Linköping studies in science and technology. Dissertations.
No. 1676

Divide and Conquer: Distributed Optimization and Robustness Analysis

Sina Khoshfetrat Pakazad

Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden
Linköping 2015

Cover illustration: A design approach for distributed optimization algorithms, used in papers C and F. It states that given the sparsity graph of a coupled optimization problem, defined in Section 5.1, we can group its variables and its constituent subproblems so that the coupling structure can be represented using a tree. We can then solve the problem distributedly using message passing, which utilizes the tree as its computational graph.

Linköping studies in science and technology. Dissertations.
No. 1676

**Divide and Conquer:
Distributed Optimization and Robustness Analysis**

Sina Khoshfetrat Pakazad

*sina.kh.pa@isy.liu.se
www.control.isy.liu.se
Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping
Sweden*

ISBN 978-91-7519-050-1

ISSN 0345-7524

Copyright © 2015 Sina Khoshfetrat Pakazad

Printed by LiU-Tryck, Linköping, Sweden 2015

To my parents

Abstract

As control of large-scale complex systems has become more and more prevalent within control, so has the need for analyzing such controlled systems. This is particularly due to the fact that many of the control design approaches tend to neglect intricacies in such systems, e.g., uncertainties, time delays, nonlinearities, so as to simplify the design procedure.

Robustness analysis techniques allow us to assess the effect of such neglected intricacies on performance and stability. Performing robustness analysis commonly requires solving an optimization problem. However, the number of variables of this optimization problem, and hence the computational time, scales badly with the dimension of the system. This limits our ability to analyze large-scale complex systems in a centralized manner. In addition, certain structural constraints, such as privacy requirements or geographical separation, can prevent us from even forming the analysis problem in a centralized manner.

In this thesis, we address these issues by exploiting structures that are common in large-scale systems and/or their corresponding analysis problems. This enables us to reduce the computational cost of solving these problems both in a centralized and distributed manner. In order to facilitate distributed solutions, we employ or design tailored distributed optimization techniques. Particularly, we propose three distributed optimization algorithms for solving the analysis problem, which provide superior convergence and/or computational properties over existing algorithms. Furthermore, these algorithms can also be used for solving general loosely coupled optimization problems that appear in a variety of fields ranging from control, estimation and communication systems to supply chain management and economics.

Populärvetenskaplig sammanfattning

Regulatorer för styrning av system är ofta designade med hjälp av en beskrivning, typiskt en matematisk modell, av det man vill styra. Dessa modeller är oftast endast approximativa beskrivningar av det verkliga systemet och därför behäftade med osäkerheter. Detta medför att det beteende man förväntar sig av ett reglerat system baserat på den matematiska modellen inte alltid stämmer överens med det observerade beteendet. Att analysera hur stor denna avvikelse kan bli kallas robusthetsanalys. Detta är ett väl studerat område för mindre och medelstora system. Dock är kunskaperna begränsade för riktigt stora system, speciellt om man inte vill göra en alltför konservativ analys. Exempel på stora system är kraftnät och flygplan. Kraftnät är stora på grund av det stora antalet komponenter som måste modelleras. Flygplan är stora på grund av att man för större flygplan också måste modellera vingarnas flexibilitet. I den här avhandlingen diskuteras hur man kan analysera riktigt stora system. Lösningen är att särkoppla systemen i mindre delsystem, vars kopplingar beskrivs t.ex. med bivillkor.

Acknowledgments

Finally I can see the light at the end of the tunnel and this time it seems it is not a train passing through (or is it?). This journey would not have been accomplished without the help and persistent guidance of my supervisor Prof. Anders Hansson. I am sincerely grateful for your remarkable patience, deep insights and diligent supervision. I am also grateful for additional and essential support I received from my co-supervisors Dr. Anders Helmersson and Prof. Torkel Glad. Moreover, I would like to thank Dr. Martin S. Andersen and Prof. Anders Rantzer for their constructive and inspiring comments and indispensable contributions. For the financial support, I would like to thank the European Commission under contract number AST5-CT-2006-030768-COFCLUO and Swedish government under the ELLIIT project, the strategic area for ICT research.

I have been very fortunate to have been given a chance to do my PhD at the impressive automatic control group in Linköping University. The group was established by Prof. Lennart Ljung and is now continuing strong under the passionate leadership of Prof. Svante Gunnarsson. Thank you both for granting me the opportunity for being part of this group. The created friendly environment in the group fosters collaboration among members with different backgrounds and research topics. I have been lucky to have been a part of such collaborations owing to Prof. Lennart Ljung, Dr. Tianshi Chen and Dr. Henrik Ohlsson. Thank you for the great experience.

The quality of the thesis has improved significantly thanks to comments from an army of proof readers. Thank you Dr. Daniel Petersson, Lic. Niklas Wahlström and Dr. Andre Carvalho Bittencourt for being brave enough to be the pioneer proof readers, and thank you Dr. Gustaf Hendeby, Lic. Jonas Linder, Isak Nielsen, Dr. Zoran Sjanic, Christian Andersson Naesseth and Hanna Nyqvist for your constructive comments. The thesis would have looked much worse without our great $\LaTeX$ template thanks to Dr. Gustaf Hendeby and Dr. Henrik Tidefelt. Also special thanks goes to Gustaf for his constant help with my never ending $\LaTeX$ related issues. I also would like to extend my gratitude to Ninna Stensgård, Åsa Karmelind and Ulla Salaneck that have made the PhD studies and the group function much smoother.

To accomplish the PhD studies, is like a long roller coaster ride with high climbs and dives, sometimes with spiky seats and broken seat belts ☺! I have been truly lucky to have shared this ride with some of the most remarkable, loving and adventurous people. Dr. Andre Carvalho Bittencourt, always saying the right thing at the wrong time, Dr. Emre Özkan, one of the most considerate and caring people I know, Dr. Daniel Ankelhed, with the warmest personality and greatest taste in music, Dr. Fredrik Linsten, with the coolest temper and an amazing knack for best beers and whiskies (both tested in Brussels!) and Hanna Nyqvist, one of the best roommate-friends with an unquenchable thirst for cookies and candies in any form (it's just ... up). Thank you guys for the brotherly friendship and all the great memories and experiences.

The people I came to know during my PhD years were only my colleagues for a brief period and became my friends almost instantly. This made going to work much much more enjoyable, even when you had teaching at 8 on a Monday morning! One of the things that amazed me the most during these years, was the coherence of the group and how almost everyone was always up for different activities ranging from Hawaii-style summertime BBQs during Swedish winters, with Dr.-Ing. Carsten Fritsche enjoying comments regarding tenderness of steaks!, to watching movies based on my rare artistic criteria (some might use other adjectives!). Speaking of great taste in movies, thank you Dr. Jonas Callmer for introducing me to Frankenfish, probably the worst best worst movie of all time, that is including Sharknado I and II.

Lic. Jonas Linder, the king of the lakes of Dalsland, thank you for including me in the great canoeing experience and honoring the tradition of "whatever happens in Dalsland stays in Dalsland". Speaking of things that should be buried in the past, thank you Dr. Patrik Axelsson for documenting and remembering such memories and making sure that they resurface every now and then during our gatherings (of course Jonas also has an exceptional talent in that area). Also thank you Lic. Johan Dahlin for making sure that we never went hungry or thirsty during the Canoe trip. Thank you Lic. Ylva Jung and Lic. Manon Kok for the lussekatt baking ritual that led to the invention of accurate saffron measuring techniques and extending my definition of "maybe". The events always become more fascinating with Dr. Zoran Sjanic and Dr. Karl Granström (The Spex masters) with their exceptional ability to steer any conversation towards more intriguing directions (sometimes to their own surprise) with help from Lic. Marek Syldatak (the shoe and booze Mooster) and Clas Veibäck (with many great premium business ideas). Speaking of interesting conversations, I should also acknowledge, Lic. Niklas Wahlström, Lic. Daniel Simon, Lic. Roger Larsson, Lic. Michael Roth, George Mathai, Christian Andersson Naesseth and Dr. Martin Skoglund for all the fascinating exchanges during the fika sessions, ranging from south park and how jobs were taken to very deep socio-economic issues.

One of the great perks of doing a PhD is that you get to travel around the world for workshops and conferences. Thank you Dr. Soma Tayamon, Dr. Henrik Ohlsson, Dr. Christian Lyzell (with strongly held views towards sandy Christmas snowmen!) and Dr. Ragnar Wallin for being great travel companions, always enhancing the culinary experience and the activities during the trip.

Within my years in Sweden I have been privileged to have made very close friends outside of RT. Lic. Behbood Borghei (The Doog) and Farham (Farangul) thank you for your unfaltering friendship and companionship through all these years, and thank you Farzad, Amin, Shirin, Mahdis, Claudia and Thomas for your caring and loving bond and constant support through all the good and not so good times. I really cherish your friendship.

I have been truly blessed to have the most caring family in the worlds! Thank you mom and dad for being endless sources of inspiration and energy and always being there for me and supporting me all the way to this point. Also I am indebted to my brothers, Soheil and Saeed, for always encouraging and helping me through all my endeavors in my life. And thank you Negi for bringing so

much love and joy into my life. It made these final steps to the PhD much easier to take. I am looking forward to our future and adventures to come.

Finally, (in case I have missed someone) I would like to thank _____

add your name here

for _____.
add all the reasons and all the good times

Linköping, June 2015
Sina Khoshfetrat Pakazad

Contents

Notation	xix
----------	-----

I Background

1 Introduction	3
1.1 Contributions and Publications	4
1.2 Additional Publications	8
1.3 Thesis Outline	9
2 Convex Optimization Problems	11
2.1 Convex Sets and Functions	12
2.1.1 Convex Sets	12
2.1.2 Convex Functions	12
2.2 Convex Optimization Problems	13
2.2.1 Linear Programs	13
2.2.2 Quadratic Programs	14
2.2.3 Generalized Inequalities	14
2.2.4 Semidefinite Programs	15
2.3 Primal and Dual Formulations	15
2.4 Optimality Conditions	16
2.5 Equivalent Optimization Problems	17
2.5.1 Removing and Adding Equality Constraints	17
2.5.2 Introduction of Slack Variables	18
2.5.3 Reformulation of Feasible Set Description	18
3 Convex Optimization Methods	19
3.1 Proximal Point Algorithm	19
3.1.1 Monotonicity and Zeros of Monotone Operators	19
3.1.2 Proximal Point Algorithm and Convex Problems	20
3.2 Primal and Primal-dual Interior-point Methods	22
3.2.1 Primal Interior-point Methods	22
3.2.2 Primal-dual Interior-point Methods	24

4	Uncertain Systems and Robustness Analysis	29
4.1	Linear Systems	29
4.1.1	Continuous-time Systems	29
4.1.2	$\mathcal{H}_\infty$ and $\mathcal{H}_2$ Norms	30
4.2	Uncertain Systems	30
4.2.1	Structured Uncertainties and LFT Representation	31
4.2.2	Robust $\mathcal{H}_\infty$ and $\mathcal{H}_2$ Norms	32
4.2.3	Nominal and Robust Stability and Performance	32
4.3	Robust Stability Analysis of Uncertain Systems Using IQCs	33
5	Coupled Problems and Distributed Optimization	35
5.1	Coupled Optimization Problems and Structure Exploitation	35
5.2	Sparsity in SDPs	36
5.2.1	Chordal Graphs	37
5.2.2	Chordal Sparsity	38
5.2.3	Domain-space Decomposition	39
5.3	Distributed Optimization Algorithms	40
5.3.1	Distributed Algorithms Purely Based on Proximal Splitting	40
5.3.2	Distributed Solutions Based on Splitting in Primal and Primal-dual Methods Using Proximal Splitting	44
5.3.3	Distributed Solutions Based on Splitting in Primal and Primal-dual Methods Using Message Passing	47
6	Examples in Control and Estimation	51
6.1	Distributed Predictive Control of Platoons of Vehicles	51
6.1.1	Linear Quadratic Model Predictive Control	52
6.1.2	MPC for Platooning	53
6.2	Distributed Localization of Scattered Sensor Networks	56
6.2.1	A Localization Problem Over Sensor Networks	56
6.2.2	Localization of Scattered Sensor Networks	58
6.2.3	Decomposition and Convex Formulation of Localization of Scattered Sensor Networks	59
7	Conclusions and Future Work	63
7.1	Future Work	64
	Bibliography	65

II Publications

A	Distributed Solutions for Loosely Coupled Feasibility Problems Using Proximal Splitting Methods	73
1	Introduction	75
2	Decomposition and Convex Feasibility	78
2.1	Error Bounds and Bounded Linear Regularity	78
2.2	Projection Algorithms and Convex Feasibility Problems	79

2.3	Decomposition and Product Space Formulation	80
2.4	Convex Minimization Formulation	82
3	Proximity Operators and Proximal Splitting	83
3.1	Forward-backward Splitting	84
3.2	Splitting Using Alternating Linearization Methods	86
3.3	Douglas-Rachford Splitting	87
4	Distributed Solution	88
4.1	Proximal Splitting and Distributed Implementation	88
4.2	Distributed Implementation of von Neumann's and Dykstra's AP method	96
4.3	Local Convergence Tests	96
5	Convergence Rate	99
5.1	Feasible Problem	100
5.2	Infeasible Problem	101
6	Numerical Results	102
6.1	Flow Feasibility Problem	102
6.2	Semidefinite Feasibility Problems	108
7	Conclusions	111
A	An Algorithm for Random Generation of Connected Directed Graphs	112
	Bibliography	114

B A Distributed Primal-dual Interior-point Method for Loosely Coupled Problems Using ADMM

		117
1	Introduction	119
2	Loosely Coupled Problems	124
3	Primal-dual Interior-point Methods	125
3.1	Step Size Computations	128
4	A Distributed Primal-dual Interior-point Method For Solving Loosely Coupled Problems	129
4.1	Distributed Primal-dual Direction Computations	131
4.2	Distributed Computations of Perturbation Parameter, Step Size and Stopping Criterion	135
5	Primal-dual Inexact Interior-point Methods	138
6	A Distributed Primal-dual Inexact Interior-point Method for Solving Loosely Coupled Problems	140
6.1	Distributed Computations of Perturbation Parameter, Step Size and Stop Criterion	142
6.2	Distributed Primal-dual Inexact Interior-point Method	143
6.3	Distributed Convergence Result	144
7	Iterative Solvers for Saddle Point Systems	145
7.1	Uzawa's Method and Fixed Point Iterations	145
7.2	Other Iterative Methods	145
8	Improving Convergence Rate of ADMM	146
9	Numerical Experiment	146
9.1	Simulation Setup 1	147

9.2	Simulation Setup 2	162
10	Conclusion	162
A	Global Convergence of Distributed Inexact Primal-dual Interior-point Method	163
A.1	Break Down	163
A.2	Convergence Properties	169
B	ADMM, Fixed Point Iterations and Uzawa's Method	171
B.1	ADMM and Fixed Point Iterations	171
B.2	ADMM and Uzawa's Method	173
	Bibliography	175
C	Distributed Primal-dual Interior-point Methods for Solving Loosely Coupled Problems Using Message Passing	179
1	Introduction	181
2	Coupled Optimization Problems	185
2.1	Coupling and Sparsity Graphs	185
3	Chordal Graphs	186
3.1	Chordal Embedding and Its Cliques	187
3.2	Clique Trees	187
4	Optimization Over Clique Trees	188
4.1	Distributed Optimization Using Message Passing	188
4.2	Modifying the Generation of the Computational Graph	193
5	Primal-dual Interior-point Method	196
6	A Distributed Primal-dual Interior-point Method	199
6.1	Loosely Coupled Optimization Problems	199
6.2	Distributed Computation of Primal-dual Directions	200
6.3	Relations to Multi-frontal Factorization Techniques	212
6.4	Distributed Step Size Computation and Termination	214
6.5	Summary of the Algorithm and Its Computational Properties	217
7	Numerical Experiments	219
8	Conclusion	223
A	Proof of Theorem 1	223
B	Proof of Theorem 2	225
	Bibliography	226
D	Robust Stability Analysis of Sparsely Interconnected Uncertain Systems	229
1	Introduction	231
1.1	Outline	233
1.2	Notation	233
2	Robustness Analysis Using IQCs	234
2.1	Integral Quadratic Constraints	234
2.2	IQC Analysis	235
3	Robust Stability Analysis of Interconnected Uncertain Systems	235
3.1	Interconnected Uncertain Systems	235

3.2	Lumped Formulation	236
3.3	Sparse Formulation	237
4	Sparsity in Semidefinite Programs (SDPs)	238
5	Numerical Experiments	239
5.1	Chain of Uncertain Systems	240
5.2	Interconnection of Uncertain Systems Over Scale-free Network	241
6	Conclusions	244
A	Subsystems Generation for Numerical Experiments	244
	Bibliography	246
E	Distributed Robustness Analysis of Interconnected Uncertain Systems Using Chordal Decomposition	249
1	Introduction	251
2	Robust Stability Analysis of Uncertain Systems Using IQCs	254
3	Interconnected Systems: A Definition	255
3.1	Uncertain Interconnections	256
4	Robust Stability Analysis of Interconnected Uncertain Systems	257
5	Chordal Graphs and Sparsity in SDPs	258
5.1	Chordal Graphs	259
5.2	Chordal Sparsity in SDPs	260
6	Numerical Experiment	262
7	Conclusions	262
	Bibliography	264
F	Distributed Semidefinite Programming with Application to Large-scale System Analysis	267
1	Introduction	269
2	Coupled and Loosely Coupled SDPs	273
3	Primal-Dual Interior-point Methods for Solving SDPs	275
4	Tree Structure in Coupled Problems and Message Passing	281
5	Distributed Primal-dual Interior-point Methods for Coupled SDPs	283
5.1	Distributed Computation of Primal-dual Directions Using Message Passing	284
5.2	Distributed Step Size Computation and Termination Check	285
5.3	Summary of the Algorithm and Its Computational Properties	288
6	Chordal Sparsity and Domain-space Decomposition	290
6.1	Sparsity and Semidefinite Matrices	290
6.2	Domain-space Decomposition	291
7	Robustness Analysis of Interconnected Uncertain Systems	292
7.1	Robustness Analysis using IQCs	292
7.2	Robustness Analysis of Interconnected Uncertain Systems using IQCs	293
8	Numerical Experiments	295
9	Conclusions	297

Bibliography	299
G Robust Finite-frequency $\mathcal{H}_2$ Analysis of Uncertain Systems with Application to Flight Comfort Analysis	303
1 Introduction	305
1.1 Notation	306
2 Problem Formulation	307
3 Mathematical Preliminaries	309
3.1 Finite-frequency Observability Gramian	309
3.2 An Upper Bound on the Robust $\mathcal{H}_2$ Norm	310
4 Gramian-based Upper Bound on the Robust Finite-frequency $\mathcal{H}_2$ Norm	314
5 Frequency-gridding-based Upper Bound on the Robust Finite-frequency $\mathcal{H}_2$ Norm	315
6 Numerical Example	319
7 Application to Flight Comfort Analysis	322
8 Discussion and General Remarks	325
8.1 The Observability-Gramian-based Method	325
8.2 The Frequency-gridding-based Method	326
9 Conclusion	326
A Proof of Lemma 2	327
B Proof of Theorem 1	327
C Proof of Theorem 3	328
D Proof of Theorem 4	329
Bibliography	330

Notation

SYMBOLS, OPERATORS AND FUNCTIONS

Notation	Meaning
$\mathbb{N}$	Set of integer numbers
$\mathbb{R}$	Set of real numbers
$\mathbb{C}$	Set of complex numbers
$\mathbb{N}_n$	Set of integer numbers $\{1, \dots, n\}$
$\mathbb{R}^n$	Set of n -dimensional real vectors
$\mathbb{C}^n$	Set of n -dimensional complex vectors
$\mathbb{R}^{m \times n}$	Set of $m \times n$ real matrices
$\mathbb{C}^{m \times n}$	Set of $m \times n$ complex matrices
$\mathbb{S}^n$	Set of $n \times n$ symmetric matrices
$\mathbb{S}_+^n$	Set of $n \times n$ positive semidefinite matrices
$\mathbb{S}_{++}^n$	Set of $n \times n$ positive definite matrices
$\mathbb{R}_+^n$	Set of n -dimensional positive real vectors
$\in$	Belongs to
$\text{dom } f$	Domain of a function f
$\text{int } \mathcal{C}$	Interior of a set $\mathcal{C}$
A^T	Transpose of a matrix A
A^*	Conjugate transpose of a matrix A
$A \leq B$	For $A, B \in \mathbb{S}^n$, $A - B$ is negative semidefinite
$a \leq b$	For $a, b \in \mathbb{R}^n$, element-wise inequality
$A \subset B$	A is a strict subset of B
$A \subseteq B$	A is a subset of B
$A \setminus B$	Set difference
$\text{tr}(A)$	Trace of a matrix A
I_n	Identity matrix of order n
$\text{rank}(A)$	Rank of a matrix A
$\bar{\sigma}(A)$	Maximum singular value of a matrix A
$\text{col}(A)$	Column-space of a matrix A
$\text{diag}(x)$	Diagonal matrix with elements of x on the diagonal
$\ll$	Much smaller
$\mathbf{1}$	A vector with only ones as its elements
$\emptyset$	Empty set
$\min$	Minimum value of a function or set
$\max$	Maximum value of a function or set
argmin	Minimizing argument of a function

