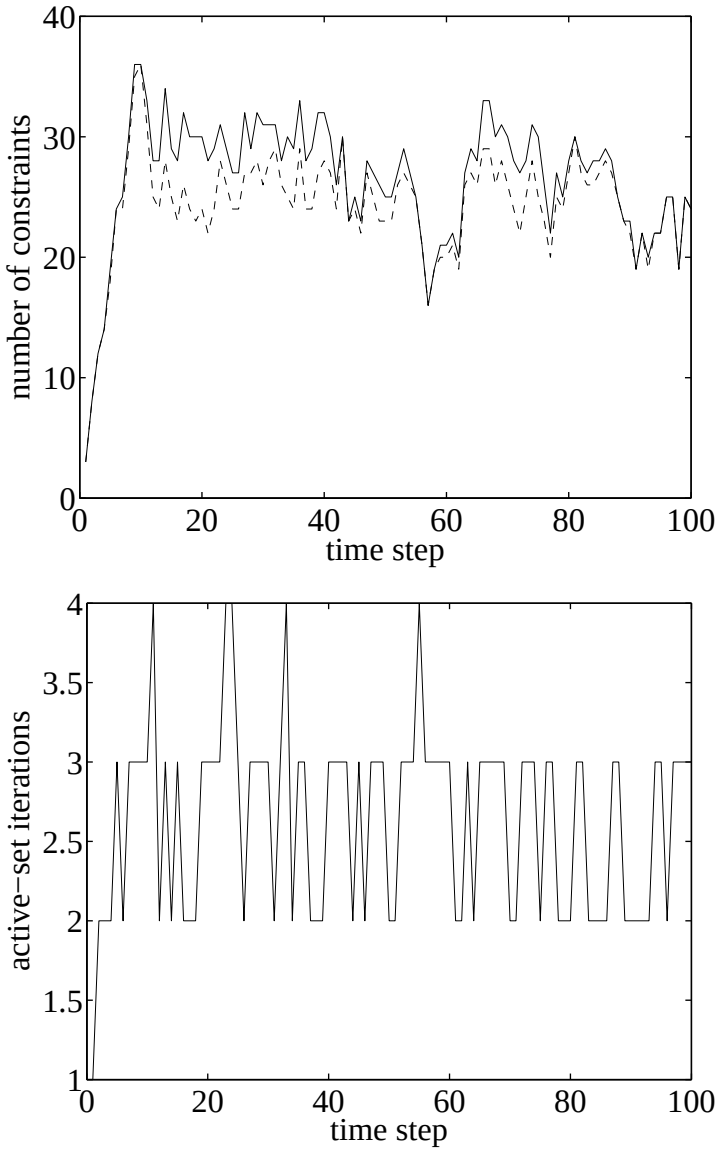



**Figure 5.5.** Schur complement active-set method applied to the waste water treatment process for MHE with horizon 10. **Top:** Number of constraints in the final working set (solid line) and number of active constraints (dashed line). **Bottom:** Number of (outer) active-set iterations.

CHAPTER 6

# Convex MHE formulations

*The focus in this chapter is on two types of robust convex MHE formulations which are particularly useful in practical applications. First, robustness with regards to occasional outliers is investigated by means of Huber penalty MHE and $\ell_1$ penalty MHE. The Huber formulation is shown to have excellent performance in terms of outlier rejection and estimation accuracy. Second, the joint estimation of states and parameters or inputs is considered. The resuling MHE problem is formulated as a convex cardinality problem yielding robustness with respect to rapid parameter changes, i.e. jumps or break points. It is shown that this leads to an MHE problem with $\ell_1$ penalty on the parameter variation and a small number of subsequent corrections to the $\ell_1$ norm MHE problem. Significant improvements in estimation performance are obtained using this procedure and a polishing step.*

## 6.1. Robust estimation using Huber penalty function

Traditionally, state estimators are based on a least-squares penalization of residuals. For linear unconstrained systems, this leads to the celebrated Kalman filter. For constrained and/or nonlinear systems, moving horizon estimation (MHE) [44, 59, 89, 108, 134, 146, 155, 157, 202], has emerged as an attractive alternative. In MHE, a finite horizon optimization problem is solved in every time step. Past data outside the window is summarized in a so-called *arrival cost*. When a new measurement becomes available, the arrival cost is updated, the window is shifted and the process is repeated.

The least-squares approach, however, is not always suitable when the process is characterized by structural defects in the model or by imperfect measurements. In such

cases, robust methods, which are less sensitivive to large errors, are desirable. In robust statistics, estimators involving explicit or recursive optimization over (robust) penalty functions are referred to as *M-estimators* [103]. According to Zhang [205] a robust esimator should satisfy the following specifications: (1) have a bounded influence function, *i.e.* derivative of the penalty function, and (2) return unique estimates, which implies that the norm function should be strictly convex. The $\ell_1$ norm is such robust measure. However, in the least absolute deviation or $\ell_1$ approach, gross errors can still have a significant impact on the estimates as they are given equal weight as small residuals. A generalization of this is the least powers method or $\ell_p$ approach, using functions $|u|^p$ which are convex for $p \geq 1$ [29]. The selection of an optimal value of $p$ for robust estimation has been ivestigated, and for $p$ around 1.2 good estimates may be expected [154, 205].

Unfortunately, both the least absolute deviation and the least powers approach tend to produce more zero residuals than can be statistically explained in many cases. These drawbacks have motivated research into even more robust approaches.

*Hybrid* $\ell_1$ - $\ell_2$ combine robust treatment of large residuals with Gaussian treatment of small residuals. The Huber penalty function, introduced in 1973 by Peter Huber [102], is one such hybrid $\ell_1$ - $\ell_2$ norm. It has been found very practical for robust estimation by several authors in certain areas.

For example, in geophysics, Guitton and Symes [29, 84, 85] have applied it to seismic data represented by a linear regression model, *i.e.* a robust *inverse problem*

$$\min_x \|Ax - b\|_{\text{huber}} \tag{6.1}$$

instead of the standard least-squares problem

$$\min_x \|Ax - b\|_2 \tag{6.2}$$

The authors do not employ the QP reformulation (see Section 6.1.2), but instead directly apply standard nonlinear optimization to the Huber function. Since the Huber function is not twice differentiable, the convergence of any Newton method might be jeopardized. Nevertheless, the authors propose a quasi-Newton method using limited-memory BFGS updates and report satisfying results.

In the area of power engineering, Kyriakides et al [122] have applied the Huber penalty to estimate the parameters of a synchronous generator using a linear regression model with structural defects, *i.e.* rank deficiency, in the process matrix. They present a statistical test and conclude that the Huber method outperforms the least-squares method especially when several parameters are unknown. Jabr [104] applied the Huber norm in the context of power system state estimation with output residual penalization only. The author derives the quadratic reformulation and applies a

primal-dual interior-point method for offline state estimation with equality and inequality constraints. The method is applied it to a network model of IEEE bus test systems. A posterior analysis of the performance and ability to detect outliers of the method is performed for two fixed values of tuning parameter for the Huber function.

Robust model identification using the $\ell_1$ norm and the Huber function with application to type 1 diabetes modelling has recently been presented by Finan et al [58].

Wang et al [193] present a data dependent heuristic for determining the optimal tuning parameter for the Huber penalty and demonstrate their method on some robust regression examples.

Estimation problems using Huber penalty function have been approximated often throughout the literature by iteratively reweighted least-squares (IRLS), which avoids explicit optimization and thereby allowed its application to large offline problems or to online problems. With the advances in numerical optimization and increasing computing power, it has been applied using optimization methods in more recent years, although applications in state estimation are rare and no publications of Huber-based MHE are known to the author. The aim of the work presented in this chapter is twofold: first, we show that the use of Huber penalty functions in online estimation can yield an estimator with excellent robustness with regards to outliers and can be used to identify and reject otuliers, and second, we show that Huber penalty MHE can be solved efficiently using (square-root) Riccati based methods in combination with interior-point (Chapter 4) or active-set (Chapter 5) methods allowing computation times comparable to standard MHE.

### 6.1.1. Robust moving horizon estimation

Let us consider the general batch estimation problem introduced in Section 1.4.2.

$$\begin{aligned} \min_{\mathbf{x},\mathbf{w},\mathbf{v}} \quad & \rho(S_0^{-T}(x_0-\hat{x}_0)) + \textstyle\sum_{k=0}^{T-1} \rho(W_k^{-T} w_k) + \sum_{k=0}^{T} \rho(V_k^{-T} v_k) \\ \text{s.t.} \quad & \begin{aligned} x_{k+1} &= f_k + A_k x_k + G_k w_k, \quad && k = 0,\ldots,T-1, \\ y_k &= h_k + C_k x_k + v_k, && k = 0,\ldots,T, \end{aligned} \end{aligned} \tag{6.3}$$

where the common least-squares terms are replaced with arbitrary penalty functions $\rho(\cdot)$ not necessarily identical for every term or for every $k$. $S_0$, $W_k$ and $V_k$ are weighting matrices.

In case $\rho(\cdot)$ is the squared $\ell_2$ norm and the weighting matrices are chosen as the Cholesky factors of $P_0$, $Q_k$ and $R_k$ respectively, the above estimation problem equals the standard least-squares batch estimation problem (1.13).

The problem (6.3) can be complemented with constraints, see Sections 4 and 5.

Traditional least squares estimation or smoothing ignores measurement anomalies and therefore can produce biased estimates in the presence of outliers. Robust estimators relying on Huber penalty functions or other hybrid $\ell_1$ - $\ell_2$ type penalty functions are less affected by extreme values and provide heuristics for identifying outliers. The main purpose of the work presented here is to show that MHE with good robustness properties may be obtained in a computational time comparable to the least squares approach. In particular, the presented robust estimator leads to a convex optimization problem for which a custom method was developed that scales linearly with the horizon length.

Several robust measures including the Huber penalty are discussed in more detail in Appendix C. From this discussion it follows that the Huber penalty is a robust and stable measure. In contrast, the $\ell_2$ norm is not robust and the $\ell_1$ norm is not stable. Furthermore, the Huber penalty is the best convex approximation to the quadratic-constant penalty which is the most robust measure. in this chapter we present a Huber based MHE method and compare it to the standard least-squares formulation and an $\ell_1$ based MHE method.

### 6.1.2. The Huber penalty function

The Huber penalty function was first proposed by and therefore named after American statistician Peter Huber ([102]). It is given by

$$\rho(u) = \begin{cases} u^2 & |u| \leq M \\ M(2|u| - M) & |u| > M, \end{cases} \tag{6.4}$$

Note that the Huber function is *not* a norm, because the triangle inequality is not satisfied.

The Huber penalty function has been found very practical for robust estimation. This penalty function agrees with $\ell_2$ for $u$ smaller than $M$, and then reverts to $\ell_1$-like linear growth for larger $u$. For a constant outlier contamination rate, the choice of tuning parameter $M$ does not influence the estimation result much. If required, the tuning parameter can be simultaneously optimized to improve the estimator performance. The 95% asymptotic efficiency on the standard normal distribution[1] is obtained with a tuning constant $M = 1.345$ [103, 205].

We cannot directly apply Newton's method to the estimation problem (6.3) if non-smooth measures like *e.g.* the Huber penalty function, are employed. However, the

[1] Meaning that 95% of the data is weighted appropriately in case the data is standard normally distributed.

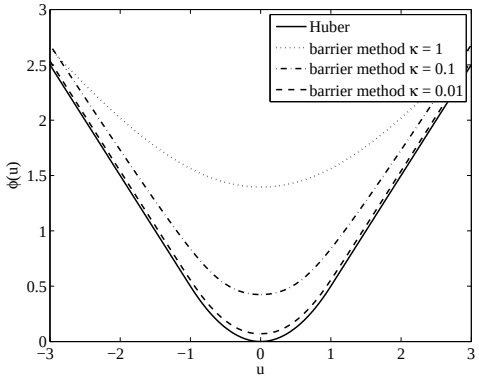



**Figure 6.1.** Approximation of the Huber function with parameter $M = 1$ by a barrier function with different values for the barrier parameter $\kappa$.

problem of optimizing a variable in Huber sense can be reformulated as a quadratic program (QP). Consider a Huber penalized variable[2] $\|u\|_{\text{huber}}$ and introduce (scalar) auxiliary variables $\alpha$ and $\beta$ such that

$$\mathcal{D} = \{(u,\alpha,\beta)\,|\,-(\alpha+\beta) \le u \le (\alpha+\beta), 0 \le \alpha \le M, 0 \le \beta\}.$$

Then

$$\|u\|_{\text{huber}} = \inf_{\mathcal{D}}(\alpha^2 + 2M\beta)$$

Hence, minimizing $\|u\|_{\text{huber}}$ is equivalent to the following QP

$$\begin{array}{ll} \min_{u,\alpha,\beta} & \alpha^2 + 2M\beta \\ \text{s.t.} & -(\alpha+\beta) \le u \le (\alpha+\beta), \\ & 0 \le \alpha \le M, \\ & 0 \le \beta. \end{array} \tag{6.5}$$

This QP can be solved with interior-point methods or active-set methods. First, let us consider a primal barrier method (see Chapter 4 or references [29, 136, Chap. 11]). Replacing the inequality constraints with a barrier term in the objective results in the approximate problem

$$\min_{u,\alpha,\beta} \quad \alpha^2 + 2M\beta + \kappa\phi(u,\alpha,\beta) \tag{6.6}$$

[2] Note that the penalization of a variable, i.e. $\|u\|_{\text{huber}}$ by itself is not very useful. The trivial example is given here to simplify the derivation and illustrate the variables involved. It becomes useful when it is employed in an optimization problem, for example $\min_x \|Ax-b\|_{\text{huber}}$

where $\kappa > 0$ is a barrier parameter, and $\phi$ is the log barrier associated with the inequality constraints, defined as

$$\phi(u,\alpha,\beta) = \begin{cases} -\log(\beta) - \log(\alpha(M-\alpha)) - \log(-u-\alpha-\beta) - \log(u-\alpha-\beta), & (u,\alpha,\beta) \in \mathcal{D} \\ \infty & \text{otherwise.} \end{cases}$$

As $\kappa$ approaches zero, the solution of (6.6) converges to the true Huber penalty function, as depicted in Figure 6.1. In a basic primal barrier method, we solve a sequence of problems of the form (6.6), using Newton's method starting from the previously computed point, for a decreasing sequence of values of $\kappa$. A typical strategy is to reduce $\kappa$ by a factor of 10 each time a solution of (6.6) is computed (within some accuracy). See Chapter 4 for a more detailed discussion of interior-point methods.

Alternatively, the QP (6.10) can be solved using the Schur-complement active-set method of Chapter 5. The Huber penalty implies two additional variables and five additional linear constraints of which three are bound constraints.

### 6.1.3. The multivariate Huber penalty function

The huber penalty is typically applied componentwise. However, this assumes the components are independent, *i.e.* diagonal covariance or weighting. As stated by Huber *univariate techniques should not be applied to multivariate data because of the existence of correlations between the variables* [103]. Sometimes multivariate outliers are simply not detectable by univariate techniques [97].

The multivariate formulation of the Huber penalty for a vector $x \in \mathbf{R}^n$ is given as

$$\rho_{\text{mvhuber}}(x) = \begin{cases} \|x\|_2^2 & \|x\|_2 \leq M \\ M(2\|x\|_2 - M) & \|x\|_2 > M. \end{cases} \tag{6.7}$$

Minimizing $\rho_{\text{mvhuber}}(x)$ is equivalent to solving an second order cone program (SOCP) (see Section 2.3.6)

$$\begin{array}{ll} \min_{x,\alpha,\beta} & \|\alpha\|_2^2 + 2M\beta \\ \text{s.t.} & \|x-\alpha\|_2 \leq \beta, \\ & \|\alpha\|_2 \leq M, \\ & \alpha \geq 0, \\ & \beta \geq 0, \end{array}$$

with auxiliary variables $\alpha \in \mathbf{R}^n$ and $\beta \in \mathbf{R}$.

This multivariate or circular Huber penalty has been formulated by other researchers, *e.g.* [9, 28]. It is also available as a function in `cvx`, a Matlab frontend for disciplined convex programming which uses SeDuMi or SDPT3 as underlying convex solver, developed by Grant, Boyd and Ye [81, 82].

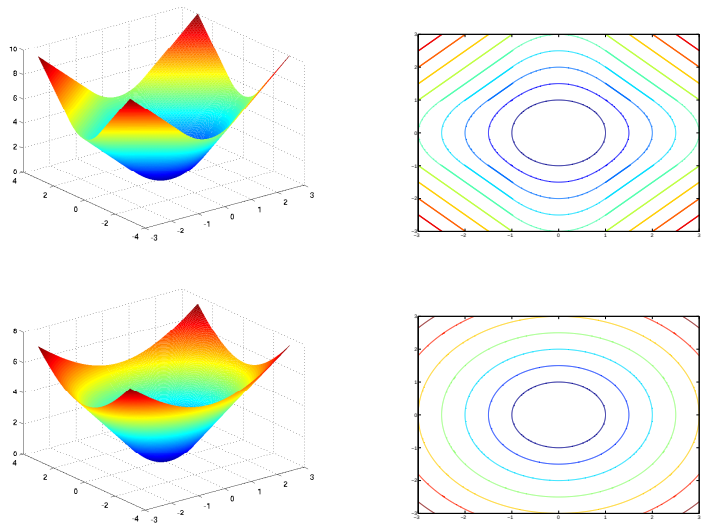


**Figure 6.2.** Graphical representation of the Huber penalty function applied componentwise (top figures) versus the multivariate Huber penalty function (bottom figures)

A graphical representation of the componentwise versus the multivariate Huber function is given in Figure 6.2.

Figure 6.2 shows the computation times in seconds using `cvx` for evaluating the multivariate Huber function and the componentwise univariate Huber function for varying vector lengths. Computing times correspond to an Intel Core2-Duo processor at 2.13 GHz with 2 MB cache and 2 GB RAM, running Matlab version R2010b and `cvx` version 1.21. For both problems the underlying solver is SeDuMi. It can be seen that the multivariate form can be motivated not only by statistical arguments but also by computational complexity. For a vector of length one, the computation times for the SOCP and for the QP are comparable. However, for vectors of length $n$, the multivariate form involves the solution of one optimization problem with $n+1$ variables while the componentwise application of the univariate form involves the solution of $n$ optimization problems with 2 variables. This implies a linear scaling of the latter with vector length while the complexity of the former is almost unaffected by the vector length, as can be seen from Figure 6.2.

### 6.1.4. Selecting the tuning parameter

As noted before, the value $M = 1.345$ yields 95% asymptotic efficiency on the standard normal distribution, so it is an obvious choice. As we will show in the numerical

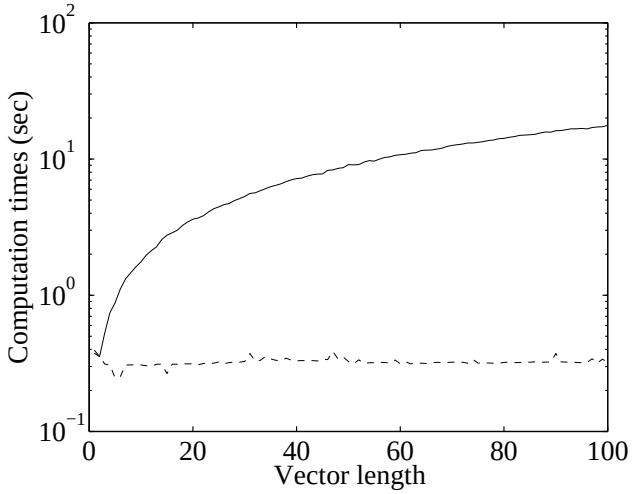



**Figure 6.3.** Computation times in seconds using `cvx` for evaluating the multivariate Huber function (dashed line) and the componentwise univariate Huber function (solid line) for varying vector lengths.

examples below, the optimal tuning parameter may change depending on the data. We investigate two methods for selecting the optimal tuning parameter.

A first method is obtained by simultaneously optimizing the tuning parameter. The resulting optimization problems are still convex, *i.e.* QP and SOCP for the univariate and multivariate Huber functions respectively.

A second method was proposed by Wang et al [193] and is based on statistical properties of the Huber function. They define an efficiency factor based on the influence function, which was also proposed by other researchers including Huber [103], and derive a data driven estimator for this efficiency factor. The value of tuning factor yielding a maximum efficiency factor is selected. However, since the estimator function is non-smooth, a grid search over a number of tuning parameter values is suggested.

### 6.1.5. Huber penalty MHE

Problem (6.6) is a convex optimization problem with smooth objective. Employing this approximate Huber penalty function in problem (6.3) again leads to an equality constrained convex program that can be solved efficiently using Newton's method. Consider the problem of state estimation with outlier contaminated measurements.

Then the batch estimation problem can be formulated as

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{w},\mathbf{v}} & \|S_0^{-T}(x_0-\hat{x}_0)\|_2+\sum_{k=0}^{T-1}\|W_k^{-T}w_k\|_2+\sum_{k=0}^{T}\|V_k^{-T}v_k\|_{\text{huber}} \\ \text{s.t.} & x_{k+1}=f_k+A\,x_k+G\,w_k, \quad k=0,\ldots,N-1, \\ & y_k=h_k+C_k\,x_k+v_k, \quad k=0,\ldots,N. \end{array}$$

This problem can be reformulated as a QP using the result of Eq. (6.10) and solved *e.g.* by an interior point method.

Similarly, if the multivariate Huber penalty is used, the problem can be formulated with two second order constraints per penalized vector, and solved as an SOCP.

### 6.1.6. Smooth hybrid $\ell_1$ - $\ell_2$ MHE

Another hybrid $\ell_1$ - $\ell_2$ measure which approximates the Huber penalty is given by

$$\rho_{\text{sh}}(u)=M(\sqrt{M^2+u^2}-M), \tag{6.8}$$

and is referred to as *smooth hybrid penalty function*.

Optimization over this penalty function also yields an SOCP

$$\min_u \rho_{\text{sh}}(u) = \quad \min_{u,t} t \quad \text{s.t.} \left\| \begin{bmatrix} 1 & \frac{u}{M} \end{bmatrix} \right\|_2 - M^2 \leq t. \tag{6.9}$$

### 6.1.7. $L1$ norm MHE

Another robust method for state estimation in the presence of outlier contaminated measurements is by $\ell_1$ norm penalization

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{w},\mathbf{v}} & \|S_0^{-T}(x_0-\hat{x}_0)\|_2+\sum_{k=0}^{T-1}\|W_k^{-T}w_k\|_2+\sum_{k=0}^{T}\|V_k^{-T}v_k\|_1 \\ \text{s.t.} & x_{k+1}=f_k+A\,x_k+G\,w_k, \quad k=0,\ldots,N-1, \\ & y_k=h_k+C_k\,x_k+v_k, \quad k=0,\ldots,N. \end{array}$$

Minimizing the $\|u\|_1$ norm is equivalent to solving the following LP

$$\begin{array}{ll} \min_{u,t} & t \\ \text{s.t.} & -u \leq t, \\ & u \leq t. \end{array} \tag{6.10}$$

Hence, the MHE formulation is adapted by $(T+1)n_y$ additional variables with associated linear terms (instead of quadratic) in the objective and twice this number additional bound constraints.

### 6.1.8. Numerical example

In order to to evaluate the robustness of Huber based MHE and investigate the influence of the tuning parameter $M$, we compare the mean squared errors of standard least-squares MHE, Huber penalty MHE and $\ell_1$ norm MHE. In our simulations, we consider a fixed random LTI system of order 4 and random noise realizations. The system is disturbed with normally distributed process noise $w$ and output noise $v$ with covariances respectively $Q = 0.01I_4$ and $R = 1$. We assume that the measurement noise is contaminated with added sparse noise, *i.e.* occasional peaks due to sensor failure. Two contamination types are considered: two-point contamination which yields values $-7$ or 7 with equal probability of $\lambda/2$, and contamination with a distant normal distribution $\mathcal{N}(0,7)$. Different contamination rates are considered: $\lambda = 1\%$, 5%, 10%, 20%, 30% and 40%.

