A Generic Interior-Point Framework for Nonsmooth and Complementarity Constrained Nonlinear Optimization

Von der Fakultät für Mathematik und Physik
der Gottfried Wilhelm Leibniz Universität Hannover

zur Erlangung des Grades
Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation
von

Dipl.-Math. Martin Schmidt

geboren am 11.03.1983 in Gifhorn

2013

Referent:	Prof. Dr. Marc Steinbach, Gottfried Wilhelm Leibniz Universität Hannover
Korreferent:	Prof. Dr. Alexander Martin, Friedrich-Alexander-Universität Erlangen-Nürnberg
Korreferent:	Prof. Dr. Andreas Wächter, Northwestern University
Tag der Promotion:	23. Januar 2013

For Christina

Abstract

Interior-point methods are one of the most powerful algorithmic concepts in optimization. In this thesis a generic interior-point framework is developed that allows the user to easily modify and extend a basic algorithm which combines techniques of state-of-the-art solvers for nonlinear and nonconvex optimization. After presenting these standard techniques, a special subclass of nonsmooth constrained problems is defined. It is shown that this subclass is practically relevant and a modified and extended interior-point method is developed that is able to solve this class of nonsmooth problems. Furthermore, algorithmic extensions and modifications of the basic method are presented that enable the algorithm to solve nonlinear mathematical programs with complementarity constraints.

As an application of the developed interior-point framework, the problem of validation of nominations in gas transport networks is considered. Highly nonlinear and nonconvex models of gas dynamics as well as modeling of controllable network devices lead to a mixed-integer, nonsmooth, nonconvex and nonlinear feasibility problem. This problem is extremely challenging and real-world instances cannot be solved by general-purpose solvers. It is shown that the problem belongs to a certain subclass of mixed-integer nonlinear problems for which a general reformulation technique is developed. This reformulation results in a nonsmooth and complementarity constrained nonlinear feasibility problem.

The presented computational experiments show that the reformulation technique combined with the extended and modified interior-point framework can be used to solve real-world instances of the problem of validation of nominations. The strength and generality of the developed framework is finally demonstrated by additional numerical results for problems from the fields of stochastic programming and nonlinear optimization with ordinary differential equations.

Keywords: interior-point methods, (mixed-integer) nonlinear optimization, nonsmooth optimization, complementarity constraints, gas transport networks

Zusammenfassung

Innere-Punkte Methoden bilden eines der stärksten algorithmischen Konzepte der Optimierung. In dieser Arbeit wird eine generische Programmbibliothek für Innere-Punkte Methoden entwickelt, die auf Standardtechniken der nichtlinearen und nichtkonvexen Optimierung basiert und die einfach durch den Nutzer modifiziert und erweitert werden kann. Nach der Beschreibung dieser Standardtechniken wird eine Klasse von speziellen nichtglatten Optimierungsproblemen definiert. Es wird gezeigt, dass diese Klasse von praktischer Bedeutung ist und es wird eine modifizierte und erweiterte Innere-Punkte Methode zur Lösung dieser nichtglatten Probleme auf der Basis der beschriebenen Standardtechniken entwickelt. Zusätzlich werden Erweiterungen beschrieben, die es dem Algorithmus ermöglichen, nichtlineare Optimierungsprobleme mit Komplementaritätsbedingungen zu lösen.

Als Anwendungsproblem wird das Problem der Nominierungsvalidierung in Gastransportnetzwerken betrachtet. Hochgradig nichtlineare und nichtkonvexe Modelle der Gasdynamik und die Modellierung steuerbarer Netzwerkelemente führen dabei zu einem gemischt-ganzzahligen, nichtglatten, nichtkonvexen und nichtlinearen Zulässigkeitsproblem. Diese Klasse von Problemen ist extrem schwer lösbar und kann daher nicht mit allgemeinen Standardlösern behandelt werden. Zur Lösung dieser Probleme wird daher gezeigt, dass das Problem der Nominierungsvalidierung einer speziellen Menge von nichtlinearen gemischt-ganzzahligen Optimierungsproblemen angehört, für die eine allgemeine Reformulierungstechnik entwickelt wird. Diese Reformulierung führt schließlich auf nichtglatte und nichtlineare Zulässigkeitsprobleme mit Komplementaritätsbedingungen.

Die vorgestellten numerischen Ergebnisse zeigen, dass die Reformulierungstechniken in Kombination mit der erweiterten und modifizierten Innere-Punkte Methode genutzt werden können, um reale Instanzen des Problems der Nominierungsvalidierung zu lösen. Die Stärke und Allgemeinheit der entwickelten Programmbibliothek wird zusätzlich durch Ergebnisse für Probleme aus dem Bereich der stochastischen Optimierung und der nichtlinearen Optimierung mit Differentialgleichungen belegt.

Schlagworte: Innere-Punkte Methoden, (gemischt-ganzzahlige) nichtlineare Optimierung, nichtglatte Optimierung, Komplementaritätsbeschränkungen, Gastransportnetzwerke

Acknowledgements

I wish to thank my supervisor Marc Steinbach for giving me the chance to work in such an interesting field of applied mathematics and for a lot of discussions that guided me during my work for this thesis.

I want to thank Alexander Martin and his working group, especially Björn Geißler and Antonio Morsi, for the nice and friendly collaboration within the ForNe project. Further I wish to thank Andreas Wächter for his valuable comments on an earlier version of the interior-point method for nonsmooth constrained problems.

Additionally, I want to thank some of my colleagues. The discussions and the joint work on some parts of Clean::IPM with Jens Hübner greatly improved the design of the implementation. I would also thank Marcus O'Connor for his help in preparing the plots of the performance profiles and for his help in developing the XML parsers of iGNO. In addition, I thank Jan Thiedau for the cooperation concerning the development of iGNO and for his valuable comments on some aspects of the nonsmooth interior-point method. A special thank is dedicated to my colleague and friend Bernhard Willert for his patient help at almost every day in the last four years.

Finally, I want to thank my wife Christina. Without her encouragement, support and belief in me, this thesis would not have been possible.

Ein allerletzter Dank gebührt meinen Eltern und meiner Schwester für ihre rückhaltlose Unterstützung in allen Lebensbereichen.

Contents

Abstract **v**

Zusammenfassung **vi**

Acknowledgements **viii**

List of Figures **xiii**

List of Tables **xvi**

List of Algorithms **xvii**

1 Introduction **1**

1.1 Computational Mathematics and Applied Mathematical Optimization 1

1.2 Interior-Point Methods 2

1.3 Contributions and Organization 3

2 Basic Concepts **5**

2.1 Nonlinear Optimization 5

2.2 Mathematical Programs with Complementarity Constraints 8

2.3 Nonsmooth Analysis 12

3 Optimization in Gas Network Planning **17**

3.1 Literature Survey 18

3.2 A Nonsmooth MINLP Model of the Problem of Validation of Nominations 20

3.2.1 Basic Physical Quantities 20

3.2.2 The Network Topology 21

3.2.3 Nodes 22

3.2.4 Arcs 23

3.2.5 Model Summary 34

3.3 An MPCC Approach for MINLPs in Gas Network Planning 36
3.3.1 A Reformulation Technique for 2-State-MINLPs 37
3.3.2 The Problem of Validation of Nominations Revisited 39

4 Interior-Point Methods 43
4.1 Interior-Point Methods for Nonlinear Optimization 44
4.1.1 Computation of the Search Direction 49
4.1.2 Globalization with a Filter Line-Search Algorithm 56
4.1.3 Feasibility Restoration Phase 62
4.1.4 Updating the Barrier Parameter 62
4.1.5 Starting Point Strategies 66
4.1.6 Problem Scaling 67
4.1.7 Heuristics and Algorithmic Details 68
4.1.8 The Complete Interior-Point Algorithm 70
4.1.9 Convergence Analysis 72
4.2 An Interior-Point Method for Nonsmooth Nonlinear Problems 75
4.2.1 Definition of the Problem Class 75
4.2.2 Basic Algorithmic Strategy 79
4.2.3 Modified Building Blocks 80
4.2.4 An Extended Interior-Point Method for Nonsmooth Nonlinear Optimization 90
4.3 An Interior-Point Method for MPCCs 92
4.3.1 MPCC Regularization by Relaxation 92
4.3.2 MPCC Regularization by Penalization 93
4.3.3 Updating the Regularization Parameter 94
4.4 An Interior-Point Method for Nonsmooth and Complementarity Constrained Nonlinear Optimization 97

5 Software Design 99
5.1 General Concepts of Software Design 99
5.2 The Software Architecture of Clean::IPM 103
5.3 Used External Libraries and Code Quality 105

6 Numerical Results 109
6.1 Computation of Recombination Probabilities 109
6.2 Large-Scale Stochastic Programming 111
6.3 The Hock–Schittkowski Test Set 112
6.4 Gas Network Planning 115

7 Conclusions and Outlook **125**

Bibliography **127**

List of Figures

1.1 Working cycle of applied mathematical optimization. 2

2.1 Feasible set of $\phi_i(x)\,\psi_i(x) \leq \xi$ with $\xi = 1$. 11
2.2 Three subgradients of a univariate function $f : \mathbb{R} \to \mathbb{R}$. 15

3.1 Compressibility factor computed by the AGA formula. 25
3.2 Profiles of gas pressure along three horizontal pipes ($L = 25$ km, $D \in \{0.9, 1.0, 1.1\}$ m, $k = 0.06$ mm; $q = 500$ kg s^{-1}). 25
3.3 HPPC friction term and smooth approximation vs. mass flow (kg s^{-1}): transition from laminar to turbulent flow (upper figure) and highly turbulent flow (lower figure). 28
3.4 Smoothing of sign(x) with different values of τ (blue and loosely dashed for $\tau = 0.1$, red and dashed for $\tau = 0.01$, green and densely dashed for $\tau = 0.001$). 30

4.1 A filter with five entries. 57
4.2 A generalized gradient for the absolute value function. 87

5.1 Orthogonal vectors v_1 and v_2 and software modules M_1 and M_2. 101
5.2 Orthogonal algorithmic building blocks; barrier parameter update rules and KKT system solution algorithms. 101
5.3 A schematic overview of the software architecture of Clean::IPM. 106
5.4 The current set of update rules for the barrier parameter implemented in Clean::IPM. 107

6.1 Clean::IPM with mixed μ-strategies on the Hock–Schittkowski test set. 114
6.2 Clean::IPM with monotone μ-strategies on the Hock–Schittkowski test set. 115
6.3 Schematic plot of the northern high-calorific gas transport network of Open Grid Europe GmbH. 116
6.4 Performance profiles for Clean::IPM applied to the nMPCC-s model. 119
6.5 Performance profiles for Clean::IPM applied to the nMPCC-a model. 121
6.6 Primal and dual infeasibility during the solution of an exemplary nMPCC-s instance. 122

6.7 Primal and dual infeasibility during the solution of an exemplary nMPCC-a instance. 123
6.8 Primal and dual step lengths during the solution of an exemplary nMPCC-s instance. 123
6.9 Primal and dual step lengths during the solution of an exemplary nMPCC-a instance. 124

List of Tables

3.1 Basic physical quantities. 21
3.2 Technical pipe parameters and physical quantities appearing in the pipe model. . . 24
3.3 Node types and network elements. 34

4.1 Vector management for IPM reduction, reduced KKT system solution and IPM expansion. 71
4.2 Vector management for IPM reduction, reduced KKT system solution and IPM expansion in the bound-feasible case. 71

5.1 Code statistics for Clean::IPM. 105

6.1 Model statistics and solution times of *huge*-scale tree-sparse convex programs. . . . 112
6.2 Network elements in the northern high-calorific gas transport network of Open Grid Europe GmbH. 117
6.3 Size of the reformulated MPCC model (for the northern high-calorific gas network). 117
6.4 Statistics of the iteration numbers (k) for Clean::IPM applied to the nMPCC-s model. 120
6.5 Statistics of the solution times (t, in s) for Clean::IPM applied to the nMPCC-s model. 120
6.6 Statistics of the iteration numbers (k) for Clean::IPM applied to the nMPCC-a model. 121
6.7 Statistics of the solution times (t, in s) for Clean::IPM applied to the nMPCC-a model. 121

List of Algorithms

1 Basic Interior-Point Framework 48
2 Solution Algorithm for the Reduced KKT System with Dense Matrix Blocks 55
3 Filter Line-Search Algorithm 61
4 Globalization Filter Line-Search Method for Mixed Barrier Parameter Updates . . . 64
5 Default Interior-Point Method Initialization Scheme 67
6 Filter Line-Search Interior-Point Algorithm with Mixed Barrier Parameter Updates 72
7 Modified Filter Line-Search for Nonsmooth Problems 81
8 Step Length Truncation Rule for Nonsmooth Problems 83
9 Extended Interior-Point Method for Nonsmooth Problems 91
10 Extended Interior-Point Method for MPCCs 96

Chapter 1

Introduction

1.1 Computational Mathematics and Applied Mathematical Optimization

Computational mathematics and especially the subfield of applied mathematical optimization connects a lot of different fields of mathematics and computer science.