SYMBOLS, OPERATORS AND FUNCTIONS

Notation	Meaning
$\mathcal{I}_C$	Indicator function of a set C
$\inf$	Infimum value of a function
$\sup$	Supremum value of a function
$ J $	Number of elements in a set J
E_C	For $C \subseteq \mathbb{N}_n$, a 0–1 matrix that is obtained from I_n by removing all rows indexed by $\mathbb{N}_n \setminus C$.
X_{CC}	The matrix $E_C X E_C^T$
x_j	The vector $E_j x$
$A \bullet B$	$\text{tr}(A^T B)$
$C \times J$	Cartesian product between two sets C and J
$\langle x, y \rangle$	Inner product between two vectors x and y
$\cap, \cup$	Set union, intersection
(x, y)	Column vector with x and y stacked
$\ \cdot\ _\infty$	For vectors the infinity norm and for matrices the induced infinity norm
$\ \cdot\ , \ \cdot\ _2$	For vectors the two norm and for matrices the induced two norm
∇f	Gradient of a function f
$\nabla^2 f$	Hessian of a function f
prox_f	Proximity operator of a function f
$\ln$	Natural logarithm
$\mathcal{RH}_\infty^{m \times n}$	The set of real, rational $m \times n$ transfer function matrices with no poles in the closed right half plane
$\mathcal{L}_2^n$	The set of n -dimensional square integrable functions
$\sim$	Distributed according to
$\mathcal{N}(\mu, \Sigma)$	Multivariate Gaussian with mean μ and covariance Σ

ABBREVIATIONS

Abbreviation	Meaning
ADMM	Alternating direction method of multipliers
IQC	Integral quadratic constraint
KKT	Karush-Kuhn-Tucker
KYP	Kalman-Yakubovich-Popov
LMI	Linear matrix inequality
LP	Linear program
LTI	Linear time-invariant
MPC	Model predictive control
PDF	Probability density function
QCQP	Quadratically constrained quadratic program
QP	Quadratic program
SDP	Semidefinite program

Part I

Background

1

Introduction

Many control design methods are model driven, e.g., see Glad and Ljung (2000); Bequette (2003); Åström and Wittenmark (1990). As a result, stability and performance provided by controllers designed using such methods are affected by the quality of the models. One of the major issues with models is their uncertainty, and some of the model-based design methods take this uncertainty into consideration, e.g., see Zhou and Doyle (1998); Skogestad and Postlethwaite (2007); Doyle et al. (1989); Horowitz (1993). However, many of the model-based design methods, specially the ones employed in industry, neglect the model uncertainty to simplify the design procedure. Hence, it is important to address how model uncertainty tampers with the performance or stability of the controlled system. To be more precise, it is essential to check whether there is any chance for the controlled system, to lose stability or desired performance under any *admissible* uncertainty. This type of analysis is called robustness analysis and is extremely important in applications where the margin for errors is very small, e.g., flight control design.

Motivated by this, robustness analysis plays a central role in this thesis. Specifically, we consider the problem of robustness analysis of large-scale uncertain systems, which appear for instance, in applications involving discretized partial differential equations (PDEs) such as flexible structures, vehicle platooning, aircraft and satellite formation flight, (Swaroop and Hedrick, 1996; Rice and Verhaegen, 2009; Massioni and Verhaegen, 2009). Here we particularly focus on two examples of such systems, namely complex systems that include a large number of states and have large uncertainty dimensions, such as aircraft control systems, and large-scale interconnected uncertain systems. Robustness analysis of such systems pose different challenges. These can be purely computational or due to structural constraints in the application.

The computational challenges commonly stem from the fact that analyzing such systems, in many cases, requires solving a large optimization problem. Depending on the dimensions of the uncertain system, e.g., number of states and input-outputs, and uncertainty dimension, this optimization problem can be very difficult, time-consuming or even impossible to solve in a centralized manner. This is mainly due to insufficient memory or processing power of available centralized work stations or in general our limited capability to solve the centralized problem in a timely manner. Such challenges are commonly addressed by exploiting structure in the problem and devising efficient solvers that use the available computational and memory resources more wisely, e.g., see Wallin et al. (2009); Hansson and Vandenbergh (2000); Rice and Verhaegen (2009); Massioni and Verhaegen (2009).

Structural challenges, however, stem from our inability or unwillingness to form the centralized analysis problem. This can be due to complicating structural constraints or requirements in the application, e.g., physical separation or privacy requirements. For instance, large-scale uncertain systems can sometimes arise from an interconnection of several uncertain subsystems, the models of which are provided by different parties. Then possible reluctance of these parties to share their models with a centralized unit prohibits us from forming the analysis problem in a centralized manner, and as a consequence prevents us from solving the problem.

In order to address these challenges, we undertake a divide and conquer strategy. Particularly, by exploiting structure in the corresponding optimization problem, we break it down into smaller subproblems that are easier to solve, and devise algorithms that deal with the smaller subproblems for solving the original problem. To this end, we first reformulate and decompose the encountered analysis problems. Then we employ tailored distributed optimization algorithms for solving the decomposed problems. These algorithms enable us to solve the decomposed problems using a network of computational agents that collaborate and communicate with one another. Hence, distributed algorithms enable us to dispense the computational burden over the network, and moreover satisfy underlying structural requirements in the application, such as privacy.

1.1 Contributions and Publications

The contributions of this thesis can be divided into two categories. The first category concerns the development of distributed optimization algorithms for solving loosely coupled convex optimization problems. The second category of the contributions concerns analysis of large-scale uncertain systems. We next list the publications included in the thesis and lay out the contributions in each paper within either of these categories.

Distributed Solutions for Loosely Coupled Feasibility Problems Using Proximal Splitting Methods

Paper A,

S. Khoshfetrat Pakazad, M. S. Andersen, and A. Hansson. Distributed solutions for loosely coupled feasibility problems using proximal splitting methods. *Optimization Methods and Software*, 30(1):128–161, 2015a.

presents distributed algorithms for solving loosely coupled convex feasibility problems. These algorithms are devised using proximal splitting methods. The convergence rate properties for these methods are used to establish unified convergence rate results for the algorithms in terms of distance of the iterates to the feasible set. These convergence rate results also hold for the case when the feasibility problem is infeasible, in which case the results are in terms of certain error-bounds.

The first author of Paper A is the corresponding author for this paper and has to a large extent produced all the material in the paper using extensive comments from the co-authors.

A Distributed Primal-dual Interior-point Method for Loosely Coupled Problems Using ADMM

Paper B,

M. Annergren, S. Khoshfetrat Pakazad, A. Hansson, and B. Wahlberg. A distributed primal-dual interior-point method for loosely coupled problems using ADMM. *Submitted to Optimization Methods and Software*, 2015.

presents a distributed algorithm for solving loosely coupled convex optimization problems. This algorithm is based on a primal-dual interior-point method and it utilizes proximal splitting, specifically ADMM, for computing the primal-dual search directions distributedly. The generated search directions are generally inexact. Hence safe-guards are put in place to assure convergence of the primal-dual iterates to an optimal solution while using these inexact directions. The proposed algorithm in this paper puts very little computational burden on the computational agents, as they only need to compute one factorization of their local KKT matrix at every iteration of the primal-dual method.

The material presented in Paper B is the result of close collaboration of the first two authors, through incorporation of comments of the other co-authors. The main idea of this paper was produced by the second author, whom was heavily involved in the writing of the first half of the paper. The second part of the paper that concerns safe incorporation of inexact search directions within the algorithm is mainly the work of the first author of the paper.

Distributed Primal-dual Interior-point Methods for Solving Loosely Coupled Problems Using Message Passing

Paper C,

S. Khoshfetrat Pakazad, A. Hansson, and M. S. Andersen. Distributed primal-dual interior-point methods for solving loosely coupled problems using message passing. *Submitted to Optimization Methods and Software*, 2015b.

proposes a distributed algorithm for solving loosely coupled convex optimization problems with chordal or an inherent tree structure. Unlike the algorithm presented in Paper B, that relies on the use of inexact directions computed by proximal splitting methods, this algorithm relies on the use of exact directions that are computed using message passing within a finite number of steps. This number only relies on the coupling structure of the problem, and hence allows us to compute a practical upper bound on the number of required steps for convergence of the algorithm. Despite this advantage over the previous distributed algorithm, one must note that this algorithm can only be applied to problems with chordal sparsity, and as a result the algorithm presented in Paper B is applicable to more general classes of problems.

The main contributor for Paper C is the first author of the paper, who is also the corresponding author. This paper is the result of incorporating extensive comments from the co-authors.

Robust Stability Analysis of Sparsely Interconnected Uncertain Systems

Paper D,

M. S. Andersen, S. Khoshfetrat Pakazad, A. Hansson, and A. Rantzer. Robust stability analysis of sparsely interconnected uncertain systems. *IEEE Transactions on Automatic Control*, 59(8):2151–2156, 2014.

concerns the analysis of large-scale sparsely interconnected uncertain systems. Classical methods for analyzing such systems generally require solving a set of dense linear matrix inequalities. This paper puts forth an alternative but equivalent formulation of the analysis problem that requires solving a set of sparse linear matrix inequalities. These inequalities are much simpler and faster to solve, in a centralized manner. This formulation of the analysis problem also lays the foundation for devising distributed robustness analysis techniques for large-scale interconnected uncertain systems.

The initial idea for Paper D was pitched by the fourth author of the paper, and was later refined by the first and third authors. The paper has been written by the second author with contributions concerning the theoretical proofs regarding the equivalence between different formulations of the analysis problem. The numerical experiment setup was also developed by the second author. The sparse solver, SMCP, that was used in the numerical experiments have been produced by the first author. Parts of this paper were also presented in

M. S. Andersen, A. Hansson, S. Khoshfetrat Pakazad, and A. Rantzer. Distributed robust stability analysis of interconnected uncertain systems. In *Proceedings of the 51st Annual Conference on Decision and Control*, pages 1548–1553, December 2012.

Distributed Robustness Analysis of Interconnected Uncertain Systems Using Chordal Decomposition

Paper E,

S. Khoshfetrat Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer. Distributed robustness analysis of interconnected uncertain systems using chordal decomposition. In *Proceedings of the 19th IFAC World Congress*, volume 19, pages 2594–2599, 2014b.

utilizes the analysis formulation proposed in Paper D for devising a distributed robustness analysis algorithm for large-scale sparsely interconnected uncertain systems, with or without uncertain interconnections. This algorithm relies on decompositions techniques based on chordal sparsity, for decomposing the analysis problem. In order to solve the decomposed problem, distributed algorithms proposed in Paper A are used.

Paper E has been put together by the first author of the paper which has been considerably refined using comments from the the co-authors.

Distributed Semidefinite Programming with Application to Robustness Analysis of Large-scale Interconnected Uncertain Systems

Paper F,

S. Khoshfetrat Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer. Distributed semidefinite programming with application to large-scale system analysis. *Submitted to IEEE Transactions on Automatic Control*, 2015.

proposes a distributed algorithm for solving coupled semidefinite programs with a tree structure. This is achieved by extending the distributed algorithm presented in Paper C. The proposed algorithm is then used for solving robustness analysis problems with this structure.

The first author of Paper F is the corresponding author of this paper. The material presented in the paper has been produced using extensive comments from the co-authors.

Robust Finite-frequency $\mathcal{H}_2$ Analysis of Uncertain Systems with Application to Flight Comfort Analysis

Paper G,

A. Garulli, A. Hansson, S. Khoshfetrat Pakazad, R. Wallin, and A. Masi. Robust finite-frequency $\mathcal{H}_2$ analysis of uncertain systems with application to flight comfort analysis. *Control Engineering Practice*, 21(6): 887–897, 2013.

investigates the problem of robustness analysis over a finite frequency range. For instance, this can be relevant if the system operates within certain frequency intervals, or if the provided models are only valid up to a certain frequency. In either of these cases, performing the analysis over the whole frequency range may result in misleading conclusions. The algorithms presented in this paper, particularly the one based on frequency-gridding, are capable of solving such analysis problems, for uncertain systems with large number of states and uncertainty dimensions.

Paper G was written by the third author of the paper. However, the algorithm labeled *Gramian-based* in the paper has been entirely developed by the other authors of the paper, see Masi et al. (2010). Parts of this paper has also been presented in

S. Khoshfetrat Pakazad, A. Hansson, and A. Garulli. On the calculation of the robust finite frequency $\mathcal{H}_2$ norm. In *Proceedings of the 18th IFAC World Congress*, pages 3360–3365, August 2011.

1.2 Additional Publications

The following publications, that were produced by the author during his PhD studies, are not included in the thesis.

R. Wallin, S. Khoshfetrat Pakazad, A. Hansson, A. Garulli, and A. Masi. Applications of IQC-based analysis techniques for clearance. In A. Varga, A. Hansson, and G. Puyou, editors, *Optimization Based Clearance of Flight Control Laws*, volume 416 of *Lecture Notes in Control and Information Sciences*, pages 277–297. Springer Berlin Heidelberg, 2012.

H. Ohlsson, T. Chen, S. Khoshfetrat Pakazad, L. Ljung, and S. Sastry. Distributed change detection. In *Proceedings of the 16th IFAC Symposium on System Identification*, pages 77–82, July 2012.

S. Khoshfetrat Pakazad, M. S. Andersen, A. Hansson, and A. Rantzer. Decomposition and projection methods for distributed robustness analysis of interconnected uncertain systems. In *Proceedings of the 13th IFAC Symposium on Large Scale Complex Systems: Theory and Applications*, pages 194–199, August 2013a.

H. Ohlsson, T. Chen, S. Khoshfetrat Pakazad, L. Ljung, and S. Sastry. Scalable anomaly detection in large homogeneous populations. *Automatica*, 50(5):1459–1465, 2014.

S. Khoshfetrat Pakazad, H. Ohlsson, and L. Ljung. Sparse control using sum-of-norms regularized model predictive control. In *Proceedings of the 52nd Annual Conference on Decision and Control*, pages 5758–5763, December 2013b.

S. Khoshfetrat Pakazad, A. Hansson, and M. S. Andersen. Distributed interior-point method for loosely coupled problems. In *Proceedings of the 19th IFAC World Congress*, pages 9587–9592, August 2014a.

1.3 Thesis Outline

This thesis is divided into two parts, where Part I concerns the background material and Part II presents the seven papers listed in Section 1.1. Note that the background material presented in the first part of the thesis is not discussed in full detail as to keep the presentation concise. It is possible to find more details in the provided references.

The outline of Part I of the thesis is as follows. In Chapter 2 we discuss some of the basics concerning convex optimization problems. Methods for solving such problems are presented in Chapter 3. We present background material relating to robustness analysis of uncertain systems in Chapter 4. Chapter 5 provides a definition of coupled optimization problems and describes three approaches for devising distributed algorithms for solving such problems. Examples of coupled problems appearing in control and estimation are presented in Chapter 6, and we end Part I of the thesis with some concluding remarks in Chapter 7.

2

Convex Optimization Problems

A general optimization problem can be written as

$$\text{minimize } f_0(x) \tag{2.1a}$$

$$\text{subject to } g_i(x) \leq 0, \quad i = 1, \dots, m, \tag{2.1b}$$

$$h_i(x) = 0, \quad i = 1, \dots, p, \tag{2.1c}$$

where $f_0 : \mathbb{R}^n \rightarrow \mathbb{R}$ is the cost or objective function and $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$ for $i = 1, \dots, m$, and $h_i : \mathbb{R}^n \rightarrow \mathbb{R}$ for $i = 1, \dots, p$, are the inequality and equality constraint functions, respectively. The set defined by the constraints in (2.1b)–(2.1c) is referred to as the feasible set. The goal is to compute a solution that minimizes the cost function while belonging to the feasible set. To be able to do this, there must exist an x such that (2.1b)–(2.1c) are satisfied. The problem of establishing if this is true or not, is called a feasibility problem. Any x satisfying (2.1b)–(2.1c) is called feasible.

Many problems in control, estimation and system analysis can be written as *convex optimization problems*. In this chapter, we briefly review the definition of such problems and discuss some of the concepts relating to them. To this end, we follow Boyd and Vandenberghe (2004). We first review the definitions of convex sets and functions in Section 2.1. These are then used to describe convex optimization problems in Section 2.2. We discuss primal and dual formulations of such problems, in Section 2.3. The optimality conditions for solutions are reviewed in Section 2.4. Finally, we define equivalence between different problem formulations in Section 2.5.

2.1 Convex Sets and Functions

In order to characterize a convex optimization problem, we first need to define convex sets and functions. These are described next.

2.1.1 Convex Sets

Let us start by defining affine sets. A set $\mathcal{C} \subseteq \mathbb{R}^n$ is *affine* if for any $x_1, x_2 \in \mathcal{C}$,

$$x = \theta x_1 + (1 - \theta)x_2 \in \mathcal{C}, \quad (2.2)$$

for all $\theta \in \mathbb{R}$. Affine sets are special cases of convex sets. A set $\mathcal{C} \subseteq \mathbb{R}^n$ is *convex* if for any $x_1, x_2 \in \mathcal{C}$,

$$x = \theta x_1 + (1 - \theta)x_2 \in \mathcal{C}, \quad (2.3)$$

for all $0 \leq \theta \leq 1$. Another important subclass of convex sets, are convex cones. A set $\mathcal{C} \subseteq \mathbb{R}^n$ is a *convex cone* if for any $x_1, x_2 \in \mathcal{C}$,

$$x = \theta_1 x_1 + \theta_2 x_2 \in \mathcal{C}, \quad (2.4)$$

for all $\theta_1, \theta_2 \geq 0$. A convex cone is called *proper* if

- it contains its boundary;
- it has nonempty interior;
- it does not contain any line.

The sets $\mathbb{S}_+^n$ and $\mathbb{R}_+^n$ which represent the symmetric positive semidefinite $n \times n$ matrices and positive real n -dimensional vectors, respectively, are examples of proper cones. Next we review the definition of convex functions.

2.1.2 Convex Functions

Convex functions play a central role in defining the cost function and constraints in a convex optimization problem. A convex function is characterized using the following definition.

Definition 2.1 (Convex functions). A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex, if $\text{dom } f$ is convex and for all $x, y \in \text{dom } f$ and $0 \leq \theta \leq 1$,

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y). \quad (2.5)$$

Also a function is *strictly convex* if the inequality in (2.5) holds strictly for $0 < \theta < 1$.

Some important examples of convex functions are affine functions, norms, distance to convex sets and the indicator function of a convex set $\mathcal{C}$ defined as

$$\mathcal{I}_{\mathcal{C}}(x) = \begin{cases} \infty & x \notin \mathcal{C} \\ 0 & x \in \mathcal{C} \end{cases}. \quad (2.6)$$

Note that the convexity of the aforementioned functions can be verified using Definition 2.1.

2.2 Convex Optimization Problems

Having defined convex sets and functions, we can now define a convex optimization problem. Consider the definition of an optimization problem given in (2.1). We refer to this problem as convex if

- the functions f_0 and g_i for $i = 1, \dots, m$, are convex;
- the functions h_i for $i = 1, \dots, p$, are affine.

This definition of the optimization problem is clearly not the only definition of a convex problem and there are other definitions that, for instance, rely on the convexity of the feasible set and the cost function, see e.g., Bertsekas (2009). Depending on the characteristics of the cost and inequality constraint functions, convex problems are categorized into several subcategories. Next we review some of these subcategories that are closely studied in the papers in Part II.

2.2.1 Linear Programs

We call an optimization problem linear if the cost function, and the inequality and equality constraint functions are all affine. These problems are also referred to as *linear programs* (LPs) and can be written as

$$\text{minimize} \quad c^T x + d \quad (2.7a)$$

$$\text{subject to} \quad Gx \leq h, \quad (2.7b)$$

$$Ax = b, \quad (2.7c)$$

where $c \in \mathbb{R}^n$, $d \in \mathbb{R}$, $G \in \mathbb{R}^{m \times n}$, $h \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$ and $b \in \mathbb{R}^p$. Here $\leq$ denotes element-wise inequality.