The results on independent test sets for batch estimation of 200 samples (in both training and test set) are shown in Table 6.1. It can be seen that the Huber method with optimized tuning constant almost always outperforms the $\ell_2$ and $\ell_1$ methods. Note that the $\ell_1$ method performs well over a wide range of contamination rates, which is surprising especially for the low contamination rates, and may be attributed to the fact that the weighting of the $\ell_1$ norm is low compared to the other terms, *i.e.* the initial condition and the process disturbance terms. Note also that the Huber method performs well for a wide range of tuning parameter values, but in the case of two-point contamination a slight improvement is achieved by optimal tuning of the value.

Next, let us analyse the capability of the robust Huber MHE method to detect outliers and compare this with the robust smooth hybrid MHE, which should yield similar performance but is characterized by a smooth transition between $\ell_1$ and $\ell_2$ behavior.

Consider thereto the same random system as before, contaminated with normally distributed sparse noise sampled from $\mathcal{N}(0,5)$ with occurence rate 2.5%. Figure 6.4 shows a typical time evolution and probability density. Our simulations are for batch estimation with 500 samples and a prediction horizon of 50. The results are averaged over 100 runs.

The mean squared error is shown in Figure 6.5 for the standard least squares estimator, the Huber estimator and the smooth hybrid estimator. It is clear that the added robustness of the Huber penalty and the smooth hybrid penalty reflects in improved state estimation performance. Moreover, the Huber estimator outperforms the smooth hybrid estimator in a wide range of tuning parameter values.

It can be seen from Figures 6.6-6.8 that the ability to detect outliers is comparable for both methods. The smooth hybrid approach yields more false positives when the cut-off point is relatively small, which can be explained by the smooth influence function. On the other hand, it can be seen that for relatively large values of the tuning

**Table 6.1.** Mean squared errors (MSE) evaluated on an independent test set for (1) standard least-squares or $\ell_2$ norm MHE, (2) $\ell_1$ norm MHE, (3) Huber MHE with fixed tuning constant $M = 1.345$, (4) Huber MHE with optimized tuning constant, and (5) Huber MHE with tuning constant determined by the data-driven heuristic. Two types of outlier contamination are considered and the contamination rate $\lambda$ is varied from 1 % to 40 %.

| $\lambda$ | 1% | 5% | 10% | 20% | 30% | 40% |
|---|---|---|---|---|---|---|
| | | | (a) two-point contamination | | | |
| L2 | 0.033 | 0.046 | 0.050 | 0.080 | 0.213 | 0.129 |
| L1 | 0.038 | 0.038 | 0.042 | 0.042 | 0.049 | 0.027 |
| Huber $M = 1.345$ | 0.037 | 0.038 | 0.035 | 0.036 | 0.055 | 0.027 |
| Huber opt | 0.033 | 0.038 | 0.035 | 0.037 | 0.049 | 0.026 |
| $M_{opt}$ | 2.3 | 0.5 | 0.6 | 0.6 | 0.3 | 0.5 |
| Huber tuned | 0.033 | 0.037 | 0.035 | 0.038 | 0.069 | 0.026 |
| $M_{tuned}$ | 2.5 | 0.2 | 1.9 | 0.5 | 1.3 | 0.6 |
| | | | (b) contamination with N(0,7) | | | |
| L2 | 0.035 | 0.030 | 0.050 | 0.063 | 0.142 | 0.109 |
| L1 | 0.036 | 0.028 | 0.029 | 0.031 | 0.036 | 0.035 |
| Huber $M = 1.345$ | 0.033 | 0.028 | 0.028 | 0.032 | 0.034 | 0.033 |
| Huber opt | 0.032 | 0.028 | 0.029 | 0.032 | 0.034 | 0.033 |
| $M_{opt}$ | 0.7 | 0.5 | 0.6 | 0.6 | 0.6 | 0.5 |
| Huber tuned | 0.031 | 0.027 | 0.030 | 0.030 | 0.036 | 0.033 |
| $M_{tuned}$ | 1.2 | 0.2 | 0.3 | 0.2 | 0.2 | 0.3 |

parameter, the Huber approach generates more false negatives. However, at this point the ratio of correctly identified outliers is already below 65% and the mean squared error starts increasing, so we can argue that the method will not be operated in this region if the tuning parameter is set right.

From these results, it is clear that the outliers (peaks) can be identified quite accurately using a Huber penalty robust batch estimator. Furthermore, if the outliers are successfully removed, the noise will be Gaussian. This motivates the following strategy: use Huber penalty robust MHE in a first step to identify and remove the outliers, then in a second step use either standard MHE or Huber penalty MHE to further improve the quality of the estimates. The results are shown in Table 6.2. The Huber tuning parameter in the simulations was set to 1.345 and the results are averaged over 50 repetitions.

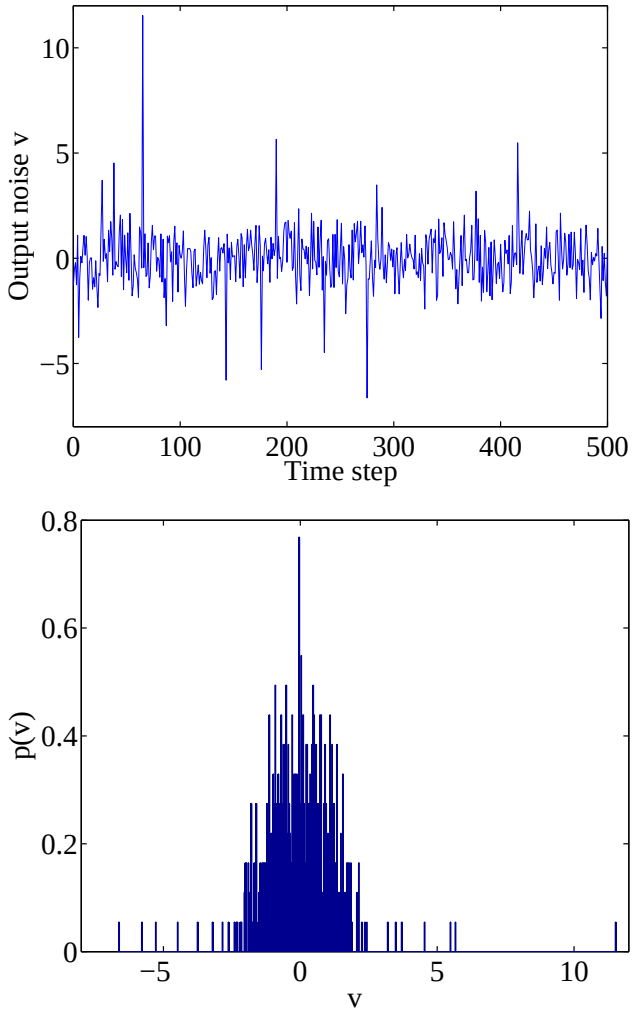



**Figure 6.4. Top panel:** Time evolution of a typical noise realization contaminated with normally distributed sparse noise $\mathcal{N}(0,7)$ with occurence rate 2.5%. **Bottom panel:** Corresponding distribution.

**Table 6.2.** Mean squared simulation and prediction errors (MSE) for batch estimation on 500 samples using different norms and contamination $\mathcal{N}(0,5)$ with occurence rate 2.5%. Prediction horizon is 50 samples and the results are averaged over 50 repetitions.

| Algorithm | Simulation | Prediction |
|---|---|---|
| standard BE | 0.0302 | 0.0486 |
| L1 norm robust BE | 0.0313 | 0.0653 |
| Huber norm robust BE | 0.0283 | 0.0529 |
| Huber norm robust BE + standard BE | 0.0256 | 0.0408 |

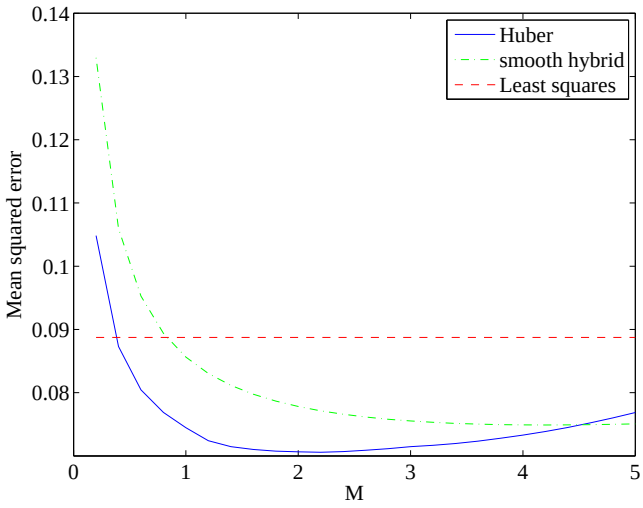



**Figure 6.5.** Influence of the tuning parameter $M$ on the mean squared error for batch estimation using Huber (solid blue line) and using the smooth hybrid penalty (dash-dotted green line). For comparison, the mean squared error of the standard least squares estimator is also shown (dashed red line).

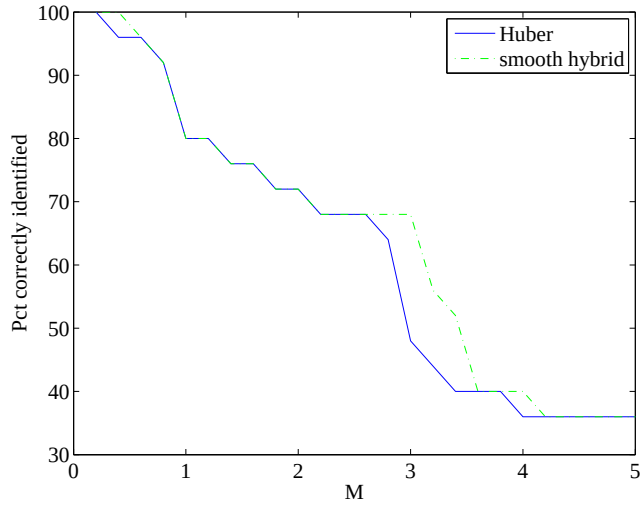



**Figure 6.6.** Percentage of correctly identified outliers as a function of the tuning parameter $M$ for batch estimation using Huber (solid blue line) and using the smooth hybrid penalty (dash-dotted green line).

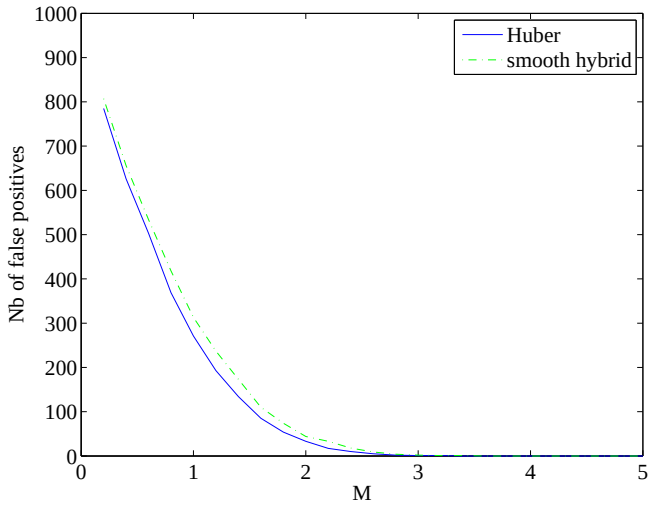



**Figure 6.7.** Number of false positives (Type I errors) as a function of the tuning parameter $M$ for batch estimation using Huber (solid blue line) and using the smooth hybrid penalty (dash-dotted green line).

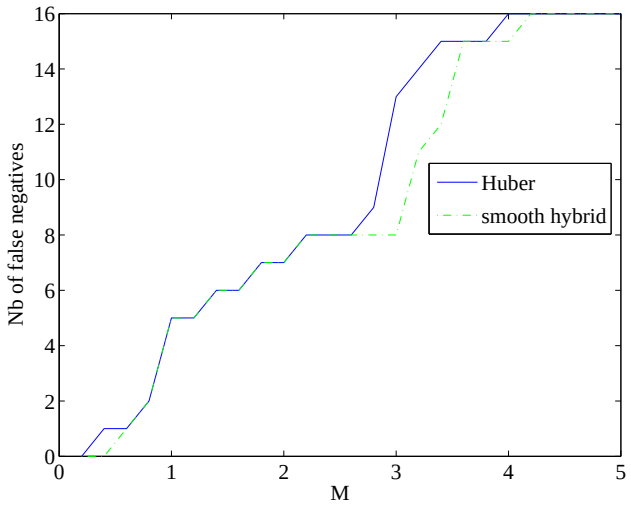



**Figure 6.8.** Number of false negatives (Type II errors) as a function of the tuning parameter $M$ for batch estimation using Huber (solid blue line) and using the smooth hybrid penalty (dash-dotted green line).

Throughout, we have assumed the weightings or covariances $Q$ and $R$ to be known. If they are unknown, they should be estimated from the data. A model based approach for estimating covariances from data was proposed by Odelson et al [137]. In the presence of contaminated data, a more robust covariance estimation method is desirable. In such a case, a median absolute value approach could be useful [193].

## 6.2. Joint input/parameter and state estimation

Recursive methods for simultaneous estimation of states and inputs have been considered in *a.o.* [72, 73, 114]. In order to permit a recursive formulation, the input variations are typically penalized in least-squares sense. However, any convex penalization of the input or parameter variation yields a convex MHE problem. Robust norms can be used for penalization of input or parameter variations. In this chapter, it is shown that abrupt changes in the parameters or inputs are appropriately modelled by robust penalizations of the variations instead of by the $\ell_2$ norm. In particular, it is shown that the joint input and state estimation problem with piecewise constant (or piecewise linear) inputs is best described by a cardinality problem which can be solved iteratively by a number of $\ell_1$ type MHE problems. It is further shown that these types of problems can also be solved by dedicated Riccati-based methods along the lines of Chapters 3-5. As such, an algorithm is obtained which allows fast detection of within-horizon parameter jumps and this is especially useful for moderate to large horizon lengths, where the standard MHE formulation smooths out these parameter switches.

Here we present an MHE algorithm for simultaneous estimation of parameters and states for systems with piecewise constant parameters or inputs. This robust formulation allows fast detection of parameter jumps and improves the quality of both parameter and state estimates compared to the standard MHE formulation, which smooths out sharp parameter variations.

### 6.2.1. Cardinality problems

Let us introduce the notion of cardinality of a vector $x \in \mathbf{R}^n$, *i.e.* the number of nonzero components

$$\text{card}(\mathbf{x}) = \sum_{i=1}^{n} \text{card}(x_i),$$

with

$$\text{card}(x_i) = \begin{cases} 0 & x_i = 0 \\ 1 & x_i \neq 0, \end{cases}$$

Some authors use the term $\ell_0$ norm instead of cardinality, *i.e.* $\|x\|_0 \equiv \text{card}(\mathbf{x})$. However, note that this function is not a norm because it is not homogeneous, *i.e.*

$$\|2e_1\|_0 = 1 \neq |2|\|e_1\|_0 = 2.$$

for a unit vector $e_1$.

It is not a convex function either because, for any $\alpha \in (0,1)$ and unit vectors $e_1$ and $e_2$ holds

$$\|(1-\alpha)e_1 + \alpha e_2\|_0 = 2$$

while

$$(1-\alpha)\|e_1\|_0 + \alpha\|e_2\|_0 = 1 - \alpha + \alpha = 1.$$

In optimization problems, the cardinality function can be useful as objective or as a constraint. For example, in sparse design the aim is to find the sparsest vector satisfying a set of specifications

$$\begin{array}{ll} \min_x & \text{card}(x) \\ \text{s.t.} & x \in \mathcal{C}. \end{array} \tag{6.11}$$

In sparse signal reconstruction or sparse regressor selection, the aim is to find the best fit to the observations using a combination of $M$ components/regressors

$$\begin{array}{ll} \min_x & J(x,\eta) \\ \text{s.t.} & \text{card}(x) \leq M. \end{array} \tag{6.12}$$

where $J$ is some (convex) objective and $\eta$ are the observations.

Both cardinality problems (6.11) and (6.12) are NP hard combinatorial problems. They can be solved globally by branch-and-bound problems or locally by convex approximations. We will only consider convex approximations here and focus on cardinality constraints since our interest is in signal reconstruction. Convex cardinality problems are discussed in more detail in [29] and [38].

Cardinality problems have recently caught increasing attention in the machine learning community. Many algorithms in machine learning are based on (approximate) convex cardinality problems, *i.e.* compressed sensing [49], sparse PCA [37, 204], and sparse support vector machines [32].

The convex least-squares cardinality problem

$$\begin{array}{ll} \min_x & \|Ax-b\|_2 \\ \text{s.t.} & \text{card}(x) \leq M, \end{array} \tag{6.13}$$

is often approximated by the following $\ell_1$ heuristic, known as the LASSO algorithm [176] with shrinkage parameter $\beta$

$$\begin{array}{ll} \min_x & \|Ax-b\|_2 \\ \text{s.t.} & \|x\|_1 \leq \beta. \end{array} \tag{6.14}$$

Another popular $\ell_1$ heuristic for (6.13) is called basis pursuit denoising and is formulated as

$$\min_x \quad \|Ax-b\|_2 + \gamma\|x\|_1 \tag{6.15}$$

### 6.2.2. Joint estimation with piecewise changing parameters

Now consider the problem of joint state and input estimation. If we know the input is piecewise constant, then the proper way to formulate it is as follows

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{w},\mathbf{v},\mathbf{u}} & \|S_0^{-T}(x_0-\hat{x}_0)\|_2 + \sum_{k=0}^{T-1}\|W_k^{-T}w_k\|_2 + \sum_{k=0}^{T}\|V_k^{-T}v_k\|_2 \\ \text{s.t.} & x_{k+1} = A\,x_k + B\,u_k + G\,w_k, \quad k=0,\ldots,N-1, \\ & y_k = h_k + C_k\,x_k + v_k, \quad k=0,\ldots,N, \\ & \text{card}(D\mathbf{u}) \leq M, \end{array}$$

where $D$ is chosen equal to the first derivative operator matrix

$$D_1 = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix}$$

and where $M$ represents the maximum number of jumps.

Note that when the input is piecewise linear, then the same strategy applies with $D$ equal to the second derivative matrix

$$D_2 = \begin{bmatrix} -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \end{bmatrix}$$

and $M$ represents the maximum number of breakpoints.

The non-convex problem (6.18) can be written as a sequence of convex problems where the cardinality constraint is moved to the objective [38]

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{w},\mathbf{v},\mathbf{u},\mathbf{t}} & \|S_0^{-T}(x_0-\hat{x}_0)\|_2+\sum_{k=0}^{T-1}\|W_k^{-T}w_k\|_2+\sum_{k=0}^{T}\|V_k^{-T}v_k\|_2+\mathbf{t}^{\mathrm{T}}(\mathbf{z}+\varepsilon\mathbf{1}) \\ \text{s.t.} & x_{k+1}=A\,x_k+B\,u_k+G\,w_k, \quad k=0,\ldots,N-1, \\ & y_k=h_k+C_k\,x_k+v_k, \quad k=0,\ldots,N, \\ & -\mathbf{t}\leq D\mathbf{u}\leq\mathbf{t}, \end{array} \tag{6.16}$$

$$\begin{array}{ll} \min_{\mathbf{z}} & \mathbf{t}^{\mathrm{T}}(\mathbf{z}+\varepsilon\mathbf{1}) \\ \text{s.t.} & 0\leq\mathbf{z}\leq\mathbf{1}, \\ & \mathbf{z}^{\mathrm{T}}\mathbf{1}=n_u-M \end{array} \tag{6.17}$$

where $n_u$ denotes (with some abuse of notation) the number of elements in $D\mathbf{u}$ and $\mathbf{t},\mathbf{z}\in\mathbf{R}^{n_u}$. $\mathbf{1}$ denotes a vector of appropriate length with all ones. This sequence is iterated until $|D\mathbf{u}^*|^{\mathrm{T}}\mathbf{z}^*$ vanishes or until a maximum number of iterations is reached. The vector $\mathbf{z}$ can be interpreted as a search direction. $\varepsilon$ is a relatively small positive constant. The motivation for adding a small positive term $\varepsilon$ is to allow determination of the absolute value $|D\mathbf{u}^*|$ for the zero components of $\mathbf{t}^*$.

By initializing $\mathbf{z}$ to $(1-\varepsilon)\mathbf{1}$, the first iteration is a problem with $\ell_1$ norm penalization of parameter changes

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{w},\mathbf{v},\mathbf{u}} & \|S_0^{-T}(x_0-\hat{x}_0)\|_2+\sum_{k=0}^{T-1}\|W_k^{-T}w_k\|_2+\sum_{k=0}^{T}\|V_k^{-T}v_k\|_2+\|D\mathbf{u}\|_1 \\ \text{s.t.} & x_{k+1}=A\,x_k+B\,u_k+G\,w_k, \quad k=0,\ldots,N-1, \\ & y_k=h_k+C_k\,x_k+v_k, \quad k=0,\ldots,N, \end{array}$$

Subsequent iterations can be interpreted as corrections to this $\ell_1$ norm regularized problem. The sequence converges to a locally optimal solution of the original cardinality problem (6.18) [38].

A similar $\ell_1$ heuristic was proposed for trend filtering in [158] and for portfolio optimization in [126]. The iterative weighted $\ell_1$ heuristic proposed there is a simplification of the one presented here. It avoids the optimization step (6.17) and instead keeps $\mathbf{z}$ fixed. We have experienced substantial convergence improvements by the optimization step and hence suggest the above strategy.

### 6.2.3. Riccati based solution

It is easily seen that problem (6.16) is a modified MHE problem which only adds inequality constraints and gradient terms, but no quadratic penalties on the auxiliary variables. Therefore, the matrix Riccati recursions remain the same.

Problem (6.17) on the other hand is a Linear Programming problem and can be solved using the simplex method or an interior point method [29, 136].

### 6.2.4. Polishing

A standard approach is to solve the above convex cardinality problem to identify the break points and then solve a standard joint estimation problem with unknown piecewise constant input with the given jump locations. This technique is described in *e.g.* [29, §10.3.2] and [158].

### 6.2.5. Numerical example

The algorithms are illustrated on a linearized model of the pitch dynamics for a highly manouvrable aircraft. The data correspond to representative trimmed[3] flight conditions during pull-up. The model and data are thoroughly described in [168]. The continuous-time representation of the model is

$$\dot{x}(t) = \begin{pmatrix} -.0193 & 8.82 & -32.2 & -.48 \\ -.000254 & -1.02 & 0 & .91 \\ 0 & 0 & 0 & 1 \\ 0 & .82 & 0 & -1.08 \end{pmatrix} x(t) + \begin{pmatrix} .17 \\ -.00215 \\ 0 \\ -.18 \end{pmatrix} u(t)$$

$$y(t) = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 \end{pmatrix} x(t),$$

where $x = \begin{pmatrix} V & \alpha & \theta & q \end{pmatrix}^{\mathrm{T}}$ with $V$ the velocity (ft/s), $\alpha$ the angle of attack (rad), $\theta$ the pitch angle (rad) and $q$ the pitch rate (rad/s). The control input $u$ is the elevator deflection (deg). The outputs are the pitch angle $\theta$ and the flight path angle $\theta - \alpha$. The linear model is discretized in time using a first-order-hold method with time step $0.1s$ resulting in an LTI discrete-time state-space model. The system is disturbed with random process and measurement noise with covariances respectively $Q = 10^{-4}$ and $R = 10^{-2}$.

The input estimates are shown in Figure 6.9 for the standard $\ell_2$ norm MHE, $\ell_1$ norm MHE and the proposed method using a cardinality formulation with a polishing step. All joint input and state estimators are able to detect the input jumps, however, the proposed method is much less sensitive and more adequate in estimating the determinstic input variations.

[3] The aircraft is in in a stable condition; no unbalanced forces or moments are acting to cause it to deviate from steady flight.