In the beginning, there is a problem in a real-world application, e.g. from industry or finance. This problem has to be translated into a mathematical model. If the application leads to a mathematical optimization model, there are a lot of possible classes the model may belong to. Prominent examples are linear and nonlinear as well as mixed-integer (non-)linear problems. When a first model is set up, it has to be investigated theoretically and, if possible, solved. The theoretical investigation includes replying to questions concerning the qualitative analysis of the model and the existence and uniqueness of solutions. Based on these theoretical analyses one has to choose or implement algorithms to solve the problem numerically. In order to do this, one makes use of several techniques from software engineering, from the design of mathematical algorithms and from numerics. Hopefully, the developed software is then able to produce solutions of the model of the real-world application. These solutions are then discussed with the client. Figure 1.1 illustrates the described *working cycle* of applied mathematical optimization. Unfortunately, it is unlikely that the customer will be satisfied with the first version of the solutions.[1] In addition, it often turns out that it is not possible to solve the first version of the mathematical model of the real-world application. Several reasons might be supposable: The model at hand might be too large in order to be solvable by standard approaches or the mathematical model may simply be wrong in some of its aspects. At that time, the cycle is complete. The model has to be slightly reformulated

[1] The author apologizes that he cannot avoid to make the, at least partly, ironic remark that he thanks his industrial partners for teaching him this painful lecture and thus, preparing him for real life.

or it has to be restated substantially in order to obtain a different problem class. For instance, it might be necessary to get rid of nonlinear aspects of a mixed-integer nonlinear model, because the mixed-integer nonlinear problem turned out to be too hard to be solved. In other situations it might not be possible to achieve a different class of models. Then it is required to improve the complete lower block of Figure 1.1 in order to be able to produce solutions of practical relevance. Later it is discussed which aspects of the working cycle are addressed in this thesis.

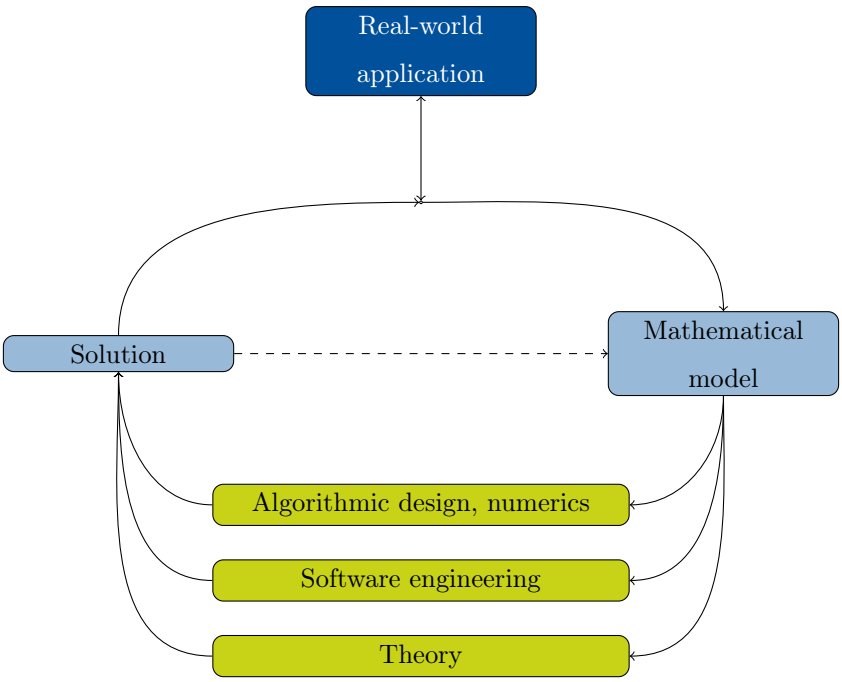

Figure 1.1: Working cycle of applied mathematical optimization.

1.2 Interior-Point Methods

Interior-point methods are the algorithmic workhorse of the solution strategies discussed in this thesis. They arose from the search for algorithms with better complexity characteristics than the simplex method in the 1980s and are one of today's most important classes of algorithms in mathematical optimization. In linear programming, interior-point methods compete with simplex methods. They are advantageous for very large-scale applications but are in a disadvantage compared to simplex type methods when applied to a series of similar problems. The latter is of particular interest in the field of mixed-integer (non-)linear optimization where a lot of problems have to be solved that only differ in a few details. In quadratic and nonlinear programming, interior-point methods compete with active-set and sequential quadratic programming (SQP) methods. The pros and cons are the same as for linear programming.

Like the last paragraph shows, interior-point methods possess the valuable property that they can be applied to a lot of problem classes. In addition, their general algorithmic framework stays predominantly the same when applied to different classes of optimization problems like linear, quadratic or (nonconvex) nonlinear problems.

This characteristic forms the basis of the algorithms developed for this thesis. Based on different existing implementations of interior-point methods a carefully designed framework is developed that enables its user to easily modify and exchange certain parts, called *building blocks*, of the interior-point algorithm. This allows the user to extend the method for solving more challenging problem classes or to easily instantiate different variants of the algorithm.

1.3 Contributions and Organization

Since the 1960s, the field of gas transport is a rich source for challenging mathematical optimization problems. Prominent examples are cost minimization or topology planning problems. In this thesis the problem of *validation of nominations* in gas transport networks is considered. This real-world problem leads to a practically intractable mixed-integer, nonsmooth, nonconvex and nonlinear feasibility problem for which a model reformulation technique as well as solution techniques are developed in this thesis.

After introducing the required basic concepts of mathematical optimization in Chapter 2, a problem description and a simplified version of the model of the problem of validation of nominations are given in Chapter 3. By simply combining the class of this problem with the size of the models of real-world problems it is evident that this mathematical model is not solvable by standard approaches. As a remedy, a reformulation technique for a certain subclass of mixed-integer nonlinear problems is developed and applied to the model at the end of Chapter 3. This reformulation leads to nonlinear and nonsmooth complementarity constrained problems. However, these problems are also not practically treatable for standard approaches. For this reason, an interior-point framework is developed that is then extended and modified in order to be able to solve this problem class. The basic interior-point algorithm and the extensions and modifications are topic of Chapter 4. The focus in this thesis, and especially in Chapter 4, is on algorithmic design and the techniques of software engineering that are used to implement the interior-point framework (Chapter 5). Chapter 6 presents solutions of the reformulated model of the problem of validation of nominations. Additionally, some numerical results from other applications of the developed interior-point framework are discussed to demonstrate its strength and generality. Finally, Chapter 7 concludes the thesis and gives some directions for future work.

Chapter 2

Basic Concepts

In this chapter the terminology and basic concepts of the fields of optimization are introduced that are topic of this thesis. The chapter is organized as follows. In Section 2.1 fundamental definitions are presented and first-order necessary conditions for nonlinear optimization problems (NLP) are stated. Section 2.2 deals with mathematical programs with complementarity constraints (MPCC). Finally, Section 2.3 briefly introduces some important definitions and theorems of nonsmooth analysis.

2.1 Nonlinear Optimization

This section reviews the basic concepts of nonlinear optimization. The presentation is based on [91].

Consider the constrained optimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.1a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{2.1b}$$

$$c_{\mathcal{I}}(x) \geq 0. \tag{2.1c}$$

Throughout this thesis, the following notation is used. $f : \mathbb{R}^n \to \mathbb{R}$ is called the *objective function*. $\mathcal{E}$ and $\mathcal{I}$ are finite index sets for *equality* and *inequality constraints* with $\mathcal{E} \cap \mathcal{I} = \emptyset$. According to this,

$$c_{\mathcal{E}} : \mathbb{R}^n \to \mathbb{R}^m \quad \text{and} \quad c_{\mathcal{I}} : \mathbb{R}^n \to \mathbb{R}^k \tag{2.2}$$

denote the vectors of equality and inequality constraints, respectively. If not stated otherwise, $|\mathcal{E}| = m$ and $|\mathcal{I}| = k$ holds. The real-valued functions $c_i : \mathbb{R}^n \to \mathbb{R}, i \in \mathcal{E}$, are the single equality and $c_i : \mathbb{R}^n \to \mathbb{R}, i \in \mathcal{I}$, are the single inequality constraints. The relations $=$ and $\geq$ in (2.1) are defined component-wise.

Definition 1 (Feasible Set)**.** *The set*

$$\mathcal{F} = \{x \in \mathbb{R}^n \colon c_{\mathcal{E}}(x) = 0 \text{ and } c_{\mathcal{I}}(x) \geq 0\} \tag{2.3}$$

is the feasible set *of* (2.1)*. A point* $x \in \mathbb{R}^n$ *is called* feasible*, if* $x \in \mathcal{F}$*.*

Next, the definition of a local solution of (2.1) is given.

Definition 2 (Local Solution)**.** *A point* $x^* \in \mathbb{R}^n$ *is a* local solution *of problem* (2.1) *if* $x^* \in \mathcal{F}$ *and there exists a neighborhood* $\mathcal{N}$ *of* x^* *such that* $f(x^*) \leq f(x)$ *for all* $x \in \mathcal{F} \cap \mathcal{N}$*.*

The goal of the rest of this section is to state first-order optimality conditions for the constrained optimization problem (2.1). For this, the definitions of the active set and the linear independence constraint qualification are needed.

Definition 3 (Active Set)**.** *Let* $x \in \mathbb{R}^n$ *be a feasible point of* (2.1)*. Then the index set*

$$\mathcal{A}(x) = \mathcal{E} \cup \{i \in \mathcal{I} \colon c_i(x) = 0\} \tag{2.4}$$

is called the active set *of* (2.1) *at* x*.*

Definition 4 (Linear Independence Constraint Qualification (LICQ))**.** *Let* $x \in \mathbb{R}^n$ *be a feasible point of* (2.1) *and let* $\mathcal{A}(x)$ *be the active set at* x*. One says that the* linear independence constraint qualification (LICQ) *holds at* x *if the set*

$$\{\nabla c_i(x) : i \in \mathcal{A}(x)\} \tag{2.5}$$

is linearly independent.

Finally, the Lagrangian function of (2.1) is defined as follows.