2.2.2 Quadratic Programs

Quadratic programs (QPs) refer to optimization problems with a convex quadratic cost function and affine equality and inequality constraint functions. Such problems can be written as

$$\text{minimize} \quad \frac{1}{2}x^T Px + q^T x + c \quad (2.8a)$$

$$\text{subject to} \quad Gx \leq h, \quad (2.8b)$$

$$Ax = b, \quad (2.8c)$$

where $P \in \mathbb{S}_+^n$, $q \in \mathbb{R}^n$, $c \in \mathbb{R}$, $G \in \mathbb{R}^{m \times n}$, $h \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$ and $b \in \mathbb{R}^p$. Note that by choosing $P = 0$, this problem becomes an LP and hence, QPs include LPs as special cases. It is possible to extend the formulation in (2.8) to also include convex quadratic inequality constraints, as

$$\text{minimize} \quad \frac{1}{2}x^T Px + q^T x + c \quad (2.9a)$$

$$\text{subject to} \quad \frac{1}{2}x^T Q^i x + (q^i)^T x + c^i \leq 0, \quad i = 1, \dots, m, \quad (2.9b)$$

$$Ax = b, \quad (2.9c)$$

where the matrices $P, Q^1, \dots, Q^m \in \mathbb{S}_+^n$, $q, q^1, \dots, q^m \in \mathbb{R}^n$, $c, c^1, \dots, c^m \in \mathbb{R}$, $A \in \mathbb{R}^{p \times n}$ and $b \in \mathbb{R}^p$. Problems of the form (2.9) are referred to as *quadratically constrained quadratic programs* (QCQPs). Clearly QPs are special cases of QCQPs. In order to discuss other classes of convex optimization problems, we need to first take a short detour to describe the concept of generalized inequalities. This will then enable us to describe problems that are defined via generalized inequality constraints.

2.2.3 Generalized Inequalities

Generalized inequalities extend the definition of inequalities using proper cones. Let K be a proper cone. Then for $x, y \in \mathbb{R}^n$

$$\begin{aligned} x \leq_K y &\Leftrightarrow x - y \in K, \\ x <_K y &\Leftrightarrow x - y \in \text{int } K. \end{aligned} \quad (2.10)$$

Note that component-wise inequalities, and inequalities involving matrices, see Section 2.2.4, are special cases of the inequalities in (2.10). This can be seen by choosing K to be $\mathbb{R}_+^n$ or $\mathbb{S}_+^n$. Definition 2.1 can also be extended using proper cones. A function f is said to be convex with respect to a proper cone K , i.e., K -convex, if for all $x, y \in \text{dom } f$ and $0 \leq \theta \leq 1$

$$f(\theta x + (1 - \theta)y) \leq_K \theta f(x) + (1 - \theta)f(y). \quad (2.11)$$

Also f is strictly K -convex if the inequality in (2.11) holds strictly for $0 < \theta < 1$.

2.2.4 Semidefinite Programs

Let K be $\mathbb{S}_+^d$. Then the problem

$$\begin{aligned} & \underset{x}{\text{minimize}} && c^T x \\ & \text{subject to} && F_0 + \sum_{i=1}^n x_i F_i \preceq_K 0, \\ & && Ax = b, \end{aligned} \quad (2.12)$$

with $F_0, F_1, \dots, F_n \in \mathbb{S}^d$, $A \in \mathbb{R}^{p \times n}$ and $b \in \mathbb{R}^p$, defines a *semidefinite program* (SDP). In the remainder of this thesis we drop K and simply use $\preceq$ to also describe matrix inequality. It will be clear from the context whether $\preceq$ denotes element-wise or matrix inequality. SDPs appear in many problems in control and estimation, e.g., see Boyd et al. (1994). Particularly in this thesis we encounter SDPs in problems pertaining to robustness analysis of uncertain systems, see papers D–G, and in localization of sensor networks as discussed in Chapter 6.

2.3 Primal and Dual Formulations

Consider the problem in (2.1). This problem is referred to as the *primal* problem. The *Lagrangian* for this problem is defined as

$$L(x, \lambda, v) = f_0(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{i=1}^p v_i h_i(x), \quad (2.13)$$

where $\lambda = (\lambda_1, \dots, \lambda_m) \in \mathbb{R}^m$ and $v = (v_1, \dots, v_p) \in \mathbb{R}^p$ are the so-called dual variables or Lagrange multipliers for the inequality and equality constraints of the problem in (2.1), respectively. Let the collective domain of the optimization problem in (2.1) be given as $D := \text{dom } f_0 \cap \left(\bigcap_{i=1}^m \text{dom } g_i \right) \cap \left(\bigcap_{i=1}^p \text{dom } h_i \right)$. Then the corresponding dual function for this problem is defined as

$$g(\lambda, v) = \inf_{x \in D} L(x, \lambda, v). \quad (2.14)$$

Note that for $\lambda \geq 0$ and any feasible x we have

$$f_0(x) \geq L(x, \lambda, v) \geq g(\lambda, v). \quad (2.15)$$

Let us define $p^* := f_0(x^*)$ with x^* the optimal primal solution for (2.1). Then (2.15) implies that $p^* \geq g(\lambda, v)$. In other words, $g(\lambda, v)$ constitutes a lower bound for the optimal value of the original problem for all $v \in \mathbb{R}^p$ and $\lambda \geq 0$. In order to compute the best lower bound for p^* , we can then maximize $g(\lambda, v)$ as

$$\begin{aligned} & \underset{\lambda, v}{\text{maximize}} && g(\lambda, v) \\ & \text{subject to} && \lambda \geq 0. \end{aligned} \quad (2.16)$$

This problem is referred to as the *dual* problem. Let λ^* and v^* be the optimal dual solutions for (2.16). Depending on the problem, the lower bound $g(\lambda^*, v^*)$ can be arbitrarily tight or arbitrarily off. However, under certain conditions, it can be guaranteed that this lower bound is equal to p^* . In this case, it is said that we have *strong duality*. Conditions under which strong duality is guaranteed are called constraint qualification. For convex problems one of such set of conditions is the so-called Slater's condition which states that if the primal problem is convex and strictly feasible, i.e., there exists a feasible solution for the primal problem such that all inequality constraints hold strictly, strong duality holds. This condition can be relaxed for affine inequalities as they do not need to hold strictly, (Boyd and Vandenberghe, 2004; Bertsekas, 2009).

For convex problems where strong duality holds and (λ^*, v^*) exists, then any optimal primal solution is also a minimizer of $L(x, \lambda^*, v^*)$. This can sometimes enable us to compute the primal optimal solution using the optimal dual solutions or through solving the dual problem. For instance, assume that $L(x, \lambda^*, v^*)$ is a strictly convex function of x . Then if the solution of

$$\underset{x}{\text{minimize}} \quad L(x, \lambda^*, v^*) \quad (2.17)$$

is primal feasible, it is also the optimal primal solution for (2.1). It can be that in some cases solving the dual problem (2.16) and (2.17) is easier than solving the primal problem (2.1). In such cases, the above mentioned discussion lays out a procedure for computing the primal solution in an easier manner. In this thesis however, we mainly focus on primal formulations of optimization problems, and we have used the dual formulation only in Paper F. This is because, for most of the problems considered in this thesis, there was no reason for us to believe that solving the dual formulation of the problem would be computationally beneficial. Based on the above discussion, we next express conditions that characterize the optimal solutions for some convex problems.

2.4 Optimality Conditions

Consider the following optimization problem

$$\text{minimize} \quad f_0(x) \quad (2.18a)$$

$$\text{subject to} \quad g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (2.18b)$$

$$Ax = b, \quad (2.18c)$$

where $f_0 : \mathbb{R}^n \rightarrow \mathbb{R}$ and $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$ for $i = 1, \dots, m$, are all twice continuously differentiable, $b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{p \times n}$ with $\text{rank}(A) = p < n$. Let us assume that strong duality holds. Then $x \in \mathbb{R}^n$, $\lambda \in \mathbb{R}^m$ and $v \in \mathbb{R}^p$ are primal and dual optimal if and only if they satisfy the Karush-Kuhn-Tucker (KKT) optimality conditions given as

$$\nabla f_0(x) + \sum_{i=1}^m \lambda_i \nabla g_i(x) + A^T v = 0, \quad (2.19a)$$

$$\lambda_i \geq 0, \quad i = 1, \dots, m, \quad (2.19b)$$

$$g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (2.19c)$$

$$-\lambda_i g_i(x) = 0, \quad i = 1, \dots, m, \quad (2.19d)$$

$$Ax = b, \quad (2.19e)$$

see (Boyd and Vandenberghe, 2004, Ch. 5). Many optimization algorithms solve this set of conditions for finding the optimal solutions. We review some of these algorithms in Chapter 3.

As was discussed in Section 2.3, it is sometimes easier to solve the dual formulation of the problem. Devising simpler ways for solving optimization problems can also sometimes be done by solving equivalent reformulations. We conclude this chapter by reviewing the definition of equivalence between problems and we also discuss some instances of such problems.

2.5 Equivalent Optimization Problems

We refer to two problems as equivalent if the solution for one can be trivially computed from the solution of the other. Given an optimization problem there are many approaches for generating equivalent reformulations. Many approaches rely on adding and/or removing constraints and variables or changing the description of the constraints such that the feasible set remains the same. In this section we will review some of these approaches.

2.5.1 Removing and Adding Equality Constraints

Consider the following optimization problem

$$\underset{x,z}{\text{minimize}} \quad f_0(x) \quad (2.20a)$$

$$\text{subject to} \quad g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (2.20b)$$

$$h_i(x) = 0, \quad i = 1, \dots, p, \quad (2.20c)$$

$$x + Hz = b. \quad (2.20d)$$

where $x \in \mathbb{R}^n$, $z \in \mathbb{R}^p$, $H \in \mathbb{R}^{n \times p}$ and $b \in \mathbb{R}^n$. It is possible to equivalently rewrite this problem as

$$\underset{z}{\text{minimize}} \quad f_0(b - Hz) \quad (2.21a)$$

$$\text{subject to} \quad g_i(b - Hz) \leq 0, \quad i = 1, \dots, m, \quad (2.21b)$$

$$h_i(b - Hz) = 0, \quad i = 1, \dots, p, \quad (2.21c)$$

where we have removed the equality constraints in (2.20d). The two problems are equivalent since any solution z^* for (2.20) is also optimal for (2.21) and from any solution z^* it is possible to produce an optimal solution for (2.20) by simply choosing $x^* = b - Hz^*$. This technique of reformulating optimization problems is extensively used for devising distributed algorithms for solving optimization problems.

2.5.2 Introduction of Slack Variables

An inequality constrained optimization problem as

$$\underset{x}{\text{minimize}} \quad f_0(x) \tag{2.22a}$$

$$\text{subject to} \quad g_i(x) \leq 0, \quad i = 1, \dots, m, \tag{2.22b}$$

$$h_i(x) = 0, \quad i = 1, \dots, p, \tag{2.22c}$$

with $x \in \mathbb{R}^n$, can be equivalently rewritten as

$$\underset{x,s}{\text{minimize}} \quad f_0(x) \tag{2.23a}$$

$$\text{subject to} \quad g_i(x) + s_i = 0, \quad i = 1, \dots, m, \tag{2.23b}$$

$$h_i(x) = 0, \quad i = 1, \dots, p, \tag{2.23c}$$

$$s_i \geq 0, \quad i = 1, \dots, m. \tag{2.23d}$$

Here we have added so-called slack variables $s \in \mathbb{R}^m$ and the constraints in (2.23d). Notice that these two problems are equivalent as any solution x^* for (2.23) is also optimal for (2.22) and moreover given any optimal solution x^* for (2.22) one can easily construct an optimal solution for (2.23) by choosing $s_i^* = -g_i(x^*)$ for $i = 1, \dots, m$.

2.5.3 Reformulation of Feasible Set Description

Note that any transformation and change of constraint description that does not change the feasible set of an optimization problem, does not change its optimal solution. Consequently, any such changes to the constraint description results in an equivalent reformulation of the problem. In this thesis, we use this approach to equivalently reformulate optimization problems particularly when we deal with sparse SDPs, see for example Section 5.2 and Paper E.

In this chapter we reviewed some basic concepts related to convex optimization problems. However, we did not discuss how to solve these problems. Methods for solving convex optimization problems are discussed in the next chapter.

3

Convex Optimization Methods

There are several methods for solving convex optimization problems. In this chapter, we review two major classes of optimization methods for solving convex problems that are used extensively later on in the thesis. In Section 3.1 we discuss some of the basics related to the proximal point algorithm. This method is the foundation of the algorithms discussed in Paper A and is also used in Paper B. In Section 3.2, we review primal and primal-dual interior-point methods which are utilized in algorithms proposed in papers B, C and F.

3.1 Proximal Point Algorithm

Solving convex problems using the proximal point algorithm is closely related to computation of zeros of monotone operators. So in order to describe this algorithm, following Eckstein (1989), we first discuss monotone operators and how we can compute zeros of such operators.

3.1.1 Monotonicity and Zeros of Monotone Operators

Let us start by reviewing the definition of multi-valued operators and monotonicity.

Definition 3.1 (Multi-valued operators and monotonicity). A *multi-valued* or *set-valued* operator $T : \text{dom } T \rightarrow Y$, with $\text{dom } T \subseteq \mathbb{R}^n$, is an operator or a mapping that maps each $x \in \text{dom } T$ to a set $T(x) \in Y := \{c \mid c \subseteq \mathbb{R}^n\}$. The *graph* of this mapping is a set $\mathbf{T} = \{(x, y) \mid x \in \text{dom } T, y \in T(x)\}$. The *inverse* of operator T is an operator $T^{-1} : Y \rightarrow \text{dom } T$ and $\mathbf{T}^{-1} = \{(y, x) \mid (x, y) \in \mathbf{T}\}$. We refer to an operator, T , as *monotone* if

$$(y_2 - y_1)^T (x_2 - x_1) \geq 0, \quad \forall x_1, x_2 \in \text{dom } T \text{ and } \forall y_1 \in T(x_1), y_2 \in T(x_2), \quad (3.1)$$

and we call it *maximal monotone* if its graph is not a strict subset of a graph of any other monotone operator. _____

Let us define $J_{\rho T} = (I + \rho T)^{-1}$. This operator is referred to as the resolvent of the operator T with parameter $\rho > 0$, (Eckstein, 1989; Parikh and Boyd, 2014). The next theorem states one of the fundamental results regarding monotone operators.

Theorem 3.1. *Assume that T is maximal monotone and that it has a zero, i.e., $\exists x \in \text{dom } T$ such that $(x, 0) \in T$. It is then possible to compute a zero of T using the following recursion*

$$x^{(k+1)} = J_{\rho T}(x^{(k)}), \quad (3.2)$$

i.e., $x^{(k)} \rightarrow x$ as $k \rightarrow \infty$, for any initial $x^{(0)} \in \text{dom } J_{\rho T}$.

Proof: See (Eckstein, 1989, Thm 3.6, Prop. 3.9). □

3.1.2 Proximal Point Algorithm and Convex Problems

It is possible to use the recursion in (3.2) for solving convex optimization problems. In order to explore this possibility we need to put forth another definition.

Definition 3.2 (Subgradient). The subgradient ∂f of a function $f : \text{dom } f \rightarrow \mathbb{R} \cup \{\infty\}$ is an operator with the following graph

$$\partial f = \{(x, g) \mid x \in \text{dom } f \text{ and } f(y) \geq f(x) + \langle y - x, g \rangle \forall y \in \text{dom } f\}. \quad (3.3)$$

The subgradient operator at each point $x \in \text{dom } f$, i.e., $\partial f(x)$, is also referred to as the subdifferential of f at x , (Bertsekas and Tsitsiklis, 1997). _____

Let us now consider the following convex optimization problem

$$\underset{x}{\text{minimize}} \quad f(x), \quad (3.4)$$

where f is such that $\text{dom } f \neq \emptyset$ and such that it is lower-semicontinuous or closed, see (Eckstein, 1989, Def. 3.7). It is possible to compute an optimal solution for this problem by computing a zero of its subgradient operator, i.e., $x \in \text{dom } f$ such that $0 \in \partial f(x)$. It can be shown that ∂f is maximal monotone, Rockafellar (1966, 1970). By Theorem 3.1, it is then possible to compute a zero of ∂f or an optimal solution of (3.4) using the following recursion

$$x^{(k+1)} = J_{\rho \partial f}(x^{(k)}) = (I + \rho \partial f)^{-1}(x^{(k)}). \quad (3.5)$$

This recursion can be rewritten as

$$x^{(k+1)} = \text{prox}_{\rho f}(x^{(k)}) := \underset{x}{\text{argmin}} \left\{ f(x) + \frac{1}{2\rho} \|x - x^{(k)}\|^2 \right\}, \quad (3.6)$$

where $\text{prox}_{\rho f}$ is referred to as the proximity operator of f with parameter $\rho > 0$. This method for solving the convex problem in (3.4) is called the proximal point algorithm. Note that this algorithm can be used for solving general convex problems given as

$$\underset{x}{\text{minimize}} \quad f_0(x) \quad (3.7a)$$

$$\text{subject to} \quad g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (3.7b)$$

$$Ax = b. \quad (3.7c)$$

Let us define $\mathcal{I}_{\mathcal{C}}$ as the indicator function for its feasible set, $\mathcal{C}$. Then this problem can be equivalently rewritten as

$$\underset{x}{\text{minimize}} \quad f_0(x) + \mathcal{I}_{\mathcal{C}}(x), \quad (3.8)$$

which is in the same format as (3.4), and hence, can be solved using the proximal point algorithm.

Remark 3.1. Notice that in many cases using the proximal point algorithm is not a viable choice for solving convex problems. This is because for many convex problems each iteration of the recursion in (3.6) is at least as hard as solving the problem itself. However, through the use of operator splitting methods, as we will touch upon in Chapter 5, such methods can become extremely efficient for solving certain classes of convex problems.

Remark 3.2. The proximal point algorithm is closely related to gradient and subgradient methods for solving convex problems. This can be observed through interpretation of these methods within a path-following context. For instance, let us assume that f is differentiable, then it is possible to solve (3.4) by computing a solution for the differential equation

$$\frac{dx}{dt} = -\nabla f(x), \quad (3.9)$$

which results in the steepest descent path, (Bruck, 1975). One way to approximately compute this path is by solving a discrete-time approximation of the differential equation. Applying the Euler forward method for this purpose, results in the following difference equation

$$x^{(k+1)} = x^{(k)} - \alpha \nabla f(x^{(k)}), \quad (3.10)$$

which recovers the gradient descent update for solving (3.4). Notice that extra care must be taken in choosing α as to assure convergence or to assure that we follow the steepest descent path with sufficient accuracy, see e.g., (Boyd and Vandenberghe, 2004, Ch. 9). It is also possible to employ Euler's backward method for discretizing (3.9). This results in

$$x^{(k+1)} = (I + \rho \nabla h)^{-1}(x^{(k)}), \quad (3.11)$$

which is the recursion in (3.6). From the numerical methods perspective for solving differential equations, the Euler backward method has several advantages over its forward variant, (Ferziger, 1998), some of which cross over to the optimization context, for instance, the freedom to choose ρ and less sensitivity to bad scaling or ill-conditioning of the function f .

As can be seen from Remark 3.2, the proximal point algorithm is a first order method. In case the cost and inequality constraints functions are twice continuously differentiable, it is possible to employ much more efficient optimization tools for solving convex optimization problems. This is discussed in the next section.

3.2 Primal and Primal-dual Interior-point Methods

Given a convex optimization problem as in (2.18), one way to compute an optimal solution for the problem is to solve its KKT optimality conditions, (Boyd and Vandenberghe, 2004), (Wright, 1997). This approach is the foundation for the methods discussed in this section.