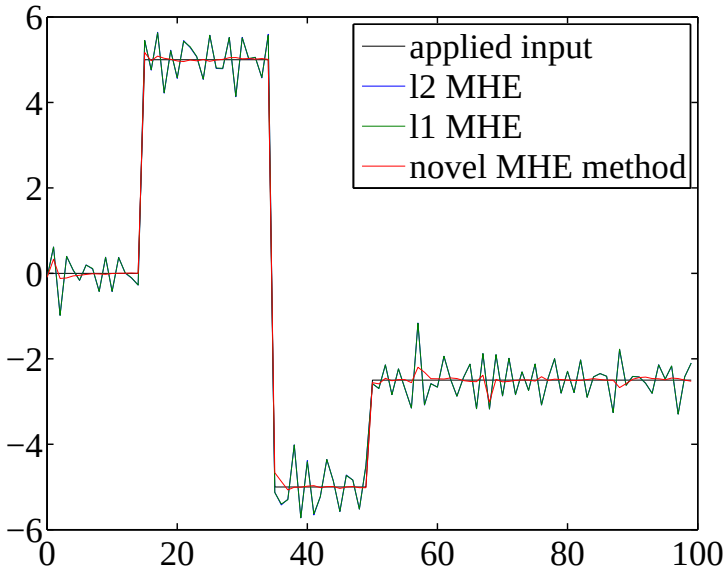



**Figure 6.9.** Input estimates using $\ell_2$ norm MHE, $\ell_1$ norm MHE and the proposed cardinality MHE formulation with a polishing step.

## 6.3. Conclusions

The goal of this chapter was to show that several robust MHE formulations may be described by a convex problem. Two types of robust convex problems were investigated. The problem of state estimation in the presence of outlier contaminated data was considered first. A Huber penalty MHE formulation and an $\ell_1$ formulation were studied and compared with the standard least-squares formulation. It was shown that Huber penalty MHE provides a performance improvement to the standard MHE in case of outlier contaminated data while preserving the performance of the standard formulation when the frequency of outlier occurance is rather low. A second convex formulation concerned the joint input and state estimation for systems with piecewise-constant or piecewise-linear inputs. It was shown that this problem is best formulated as a cardinality problem which can be solved by a sequence of convex MHE problems. More specifically, it concerns an $\ell_1$ norm MHE problem and subsequent modifications which can be solved using dedicated Riccati based methods. The performance of the methods is shown through numerical simulations.

CHAPTER 7

# Nonlinear MHE algorithms and application to estimation and control of blood-glucose in the Intensive Care

## 7.1. Introduction

Two important characteristics distinguish MHE from other estimation strategies, such as the Extended Kalman Filter (EKF). First of all, prior information in the form of constraints on the states, disturbances and parameters can be included. Second, the nonlinear nonlinear model equation is directly imposed over the horizon length. The most important advantage of using a larger window size is that this mitigates problems due to poor initialization or poor arrival cost approximation which results in improved estimation performance especially when the problems are highly nonlinear.

Stability of nonlinear moving horizon estimation (NMHE) was studied by Rao et al [149] and Alamir et al [1]. Raff et al [171] proposed the use of observability maps in the context of NMHE to yield a scheme with guaranteed convergence. Stability of NMHE with no assumptions on the noise distributions was addressed by Alessandri et al [5] Zavala et al present stability results for an advanced-step NMHE approach [201, 202]. An overview of optimal control methods for estimation and control on moving horizons is given by Binder et al [19]. An adaptive discretization scheme for NMHE in a single shooting framework was investigated by Binder et al [20]. Multiple shooting for NMHE has been proposed recently [45, 117, 118]. An overview of numerical aspects of NMHE and NMPC was given by Diehl et al in [44].

In this chapter we outline an algorithm for nonlinear MHE and MPC using a direct multiple shooting Gauss-Newton method which we then apply to a biomedical problem: the normalization of blood-glucose of patients in the intensive care unit. The application of multiple shooting to MHE including the implementation of a real time iteration scheme has been investigated and presented by Kraus et al [117, 118] and Kühl [119]. The aim in this chapter is to introduce the framework and show that Gauss-Newton SQP iterations yield quadratic subproblems which can be solved efficiently using the methods presented in the previous chapters.

## 7.2. Brief overview of recursive nonlinear estimation methods

### 7.2.1. The Extended Kalman Filter (EKF)

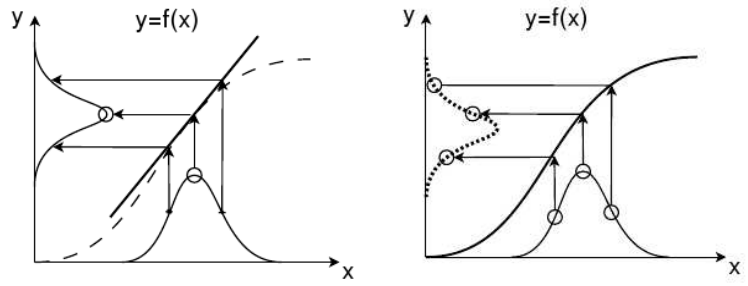
Optimal filtering for nonlinear systems is in general infeasible for practical application. Therefore approximate nonlinear filtering algorithms have been proposed in literature. The best known approximate nonlinear filter is the *Extended Kalman Filter (EKF)* (see *e.g.* [6]). The EKF linearizes around the trajectory of estimates and applies the time-varying Kalman filter recursions in any of the formulations described above. At time instant $k$ the nonlinear discrete-time model (1.5) is linearized as follows. For the measurement model

$$C_k = \left.\frac{\partial h_k(x,u)}{\partial x}\right|_{x=\hat{x}_k, u=u_k}. \tag{7.1}$$

And for the state model

$$A_k = \left.\frac{\partial f_k(x,u,p,w)}{\partial x}\right|_{x=\hat{x}_{k+}, u=u_k, p=p_k, w=0}, \quad G_k = \left.\frac{\partial f_k(x,u,p,w)}{\partial w}\right|_{x=\hat{x}_{k+}, u=u_k, p=p_k, w=0}. \tag{7.2}$$

Although the implementation of EKF is straightforward and intuitive, the resulting error from linearization may cause filter divergence. As shown by Song and Grizzle [165] the EKF provides only weak local stability guarantees. Problems of divergence and poor performance of the EKF (see *e.g.* [89, 110]) are related not only to the extent of nonlinearity of the model but also to the amount of noise entering the system [197]. If the noise is small and the nonlinearities are mild, the performance of EKF can be expected to be nearly optimal [146].




(a) (b)

**Figure 7.1.** Comparison of Extended and Unscented Kalman filter information propagation. (a) The EKF linearizes the nonlinear function around the mean of a Gaussian distribution and propagates the mean and covariance matrix through this linearized model. (b) The Unscented Kalman filter propagates a set of sigma-points through the nonlinear function and constructs a Gaussian distribution by calculating the mean and covariance of the set of propagated sigma-points.

### 7.2.2. The Unscented Kalman Filter (UKF)

An improvement to the Extended Kalman Filter led to the development of the Unscented Kalman filter (UKF), also a nonlinear filter. In the UKF, the probability density is approximated by the nonlinear transformation of a random variable, which returns much more accurate results than the first-order Taylor expansion of the nonlinear functions in the EKF. This is depicted in Figure 7.1.

When the system model is highly nonlinear, the Extended Kalman Filter can give particularly poor performance [89, 110]. This is because the mean and covariance are propagated through linearization of the underlying non-linear model. The Unscented Kalman Filter (UKF) [110] uses a deterministic sampling technique known as the unscented transform (UT) to select a minimal set of sample points, called sigma points, around the mean. This is in contrast to techniques such as particle filtering which sample randomly and it allows to use a relatively small number of samples. These sigma points are propagated through the nonlinear functions, from which the mean and covariance of the estimate are then recovered. The result is a filter which captures the mean and covariance accurately to the second order for any nonlinear state equation. For a state dimension $n_x$ a set of $2n_x+1$ points are necessary. This number was further reduced to $n_x+1$ by Julier in 2003 [109]. In addition, this technique removes the requirement to explicitly calculate Jacobians and therefore it is sometimes referred to as *derivative-free filter*.

## 7.3. Introduction to nonlinear MHE using multiple shoothing and SQP

Methods for solving optimal control problems can be classified in three major groups

I Dynamic programming,

II Indirect methods,

III Direct methods.

Dynamic programming, introduced by Bellman in the 1940s [12], describes a process of breaking the dynamic optimization problem into simpler subproblems based on the *principle of optimality*. For every subproblem a value function is defined which can be found by recursively working backwards from the final time step. For linear quadratic problems there is an analytic solution, viz the LQR Riccati equation. For nonlinear problems, however, this approach involves a discretization and tabulation of states and controls and therefore suffers from *the curse of dimensionality*. For this reason dynamic programming is typically restricted to control problems with very small state dimension and has not found acceptance for state and parameter estimation problems.

Indirect methods are based on *Pontryagin's Maximum Principle* [143] and proceed by maximizing the Hamiltonian matrix. An intricate multi-point boundary value problem is formulated. In every iteration, the model is numerically integrated forward in time and the adjoint equations are integrated backward in time. Indirect methods suffer a number of practical drawbacks. First, significant knowledge and experience in optimal control is required from the user of an indirect method since the adjoint equations and boundary conditions need to be derived for every specific application. Furthermore, (de)activiation of a state constraint leads to a discontinuity in the adjoint equation. Finally, stable processes have associated unstable adjoint equations and vice versa, hence, indirect methods always involve solving an unstable differential equation.

The basic idea of direct methods is to transcribe the infinite-dimensional dynamic optimization problem into a finite-dimensional NLP by discretizing the nonlinear functions. Direct methods have proven to be the most successfull for real-time, moderate to large scale, optimal control problems [16, 19, 43, 45, 46, 47]. Only direct methods are considered in this thesis.

For an overview of solution methods for optimal control problems, we refer to [19] and the references therein.

### 7.3.1. Discretization

In order to obtain a tractable finite-dimensional optimization problem, the continuous nonlinear model needs to be discretized. We distinguish between the following discretization strategies

I Single shooting,

II Multiple shooting,

III Collocation.

In single shooting, the system equations are used to eliminate the states from the optimization problem, regarding them as a function of the *controls*, or in the context of MHE the state disturbances. As such, the system equations and the optimization problem are treated sequentially, one after the other, in each optimization iteration. This approach, although very intuitive and popular throughout the literature, frequently fails even when good initial estimates are available [124]. The problem is that the error introduced by discretization, roundoff or poor initial data may be propagated by inherent instabilities in the differential equations and grow exponentially thus preventing numerical integration to the end point [124].

These convergence problems can be considerably improved by using the multiple shooting method [23, 25]. The basic idea of the direct multiple shooting approach is to do a time-discretization of all state and control trajectories in the overall optimization interval and to solve the resulting subproblems simultaneously in each of the discretization intervals, which are referred to as multiple-shooting intervals. This means that the original problem is divided into multiple subproblems that can be solved in parallel. Additionally, continuity constraints or *matching conditions* have to be imposed between neighboring intervals to guarantee a continuous solution over the whole optimization interval. By dividing the initial value problems into subintervals and allowing discontinuous trajectories at intermediate iterations, the growth of error in case of inherent instabilities is effectively limited. Compared to single shooting, the method shows greatly improved convergence and numerical stability [124]. We refer to the original papers by Bock and Plitt for a discussion on direct multiple shooting for optimal control [23, 25]. The multiple shooting method can be viewed as a hybrid sequential-simultaneous method.

The collocation method is a simultaneous discretization approach in which the differential equations are discretized in both state and control space. The simultaneous discretization and optimization potentially leads to faster convergence compared to shooting methods. However, due to this full discretization, the resulting NLPs tend to become large but are typically sparse [19]. Efficient collocation methods therefore

exploit the structure inherent in the NLPs. Compared to shooting methods, collocation cannot make use of existing highly reliable integration routines. Both collocation and multiple shooting can make use of initial guesses over the whole state trajectory, which is advantageous especially in online applications, *i.e.* MHE and MPC. Both are well-applicable to highly unstable systems in contrast to single shooting.

In this thesis we use the multiple shooting method for nonlinear MHE and MPC.

### 7.3.2. Constrained Gauss-Newton Sequential Quadratic Programming (CGN-SQP)

Consider the nonlinear MHE problem

$$\begin{aligned} \min_{\mathbf{x},\mathbf{w}} \quad & \|S_0^{-T}(x_0-\hat{x}_0)\|^2+\sum_{k=0}^{N-1}\|W_k^{-T}w_k\|^2+\|V_k^{-T}(h_k(x_k,w_k)-y_k)\|^2 \\ & +\|V_N^{-T}(h_N(x_N)-y_N)\|^2 \\ \text{s.t.} \quad & x_{k+1}=f_k(x_k,w_k,u_k,p), \quad k=0,\dots,N-1, \\ & g_k(x_k,w_k)\leq 0, \qquad k=0,\dots,N-1, \\ & g_N(x_N)\leq 0, \end{aligned} \tag{7.3}$$

Let us denote the overall optimization vector as

$$z=(x_0,w_0,\dots,x_{N-1},w_{N-1},x_N)\in\mathbf{R}^{n_z},$$

where $n_z=(N+1)n_x+Nn_w$.

Then problem (7.3) can be written as

$$\begin{aligned} \min_{\mathbf{z}} \quad & \tfrac{1}{2}\|F_1(z)\|_2^2 \\ \text{s.t.} \quad & F_2(z)=0, \\ & F_3(z)\leq 0, \end{aligned} \tag{7.4}$$

where

$$F_1(z)=\begin{bmatrix} S_0^{-\mathrm{T}}(x_0-\hat{x}_0) \\ W_0^{-\mathrm{T}}w_0 \\ V_0^{-\mathrm{T}}(h_0(x_0,w_0)-y_0) \\ \vdots \\ W_{N-1}^{-\mathrm{T}}w_{N-1} \\ V_N^{-\mathrm{T}}(h_N(x_N)-y_N) \end{bmatrix},$$

$$F_2(z) = \begin{bmatrix} f_0(x_0, w_0) - x_1 \\ f_1(x_1, w_1) - x_2 \\ \vdots \\ f_{N-1}(x_{N-1}, w_{N-1}) - x_N \end{bmatrix},$$

$$F_3(z) = \begin{bmatrix} g_0(x_0, w_0) \\ g_1(x_1, w_1) \\ \vdots \\ g_{N-1}(x_{N-1}, w_{N-1}) \\ g_N(x_N) \end{bmatrix}.$$

The Lagrangian is given by

$$\mathcal{L}(z, \lambda, \mu) \quad = \quad \frac{1}{2} F_1(z)^{\mathrm{T}} F_1(z) + F_2(z)^{\mathrm{T}} \lambda + F_3(z)^{\mathrm{T}} \mu. \tag{7.5}$$

where $\lambda$ and $\mu$ are multiplier vectors associated with respectively the equality and inequality constraints.

Assume the nonlinear functions $f$, $g$ and $h$ are twice differentiable. Their first order Taylor expansions are given by

$$\begin{array}{lll} f_k(x_k^l, w_k^l) & \approx f_k + A_k \Delta x_k + G_k \Delta w_k, & k = 0, \ldots, N-1, \\ g_k(x_k^l, w_k^l) & \approx g_k + D_k \Delta x_k + E_k \Delta w_k, & k = 0, \ldots, N-1, \\ h_k(x_k^l, w_k^l) & \approx h_k + C_k \Delta x_k + H_k \Delta w_k, & k = 0, \ldots, N-1, \\ g_N(x_k^l) & \approx g_N + D_N \Delta x_N. & \\ h_N(x_k^l) & \approx h_N + C_N \Delta x_N. & \end{array} \tag{7.6}$$

Any locally optimal solution to (7.4) defined by the triplet $(z^*, \lambda^*, \mu^*)$ has to satisfy under mild conditions the following KKT conditions

$$J_1(z^*)^{\mathrm{T}} F_1(z^*) + \lambda^{*\mathrm{T}} J_2(z^*) + \mu^{*\mathrm{T}} J_3(z^*) \quad = \quad 0 \tag{7.7}$$

$$F_2(z^*) \quad = \quad 0 \tag{7.8}$$

$$F_3(z^*) \quad \leq \quad 0 \tag{7.9}$$

$$\mu^* \quad \geq \quad 0 \tag{7.10}$$

$$\mu_j^* F_{3,j}(x^*) \quad = \quad 0, \quad j = 1, \ldots, n_p \tag{7.11}$$

Here, the Jacobians $J_i$ and function evaluations $F_i$ are highly structured and sparse

$$J_1 = \begin{bmatrix} S_0^{-\mathrm{T}} & & & & & & & \\ V_0^{-\mathrm{T}}C_0 & V_0^{-\mathrm{T}}H_0 & & & & & & \\ & W_0^{-\mathrm{T}} & & & & & & \\ & & \ddots & & & & & \\ & & & V_{N-1}^{-\mathrm{T}}C_{N-1} & V_{N-1}^{-\mathrm{T}}H_{N-1} & & & \\ & & & & W_{N-1}^{-\mathrm{T}} & & & \\ & & & & & & V_N^{-\mathrm{T}}C_N \end{bmatrix}, \quad F_1 = \begin{bmatrix} S_0^{-T}(x_0^l - \hat{x}_0) \\ V_0^{-\mathrm{T}}(h_0 - y_0) \\ W_0^{-\mathrm{T}}w_0^l \\ \vdots \\ V_{N-1}^{-\mathrm{T}}(h_{N-1} - y_{N-1}) \\ W_{N-1}^{-\mathrm{T}}w_{N-1}^l \\ V_N^{-\mathrm{T}}(h_N - y_N) \end{bmatrix},$$

$$J_2 = \begin{bmatrix} A_0 & G_0 & -I & & & & & & \\ & & A_1 & G_1 & -I & & & & \\ & & & & & \ddots & & & \\ & & & & & & A_{N-1} & G_{N-1} & -I \end{bmatrix}, \quad F_2 = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{bmatrix},$$

$$J_3 = \begin{bmatrix} D_0 & E_0 & & & & & & & \\ & & D_1 & E_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & D_{N-1} & E_{N-1} & & \\ & & & & & & & D_N \end{bmatrix}, \quad F_3 = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{N-1} \\ g_N \end{bmatrix}.$$

Newton-type methods search for a point satisfying the KKT conditions by successively linearizing the problem functions. The main difference between the two big families of nonlinear optimization methods, Sequential Quadratic Programming and Nonlinear Interior-Point methods, is in how the complementarity condition (7.11) is handled (see Chapter 2). We will only consider Sequential Quadratic Programming here.

By linearizing in every iteration all nonlinear functions occurring in the KKT conditions around the current iterate, one arives at a linear complementarity problem which can be interpreted as the KKT conditions of a QP (see also (2.65))

$$\begin{array}{ll} \min_z & J_1(z^l)^{\mathrm{T}}z + \frac{1}{2}(z-z^l)^{\mathrm{T}}\nabla_z^2\mathcal{L}(z^l,\lambda^l,\mu^l)(z-z^l) \\ \text{s.t.} & F_2(z^l) + J_2(z^l)^{\mathrm{T}}(z-z^l) = 0, \\ & F_3(z^l) + J_3(z^l)^{\mathrm{T}}(z-z^l) \leq 0, \end{array} \tag{7.12}$$

where $z^l$ is the current iterate around which the functions are linearized.

The Lagrange Hessian is given by

$$\nabla_z^2\mathcal{L} = \frac{1}{2}J_1^{\mathrm{T}}J_1 + \sum_j F_{1,j}\nabla^2 F_{1,j} + \sum_j \lambda_j \nabla^2 F_{2,j} + \sum_j \mu_j \nabla^2 F_{3,j} \tag{7.13}$$

The first term often dominates the summations either because of near-linearity of the model close to the solution, *i.e.* small second order derivatives, or because of small residuals, *i.e.* $F_{1,j}$ small. One can show that $\lambda$ becomes small if $F_1$ is small and the last

term vanishes at the solution by the complementarity condition (7.11). This motivates the following Gauss-Newton approximation to the Hessian for least-squares problems

$$\nabla_z^2 \mathcal{L} \approx \frac{1}{2} J_1^{\mathrm{T}} J_1 \tag{7.14}$$

Hence, in a generalized Gauss-Newton method the original problem (7.4) is solved iteratively by computing in every iteration $l$ a search direction $\Delta z^l$ as the solution to the following QP

$$\begin{array}{ll} \min_{\Delta z} & \frac{1}{2}\|J_1(z^l)\Delta z + F_1(z^l)\|_2^2 \\ \text{s.t.} & J_2(z^l)\Delta z + F_2(z^l) = 0, \\ & J_3(z^l)\Delta z + F_3(z^l) \leq 0. \end{array} \tag{7.15}$$

With the functions and Jacobians as defined above this corresponds to the constrained linear MHE problem (4.6)-(4.7) or (5.1)-(5.15) considered in the previous chapters.

The constrained Gauss-Newton method has only linear convergence but often with a surprisingly fast contraction rate [41, 44]. The contraction rate is fast when the residual norm is small or the second derivatives are small [44, 136]. It was developed and extensively investigated by Bock and coworkers, see *e.g.* [23, 24, 42, 124]. For parameter estimation problems, the method is typically able to outperform methods using higher order derivatives or derivative-free methods. This is because the method is not attracted by large residual local minima, which are so-called *statistically unstable estimates* [24].

### Globalization

The quadratic problems (7.15) which are solved in every Gauss-Newton iteration are only local approximations of the true nonlinear problem. To enforce convergence from arbitrary starting points two types of globalization strategies are commonly used for the search directions obtained from the QP subproblems: line-search and trust-region methods.

In a trust-region approach, the constraint $\|\Delta z\| \leq d$ is added to the QP. The trust-region parameter $d$ is adjusted based on the agreement of predicted reduction (from the QP) and actual reduction of a merit function. If sufficient reduction is obtained for the next iterate $z^{l+1} = z^l + \Delta z^l$ then the step is taken, otherwise the trust region is shrinked, *i.e.* $d$ is decreased. On the other hand, if the agreement of predicted and actual reduction is good, the trust-region may be enlarged in order to speed up convergence.

In a line-search strategy on the other hand, QP (7.15) is solved in every iteration without additional constraints, and a suitable step size or damping factor $\alpha^l \in (0,1]$

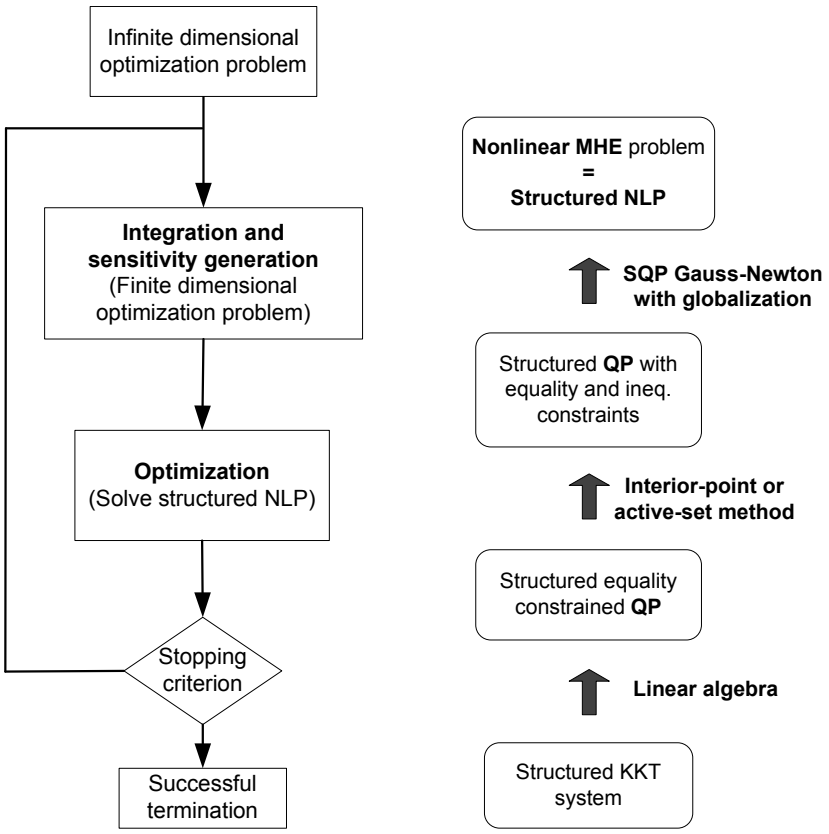



**Figure 7.2.** Solution strategy for continuous-time nonlinear MHE problems: (a) flowchart for a multiple shooting strategy, (b) hierarchy for an Sequential Quadratic Programming strategy with active-set or interior-point method and structure-exploiting KKT solver.

is chosen such that for the next iterate $z^{l+1} = z^l + \alpha^l \Delta z^l$ sufficient decrease of a merit function is obtained. Hence, after computing the search direction the step size can be obtained by solving a one-dimensional optimization problem (in $\alpha$). However, in almost all implementations the step size is computed approximately by backtracking as it does not make sense to aim for an exact solution to a problem which is only an approximation of the real (nonlinear) problem. The choice of merit function is crucial for fast convergence. A merit function which is affine invariant and has been found to yield excellent convergence performance in difficult applications [24, 41] is the

natural level function

$$T_{\mathrm{A}}(z) \quad = \quad \|J(z^l)^{-1}F(z)\|_2^2 \tag{7.16}$$

with

$$F(z) = \begin{bmatrix} F_1(z) \\ F_2(z) \\ F_3(z) \end{bmatrix}, \quad J(z) = \begin{bmatrix} J_1(z) \\ J_2(z) \\ J_3(z) \end{bmatrix}. \tag{7.17}$$

The natural level function will be used for globalizing convergence in the numerical examples presented below.