Definition 5 (Lagrangian Function)**.** *The function*

$$\mathcal{L}(x, \lambda_{\mathcal{E}}, \lambda_{\mathcal{I}}) = f(x) - \sum_{i \in \mathcal{E}} \lambda_i c_i(x) - \sum_{i \in \mathcal{I}} \lambda_i c_i(x) \tag{2.6}$$

is called the Lagrangian function *of* (2.1)*. The vectors* $\lambda_{\mathcal{E}} := (\lambda_i)_{i \in \mathcal{E}} \in \mathbb{R}^{|\mathcal{E}|}$ *and* $\lambda_{\mathcal{I}} := (\lambda_i)_{i \in \mathcal{I}} \in \mathbb{R}^{|\mathcal{I}|}$ *are the so-called* Lagrange multipliers *(also known as* dual variables *or* dual multipliers*) corresponding to equality and inequality constraints, respectively.*

The following theorem about first-order necessary conditions forms the basis of the algorithms that are discussed in this thesis.

Theorem 1 (First-Order Necessary Conditions)**.** *Let* $x^* \in \mathbb{R}^n$ *be a local solution of* (2.1) *and let* $f, c_{\mathcal{E}}$ *and* $c_{\mathcal{I}}$ *be continuously differentiable. Furthermore, assume that the LICQ condition holds at*

x^*. Then there exist vectors of Lagrange multipliers $\lambda_{\mathcal{E}}^* \in \mathbb{R}^m$ and $\lambda_{\mathcal{I}}^* \in \mathbb{R}^k$ such that the following conditions are satisfied:

$$\nabla_x \mathcal{L}\left(x^*, \lambda_{\mathcal{E}}^*, \lambda_{\mathcal{I}}^*\right) = 0, \tag{2.7a}$$

$$c_{\mathcal{E}}\left(x^*\right) = 0, \tag{2.7b}$$

$$c_{\mathcal{I}}\left(x^*\right) \geq 0, \tag{2.7c}$$

$$\lambda_{\mathcal{I}}^* \geq 0, \tag{2.7d}$$

$$\lambda_i^* c_i\left(x^*\right) = 0 \quad \forall i \in \mathcal{E} \cup \mathcal{I}. \tag{2.7e}$$

x^* is then called a KKT point *or* stationary point *of* (2.1).

Condition (2.7a) is called *dual feasibility*, conditions (2.7b) and (2.7c) are called *primal feasibility*. (2.7d) is referred to as *positivity* of the Lagrange multipliers corresponding to inequality constraints. (2.7e) is the so-called *complementarity* condition. Conditions (2.7) are known as the *Karush-Kuhn-Tucker conditions* (or *KKT conditions* for short). A proof of Theorem 1 can be found in [91, Chap. 12].

Standard solution techniques for NLP problems include interior-point methods, penalty methods and sequential quadratic programming (SQP) methods. For penalty and SQP methods see the book of Nocedal and Wright [91] and the references therein. Since interior-point methods are one of the central topics of this thesis, a more detailed overview of the existing methods is given. One can distinguish mainly between line-search and trust-region interior-point methods. The most prominent line-search implementations are LOQO [128], KNITRO/DIRECT [135], Ipopt [134] and MOSEK (only for convex problems, [4]). KNITRO/CG [19] implements a trust-region interior-point approach. Moreover, the complete KNITRO package combines both interior-point and active-set strategies [20]. The only non-commercial and open source code of the above mentioned is Ipopt.

A main characteristic of (2.1) is that all decision variables x are continuous, i.e. $x \in \mathbb{R}^n$. Nevertheless, in practice one often requires discrete variables to model real-world problems. This leads to the generalization of (2.1) that is called a *mixed-integer nonlinear optimization problem* (MINLP):

$$\min_{x,z} \quad f(x,z) \tag{2.8a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x,z) = 0, \tag{2.8b}$$

$$c_{\mathcal{I}}(x,z) \geq 0, \tag{2.8c}$$

$$x \in \mathbb{R}^{n_x}, z \in \mathbb{Z}^{n_z}. \tag{2.8d}$$

In (2.8) the constraints and the objective function may depend on additional discrete decision variables $z \in \mathbb{Z}^{n_z}$, i.e.

$$f : \mathbb{R}^{n_x} \times \mathbb{Z}^{n_z} \to \mathbb{R}, \quad c_{\mathcal{E}} : \mathbb{R}^{n_x} \times \mathbb{Z}^{n_z} \to \mathbb{R}^m, \quad c_{\mathcal{I}} : \mathbb{R}^{n_x} \times \mathbb{Z}^{n_z} \to \mathbb{R}^k. \tag{2.9}$$

An important special case of (2.8) are mixed-integer nonlinear programs in which all discrete variables are restricted to the binaries. This means that the integrality condition in (2.8d) is tightened to $z \in \{0,1\}^{n_z}$.

The theory as well as the algorithmic techniques for solving MINLPs are beyond the scope of this thesis. It is referred to the book of Floudas [43] for both the fundamentals and applications of MINLP. In the context of this thesis, MINLPs are used to model the problem of validation of nominations in Chapter 3.

2.2 Mathematical Programs with Complementarity Constraints

Mathematical problems with complementarity constraints (or MPCC for short) form a practically and theoretically important generalization of standard nonlinear programs (see [77] for an overview).

A standard form of MPCC problems is

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) && \text{(2.10a)} \\ \text{s.t.} \quad & c_{\mathcal{E}}(x) = 0, && \text{(2.10b)} \\ & c_{\mathcal{I}}(x) \geq 0, && \text{(2.10c)} \\ & 0 \leq \phi(x) \perp \psi(x) \geq 0, && \text{(2.10d)} \end{aligned}$$

where $f, c_{\mathcal{E}}, c_{\mathcal{I}}, \phi$ and ψ are sufficiently smooth, i.e. $\mathcal{C}^2$ if not stated otherwise. The functions

$$\phi, \psi : \mathbb{R}^n \to \mathbb{R}^p \tag{2.11}$$

form the so-called *complementarity condition* (2.10d) that is defined by

$$\phi_i(x), \psi_i(x) \geq 0 \quad \text{and} \quad \phi_i(x) = 0 \quad \text{or} \quad \psi_i(x) = 0 \quad \forall i = 1, \ldots, p. \tag{2.12}$$

$\phi_i(x)$ and $\psi_i(x)$ are the components of the vector-valued functions ϕ, ψ. The pairs ϕ_i, ψ_i are also called *complementarity pairings*. In practice, $\phi_i(x)$ and $\psi_i(x)$ are often chosen to be simple variables, i.e. $\phi_i(x) = x_{i_1}$ and $\psi_i(x) = x_{i_2}$ with variable indices $i_1, i_2 \in \{1, \ldots, n\}$. Obviously, (2.10d) is a logical condition and models a disjunction. In order to solve problems of the form (2.10), the complementarity condition has to be reformulated in an analytic form. The following

reformulation is straightforward:

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.13a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{2.13b}$$

$$c_{\mathcal{I}}(x) \geq 0, \tag{2.13c}$$

$$\phi(x) \geq 0, \psi(x) \geq 0, \tag{2.13d}$$

$$\phi_i(x)\,\psi_i(x) = 0 \quad \forall i = 1, \ldots, p. \tag{2.13e}$$

The main problem when solving (2.13) is the failure of the LICQ condition.

Theorem 2. *Let $x^* \in \mathbb{R}^n$ be a feasible point of* (2.13)*. Then the LICQ condition does not hold at x^*.*

A proof can be found in [143]. The last theorem results in a collapse of NLP optimality theory and algorithmic concepts for (2.13). This is the reason why several MPCC-tailored regularity and stationarity concepts are developed in the last decades.

In the rest of this section the main definitions of these concepts are presented and reformulation techniques are discussed that lead to regular (in the sense of constraint qualifications) versions of (2.13). First, one needs an MPCC-tailored version of the definition of the LICQ condition.

Definition 6 (MPCC-LICQ)**.** *Let $x^* \in \mathbb{R}^n$ be a feasible point of* (2.13)*. One says that the* MPCC linear independence constraint qualification (MPCC-LICQ) *holds at x^* if the standard LICQ condition holds for the set of constraints of* (2.13) *missing the complementarity conditions* (2.13e)*.*

For the following, the definition of the index sets

$$\mathcal{A}_\phi(x) := \{i \in \{1, \ldots, p\} : \phi_i(x) = 0\}, \tag{2.14a}$$

$$\mathcal{A}_\psi(x) := \{i \in \{1, \ldots, p\} : \psi_i(x) = 0\}, \tag{2.14b}$$

is required. The next theorem is the analogous result to the standard first-order KKT theorem for NLP (cf. Theorem 1). See [106] for a proof.

Theorem 3 (MPCC First-Order Necessary Conditions)**.** *Let $f, c_{\mathcal{E}}, c_{\mathcal{I}}, \phi$ and ψ be continuously differentiable, let $x^* \in \mathbb{R}^n$ be a local solution of* (2.13) *and assume that the MPCC-LICQ condition holds at x^*. Then there exist vectors of Lagrange multipliers $\lambda^*_{\mathcal{E}} \in \mathbb{R}^{|\mathcal{E}|}, \lambda^*_{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}|}$ and $\lambda^*_\phi, \lambda^*_\psi \in \mathbb{R}^p$*

such that

$$\nabla f(x^*) - \nabla c_{\mathcal{E}}(x^*)^T \lambda_{\mathcal{E}}^* - \nabla c_{\mathcal{I}}(x^*)^T \lambda_{\mathcal{I}}^* - \nabla \phi(x^*)^T \lambda_{\phi}^* - \nabla \psi(x^*)^T \lambda_{\psi}^* = 0, \tag{2.15a}$$

$$c_{\mathcal{E}}(x^*) = 0, \quad c_{\mathcal{I}}(x^*) \geq 0, \quad \phi_i(x^*) \geq 0, \quad \psi_i(x^*) \geq 0, \tag{2.15b}$$

$$\phi_i(x^*) = 0 \quad or \quad \psi_i(x^*) = 0, \quad i = 1, \ldots, p, \tag{2.15c}$$

$$c_i(x^*)\lambda_i^* = 0, \quad i \in \mathcal{I}, \quad \phi_i(x^*)\lambda_{\phi_i}^* = 0 \quad and \quad \psi_i(x^*)\lambda_{\psi_i}^* = 0, \quad i = 1, \ldots, p, \tag{2.15d}$$

$$\lambda_i^* \geq 0, \quad i \in \mathcal{I}, \tag{2.15e}$$

$$\lambda_{\phi_i}^* \geq 0, \quad \lambda_{\psi_i}^* \geq 0, \quad i \in \mathcal{A}_{\phi}(x^*) \cap \mathcal{A}_{\psi}(x^*), \tag{2.15f}$$

holds.

(2.15a) corresponds to standard dual feasibility, (2.15b) and (2.15c) cover primal feasibility and (2.15d) is the standard KKT complementarity condition of inequality constraints and their Lagrange multipliers. At last, (2.15e) and (2.15f) correspond to positivity of Lagrange multipliers of inequality constraints. For the latter, the first-order MPCC conditions only require positivity of Lagrange multipliers of complementarity pairings for so-called *corner pairings* (cf. [74]), i.e. a complementarity pairing for which $\phi_i(x^*) = \psi_i(x^*) = 0$ holds.

Theorem 3 allows to introduce MPCC-tailored stationarity concepts:

Definition 7 (MPCC Stationarity)**.** *Let* $x^* \in \mathbb{R}^n$ *be an MPCC-feasible point, i.e.* (2.15b) *and* (2.15c) *hold. Furthermore, assume that there exist Lagrange multipliers* $\lambda_{\mathcal{E}}^* \in \mathbb{R}^{|\mathcal{E}|}, \lambda_{\mathcal{I}}^* \in \mathbb{R}^{|\mathcal{I}|}, \lambda_{\phi}^*, \lambda_{\psi}^* \in \mathbb{R}^p$ *that satisfy* (2.15a), (2.15d) *and* (2.15e). x^* *is called*

1. Clarke-stationary *(or* C-stationary *for short), if in addition* $\lambda_{\phi_i}^* \lambda_{\psi_i}^* \geq 0$ *holds for all* $i \in \mathcal{A}_{\phi}(x^*) \cap \mathcal{A}_{\psi}(x^*)$ *and*

2. strongly stationary*, if in addition to 1.* (2.15f) *holds.*

The reader interested in additional MPCC-tailored constraint qualifications and stationarity concepts is referred to [106].