3.2.1 Primal Interior-point Methods

Primal interior-point methods compute a solution for (2.19) by solving a sequence of perturbed KKT conditions given as

$$\nabla f_0(x) + \sum_{i=1}^m \lambda_i \nabla g_i(x) + A^T v = 0, \quad (3.12a)$$

$$\lambda_i \geq 0, \quad i = 1, \dots, m, \quad (3.12b)$$

$$g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (3.12c)$$

$$-\lambda_i g_i(x) = \frac{1}{t}, \quad i = 1, \dots, m, \quad (3.12d)$$

$$Ax = b, \quad (3.12e)$$

for increasing values of $t > 0$. Within a primal framework this is done by first eliminating the equations in (3.12d) and the dual variables, λ , which results in

$$\nabla f_0(x) + \sum_{i=1}^m \frac{1}{-tg_i(x)} \nabla g_i(x) + A^T v = 0, \quad (3.13a)$$

$$g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (3.13b)$$

$$Ax = b. \quad (3.13c)$$

For every given t , primal interior-point methods employ Newton's method for solving (3.13). At each iteration of Newton's method and given an iterate $x^{(k)}$ such that $g_i(x^{(k)}) < 0$ for $i = 1, \dots, m$, the search directions are computed by solving the following linear system of equations

$$\begin{bmatrix} H_p^{(k)} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta v \end{bmatrix} = - \begin{bmatrix} r_{p,dual}^{(k)} \\ r_{primal}^{(k)} \end{bmatrix}, \quad (3.14)$$

Algorithm 1 Infeasible Start Newton's Method

```

1: Given  $k = 0$ ,  $\epsilon > 0$ ,  $\gamma \in (0, 1/2)$ ,  $\beta \in (0, 1)$ ,  $v^{(0)}$  and  $x^{(0)}$  such that  $g_i(x^{(0)}) < 0$ 
   for  $i = 1, \dots, m$ 
2: repeat
3:   Given  $x^{(k)}$ , compute  $\Delta x^{(k+1)}$  and  $\Delta v^{(k+1)}$  by solving (3.14)
4:   Compute the step size  $\alpha^{(k+1)}$  using line search as
5:     start with  $\alpha^{(k+1)} = 1$ 
6:     while  $\|r(x^{(k)} + \alpha^{(k+1)}\Delta x^{(k+1)}, v^{(k)} + \alpha^{(k+1)}\Delta v^{(k+1)})\| > (1 -$ 
7:        $\gamma\alpha^{(k+1)})\|r(x^{(k)}, v^{(k)})\|$ 
8:        $\alpha^{(k)} = \beta\alpha^{(k)}$ 
9:     end while
10:     $x^{(k+1)} = x^{(k)} + \alpha^{(k+1)}\Delta x^{(k+1)}$ 
11:     $v^{(k+1)} = v^{(k)} + \alpha^{(k+1)}\Delta v^{(k+1)}$ 
12:     $k = k + 1$ 
13: until  $\|(r_{\text{p,dual}}^{(k)}, r_{\text{primal}}^{(k)})\|^2 \leq \epsilon$ 

```

where

$$r_{\text{p,dual}}^{(k)} = \nabla f_0(x^{(k)}) - \sum_{i=1}^m \frac{1}{tg_i(x^{(k)})} \nabla g_i(x^{(k)}) + A^T v^{(k)}$$

$$r_{\text{primal}}^{(k)} = Ax^{(k)} - b$$

$$H_p^{(k)} = \nabla^2 f_0(x^{(k)}) + \sum_{i=1}^m \left[\frac{1}{tg_i(x^{(k)})^2} \nabla g_i(x^{(k)}) \nabla g_i(x^{(k)})^T - \frac{1}{tg_i(x^{(k)})} \nabla^2 g_i(x^{(k)}) \right].$$

In order for the computed direction to constitute a proper Newton direction we need to assume that the coefficient matrix of (3.14) is nonsingular (Boyd and Vandenberghe, 2004). See (Boyd and Vandenberghe, 2004, Sec. 10.1) for some of the conditions that guarantee this assumption. In case $H_p^{(k)}$ is nonsingular, instead of directly solving (3.14), it is also possible to solve this system of equations by first eliminating the primal variables direction as

$$\Delta x = -(H_p^{(k)})^{-1} \left(A^T \Delta v + r_{\text{p,dual}}^{(k)} \right), \quad (3.15)$$

and then solving

$$A(H_p^{(k)})^{-1} A^T \Delta v = r_{\text{primal}}^{(k)} - A(H_p^{(k)})^{-1} r_{\text{p,dual}}^{(k)}, \quad (3.16)$$

for Δv . These equations is referred to as the Newton equations. Having described the computation of the Newton directions, we can describe an infeasible Newton method in Algorithm 1 with $r(x, v) = (r_{\text{p,dual}}, r_{\text{primal}})$. This algorithm is used for solving (3.13) for any given t . A generic description of a primal interior-point method based on Algorithm 1 is given in Algorithm 2.

Algorithm 2 Primal Interior-point Method

```

1: Given  $q = 0$ ,  $t^{(0)} > 0$ ,  $\mu > 1$ ,  $\epsilon_p > 0$  and  $x^{(0)}$  such that  $g_i(x^{(0)}) < 0$  for all
    $i = 1, \dots, m$ 
2: repeat
3:   Given  $t^q$ , compute  $x^{(q+1)}$  by solving (3.13) using Alg. 1 starting at  $x^{(q)}$ 
4:   if  $m/t^{(q)} < \epsilon_p$  then
5:      $x^* = x^{(q+1)}$ 
6:     Terminate the iterations
7:   end if
8:    $t^{(q+1)} = \mu t^{(q)}$ 
9:    $q = q + 1$ 
10: until iterations are terminated
  
```

3.2.2 Primal-dual Interior-point Methods

Similar to a primal method, primal-dual methods also compute a solution for (2.19) by dealing with a sequence of the perturbed KKT conditions as in (3.12). Particularly, at each iteration of a primal-dual method and given primal and dual iterates $x^{(k)}$, $\lambda^{(k)}$ and $v^{(k)}$ such that $g_i(x^{(k)}) < 0$ and $\lambda_i^{(k)} > 0$ for all $i = 1, \dots, m$, these iterates are updated along the Newton directions computed from the linearized version of (3.12), given as

$$\left(\nabla^2 f_0(x^{(k)}) + \sum_{i=1}^m \lambda_i^{(k)} \nabla^2 g_i(x^{(k)}) \right) \Delta x + \sum_{i=1}^m \nabla g_i(x^{(k)}) \Delta \lambda_i + A^T \Delta v = -r_{\text{dual}}^{(k)}, \quad (3.17a)$$

$$-\lambda_i^{(k)} \nabla g_i(x^{(k)})^T \Delta x - g_i(x^{(k)}) \Delta \lambda_i = -\left(r_{\text{cent}}^{(k)}\right)_i, \quad i = 1, \dots, m, \quad (3.17b)$$

$$A \Delta x = -r_{\text{primal}}^{(k)}, \quad (3.17c)$$

where

$$r_{\text{dual}}^{(k)} = \nabla f_0(x^{(k)}) + \sum_{i=1}^m \lambda_i \nabla g_i(x^{(k)}) + A^T v^{(k)}, \quad (3.18a)$$

$$\left(r_{\text{cent}}^{(k)}\right)_i = -\lambda_i^{(k)} g_i(x^{(k)}) - 1/t, \quad i = 1, \dots, m, \quad (3.18b)$$

and $r_{\text{primal}}^{(k)}$ is defined as for the primal approach. The linearized perturbed KKT conditions in (3.17) can be rewritten compactly as

$$\begin{bmatrix} \nabla^2 f_0(x^{(k)}) + \sum_{i=1}^m \lambda_i^{(k)} \nabla^2 g_i(x^{(k)}) & Dg(x^{(k)})^T & A^T \\ -\text{diag}(\lambda^{(k)}) Dg(x^{(k)}) & -\text{diag}(g_1(x^{(k)}), \dots, g_m(x^{(k)})) & 0 \\ A & 0 & 0 \end{bmatrix} \times \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta v \end{bmatrix} = - \begin{bmatrix} r_{\text{dual}}^{(k)} \\ r_{\text{cent}}^{(k)} \\ r_{\text{primal}}^{(k)} \end{bmatrix}, \quad (3.19)$$

where

$$Dg(x) = \begin{bmatrix} \nabla g_1(x)^T \\ \vdots \\ \nabla g_m(x)^T \end{bmatrix}. \quad (3.20)$$

It is common to solve (3.19) by first eliminating $\Delta \lambda$ as

$$\Delta \lambda = -\text{diag}(g_1(x^{(k)}), \dots, g_m(x^{(k)}))^{-1} \left(\text{diag}(\lambda^{(k)}) Dg(x^{(k)}) \Delta x - r_{\text{cent}}^{(k)} \right), \quad (3.21)$$

and then solving the following linear system of equations

$$\begin{bmatrix} H_{\text{pd}}^{(k)} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta v \end{bmatrix} = - \begin{bmatrix} r_{\text{dual}}^{(k)} \\ r_{\text{primal}}^{(k)} \end{bmatrix} \quad (3.22)$$

where

$$H_{\text{pd}}^{(k)} = \nabla^2 f_0(x^{(k)}) + \sum_{i=1}^m \lambda_i^{(k)} \nabla^2 g_i(x^{(k)}) - \sum_{i=1}^m \frac{\lambda_i^{(k)}}{g_i(x^{(k)})} \nabla g_i(x^{(k)}) \nabla g_i(x^{(k)})^T, \quad (3.23)$$

and

$$r^{(k)} = r_{\text{dual}}^{(k)} + Dg(x^{(k)})^T \text{diag}(g_1(x^{(k)}), \dots, g_m(x^{(k)}))^{-1} r_{\text{cent}}^{(k)}.$$

for the remainder of the directions. The system of equations in (3.22) is also referred to as the augmented system. Having expressed a way to compute the primal-dual directions, we outline a primal-dual interior-point method in Algorithm 3.

Remark 3.3. In case $H_{\text{pd}}^{(k)}$ is invertible, instead of directly solving (3.22) for computing the search directions, similar to the primal case, it is possible to first eliminate Δx as

$$\Delta x = -(H_{\text{pd}}^{(k)})^{-1} (A^T \Delta v + r^{(k)}), \quad (3.24)$$

and form and solve the Newton equations instead. _____

Algorithm 3 Primal-dual Interior-point Method

-
- 1: Given $k = 0$, $\mu > 1$, $\epsilon > 0$, $\gamma \in (0, 1/2)$, $\beta \in (0, 1)$, $\epsilon_{\text{feas}} > 0$, $\lambda^{(0)} > 0$, $v^{(0)}$, $x^{(0)}$
such that $g_i(x^{(0)}) < 0$ for all $i = 1, \dots, m$ and $\hat{\eta}^{(0)} = \sum_{i=1}^m -\lambda_i^{(0)} g_i(x^{(0)})$
 - 2: **repeat**
 - 3: $t = \mu m / \hat{\eta}^{(k)}$
 - 4: Given t , $\lambda^{(k)}$, $v^{(k)}$ and $x^{(k)}$ compute $\Delta x^{(k+1)}$, $\Delta \lambda^{(k+1)}$, $\Delta v^{(k+1)}$ by solving
(3.21) and (3.22)
 - 5: Compute the step size $\alpha^{(k+1)}$ using line search as
 - 6: start with $\alpha_{\max} = \min \left\{ 1, \min_i \left\{ -\lambda_i^{(k)} / \Delta \lambda_i^{(k+1)} \mid \Delta \lambda_i^{(k+1)} < 0 \right\} \right\}$,
 - 7: **while** $\exists i : g_i(x^{(k)} + \alpha^{(k+1)} \Delta x^{(k+1)}) \geq 0$
 - 8: $\alpha^{(k+1)} = \beta \alpha^{(k+1)}$
 - 9: **end while**
 - 10: **while** $\left\| \begin{pmatrix} r_{\text{primal}}^{(k+1)} \\ r_{\text{dual}}^{(k+1)} \end{pmatrix} \right\| > (1 - \gamma \alpha^{(k+1)}) \left\| \begin{pmatrix} r_{\text{primal}}^{(k)} \\ r_{\text{dual}}^{(k)} \end{pmatrix} \right\|$
 - 11: $\alpha^{(k+1)} = \beta \alpha^{(k+1)}$
 - 12: **end while**
 - 13: $x^{(k+1)} = x^{(k)} + \alpha^{(k+1)} \Delta x^{(k+1)}$
 - 14: $\lambda^{(k+1)} = \lambda^{(k)} + \alpha^{(k+1)} \Delta \lambda^{(k+1)}$
 - 15: $v^{(k+1)} = v^{(k)} + \alpha^{(k+1)} \Delta v^{(k+1)}$
 - 16: $k = k + 1$
 - 17: $\hat{\eta}^{(k)} = \sum_{i=1}^m -\lambda_i^{(k)} g_i(x^{(k)})$
 - 18: **until** $\|r_{\text{primal}}^{(k)}\|^2, \|r_{\text{dual}}^{(k)}\|^2 \leq \epsilon_{\text{feas}}$ and $\hat{\eta}^{(k)} \leq \epsilon$
-

Remark 3.4. There are different types of primal and primal-dual interior-point methods that mainly differ in the choice of search directions and step size computations. However, in spite of these differences, they commonly have the same algorithmic structure. In order to keep the presentation simple, we here abstain from discussing other types of such methods and we refer to Boyd and Vandenberghe (2004); Wright (1997); Forsgren et al. (2002) for further reading.

Remark 3.5. In this section we have studied application of primal and primal-dual interior-point methods for solving non-conic optimization problems. It is also possible to use these methods for solving conic problems, such as SDPs. Interior-point methods for solving such problems are commonly more complicated and this is mainly due to intricacies of computing the (matrix-valued) search directions, see Wright (1997); Todd et al. (1998); de Klerk et al. (2000) for more information on this topic. We also discuss and study such methods in Paper F.

Remark 3.6. The primal and primal-dual interior-point methods discussed in this section, compute a solution for (2.19) using different approaches. The primal method does so by tracing the so-called central path to the optimal solution. This path is defined by the solutions of (3.12) for different values of t . As a result, at every iteration of Algorithm 2, we need to solve (3.13) to place the iterates on the central path. Due to this, given feasible starting points, the iterates in Algorithm 2 remain feasible. In contrast to this, the primal-

dual method in Algorithm 3 follows the central path to the optimal solution by staying in its close vicinity. This means that at every iteration of the primal-dual method we do not need to solve (3.12) and instead we take a single step along the Newton directions computed from (3.19), with a step size that assures that we stay close to the central path and that we reduce the primal and dual residuals at every iteration. Because of this, primal-dual methods, commonly, converge faster and require less computational effort. However, this comes at a cost that the iterates in Algorithm 3 do not have to be feasible. _____

The distributed algorithms proposed in this thesis rely on the optimization methods described in this chapter. Particularly, these algorithms are achieved by applying these methods to reformulated and decomposed formulations of convex problems, and/or by exploiting structure in the computations within different stages of these methods. This is discussed in more detail in Chapter 5. Next we discuss some basics relating to robustness analysis of uncertain systems which are used in papers D–G.

4

Uncertain Systems and Robustness Analysis

In this chapter, we explore some of the basic concepts in robustness analysis of uncertain systems. Firstly we discuss linear systems and some related topics in Section 4.1. Some concepts regarding uncertain systems are reviewed in Section 4.2. Finally in Section 4.3, we briefly discuss integral quadratic constraints (IQCs) and describe how they can be used for analyzing robust stability of uncertain systems. This chapter follows the material in Zhou and Doyle (1998); Megretski and Rantzer (1997); Jönsson (2001) and it is recommended to consult these references for more details on the discussed topics.

4.1 Linear Systems

In this section we review some of the basics of linear systems, that are extensively used in papers D–G.

4.1.1 Continuous-time Systems

Consider the following linear time-invariant (LTI) system

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t),\end{aligned}\tag{4.1}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$ and $D \in \mathbb{R}^{p \times m}$. This is a state-space representation for the system. The corresponding transfer function matrix for this system is defined as $G(s) = C(sI - A)^{-1}B + D \in \mathbb{C}^{p \times m}$, with $s \in \mathbb{C}$. The relationship between these two representations is sometimes denoted as

$$G(s) := \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array} \right].\tag{4.2}$$

The system is said to be stable, if for all the eigenvalues, λ_i , of the matrix A we have that $\text{Re } \lambda_i < 0$. Given an initial state, $x(t_0)$ and an input $u(t)$, the system responses $x(t)$ and $y(t)$ are given by

$$\begin{aligned} x(t) &= e^{A(t-t_0)}x(t_0) + \int_{t_0}^t e^{A(t-\tau)}Bu(\tau) d\tau, \\ y(t) &= Cx(t) + Du(t). \end{aligned} \quad (4.3)$$

4.1.2 $\mathcal{H}_\infty$ and $\mathcal{H}_2$ Norms

The $\mathcal{H}_2$ norm for a stable system of the form (4.1), is defined as

$$\|G\|_2^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} \text{tr} \{G(j\omega)^*G(j\omega)\} d\omega. \quad (4.4)$$

For the $\mathcal{H}_2$ norm to be finite it is required that $D = 0$. In that case, the $\mathcal{H}_2$ norm of the system can be calculated as

$$\|G\|_2^2 = \text{tr} \{B^T Q B\} = \text{tr} \{C P C^T\}, \quad (4.5)$$

where $P \in \mathbb{S}_+^n$ and $Q \in \mathbb{S}_+^n$ are the controllability and observability Gramians, respectively, (Zhou and Doyle, 1998). These are the unique solutions to the following Lyapunov equations

$$\begin{aligned} AP + PA^T + BB^T &= 0, \\ AQ^T + QA + C^T C &= 0. \end{aligned} \quad (4.6)$$

The $\mathcal{H}_\infty$ norm for a stable system of the form (4.1) is defined as below

$$\|G\|_\infty = \sup_{\omega} \bar{\sigma} \{G(j\omega)\}, \quad (4.7)$$

where $\bar{\sigma}$ denotes the maximum singular value for a matrix. In this thesis, methods for computing the $\mathcal{H}_\infty$ norm are not discussed and we just state that this quantity can be computed by solving an SDP, (Boyd et al., 1994), or by iterative algorithms using bisection, (Boyd et al., 1988; Zhou and Doyle, 1998).

4.2 Uncertain Systems

There are different ways for expressing uncertainty in a system. These descriptions fall mainly into the two categories of structured and unstructured uncertainties. In this section, we only discuss structured uncertainties. For a discussion on unstructured uncertainties, e.g., see Zhou and Doyle (1998).

Figure 4.1: Uncertain system with structured uncertainty.

4.2.1 Structured Uncertainties and LFT Representation

In case of a bounded uncertainty and rational system uncertainty dependence, it is possible to describe an uncertain system using a feedback connection of an uncertainty block and a time-invariant system, as illustrated in Figure 4.1, where Δ represents the extracted uncertainties from the system and

$$G = \begin{bmatrix} G_{pq} & G_{pw} \\ G_{zq} & G_{zw} \end{bmatrix},$$

is the system transfer function matrix. Let $G(s) \in \mathcal{RH}_{\infty}^{(d+l) \times (d+m)}$, $G_{pq}(s) \in \mathcal{RH}_{\infty}^{d \times d}$, $G_{pw}(s) \in \mathcal{RH}_{\infty}^{d \times m}$, $G_{zq}(s) \in \mathcal{RH}_{\infty}^{l \times d}$ and $G_{zw}(s) \in \mathcal{RH}_{\infty}^{l \times m}$. The system transfer function matrix can always be constructed such that Δ has the following block diagonal structure

$$\Delta = \text{blk diag}(\delta_1 I_{r_1}, \dots, \delta_L I_{r_L}, \Delta_{L+1}, \dots, \Delta_{L+F}), \quad (4.8)$$

where $\delta_i \in \mathbb{C}$ for $i = 1, \dots, L$, $\Delta_{L+j} \in \mathbb{C}^{m_j \times m_j}$ for $j = 1, \dots, F$, and $\sum_{i=1}^L r_i + \sum_{j=1}^F m_j = d$,

and where all blocks have a bounded induced ∞ -norm less than or equal to 1, i.e., $|\delta_i| \leq 1$, for $i = 1, \dots, L$, and $\|\Delta_i\|_{\infty} \leq 1$, for $i = L+1, \dots, L+F$. Systems with an uncertainty structure as in (4.8) are referred to as uncertain systems with structured uncertainties, (Zhou and Doyle, 1998), (Fan et al., 1991).

The representation of the uncertain system in Figure 4.1 is also referred to as the linear fractional transformation (LFT) representation of the system, (Magni, 2006). Using this representation it is possible to describe the mapping between $w(t)$ and $z(t)$ as below

$$\Delta * G := G_{zw} + G_{zq} \Delta (I - G_{pq} \Delta)^{-1} G_{pw}, \quad (4.9)$$

which is referred to as the upper LFT with respect to Δ , (Zhou and Doyle, 1998). This type of representation of uncertain systems is used in papers D–G.

Figure 4.2: Uncertain system with a structured uncertainty and a controller.