#### Calculating derivatives

The succesful convergence of the multiple shooting method is depending on the availability of accurate function evaluations and derivatives. With the development and increasing application of automatic differentiation tools, accurate first and second order derivatives can be generated [18]. We use the publicly available set of integration routines of ACADO [99]. The ACADO integrator package consists of several Runge-Kutta and a BDF integrator which allow the simulation and sensitivity generation for ODE and DAE systems based on internal numerical or automatic differentiation.

### 7.3.3. Arrival cost updates

As discussed in Chapter 1, a popular strategy for computing an approximate arrival cost is to use a first-order Taylor expansion around the trajectory of past estimates. This is equivalent to applying an EKF recursion for the covariance update. The smoothed update linearizes around the current best estimate but applies a correction to the state prediction in order to prevent accounting for some measurements twice.

The covariance update is given by the standard EKF update formula (2.75) using the following first order derivative matrices around the current best estimate

$$C_{k-N+1|k}, \quad A_{k-N+1|k}, \quad G_{k-N+1|k} \tag{7.18}$$

The state update can be derived by linearizing around the current best estimate $x_{k-N+1|k}$ and relating it to the filter estimate $x_{k-N+1|k-N}$. This was done in [117] and is pre-

sented here for compactness of presentation.

$$\begin{aligned} h(x) &\approx h(x_{k-N+1|k}) + C_{k-N+1|k}\left(x - x_{k-N+1|k}\right) && (7.19)\\ &= h(x_{k-N+1|k}) + C_{k-N+1|k}\left(x - x_{k-N+1|k}\right) \\ &\quad + C_{k-N+1|k}\left(x_{k-N+1|k-N} - x_{k-N+1|k-N}\right) && (7.20)\\ &= h(x_{k-N+1|k}) + C_{k-N+1|k}\left(x_{k-N+1|k-N} - x_{k-N+1|k}\right) \\ &\quad + C_{k-N+1|k}\left(x - x_{k-N+1|k-N}\right) && (7.21) \end{aligned}$$

Hence the constant term in the linearization is

$$h(x_{k-N+1|k}) + C_{k-N+1|k}\left(x_{k-N+1|k-N} - x_{k-N+1|k}\right) \tag{7.22}$$

and the state update is given by

$$\begin{aligned} \hat{x}_{k-N+1} &= A_{k-N+1|k}x_{k-N+1|k-N} + \bar{K}(-y_{k-N+1} \\ &\quad + h(x_{k-N+1|k}) + C_{k-N+1|k}(x_{k-N+1|k-N} - x_{k-N+1|k})) && (7.23) \end{aligned}$$

where $\bar{K}$ is the gain matrix, see Eq. (2.76), calculated around the current best estimate (7.18).

A major advantage of using Riccati-based methods at the core of an SQP strategy for NMHE is that the arrival cost updates are obtained as a natural outcome of the solution process. That is, the matrix Riccati recursion computes covariances around successive trajectories and the covariance update (7.23) is obtained as the forward second order matrix associated with the second within-horizon state. The state update is obtained by applying the above described correction to the second state calculated by the forward vector recursion, see Figure 3.1.

Although this arrival cost update is only locally valid and discards constraints. It often provides a reasonably well approximation and yields a performant MHE estimator. If this is not the case, and poor estimator performance or divergence is encountered, which may be due to plant-model mismatch or highly nonlinear dynamics, then an increase of the horizon length and/or omitting of the arrival cost term solves the problem in many practical cases. Other people have looked at better arrival cost approximations, *e.g.* using a particle filter [153].

## 7.4. Application of MHE based NMPC to normalize glycemia of critically ill patients

In this section the clinical ICU dataset that is used for the design and evaluation of the control system is described. Next, the considered patient model is described, followed

by a short description of the NMPC control strategy. Finally, the results of numerical simulations are discussed.

### 7.4.1. Tight glycemic control in the Intensive Care Unit

Hyperglycemia (i.e., an increased glucose concentration in the blood) and insulin resistance (i.e., the resistance of the glucose utilizing tissues to insulin) are common in critically ill patients even if they have not had diabetes before and are associated with adverse outcomes. Tight glycemic control (between 80 and 110 mg/dl = target range) by applying intensive insulin therapy in patients admitted to the medical and the surgical Intensive Care Unit (ICU) results in a spectacular reduction in mortality and morbidity [179, 181].

Currently, ICU patients are treated through a manual and rigorous administration of insulin [180]. In literature several physical models that describe the glucose dynamics and the insulin kinetics of healthy and diabetic subjects are used for glycemia control simulations in 'mathematical' diabetic (type I) patients (e.g., Hovorka et al. [101], Parker et al. [141, 142], among others). Analogously, we want to design a semi-automatated control system for glycemia control in the ICU. This system could reduce the workload for medical staff and could also introduce the glycemia normalization concept in hospitals that are currently *not* making use of the manual intensive insulin protocol [180], world-wide leading to a possible further reduction of mortality and morbidity [184].

### 7.4.2. ICU Dataset

A set of real patient data is used to evaluate the NMPC control strategy with state and disturbance estimation. The dataset is extensively described and discussed in [182]. It contains data of 19 adult critically ill patients who were admitted to the surgical ICU of the University Hospital K.U. Leuven (Belgium) for a variety of reasons. The data comprises recorded near-continuous subcutaneous glucose levels, more specifically three-minute-averaged values, as well as the administered flows of carbohydrate calories and insulin. In our setting, the Glucoday system (A. Menarini Diagnostics, Florence, Italy), a portable instrument provided with a micro-pump and a biosensor, coupled to a microdialysis system, was used to measure the glucose concentration (see [182]).

We want to point out [182, 185] that a near-continuous glucose sensor device is currently not standard practice in the ICU and was only used for this study. In current ICU practice, the used protocol [180] requires blood glucose levels to be measured

every four hours, or more frequently, for instance in the initial phase or after complications, and is done by blood gas sampling using the ABL machine (Radiometer, Copenhagen, Denmark). This method of measurement was also used for retrospective sensor calibration of the the near-continuous sensor (see [182]). Thus, for practical application of a control system in a semi-automated setting, the frequency of blood glucose measurement and adaption of insulin rate should be restricted. In this chapter we consider two realistic frequencies: once per hour and once per four hours.

In this chapter the observed near-continuous glucose test data are used for estimating the model and for comparing the MPC proposed control actions with the control behavior of a trained nurse.

### 7.4.3. ICU Minimal Model (ICU-MM)

The presented model structure originates from the known *minimal* model that is developed by Bergman et al ([13]). In ([183]) the original minimal model was extended to the ICU minimal model (ICU-MM) by taking into consideration some features typical of ICU patients. The new model was also validated on a real-life clinical ICU data set. The ICU-MM is presented as follows:

$$\frac{dG(t)}{dt} = (P_1 - X(t))G(t) - P_1 G_b + \frac{F_G}{V_G} + F_M, \tag{7.24a}$$

$$\frac{dX(t)}{dt} = P_2 X(t) + P_3(I_1(t) - I_b), \tag{7.24b}$$

$$\frac{dI_1(t)}{dt} = \alpha \ \max(0, I_2) - n(I_1(t) - I_b) + \frac{F_I}{V_I}, \tag{7.24c}$$

$$\frac{dI_2(t)}{dt} = \beta \ \gamma\,(G(t) - h) - nI_2(t), \tag{7.24d}$$

where $G$ and $I_1$ are the glucose and the insulin concentration in the blood plasma. The second insulin variable, $I_2$, is a purely mathematical manipulation such that $I_2$ does not have any direct clinical interpretation. The variable $X$ describes the effect of insulin on net glucose disappearance and is proportional to insulin in the remote compartment. $G_b$ and $I_b$ are the basal value of plasma glucose and plasma insulin, respectively. The model consists of two input variables: the intravenously administered (exogenous) insulin flow ($F_I$) and the parenteral carbohydrate calories flow ($F_G$). The glucose distribution space and the insulin distribution volume are denoted as $V_G$ and $V_I$, respectively. There is an unknown disturbance input that we ascribe to administered medication ($F_M$) and which directly influences the glucose concentration. This to the MPC unknown input could also account for other unknown disturbance factors.

The coefficient $P_1$ represents the glucose effectiveness (i.e., the fractional clearance of glucose) when insulin remains at the basal level; $P_2$ and $P_3$ are the fractional rates of net remote insulin disappearance and insulin dependent increase, respectively. The endogenous insulin is represented as the insulin flow that is released in proportion (by $\gamma$) to the degree by which glycemia exceeds a glucose threshold level $h$. The time constant for insulin disappearance is denoted as $n$. In case glycemia does not surpass the glucose threshold level $h$, the first part of 7.24c (that represents the endogenous insulin production) equals 0. In order to keep the correct units, an additional model coefficient, $\beta = 1$ min, was added. Finally, the coefficient $\alpha$ amplifies the mathematical second insulin variable $I_2$.

The units of all used variables and parameters and their initial coefficient values are represented in Table 7.1.

**Table 7.1.** States, inputs, patient features (constants) and parameters of the ICU minimal model.

| States | Units | |
|---|---|---|
| $G$ | mg/dl | |
| $X$ | 1/min | |
| $I_1$ | $\mu$U/ml | |
| $I_2$ | $\mu$U/ml | |
| **Inputs** | **Units** | |
| $F_I$ | $\mu$U/min | |
| $F_G$ | mg/min | |
| $F_M$ | mg/dl/min | |
| **Patient features** | **Units** | **Value** |
| $V_G$ | dl | Dep. on body mass |
| $V_I$ | ml | Dep. on |
| $G_b$ | mg/dl | Basal glycemia |
| $I_b$ | $\mu$U/ml | Basal insulin |
| **Parameters** | **Units** | **Value** (1) |
| $P_1$ | 1/min | -1.31 $10^{-2}$ (1) |
| $P_2$ | 1/min | -1.35 $10^{-2}$ (1) |
| $P_3$ | ml/(min$^2$ $\mu$U) | 2.90 $10^{-6}$ (1) |
| $h$ | mg/dl | 136 (1) |
| $n$ | 1/min | 0.13 (1) |
| $\alpha$ | 1/min | 3.11 |
| $\beta$ | min | 1 |
| $\gamma$ | $\frac{\frac{\mu U}{ml}\frac{dl}{mg}}{min^2}$ | 5.36 $10^{-3}$ (1) |

(1) As initial value for the model estimation process, the mean model coefficient values for the obese - low glucose tolerance patient group (described in [13]), are used.

### 7.4.4. Smoothing of discontinuities

The nonlinear model (7.24) contains a discontinuity in the form of a *max* term. In order to avoid problems of differentiability the *max* term was smoothed using exponential smoothing $\max(0, I_2) \sim s\ln(1+\exp(\frac{I_2}{s}))$ with a smoothing parameter $s = 0.1$.

### 7.4.5. Closed-loop nonlinear control system set-up

The complete closed-loop control system is depicted in Figure 7.3. Its components will be described in detail in this section. A nonlinear continuous-time system model is assumed (1.1). That is, the system is described by a set of nonlinear index-one ordinary differential equations of the form

$$\dot{x}(t) = f(x(t), u(t), w(t), d, p), \tag{7.25}$$

where $x$ are the differential states, $u$ the inputs, $w$ the system noise accounting for modeling errors, $d$ the unknown disturbances and $p$ the set of free parameters. We will also allow bounds on the variables

$$\begin{aligned} x_{\min} &\leq x(t) \leq x_{\max}, \\ u_{\min} &\leq u(t) \leq u_{\max}, \\ w_{\min} &\leq w(t) \leq w_{\max}, \\ d_{\min} &\leq d \leq d_{\max}, \\ p_{\min} &\leq p \leq p_{\max}. \end{aligned}$$

The measurement data are generated as

$$y_k = h(x(t_k)) + v_k, \tag{7.26}$$

where $v_k$ represents measurement noise (sensor noise) and the subscript $k$ indicates the fact that measurements are obtained at discrete time instants.

Thus, the disturbances that enter the closed loop system can be summarized as

1. process noise $w$, which is usually assumed to be zero-mean random noise but in the MHE-setting can also be regarded purely deterministic as bounded optimization variables with the only assumption that zero is contained in the feasible set,

2. unknown model disturbance $d$ which is assumed to be slowly varying. In the presented application the unknown disturbance represents the effect of medication, to which we assigned a typical realization and which we assumed to have direct influence on the glycemic level,

3. sensor noise $v$ which we will assume to be normally distributed with mean zero and known covariance matrix,

4. unknown initial states, disturbance and parameters. We will assume that expected values for the states, disturbance and parameters ($\bar{x}_0$, $\bar{d}_0$ and $\bar{p}_0$ resp.) are given as well as the corresponding covariance $P_0$. After a transient, the effect of the initial conditions usually diminishes and the estimates converge to the true values provided that the measurements contain sufficient information.

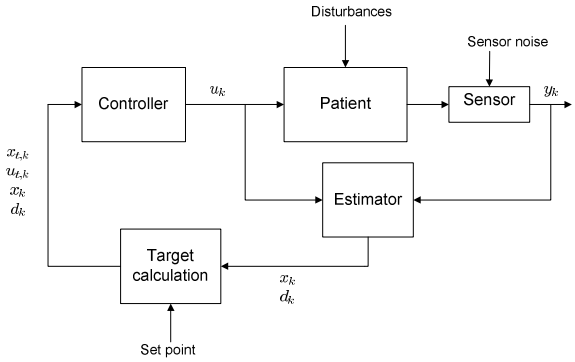



**Figure 7.3.** Illustration of the closed-loop control scheme.

## 7.4.6. Target calculation

The goal of target calculation is to find a steady state of the closed loop system and a corresponding input that yields the output at the set point. This is an inverse problem that can be formulated as an optimization problem. Due to constraints or nonlinearities it might occur that no steady-state targets can be found corresponding to the set point. In that case we require the output target to be the closest output to the set point for which a steady state exists. If there are multiple steady states satisfying the output set point, the one that is closest to the previous input target is selected. At each time instant a new target must be calculated to account for changing parameters and integrated disturbances.

We formulate the target calculation as the following optimization problem (see [172])

$$\min_{x^t(t_0),u^t(t_0),\eta} \quad \frac{1}{2}\eta^T \bar{Q}\eta + \bar{q}^T\eta + \frac{1}{2}(u^t(t_0) - u^t(t_{-1}))^T \bar{R}(u^t(t_0) - u^t(t_{-1})) \tag{7.27a}$$

subject to

$$x^t(t_0) = f(x^t(t_0), u^t(t_0), \hat{d}, \hat{p}), \tag{7.27b}$$

$$h(x^t(t_0)) - \eta \leq y_{\text{set}} \leq g(x^t(t_0)) + \eta, \tag{7.27c}$$

$$u_{\min} \leq u^t(t_0) \leq u_{\max}, \tag{7.27d}$$

$$x_{\min} \leq x^t(t_0) \leq x_{\max}, \tag{7.27e}$$

$$\eta \geq 0. \tag{7.27f}$$

Here $u^t(t_{-1})$ is the input target calculated in the previous time step. This is an exact penalty method ([61, 147]) which relaxes the problem in a $l_1/l_2^2$ sense if the set point is infeasible by introducing the *slack variable* $\eta$. In general, $\bar{q}$ is chosen to be relatively large and strictly positive and both $\bar{Q}$ and $\bar{R}$ are positive definite. By shifting the state and input targets, the target calculation accounts for modeling error and adjusts the model to remove offset from the closed-loop system.

### 7.4.7. Model predictive control

Given the current state, disturbance and parameter estimates $\hat{x}(t_0)$, $\hat{d}$, $\hat{p}$ of the system at time $t_0$, NMPC predicts the future dynamic behavior of the system over a horizon $T$ and determines the future inputs such that an open-loop performance objective function is optimized. Due to disturbances and/or model-plant mismatch the true system behavior is different from the predicted behavior. Therefore, in order to incorporate feedback, only the first of this optimal input sequence is applied to the system. When a new measurement and new estimates are obtained the horizon is shifted and the previous steps are repeated.

The use of Model based Predictive Control to normalize glycemia in the ICU allows to take into account the effect of current and future control moves, *i.e.* insulin dosing, on the future outputs,*i.e.* glycemia. For medical reasons the maximum insulin infusion rate (i.e., the control input) is 50 U/hr. In addition, the administered insulin flow is obviously constrained to be positive.

The open-loop optimization problem addressed in NMPC is

$$\min_{x(\cdot),u(\cdot)} \int_{t_0}^{t_0+T} \|\tilde{x}(t)\|_Q^2 + \|\tilde{u}(t)\|_R^2 \, dt \tag{7.28a}$$

$$\text{subject to} \quad x(t_0) = \hat{x}(t_0), \tag{7.28b}$$

$$\dot{x}(t) = f(x(t), u(t), \hat{d}, \hat{p}), \tag{7.28c}$$

$$c(x(t), u(t), \hat{d}, \hat{p}) \geq 0, \quad t \in [t_0, t_0 + T]. \tag{7.28d}$$

Here $\tilde{x}(t) = x(t) - x^t$ and $\tilde{u}(t) = u(t) - u^t$ with $x^t$ and $u^t$ the target state and input determined by the preceding target calculation. This approach of penalizing deviations from target states and inputs provides integral (offset free) control. In order to guarantee theoretical stability of the MPC controller, one should add to the above formulation either a terminal constraint, or a terminal cost, or both. We implemented a terminal constraint but it was found that in order to achieve guaranteed theoretical stability the control performance was deteriorated. Other stability measures are currently being investigated. For a detailed treatment of stability theory for NMPC, we refer to the excellent textbook by Rawlings and Mayne [152].

### Move blocking

A rule of thumb in control theory (and practice) states that the output should be sampled fast enough to capture all the important system dynamics. Often, however, the inputs are allowed to change only at a lower rate. In such cases integration time intervals for the state-space model are taken as short as necessary while the inputs are *blocked* during several time intervals. A strategy for *move blocking* of the inputs was added to the MPC formulation. For the glycemia control problem, integration time intervals of 5 min and a future horizon of $N_{\text{mpc}} = 240$ min are used, while the insulin flow input is allowed to change only every 60 min. This specification is imposed by the medical staff for clinical validation reasons.

## 7.4.8. ICU-MM parameters

The model has been described before and units of state variables, inputs and parameters are given in Table 7.2. The used parameter values are represented in Table 7.2. These values result from an estimation process applied to a real-life data set of 19 critically ill patients as described by Van Herpe et al [185]. The parameter and state values after the first 24 hour estimation for an arbitrary patient were chosen. Proceeding in this way the control system could be assessed using a realistic parameter realization.

**Table 7.2.** Units and values of the parameters applicable in the ICU minimal model. These values result from an estimation process applied to a real-life data set of 19 critically ill patients.

| Parameters | Units | Value |
|---|---|---|
| $V_G$ | dl | 116.8 |
| $V_I$ | ml | 8760 |
| $G_b$ | mg/dl | 95 |
| $I_b$ | $\mu$U/ml | 10.7 |
| $P_1$ | 1/min | -1.71 $10^{-2}$ |
| $P_2$ | 1/min | -2.24 $10^{-2}$ |
| $P_3$ | ml/(min$^2\mu$U) | 2.5 $10^{-3}$ |
| $h$ | mg/dl | 107.4 |
| $n$ | 1/min | 0.2623 |
| $\alpha$ | 1/min | 0.35 |
| $\beta$ | min | 1 |
| $\gamma$ | $\frac{\frac{\mu U}{ml}\frac{dl}{mg}}{min^2}$ | 1.4001 $10^{-4}$ |

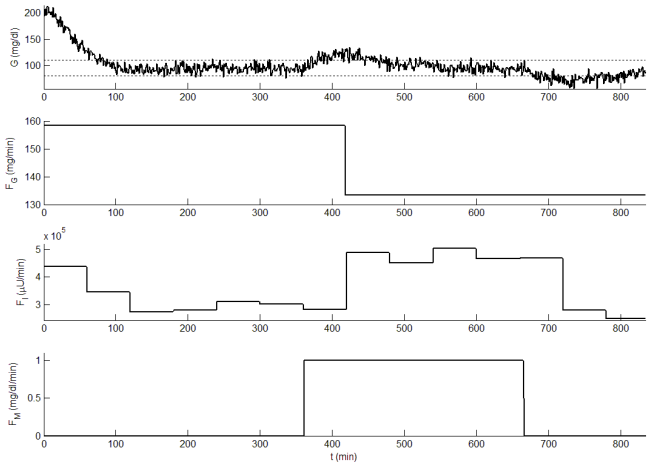



**Figure 7.4.** The top panel shows the evolution of the simulated glycemia $G$ with added sensor noise (solid line) and the target range of 80 – 110 mg/dl (dashed lines). The flow of the carbohydrate calories $F_G$ (second panel) is the known disturbance factor whereas the insulin rate $F_I$ (third panel) is the insulin sequence that is proposed by MPC controller. A fictitious medication disturbance factor $F_M$ (that is unknown to the MPC) is visualized in the bottom panel.

## 7.4.9. Results and discussion

### Closed-loop control performance

In Figure 7.4 the simulated glucose course with added measurement noise and the administered known and unknown input flows are illustrated. Starting from a high initial

blood glucose concentration, the closed-loop control system is able to regulate to the normoglycemic range ($80-110$ mg/dl) in a considerably short time span by administering a still clinically acceptable insulin flow. The MPC controller was precisely tuned to obtain both good control performance and clinical acceptability. Furthermore, the control system is able to suppress the unknown disturbance input. When the rather large disturbance (i.e., medication) enters, the glycemic level is raised into the *modest* hyperglycemic range, after which the insulin flow is adjusted and the glycemic level is steered to the normoglycemic range again. Further on, a *slight* hypoglycemic event occurs when the large disturbance suddenly drops. This result shows the potential of the proposed control system to normalize the blood glucose exploiting the nonlinear model dynamics and taking into consideration unknown disturbance factors that are omnipresent in the ICU.

### Moving horizon estimation

In Figure 7.5 the courses of the four states and their estimates are depicted as well as the unknown disturbance and its estimate. The measurements are corrupted with zero-mean random noise with standard deviation $\sigma = 7.5$ mg/dl. For the estimator a time horizon of $N_{\text{mhe}} = 5$ min was employed. The true initial state of the system was $x_0 = \begin{bmatrix} 207 & 58.0 & 0.0005 & 1.49 \end{bmatrix}$ (see Table 7.1) while the estimator was initialized with $\bar{x}_0 = \begin{bmatrix} 180 & 20 & 0 & 0 \end{bmatrix}$. Despite this rather large initial error a fast convergence to the true state values could be obtained leading to a minimal impact of initial error on the closed loop performance. Furthermore, also the unknown disturbance could be perceived with reasonable accuracy from the output measurements.