Beside the theoretical development concerning MPCC-tailored constraint qualifications and stationarity concepts, a lot of research deals with reformulation techniques for (2.13). These techniques have in common that they replace the original MPCC (2.13) by a parameterized sequence of regularized NLPs that fulfill standard constraint qualifications for NLPs like the LICQ condition. The first regularization technique for MPCCs is proposed in [110]. The regularization is done by a relaxation of the complementarity constraints, leading to the parameterized nonlinear program

NLP(ξ) with relaxation parameter $\xi \geq 0$:

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.16a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{2.16b}$$

$$c_{\mathcal{I}}(x) \geq 0, \tag{2.16c}$$

$$\phi(x) \geq 0, \psi(x) \geq 0, \tag{2.16d}$$

$$\phi_i(x)\,\psi_i(x) \leq \xi, \quad i = 1, \ldots, p. \tag{2.16e}$$

Obviously, the feasible set $\mathcal{F}(\xi)$ of NLP(ξ) with $\xi = 0$ is the feasible set of problem (2.13). In addition, $\mathcal{F}(\xi_0) \subset \mathcal{F}(\xi)$ holds for all $\xi > \xi_0 \geq 0$. Figure 2.1 illustrates the relaxation of the feasible region of (2.16e). For the relaxation scheme (2.16) it is shown in [110] that the sequence of stationary points of the relaxed MPCCs converges to a C-stationary point if the MPCC-LICQ condition holds in the limit. If NLP(ξ) is solved by an interior-point method, (2.16) has the

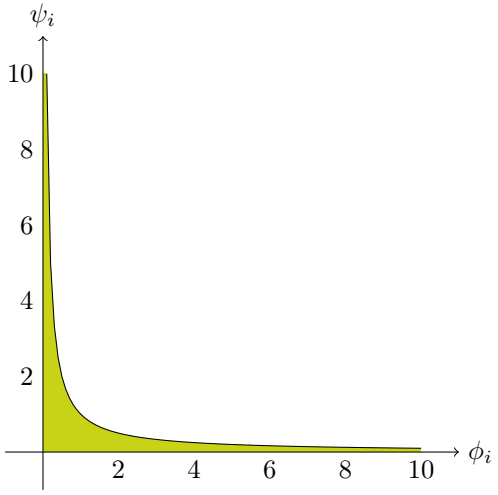

Figure 2.1: Feasible set of $\phi_i(x)\,\psi_i(x) \leq \xi$ with $\xi = 1$.

drawback that it lacks strict interior points in the limit, i.e. for $\xi \to 0$. [31] addresses this drawback by additionally relaxing the bounds in (2.16d). The resulting relaxation then reads

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.17a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{2.17b}$$

$$c_{\mathcal{I}}(x) \geq 0, \tag{2.17c}$$

$$\phi(x) \geq -\theta, \psi(x) \geq -\theta, \tag{2.17d}$$

$$\phi_i(x)\,\psi_i(x) \leq \xi, \quad i = 1, \ldots, p, \tag{2.17e}$$

with $\theta \geq 0$. The method proposed in [31] is designed in a way that it drives only θ or ξ to zero in the limit but not both. Thereby, it ensures the existence of a strict interior of the problem.

A different family of regularization techniques is based on penalization. Here, the complementarity constraints (2.13e) are removed from the set of constraints and their violation is penalized in the objective function. A general formulation reads

$$\min_{x\in\mathbb{R}^n} \quad f(x) + \frac{1}{\xi}\Pi(\phi(x), \psi(x)) \tag{2.18a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{2.18b}$$

$$c_{\mathcal{I}}(x) \geq 0, \tag{2.18c}$$

$$\phi(x) \geq 0, \psi(x) \geq 0. \tag{2.18d}$$

Penalization techniques for MPCC regularization are first proposed in [59]. Under certain assumptions on the penalty function $\Pi : \mathbb{R}^{2p} \to \mathbb{R}$ in (2.18a) it is shown in [59] that the sequence of stationary points of (2.18) converges to a C-stationary point if the MPCC-LICQ condition holds in the limit. Later, the concrete choice

$$\Pi(\phi(x), \psi(x)) = \sum_{i=1}^{p} \phi_i(x)\,\psi_i(x), \tag{2.19}$$

is used for which the above stated convergence result holds.

Stronger convergence results for both relaxation and penalization schemes can be shown under stronger assumptions like the *weak second-order necessary condition* and *upper strict level complementarity* (cf. [59, 110]).

Another kind of regularization technique is based on nonsmooth reformulations of the complementarity constraints $\phi_i(x)\psi_i(x) = 0$, such as

$$\min(\phi_i(x), \psi_i(x)) = 0. \tag{2.20}$$

Then nonsmooth optimization techniques are applied to the restated nonsmooth problem. Other approaches use smoothing techniques that exploit modified NCP-functions like the perturbed Fischer–Burmeister function (see [39])

$$\zeta(\phi, \psi; \xi) = \phi + \psi - \sqrt{\phi^2 + \psi^2 + \xi} = 0. \tag{2.21}$$

See [123] for an overview of different NCP-functions.

2.3 Nonsmooth Analysis

For generalizing an interior-point method for smooth problems to a certain class of nonsmooth constrained problems, some basic definitions and results from nonsmooth analysis are required. The presentation is based on the books [24, 25].

An important definition is that of a locally Lipschitz-continuous function.

Definition 8 (Lipschitz-Continuous Function)**.** *Let X be a real Banach space. A function $f : U \subset X \to \mathbb{R}$ is called* locally Lipschitz-continuous *(or* Lipschitz-continuous *for short)* near $x \in U$*, if there exists a neighborhood $\mathcal{N}$ of x and a constant $L = L(x)$ with*

$$|f(y) - f(z)| \leq L\|y - z\| \quad \forall y, z \in \mathcal{N}. \tag{2.22}$$

The function f is called locally Lipschitz-continuous on $U \subset X$ *if f is locally Lipschitz-continuous near every point $x \in U$.*

Definition 9 (Clarke's Generalized Gradient)**.** *Let X be a real Banach space and $f : X \to \mathbb{R}$ locally Lipschitz-continuous in a neighborhood $\mathcal{N}$ of $x \in X$. Furthermore, let $d \in X$.* Clarke's generalized directional derivative *of f at x in the direction d is defined as*

$$f^\circ(x; d) = \limsup_{\substack{y \to x \\ t \downarrow 0}} \frac{f(y + td) - f(y)}{t}. \tag{2.23}$$

Clarke's generalized gradient *of f at x is given by*

$$\partial f(x) := \{y \in X^* \colon f^\circ(x; d) \geq \langle y, d\rangle \text{ for all } d \in X\}, \tag{2.24}$$

where X^* denotes the dual space of X and $\langle a, b\rangle, a \in X^*, b \in X$, is the associated dual pairing.

Definition 9 is valid for finite as well as infinite dimensional Banach spaces. The aim of the following is to state a more practicable characterization of $\partial f(x)$ in finite dimensions. This is done by the following theorem (see [25] for a proof).

Theorem 4. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz-continuous in a neighborhood of $x \in \mathbb{R}^n$ and suppose S to be any set of Lebesgue measure 0 in $\mathbb{R}^n$. Then*

$$\partial f(x) = \operatorname{conv}\left\{\lim_{i \to \infty} \nabla f(x_i) : x_i \to x, x_i \notin S, x_i \notin \mathcal{K}\right\} \tag{2.25}$$

holds, where $\mathcal{K}$ is the set of points at which f fails to be differentiable. As usual, conv M *denotes the convex hull of the set M.*

To avoid the impractical condition "$x_i \notin S$" Rademacher's theorem is useful:

Theorem 5 (Rademacher, [24])**.** *Let $f : U \subset \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz-continuous on an open set U. Then f is differentiable almost everywhere on U (w.r.t. the Lebesgue measure).*

By applying Rademacher's theorem to Theorem 4 one obtains the following result.

Lemma 1. *For an open subset $U \subset \mathbb{R}^n$ let $f : U \to \mathbb{R}$ be locally Lipschitz-continuous and $x \in U$. Let $\mathcal{K}$ be the set of points at which f fails to be differentiable. Moreover, let $(x_i) \subset \mathbb{R}^n \setminus \mathcal{K}$ be a sequence of points converging to x. Furthermore, assume that $\lim_{i \to \infty} \nabla f(x_i)$ exists. Then $\lim_{i \to \infty} \nabla f(x_i) \in \partial f(x)$.*

Proof. Using Rademacher's theorem it follows that $\mathbb{R}^n \setminus \mathcal{K}$ is not a subset of Lebesgue measure 0. Thus, the lemma follows directly from Theorem 4. □

Finally, first-order necessary conditions are stated for (2.1) with possibly nonsmooth but locally Lipschitz-continuous objective function f and constraints $c_\mathcal{E}$ and $c_\mathcal{I}$ (cf. [25]).

Theorem 6. *Let $x^* \in \mathbb{R}^n$ be a local solution of (2.1) with locally Lipschitz-continuous functions $f, c_\mathcal{E}$ and $c_\mathcal{I}$. Then there exist Lagrange multipliers $\lambda_f^* \in \mathbb{R}, \lambda_\mathcal{E}^* \in \mathbb{R}^{|\mathcal{E}|}, \lambda_\mathcal{I}^* \in \mathbb{R}^{|\mathcal{I}|}$, not all zero, such that*

$$0 \in \partial_{x^*} \mathcal{L}\left(x^*, \lambda_f^*, \lambda_\mathcal{E}^*, \lambda_\mathcal{I}^*\right), \tag{2.26a}$$

$$0 = c_\mathcal{E}\left(x^*\right), \tag{2.26b}$$

$$0 \leq c_\mathcal{I}\left(x^*\right), \tag{2.26c}$$

$$0 \leq \lambda_i^*, \quad \forall i \in \mathcal{I}, \tag{2.26d}$$

$$0 = \lambda_i^* c_i\left(x^*\right), \quad \forall i \in \mathcal{E} \cup \mathcal{I}. \tag{2.26e}$$

Here, $\partial_{x^*} \mathcal{L}$ denotes Clarke's generalized gradient of the Lagrangian function with an additional multiplier for the objective function, i.e.

$$\partial_{x^*} \mathcal{L}\left(x^*, \lambda_f^*, \lambda_\mathcal{E}^*, \lambda_\mathcal{I}^*\right) = \lambda_f^* \partial f\left(x^*\right) - \sum_{i \in \mathcal{E}} \lambda_i^* \partial c_i\left(x^*\right) - \sum_{i \in \mathcal{I}} \lambda_i^* \partial c_i\left(x^*\right). \tag{2.27}$$

The only differences between the KKT conditions (2.7) for smooth constrained problems and the KKT conditions (2.26) for nonsmooth constrained problems are

- the generalization from $=$ to $\in$ in the dual feasibility condition (2.26a),
- the generalization from standard gradients ∇c_i to Clarke's generalized gradients ∂c_i and
- the additional Lagrange multiplier λ_f of the objective function.

Finally, the definition of a subgradient is given, which is a generalization of the gradient for convex functions in the nonsmooth case.