4.2.2 Robust $\mathcal{H}_\infty$ and $\mathcal{H}_2$ Norms

Robust $\mathcal{H}_2$ and $\mathcal{H}_\infty$ norms for uncertain systems with structured uncertainties are defined as

$$\sup_{\Delta \in B_\Delta} \|\Delta * G\|_2^2 = \sup_{\Delta \in B_\Delta} \frac{1}{2\pi} \int_{-\infty}^{\infty} \text{tr} \{(\Delta * G)^*(\Delta * G)\} d\omega, \quad (4.10a)$$

$$\sup_{\Delta \in B_\Delta} \|\Delta * G\|_\infty = \sup_{\Delta \in B_\Delta} \sup_{\omega} \bar{\sigma} \{\Delta * G\}, \quad (4.10b)$$

respectively, where B_Δ represents the unit ball for the induced ∞ -norm for the uncertainty structure in (4.8). These quantities are of great importance and in case $w(t)$ and $z(t)$ are viewed as disturbance acting on the system and controlled output of the system, respectively, these norms quantify the worst-case effect of the disturbance on the performance of the system. The computation of these quantities is not discussed in this section and for more information on the calculation of upper bounds for these norms refer to Paganini and Feron (2000); Paganini (1997, 1999a); Doyle et al. (1989); Zhou and Doyle (1998).

4.2.3 Nominal and Robust Stability and Performance

Consider the interconnection of a controller K with an uncertain system with structured uncertainty as in Figure 4.2. Let us assume that K has been designed such that the nominal system, i.e., when $\Delta = 0$, together with the controller is stable and satisfies certain requirements over the performance of the closed loop system. Then it is said that the system is *nominally stable* and satisfies *nominal performance* requirements.

Under this assumption, if the process G and the controller K are combined, then the setup in Figure 4.2 will transform into the one presented in Figure 4.1. In this case, if the resulting system together with the uncertainty is stable and satisfies the nominal requirements on its behavior, it is said that the system is

Figure 4.3: Uncertain system with system transfer function matrix G and uncertainty Δ .

robustly stable and has *robust performance*. Next we briefly review a framework for analyzing robust stability of uncertain systems with structured uncertainty.

4.3 Robust Stability Analysis of Uncertain Systems Using IQCs

There are different approaches for analyzing robust stability of uncertain systems, one of which is referred to as IQC analysis. This method is based on integral quadratic constraints (IQCs) which allow us to describe the uncertainty in the system. Particularly, it is said that a bounded and causal operator Δ satisfies the IQC defined by Π , i.e., $\Delta \in \text{IQC}(\Pi)$, if

$$\int_0^\infty \begin{bmatrix} v \\ \Delta(v) \end{bmatrix}^T \Pi \begin{bmatrix} v \\ \Delta(v) \end{bmatrix} dt \geq 0, \quad \forall v \in \mathcal{L}_2^d, \quad (4.11)$$

where Π is a bounded and self-adjoint operator, (Megretski and Rantzer, 1997), (Jönsson, 2001). This constraint can also be rewritten in the frequency domain as

$$\int_{-\infty}^\infty \begin{bmatrix} \hat{v}(j\omega) \\ \widehat{\Delta(v)}(j\omega) \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} \hat{v}(j\omega) \\ \widehat{\Delta(v)}(j\omega) \end{bmatrix} d\omega \geq 0, \quad (4.12)$$

where $\hat{v}$ and $\widehat{\Delta(v)}$ are the Fourier transforms of the signals, and Π is a transfer function matrix such that $\Pi(j\omega) = \Pi(j\omega)^*$. Consider the following uncertain system

$$\begin{aligned} p &= Gq \\ q &= \Delta(p), \end{aligned} \quad (4.13)$$

where $p, q \in \mathcal{L}_2^m$, $G \in \mathcal{RH}_\infty^{m \times m}$ is the system transfer function matrix and $\Delta \in \text{IQC}(\Pi)$ is the uncertainty in the system, see Figure 4.3. Then robust stability of the system in (4.13) can be established using the following theorem.

Theorem 4.1 (IQC analysis, (Megretski and Rantzer, 1997)). *The uncertain system in (4.13) is robustly stable, if*

1. *for all $\tau \in [0, 1]$ the interconnection described in (4.13), with $\tau\Delta$, is well-posed;*
2. *for all $\tau \in [0, 1]$, $\tau\Delta \in \text{IQC}(\Pi)$;*
3. *there exists $\epsilon > 0$ such that*

$$\begin{bmatrix} G(j\omega) \\ I \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} G(j\omega) \\ I \end{bmatrix} \leq -\epsilon I, \quad \forall \omega \in [0, \infty]. \quad (4.14)$$

Theorem 4.1 provides a sufficient condition for robust stability of uncertain systems. In order to establish this condition, we are required to find a multiplier Π , such that $\Delta \in \text{IQC}(\Pi)$ and such that the semi-infinite LMI in (4.14) is satisfied. Commonly, the condition $\Delta \in \text{IQC}(\Pi)$ imposes structural constraints on Π . The robust stability analysis problem will then boil down to finding a multiplier, with the required structure, such that the LMI in (4.14) is satisfied for all frequencies. There are mainly two approaches for solving this LMI, which are based on frequency gridding and the use of the KYP lemma, (Rantzer, 1996; Paganini, 1999b). The frequency-gridding-based approach solves the analysis problem approximately, by establishing the feasibility of the LMI in (4.14) for a finite number of frequencies. This approach preserves the structure in the LMI which, as is discussed in papers D–F, will enable us to efficiently solve the analysis problem for some uncertain systems.

So far we have discussed some of the basics regarding convex optimization and robustness analysis. Next, we describe the design procedure for optimization algorithms that enable us to solve some of the encountered optimization problems, especially relating to robustness analysis, distributedly.

5

Coupled Problems and Distributed Optimization

In this chapter we lay out how the methods described in Chapter 3 can be used for devising distributed algorithms for solving coupled optimization problems. To this end, we first put forth definitions of coupled and loosely coupled problems in Section 5.1. We then consider sparse SDPs and show how they can be equivalently reformulated as coupled optimization problems in Section 5.2. The importance of these problems will become clear as we discuss the applications in Chapter 6. Also such problems particularly take a central role in papers D–F. We describe the design procedure for distributed algorithms in Section 5.3, where we briefly discuss the design of distributed algorithms based on proximal splitting, primal and primal-dual methods in sections 5.3.1–5.3.3.

5.1 Coupled Optimization Problems and Structure Exploitation

We refer to the problem

$$\text{minimize} \quad f_1(x) + \cdots + f_N(x) \tag{5.1a}$$

$$\text{subject to} \quad x \in \mathcal{C}_i, \quad i = 1, \dots, N, \tag{5.1b}$$

with $x \in \mathbb{R}^n$, as a *coupled* convex optimization problem, since this problem can be viewed as a combination of N coupled subproblems each of which is defined by a cost function f_i and a constraint set $\mathcal{C}_i$. We assume that each subproblem, i , depends only on some of the elements of x . We denote the ordered set of indices of these variables by $J_i \subseteq \mathbb{N}_n$. Let us also denote the ordered set of subproblems that depend on x_j with $\mathcal{I}_j \subseteq \mathbb{N}_N$. We refer to this problem as *loosely* or *partially* coupled if $|J_i| \ll n$ and $|\mathcal{I}_j| \ll N$ for all $i = 1, \dots, N$, and $j = 1, \dots, n$. As we will

Figure 5.1: The coupling and sparsity graphs for the problem in (5.2), illustrated on the left and right figures, respectively.

discuss later, our proposed distributed algorithms can be efficiently applied for solving such problems.

Notice that the sets J_i for $i = 1, \dots, N$, and $\mathcal{I}_j$ for $j = 1, \dots, n$, fully describe the coupling structure in the problem. It is also possible to express the coupling structure in the problem graphically, using graphs. For this purpose, we introduce coupling and sparsity graphs. The *coupling graph* $G_c(V_c, \mathcal{E}_c)$ of a coupled problem is an undirected graph with the vertex set $V_c = \{1, \dots, N\}$ and the edge set $\mathcal{E}_c = \{(i, j) \mid i, j \in V_c, J_i \cap J_j \neq \emptyset\}$. Similarly, the *sparsity graph* $G_s(V_s, \mathcal{E}_s)$ for the problem is also an undirected graph but with the vertex set $V_s = \{1, \dots, n\}$ and the edge set $\mathcal{E}_s = \{(i, j) \mid i, j \in V_s, \mathcal{I}_i \cap \mathcal{I}_j \neq \emptyset\}$. In order to better understand the introduced concepts, let us consider the following example

$$\text{minimize} \quad f_1(x_1, x_2, x_3) + f_2(x_3, x_4, x_5) + f_3(x_5, x_6, x_7) \quad (5.2a)$$

$$\text{subject to} \quad g_1(x_1, x_2, x_3) \leq 0, \quad (5.2b)$$

$$g_2(x_3, x_4, x_5) \leq 0, \quad (5.2c)$$

$$g_3(x_5, x_6, x_7) \leq 0, \quad (5.2d)$$

The coupling and sparsity graphs for this problem are illustrated in Figure 5.1. The coupling structure and its graphical representation play a central role in design and description of distributed algorithms for solving coupled problems. This is discussed later in Section 5.3. Next we briefly discuss how sparsity in SDPs can be used for reformulating them as coupled problems.

5.2 Sparsity in SDPs

SDPs appear in many robustness analysis problems, and hence in this thesis we encounter many coupled version of such problems. These problems are commonly the result of reformulation of sparse SDPs which are discussed in this section.

Figure 5.2: A nonchordal graph (on the left) and its chordal embedding (on the right). As can be seen, the chordal embedding has been achieved by adding an edge between vertices 5 and 7.

5.2.1 Chordal Graphs

Given a graph $G(V, \mathcal{E})$, vertices $i, j \in V$ are *adjacent* if $(i, j) \in \mathcal{E}$. We denote the set of adjacent vertices or neighbors of i by $\text{Ne}(i) = \{j \in V \mid (i, j) \in \mathcal{E}\}$. A graph is said to be *complete* if all its vertices are adjacent. An *induced graph* by $V' \subseteq V$ on $G(V, \mathcal{E})$, is a graph $G_I(V', \mathcal{E}')$ where $\mathcal{E}' = \mathcal{E} \cap (V' \times V')$. A *clique* C_i of $G(V, \mathcal{E})$ is a maximal subset of V that induces a complete subgraph on G . Consequently, no clique is properly contained in another clique, (Blair and Peyton, 1994). Assume that all cycles of length at least four of $G(V, \mathcal{E})$ have a chord, where a chord is an edge between two non-consecutive vertices in a cycle. This graph is then called *chordal*, (Golumbic, 2004, Ch. 4). It is possible to make a non-chordal graph chordal by adding edges to the graph. The resulting graph from this procedure is referred to as a *chordal embedding*.

Figure 5.2 illustrates an example of this process. The figure on the left is a non-chordal graph. This can be seen from the cycle defined over the vertices 5, 6, 7, 8 that does not have a chord. We can produce a chordal embedding by adding a chord to this cycle, i.e., an edge between vertices 5 and 7 or 6 and 8. The figure on the right illustrates a chordal embedding of this graph which has been generated by adding an edge between vertices 5 and 7. Let $\mathbf{C}_G = \{C_1, \dots, C_l\}$ denote the set of its cliques, where l is the number of cliques of the graph. Then there exists a tree defined on $\mathbf{C}_G$ such that for every $C_i, C_j \in \mathbf{C}_G$ where $i \neq j$, $C_i \cap C_j$ is contained in all the cliques in the path connecting the two cliques in the tree. This property is called the *clique intersection property*, (Blair and Peyton, 1994). Trees with this property are referred to as *clique trees*. Consider the example in Figure 5.2. The marked cliques of the chordal embedding and its clique tree are depicted in Figure 5.3. Having introduced some definitions related to chordal graphs we next discuss sparsity in matrices.

Figure 5.3: Cliques and clique tree for the chordal embedding illustrated in Figure 5.2.

5.2.2 Chordal Sparsity

A matrix is sparse if the number of non-zero elements in the matrix is considerably smaller than the number of zero elements. The sparsity pattern of a matrix describes where the non-zero elements are located in the matrix. Let $A \in \mathbb{R}^{n \times n}$ be a sparse matrix, then the sparsity pattern of this matrix can be described using the following set

$$S = \{(i, j) \mid A_{ij} \neq 0\}, \quad (5.3)$$

where A_{ij} denotes the element on the i^{th} row and j^{th} column. In the applications in this thesis we mainly deal with symmetric matrices. As a result, from now on we only study the sparsity pattern for symmetric matrices. Note that if the matrix is symmetric and $(i, j) \in S$ then $(j, i) \in S$. The sparsity pattern for a sparse symmetric matrix can also be described through its associated *sparsity pattern graph*. The sparsity pattern graph for a matrix A is an undirected graph with vertex set $V = \{1, \dots, n\}$ and edge set $\mathcal{E} = \{(i, j) \mid A_{ij} \neq 0, i \neq j\}$. Note that by this definition, the sparsity pattern graph does not include any self-loops and does not represent any of the diagonal entries of the matrix.

Graphs can also be used to characterize partially symmetric matrices. Partially symmetric matrices correspond to symmetric matrices where only a subset of their elements are specified and the rest are free. We denote the set of all $n \times n$ partially symmetric matrices on a graph $G(V, \mathcal{E})$ by $\mathbb{S}_G^n$, where only elements with indices belonging to $\mathbf{I}_s = \mathcal{E} \cup \{(i, i) \mid i = 1, \dots, n\}$ are specified. Now consider a matrix $X \in \mathbb{S}_G^n$. Then X is said to be negative semidefinite completable if by choosing its free elements, i.e., elements with indices belonging to $\mathbf{I}_f = (V \times V) \setminus \mathbf{I}_s$, we can obtain a negative semidefinite matrix. The following theorem states a fundamental result on negative semidefinite completion.

Theorem 5.1. (Grone et al., 1984, Thm. 7) Let $G(V, \mathcal{E})$ be a chordal graph with cliques $C_1, \dots, C_l$ such that clique intersection property holds. Then $X \in \mathbb{S}_G^n$ is negative semidefinite completable, if and only if

$$X_{C_i C_i} \leq 0, \quad i = 1, \dots, l, \quad (5.4)$$

where $X_{C_i C_i} = E_{C_i} X E_{C_i}^T$ with E_{C_i} a 0–1 matrix that is obtained from I_n by removing all rows indexed by $\mathbb{N}_n \setminus C_i$.

Note that the matrices $X_{C_i C_i}$ for $i = 1, \dots, l$, are the fully specified principle submatrices of X . Hence, Theorem 5.1 states that a chordal matrix $X \in \mathbb{S}_G^n$ is negative semidefinite completable if and only if all its fully specified principle submatrices are negative semidefinite. As we will see next, this property can be used for decomposing SDPs with this structure.

5.2.3 Domain-space Decomposition

Consider a chordal graph $G(V, \mathcal{E})$, with $\{C_1, \dots, C_l\}$ the set of cliques such that clique intersection property holds. Let us define sets $\bar{J}_i \subset \mathbb{N}_n$ such that the sparsity pattern graph for $\sum_{i=1}^N E_{\bar{J}_i}^T E_{\bar{J}_i}$ is $G(V, \mathcal{E})$. Then for the following SDP

$$\underset{X}{\text{minimize}} \quad \sum_{i=1}^N W^i \bullet (E_{\bar{J}_i} X E_{\bar{J}_i}^T) \quad (5.5a)$$

$$\text{subject to} \quad Q^i \bullet (E_{\bar{J}_i} X E_{\bar{J}_i}^T) = b_i, \quad i = 1, \dots, N, \quad (5.5b)$$

$$X \leq 0, \quad (5.5c)$$

with $W^i, Q^i \in \mathbb{S}^{|\bar{J}_i|}$ for $i = 1, \dots, N$ and $X \in \mathbb{S}^n$, the only elements of X that affect the cost function in (5.5a) and the constraint in (5.5b) are elements specified by indices in $\mathbf{I}_s$. Using Theorem 5.1, the optimization problem in (5.5) can then be equivalently rewritten as

$$\underset{X_{C_1 C_1}, \dots, X_{C_l C_l}}{\text{minimize}} \quad \sum_{i=1}^N W^i \bullet (E_{\bar{J}_i} X E_{\bar{J}_i}^T) \quad (5.6a)$$

$$\text{subject to} \quad Q^i \bullet (E_{\bar{J}_i} X E_{\bar{J}_i}^T) = b_i, \quad i = 1, \dots, N, \quad (5.6b)$$

$$X_{C_i C_i} \leq 0, \quad i = 1, \dots, l, \quad (5.6c)$$

where notice that the constraint in (5.5c) has been rewritten as several coupled semidefinite constraints in (5.6c), (Fukuda et al., 2000; Kim et al., 2011). This method of reformulating (5.5) as the coupled SDP in (5.6) is referred to as the domain-space decomposition, (Kim et al., 2011; Andersen, 2011). There are other approaches for decomposing sparse SDPs, one of which is the so-called range-space decomposition that is discussed in Paper E. This decomposition scheme is in fact the dual of the procedure discussed in this section. Domain-space decomposition is later used in Paper F and in Chapter 6. Next we discuss design procedures for generating optimization algorithms for solving coupled problems distributedly.

5.3 Distributed Optimization Algorithms

It is possible to solve coupled optimization problems using a network of several computational agents that collaborate or communicate with one another. The algorithms that facilitate such solutions are referred to as *distributed algorithms*. The collaboration and/or communication protocol among these agents can be described using a graph, where each vertex of the graph corresponds to each of the agents and there exists an edge between two vertices if the corresponding agents need to collaborate and communicate with each other. This graph is referred to as the *computational graph* of the algorithm and is usually set by the coupling structure in the problem. Next we describe three approaches for devising distributed algorithms that have been used in papers A–C.

5.3.1 Distributed Algorithms Purely Based on Proximal Splitting

Operator splitting is at the heart of the design procedure discussed in this section. So before we go any further let us discuss some of the basics related to this topic. Recall that given a maximal monotone operator T , we can compute a zero of this operator using the following iterative scheme

$$x^{(k+1)} = (I + \rho T)^{-1} (x^{(k)}), \quad (5.7)$$

with $x^{(k)} \in \mathbb{R}^n$. Now consider the case where $T = T_1 + T_2$, with T_1 and T_2 both maximal monotone. Let us assume that computing $(I + \rho T)^{-1}$ is computationally much more demanding than computing either of $(I + \rho T_1)^{-1}$ or $(I + \rho T_2)^{-1}$. Operator splitting methods allow us to compute a zero of T using algorithms that rely on repeated applications of operators T_1 , T_2 and/or their corresponding resolvents. For instance, one of such methods is the so-called double-backward method which can be written as

$$y^{(k+1)} = (I + \rho T_1)^{-1} (x^{(k)}), \quad (5.8a)$$

$$x^{(k+1)} = (I + \rho T_2)^{-1} (y^{(k+1)}), \quad (5.8b)$$

with $y^{(k)} \in \mathbb{R}^n$, where it can be seen that each stage of the algorithm only relies on the resolvent of one of the subsequent operators. As a result, each iteration of (5.8) is much cheaper than that of (5.7). There are many other operator splitting methods such as forward-backward, Douglas-Rachford and Peaceman-Rachford, see Eckstein (1989); Combettes and Pesquet (2011); Bauschke and Combettes (2011) for more instances.