### Target calculation

Figure 7.6 shows the target input (insulin flow) and the corresponding optimal input computed by the MPC controller. The target input is influenced by the estimated unknown disturbance input and (less noticeable) by the changing carbohydrate calories flow. The MPC computed input is expected to track the big changes of the target but not the fast fluctuations, which is reasonably well achieved as can be seen from the figure. The effect of move blocking can be seen when a change in the influencing parameters occurs, for example at time instants $t = 360$ min and $t = 665$ min a sudden change in the unknown disturbance occurs, which is detected (estimated) shortly thereafter. Due to move blocking of the input the controller is not able to instantly react to these changes, leading for instance to a short hypoglycemic event around time $t = 750$ min (see Figure 7.4).

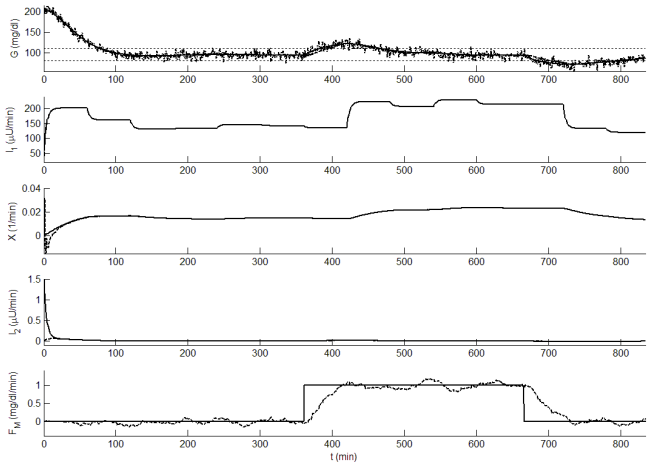



**Figure 7.5.** The four top panels show the evolution of the true states (solid lin) and its estimates (dashed line). The bottom panel shows the true (solid line) and estimated (dashed line) unknown disturbance input.

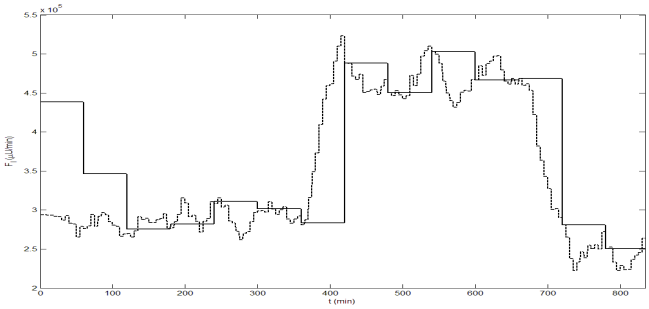



**Figure 7.6.** Evolution of the computed target input (dashed line) and the optimal input proposed by the MPC controller (solid line).

## 7.5. Conclusions

In this chapter nonlinear MHE algorithms were discussed. In particular, a multiple shooting Gauss-Newton SQP type algorithm was presented for nonlinear MHE, using at its core the efficient MHE algorithms presented in the previous chapters.

MHE based MPC was proposed as a control strategy for regulating the blood-glucose level in critically ill patients at the Intensive Care Unit. The moving horizon estimator accurately estimates the true states from noisy output measurements of the blood-glucose. MHE is able to recover quickly from a wrong initial guess of the state vector. A target calculation is proposed to remove the effect of disturbances and changing parameters. The ability of the closed-loop control system to regulate to the

normoglycemic range in a short time span and to suppress disturbances is shown for a realistic disturbance realization. The proposed control system shows potential for application to glycemia regulation in the ICU. Future research is directed towards investigating other patient models in a closed-loop control strategy, see [27, 186] for prediction performance comparisons of several patient models. In addition, we aim to further investigate the robustness of the methods under different scenarios as well as the impact of data quality. If these further steps are successful, the control strategy may be proposed for clinical trials at the ICU in the near future.

CHAPTER 8

# General conclusions and outlook

The framework of MHE allows incorporation of first-principles dynamic models and constraints and permits flexibility in the objective functions. This provides an unprecedented potential for real-time state and parameter estimation. However, it also presents many computational challenges. In this chapter, we summarize our contributions and present suggestions for future research.

## 8.1. Conclusions

The optimality conditions simply reveal the structure and symmetry inherent in MHE problems. They are due to the Markov property of the dynamic models and the symmetric objectives. These properties are also fundamental to the recursive solution provided by the celebrated Kalman filter. In this thesis, we exploit the structure and symmetry by using exactly these recursive methods for solving the systems of KKT optimality conditions. Since MHE considers a past window of data this corresponds to a smoothing problem. In Chapter 3 Riccati based methods were derived from decomposition of the KKT system. In particular a square-root version using numerically stable orthogonal operations was proposed as a robust and efficient method for solving MHE problems.

Constraints can be imposed in a natural way, leading to quadratic programs which can be solved using either interior-point or active-set methods.

In Chapter 4, interior-point methods were investigated. It is shown that the log-barrier terms enter on the block diagonal of the KKT system and thereby preserve the structure and symmetry. More specifically, the barrier terms alter the measurement update step and constraints can be interpreted as perfect measurements. The Riccati based

methods are modified to account for these terms. Several types of constraints were considered: mixed or separate, general or bound constraints. It was demonstrated that an intelligent initialization can significantly reduce the number of iterations required for convergence. In this context, a hot-starting strategy was proposed. It was shown that in a primal barrier interior-point method the barrier Hessian of an actively constrained variable grows logarithmically as the constraints are approached. This directly yields a logarithmic growth of the condition number of the matrices in the recursions. This strongly motivated the use of the square-root version of the Riccati based method within an interior-point MHE method. A C implementation demonstrated the performance of the square-root algorithm. It was shown by numerical examples that the method can be run in the milisecond range on a standard computer for moderate horizons.

In Chapter 5, a Schur complement active-set method for MHE was presented. The method was motivated by the observation that for MHE typically a small number of constraints is expected to be active at the solution. Therefore, it uses the unconstrained MHE solution, which is efficiently computed by Riccati based methods, as a starting point. By projecting onto the reduced space of active constraints a small non-negativity constrained QP is obtained. If a square-root Riccati method is used, the Hessian of this reduced QP is computed by partial forward solves. The Schur complement method allows for multiple updates to the active set per iteration. Between active-set iterations the Hessian and gradient change only by addition constraints to the working set. This is because constraints are only added to prevent cycling. The proposed method exploits these Hessian changes by applying a Cholesky downdate. In order to solve the reduced non-negativity constrained QPs, a gradient projection method is proposed, consisting of two stages per iteration: a Cauchy step and a projected Newton step. The projected Newton step uses the Cholesky factorization of the Hessian and applies Cholesky updates during the iterations. The solution to the reduced QP is expanded by a Schur complement technique to update the primal and dual variables. It is demonstrated that the method typically needs only a few active-set iterations. C implementations demonstrated the performance of the novel Schur complement active-set method for MHE.

The MHE framework offers great flexibility in the problem formulation. Not only can constraints and nonlinear dynamics be imposed, but also non-standard objectives can be considered. The aim of Chapter 6 was to show that certain convex formulations may be more adequate than the least-squares formulation for certain problems and can still be solved by efficient Riccati based methods. Robust penalties such as the $\ell_1$ norm and the Huber penalty were investigated in the context of MHE with the aim of improved robustness with respect to outliers. It was concluded that the Huber penalty MHE formulation provides a performance improvement in case of outlier contaminated data while preserving the performance of the standard formulation when the frequency of outlier occurance is rather low. A second convex formulation

investigated in Chapter 6 concerned the joint input and state estimation for systems with piecewise-constant or piecewise-linear inputs. It is shown that this problem is best formulated as a cardinality problem which can be solved by a sequence of convex MHE problems. More specifically, it concerns an $\ell_1$ norm MHE problem and subsequent modifications which can be solved using dedicated Riccati based methods. The performance of the methods is shown through numerical simulations.

The developed quadratic MHE methods can be used for nonlinear MHE (NMHE) in an SQP algorithm. In Chapter 7, a NMHE method is investigated using multiple shooting and a constrained Gauss-Newton method. A major advantage of using Riccati-based methods at the core of an SQP strategy for NMHE is that the arrival cost updates and final estimate covariance are obtained as a natural outcome of the solution process. The NMHE method is demonstrated on some numerical examples. Finally, the application of an MHE based predictive control strategy to regulate the blood-glucose level in critically-ill patients at the intensive care unit was investigated. It was demonstrated that the moving horizon estimator accurately estimates the true states from noisy output measurements of the blood-glucose and is able to recover quickly from a wrong initial guess of the state vector. A target calculation was proposed to reduce the effect of disturbances and changing parameters. The ability of the closed-loop control system to regulate to the normoglycemic range in a short time span and to suppress disturbances was shown for a realistic disturbance realization. Despite model imperfections and large intra and inter patient variability, the proposed control system seems applicable for normalizing glycemia in the ICU and will possibly be tested in real-life circumstances in the near future.

## 8.2. Future work

### 8.2.1. Sum-of-norms regularization

Recently, another generalization of $\ell_1$ regularization for state smoothing was presented by Ohlsson [138]. Their (convex) formulation bears resemblances with the cardinality formulation proposed in Chapter 6 for MHE with abrupt input changes. Furture work will be directed to compare both approaches in accuracy and computational efficiency.

### 8.2.2. Emerging applications: fast and large-scale systems

The development of fast and robust methods creates a multiple of opportunities to expand the application scope of MHE and MHE based NMPC.

One emerging field of applications concerns *fast systems*, i.e., operating at a sampling rate of 1kHz or more. Applications are in mechatronics, automotive, power electronics, aerospace, and many other areas. In this thesis the focus was on fast structured methods for MHE optimization problems. Further progress towards ultra fast NMHE methods can be made by developing dedicated integration routines, since the integration and sensitivity generation is the most time consuming step in the current implementations. One promising direction for such a fast dedicated ODE/DAE solver could be an explicit Runge-Kutta scheme with limited number of stages or with a grid adapted to the measurements. Finally, automatic code generation and implementation on embedded hardware, *e.g.* FPGA or DSP, will further reduce computation times and widen the application scope.

On the other side of the spectrum are *large-scale systems*, i.e., with system orders of $10,000$ to one million or larger. Here applications arise for instance in process industry, and in forecasting and *nowcasting* of weather, oceans, climatology and ecosystems. While Riccati based methods are well-suited for small to medium scale systems and large horizon lengths due to their linear scaling with horizon length, they become intractable for (extremely) large-scale systems due to their cubic scaling with the system order. In this case, other structure-exploiting MHE methods should be developed. One possibility is to use preconditioned conjugate gradient methods. If the system dimensions are such that even a single state covariance cannot be stored, one could resort to reduced rank Kalman filter recursions or Ensemble Kalman filter type methods. However, the question arises if there is a benefit in using a larger window, *i.e.* MHE versus Kalman filtering, if a *crude* approximation to the full state evolution is made. Mostly, however, the large system dimensions stem from a discretization of partial differential equations. Therefore, instead of discretizing first and applying approximate methods or model reduction methods in order to make the large discretized system tractable, a better compromise is probably found in simultaneous discretization and optimization methods such as collocation with an adaptive grid. Also here, preconditioned conjugate gradient can be used to solve the optimization problems. PDE constrained MPC is an active research area, see *e.g.* [123].

### 8.2.3. Decentralized and distributed MHE

Decentralized or distributed MHE can also provide a solution to MHE for large-scale systems, but its scope is broader. It is especially useful when when there are different time scales into play. The fast dynamics can be considered in small scale estimation problems using fast MHE methods implemented on embedded systems, while slower time scales can be handled on a higher hierarchical level. The topic is being actively researched [53, 54].

### 8.2.4. Interaction between MHE and MPC

Although in reality the state of a dynamic system has to be estimated based on output measurements of the system, an MPC controller typically assumes that the current state is known exactly at all times. The MPC controller is designed assuming exact state information after which in reality it is then connected to a state estimator. When one uses an LQR controller combined with a Kalman filter one can prove that stability and optimality is preserved. However, due to the nonlinear nature of MPC controllers, this property does not hold anymore and one has to explicitly take into account the estimation errors. This methodology, called output feedback MPC, has received some attention in recent years [60]. but the existing results are only valid for rather restrictive settings. A future research direction could be to extend the existing results of output feedback MPC towards MHE based MPC. The interaction between MHE and MPC goes in two directions:

I Extension of the MPC algorithms to include the MHE estimation error. Instead of one current state a set of possible current states is used and a robust MPC is applied to this set. The set of states can for example be defined by the covariance of the state estimate possibly restricted to a feasible set.

II MPC algorithms that yield improved future state estimates. Recent results [100] show that it is possible to consider this interaction in the design of MPC controllers. When current estimates are uncertain, the system is probed in a direction where the uncertainty is present so as to improve the quality of future estimates. Researchers have already looked into the related topics of simultaneous regulation and model identification [65] and adaptive control, however, for the combination of MPC with MHE the topic is nearly unexplored apart from [100]. We believe this combination could lead to very strong results although finding formal proofs of asymptotic stability of the closed-loop system in the presence of constraints can be very difficult.

APPENDIX A

# Simplification of Riccati methods for case $H \equiv 0$

In Chapter 3 we presented Riccati based methods for a general MHE formulation (3.1), *i.e.* with $H \neq 0$. If $H_k = 0$ for all $k$, there is no *mixing* between $x$ and $w$ and the Riccati recursion involves covariances of order $n \times n$. In this case the algorithms can be simplified.

## A.1. Normal Riccati based method

If $H \equiv 0$, the $LU$ decomposition (see Lemma 3.2.1) modifies to

$$L = \begin{bmatrix} P_{0+}^{-1} & & & & & & & & \\ & Q_k^{-1} & & & & & & & \\ A_0 & G_0 & -P_1 & & & & & & \\ 0 & 0 & -I & P_{1+}^{-1} & & & & & \\ & & & & \ddots & & & & \\ & & & & & P_{N-1+}^{-1} & & & \\ & & & & & & Q_{N-1}^{-1} & & \\ & & & & & A_{N-1} & G_{N-1} & -P_N & \\ & & & & & 0 & 0 & -I_n & P_{N+}^{-1} \end{bmatrix},$$

$$U = \begin{bmatrix} I_n & 0 & P_{0+}A_0^{\mathrm{T}} & & & & & & & & \\ & I_m & Q_0 G_0^{\mathrm{T}} & & & & & & & & \\ & & I_n & P_1^{-1} & & & & & & & \\ & & & I_n & 0 & P_{1+}A_1^{\mathrm{T}} & & & & & \\ & & & & I_m & Q_1 G_1^{\mathrm{T}} & & & & & \\ & & & & & & \ddots & & & & \\ & & & & & & & I_n & & P_{N-1+}A_{N-1}^{\mathrm{T}} & \\ & & & & & & & & I_m & Q_{N-1}G_{N-1}^{\mathrm{T}} & \\ & & & & & & & & & I_n & P_N^{-1} \\ & & & & & & & & & & I_n \end{bmatrix},$$

The algorithms can be modified accordingly.

**Algorithm 1** (b). *[Riccati recursion, case $H \equiv 0$]*

1. *Initialization:* $P_0$
2. *For* $k = 0, \ldots, N-1$*:*
   - *a) Measurement update step*
   $$P_{k+} = P_k - P_k C_k^{\mathrm{T}} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-1} C_k P_k$$
   - *b) Model forwarding step*
   $$P_{k+1} = A_k P_{k+} A_k^{\mathrm{T}} + G_k Q_k G_k^{\mathrm{T}}$$

   *endfor.*
3. *Final time step:*
   - *a) Measurement update step*
   $$P_{N+} = P_N - P_N C_N^{\mathrm{T}} \left(R_N + C_N P_N C_N^{\mathrm{T}}\right)^{-1} C_N P_N$$

**Algorithm 4** (b). *[Forward vector recursion, case $H \equiv 0$]*

1. *Initialization:*

$$\hat{x}_0 \quad := \quad \hat{x}_0 - \bar{x_0}$$

2. *For $k = 0, \ldots, N-1$:*

$$\begin{aligned} x_k^{'} &= \hat{x}_k - P_k C_k^{\mathrm{T}} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-1} (C_k \hat{x}_k + h_k) \\ w_k^{'} &= -\bar{w}_k \\ \hat{x}_{k+1} &= f_k + A_k x_k^{'} + G_k w_k^{'} \end{aligned}$$

*endfor.*

3. *Final time step:*

$$x_N^{'} \quad = \quad \hat{x}_N - P_N C_N^{\mathrm{T}} \left(R_N + C_N P_N C_N^{\mathrm{T}}\right)^{-1} (C_N \hat{x}_N + h_N)$$

**Algorithm 5** (b). *[Backward vector recursion, case $H \equiv 0$]*

1. *Initialization ($k = N$:*

$$\begin{aligned} x_N &= x_N^{'} \\ \lambda_{N-1} &= C_N^{\mathrm{T}} \left(R_N + C_N P_N C_N^{\mathrm{T}}\right)^{-1} (C_N \hat{x}_N + h_N) \end{aligned}$$

2. *For $k = N-1, \ldots, 1$:*

$$\begin{aligned} w_k &= w_k^{'} - Q_k G_k^{\mathrm{T}} \lambda_k \\ x_k &= x_k^{'} - P_{k+} A_k^{\mathrm{T}} \lambda_k \\ \lambda_{k-1} &= A_k^{\mathrm{T}} \lambda_k + C_k^{\mathrm{T}} \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-1} \left[C_k (\hat{x}_k - P_k A_k^{\mathrm{T}} \lambda_k) + h_k\right] \end{aligned}$$

*endfor.*

3. *Final step ($k = 0$):*

$$\begin{aligned} w_0 &= w_0^{'} - Q_0 G_0^{\mathrm{T}} \lambda_0 \\ x_0 &= x_0^{'} - P_{0+} A_0^{\mathrm{T}} \lambda_0 \end{aligned}$$

## A.2. Square-root Riccati based method

Also for the square-root Riccati method, the algorithms can be simplified if $H \equiv 0$. In this case the $L$ factor of the $LDL^{\mathrm{T}}$ decomposition (see Lemma 3.2.5) modifies to

$$L = \begin{bmatrix} S_{0+}^{-1} & & & & & & & & \\ & W_0^{-1} & & & & & & & \\ A_0 S_{0+}^{T} & G_0 W_0^{\mathrm{T}} & S_1^{T} & & & & & & \\ 0 & 0 & -S_1^{-1} & S_{1+}^{-1} & & & & & \\ & & & & \ddots & & & & \\ & & & & & S_{N-1+}^{-1} & & & \\ & & & & & A_{N-1} S_{N-1+}^{T} & G_{N-1} W_{N-1}^{\mathrm{T}} & S_N^{T} & \\ & & & & & 0 & 0 & -S_N^{-1} & S_{N+}^{-1} \end{bmatrix},$$

And the algorithms are adapted accordingly as shown below.

**Algorithm 6** (b). *[Square-root Riccati recursion, case $H \equiv 0$]*

1. *Initialization:* $S_0$
2. *For* $k = 0, \ldots, N-1$*:*

   a) *Measurement update step*

   *Compute QR-factorization:*

   $$\begin{bmatrix} V_k & \\ S_k C_k^{\mathrm{T}} & S_k \end{bmatrix} = \begin{bmatrix} \bar{Q}_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & S_{k+} \end{bmatrix}$$

   *with* $\tilde{R}_k^e = \left(R_k + C_k P_k C_k^{\mathrm{T}}\right)^{-1/2}$ *and* $\tilde{K}_k = P_k C_k^{\mathrm{T}} \tilde{R}_k^e$.

   b) *Model forwarding step*

   *Compute QR-factorization:*

   $$\begin{bmatrix} S_{k+} A_k^{T} \\ W_k G_k^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \hat{Q}_{k+1} & \check{Q}_{k+1} \end{bmatrix} \begin{bmatrix} \hat{S}_{k+1} \\ 0 \end{bmatrix}$$

   *endfor.*

3. *Final time step:*

   a) *Measurement update step*

   *Compute QR-factorization:*

   $$\begin{bmatrix} V_N & \\ S_N C_N^{\mathrm{T}} & S_N \end{bmatrix} = \begin{bmatrix} \bar{Q}_N & \tilde{Q}_N \end{bmatrix} \begin{bmatrix} (\tilde{R}_N^e)^{-1} & \tilde{K}_N^{\mathrm{T}} \\ 0 & S_{N+} \end{bmatrix}$$

   *with* $\tilde{R}_N^e = \left(R_N + C_N P_N C_N^{\mathrm{T}}\right)^{-1/2}$ *and* $\tilde{K}_N = P_N C_N^{\mathrm{T}} \tilde{R}_N^e$.

**Algorithm 7** (b). *[Square-root forward vector recursion, case $H \equiv 0$]*

1. *Initialization:*
$$\hat{x}_0 \quad := \quad \hat{x}_0 - \bar{x_0}$$

2. *For $k = 0, \ldots, N-1$:*
$$\begin{aligned} x_k^{'} &= \hat{x}_k - \tilde{K}_k (\tilde{R}_k^e)^{\mathrm{T}} (C_k \hat{x}_k + h_k) \\ w_k^{'} &= -\bar{w}_k \\ \hat{x}_{k+1} &= f_k + A_k x_k^{'} + G_k w_k^{'} \end{aligned}$$
*endfor.*

3. *Final time step:*
$$x_N^{'} \quad = \quad \hat{x}_N - \tilde{K}_N (\tilde{R}_N^e)^{\mathrm{T}} (C_N \hat{x}_N + h_N)$$

**Algorithm 8** (b). *[Square-root backward vector recursion, case $H \equiv 0$]*

1. *Initialization ($k = N$:*
$$\begin{aligned} x_N &= x_N^{'} \\ \lambda_{N-1} &= C_N^{\mathrm{T}} \tilde{R}_N^e (\tilde{R}_N^e)^{\mathrm{T}} (C_N \hat{x}_N + h_N) \end{aligned}$$

2. *For $k = N-1, \ldots, 1$:*
$$\begin{aligned} w_k &= w_k^{'} - W_k^{\mathrm{T}} W_k G_k^{\mathrm{T}} \lambda_k \\ x_k &= x_k^{'} - S_{k+}^{\mathrm{T}} S_{k+} A_k^{\mathrm{T}} \lambda_k \\ \lambda_{k-1} &= A_k^{\mathrm{T}} \lambda_k + C_k^{\mathrm{T}} \tilde{R}_k^e (\tilde{R}_k^e)^{\mathrm{T}} \left[ C_k (\hat{x}_k - S_k^{\mathrm{T}} S_k A_k^{\mathrm{T}} \lambda_k) + h_k \right] \end{aligned}$$
*endfor.*

3. *Final step ($k = 0$):*
$$\begin{aligned} w_0 &= w_0^{'} - W_0^{\mathrm{T}} W_0 G_0^{\mathrm{T}} \lambda_0 \\ x_0 &= x_0^{'} - S_{0+}^{\mathrm{T}} S_{0+} A_0^{\mathrm{T}} \lambda_0 \end{aligned}$$

APPENDIX B

# QR factorization methods

This appendix considers QR factorization based on Householder reflections and Givens rotations. Both methods are known to have excellent numerically properties: both are stable. The Givens QR method is about twice as expensive as the Householder QR method, but can be more selective in zeroing elements and is therefore generally used or recommended for sparse matrices [66].

## B.1. Dense QR methods

The QR factorization of a rectangular matrix $A \in \mathbf{R}^{m\times n}$ with $m \geq n$ is given by

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{B.1}$$

with $Q_1 \in \mathbf{R}^{m\times n}$, $Q_2 \in \mathbf{R}^{m\times(n-m)}$ orthogonal and $R \in \mathbf{R}^{n\times n}$ triangular. If $A$ has full column rank, then the columns of $Q_1$ form an orthonormal basis for the range space of $A$. This computation is typically done by Householder or Givens transformations or by Gram-Schmidt orthogonalization. Here only Householder and Givens QR methods are discussed because numerical stability is guaranteed for these methods while this is not the case for (classical or modified) Gram-Schmidt.