Definition 10 (Subgradient, Subdifferential)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function. A vector $g \in \mathbb{R}^n$ is called a* subgradient *of f at $x \in \mathbb{R}^n$ if*

$$f\left(y\right) \geq f\left(x\right) + \langle g, y - x \rangle \tag{2.28}$$

holds for all $y \in \mathbb{R}^n$. The set of all subgradients of f at x is called the subdifferential *of f at x and is denoted by $\partial f\left(x\right)$.*

It can be shown that Clarke's generalized gradient coincides with the subdifferential in the convex case (cf. [25]). Thus, Definition 10 does not lead to a conflict in the notation. The concept of a subgradient has a useful geometric interpretation: The defining inequality (2.28) states that the epigraph of f is located on or above the graph of the linear function $f(x) + \langle g, x - y \rangle$.

Example 1. *Consider the nonsmooth but convex function*

$$f(x) := \begin{cases} f_1(x) := -\frac{1}{2}x + 4, & x \leq \bar{x}, \\ f_2(x) := (x-2)^2 + 1, & x > \bar{x} \end{cases} \tag{2.29}$$

with $x \in I := [1, 4]$. $\bar{x}$ *is the intersection point of* f_1 *and* f_2 *in* I. *Obviously,* f *is not differentiable at* $\bar{x}$. *Figure 2.2 illustrates three examples of subgradients of* f *at* $\bar{x}$. *In the figure it can also be seen that the graphs of the functions* $f(\bar{x}) + \langle g, \bar{x} - y \rangle$ *are below or on the epigraph of* f *(filled area).*

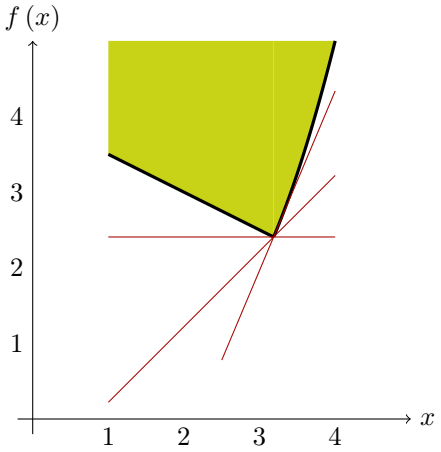

Figure 2.2: Three subgradients of a univariate function $f : \mathbb{R} \to \mathbb{R}$.

The following lemma about subdifferentials of univariate functions is needed later in Section 4.2.

Lemma 2. *Let* $f : \mathbb{R} \to \mathbb{R}$ *be a convex function. Then* $\partial f(x)$ *is a nonempty interval.*

Proof. The Lemma follows directly from the fact that a subdifferential is a nonempty, convex, closed and bounded set (see [104, Theorem 2.74]). As a subset of $\mathbb{R}$, $\partial f(x)$ is a nonempty interval. □

Chapter 3

Optimization in Gas Network Planning

Gas networks are used to transport natural gas over long distances. As an important source of primary energy, it is used e.g. for heating and in industrial processes. The tasks of network operators can be distinguished into long, mid and short term planning. The focus of this chapter is on mid term planning in transport networks. The latter are characterized by an operation on high pressure levels and very large pipeline systems. In the following, different models are developed for the problem of *validation of nominations*:

> Given a transport network and a set of contracts with supplying and discharging customers, determine whether there is a technically and physically feasible operation of the network that satisfies all contracts.

Since the problem of validation of nominations is a mid term planning task, it is reasonable to only consider the stationary case. That is, all aspects with respect to varying time are neglected in the following.

The problem of validation of nominations is only one of a lot of tasks of network planners. This research on gas network planning in general, and in particular on validation of nominations, is part of the industrial research project ForNe in cooperation with Open Grid Europe GmbH[1] and other scientific working groups.[2]

[1] See `http://www.open-grid-europe.com`.

[2] The scientific partners are Friedrich-Alexander Universität Erlangen-Nürnberg, Konrad Zuse Zentrum für Informationstechnik Berlin (ZIB), Universität Duisburg-Essen, Weierstraß Institut für Angewandte Analysis und Stochastik (WIAS), Humboldt Universität zu Berlin, Technische Universität Darmstadt and Leibniz Universität Hannover.

This chapter is organized as follows. First, Section 3.1 reviews the literature on mathematical optimization for gas transport networks. In Section 3.2 a problem description is given and a mixed-integer, nonsmooth and nonlinear model is stated for the problem of validation of nominations. Finally, an MPCC-based reformulation technique for a certain class of mixed-integer nonlinear optimization problems is presented in Section 3.3. This technique is then applied to the model that is developed in Section 3.2.

3.1 Literature Survey

Gas network optimization, especially fuel gas minimization, and simulation are very active fields of research in applied mathematics. Early approaches consider simple networks like *gun barrel* or *tree-structured* topologies and apply dynamic programming on coarse models of steady-state gas dynamics in pipes and compressor stations [138, 139]. The considered models only include the classical Weymouth approximation for pressure loss in pipes [65, 137] and compressor stations are simply modeled by maximum compression ratios. The objective function is an idealized model of compressor power using the adiabatic head of the machines. The adiabatic head is simplified by considering formulas that only depend on the compression ratio and on machine specific, prescribed constants. A review of these dynamic programming approaches can be found in [22], where other heuristics for solving the fuel gas minimization problem are also discussed. Other early research concentrates on transient simulation models [69, 70, 71, 136, 145] that are subsequently extended for steady-state [61] and transient optimization of gas networks [62, 131]. As it is typical for simulation, in [61] and [131] a continuous model with prescribed discrete decisions for the active network elements is solved. Moreover, the network sizes tackled by these approaches are only up to a dozen of nodes. Since compressor stations are the most important elements to overcome energy loss due to friction and heat exchange, a lot of research deals with modeling and optimization of single compressor stations with a fixed boundary situation, i.e. in- and outgoing pressure as well as flow through the station [21, 93, 140]. Beside heuristic approaches there are first mixed-integer nonlinear (MINLP) formulations. Most of the considered models incorporate parallel arrangements of compressor units. The type or performance characteristics of single machines are neglected in order to be able to handle the additional complexity introduced by integer variables determining the usage of single units in the parallel arrangement. The most detailed models consider compressor power based on simplified formulas of the adiabatic head, whereas other consider power functions only depending on the flow through the machine. First studies concentrating on combinatorial aspects of these models are presented in [23].

The performance of early computer hardware and software leads to very simplified models of gas dynamics and, in particular, compressor stations. Thus, up to this time no realistic operating

ranges of single compressor units or drives are considered and only coarse physical and technical approximations are included in the models. Early attempts at modeling realistic operating ranges of compressor machines of centrifugal type can be found in [16, 142]. For the first time in an optimization context, the interdependence of flow rate, compression ratio as well as of technical effects like fuel gas consumption, compressor speed and efficiency is discussed (see [92] for a recent survey on modeling of compressor machines). These papers also consider relaxation schemes for the detailed compressor unit model. On the one hand, these relaxations yield lower bounds on fuel gas consumption that are used to measure the quality of other optimization approaches to fuel gas minimization. On the other hand, the developed linear outer approximations of operating ranges and convex underestimators of cost functions allow the authors to solve the relaxed models.

More recent research tries to handle nonlinear aspects of gas dynamics and engineering together with combinatorial aspects of active elements like opening or closing of valves and activating or deactivating of pressure regulators and compressor stations. Since the solution of full MINLP models for real-world network sizes is beyond the scope of today's algorithms and software (cf. [26]), two main lines of research have been followed. The first one linearizes the nonlinearities (mostly by piecewise linear approximations) yielding mixed-integer linear (MILP) models [30, 54, 82, 83, 88, 97, 111, 112]. The techniques are applied to networks with up to 60 nodes. A very recent result also shows the applicability of these approaches to real-world network sizes [47]. Based on these results, transient gas network optimization is addressed heuristically in [81, 89]. The other approach is to assume prescribed discrete decisions and to concentrate on the remaining continuous and nonlinear model. Stationary nonlinear models are discussed in [14, 15, 101, 102, 130, 141], whereas transient models are within the scope of [34, 35, 121]. A comparison of two concrete instantiations of the MILP and the NLP approach can be found in [33]. The research cited in this paragraph additionally considers further network elements than pipes and compressor stations, namely control valves and valves. These new elements introduce additional combinatorial complexity because the algorithms have to choose between different discrete states of the elements. However, the behavior of fixed states is linear and easy to handle. Most of the discussed approaches still model pressure loss effects with (linearized versions of) Weymouth's approximation. In contrast to that, models based on prescribed discrete decisions like the ones in [34, 35, 121] consider discretization schemes of the underlying differential equations of gas dynamics.

More theoretical work considers controllability and stabilization of the governing PDE systems of gas transport networks, namely the Euler equations [7, 8, 17, 51, 52, 53, 72]. However, these detailed models of gas dynamics are only applied to very small and simple structured networks without active elements like compressor stations or valves.

Beside the effects of pressure loss in pipes and compression processes in compressor machines, two other physical effects are only handled step-motherly: the gas temperature and the compo-

sition of gas. To the best of the author's knowledge, there is no optimization approach to gas networks that incorporates a reasonable, i.e. non-isothermal, model of gas temperature. Since gas temperature is coupled with gas pressure, the nonlinearity of almost every physical equation is increased in non-isothermal models. In addition to that, new network elements like gas coolers and preheaters come into play that may be neglected in isothermal models. The effects of gas composition is (again, to the best of the author's knowledge) only mentioned in [125], where all effects are simplified by linearization.

In the already mentioned ForNe project, a first attempt for stationary network optimization is made that handles full nonlinear gas dynamics and engineering issues coupled with discontinuous mixed-integer aspects as well as with stochastic influences on demand profiles of customers [80]. Papers in preparation include [45, 67, 95]. Finally, it is explicitly mentioned that parts of this chapter are about to be published in [107, 108, 109].

3.2 A Nonsmooth MINLP Model of the Problem of Validation of Nominations

This section deals with the description of the problem of validation of nominations and states a concrete model of the problem. It turns out that the model contains mixed-integer, nonsmooth as well as nonlinear aspects, yielding a mixed-integer, nonsmooth and nonlinear optimization (or feasibility) problem.

A more detailed description of those technical details that are required to state a model of the problem of validation of nominations can be found in [107]. As it is characteristic for problems from the field of engineering, there exist a lot of different models for the same technical or physical aspect. These models mainly differ in their physical and technical accuracy. In the following, only the model is presented that is used in the implementation. For other possible model formulations see [107].

3.2.1 Basic Physical Quantities

Gas flow in transport networks is mainly described by the state variables pressure p, temperature T and density ρ as well as the gas mass flow q. Here, only the isothermal case is considered. That is, the gas temperature T is approximated by a globally constant value, e.g. 283.15 K. In what follows, all quantities indexed with 0, e.g. p_0, T_0, ρ_0, denote the corresponding quantity under normal conditions. These are defined by the normal temperature $T_0 = 273.15$ K and the normal pressure $p_0 = 1.01325$ bar.

An overview of the basic physical quantities and their units is given in Table 3.1. All additionally

Symbol	Explanation	Unit
p	Gas pressure	Pa
T	Gas temperature	K
ρ	Gas density	kg m^{-3}
v	Gas velocity	m s^{-1}
q	Gas mass flow	kg s^{-1}

Table 3.1: Basic physical quantities.

required physical and technical quantities are introduced where they are used for the first time.