We can naturally use operator splitting methods for solving convex problems of the form

$$\text{minimize} \quad F(x) := F_1(x) + F_2(x), \quad (5.9)$$

where $x \in \mathbb{R}^n$, $\text{dom } F_1 \neq \emptyset$, $\text{dom } F_2 \neq \emptyset$ and the functions F_1 and F_2 are both lower-semicontinuous. Solving this optimization problem is equivalent to computing a

zero of the subdifferential operator $\partial F = \partial F_1 + \partial F_2$. Notice that ∂F_1 and ∂F_2 are both maximal monotone. Let us assume that computing the proximity operator of F is much more computationally demanding than computing those of F_1 and F_2 . It is of course possible to solve (5.9) using the proximal point algorithm. However, using operator splitting methods, one can compute an optimal solution for (5.9) using iterative methods that rely on proximity operators of F_1 and/or F_2 in a computationally cheaper manner. This technique for solving (5.9) is referred to as *proximal splitting*. Now let us consider the problem in (5.1). This problem can be equivalently rewritten as

$$\underset{S, x}{\text{minimize}} \quad \bar{f}_1(s^1) + \cdots + \bar{f}_N(s^N) \quad (5.10a)$$

$$\text{subject to} \quad s^i \in \bar{\mathcal{C}}_i, \quad i = 1, \dots, N, \quad (5.10b)$$

$$s^i = E_{J_i} x, \quad i = 1, \dots, N, \quad (5.10c)$$

where $s^i \in \mathbb{R}^{|J_i|}$ for $i = 1, \dots, N$ are so-called *local* variables, $x \in \mathbb{R}^n$, E_{J_i} is a 0–1 matrix that is obtained from I_n by removing all rows indexed by $\mathbb{N}_n \setminus J_i$, and $\bar{\mathcal{C}}_i$ and $\bar{f}_i$ are lower-dimensional descriptions of $\mathcal{C}_i$ and f_i , given as $\bar{\mathcal{C}}_i = \{s^i \in \mathbb{R}^{|J_i|} \mid E_{J_i}^T s^i \in \mathcal{C}_i\}$ and $\bar{f}_i(s^i) = f_i(E_{J_i}^T s^i)$. The constraints in (5.10c) are referred to as the *consensus* or *consistency* constraints. We can further rewrite this problem as

$$\underset{S}{\text{minimize}} \quad \sum_{i=1}^N \bar{f}_i(s^i) + \mathcal{I}_{\bar{\mathcal{C}}_i}(s^i) + \mathcal{I}_{\mathcal{D}}(S), \quad (5.11)$$

where $S = (s^1, \dots, s^N) \in \mathbb{R}^{\sum_{i=1}^N |J_i|}$ and $\mathcal{D} = \{S \mid S \in \text{col}(\bar{E})\}$ is the so-called *consensus set* with $\bar{E} = \begin{bmatrix} E_{J_1}^T & \cdots & E_{J_N}^T \end{bmatrix}^T$. This problem is in the same format as (5.9). Specifically we can identify $F_1(S) = \sum_{i=1}^N \bar{f}_i(s^i) + \mathcal{I}_{\bar{\mathcal{C}}_i}(s^i)$ and $F_2(S) = \mathcal{I}_{\mathcal{D}}(S)$. We can employ the proximal point algorithm for solving this problem. However, notice that doing so results in an algorithm where each of its iterations is as computationally intensive as solving the problem itself. Consequently and in order to devise more efficient algorithms for solving this problem we harbor to proximal splitting. Firstly, note that based on the definitions of F_1 and F_2 , we have

$$\text{prox}_{\rho F_1}(Y) = \underset{S}{\text{argmin}} \left\{ \sum_{i=1}^N \bar{f}_i(s^i) + \mathcal{I}_{\bar{\mathcal{C}}_i}(s^i) + \frac{1}{2\rho} \|s^i - y^i\|^2 \right\}, \quad (5.12a)$$

$$\text{prox}_{\rho F_2}(Y) = \underset{S}{\text{argmin}} \left\{ \mathcal{I}_{\mathcal{D}}(S) + \frac{1}{2\rho} \|S - Y\|^2 \right\} = P_{\mathcal{D}}(Y) := \bar{E} (\bar{E}^T \bar{E})^{-1} \bar{E}^T Y, \quad (5.12b)$$

where $Y = (y^1, \dots, y^N) \in \mathbb{R}^{\sum_{i=1}^N |J_i|}$. The problem in (5.12a) is separable and hence, the proximity operator for F_1 can be computed by solving N independent sub-

problems, each of which is much cheaper than solving the original problem. Furthermore, note that the proximity operator of the indicator function for the consensus set $\mathcal{D}$ is the linear orthogonal projection onto this set which assures the consistency of the iterates at each iteration. Let us again consider the double-backward algorithm. Applying this algorithm to (5.11) results in

$$S^{(k+1)} = \text{prox}_{\rho F_1}(\bar{E}x^{(k)}), \quad (5.13a)$$

$$x^{(k+1)} = (\bar{E}^T \bar{E})^{-1} \bar{E}^T S^{(k+1)}. \quad (5.13b)$$

Due to the separability of the problem in (5.12a), this iterative algorithm can be implemented over a network of N computational agents. Particularly the first stage of (5.13) describes the independent computations that need to be conducted locally by each agent, i , that is

$$s^{i,(k+1)} = \underset{s^i}{\text{argmin}} \left\{ \bar{f}_i(s^i) + \mathcal{I}_{\bar{C}_i}(s^i) + \frac{1}{2\rho} \|s^i - E_{J_i} x^{(k)}\|^2 \right\}, \quad (5.14)$$

and the second stage describes the communication or collaboration protocol among these agents. Specifically it expresses that all agents i, j that share variables, i.e., when $J_i \cap J_j \neq \emptyset$, should communicate their computed quantities for these variables to one another. Once each agent have received all these quantities from its neighboring agents, $\text{Ne}(i) = \{j \mid J_i \cap J_j \neq \emptyset\}$, it then computes

$$x_j^{(k+1)} = \frac{1}{|\mathcal{I}_j|} \sum_{q \in \mathcal{I}_j} \left(E_{J_q}^T s^{q,(k+1)} \right)_j, \quad (5.15)$$

for all $j \in J_i$. Notice that based on this description of the communication protocol, we see that the computational graph of the resulting algorithm coincides with the coupling graph of the problem. As a result, by applying the double-backward algorithm to the problem in (5.11) we have virtually devised a distributed algorithm for solving this problem that utilizes a network of N computational agents, with the coupling graph of the problem as its computational graph. The spawned distributed algorithms using proximal splitting methods commonly have the same structural properties and they only differ in local computations description and what each agent needs to communicate to its neighbors, see e.g., Parikh and Boyd (2014); Combettes and Pesquet (2011) and references therein. We illustrate the expressed design procedure in Figure 5.4. This technique is used in Paper A for designing distributed solutions for solving coupled convex feasibility problems.

Remark 5.1. Notice that distributed algorithms of this form can be efficiently applied for solving loosely coupled problems. This is because for such problems each agent needs to communicate with only few other agents.

Figure 5.4: Illustration of the design procedure for distributed algorithms based on proximal splitting. The figure reads from top to bottom. It states that given a coupled problem and its coupling graph, as defined in Section 5.1, we can distribute the constituent subproblems among computational agents, and solve the problem distributedly using proximal splitting. As illustrated at the bottom of the figure, the computational graph for the resulting algorithm is the same as the coupling graph of the problem.

In this section we described an approach for devising distributed algorithms that solely relied on the use of proximal point and proximal splitting methods. Notice that these methods only use first order information about the problem for solving it. As a result the algorithms produced using this approach can potentially require many iterations to converge to a highly accurate solution. Furthermore, as can be seen from (5.14), the subproblems that each agent needs to solve at each iteration are in general themselves constrained optimization problems and hence local computations can still be computationally costly. Next we briefly discuss how we can alleviate this issue by devising distributed algorithms that are based on primal and primal-dual interior-point methods.

5.3.2 Distributed Solutions Based on Splitting in Primal and Primal-dual Methods Using Proximal Splitting

Let us reconsider the problem

$$\text{minimize } f_0(x) \quad (5.16a)$$

$$\text{subject to } g_i(x) \leq 0, \quad i = 1, \dots, m, \quad (5.16b)$$

$$Ax = b, \quad (5.16c)$$

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{p \times n}$ and $b \in \mathbb{R}^p$. We can employ primal and primal-dual interior-point methods for solving this problem. Recall that at each iteration of these methods, the search directions are computed by solving a nonsingular system of equations of the form

$$\begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta v \end{bmatrix} = - \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}, \quad (5.17)$$

where $H \in \mathbb{S}_+^n$ and $r_1 \in \mathbb{R}^n$ have different descriptions for primal and primal-dual methods. Notice that this system of equations describes the KKT optimality conditions for the QP

$$\text{minimize } \frac{1}{2} \Delta x^T H \Delta x + r_1^T \Delta x \quad (5.18a)$$

$$\text{subject to } A \Delta x = -r_2, \quad (5.18b)$$

and hence, the solutions to (5.17) can be computed by solving this problem. Now consider the coupled convex problem

$$\text{minimize } f_1(x) + \dots + f_N(x) \quad (5.19a)$$

$$\text{subject to } G^i(x) \leq 0, \quad i = 1, \dots, N, \quad (5.19b)$$

$$A^i x = b^i, \quad i = 1, \dots, N, \quad (5.19c)$$

where $f_i: \mathbb{R}^n \rightarrow \mathbb{R}$, $G^i: \mathbb{R}^n \rightarrow \mathbb{R}^{m_i}$, $b^i \in \mathbb{R}^{p_i}$ and $A^i \in \mathbb{R}^{p_i \times n}$ with $p_i < n$ and $\text{rank}(A^i) = p_i$ for all $i = 1, \dots, N$. As in Section 5.1, we can equivalently rewrite this problem as

$$\text{minimize}_{S, x} \quad \bar{f}_1(s^1) + \dots + \bar{f}_N(s^N) \quad (5.20a)$$

$$\text{subject to } \bar{G}^i(s^i) \leq 0, \quad i = 1, \dots, N, \quad (5.20b)$$

$$\bar{A}^i s^i = b^i, \quad i = 1, \dots, N, \quad (5.20c)$$

$$S = \bar{E}x, \quad (5.20d)$$

where $\bar{A}^i \in \mathbb{R}^{p_i \times |J_i|}$, and the functions $\bar{f}_i$ and $\bar{G}^i$ are lower-dimensional versions of f_i and G^i for $i = 1, \dots, N$. Let us now apply primal and primal-dual methods to this problem. For the sake of notational brevity we here drop the iteration index.

In order to compute the search directions at each iteration of these methods, we would then need to solve the coupled QP

$$\underset{\Delta S, \Delta x}{\text{minimize}} \quad \sum_{i=1}^N \frac{1}{2} (\Delta s^i)^T H^i \Delta s^i + (r^i)^T \Delta s^i - (v_c)^T \bar{E} \Delta x \quad (5.21a)$$

$$\text{subject to} \quad \bar{A}^i \Delta s^i = -r_{\text{primal}}^i, \quad i = 1, \dots, N, \quad (5.21b)$$

$$\Delta S - \bar{E} \Delta x = -r_c, \quad (5.21c)$$

where $r_{\text{primal}}^i = \bar{A}^i s^i - b^i$, $r_c = S - \bar{E}x$, and for primal methods

$$H^i = \nabla^2 \bar{f}_i(s^i) + \sum_{j=1}^{m_i} \left[\frac{1}{t \bar{G}_j^i(s^i)^2} \nabla \bar{G}_j^i(s^i) \nabla \bar{G}_j^i(s^i)^T - \frac{1}{t \bar{G}_j^i(s^i)} \nabla^2 \bar{G}_j^i(s^i) \right],$$

$$r^i = \nabla \bar{f}_i(s^i) - \sum_{j=1}^{m_i} \frac{1}{t \bar{G}_j^i(s^i)} \nabla \bar{G}_j^i(s^i) + (\bar{A}^i)^T v^i + v_c^i,$$

and for primal-dual methods

$$H^i = \nabla^2 \bar{f}_i(s^i) + \sum_{j=1}^{m_i} \lambda_j^i \nabla^2 \bar{G}_j^i(s^i) - \sum_{j=1}^{m_i} \frac{\lambda_j^i}{\bar{G}_j^i(s^i)} \nabla \bar{G}_j^i(s^i) (\nabla \bar{G}_j^i(s^i))^T,$$

$$r^i = \nabla \bar{f}_i(s^i) + \sum_{j=1}^{m_i} \lambda_j^i \nabla \bar{G}_j^i(s^i) + (\bar{A}^i)^T v^i + v_c^i + D \bar{G}^i(s^i) \text{diag}(\bar{G}^i(s^i))^{-1} r_{\text{cent}}^i,$$

with

$$r_{\text{cent}}^i = -\text{diag}(\lambda^i) \bar{G}^i(s^i) - \frac{1}{t} \mathbf{1}.$$

We can employ proximal splitting algorithms, such as alternating direction method of multipliers (ADMM), for solving the problem in (5.21). This allows us to devise distributed optimization algorithms based on primal or primal-dual methods for solving coupled problems. Notice that the coupling structure for (5.20) and (5.21) are the same. Consequently, as for the distributed algorithms discussed in Section 5.3.1, the computational graphs for these algorithms are also the same as the coupling graph of the original problem. Furthermore, notice that the local subproblems that need to be solved by each agent are all equality constrained QPs. These subproblems are in general much simpler to solve than the subproblems for the previous approach, as they were general constrained problems. Hence the distributed algorithms generated using this approach, potentially, enjoy much better computational properties. This approach for designing distributed algorithms is illustrated in Figure 5.5, and it is discussed in full detail in Paper B.

Figure 5.5: Illustration of the design procedure for distributed algorithms based on primal or primal-dual methods using proximal splitting. The figure reads from top to bottom. It states that given a coupled problem and its coupling graph, as defined in Section 5.1, we first apply a primal or primal-dual interior-point to the problem. Using proximal splitting, we can then distribute the computations, at every iteration, among computational agents and solve the problem distributedly. As illustrated at the bottom of the figure, the computational graph for the resulting algorithm is the same as the coupling graph of the problem.

So far we have discussed two major approaches for devising distributed algorithms, namely one based on proximal splitting methods and one as a combination of primal or primal-dual methods and proximal splitting algorithms. Algorithms generated from both of these approaches utilize proximal splitting

Figure 5.6: Illustration of extraction of tree structure in a sparsity graph.

methods for distributing the computations. This means that such algorithms potentially require many iterations to converge. Next, we show that in case there exist a certain additional structure in the problem, we can devise distributed algorithms that are more efficient than the previous ones.

5.3.3 Distributed Solutions Based on Splitting in Primal and Primal-dual Methods Using Message Passing

Let us now reconsider the example in (5.2). Note that the sparsity graph for this problem can be represented using a tree, or in other words, the sparsity graph has an inherent tree structure. This is illustrated in Figure 5.6. This type of structure is not a general property of coupled problems. However, it is somewhat common, see Chapter 6 and Paper F for some examples. Let us assume that the problem in (5.19) has this type of coupling structure. This problem can be equivalently rewritten as

$$\underset{x}{\text{minimize}} \quad \bar{f}_1(E_{J_1}x) + \cdots + \bar{f}_N(E_{J_N}x), \quad (5.22a)$$

$$\text{subject to} \quad \bar{G}^i(E_{J_i}x) \leq 0, \quad i = 1, \dots, N, \quad (5.22b)$$

$$\bar{A}^i E_{J_i}x = b^i, \quad i = 1, \dots, N, \quad (5.22c)$$

which has the same coupling structure as the original problem. Applying primal or primal-dual interior-point methods for solving this problem, require solving

$$\underset{\Delta x}{\text{minimize}} \quad \sum_{i=1}^N \frac{1}{2} (E_{J_i} \Delta x)^T H^i E_{J_i} \Delta x + (r^i)^T E_{J_i} \Delta x \quad (5.23a)$$

$$\text{subject to} \quad \bar{A}^i E_{J_i} \Delta x = -r_{\text{primal}}^i, \quad i = 1, \dots, N, \quad (5.23b)$$

at each iteration for computing the search directions. Note that for primal methods

$$r^i = \nabla \bar{f}_i(x_{J_i}) - \sum_{j=1}^{m_i} \frac{1}{t \bar{G}_j^i(x_{J_i})} \nabla \bar{G}_j^i(x_{J_i}) + (\bar{A}^i)^T v^i,$$

with $x_{J_i} = E_{J_i} x$, and for primal-dual methods

$$r^i = \nabla \tilde{f}_i(x_{J_i}) + \sum_{j=1}^{m_i} \lambda_j^i \nabla \tilde{G}_j^i(x_{J_i}) + (\bar{A}^i)^T v^i + D \tilde{G}^i(x_{J_i}) \text{diag} \left(\tilde{G}^i(x_{J_i}) \right)^{-1} r_{\text{cent}}^i,$$

with

$$r_{\text{cent}}^i = -\text{diag}(\lambda^i) \tilde{G}^i(x_{J_i}) - \frac{1}{t} \mathbf{1}.$$

The coupling structure in (5.22) is also reflected in this problem. Considering this observation, we can employ a more efficient technique, called message passing (Koller and Friedman, 2009), for computing the search directions within primal or primal-dual methods iterations. Unlike proximal splitting methods which solve for search directions iteratively, this technique computes these directions by performing a recursion over the inherent tree structure in the sparsity graph of the problem. This approach for devising distributed algorithms is illustrated in Figure 5.7, and is fully discussed in paper C. Notice that the distributed algorithms generated using this approach use the inherent tree structure in the sparsity graph as their computational graph. This is in contrast to the algorithms generated using the previous two approaches which utilize the coupling graph as their computational graph.

Having described coupled problems and algorithms to solve them distributedly, we present applications for these algorithms within different areas in the next chapter.

Figure 5.7: Illustration of the design procedure for distributed algorithms based on primal or primal-dual methods using message passing. The figure reads from top to bottom. It states that given a coupled problem and its sparsity graph, as defined in Section 5.1, we first apply a primal or primal-dual interior-point to the problem. Using message passing, we can then distribute the computations, at every iteration, among computational agents and solve the problem distributedly. As illustrated at the bottom of the figure, the computational graph for the resulting algorithm is a clique tree of the sparsity graph.

6

Examples in Control and Estimation

In order to solve large-scale robustness analysis problems, in papers D–F we first showed how to reformulate the problem as an equivalent coupled problem, and then employed efficient algorithms to solve the coupled problem distributedly. A similar procedure can also be used for solving large-scale optimization problems that appear in other fields. To illustrate this, we consider two examples in control and estimation, and we will discuss how to reformulate and exploit structure in these problems so as to enable the use of efficient distributed solvers. Particularly, in Section 6.1 we consider a platooning problem and in Section 6.2 we consider a sensor network localization problem.

6.1 Distributed Predictive Control of Platoons of Vehicles

In this section, we consider a vehicle platooning problem. Here, a platoon refers to a group of vehicles that are moving at close inter-vehicular distances, commonly, in a chain. The platooning problem then corresponds to the problem of devising control strategies that enable the vehicles to safely operate at such close distances so that the platoon moves with a speed that is close to a given reference value, see e.g., Turri et al. (2014); Dunbar and Caveney (2012); di Bernardo et al. (2015); Dold and Stursberg (2009). Here we use model predictive control (MPC) for solving this control problem. Note that a centralized control strategy for platooning is not suitable, since such a solution would require high amount of communications, has a single point of failure, and does not accommodate addition and removal of vehicles from the platoon. This is because such an action would require reformulating the whole problem. Next we describe how to devise a distributed solution for this problem. But let us first review some basic

concepts relating to MPC.

6.1.1 Linear Quadratic Model Predictive Control

Let us assume that the discrete-time dynamics of the system to be controlled is given by

$$x(k+1) = Ax(k) + Bu(k) + d(k), \quad (6.1)$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$, $d(k) \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$. Here $d(k)$ represents a measurable disturbance. In a linear quadratic control framework, given the initial state x_0 , we would like to compute a control sequence that is the minimizing argument for the following problem

$$\text{minimize } \sum_{k=0}^{\infty} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix}^T \mathbf{Q} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} + q^T x(k) + r^T u(k) \quad (6.2a)$$

$$\text{subject to } x(k+1) = Ax(k) + Bu(k) + d(k) \quad k = 0, 1, \dots, \quad (6.2b)$$

$$C_x x(k) + C_u u(k) \leq b, \quad k = 0, 1, \dots, \quad (6.2c)$$

$$x(0) = x_0, \quad (6.2d)$$

where $C_x \in \mathbb{R}^{p \times n}$, $C_u \in \mathbb{R}^{p \times m}$, $\mathbf{Q} = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0$, with $Q \in \mathbb{S}_+^n$ and $R \in \mathbb{S}_{++}^m$, $q \in \mathbb{R}^n$, $r \in \mathbb{R}^m$ and $b \in \mathbb{R}^p$. In this problem $u(0), u(1), \dots$ and $x(0), x(1), \dots$ are the optimization variables which are referred to as the control and state sequences, respectively. Here the matrix $\mathbf{Q}$ and the vectors q, r , that define the cost function, are commonly chosen to reflect our control specifications, the constraints in (6.2b) describe the dynamic behavior of the system and the constraints in (6.2c) express the feasible operating region of the system. This control strategy is referred to as infinite-horizon control, and as can be seen from (6.2), defines an infinite-dimensional optimization problem. One common way to approximately solve this problem is by using a suboptimal heuristic called model predictive control (MPC). In this heuristic method the horizon of the control problem is truncated to a finite value, H_p , and a receding horizon strategy is used to mimic an infinite horizon, (Rawlings, 2000; Maciejowski, 2002). This means that at each time instant t and given the state of the system, $x(t)$, we solve the following finite-horizon quadratic control problem

$$\text{minimize } \sum_{k=0}^{H_p-1} \begin{bmatrix} \bar{x}(k) \\ \bar{u}(k) \end{bmatrix}^T \mathbf{Q} \begin{bmatrix} \bar{x}(k) \\ \bar{u}(k) \end{bmatrix} + q^T \bar{x}(k) + r^T \bar{u}(k) \quad (6.3a)$$

$$+ \bar{x}(H_p)^T Q_t \bar{x}(H_p) + q_t^T \bar{x}(H_p) \quad (6.3b)$$

$$\text{subject to } \bar{x}(k+1) = A\bar{x}(k) + B\bar{u}(k) + d(k), \quad k = 0, 1, \dots, H_p - 1 \quad (6.3c)$$

$$C_x \bar{x}(k) + C_u \bar{u}(k) \leq b, \quad k = 1, \dots, H_p - 1 \quad (6.3d)$$

$$C_t \bar{x}(H_p) \leq b_t, \quad C_0 \bar{u}(0) \leq b_0 \quad (6.3e)$$

$$\bar{x}(0) = x(t), \quad (6.3f)$$

where $\bar{u}(0), \dots, \bar{u}(H_p - 1), \bar{x}(0), \dots, \bar{x}(H_p)$ are the optimization variables, $C_t \in \mathbb{R}^{p_t \times n}$, $C_0 \in \mathbb{R}^{q_0 \times m}$, $Q_t \in \mathbb{S}_+^n$, $b_t \in \mathbb{R}^{p_t}$ and $b_0 \in \mathbb{R}^{q_0}$.