### B.1.1. Householder QR methods

A *Householder reflection* is a symmetric and orthogonal matrix $H = I - \beta vv^{\mathrm{T}}$, where $b$ is a scalar and $v$ is a column vector of length $n$. Applying $H$ to any vector of

length $n$ takes about $3n$ flops compared to a general matrix vector product which requires about $2n^2$ flops. Geometrically the operation $Hx$ can be interpreted as a reflection of the vector $x$ through the plane with normal vector $v$. The importance of Householder reflections follows from the orthogonality and the fact that by choosing $b$ and $v$ appropriately, the reflection can be used to zero selected components of a vector. The Householder reflection is a rank-one modification of the identity.

The algorithm below, written in C, computes the QR factorization of $A$ based on Householder reflections.

**Algorithm 9.** *[Householder QR.]*

***for*** $k = 1$ *to* $n$ ***do***
  $v = A_{k:m,k}$
  $\rho = \max(|v|)$
  $r = 0$
  ***for*** $i = 1$ *to* $m-k+1$ ***do***
    $v_i = v_i/\rho$
    $r = r + v_i v_i$
  ***end for***
  ***if*** $v_1 > 0$ ***then***
    $\sigma = \sqrt{r}$
  ***else***
    $\sigma = -\sqrt{r}$
  ***end if***
  $t = \sigma v_1 + r$
  $\beta = 1/t$
  $v_1 = v_1 + \sigma$
  $A_{k,k} = -\rho\sigma$
  ***for*** $j = k+1$ *to* $n$ ***do***
    $l = 0$
    ***for*** $i = k$ *to* $m$ ***do***
      $l = l + v_{i-k+1} A_{i,j}$
    ***end for***
    $l = \beta l$
    ***for*** $i = k$ *to* $m$ ***do***
      $A_{i,j} = A_{i,j} - v_{i-k+1} l$
    ***end for***
  ***end for***
***end for***

This algorithm requires about $n^2(m - \frac{n}{3}) = 2mn^2 - \frac{2}{3}n^2$ flops (for $m$ and $n$ large). The algorithm applies a preliminary scaling to avoid overflow. Note that the algortihm

does not compute an explicit representation of $Q$ (*Q-less QR factorization*). If $Q$ is needed, it can be computed out of the vectors $v$.

### B.1.2. Givens QR methods

A Givens rotation is an antisymmetric and orthogonal $n$ by $n$ matrix

$$G(i,k,\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ i \\ \\ k \\ \\ \\ \end{matrix}$$
$$\begin{matrix} & & i & & k & & \end{matrix}$$

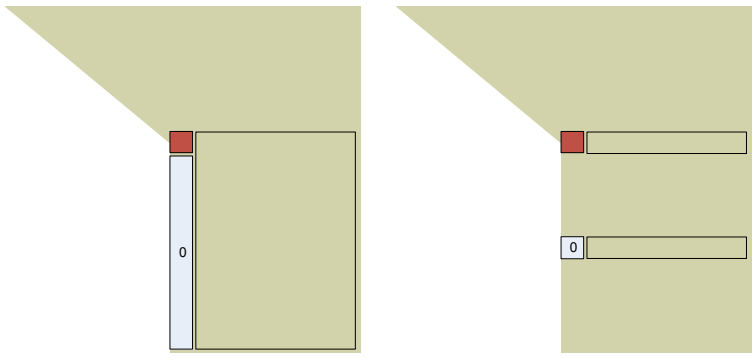
with $c = \cos(\theta)$ and $s = \sin(\theta)$ for some $\theta$ (consequently $c^2 + s^2 = 1$). Premultiplying a vector with $G(i,k,\theta)^{\mathrm{T}}$ can be interpreted as a counterclockwise rotation in the $(i,k)$ plane over $\theta$ radians. This operation requires about $6n$ flops. The Givens rotation is a rank-two modification of the identity.

The standard Householder QR method proceeds by zeroing the elements below the diagonal column by column. The Givens method can be implemented in a row-wise or column-wise elimination form. Both have instead been preferred and used [129]. The column-wise strategy uses the same elimination order as used in the Householder method. The Householder and column-wise Givens method are illustrated in Figure B.1. It is well known that application of a Householder reflection or a Givens rotation to a matrix with two rows yields identical results [66, 69], which can be seen from the figure.

The algorithm below computes the QR factorization of $A$ based on Givens rotations.

**Algorithm 10.** *[Givens QR.]*

***for*** $k = 1$ *to* $n$ ***do***
  ***for*** $j = k+1$ *to* $m$ ***do***
    ***if*** $|A_{j,k}| == 0$ ***then***
      $s = 0$
      $c = 1$
    ***else if*** $|A_{j,k}| > |A_{k,k}|$ ***then***
      $t = -\frac{A_{k,k}}{A_{j,k}}$




(a) (b)

**Figure B.1.** Illustration of QR factorization using Householder reflections (left) and using Givens rotations (right).

$s = \frac{1}{\sqrt{1+tt}}$
$c = st$
***else***
$t = -\frac{A_{j,k}}{A_{k,k}}$
$c = \frac{1}{\sqrt{1+tt}}$
$s = ct$
***end if***
$A_{k,k} = cA_{k,k} - sA_{j,k}$
***for*** $i = k+1$ *to* $n$ ***do***
$x = cA_{k,i} - sA_{j,i}$
$y = sA_{k,i} + cA_{j,i}$
$A_{k,i} = x$
$A_{j,i} = y$
***end for***
***end for***
***end for***

This algorithm requires about $3n^2(m - \frac{n}{3})$ flops (for $m$ and $n$ large). The algorithm guards against overflow. Note that the angle $\theta$ does not need to be computed and no trigonometric functions are required.

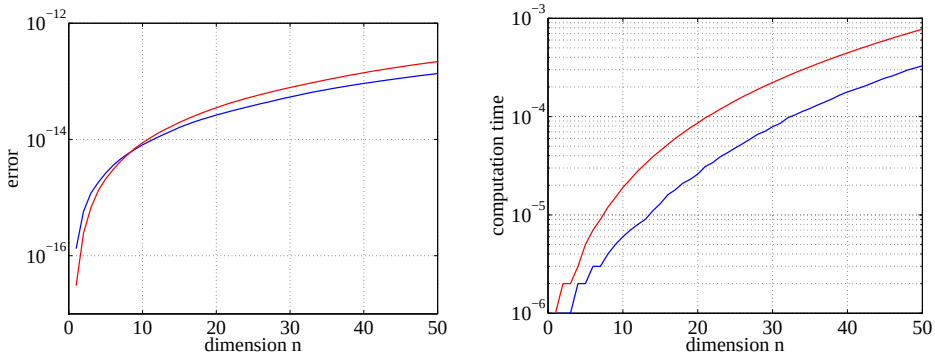



**Figure B.2.** Numerical results for dense Householder (blue line) and Givens (red line) QR methods on typical time update matrices of Riccati based MHE algorithms, *i.e.* $m+n$ rows with $m=n$ and $n$ columns, simulated with random data. Left: error versus number of columns. Right: computation time versus number of columns.

## B.1.3. Numerical results

We have implemented the standard Householder and Givens QR methods in C and simulated them on typical time update matrices of Riccati based MHE algorithms, *i.e.* $m+n$ rows with $m=n$ and $n$ columns. Numerical results were obtained for randomly generated matrices sampled from the standard normal distribution[1] and averaged[2] over 500 repetitions. Computing times correspond to an Intel Core2-Duo processor at 2.13 GHz with 2 MB cache and 2 GB RAM, and using the compiler gcc version 4.4.5. The results are presented in Table B.1 and Figure B.2. It can be seen from these results that the Givens method is more accurate for small systems, up to $n=7$, for moderate or large systems Householder is more accurate by about a factor 1.5. The Householder method is consistently faster than the Givens method by about a factor 2 as predicted by the flop counts. The error of QR factorization can be computed by checking the orthogonality of $Q$, *i.e.* evaluating[3] $\|Q^{\mathrm{T}}Q-I\|_2$. Since this measure does not take the error in the $R$ matrix into account which is our primary concern, also the error $\|QR-A\|_2$ was calculated. We observed that the trends for both errors are actually very similar. In the figures only this error is shown.

[1] Using the random number generator from GNU Scientific Library (GSL)

[2] The median was used rather than the mean in order to remove bursts originating from processes running in the background.

[3] The Frobenius norm was used for matrix norms throughout these numerical simulations

**Table B.1.** Numerical results for dense Householder and Givens QR methods on typical time update matrices of Riccati based MHE algorithms, *i.e.* $m+n$ rows with $m=n$ and $n$ columns, simulated with random data.

| | | avg time (s) | avg error $\|Q^{\mathrm{T}}Q-I\|_2$ | avg error $\|QR-A\|_2$ |
|---|---|---|---|---|
| n = m = 2 | Householder | 1,00E-06 | 2,75E-16 | 5,78E-16 |
| | Givens | 2,00E-06 | 1,19E-16 | 2,45E-16 |
| n = m = 5 | Householder | 2,00E-06 | 4,64E-16 | 2,62E-15 |
| | Givens | 5,00E-06 | 3,81E-16 | 2,10E-15 |
| n = m = 10 | Householder | 6,00E-06 | 6,86E-16 | 8,06E-15 |
| | Givens | 1,90E-05 | 7,30E-16 | 8,75E-15 |
| n = m = 20 | Householder | 2,60E-05 | 1,08E-15 | 2,63E-14 |
| | Givens | 8,60E-05 | 1,39E-15 | 3,51E-14 |
| n = m = 50 | Householder | 3,28E-04 | 2,12E-15 | 1,36E-13 |
| | Givens | 7,73E-04 | 3,35E-15 | 2,19E-13 |

## B.2. Structured QR methods

When applying direct methods to sparse systems, the factorization process is usually preceded by an *ordering and symbolic analysis* phase. This phase typically involves graph theoretical models to find optimal row and column orderings and to construct elimination trees. If the sparsity pattern is known in advance, it can be imposed during the factorization proces and this preceding phase is omitted.

Consider the QR factorization in the measurement update step of the Riccati based MHE algorithms investigated in this thesis

$$\begin{bmatrix} V_k & \\ T_k D_k^{\mathrm{T}} & T_k \end{bmatrix} = Q \begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}$$

Following the dense Householder QR procedure, large and sparse matrices lead to long and sparse Householder vectors . This can, however, easily be avoided by sorting the rows in such a way that rows with leading nonzero element in the first column are permuted first in the matrix, *i.e.* a staircase row ordering. This simple rule of thumb can have substantial impact on the amount of intermediate fill-in and work associated with the QR procedure[39]. Since $V_k$ is upper triangular, applying the rule of thumb, yields the following row-reordered fatcorization

$$\underbrace{\begin{bmatrix} T_k D_k^{\mathrm{T}} & T_k \\ V_k & \end{bmatrix}}_{A} = \tilde{Q} \underbrace{\begin{bmatrix} (\tilde{R}_k^e)^{-1} & \tilde{K}_k^{\mathrm{T}} \\ 0 & T_{k+} \end{bmatrix}}_{\tilde{R}}$$

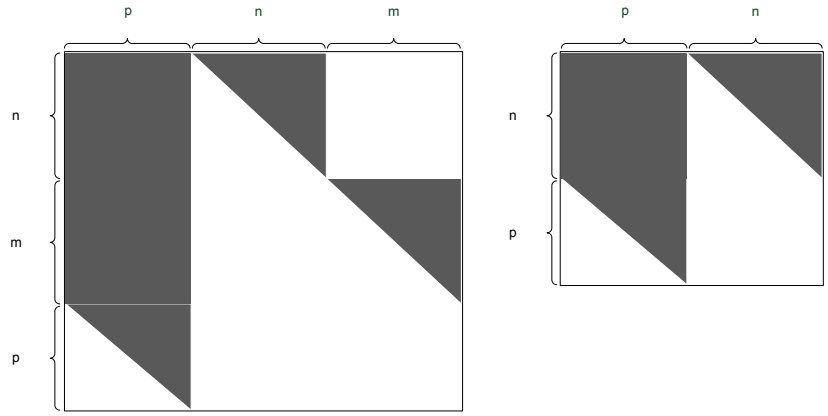



**Figure B.3.** Visualization of the block structure in the original matrix for the general case (left) and for the case $H \equiv 0$ (right).

Note that the R-factor is unaffected by the row re-ordering. The sparsity pattern of the matrices encountered in the Riccati based MHE algorithms is depicted in Figure B.3. This row-reordering for the measurement update step is known to lead to computational savings in the factorizations, see standard textbooks, *i.e.* by Verhaegen and Verduld [190] or Kailath et al [111].

Given a fixed column ordering, it is clear that columns in general must be eliminated from left to right. Fill-in is otherwise introduced in already processed columns. However, in some cases the sparsity pattern is such that sets of columns are structurally independent and it becomes possible to eliminate in any order, or in parallel. This observation forms the basis of the multifrontal sparse QR factorization methods [96]. Unfortunately, the matrices considered in this thesis in general do not fall into this class. It can be shown that, although the original matrix $A$ is quite sparse (see Figure B.3), the R-factor is in general not sparse. Nevertheless, the sparsity can be exploited during the QR-iterations. Structured versions of QR methods using Householder reflections and Givens rotations have been implemented for the matrices considered in this thesis and are compared in this section.

### B.2.1. Structured Householder QR method

The row re-ordering can yield a substantial reduction in flops since the first $p$ iterations deals shorter vectors and matrices. The zeros top right for the general case (see Figure B.3) and the zeros due to triangularity of $S_k$ and $W_k$ can be exploited in the first iteration only for the structured Householder QR. After the first iteration this spar-

sity is destroyed. Nevertheless, for small systems the reduction in operations can be significant.

### B.2.2. Structured Givens QR method

Similarly to the structured Householder method the row re-ordering can yield a substantial reduction in flops for the first $p$ iterations for the structured Givens method. The zeros top right for the general case (see Figure B.3) can be exploited during the first $n$ iterations. The zeros due to triangularity of $S_k$ and $W_k$ can be exploited in the first iteration only. After the first iteration this sparsity is destroyed. The sparsity in the matrices can be exploited more in the structured Givens method than in the structured Householder method.

### B.2.3. Numerical results

We have implemented structured versions of Householder and Givens QR methods that fully exploit the sparisty pattern in C and simulated them on typical measurement update matrices of Riccati based MHE algorithms.

Again numerical results were obtained for randomly generated matrices sampled from the standard normal distribution and averaged over 500 repetitions. Computing time corresponds to an Intel Core2-Duo processor at 2.13 GHz with 2 MB cache and 2 GB RAM, and using the compiler gcc version 4.4.5. The results are presented in Table B.2 and in Figures B.4 and B.5 for the case $H \equiv 0$ and for the general case respectively. It can be seen from these results that also for these cases the Givens method is more accurate for small systems, up to $n = 7$, while the Householder is more accurate for moderate or large systems. The Householder method is cosistently faster than the Givens method by about a factor 2 as predicted by the flop counts. Furthermore, the structured versions of the QR methods are consistently faster (apart from some anomalies in the simulation results). The speed improvement is more pronounced for large $p$. Note that the accuracy of the structured versions is identical to the standard versions since the same operations are involved, only sparsity is exploited. Therefore, the developed structured QR methods are used throughout this thesis.

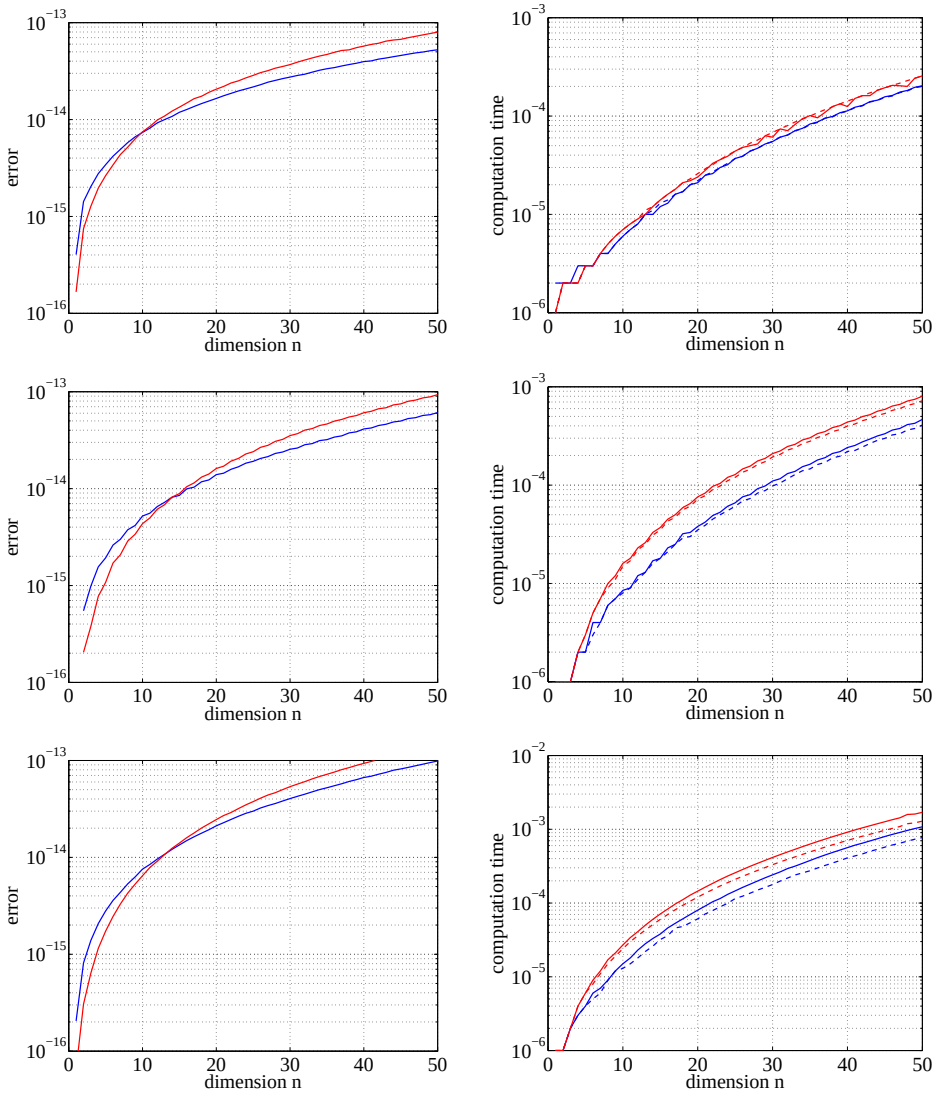



**Figure B.4.** Numerical results for structured Householder and Givens QR methods on typical measurement update matrices of Riccati based MHE algorithms for the case $H \equiv 0$, *i.e.* $n+p$ rows and $n+p$ columns, simulated with random data. Left: error versus dimension $n$. Right: computation time versus dimension $n$. Top: $p = 2$; middle: $p = n/2$; bottom: $p = n$. Standard Givens in solid red, structured Givens in dashed red, standard Householder in solid blue, structured Householder in dashed blue. Note that the error is identical for the standard and structured method, since it involves the same computations except sparsity is exploited.

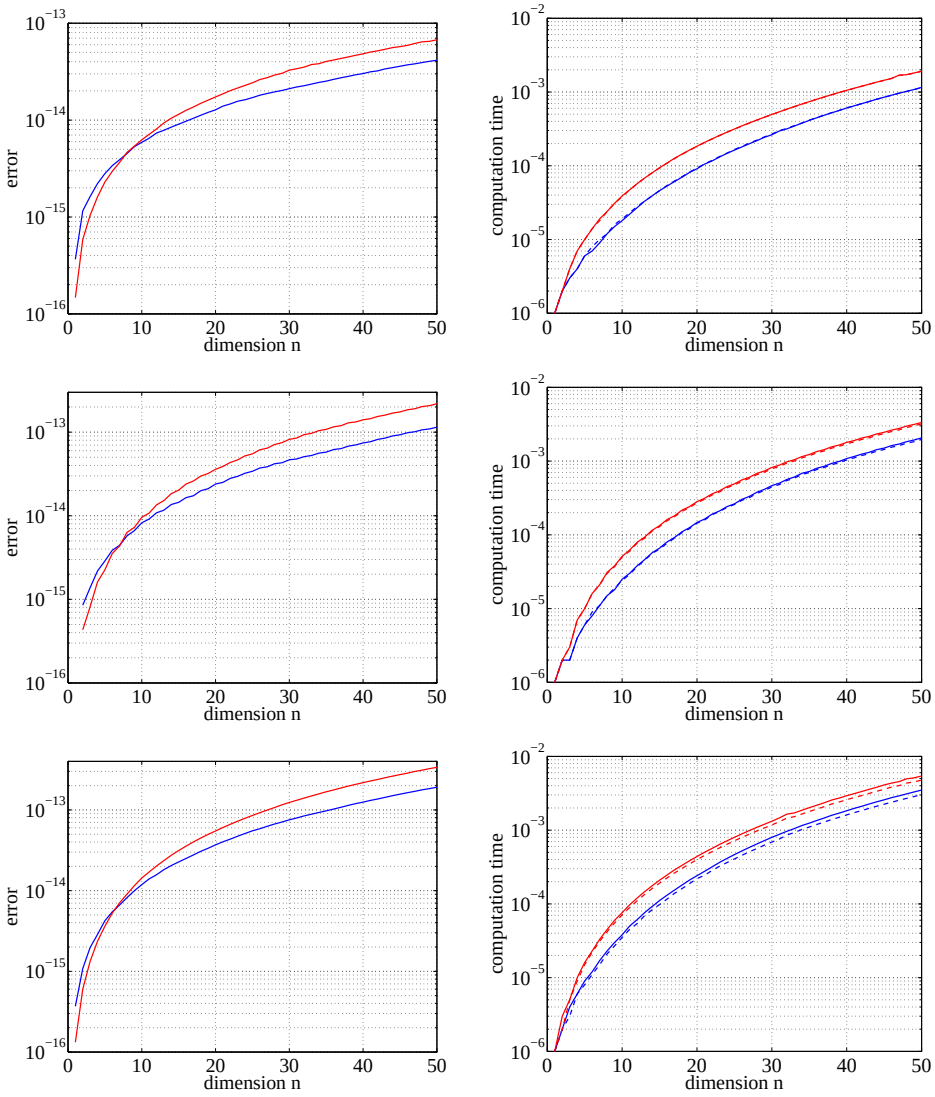



**Figure B.5.** Numerical results for structured Householder and Givens QR methods on typical measurement update matrices of Riccati based MHE algorithms for the general case, *i.e.* $n+m+p$ rows and $n+m+p$ columns, simulated with random data. Left: error versus number of columns. Right: computation time versus number of columns. Left: error versus dimension $n$. Right: computation time versus dimension $n$. Top: $p=2$; middle: $p=n/2$; bottom: $p=n$. We set $m=n$ throughout. Standard Givens in solid red, structured Givens in dashed red, standard Householder in solid blue, structured Householder in dashed blue. Note that the error is identical for the standard and structured method, since it involves the same computations except sparsity is exploited.