3.2.2 The Network Topology

The gas transport network is modeled as a directed graph $G = (\mathbb{V}, \mathbb{A})$. The set of nodes $\mathbb{V}$ is made up of different types of nodes. It is distinguished between *entry nodes* $\mathbb{V}_+$ at which gas is supplied to the network, *exit nodes* $\mathbb{V}_-$ at which gas is discharged and *junctions* $\mathbb{V}_0$ that simply connect the network elements;

$$\mathbb{V} = \mathbb{V}_+ \cup \mathbb{V}_- \cup \mathbb{V}_0. \tag{3.1}$$

The network elements are divided into *active* and *passive* network elements, $\mathbb{A}_{\text{active}}$ and $\mathbb{A}_{\text{passive}}$, respectively. Active network elements are components that can be controlled by the network operator. In this subset of arcs, compressor stations $\mathbb{A}_{\text{cs}}$, control valves $\mathbb{A}_{\text{cv}}$ and valves $\mathbb{A}_{\text{vl}}$ are considered. Passive network elements cannot be controlled by the network operator. The corresponding components are pipes $\mathbb{A}_{\text{pi}}$, resistors $\mathbb{A}_{\text{re}}$ and short cuts $\mathbb{A}_{\text{sc}}$. In summary,

$$\mathbb{A} = \mathbb{A}_{\text{active}} \cup \mathbb{A}_{\text{passive}} \tag{3.2}$$

with

$$\mathbb{A}_{\text{active}} = \mathbb{A}_{\text{cs}} \cup \mathbb{A}_{\text{cv}} \cup \mathbb{A}_{\text{vl}} \quad \text{and} \quad \mathbb{A}_{\text{passive}} = \mathbb{A}_{\text{pi}} \cup \mathbb{A}_{\text{re}} \cup \mathbb{A}_{\text{sc}}. \tag{3.3}$$

To give a complete description of the model containing all considered network elements, some basic notation from graph and network theory is introduced. Individual arcs are denoted by $a \in \mathbb{A}$ or by $ij \in \mathbb{A}$ with tail i and head j. The sets δ_i^- and δ_i^+ are the sets of ingoing and outgoing arcs of node i, i.e.

$$\delta_i^- = \{a \in \mathbb{A} \colon a = ji\} \quad \text{and} \quad \delta_i^+ = \{a \in \mathbb{A} \colon a = ij\}. \tag{3.4}$$

Model Notation For the presentation of the model the following notation is fixed. Constraints are denoted by c and can be additionally indexed with a constraint index, a network element or sets thereof. For instance, c_a is the vector of constraints of the component model of arc $a \in \mathbb{A}$. In addition, super-indices are used to indicate the semantics of constraints. As a special suffix in

super-indices, s denotes a smoothed version of a constraint. Continuous variables are referred to as x and discrete ones as z. Additional sub-indices refer to network elements or sets thereof. For instance, x_a denotes the continuous variables of the component model of arc a.

3.2.3 Nodes

Nodes $i \in \mathbb{V}$ are used to connect network elements and to supply and discharge gas at entry and exit nodes. Because nodes do not have a capacity, they are modeled by the mass conservation constraint

$$0 = c_i^{\text{flow}}(x) = q_i + \sum_{a \in \delta_i^-} q_a - \sum_{a \in \delta_i^+} q_a. \tag{3.5}$$

Here, q_i is the amount of flow that is supplied or discharged at node i, i.e.

$$q_i \geq 0, \quad i \in \mathbb{V}_+, \tag{3.6a}$$

$$q_i \leq 0, \quad i \in \mathbb{V}_-, \tag{3.6b}$$

$$q_i = 0, \quad i \in \mathbb{V}_0. \tag{3.6c}$$

Moreover, every node $i \in \mathbb{V}$ has a given constant geodesic height h_i and a gas pressure variable p_i that is bounded due to technical or contractual restrictions;

$$p_i \in \left[p_i^-, p_i^+\right]. \tag{3.7}$$

Summarizing, the basic node model consists of the continuous variable

$$x_i = p_i \tag{3.8}$$

and the equality constraint

$$0 = c_i^{\text{flow}}(x). \tag{3.9}$$

Since there are no other effects to be modeled for junctions, (3.8) and (3.9) represent the model for junctions $i \in \mathbb{V}_0$.

Entries and Exits

For modeling entries and exits the basic node model is slightly extended. There has to be an additional specification of the flow demands, i.e. a bounded mass flow variable q_i is required for the supplied or discharged mass flow;

$$q_i \in \left[q_i^-, q_i^+\right]. \tag{3.10}$$

To make (3.5) a valid mass balance equation it is assumed that $q_i^- \geq 0$ holds for entries $i \in \mathbb{V}_+$ and $q_i^+ \leq 0$ for exits $i \in \mathbb{V}_-$ (cf. (3.6)). Concluding, for entry and exit nodes only the variable vector has to be extended to

$$x_i = \begin{pmatrix} p_i \\ q_i \end{pmatrix}. \tag{3.11}$$

3.2.4 Arcs

The variable vector of every arc $a = ij \in \mathbb{A}$ includes a mass flow variable q_a that is bounded for every arc in dependence on its technical data;

$$q_a \in \left[q_a^-, q_a^+\right]. \tag{3.12}$$

In the following, the network elements are discussed one after another and the corresponding element models are stated.

Pipes

Pipes are used to transport gas through a network and outnumber all other elements in real-world gas transport networks. The gas flow in pipes is governed by the Euler equations for compressible fluids in cylindrical pipes. This system of partial differential equations consists of the continuity equation (3.13a), the momentum equation (3.13b) and the energy equation (3.13c) (see [36, 78]):

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v)}{\partial x} = 0, \tag{3.13a}$$

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial p}{\partial x} + \frac{\partial(\rho v^2)}{\partial x} + g\rho\frac{\partial h}{\partial x} + \lambda(q)\frac{v|v|}{2D}\rho = 0, \tag{3.13b}$$

$$\begin{aligned} A\rho c_p\left(\frac{\partial T}{\partial t} + v\frac{\partial T}{\partial x}\right) - A\left(1 + \frac{T}{z}\frac{\partial z}{\partial T}\right)\frac{\partial p}{\partial t} - \\ Av\frac{T}{z}\frac{\partial z}{\partial T}\frac{\partial p}{\partial x} + A\rho v g\frac{\partial h}{\partial x} + \pi D c_{\mathrm{HT}}\left(T - T_{\mathrm{soil}}\right) = 0. \end{aligned} \tag{3.13c}$$

Here, the state variables are averaged over the cross-sectional area A of the pipe and only depend on the position $x \in [0, L]$, where L denotes the length of the pipe. The quantities appearing in the Euler equations are the gas velocity v, the gravitational acceleration g, the height $h(x)$ of the pipe and its diameter D. The slope of the pipe is assumed to be constant, i.e. $h(x)$ is linear. c_p denotes the specific isobaric (i.e. for constant pressure) heat capacity of real gas. Here, c_p is approximated by a constant value. A more detailed model can be found in [107]. c_{HT} is the heat transfer coefficient determined by the material of the pipe wall and T_{soil} is the temperature of the surrounding soil. Table 3.2 gives an overview of the considered quantities and their units.

In addition to the Euler equations, the gas state variables pressure, temperature and density are coupled by an equation of state. Here, the thermodynamical standard equation

$$\rho = \frac{p}{R_s z T} \tag{3.14}$$

is used, where $R_s = R/m$ is the specific gas constant that is defined by the universal gas constant R and the molar mass m of the gas.

Furthermore, the deviation between ideal and real gas is given by the compressibility factor z. There is no physically exact formula for z but a lot of empirically motivated ones (e.g. AGA[3]

[3] American Gas Association

Symbol	Explanation	Unit
D	Diameter	m
A	Cross-sectional area	m^2
k	Roughness of inner wall	m
$h(x)$	Geodesic height	m
c_{HT}	Heat transfer coefficient	$\mathrm{J\ m^{-2}\ K^{-1}\ s^{-1}}$
T_{soil}	Soil temperature	K
c_p	Specific heat capacity of real gas	$\mathrm{J\ m^{-1}\ s^{-1}}$
p_c	Pseudocritical pressure	Pa
T_c	Pseudocritical temperature	K
η	Dynamic viscosity	$\mathrm{kg\ m^{-1}\ s^{-1}}$
z	Compressibility factor	1
λ	Friction factor	1
g	Gravitational acceleration	$\mathrm{m\ s^{-2}}$
R_s	Specific gas constant	$\mathrm{J\ kg^{-1}\ K^{-1}}$

Table 3.2: Technical pipe parameters and physical quantities appearing in the pipe model.

[76, 112], Papay [94, 105], AGA DC 92 [115]). Here, the AGA formula

$$z = z(p,T) = 1 + 0.257p_r - 0.533\frac{p_r}{T_r} \tag{3.15}$$

is used, where p_r and T_r are the relative pressure and the relative temperature defined by

$$p_r = \frac{p}{p_c}, \quad T_r = \frac{T}{T_c}. \tag{3.16}$$

p_c is the pseudocritical pressure and T_c is the pseudocritical temperature. Figure 3.1 shows plots of the compressibility factor z in dependence of the gas pressure p for different constant values of T. By neglecting all partial derivatives with respect to time one obtains the stationary Euler equations

$$\frac{\partial(\rho v)}{\partial x} = 0, \tag{3.17a}$$

$$\frac{\partial p}{\partial x} + \frac{\partial(\rho v^2)}{\partial x} + g\rho\frac{\partial h}{\partial x} + \lambda(q)\frac{|v|v}{2D}\rho = 0, \tag{3.17b}$$

$$A\rho c_p v\frac{\partial T}{\partial x} - Av\frac{T}{z}\frac{\partial z}{\partial T}\frac{\partial p}{\partial x} + A\rho v g\frac{\partial h}{\partial x} + \pi D c_{\mathrm{HT}}\left(T - T_{\mathrm{soil}}\right) = 0. \tag{3.17c}$$

Since the mass flow is the basic variable on all arcs in the network, system (3.17) can finally be rewritten by using the mass flow q instead of the gas velocity v. Using the relationship $q = A\rho v$

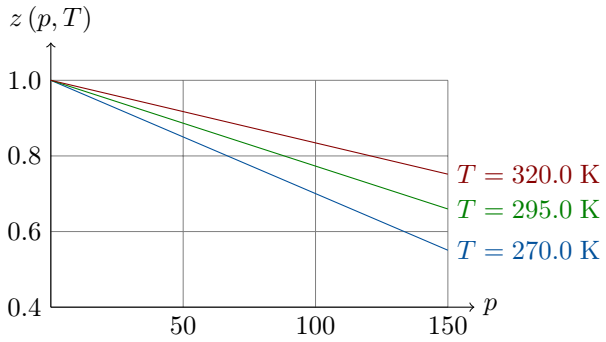

Figure 3.1: Compressibility factor computed by the AGA formula.

gives

$$\frac{1}{A}\frac{\partial q}{\partial x} = 0, \tag{3.18a}$$

$$\frac{\partial p}{\partial x} + \frac{q^2}{A^2}\frac{\partial}{\partial x}\frac{1}{\rho} + g\rho s + \lambda(q)\frac{|q|q}{2A^2 D\rho} = 0, \tag{3.18b}$$

$$qc_p\frac{\partial T}{\partial x} - \frac{qT}{\rho z}\frac{\partial z}{\partial T}\frac{\partial p}{\partial x} + qgs + \pi D c_{\mathrm{HT}}(T - T_{\mathrm{soil}}) = 0. \tag{3.18c}$$

$s := (h(L) - h(0))/L$ is the slope of the pipe. The latter system of ordinary differential equations is the basis of the following stationary model of gas flow in pipes. Figure 3.2 shows some typical pressure profiles, i.e. solutions of (3.18) along a pipe. There are different possibilities how to