Having computed the optimal solution $\bar{u}^*(0), \dots, \bar{u}^*(H_p - 1)$, and $\bar{x}^*(1), \dots, \bar{x}^*(H_p)$, the MPC controller chooses $\bar{u}^*(0)$ as the next control input to the system, i.e., $u(t) = \bar{u}^*(0)$. This procedure is then repeated at time $t + 1$, this time with starting point $x(t + 1)$. The matrices and vectors C_t, b_t, Q_t and q_t describe the so-called terminal cost and terminal constraints, and are commonly designed to assure stability and recursive feasibility, (Maciejowski, 2002; Rawlings, 2000). Here, we do not discuss the design choice for these matrices, and instead we focus on the underlying structure of the optimization problem for our application. Next we describe the use of MPC for the platooning application.

6.1.2 MPC for Platooning

It is possible to address the platooning problem using MPC. To this end and in order to express the underlying optimization problem, we describe its different building blocks. We start with the constraints in the problem.

Dynamic Model

We consider the following model for each vehicle i in the platoon

$$\bar{p}^i(k + 1) = \bar{p}^i(k) + T_s \bar{v}^i(k), \quad (6.4a)$$

$$\bar{v}^i(k + 1) = \bar{v}^i(k) + T_s (F_e^i(k) - F_b^i(k) + F_{\text{ext}}^i(k)), \quad (6.4b)$$

where T_s is the sampling time, $\bar{p}^i$ and $\bar{v}^i$ denote the position and velocity of the i^{th} vehicle, respectively, F_e^i and F_b^i are the engine and braking forces scaled with the mass of the vehicle. Also F_{ext}^i represents the known disturbance on the vehicle that summarizes the forces from gravity, roll resistance and aerodynamic resistance, (Turri et al., 2014). Here, we assume that this disturbance is constant throughout the prediction horizon. The model in the MPC problem can then be written as

$$\bar{x}^i(k + 1) = A^i \bar{x}^i(k) + B^i \bar{u}^i(k) + d^i \quad (6.5)$$

with $\bar{x}^i(k) = (\bar{p}^i(k), \bar{v}^i(k))$, $\bar{u}^i(k) = (F_e^i(k), F_b^i(k))$ and $d^i = (0, T_s F_{\text{ext}}^i)$.

Constraints Defining the Feasible Operating Area

The first set of constraints we discuss here, describes the limits on the delivered engine and braking forces for each vehicle i and is given as

$$0 \leq F_e^i(k) \leq F_{e, \max}^i, \quad (6.6a)$$

$$0 \leq F_b^i(k) \leq F_{b, \max}^i. \quad (6.6b)$$

Figure 6.1: The figure illustrates a platooning problem, where the platoon is to follow a given a reference speed v_r . This should be achieved while each vehicle keeps a desired distance, d_i , to its leading vehicle. In this figure, the length of each vehicle i is denoted by l_i .

Let us compactly denote this set of constraints as $\bar{u}^i(k) \in \mathcal{U}^i$ for some set $\mathcal{U}^i$. The next set of constraints incorporates the speed limits in the problem and is given by

$$v_{\min}^i \leq v^i(k) \leq v_{\max}^i, \quad (6.7)$$

which we compactly denote as $\bar{x}^i(k) \in \mathcal{X}^i$, for some set $\mathcal{X}^i$. The last set of constraints in the problem assures that each vehicle in the platoon stays a safe distance away from the neighboring vehicles, and in the worst case can brake and stop without colliding with the front vehicle. This constraint for each vehicle, i , can be written as

$$p^i(t_d) - \frac{(v^i(t_d))^2}{-2a_{b,\min}} \leq p^{i-1}(0) - \frac{(v^i(0))^2}{-2a_{b,\max}} - l_{i-1}, \quad (6.8)$$

with $t_d = T_d/T_s$ and l_i is the length of the i^{th} vehicle. Here we have assumed that the reaction time-delay, T_d , is an integer multiple of T_s , and $a_{b,\min}, a_{b,\max} \geq 0$ are the minimum and maximum achievable braking accelerations, respectively, (Turri et al., 2014). This constraint is denoted as $q_s(x^i(t_d)) \leq 0$. Having defined the constraints in the problem, we next discuss the cost function of the optimization problem.

Cost Function

The goal of each vehicle in a platoon is to move with a given reference speed, i.e., the speed of the first vehicle in the platoon, and keep a given distance to the vehicle in the front with least amount of brakes and engine use. This is illustrated in Figure 6.1. Then for each vehicle in the platoon, except for the first vehicle, the cost function that reflects these criteria can be written as

$$\begin{aligned} J^i(\bar{p}^i(k), \bar{p}^{i-1}(k), \bar{v}^i(k), \bar{v}^{(i-1)}(k), \bar{u}^i(k)) = \\ \frac{1}{2} \left[\sigma^i \times (\bar{p}^{i-1}(k) - \bar{p}^i(k) - l_{i-1} - d_{i-1})^2 \right. \\ \left. + \delta^i \times (\bar{v}^i(k) - \bar{v}^{i-1}(k))^2 + (\bar{u}^i(k))^T R^i \bar{u}^i(k) \right], \quad (6.9) \end{aligned}$$

Figure 6.2: Coupling and sparsity graphs of the platooning problem depicted in the top and bottom figures, respectively. The clusters in the sparsity graph mark the variables for each subproblem.

where d_{i-1} denotes the desired distance between the $(i-1)^{\text{th}}$ and i^{th} vehicles, and $\sigma^i, \delta^i > 0$, and $R^i > 0$ are design parameters. The cost function for the first vehicle can also be written similarly as

$$J^1(\bar{p}^1(k), \bar{v}^1(k), \bar{u}^1(k)) = \frac{1}{2} \left[\delta^1 \times (\bar{v}^1(k) - v_r(k))^2 + (\bar{u}^1(k))^T R^1 \bar{u}^1(k) \right], \quad (6.10)$$

where v_r is the reference speed for the platoon. Having defined these local cost functions, we can now describe the MPC problem for platooning. This is given as

$$\text{minimize} \quad \sum_{k=0}^{H_p-1} \left[J^1(\bar{x}^1(k), \bar{u}^1(k)) + \sum_{i=2}^N J^i(\bar{x}^i(k), \bar{x}^{i-1}(k), \bar{u}^i(k)) \right] \quad (6.11a)$$

$$\text{subject to} \quad \bar{x}^i(k+1) = A^i \bar{x}^i(k) + B^i \bar{u}^i(k) + d^i, \quad i = 1, \dots, N, \quad k = 0, \dots, H_p - 1, \quad (6.11b)$$

$$\bar{x}^i(k) \in \mathcal{X}^i, \quad i = 1, \dots, N, \quad k = 1, \dots, H_p - 1, \quad (6.11c)$$

$$\bar{u}^i(k) \in \mathcal{U}^i, \quad i = 1, \dots, N, \quad k = 0, \dots, H_p - 1, \quad (6.11d)$$

$$q_s(\bar{x}^i(t_d)) \leq 0, \quad i = 2, \dots, N, \quad (6.11e)$$

$$\bar{x}^i(0) = x^i(t), \quad i = 1, \dots, N, \quad (6.11f)$$

with $x^i(t)$ denoting the state of the i^{th} vehicle at time t , which is a coupled problem. The coupling and sparsity graphs for this problem are depicted in Figure 6.2. For the sparsity graph illustrated in the figure it is assumed that $H_p = 2$. Notice that the coupling graph of the problem has the same structure as the platoon itself. This means that this problem can be solved distributedly using algorithms generated from the approaches discussed in sections 5.3.1 and 5.3.2, over the platoon, where each vehicle then takes the role of a computational agent.

Also it is possible to establish that the sparsity graph for this problem is chordal, with a clique tree that has the same structure as the coupling graph of the prob-

Figure 6.3: Localization problem with $N = 7$ sensors, marked with large dots, and $m = 6$ anchors. An edge between two sensors shows the availability of range measurement between the two.

lem. As a result, we can also use the algorithm in Section 5.3.3 for solving this problem over the platoon of the vehicles. Consequently, we can devise distributed algorithms for solving the platooning problem using any of the approaches discussed in Section 5.3, and all the resulting algorithms use the platoon structure as their computational graph.

6.2 Distributed Localization of Scattered Sensor Networks

In this section we discuss a localization problem in sensor networks. Let us assume that we are given a network of N sensors distributed in an area, and in presence of m anchors. The localization problem then concerns the estimation of unknown sensor positions, x_s^i , using inter-sensor and sensor-anchor distance measurements. Here we assume that the position of the anchors, x_a^i , are known. Next we describe how this problem can be formulated as a coupled optimization problem that can be solved using the algorithms discussed in Section 5.3. Note that similar to the previous case a centralized solution may not be a viable choice for solving this problem. This is because, aside from the huge communications demand, for large populations of sensors solving the problem could be computationally intractable. Let us start by describing a localization problem in more details.

6.2.1 A Localization Problem Over Sensor Networks

Let us assume that the sensors in the network are capable of performing computations and some can measure their distance to certain other sensors and some

of the anchors. We assume that if sensor i can measure its distance to sensor j , this measurement is also available for sensor j . This then allows us to describe the range measurement availability among sensors using an undirected graph $G_r(V_r, \mathcal{E}_r)$ with vertex set $V_r = \{1, \dots, N\}$ and edge set $\mathcal{E}_r$. An edge $(i, j) \in \mathcal{E}_r$ if and only if a range measurement between sensors i and j is available, see Figure 6.3. Here, we assume that $G_r(V_r, \mathcal{E}_r)$ is connected. Let us define the set of neighbors of each sensor i , $\text{Ne}_r(i)$, as the set of sensors to which this sensor has an available range measurement. In a similar fashion let us denote the set of anchors to which sensor i can measure its distance to by $\text{Ne}_a(i) \subseteq \{1, \dots, m\}$. We can describe the inter-sensor range measurements for each sensor i as

$$R_{ij} = \mathcal{D}_{ij} + E_{ij}, \quad j \in \text{Ne}_r(i), \quad (6.12)$$

where $\mathcal{D}_{ij} = \|x_s^i - x_s^j\|_2$ defines the sensor distance matrix and E_{ij} is the inter-sensor measurement noise matrix. Notice that the sparsity pattern of $\mathcal{D}$ is defined by the graph $G_r(V_r, \mathcal{E}_r)$. Similarly we can describe the anchor range measurements for each sensor i as

$$Y_{ij} = \mathcal{Z}_{ij} + V_{ij}, \quad j \in \text{Ne}_a(i), \quad (6.13)$$

where $\mathcal{Z}_{ij} = \|x_s^i - x_a^j\|_2$ defines the anchor-sensor distance matrix and V_{ij} is the anchor-sensor measurement noise matrix. We assume that the inter-sensor and anchor-sensor measurement noises are independent and that $E_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$ and $V_{ij} \sim \mathcal{N}(0, \delta_{ij}^2)$.

Having defined the setup of the sensor network, we can write the localization problem in a maximum likelihood setting as

$$X_{\text{ML}}^* = \underset{X}{\operatorname{argmax}} \left\{ \sum_{i=1}^N \left(\sum_{j \in \text{Ne}_r(i)} \frac{1}{\sigma_{ij}^2} \left(\mathcal{D}_{ij}(x_s^i, x_s^j) - R_{ij} \right)^2 + \sum_{j \in \text{Ne}_a(i)} \frac{1}{\delta_{ij}^2} \left(\mathcal{Z}_{ij}(x_s^i, x_a^j) - Y_{ij} \right)^2 \right) \right\}, \quad (6.14)$$

where $X = [x_s^1 \ \dots \ x_s^N] \in \mathbb{R}^{d \times N}$ with $d = 2$ or $d = 3$. This problem can be formulated as a constrained optimization problem, as was described in Simonetto and Leus (2014). Let us define matrices Λ and Ξ such that $\Lambda_{ij} = \mathcal{D}_{ij}^2$ and $\Xi_{ij} = \mathcal{Z}_{ij}^2$. Then the problem in (6.14) can be equivalently rewritten as the following constrained optimization problem

$$\begin{aligned} \underset{X, S, \Lambda, \Xi, D, Z}{\text{minimize}} \quad & \sum_{i=1}^N \left(\sum_{j \in \text{Ne}_r(i)} \frac{1}{\sigma_{ij}^2} (\Lambda_{ij} - 2D_{ij}R_{ij} + R_{ij}^2) \right. \\ & \left. + \sum_{j \in \text{Ne}_a(i)} \frac{1}{\delta_{ij}^2} (\Xi_{ij} - 2Z_{ij}Y_{ij} + Y_{ij}^2) \right) \end{aligned} \quad (6.15a)$$

subject to

$$\left. \begin{aligned} S_{ii} + S_{jj} - 2S_{ij} &= \Lambda_{ij} \\ \Lambda_{ij} &= D_{ij}^2, \quad D_{ij} \geq 0, \quad j \in \text{Ne}_r(i) \end{aligned} \right\}, \quad i \in \mathbb{N}_N, \quad (6.15b)$$

$$\left. \begin{aligned} S_{ii} - 2(x_s^i)^T x_a^j + \|x_a^j\|_2^2 &= \Xi_{ij} \\ \Xi_{ij} &= Z_{ij}^2, \quad Z_{ij} \geq 0, \quad j \in \text{Ne}_a(i) \end{aligned} \right\}, \quad i \in \mathbb{N}_N, \quad (6.15c)$$

$$S = X^T X. \quad (6.15d)$$

So far we have described a way to formulate the localization problem over general sensor networks as a constrained optimization problem. In this section, however, we are particularly interested in localization of scattered sensor networks which corresponds to additional assumptions on the graph $G_r(V_r, \mathcal{E}_r)$.

6.2.2 Localization of Scattered Sensor Networks

Let us now assume that the graph $G_r(V_r, \mathcal{E}_r)$ is chordal or that it is possible to compute its chordal embedding by adding only a few edges. Furthermore, given the set of its cliques $\mathbf{C}_{G_r} = \{C_1, \dots, C_l\}$, we assume that

- $|C_i| \ll N$;
- $|C_i \cap C_j| \forall i \neq j$ are small.

Note that the assumptions above imply that each sensor i has access to range measurements only from a few other sensors. We refer to such sensor networks as scattered. The localization problem of scattered sensor networks can also be formulated as a constrained optimization problem using the approach discussed in Section 6.2.1. However, the formulation of the problem in (6.15) is not fully representative of the structure in the problem. In order to exploit the structure in our localization problem we modify (6.15), and equivalently rewrite it as

$$\begin{aligned} \underset{X, S, \Lambda, \Xi, D, Z}{\text{minimize}} \quad & \sum_{i=1}^N \left(\sum_{j \in \text{Ne}_r(i)} \frac{1}{\sigma_{ij}^2} (\Lambda_{ij} - 2D_{ij}R_{ij} + R_{ij}^2) \right. \\ & \left. + \sum_{j \in \text{Ne}_a(i)} \frac{1}{\delta_{ij}^2} (\Xi_{ij} - 2Z_{ij}Y_{ij} + Y_{ij}^2) \right) \end{aligned} \quad (6.16a)$$

subject to

$$\left. \begin{aligned} S_{ii} + S_{jj} - 2S_{ij} &= \Lambda_{ij} \\ \Lambda_{ij} &= D_{ij}^2, \quad D_{ij} \geq 0, \quad j \in \text{Ne}_r(i) \end{aligned} \right\}, \quad i \in \mathbb{N}_N, \quad (6.16b)$$

$$\left. \begin{aligned} S_{ii} - 2(x_s^i)^T x_a^j + \|x_a^j\|_2^2 &= \Xi_{ij} \\ \Xi_{ij} &= Z_{ij}^2, \quad Z_{ij} \geq 0, \quad j \in \text{Ne}_a(i) \end{aligned} \right\}, \quad i \in \mathbb{N}_N, \quad (6.16c)$$

$$S \geq 0, \quad S_{ij} = (x_s^i)^T x_s^j, \quad \forall (i, j) \in \mathcal{E}_r \cup \{(i, i) \mid i \in V_r\}. \quad (6.16d)$$

Note that, here, we have modified the constraint in (6.15d) so that the structure in the problem is more explicit. This modification is based on the observation that not all the elements of S are used in (6.15b) and (6.15c), and hence we only have to specify the ones that are needed and leave the rest free. As we will discuss in the next section, the structure in this problem enables us to use domain-space decomposition for reformulating the problem to facilitate the use of efficient distributed solvers.

6.2.3 Decomposition and Convex Formulation of Localization of Scattered Sensor Networks

Consider the inter-sensor measurement graph $G_r(V_r, \mathcal{E}_r)$, and assume that this graph is chordal. In case $G_r(V_r, \mathcal{E}_r)$ is not chordal the upcoming discussions hold for any of its chordal embeddings. Let $\mathbf{C}_{G_r} = \{C_1, \dots, C_l\}$ and $T(V_r, \mathcal{E}_r)$ be the clique tree over the cliques. Based on the discussion in Section 5.2.3, then for the problem in (6.16) we have $S \in \mathbb{S}_{G_m}^N$. Hence, we can rewrite (6.16) as

$$\begin{aligned} \underset{X, S_{C_i C_i}, \Lambda, \Xi, D, Z}{\text{minimize}} \quad & \sum_{i=1}^N \left(\sum_{j \in \text{Ne}_r(i)} \frac{1}{\sigma_{ij}^2} (\Lambda_{ij} - 2D_{ij}R_{ij} + R_{ij}^2) \right. \\ & \left. + \sum_{j \in \text{Ne}_a(i)} \frac{1}{\delta_{ij}^2} (\Xi_{ij} - 2Z_{ij}Y_{ij} + Y_{ij}^2) \right) \end{aligned} \quad (6.17a)$$

subject to

$$\left. \begin{aligned} S_{ii} + S_{jj} - 2S_{ij} &= \Lambda_{ij} \\ \Lambda_{ij} &= D_{ij}^2, \quad D_{ij} \geq 0, \quad j \in \text{Ne}_r(i) \end{aligned} \right\}, \quad i \in \mathbb{N}_N, \quad (6.17b)$$

$$\left. \begin{aligned} S_{ii} - 2(x_s^i)^T x_a^j + \|x_a^j\|_2^2 &= \Xi_{ij} \\ \Xi_{ij} &= Z_{ij}^2, \quad Z_{ij} \geq 0, \quad j \in \text{Ne}_a(i) \end{aligned} \right\}, \quad i \in \mathbb{N}_N, \quad (6.17c)$$

$$S_{C_i C_i} \geq 0, \quad S_{C_i C_i} = E_{C_i} X^T X E_{C_i}^T, \quad i \in \mathbb{N}_l, \quad (6.17d)$$

Notice that even though the cost function for this problem is convex, the constraints in (6.17b)–(6.17d) are non-convex and hence the problem is non-convex. Consequently, we next address the localization problem by considering a convex relaxation of this problem. This allows us to solve the localization problem approximately.