**Table B.2.** Numerical results for dense Householder and Givens QR methods on typical measurement update matrices of Riccati based MHE algorithms for the case $H \equiv 0$, *i.e.* $n+p$ rows and $n+p$ columns, simulated with random data.

| | | (s) | avg time $\|Q^{\mathrm{T}}Q-I\|_2$ | avg error $\|QR-A\|_2$ | avg error |
|---|---|---|---|---|---|
| n = m = 2 | p = 2 | Householder | 1,00E-06 | 5,14E-16 | 8,14E-16 |
| | | Givens | 1,00E-06 | 2,29E-16 | 3,05E-16 |
| n = m = 5 | p = 2 | Householder | 3,00E-06 | 1,48E-15 | 3,42E-15 |
| | | Givens | 3,00E-06 | 1,07E-15 | 2,64E-15 |
| | p = 5 | Householder | 4,00E-06 | 9,56E-16 | 2,80E-15 |
| | | Givens | 6,00E-06 | 6,16E-16 | 1,73E-15 |
| n = m = 10 | p = 2 | Householder | 6,00E-06 | 2,20E-15 | 7,36E-15 |
| | | Givens | 7,00E-06 | 2,01E-15 | 7,47E-15 |
| | p = 10 | Householder | 1,30E-05 | 1,47E-15 | 7,57E-15 |
| | | Givens | 2,40E-05 | 1,25E-15 | 6,46E-15 |
| n = m = 20 | p = 2 | Householder | 2,20E-05 | 3,51E-15 | 1,65E-14 |
| | | Givens | 2,60E-05 | 3,82E-15 | 2,06E-14 |
| | p = 20 | Householder | 6,10E-05 | 2,35E-15 | 2,12E-14 |
| | | Givens | 1,20E-04 | 2,46E-15 | 2,45E-14 |
| n = m = 50 | p = 2 | Householder | 2,00E-04 | 7,15E-15 | 5,25E-14 |
| | | Givens | 2,56E-04 | 9,19E-15 | 8,00E-14 |
| | p = 50 | Householder | 7,72E-04 | 4,89E-15 | 9,86E-14 |
| | | Givens | 1,29E-03 | 5,96E-15 | 1,45E-13 |

APPENDIX C

# Robust measures

In this appendix we compare the robustness of several commonly encountered penalty functions. For every penalty function, we investigate (1) its associated derivative function or influence function $psi(u) = \frac{d\rho(u)}{du}$ which measures the influence of a residual on the estimate, and (2) the weight function $\frac{\psi(u)}{u}$ which measures the relative weight that is given to a certain residual

First, let us compare two commonly used penalty functions, $\rho_1(u) = |u|$, associated with the $\ell_1$-norm and $\rho_2(u) = u^2$, associated with the $\ell_2$-norm. For small $u$ we have $\rho_1(u) \gg \rho_2(u)$, so $\ell_1$-norm approximation puts relatively larger emphasis on small residuals compared to $\ell_2$-norm approximation. For large $u$ we have $\rho_2(u) \gg \rho_1(u)$, so $\ell_1$-norm approximation puts less weight on large residuals, compared to $\ell_2$-norm approximation. Therefore, the solution of the an estimation problem using $\ell_1$-norms is less influenced by large residuals or outliers. Although the $\ell_1$-norm shows improved robustness with respect to outliers, it also exhibits some undesirable effects. The $\ell_1$-norm solution will tend to have more zero residuals, compared to the $\ell_2$-norm solution, but very few small residuals, which may not always be desirable, *e.g.* , in the case of normally distributed residuals contaminated with occasional outliers. Moreover, the $\ell_1$-norm is not *stable* because $\rho_1(u)$ is not strictly convex, *i.e.* the second derivative of $\rho_1(u)$ at zero is infinite and an indeterminant solution may result.

The advantages and disadvantages of both norms discussed above, have inspired researchers to propose hybrid $\ell_1$ - $\ell_2$ measures. One such hybrid $\ell_1$ - $\ell_2$ measure is the penalty function

$$\rho_{\text{hub}}(u) = \begin{cases} u^2 & |u| \leq M \\ M(2|u| - M) & |u| > M, \end{cases}$$

which is sometimes called *quadratic-tangent penalty function*, but is usually referred to as *Huber penalty function* after its proposer Huber [102]. This penalty function agrees with $\rho_2(u)$ for $u$ smaller than $M$, and then reverts to $\ell_1$-like linear growth for larger $u$. For vectors, we define $\rho_{\text{hub}}(u)$ to be the sum of the Huber penalty function applied componentwise.
Another hybrid $\ell_1$ - $\ell_2$ measure is

$$\rho_{\text{sh}}(u) = M(\sqrt{M^2+u^2} - M), \tag{C.1}$$

which is referred to as *smooth hybrid penalty function*. Like the Huber penalty function it corresponds to $\rho_2(u)$ for small $u$ and it resembles $\rho_1(u)$ for large $u$, but as opposed to the Huber penalty function it does not have a cut off point. The parameter $M$ allows to control the smooth transition from quadratic to linear. Again, for vectors, we define $\rho_{\text{sh}}(u)$ to be the sum of the smooth hybrid penalty function applied componentwise. From Figure C.1 it can be seen that

- the $\ell_2$-norm is *not* a robust measure, since its influence function is unbounded, *i.e.* extremely large residuals have extremely large influence,
- the $\ell_1$-norm is a robust measure - its influence function is bounded, implying that less relative weight is given to larger residuals - but it is not stable,
- the Huber penalty and the smooth hybrid penalty are robust and stable measures.

The robust penalties described above are still influenced by extreme values, albeit less than the $\ell_2$-norm. For a measure to be extremely robust, the influence function should fall to zero quickly after the cut off point, with the ultimate robust measure being the *quadratic-constant penalty function*

$$\rho_{\text{qc}}(u) = \begin{cases} u^2 & |u| \leq M \\ M^2 & |u| > M, \end{cases}$$

which nullifies the influence of outliers. However, this requirement of decreasing influence function unavoidably leads to non-convex penalty functions. Smooth approximations to the quadratic-constant penalty are given amongst others by the Cauchy function $M^2 \log\left(1+(u/M)^2\right)$, which provides a linear decay of influence of gross errors, or the Welsch function $M^2\left[1-\exp\left(-(u/M)^2\right)\right]$, which further reduces the effect of large errors. Both functions cause numerical issues due to non-convexity and moreover they tend to over-smooth the data [194]. Other, even more robust yet complex measures are the Tukey bi-weight criterion or Hampel's three-part redescending function [86]. The Huber penalty function is the convex function which approximates the quadratic-constant pentalty function closest and has proven to be very useful in practical applications [29, 58, 84, 85, 104, 122]. It can also be used as an convex initializer for numerically hard non-convex measures such as the Hampel function [205]. According to Zhang [205] a robust esimator should be strictly convex and have a bounded influence function.

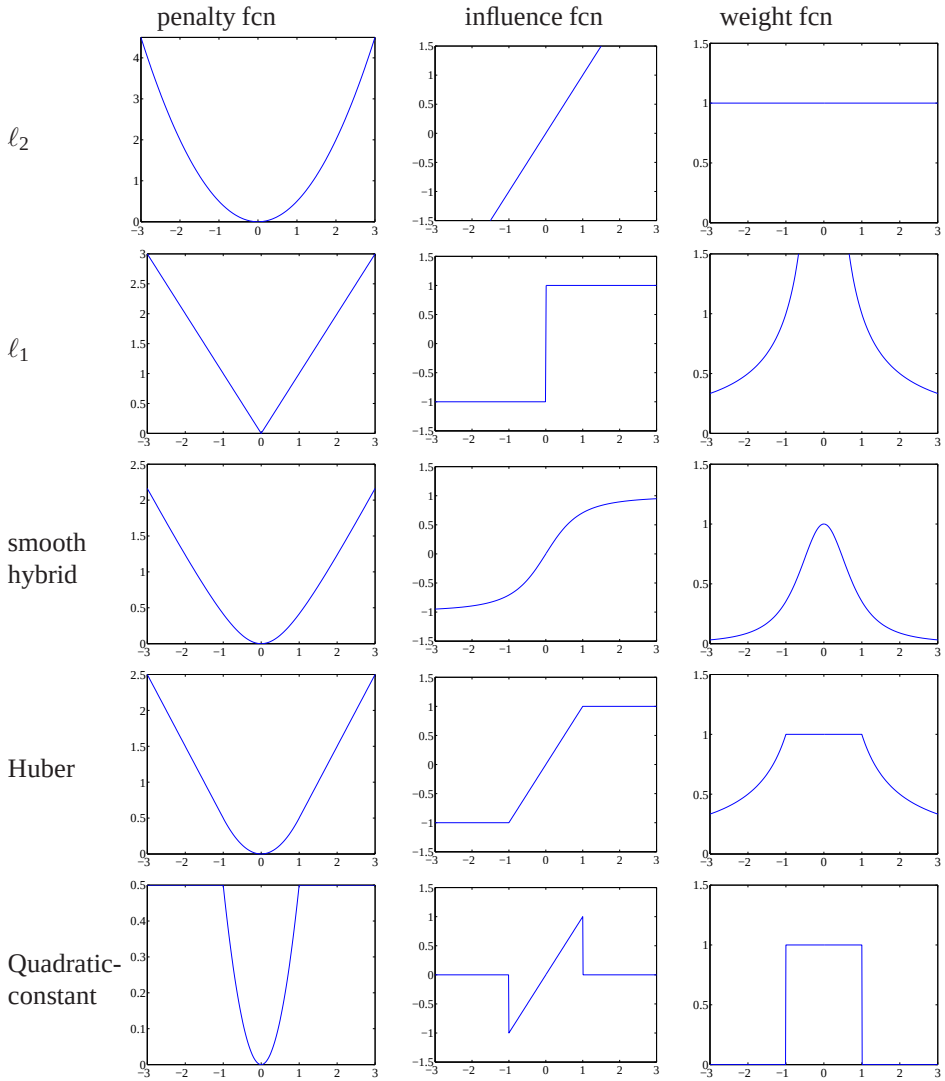



**Figure C.1.** Graphical representation of some commonly used penalty functions.

# Bibliography


[1] M. Alamir and L. A. Calvillo-Corona. Further results on nonlinear receding horizon observers. *IEEE Transactions on Automatic Control*, 47(7):1188–1193, 2002.

[2] M. Alamir and J. P. Corriou. Nonlinear receding-horizon state estimation for dispersive adsorption columns with nonlinear isotherm. *J. Process. Contr.*, 13:517–523, 2003.

[3] J.S. Albuquerque and L.T. Biegler. Decomposition algorithms for on-line estimation with nonlinear models. *Computers Chem. Eng.*, 19(10):1031–1039, 1995.

[4] A. Alessandri, M. Baglietto, and G. Battistelli. Receding-horizon estimation for discrete-time linear systems. *IEEE Transactions on Automatic Control*, 48:473–478, 2003.

[5] A. Alessandri, M. Baglietto, and G. Battistelli. Moving-horizon state estimation for nonlinear discrete-time systems: New stability results and approximation schemes. *Automatica*, 44:1753–1765, 2008.

[6] B.D. Anderson and J.B. Moore. *Optimal Filtering*. Dover Publications, January 2005.

[7] D. Axehill and A. Hansson. A dual gradient projection quadratic programming algorithm tailored for mixed integer predictive control. Technical Report LiTH-ISY-R-2833, Linköping University, 2008.

[8] 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.

[9] M. Barni, V. Cappellini, and L. Mirri. Multichannel m-filtering for color image restoration. In *Image Processing, 2000. Proceedings. 2000 International Conference on*, volume 1, pages 529 –532 vol.1, 2000.

[10] R.A. Bartlett, L.T. Biegler, J. Backstrom, and V. Gopal. Quadratic programming algorithms for large-scale model predictive control. *Journal of Process Control*, 12:775–795, 2002.

[11] R.A. Bartlett, A. Wächter, and L.T. Biegler. Active set vs interior point strategies for model predictive control. In *Proceedings of the American Control Conference*, pages 4229–4233, Chicago, IL, 2000.

[12] R.E. Bellman. *Dynamic programming*. Princeton University Press, Princeton, New Jersey, 1957.

[13] R.N. Bergman, L.S. Phillips, and C. Cobelli. Physiologic evaluation of factors controlling glucose tolerance in man: measurement of insulin sensitivity and beta-cell glucose sensitivity from the response to intravenous glucose. *J Clin Invest*, 68:1456–1467, Dec 1981.

[14] D.S. Bernstein. *Matrix Mathematics: Theory, Facts and Formulas.* Princeton University Press, Princeton, New Jersey, 2005.

[15] D. Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on control and optimization*, 20:222–246, 1982.

[16] J.T. Betts. *Practical methods for optimal control using nonlinear programming*. SIAM, Philadelphia, PA, 2001.

[17] L.T. Biegler. Efficient solution of dynamic optimization and NMPC problems. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, volume 26 of *Progress in Systems Theory*, pages 219–244, Basel Boston Berlin, 2000. Birkhäuser.

[18] L.T. Biegler and I.E. Grossmann. Retrospective on optimization. *Computers and Chemical Engineering*, 28:1169–1192, 2004.

[19] T. Binder, L. Blank, H.G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schlöder, and O.v. Stryk. Introduction to Model Based Optimization of Chemical Processes on Moving Horizons. In M. Grötschel, S.O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems: State of the Art*, pages 295–340. Springer, 2001.

[20] Th. Binder, L. Blank, W. Dahmen, and W. Marquardt. On the regularization of dynamic data reconciliation problems. *J. Proc. Cont.*, 12(4):557–567, 2002.

[21] Å. Björck. *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia, 1996.

[22] A. Björk. *Least Squares Methods: Handbook of Numerical Analysis Vol. 1*. Elsevier, North Holland, Amsterdam, 1990.

[23] H.G. Bock. *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen*, volume 183 of *Bonner Mathematische Schriften*. Universität Bonn, Bonn, 1987.

[24] H.G. Bock, E.A. Kostina, and J.P. Schlöder. On the role of natural level functions to achieve global convergence for damped Newton methods. In M. Powell et al, editor, *System Modelling and Optimization. Methods, Theory and Applications*, pages 51–74. Kluwer, 2000.

[25] H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings 9th IFAC World Congress Budapest*, pages 243–247, 1984.

[26] N.L. Boland. A dual active-set algorithm for positive semi-definite quadratic programming. *Mathematical Programming*, 78:1–27, 1997.

[27] J. Bouckaert and D. Telen. Modellering van glucosedynamica bij kritiek zieke patiÃ«nten. Master's thesis, Katholieke Universiteit Leuven, 2011.

[28] S. Boyd and L. Vandenberghe. Additional exercises for convex optimization. Available at http://www. stanford. edu/ boyd/cvx.

[29] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[30] A.E. Bryson and M. Frazier. Smoothing for linear and nonlinear dynamic systems. Technical Report ASD-RDR-063-119, U.S. Airforce, 1963.

[31] E.F. Camacho and C. Bordóns. *Model Predictive Control in the Process Industry*. Springer-Verlag, 1995.

[32] A.B. Chan, N. Vasconcelos, and G.R. Lanckriet. Direct convex relaxations of sparse SVM. In *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

[33] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, 2000.

[34] H. Cox. On the estimation of state variables and parameters for noisy dynamic systems. *IEEE Transactions on Automatic Control*, 9, 1964.

[35] J.L. Crassidis and J.L. Junkins. *Optimal estimation of synamic systems*. Chapman and Hall, London, 2004.

[36] M.L. Darby and M. Nikolaou. A parametric programming approach to moving-horizon state estimation. *Automatica*, 43(5):885–891, 2007.

[37] A. d'Aspremont, L. El Ghaoui, M.I. Jordan, and G.R. Lanckriet. A direct formulation for sparse pca using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

[38] J. Datorro. *Convex Optimization and Euclidean Distance Gemoetry*. Mεβoo Publishing, USA, 2005.

[39] T. Davis. *Direct Methods for Sparse Linear Systems*. Part of the SIAM Book Series on the Fundamentals of Algorithms. SIAM.

[40] T.A. Davis and W.W. Hager. Row modifications of a sparse cholesky factorization. *SIAM Journal on Matrix Analysis and Applications*, 26:1–16, 2005.

[41] P. Deuflhard. *Newton Methods for Nonlinear Problems.* Springer, New York, 2004.

[42] M. Diehl. *Real-Time Optimization for Large Scale Nonlinear Processes*. PhD thesis, Universität Heidelberg, 2001. http://www.ub.uni-heidelberg.de/archiv/1659/.

[43] M. Diehl, H.G. Bock, and J.P. Schlöder. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, 43(5):1714–1736, 2005.

[44] M. Diehl, H. J. Ferreau, and N. Haverbeke. *Nonlinear model predictive control*, volume 384 of *Lecture Notes in Control and Information Sciences*, chapter Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation, pages 391–417. Springer, 2009.

[45] M. Diehl, P. Kühl, H.G. Bock, J.P. Schlöder, B. Mahn, and J. Kallrath. Combined NMPC and MHE for a copolymerization process. In W. Marquardt and C. Pantelides, editors, *Computer-aided chemical engineering*, volume 21B, pages 1527–1532. DECHEMA, Elsevier, 2006.

[46] M. Diehl, I. Uslu, R. Findeisen, S. Schwarzkopf, F. Allgöwer, H.G. Bock, T. Bürner, E.D. Gilles, A. Kienle, J.P. Schlöder, and E. Stein. Real-Time Optimization for Large Scale Processes: Nonlinear Model Predictive Control of a High Purity Distillation Column. In M. Grötschel, S. O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems: State of the Art*, pages 363–384. Springer, 2001.

[47] A. E. Dieses, J.P. Schlöder, H.G. Bock, and O. Richter. Parameter Estimation for nonlinear transport and degradation processes of xenobiotica in soil. In F. Keil et al., editor, *Scientific Computing in Chemical Engineering II*, volume 2, pages 290–297. Springer Verlag, 1999.

[48] J. Dongara, C. Moler, J. Bunch, and G. Stewart. LINPACK user's guide. *Society for industrial and applied mathematics*, 1979.

[49] D. Donoho. Compressed sensing. *IEEE Trans on Information Theory*, 52(2):1289–1306.

[50] I.S. Duff. Ma27: a set of Fortran subroutines for solving sparse symmetric sets of linear equations. Technical Report R-10533, Computer Science and Systems division, AERE Harwell, Oxford, England, 1982.

[51] I.S. Duff. Ma57 - ACM. *Transactions on Mathematical Software*, 30:118–144, 2004.

[52] European Commission DG Information Society and Media. Workshop (9 oktober 2008) on monitoring and control – today's market, its evolution till 2020 and the impact of ICT on these. Report prepared by Decision and RPA; Available for download at `http://www.decision.eu/smart2007.htm`.

[53] M. Farina, G. Ferrari-Trecate, and R. Scattolini. State estimation for large-scale partitioned systems: a moving horizon approach. In *Proceedings of the 48th Conference on Decision and Control (CDC 2009)*, pages 1818–1823, Shangai, China, 2009.

[54] M. Farina, G. Ferrari-Trecate, and R. Scattolini. A moving horizon scheme for distributed state estimation. In *Proceedings of the 2010 American Control Conference (ACC)*, Baltimore, MD, 2010.

[55] H.J. Ferreau. qpOASES. `http://www.kuleuven.be/optec/software/qpOASES`.

[56] H.J. Ferreau. An online active set strategy for fast solution of parametric quadratic programs with applications to predictive engine control. Master's thesis, University of Heidelberg, 2006.

[57] H.J. Ferreau, H.G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit mpc. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.

[58] J.B. Poulsen N.K. Madsen H. Finan, D.A. JÃ¸rgensen. Robust model identification applied to type 1 diabetes. In *Proceedings of the American Control Conference (ACC)*, pages 2021 – 2026, Baltimore, MD, 2010.

[59] P.K. Findeisen. Moving horizon estimation of discrete time systems. Master's thesis, University of Wisconsin-Madison, July 1997.

[60] R. Findeisen, L. Imsland, F. Allgöwer, and B. A. Foss. State and output feedback nonlinear model predictive control: An overview. *European Journal of Control*, 9:190–206, 2003.

[61] R. Fletcher. *Practical methods of optimization*. Wiley, New York, 1987.

[62] D. Fraser and J. Potter. The optimum linear smoother as a combination of two optimum linear filters. *IEEE Transactions on Automatic Control*, 14:387–390, 1969.

[63] D. Mignone G. Ferrari-Trecate and M. Morari. Moving horizon estimation for hybrid systems. *IEEE Trans. Autom. Control*, 47(10):1663–1676, 2002.

[64] A. Vande Wouwer G. Goffaux. Design of a robust nonlinear receding-horizon observer - application to a biological system. In *IFAC World Congress 2008 (IFAC08)*, Seoul, Korea, 2008.

[65] H. Genceli and M. Nikolaou. New approach to constrained predictive control with simultaneous model identification. *Journal of Aiche*, 42:2857–2868, 1996.

[66] A. George and J.W.H. Liu. Householder reflections versus givens rotations in sparse orthogonal decomposition. Technical Report CS-84-42, University of Waterloo, Canada and York University, Canada, 1963.

[67] R. Gesthuisen and S. Engell. Determination of the mass transport in the polycondensation of polyethyleneterephtalate by nonlinear estimation techniques. In *Proceedings of the International Symposium on Dynamics and Control of Process Systems (DYCOPS)*, Corfu, Greece, 1998.

[68] P. E. Gill, W. Murray, and M. H. Wright. *Numerical Linear Algebra and Optimization*, volume 2. Addison Wesley Publishing Company, Amsterdam, The Netherlands, 1991.

[69] P.E. Gill, G.H. Golub, W. Murray, and M.A. Saunders. Methods for modifying matrix factorizations. *Math. Comp.*, 28:505–535, 1974.

[70] P.E. Gill, W. Murray, M.A. Saunders, and M.H. Wright. A Schur-complement method for sparse quadratic programming. In *Reliable Numerical Computation*, pages 113–138. Oxford University Press, 1990.

[71] S. Gillijns. *Kalman filtering techniques for system inversion and data assimilation*. PhD thesis, Katholieke Universiteit Leuven, 2007.

[72] S. Gillijns and B. De Moor. Unbiased minimum-variance input and state estimation for linear discrete-time systems. *Automatica*, 435:111–116, Jan 2007.

[73] S. Gillijns, N. Haverbeke, and De Moor B. Information, covariance and square-root filtering in the presence of unknown inputs. In *Proceedings of the European Control Conference (ECC 2007)*, pages 2213–2217, Kos, Greece, 2007.

[74] W. Givens. Computation of plane unitary rotations transforming a general matrix to triangular form. *SIAM Journal of Appl. Math*, 6:26–50, 1958.

[75] T. Glad and H. Jonson. A method for state and control constrained linear quadratic control problems. In *Proceedings of IFAC World Congress*, pages 1583 – 1587, 1984.

[76] D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27:1–33, 1983.

[77] Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[78] G. C. Goodwin, M. M. Seron, and J.A. De Doná. *Constrained control and estimation: an optimisation approach*. Springer, 2005.

[79] G.C. Goodwin, J.A. De Doná, M.M. Seron, and X.W. Zhuo. Lagrangian duality between constrained estimation and control. *Automatica*, 41:935–944, 2005.

[80] P.J. Goulart, E.C. Kerrigan, and D. Ralph. Efficient robust optimization for robust control with constraints. *Mathematical Programming*, 114:115–147, 2008.

[81] M. Grant, S. Boyd, and Y. Ye. CVX: Matlab software for disciplined convex programming, 2006. Available at http://www. stanford. edu/ boyd/cvx.

[82] M. Grant, S. Boyd, and Y. Ye. CVX usersǵuide, 2006. http://www. stanford. edu/ boyd/cvx /cvx usrguide .pdf.

[83] M. S. Grewal and A. P. Andrews. *Kalman Filtering : Theory and Practice Using MATLAB*. Wiley-Interscience, 2 edition, 2001.

[84] A. Guitton and W.W. Symes. Robust and stable velocity analysis using the huber function. Technical Report Report 100, Stanford Exploration Project, 1999.

[85] A. Guitton and W.W. Symes. Robust inversion of seismic data using the huber norm. *Geophysics*, (4):1310–1319, 2003.

[86] F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, and W.A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley and Sons, New York, 1986.

[87] A. Hansson. A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE Transactions on Automatic Control*, Jan 2000.

[88] A. Hansson and S. Boyd. Robust optimal control of linear discrete time systems using primal-dual interior-point methods. In *Proceedings of the American Control Conference*, Jan 1998.

[89] E.L. Haseltine and J.B. Rawlings. Critical evaluation of extended kalman filtering and moving horizon estimation. *Ind. Eng. Chem. Res.*, 44(8):2451–2460, Apr 2005.

[90] N. Haverbeke, M. Diehl, and B. De Moor. A structure exploiting interior-point method for moving horizon estimation. In *Proceedings of the 48th IEEE Conference on Decision and Control Conference*, pages 1273–1278, Shanghai, China, 2009.