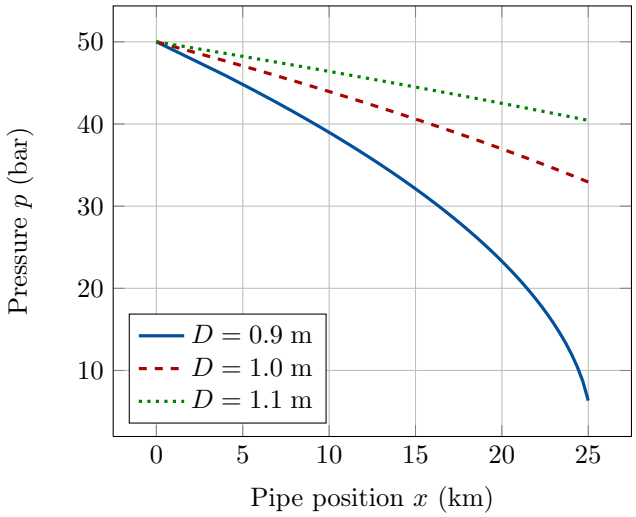

Figure 3.2: Profiles of gas pressure along three horizontal pipes ($L = 25$ km, $D \in \{0.9, 1.0, 1.1\}$ m, $k = 0.06$ mm; $q = 500$ kg s^{-1}).

incorporate the given system of ODEs into an optimization model. From (3.18a) it follows that the mass flow is constant in a pipe. This is already addressed by the introduction of a single mass flow variable q_a for every arc. Thus, one is left with the momentum and the energy equation. The

energy equation can be neglected since only the isothermal case is considered. For the momentum equation there exist a lot of well-known approximations. In the presented model, the quadratic approximation

$$0 = c_a^{\text{ploss}}(x) = p_j^2 - \left(p_i^2 - \Lambda_a q_a |q_a| \frac{e^{S_a} - 1}{S_a}\right) e^{-S_a} = 0 \tag{3.19}$$

is used with

$$\Lambda_a = \frac{L_a z_m T_m R_s \lambda_a}{A_a^2 D_a}, \quad S_a = \frac{2 L_a g s_a}{R_s z_m T_m}. \tag{3.20}$$

A derivation can be found in [78] or [6]. $z_m = z(p_m, T_m)$ is the mean compressibility factor that can be computed using a mean pressure p_m and a mean temperature $T_m = T$. Different equations for the mean pressure exist (cf. [107]). Here, the a-priori evaluable formula

$$p_m = \frac{1}{2}\left(\max\left(p_i^-, p_j^-\right) + \min\left(p_i^+, p_j^+\right)\right) \tag{3.21}$$

is used. Finally, a definition of the friction factor λ_a (cf. (3.20)) has to be given. Since the friction at rough inner pipe walls is one of the most important effects concerning the pressure drop in pipes, a highly accurate formulation should be integrated. First, the distinction between *laminar* and *turbulent* flows is introduced. With the Reynolds number

$$\text{Re}(q_a) = \frac{D_a}{A_a \eta} |q_a| \tag{3.22}$$

one defines these states as follows:

$$q \text{ is laminar}, \quad \text{if } \text{Re}(q) \leq \text{Re}_{\text{crit}}, \tag{3.23a}$$

$$q \text{ is turbulent}, \quad \text{if } \text{Re}(q) > \text{Re}_{\text{crit}}, \tag{3.23b}$$

with $\text{Re}_{\text{crit}} \approx 2320$. In (3.22) η denotes the dynamic viscosity of the gas that is approximated by a suitable constant in this thesis. For laminar flow the exact friction model

$$\lambda^{\text{HP}}(q_a) = \frac{64}{\text{Re}(q_a)} \tag{3.24}$$

of Hagen–Poiseuille [38] is used, whereas for turbulent flow the implicit equation

$$\frac{1}{\sqrt{\lambda^{\text{PC}}(q_a)}} = -2 \log_{10}\left(\frac{2.51}{\text{Re}(q_a)\sqrt{\lambda^{\text{PC}}(q_a)}} + \frac{k_a}{3.71 D_a}\right) \tag{3.25}$$

of Prandtl–Colebrook (cf. [27, 105]) is used. Finally, the constraint

$$0 = c_a^{\text{HPPC}}(x) = \lambda_a - \begin{cases} \lambda^{\text{HP}}(q_a), & q_a \leq \text{Re}_{\text{crit}}, \\ \lambda^{\text{PC}}(q_a), & q_a > \text{Re}_{\text{crit}} \end{cases} \tag{3.26}$$

is added to the model.

Summarizing, the pipe model reads

$$0 = c_a(x) = \begin{pmatrix} c_a^{\text{ploss}}(x) \\ c_a^{\text{HPPC}}(x) \end{pmatrix}, \quad x_a = \begin{pmatrix} q_a \\ \lambda_a \end{pmatrix}. \tag{3.27}$$

Reformulation of the Pipe Model The pipe model (3.27) has some undesirable properties. First, c_a^{HPPC} (cf. (3.26)) is discontinuous at the transition between laminar and turbulent flow. See Figure 3.3 for a plot of the corresponding functions. Secondly, c_a^{ploss} (cf. (3.19)) contains a second-order discontinuity due to the term $|q_a|q_a$. In the MINLP model, a global smooth approximation is used that addresses both problems. More precisely, the term $\lambda_a q_a|q_a|$ is approximated by

$$\phi(q_a) = r_a \left(\sqrt{q_a^2 + e_a^2} + b_a + \frac{c_a}{\sqrt{q_a^2 + d_a^2}} \right) q_a, \tag{3.28}$$

where

$$r_a = \left(2 \log_{10} \beta_a\right)^{-2}, \quad b_a = 2\delta_a, \quad c_a = \left(\ln \beta_a + 1\right) \delta_a^2 - \frac{e_a^2}{2} \tag{3.29}$$

and

$$\alpha_a = \frac{2.51 A_a \eta}{D_a}, \quad \beta_a = \frac{k_a}{3.71 D_a}, \quad \delta_a = \frac{2\alpha_a}{\beta_a \ln 10}. \tag{3.30}$$

The parameters e_a and d_a can be chosen arbitrarily to determine the slope of $\phi(q_a)$ at $q_a = 0$. This approximation is originally developed by Burgschweiger, Gnädig and Steinbach in [18] for water network optimization models and is afterwards adapted for gas transport models in [107]. In [18] it is also shown that ϕ provides an asymptotically correct second-order approximation of the combined Hagen–Poiseuille/Prandtl–Colebrook (HPPC) friction model.

Summarizing, (3.27) is replaced by the smoothed pipe model

$$0 = c_a^{\mathrm{smooth}}(x) = \begin{pmatrix} c_a^{\text{ploss-s}}(x) \\ c_a^{\text{HPPC-s}}(x) \end{pmatrix}, \quad x_a^{\mathrm{smooth}} = \begin{pmatrix} q_a \\ \phi_a \end{pmatrix} \tag{3.31}$$

with

$$0 = c_a^{\text{ploss-s}}(x) = p_j^2 - \left(p_i^2 - \tilde{\Lambda}_a \phi_a \frac{e^{S_a} - 1}{S_a} \right) e^{-S_a} = 0, \tag{3.32}$$

$$0 = c_a^{\text{HPPC-s}}(x) = \phi_a - r_a \left(\sqrt{q_a^2 + e_a^2} + b_a + \frac{c_a}{\sqrt{q_a^2 + d_a^2}} \right) q_a \tag{3.33}$$

and constants

$$\tilde{\Lambda}_a = \frac{L_a z_m T_m R_s}{A_a^2 D_a}, \quad S_a = \frac{2 L_a g s_a}{R_s z_m T_m}. \tag{3.34}$$

Resistors

Friction caused by rough inner walls of pipes is not the only reason for pressure loss in gas transport networks. Additional pressure loss can be caused by measurement devices, partly closed valves, filter systems, etc. For all of these effects neither an appropriate model nor appropriate data are available. For this reason, two simple pressure loss models are considered. The respective choice of the model depends on the data of the concrete gas transport network.

The first model (for resistors $a \in \mathbb{A}_{\mathrm{re}}^{\mathrm{pwc}}$) is characterized by a piecewise constant pressure loss

$$0 = c_a^{\text{ploss-pwc}}(x) = p_i - p_j - s_a \xi_a = 0, \tag{3.35}$$

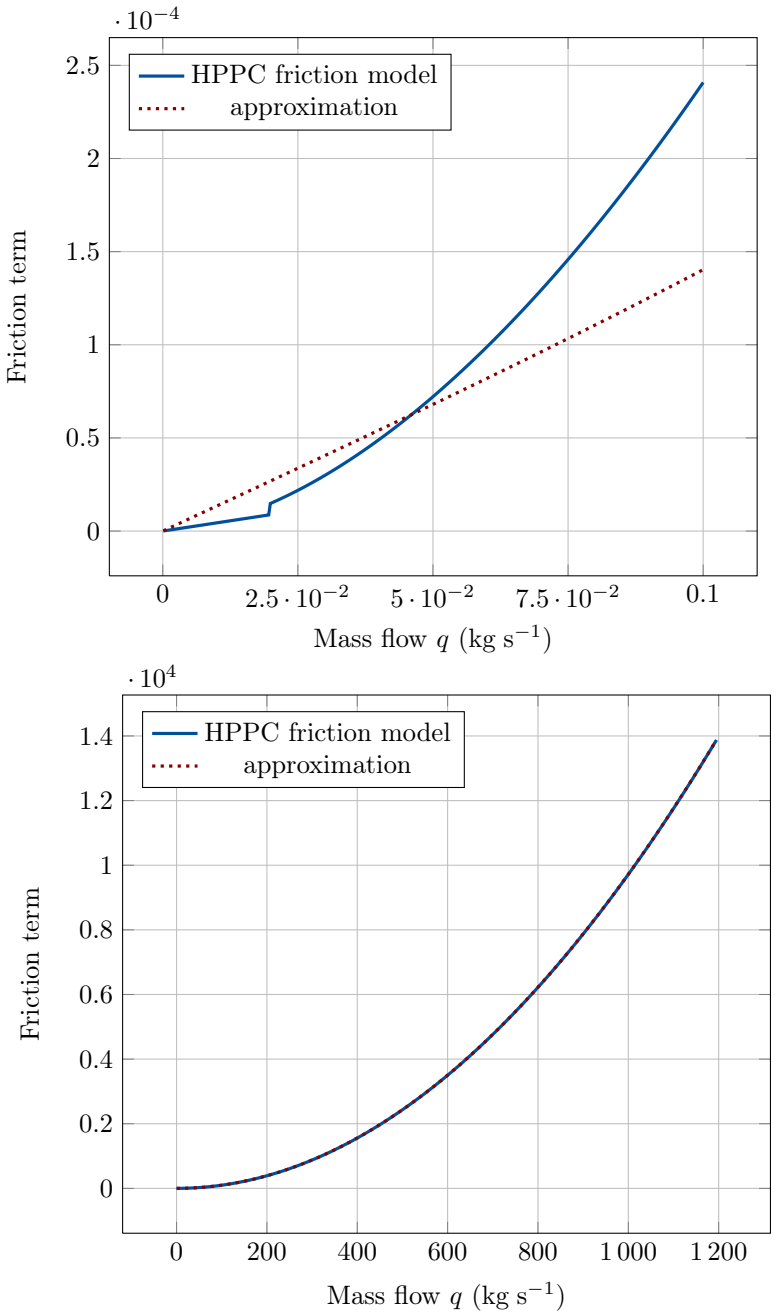

Figure 3.3: HPPC friction term and smooth approximation vs. mass flow (kg s^{-1}): transition from laminar to turbulent flow (upper figure) and highly turbulent flow (lower figure).

where the variable s_a is subject to the constraint

$$0 = c_a^{\text{flow-sign}}(x) = s_a - \text{sign}(q_a). \tag{3.36}$$

ξ_a is the constant and prescribed pressure loss at the resistor. Summarizing, the piecewise constant pressure loss model reads

$$0 = c_a(x) = \begin{pmatrix} c_a^{\text{ploss-pwc}}(x) \\ c_a^{\text{flow-sign}}(x) \end{pmatrix}, \quad x_a = \begin{pmatrix} q_a \\ s_a \end{pmatrix}. \tag{3.37}$$