One of the ways to convexify the problem in (6.17) is to relax the quadratic equality constraints in (6.17b)–(6.17d) using Schur complements, which results in

$$\underset{X, S_{C_i C_i}, \Lambda, \Xi, D, Z}{\text{minimize}} \quad \sum_{i=1}^N \left(\sum_{j \in \text{Ne}_r(i)} f_{ij}(\Lambda_{ij}, D_{ij}) + \sum_{j \in \text{Ne}_a(i)} g_{ij}(\Xi_{ij}, Z_{ij}) \right) \quad (6.18a)$$

subject to

$$(S_{ii}, S_{jj}, S_{ij}, \Lambda_{ij}, D_{ij}) \in \Omega_{ij}, \quad (i, j) \in \mathcal{E}_r, \quad (6.18b)$$

$$(S_{ii}, x_s^i, \Xi_{ij}, Z_{ij}) \in \Theta_{ij}, \quad j \in \text{Ne}_a(i), \quad i \in \mathbb{N}_N, \quad (6.18c)$$

$$\begin{bmatrix} I & X E_{C_i}^T \\ E_{C_i} X^T & S_{C_i C_i} \end{bmatrix} \geq 0, \quad i \in \mathbb{N}_l, \quad (6.18d)$$

where

$$f_{ij}(\Lambda_{ij}, D_{ij}) = \frac{1}{\sigma_{ij}^2} (\Lambda_{ij} - 2D_{ij}R_{ij} + R_{ij}^2),$$

$$g_{ij}(\Xi_{ij}, Z_{ij}) = \frac{1}{\delta_{ij}^2} (\Xi_{ij} - 2Z_{ij}Y_{ij} + Y_{ij}^2),$$

and

$$\begin{aligned} \Omega_{ij} = & \left\{ (S_{ii}, S_{jj}, S_{ij}, \Lambda_{ij}, D_{ij}) \mid S_{ii} + S_{jj} - 2S_{ij} = \Lambda_{ij}, \right. \\ & \left. \begin{bmatrix} 1 & D_{ij} \\ D_{ij} & \Lambda_{ij} \end{bmatrix} \geq 0, D_{ij} \geq 0 \right\}, \\ \Theta_{ij} = & \left\{ (S_{ii}, x_s^i, \Xi_{ij}, Z_{ij}) \mid S_{ii} - 2(x_s^i)^T x_a^j + \|x_a^j\|_2^2 = \Xi_{ij}, \right. \\ & \left. \begin{bmatrix} 1 & Z_{ij} \\ Z_{ij} & \Xi_{ij} \end{bmatrix} \geq 0, Z_{ij} \geq 0, j \in \text{Ne}_a(i) \right\}. \end{aligned}$$

This problem is a coupled SDP and can be solved distributedly using l computational agents. In order to see this with more ease, let us introduce a grouping

of the cost function terms and constraints in (6.18a)–(6.18c). To this end we first describe a set of assignment rules. It is possible to assign

1. the constraint $(S_{ii}, S_{jj}, S_{ij}, \Lambda_{ij}, D_{ij}) \in \Omega_{ij}$ and the cost function term f_{ij} to agent k if $(i, j) \in C_k \times C_k$;
2. the set of constraints $(S_{ii}, x_s^i, \Xi_{ij}, Z_{ij}) \in \Theta_{ij}$, $j \in \text{Ne}_a(i)$ and the cost function terms g_{ij} , $j \in \text{Ne}_a(i)$ to agent k if $i \in C_k$.

We denote the indices of the constraints and cost function terms assigned to agent k through Rule 1 above as ϕ_k , and similarly we denote the set of constraints and cost function terms that are assigned to agent k through Rule 2 by $\bar{\phi}_k$. Using the mentioned rules and the defined notations, we can now group the constraints and the cost function terms and rewrite the problem in (6.18) as

$$\underset{X, S_{C_k C_k}, \Lambda, \Xi, D, Z}{\text{minimize}} \quad \sum_{k=1}^l \left[\sum_{(i,j) \in \phi_k} f_{ij}(\Lambda_{ij}, D_{ij}) + \sum_{i \in \bar{\phi}_k} \sum_{j \in \text{Ne}_a(i)} g_{ij}(\Xi_{ij}, Z_{ij}) \right] \quad (6.19a)$$

subject to

$$\left. \begin{aligned} &(S_{ii}, S_{jj}, S_{ij}, \Lambda_{ij}, D_{ij}) \in \Omega_{ij}, \quad (i, j) \in \phi_k \\ &(S_{ii}, x_s^i, \Xi_{ij}, Z_{ij}) \in \Theta_{ij}, \quad j \in \text{Ne}_a(i) \quad i \in \bar{\phi}_k \\ &\begin{bmatrix} I & X E_{C_k}^T \\ E_{C_k} X^T & S_{C_k C_k} \end{bmatrix} \geq 0 \end{aligned} \right\}, \quad k \in \mathbb{N}_l, \quad (6.19b)$$

Notice that this problem can now be seen as a combination of l coupled subproblems, each defined by a term in the cost function together with its corresponding set of constraints in (6.19b). It is possible to employ distributed optimization algorithms generated using the approaches discussed in sections 5.3.1 and 5.3.2 for solving this coupled problem. The computational graphs for these algorithms are a clique tree of $G_r(V_r, \mathcal{E}_r)$. It is also possible to show that the sparsity graph of this problem is chordal with l cliques, and its corresponding clique tree has the same structure as that of $G_r(V_r, \mathcal{E}_r)$. This enables us to also use the approach in Section 5.3.3 for designing distributed solutions for (6.19) that share the same computational graph as that of the other algorithms.

In this section, we showed that the algorithms produced using the approaches described in Chapter 5, can be used for solving problems in different disciplines. In the next chapter, we finish Part I of the thesis with some concluding remarks.

7

Conclusions and Future Work

The research conducted towards this thesis was driven by robustness analysis problems of large-scale uncertain systems. As was discussed in the introduction, performing such analysis commonly requires solving large-scale optimization problems. These problems are often either impossible or too time-consuming to solve. This could be due to the sheer size of the problem and/or due to certain structural constraints. In order to address the associated computational challenges, we firstly exploited the inherent structure in such problems, which then allowed us to employ efficient centralized optimization algorithms for solving these problems. This enabled us to handle large analysis problems with much less computational time, see papers D and G. The structure exploitation also paved the way for decomposing the problem and formulating it as a coupled optimization problem. This in turn enabled us to utilize or devise tailored distributed optimization algorithms for solving the decomposed problem. The use of distributed algorithms extended our ability to handle very large analysis problems and also to address certain structural constraints, such as privacy, see Paper E.

In order to improve the efficiency of the proposed distributed analysis algorithm, we then turned our focus towards devising more efficient distributed optimization algorithms. This led to the papers A–C and F, see Chapter 5 for an overview of these algorithms. Although the proposed algorithms in these papers were initially devised for solving decomposed analysis problems, they can also be used for solving general coupled optimization problems that appear in other engineering fields. We touched upon this in Chapter 6.

7.1 Future Work

In this thesis we utilized a two phase approach for devising distributed algorithms for solving coupled optimization problems. The first phase concerned the reformulation or decomposition of the problem and the second phase involved devising efficient algorithms for solving the decomposed problem distributedly. It is possible to improve the efficiency and extend the application domain of the resulting distributed algorithms by wisely tweaking flexibilities in each of these phases. We next describe some examples of such improvements.

Let us start by focusing on the first phase of the approach. The outcome of this phase is commonly a decomposed or coupled optimization problem. The constituent subproblems of the coupled problem can have different sizes. Recall that at every iteration of distributed algorithms, each agent needs to solve a local problem that is based on its assigned subproblem. As a result, the computational burden on agents with disproportionately bigger subproblems can be potentially significantly higher than for the other agents. Furthermore, as a consequence, iterate updates for these agents can take considerably longer time than the other agents. The unbalanced size of subproblems can hence introduce latencies and can adversely affect the runtime and efficiency of the proposed algorithms. Consequently, by designing reformulation strategies that produce coupled problems with balanced subproblems sizes, we can avoid the mentioned issues and potentially improve the computational and convergence properties of the proposed algorithms. The reformulation procedure can also be modified such that the coupling or sparsity graph of the resulting coupled problem better represent physical coupling structure in the problem. This then results in distributed algorithms with more sensible computational graphs.

Currently the proposed distributed optimization algorithms can be used for solving convex coupled optimization problems. This limits the scope of problems that can be solved using these algorithms. As a result a viable research avenue would be to investigate the possibility of extending the application of the proposed algorithms to non-convex problems. This, for instance, can be done by combining the approaches discussed in sections 5.3.2 and 5.3.3 with general interior-point methods for nonlinear optimization problems.

In this thesis, we mainly focused on applications related to robustness analysis of uncertain systems. A clear extension of applicability of the proposed algorithms is to use them for robust control synthesis for large-scale uncertain systems, e.g., using IQCs (Köse and Scherer, 2009; Veenman and Scherer, 2014). Moreover, as we briefly discussed in Chapter 6, the proposed algorithms can also be used in applications from other disciplines. We will also explore such possibilities as part of future research work.

Bibliography

- M. S. Andersen. *Chordal Sparsity in Interior-Point Methods for Conic Optimization*. PhD dissertation, university of California, Los Angeles, 2011.
- M. S. Andersen, A. Hansson, S. Khoshfetrat Pakazad, and A. Rantzer. Distributed robust stability analysis of interconnected uncertain systems. In *Proceedings of the 51st Annual Conference on Decision and Control*, pages 1548–1553, December 2012.
- M. S. Andersen, S. Khoshfetrat Pakazad, A. Hansson, and A. Rantzer. Robust stability analysis of sparsely interconnected uncertain systems. *IEEE Transactions on Automatic Control*, 59(8):2151–2156, 2014.
- M. Annergren, S. Khoshfetrat Pakazad, A. Hansson, and B. Wahlberg. A distributed primal-dual interior-point method for loosely coupled problems using ADMM. *Submitted to Optimization Methods and Software*, 2015.
- H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer Science & Business Media, 2011.
- B. W. Bequette. *Process Control–Modeling, Design and Simulation*. Pearson Education, 2003.
- D. P. Bertsekas. *Convex Optimization Theory*. Athena Scientific, 2009.
- D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.
- J. R. S. Blair and B. W. Peyton. An introduction to chordal graphs and clique trees. In J. A. George, J. R. Gilbert, and J. W-H. Liu, editors, *Graph Theory and Sparse Matrix Computations*, volume 56, pages 1–27. Springer-Verlag, 1994.
- S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- S. Boyd, V. Balakrishnan, and P. Kabamba. On computing the $\mathcal{H}_\infty$ norm of a transfer matrix. In *Proceedings of the American Control Conference*, pages 2412–2417, 1988.

- S. Boyd, L. ElGhaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics, 1994.
- R. E. Bruck. An iterative solution of a variational inequality for certain monotone operators in hilbert space. *Bulletin of the American Mathematical Society*, 81(5):890–892, September 1975.
- P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, volume 49 of *Springer Optimization and Its Applications*, pages 185–212. Springer New York, 2011.
- E. de Klerk, T. Terlaky, and K. Roos. Self-dual embeddings. In H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors, *Handbook of semidefinite programming: Theory, algorithms, and applications*, volume 27, pages 111–138. Springer Science & Business Media, 2000.
- M. di Bernardo, A. Salvi, and S. Santini. Distributed consensus strategy for platooning of vehicles in the presence of time-varying heterogeneous communication delays. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):102–112, February 2015.
- J. Dold and O. Stursberg. Distributed predictive control of communicating and platooning vehicles. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 561–566, December 2009.
- J. C. Doyle, K. Glover, P. P. Khargonekar, and B. A. Francis. State-space solutions to standard $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems. *IEEE Transactions on Automatic Control*, 34(8):831–847, August 1989.
- W. B. Dunbar and D. S. Caveney. Distributed receding horizon control of vehicle platoons: Stability and string stability. *IEEE Transactions on Automatic Control*, 57(3):620–633, March 2012.
- J. Eckstein. *Splitting methods for monotone operators with application to parallel optimization*. PhD dissertation, Massachussets Intitute of Technology, 1989.
- M. K. H. Fan, A. L. Tits, and J. C. Doyle. Robustness in the presence of mixed parametric uncertainty and unmodeled dynamics. *IEEE Transactions on Automatic Control*, 36(1):25–38, 1991.
- J. H. Ferziger. *Numerical Methods for Engineering Applications*. Wiley, 1998.
- A. Forsgren, P. E. Gill, and M. H. Wright. Interior methods for nonlinear optimization. *SIAM Rev.*, 44(4):525–597, April 2002.
- M. Fukuda, M. Kojima, M. Kojima, K. Murota, and K. Nakata. Exploiting sparsity in semidefinite programming via matrix completion I: General framework. *SIAM Journal on Optimization*, 11:647–674, 2000.

- A. Garulli, A. Hansson, S. Khoshfetrat Pakazad, R. Wallin, and A. Masi. Robust finite-frequency $\mathcal{H}_2$ analysis of uncertain systems with application to flight comfort analysis. *Control Engineering Practice*, 21(6):887–897, 2013.
- T. Glad and L. Ljung. *Control Theory, Multivariable and Nonlinear Methods*. CRC Press, 2000.
- M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Elsevier, 2nd edition, 2004.
- R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. Positive definite completions of partial Hermitian matrices. *Linear Algebra and its Applications*, 58: 109–124, 1984.
- A. Hansson and L. Vandenberghe. Efficient solution of linear matrix inequalities for integral quadratic constraints. In *Proceedings of the 39th IEEE Conference on Decision and Control*, volume 5, pages 5033–5034, 2000.
- I. M. Horowitz. *Quantitative Feedback Design Theory*. QFT Publications, 1993.
- U. Jönsson. Lecture notes on integral quadratic constraints, 2001.
- S. Khoshfetrat Pakazad, A. Hansson, and A. Garulli. On the calculation of the robust finite frequency $\mathcal{H}_2$ norm. In *Proceedings of the 18th IFAC World Congress*, pages 3360–3365, August 2011.
- S. Khoshfetrat Pakazad, M. S. Andersen, A. Hansson, and A. Rantzer. Decomposition and projection methods for distributed robustness analysis of interconnected uncertain systems. In *Proceedings of the 13th IFAC Symposium on Large Scale Complex Systems: Theory and Applications*, pages 194–199, August 2013a.
- S. Khoshfetrat Pakazad, H. Ohlsson, and L. Ljung. Sparse control using sum-of-norms regularized model predictive control. In *Proceedings of the 52nd Annual Conference on Decision and Control*, pages 5758–5763, December 2013b.
- S. Khoshfetrat Pakazad, A. Hansson, and M. S. Andersen. Distributed interior-point method for loosely coupled problems. In *Proceedings of the 19th IFAC World Congress*, pages 9587–9592, August 2014a.
- S. Khoshfetrat Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer. Distributed robustness analysis of interconnected uncertain systems using chordal decomposition. In *Proceedings of the 19th IFAC World Congress*, volume 19, pages 2594–2599, 2014b.
- S. Khoshfetrat Pakazad, M. S. Andersen, and A. Hansson. Distributed solutions for loosely coupled feasibility problems using proximal splitting methods. *Optimization Methods and Software*, 30(1):128–161, 2015a.
- S. Khoshfetrat Pakazad, A. Hansson, and M. S. Andersen. Distributed primal-dual interior-point methods for solving loosely coupled problems using message passing. *Submitted to Optimization Methods and Software*, 2015b.

- S. Kim, M. Kojima, M. Mevissen, and M. Yamashita. Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion. *Mathematical Programming*, 129(1):33–68, 2011.
- D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.
- I. E. Köse and C. W. Scherer. Robust $\mathcal{L}_2$ -gain feedforward control of uncertain systems using dynamic IQCs. *International Journal of Robust and Nonlinear Control*, 19(11):1224–1247, 2009.
- J. M. Maciejowski. *Predictive Control: With Constraint*. Pearson Education. Prentice Hall, 2002.
- J. Magni. User Manual of the Linear Fractional Representation Toolbox Version 2.0. Technical report, ONERA, Systems Control and Flight Dynamics Department, February 2006.
- A. Masi, R. Wallin, A. Garulli, and A. Hansson. Robust finite-frequency $\mathcal{H}_2$ analysis. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 6876–6881, December 2010.
- P. Massioni and M. Verhaegen. Distributed control for identical dynamically coupled systems: A decomposition approach. *IEEE Transactions on Automatic Control*, 54(1):124–135, January 2009.
- A. Megretski and A. Rantzer. System analysis via integral quadratic constraints. *IEEE Transactions on Automatic Control*, 42(6):819–830, June 1997.
- H. Ohlsson, T. Chen, S. Khoshfetrat Pakazad, L. Ljung, and S. Sastry. Distributed change detection. In *Proceedings of the 16th IFAC Symposium on System Identification*, pages 77–82, July 2012.
- H. Ohlsson, T. Chen, S. Khoshfetrat Pakazad, L. Ljung, and S. Sastry. Scalable anomaly detection in large homogeneous populations. *Automatica*, 50(5):1459–1465, 2014.
- F. Paganini. State-space conditions for robust $\mathcal{H}_2$ analysis. In *Proceedings of the American Control Conference*, volume 2, pages 1230–1234, June 1997.
- F. Paganini. Convex methods for robust $\mathcal{H}_2$ analysis of continuous-time systems. *IEEE Transactions on Automatic Control*, 44(2):239–252, Feb 1999a.
- F. Paganini. Frequency domain conditions for robust $\mathcal{H}_2$ performance. *IEEE Transactions on Automatic Control*, 44(1):38–49, January 1999b.
- F. Paganini and E. Feron. Linear matrix inequality methods for robust $\mathcal{H}_2$ analysis: A survey with comparisons. In L. El Ghaoui and S. Niculescu, editors, *Advances in linear matrix inequality methods in control*, pages 129–151. Society for Industrial and Applied Mathematics, 2000.

- S. Khoshfetrat Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer. Distributed semidefinite programming with application to large-scale system analysis. *Submitted to IEEE Transactions on Automatic Control*, 2015.
- N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014.
- A. Rantzer. On the Kalman-Yakubovich-Popov lemma. *Systems and Control Letters*, 28(1):7–10, 1996.
- J. B. Rawlings. Tutorial overview of model predictive control. *IEEE Control Systems*, 20(3):38–52, Jun. 2000.
- J. K. Rice and M. Verhaegen. Distributed control: A sequentially semi-separable approach for spatially heterogeneous linear systems. *IEEE Transactions on Automatic Control*, 54(6):1270–1283, June 2009.
- R. T. Rockafellar. Characterization of the subdifferentials of convex functions. *Pacific Journal of Mathematics*, 17(3):497–510, 1966.
- R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.
- A. Simonetto and G. Leus. Distributed maximum likelihood sensor network localization. *IEEE Transactions on Signal Processing*, 62(6):1424–1437, March 2014.
- S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control*. Wiley, 2007.
- D. Swaroop and J. K. Hedrick. String stability of interconnected systems. *IEEE Transactions on Automatic Control*, 41(3):349–357, March 1996.
- K. J. Åström and B. Wittenmark. *Computer Controlled Systems: Theory and Design*. Prentice Hall, 1990.
- M. J. Todd, K. C. Toh, and R. H. Tütüncü. On the Nesterov–Todd direction in semidefinite programming. *SIAM Journal on Optimization*, 8(3):769–796, March 1998.
- V. Turri, B. Besselink, J. Martensson, and K. H. Johansson. Fuel-efficient heavy-duty vehicle platooning by look-ahead control. In *Proceedings of the 53rd Annual Conference on Decision and Control*, pages 654–660, December 2014.
- J. Veenman and C. W. Scherer. IQC-synthesis with general dynamic multipliers. *International Journal of Robust and Nonlinear Control*, 24(17):3027–3056, 2014.
- R. Wallin, A. Hansson, and J. H. Johansson. A structure exploiting preprocessor for semidefinite programs derived from the Kalman-Yakubovich-Popov lemma. *IEEE Transactions on Automatic Control*, 54(4):697–704, April 2009.

- R. Wallin, S. Khoshfetrat Pakazad, A. Hansson, A. Garulli, and A. Masi. Applications of IQC-based analysis techniques for clearance. In A. Varga, A. Hansson, and G. Puyou, editors, *Optimization Based Clearance of Flight Control Laws*, volume 416 of *Lecture Notes in Control and Information Sciences*, pages 277–297. Springer Berlin Heidelberg, 2012.
- S. J. Wright. *Primal-dual Interior-point Methods*. Society for Industrial and Applied Mathematics, 1997.
- K. Zhou and J. C. Doyle. *Essentials of Robust Control*. Prentice Hall, 1998.

Part II

Publications