[91] N. Haverbeke, M. Diehl, and B. De Moor. Moving horizon state and input estimation as a convex cardinality problem. Technical Report TR-11-70, Katholieke Universiteit Leuven, 2011.

[92] N. Haverbeke, M. Diehl, and B. De Moor. Robust moving horizon estimation using the Huber function. Technical Report TR-11-69, Katholieke Universiteit Leuven, 2011.

[93] N. Haverbeke, M. Diehl, and B. De Moor. A Schur complement active-set method for fast moving horizon estimation. Technical Report TR-11-68, Katholieke Universiteit Leuven, 2011.

[94] N. Haverbeke, T. Van Herpe, M. Diehl, G. Van den Berghe, and B. De Moor. Nonlinear model predictive control with moving horizon state and disturbance estimation - application to the normalization of blood glucose in the critically ill. In *IFAC World Congress 2008 (IFAC08)*, pages 9069–9074, Seoul, Korea, 2008.

[95] J.D. Hedengren, K.V. Allsford, and J. Ramlal. Moving horizon estimation and control for an industrial gas phase polymerization reactor. In *Proceedings of the American Control Conference (ACC)*, New York, 2007.

[96] P. Heggernes and P. Matstoms. Finding good column orderings for sparse qr factorization. Technical report, In Second SIAM Conference on Sparse Matrices, 1996.

[97] K.A. Hoo, K.J. Tvarlapatib, M.J. Piovosoc, and R. Hajared. A method of robust multivariate outlier replacement. *Computers and Chemical Engineering*, 26:17–39, 2002.

[98] A.S. Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM (JACM)*, 5(4):339–342, 1958.

[99] B. Houska, H.J. Ferreau, and M. Diehl. ACADO toolkit - an open-source framework for automatic control and dynamic optimization. *Optimal Control Methods and Application*, page 57, 2010.

[100] M. Hovd and R. Bitmead. Interaction between control and state estimation in nonlinear MPC. In *Proceedings of the 7th International Symposium on Dynamics and Control of Process Systems*, Boston, MA, July 2004.

[101] R. Hovorka, V. Canonico, L.J. Chassin, U. Haueter, M. Massi-Benedetti, M. Orsini Federici, T.R. Pieber, H.C. Schaller, L. Schaupp, T. Vering, and M.E. Wilinska. Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes. *Physiol Meas*, 25(4):905–920, Aug 2004.

[102] P.J. Huber. Robust regression: asymptotics, conjectures, and monte carlo. *Ann. Statist.*, 1:799–821, 1973.

[103] P.J. Huber. *Robust Statistics*. John Wiley and Sons Inc., Hoboken, NJ, 1981.

[104] R.A. Jabr. Power system huber m-estimation with equality and inequality constraints. *Electric Power Systems Research*, 74:239–246, 2005.

[105] S.S. Jang, B. Joseph, and H. Mukai. Comparison of two approaches to on-line parameter and state estimation of nonlinear systems. *Ind. Eng. Chem. Proc. Des. Dev.*, 25:809–814, 1986.

[106] A.H. Jazwinski. Limited memory optimal filtering. *IEEE Transactions on Automatic Control*, 13:558–563, 1968.

[107] A.H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, New York, 1970.

[108] J.B. Jorgensen, J.B. Rawlings, and S.B. Jorgensen. Numerical methods for large-scale moving horizon estimation and control. In *Proceedings of Int. Symposium on Dynamics and Control Process Systems (DYCOPS)*, 2004.

[109] S.J. Julier. The spherical simplex unscented transformation. In *Proceedings of the American Control Conference*, pages 2430 – 2434, Denver, Colorado, June 2003.

[110] S.J. Julier and J.K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *SPIE AeroSense Symposium*, Orlando, FL, April 1997.

[111] T. Kailath. A view of three decades of linear filtering theory. *IEEE Transactions on Information Theory*, 20:146–181, 1974.

[112] R.E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, 82:35–45, 1960.

[113] R.E. Kalman and R.S Bucy. New results in linear filtering and prediction theory. *Transactions of the ASME–Journal of Basic Engineering*, 83:95–108, 1960.

[114] S. Kanev and M. Verhaegen. Two-stage kalman filtering via structured square-root. *Communications in information and systems*, 5(2):143–168, 2005.

[115] V.L. Klee and G.J. Minty. How good is the simplex method? In O. Shisha, editor, *Inequalities III*, pages 159–175. Academic Press, 1972.

[116] D.L. Kleinman. On an iterative technique for riccati equation computations. *IEEE Transactions on Automatic Control*, 1968.

[117] T. Kraus. Real-time state and parameter estimation for nmpc-based feedback control with application to the tennessee eastman benchmark process. Master's thesis, University of Heidelberg, Germany, 2000.

[118] T. Kraus, P. Kühl, L. Wirsching, and H.G. Bock. State and Parameter Estimation on Moving Horizons with Application to the Tennessee Eastman Benchmark Process. *Journal of Robotics and Autonomous Systems*, 2008.

[119] P. Kühl, Diehl. M., T. Kraus, J. Schlöder, and H.G. Bock. A real-time algorithm for moving horizon state and parameter estimation. *Computers and Chemical Engineering*, 35(1):71–83, 2011.

[120] H.J. Kushner. On the differential equations satisfied by conditional probability densities of Markov processses. *SIAM Journal of Control*, 2(106), 1964.

[121] W.H. Kwon and S. Han. *Receding Horizon Control*. Springer, 2005.

[122] E. Kyriakides, S. Suryanarayanan, and G.T. Heydt. State estimation in power engineering using the huber robust regression technique. *IEEE Transactions on Power Systems*, 20:1183–1184, 2005.

[123] M. Heinkenschloss L. T. Biegler, O. Ghattas and B. van Bloemen Waanders. Large-scale pde-constrained optimization: An introduction. In M. Heinkenschloss L. T. Biegler, O. Ghattas and B. van Bloemen Waanders, editors, *Large-Scale PDE-Constrained Optimization*, volume 30 of *Lecture Notes in Computational Science and Engineering*, pages 5–15. Springer-Verlag, Berlin, Heidelberg, New York, 2003.

[124] D.B. Leineweber. The theory of MUSCOD in a nutshell. IWR-Preprint 96-19, Universität Heidelberg, 1996.

[125] M.J. Liebman, T.F. Edgar, and L.S. Lasdon. Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques. *Comp. Chem. Eng.*, 16:963, 1992.

[126] M. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152:341–365, 2006.

[127] D.G. Luenberger. Observing the state of a linear system. *IEEE Trans. Mil. Elec.*, 8:74–80, 1964.

[128] J. Maciejowski. *Predictive Control with Constraints*. Pearson Education (Prentice-Hall imprint), 2001.

[129] P. Matstoms. Sparse linear least squares problems in optimization. *Computational Optimization and Applications*, 7:89–110, 1997.

[130] W. Mauntz, M. Diehl, and S. Engell. Moving horizon estimation and optimal excitation in temperature oscillation calorimetry. In *International Symposium on Dynamics and Control of Process Systems (DYCOPS)*, volume 2, pages 57–62, 2007. Cancun/Mexico.

[131] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[132] J. Meditch. A survey of data smoothing for linear and nonlinear dynamic systems. *Automatica*, 9:151–162, 1973.

[133] C. Müller, X.W. Zhuo, and J.A. De Doná. Duality and symmetry in constrained estimation and control problems. *Automatica*, 42:2183–2188, 2006.

[134] K.R. Muske, J.B. Rawlings, and J.H. Lee. Receding horizon recursive state estimation. In *American Control Conference*, pages 900–904, 1993.

[135] G.O. Mutambara. *Decentralized estimation and control for multi-sensor systems*. CRC press, 1998.

[136] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[137] B. J. Odelson, M. R. Rajamani, and J. B. Rawlings. A new autocovariance least-squares method for estimating noise covariances. *Automatica*, 42:303–308, 2006.

[138] Henrik Ohlsson, Fredrik Gustafsson, Lennart Ljung, and Stephen Boyd. State smoothing by sum-of-norms regularization. In *CDC'10*, pages 2880–2885, 2010.

[139] D.V. Ouellette. Schur complements and statistics. *Linear Algebra Appl*, 36:187–295, 1981.

[140] W. Murray P. E. Gill and M. A. Saunders. User guide for sqopt 7: Software for large-scale linear and quadratic programming. Technical Report NA 051, University of California, San Diego, 2005.

[141] R.S. Parker, F.J. 3rd Doyle, and N.A. Peppas. A model-based algorithm for blood glucose control in type I diabetic patients. *IEEE Trans Biomed Eng*, 46(2):148–157, Feb 1999.

[142] R.S. Parker, J.H. Ward, F.J. 3rd Doyle, and N.A. Peppas. Robust discrete H1 glucose control in diabetes using a physiological model. *AIChE Journal*, 46(12):2537–2545, 2000.

[143] L.S. Pontryagin, V.G. Boltyanski, R.V. Gamkrelidze, and E.F. Miscenko. *The mathematical theory of optimal processes*. Wiley, 1962.

[144] J.E. Potter and R.G. Stern. Statistical filtering of space navigation measurements. In *Proceedings of AIAA Guidance and Control Conference*, 1963.

[145] S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[146] C.V. Rao. *Moving horizon strategies for the constrained monitoring and control of nonlinear discrete-time systems*. PhD thesis, University of Wisconsin, Madison, 2000.

[147] C.V. Rao and J.B. Rawlings. Constrained process monitoring: moving-horizon approach. *AIChE Journal*, 48:97–109, 2002.

[148] C.V. Rao, J.B. Rawlings, and J.H. Lee. Constrained linear state estimation - a moving horizon approach. *Automatica*, 37(10):1619–1628, February 2001.

[149] C.V. Rao, J.B. Rawlings, and D. Mayne. Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations. *IEEE Transactions on Automatic Control*, 48:246–258, 2003.

[150] 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*, 99:723–757, 1998.

[151] H.E. Rauch, F. Tung, and C.T. Striebel. Maximum likelihood estimates of linear dynamic systems. *J. Amer. Inst. Aeronautics and Astronautics*, 3:1445–1450, 1965.

[152] J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2009.

[153] J.B. Rawlings and M.R. Rajamani. A hybrid approach for state estimation: Combining moving horizon estimation and particle filtering. In *Sandia CSRI Workshop – Large-Scale Inverse Problems and Quantification of Uncertainty*, Santa Fe, New Mexico, Spetember 2007.

[154] W.J.J. Rey. *Introduction to robust and quasi-robust statistical methods*. Springer, Berlin, Heidelberg, 1983.

[155] D.G. Robertson, J.H. Lee, and J.B. Rawlings. A moving horizon-based approach for least-squares estimation. *AIChE*, 42:2209–2224, 1996.

[156] J.A. Rossiter. *Model Based Predictive Control*. CRC Press, 2003.

[157] L.P. Russo and R.E. Young. Moving horizon state estimation applied to an industrial polymerization process. In *Proceedings of the American Control Conferenec (ACC)*, San Diego, California, 1999.

[158] S. Boyd S.-J. Kim, K. Koh and D. Gorinevsky. $l_1$ trend filtering. *SIAM Review, problems and techniques section*, 51(2):339–360, 2009.

[159] K. Schittkowski. *More test examples for nonlinear programming codes. Lecture notes in economics and mathematical systems (Vol. 282)*. Springer-Verlag, Berlin, 1987.

[160] J.P. Schlöder. *Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung*, volume 187 of *Bonner Mathematische Schriften*. Universität Bonn, Bonn, 1988.

[161] C. Schmid and L.T. Biegler. Quadratic programming methods for reduced hessian SQP. *Comp. Chem. Eng.*, 18:817, 1994.

[162] F.C. Schweppe. Algorithms for estimating a re-entry body's position, velocity and ballistic coefficient in real-time or from post-flight analysis. Technical Report ESD-TDR-64-583, M.I.T. Lincoln Lab Lexington, 1964.

[163] M. Seeger. Low rank updates for the Cholesky decomposition. Technical report, Department of EECS, UC Berkeley, 2008.

[164] D. Simon. *Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches*. John Wiley and Sons, 2006.

[165] Y. Song and J.W. Grizzle. The extended Kalman filter as a local asymptotic observer for discrete-time nonlinear systems. *Journal of Mathematical Systems, Estimation and Control*, 5:59–78, 1995.

[166] H.W. Sorenson. Least-squares estimation: from Gauss to Kalman. *IEEE Spectrum*, 7(7):63–68, July 1970.

[167] M.C. Steinbach. A structured interior point sqp method for nonlinear optimal control problems. In *Computational Optimal Control, volume 115 of Int. Series Numer. Math*, pages 213–222. Verlag, 1994.

[168] B.L. Stevens and F.L. Lewis. *Aircraft Control and Simulation*. John Wiley and Sons, New York, 1992.

[169] D. Sui, L. Feng, and M. Hovd. Robust explicit moving horizon control and estimation: A batch polymerization case study. *Modeling, Identification and Control*, 30:17–25, 2009.

[170] P. Swerling. Modern state estimation from the viewpoint of the method of leas squares. *IEEE Transactions on Automatic Control*, 16(6):707–719, 1971.

[171] R. Findeisen T. Raff, C. Ebenbauer and F. AllgÃ¶wer. Remarks on moving horizon state estimation with guaranteed convergence. In K. Graichen T. Meurer and E.D. Gilles, editors, *Control and Observer Design for Nonlinear Finite and Infinite Dimensional Systems*, pages 67–80. Springer Lecture Notes in Control and Information Sciences, Springer Verlag, 2005.

[172] M.J. Tenny. *Computational strategies for nonlinear model predictive control*. PhD thesis, University of Wisconsin, Madison, 2002.

[173] M.J. Tenny and J.B. Rawlings. Efficient Moving Horizon Estimation and Nonlinear Model Predictive Control. In *Proceedings of the American Control Conference*, 2002.

[174] N.A. Thacker and A.J. Lacey. Tutorial: The likelihood interpretation of the Kalman filter. Technical report, TINA Memos: Advanced Applied Statistics, 1996.

[175] Y.A. Thomas. Linear quadratic optimal estimation and control with receding horizon. *Electronic Letters*, 11:19–21, 1975.

[176] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, 58(1):267–288, 1996.

[177] I.B. Tjoa and L.T. Biegler. Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *I&EC Research*, 30:376, 1991.

[178] E. Todorov. *Bayesian Brain: Probabilistic Approaches to Neural Coding*, chapter 12: Optimal control theory, pages 269–298. MIT Press, 2006.

[179] G. Van den Berghe, A. Wilmer, G. Hermans, W. Meersseman, P.J. Wouters, I. Milants, E. Van Wijngaerden, H. Bobbaers, and R. Bouillon. Intensive insulin therapy in the medical ICU. *N Engl J Med*, 354(5):449–461, Feb 2006.

[180] G. Van den Berghe, P. Wouters, R. Bouillon, F. Weekers, C. Verwaest, F., M. Schetz, D. Vlasselaers, P. Ferdinande, and P. Lauwers. Outcome benefit of intensive insulin therapy in the critically ill: Insulin dose versus glycemic control. *Crit Care Med*, 31(2):359–366, Feb 2003. Clinical Trial.

[181] G. Van den Berghe, P. Wouters, F. Weekers, C. Verwaest, F. Bruyninckx, M. Schetz, D. Vlasselaers, P. Ferdinande, P. Lauwers, and R. Bouillon. Intensive insulin therapy in the critically ill patients. *N Engl J Med*, 345(19):1359–1367, Nov 2001. Clinical Trial.

[182] T. Van Herpe. *Blood Glucose Control in Critically Ill Patients: Design of Assessment Procedures and a Control System*. PhD thesis, Katholieke Universiteit Leuven, 2008.

[183] T. Van Herpe, M. Espinoza, N. Haverbeke, B. De Moor, and G. Van den Berghe. Glycemia prediction in critically ill patients using an adaptive modeling approach. *Journal of Diabetes Science and Technology*, 1:348–356, May 2007.

[184] T. Van Herpe, M. Espinoza, B. Pluymers, I. Goethals, P. Wouters, G. Van den Berghe, and B. De Moor. An adaptive inputoutput modeling approach for predicting the glycemia of critically ill patients. *Physiol Meas*, 27:1057–1069, Sep 2006.

[185] T. Van Herpe, N. Haverbeke, B. Pluymers, G. Van den Berghe, and B. De Moor. The application of model predictive control to normalize glycemia of critically ill patients. In *Proc. of the European Control Conference 2007 (ECC 2007), Kos, Greece*, pages 3116–3123, Jul 2007.

[186] T. Van Herpe, N. Haverbeke, G. Van den Berghe, and B. De Moor. Prediction performance comparison between three intensive care unit glucose models. In *Proc. of the 7th IFAC Symposium on Modelling and Control in Biomedical Systems (MCBMS'09), Aalborg, Denmark*, pages 7–12, Aug 2009.

[187] C. Van Loan. On the method of weighting for equality constrained least squares problems. *SIAM Journal of Numer. Analysis*, 22:851–864, 1985.

[188] L. Vandenberghe, S. Boyd, and M. Nouralishahi. Robust Linear Programming and Optimal Control. In *Proceedings of IFAC World Congress*, pages 271–276, 2002.

[189] M. Verhaegen and P. Van Dooren. Numerical Aspects of Different Kalman Filter Implementations. *IEEE Transactions on Automatic Control*, 31:907 – 917, 1986.

[190] M. Verhaegen and V. Verduld. *Filtering and Identification: A Least Squares Approach*. Cambridge University Press, 2007.

[191] A. Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*. PhD thesis, Carnegie Mellon University, 2002.

[192] Y. Wang and S. Boyd. Fast Model Predictive Control Using Online Optimization. In *Proceedings of IFAC World Congress*, pages 6974 – 6997, 2008.

[193] Y-G. Wang, X. Lin, M. Zhu, and Z. Bai. Robust estimation using the Huber function with a data dependent tuning constant. *Journal of Computational and Graphical Statistics*, 16(2):468 – 481, 2007.

[194] D. S. Weller, J. R. Polimeni, L. Grady, L.L. Wald, E. Adalsteinsson, and V. K. Goyal. Evaluating sparsity penalty functions for combined compressed sensing and parallel mri. In *IEEE Int. Symp. on Biomedical Imaging*, pages 1–4, 2011.

[195] J.C. Willems. Dissipative dynamical systems - part i: General theory, part ii: Linear systems with quadratic supply rates. *Archive for Rational Mechanics and Analysis*, 45:321–393, 1972.

[196] J.C. Willems. Deterministic least squares filtering. *Journal of Econometrics*, 118:341–373, 2004.

[197] W.M. Wonham. optimal stochastic control. *Automatica*, 5:113–118, 1969.

[198] S.J. Wright. *Primal-dual interior-point methods*. SIAM publications, Philadelphia, 1997.

[199] V.M. Zavala. *Computational Strategies for the Optimal Operation of Large-Scale Chemical Processes*. PhD thesis, Carnegie Mellon University, 2008.

[200] V.M. Zavala. Stability analysis of an approximate scheme for moving horizon estimation. *Submitted to Comp. Chem. Eng.*, 2009.

[201] V.M. Zavala, C.D. Laird, and L.T. Biegler. A fast computational framework for large-scale moving horizon estimation. In *Proceedings of the 8th International Symposium on Dynamics and Control of Process Systems (DYCOPS)*, pages 6974 – 6997, Cancun, Mexico, 2007.

[202] V.M. Zavala, C.D. Laird, and L.T. Biegler. A fast moving horizon estimation algorithm based on nonlinear programming sensitivity. *Journal of Process Control*, 18:876–884, 2008.

[203] F. Zhang. *The Schur Complement and Its Applications*. Springer, 2005.

[204] Y. Zhang, A. d'Aspremont, and L. El Ghaoui. Sparse pca: Convex relaxations, algorithms and applications. *ArXiv e-prints*, November 2010.

[205] Z. Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image Vision Computing*, 15:59–76, 1997.

# List of Publications

## *Articles in International Journals*

N. Haverbeke, M. Diehl, B. De Moor. "A Schur complement active-set method for fast moving horizon estimation". *Internal Report 11-68, ESAT-SISTA, K.U.Leuven, Leuven, 2011*. Submitted for review.

T. Barjas Blanco, P. Willems, P. Po-Kuan Chiang, N. Haverbeke, J. Berlamont, B. De Moor. "Flood Regulation using Nonlinear Model Predictive Control". *Control Engineering Practice*, vol. 18, 2010, pp. 1147-1157.

T. Van Herpe, M. Espinoza, N. Haverbeke, B. De Moor, G. Van den Berghe. "Glycemia prediction in critically ill patients using an adaptive modeling approach". *Journal of Diabetes Science and Technology*, vol. 1, no. 3, May 2007, pp. 348-356.

## *Articles in the Proceedings of International Conferences*

N. Haverbeke, M. Diehl, B. De Moor. "A structure exploiting interior-point method for moving horizon estimation". In *Proceedings of the 48th IEEE Conference on Decision and Control Conference (CDC09)*, pages 1273-1278. Shanghai, China, Dec. 2009.

T. Van Herpe, N. Haverbeke, G. Van den Berghe, B. De Moor. "Prediction Performance Comparison between three Intensive Care Unit Glucose Models". In *Proceedings of the 7th IFAC Symposium on Modelling and Control in Biomedical Systems (MCBMS'09)*, pages 7-12. Aalborg, Denmark, Aug. 2009.

N. Haverbeke, T. Van Herpe, M. Diehl, G. Van den Berghe, B. De Moor. "Nonlinear model predictive control with moving horizon state and disturbance estimation - Application to the normalization of blood glucose in the critically ill". In *Proceedings of the IFAC World Congress 2008 (IFAC08)*, pages 9069-9074. Seoul, Korea, Jul. 2008.

T. Van Herpe, N. Haverbeke, M. Espinoza, G. Van den Berghe, B. De Moor. "Adaptive modeling for control of glycemia in critically ill patients". In *Proceedings of the 10th International IFAC Symposium on Computer Applications in Biotechnology (Vol. 1) (CAB)*, pages. 159-164. Cancun, Mexico, Jun. 2007.

T. Van Herpe, N. Haverbeke, B. Pluymers, G. Van den Berghe, B. De Moor. "The application of Model Predictive Control to normalize glycemia of critically ill patients". In *Proceedings of the European Control Conference 2007 (ECC 2007)*, pages 3116-3123. Kos, Greece, 2007.

S. Gillijns, N. Haverbeke, B. De Moor. "Information, Covariance and Square-Root Filtering in the Presence of Unknown Inputs". In *Proceedings of the European Control Conference (ECC 2007)*, pages 2213-2217. Kos, Greece, 2007.

## Book Chapters

M. Diehl and H. J. Ferreau and N. Haverbeke. "Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation". Pages 391-417. In Chapter 32 of *Nonlinear model predictive control.* Volume 384 of *Lecture Notes in Control and Information Sciences*. Springer, 2009.

## Internal Reports

N. Haverbeke, M. Diehl, B. De Moor. "Robust Moving Horizon Estimation using the Huber function". In *Internal Report 11-69, ESAT-SISTA, K.U.Leuven, Leuven, 2011*.

N. Haverbeke, M. Diehl, B. De Moor. "Moving Horizon State and Input Estimation as a Convex Cardinality Problem". In *Internal Report 11-70, ESAT-SISTA, K.U.Leuven, Leuven, 2011*.

# Curriculum Vitae

Niels Haverbeke was born in Neerpelt, Belgium on November 14, 1981. He received his Master of Science in Electrical Engineering from the Katholieke Universiteit Leuven (University of Leuven), Belgium in 2005. In January 2006, he obtained a scholarship from the Agency for Innovation by Science and Technology in Flanders (IWT), to pursue his Ph.D. degree in the research division SCD-Sista of the Department of Electrical Engineering of the K.U.Leuven, under the supervision of Prof. Bart De Moor and Prof. Moritz Diehl. In 2008, he was a visiting researcher in the Information Systems Lab at Stanford University under supervision of Prof. Stephen Boyd, supported by a travel grant of FWO-Vlaanderen. Since March 2010, he is a researcher at Flanders' DRIVE, the research institute for the Flemish automotive industry.