The second model is slightly more complicated and states a pressure loss with respect to a nonlinear Darcy–Weisbach type model (see [38, 78]) for all resistors $a \in \mathbb{A}_{\text{re}}^{\text{pwq}}$:

$$0 = c_a^{\text{ploss-pwq}}(x) = p_i - p_j - \frac{8\zeta_a}{\pi^2 D_a^4} \frac{|q_a| q_a}{\rho_{a,k}}. \tag{3.38}$$

Here, ζ_a is a dimensionless resistance coefficient that determines the flow-dependent quadratic pressure loss at the resistor. D_a is a fictitious diameter. $\rho_{a,k}$ is the inflow gas density according to the equation of state (3.14);

$$0 = c_a^{\text{dens-in}}(x) = \rho_{a,k} - \frac{p_k}{R_s z_{a,k} T} \quad \text{with} \quad k := \begin{cases} i, & q_a \geq 0, \\ j, & q_a < 0. \end{cases} \tag{3.39}$$

Due to (3.39), the compressibility factor has to be evaluated at node k as well;

$$0 = c_a^{\text{z-in}}(x) = z_{a,k} - z(p_k, T). \tag{3.40}$$

In summary, the piecewise quadratic resistor model reads

$$0 = c_a(x) = \begin{pmatrix} c_a^{\text{ploss-pwq}}(x) \\ c_a^{\text{dens-in}}(x) \\ c_a^{\text{z-in}}(x) \end{pmatrix}, \quad x_a = \begin{pmatrix} q_a \\ \rho_{a,k} \\ z_{a,k} \end{pmatrix}. \tag{3.41}$$

Reformulation of the Resistor Models The stated resistor models are problematic due to the following reasons:

1. The discontinuous sign function in $c_a^{\text{flow-sign}}(x)$ (cf. (3.36)),
2. the second-order discontinuity in $c_a^{\text{ploss-pwq}}(x)$ (cf. (3.38)) and
3. the discontinuity due to the unknown inflow node k (cf. (3.39)).

First, the piecewise constant pressure loss model for resistors $a \in \mathbb{A}_{\text{re}}^{\text{pwc}}$ is discussed. The sign function is smoothed using the identity

$$\text{sign}(x) = \frac{x}{|x|} \tag{3.42}$$

and the absolute value function smoothing

$$|x| \approx \sqrt{x^2 + \tau} \tag{3.43}$$

yielding

$$0 = c_a^{\text{ploss-pwc-s}}(x) = p_i - p_j - \frac{q_a}{\sqrt{q_a^2 + \tau}}\xi_a = 0. \tag{3.44}$$

Figure 3.4 shows some sign function smoothings for different parameters τ. Thus, the smoothed

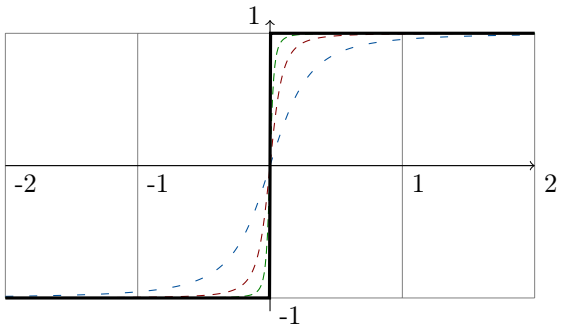

Figure 3.4: Smoothing of sign(x) with different values of τ (blue and loosely dashed for $\tau = 0.1$, red and dashed for $\tau = 0.01$, green and densely dashed for $\tau = 0.001$).

piecewise constant pressure loss model simplifies to

$$0 = c_a^{\text{smooth}}(x) = c_a^{\text{ploss-pwc-s}}(x), \quad x_a^{\text{smooth}} = q_a. \tag{3.45}$$

The piecewise quadratic resistor model is reformulated in a different way. The second-order discontinuity remains in the model since in Chapter 4 an interior-point method is developed that is able to solve problems containing continuous and nonsmooth but piecewise smooth constraints. Nevertheless, the discontinuity coming from the unknown inflow node k has to be removed. Here, the inflow gas density $\rho_{a,k}$ is approximated by a mean gas density $\rho_{a,m}$ given by

$$0 = c_a^{\text{dens-mean}}(x) = \rho_{a,m} - \frac{1}{2}(\rho_{a,\text{in}} + \rho_{a,\text{out}}). \tag{3.46}$$

For this, it is necessary to compute the gas densities at node i and j as well as the corresponding compressibility factors;

$$0 = c_a^{\text{dens-in}}(x) = \rho_{a,\text{in}} - \frac{p_i}{R_s z_{a,\text{in}} T}, \tag{3.47a}$$

$$0 = c_a^{\text{dens-out}}(x) = \rho_{a,\text{out}} - \frac{p_j}{R_s z_{a,\text{out}} T}, \tag{3.47b}$$

$$0 = c_a^{\text{z-in}}(x) = z_{a,\text{in}} - z(p_i, T), \tag{3.47c}$$

$$0 = c_a^{\text{z-out}}(x) = z_{a,\text{out}} - z(p_j, T). \tag{3.47d}$$

At last, $c_a^{\text{ploss-pwq}}(x)$ is replaced by

$$0 = c_a^{\text{ploss-pwq-s}}(x) = p_i - p_j - \frac{8\zeta_a}{\pi^2 D_a^4} \frac{|q_a| q_a}{\rho_{a,m}}, \tag{3.48}$$

yielding the reformulated model

$$0 = c_a^{\text{smooth}}(x) = \begin{pmatrix} c_a^{\text{ploss-pwq-s}}(x) \\ c_a^{\text{dens-mean}}(x) \\ c_a^{\text{dens-in}}(x) \\ c_a^{\text{dens-out}}(x) \\ c_a^{\text{z-in}}(x) \\ c_a^{\text{z-out}}(x) \end{pmatrix}, \quad x_a^{\text{smooth}} = \begin{pmatrix} q_a \\ \rho_{a,\text{in}} \\ \rho_{a,\text{out}} \\ \rho_{a,m} \\ z_{a,\text{in}} \\ z_{a,\text{out}} \end{pmatrix}. \tag{3.49}$$

Short Cuts

Short cuts $a = ij \in \mathbb{A}_{\text{sc}}$ are fictitious network elements that are mainly used to model complex supply or discharge situations at a single boundary node, where more than one customer supplies or discharges gas. They are modeled by the simple pressure propagation constraint

$$0 = c_a^{\text{press-sc}}(x) = p_i - p_j, \tag{3.50}$$

leading to the model

$$0 = c_a(x) = c_a^{\text{press-sc}}(x), \quad x_a = q_a. \tag{3.51}$$

Valves

Valves $a = ij \in \mathbb{A}_{\text{vl}}$ are active network elements that can be controlled by the network operator. From a stationary point of view, they can be in the states *open* or *closed*. With these states, network operators use valves to block gas flowing to certain regions of the network in order to control the overall flow situation in the network or, e.g. for maintenance work. Valves are of negligible length so that the assumption is made that the pressure loss at an open valve is negligible, too. Hence, open valves lead to identical pressures and a flow that is arbitrary within its technical bounds;

$$p_j = p_i, \quad q_a \in \left[q_a^-, q_a^+\right]. \tag{3.52}$$

Closed valves block the gas flow, leading to decoupled pressures;

$$q_a = 0, \quad p_i \in \left[p_i^-, p_i^+\right], \quad p_j \in \left[p_j^-, p_j^+\right]. \tag{3.53}$$

Both states can be modeled with one binary variable $z_a \in \{0,1\}$ and big-M constraints:

$$0 \leq c_a^{\text{flow-lb}}(x,z) = q_a - z_a q_a^-, \tag{3.54a}$$

$$0 \leq c_a^{\text{flow-ub}}(x,z) = -q_a + z_a q_a^+, \tag{3.54b}$$

$$0 \leq c_a^{\text{p-coupl-1}}(x,z) = M_{a,1}(1-z_a) - (p_j - p_i), \tag{3.54c}$$

$$0 \leq c_a^{\text{p-coupl-2}}(x,z) = M_{a,2}(1-z_a) + (p_j - p_i). \tag{3.54d}$$

It is easy to see that $z_a = 0$ corresponds to the state *closed* and that $z_a = 1$ corresponds to the state *open*. In summary, the MINLP model for valves reads

$$0 \leq c_a(x,z) = \begin{pmatrix} c_a^{\text{flow-lb}}(x,z) \\ c_a^{\text{flow-ub}}(x,z) \\ c_a^{\text{p-coupl-1}}(x,z) \\ c_a^{\text{p-coupl-2}}(x,z) \end{pmatrix}, \quad x_a = q_a. \tag{3.55}$$

Control Valves

Like valves, control valves $a = ij \in \mathbb{A}_{\text{cv}}$ are active elements. They can be in one of the states *closed*, *bypass* or *active*. In the closed state, control valves block the gas flow, leading to the same state model as for closed valves (cf. (3.53)). If a control valve is open, it can either be active or in bypass mode. The latter is modeled by the same state model as the open state of valves, cf. (3.52). If active, control valves can decrease the ingoing gas pressure by a certain amount Δp_a that is bounded due to technical limitations. In addition, a control valve can only regulate the pressure in a prescribed and fixed direction that is assumed to be the graph direction $i \to j$ of the arc modeling the control valve. Summarizing, the active state model reads

$$p_j = p_i - \Delta p_a, \quad \Delta p_a \in \left[\Delta p_a^-, \Delta p_a^+\right], \quad q_a \in \left[q_a^-, q_a^+\right] \cap \mathbb{R}_+. \tag{3.56}$$

In order to state an MINLP model of control valves, the binary variables $z_{a,1}$ and $z_{a,2}$ are introduced. $z_{a,1}$ determines whether the control valve is open ($z_{a,1} = 1$) or closed ($z_{a,1} = 0$). $z_{a,2}$ determines if the control valve is active ($z_{a,2} = 1$) or inactive, i.e. in bypass mode ($z_{a,2} = 0$). With the constraints

$$0 \leq c_a^{\text{flow-lb-open}}(x,z) = q_a - z_{a,1} q_a^-, \tag{3.57a}$$

$$0 \leq c_a^{\text{flow-ub-open}}(x,z) = -q_a + z_{a,1} q_a^+, \tag{3.57b}$$

$$0 \leq c_a^{\text{flow-lb-active}}(x,z) = q_a - (1 - z_{a,2})\, q_a^-, \tag{3.57c}$$

$$0 \leq c_a^{\text{p-coupl-1}}(x,z) = M_{a,1}(1-z_{a,1}) + \Delta p_a^+ z_{a,2} - (p_i - p_j), \tag{3.57d}$$

$$0 \leq c_a^{\text{p-coupl-2}}(x,z) = M_{a,2}(1-z_{a,1}) - \Delta p_a^- z_{a,2} - (p_j - p_i), \tag{3.57e}$$

$$0 \leq c_a^{\text{consistent-states}}(x,z) = z_{a,1} - z_{a,2}, \tag{3.57f}$$

the resulting mixed-integer model reads

$$0 \leq c_a(x,z) = \begin{pmatrix} c_a^{\text{flow-lb-open}}(x,z) \\ c_a^{\text{flow-ub-open}}(x,z) \\ c_a^{\text{flow-lb-active}}(x,z) \\ c_a^{\text{p-coupl-1}}(x,z) \\ c_a^{\text{p-coupl-2}}(x,z) \\ c_a^{\text{consistent-states}}(x,z) \end{pmatrix}, \quad x_a = \begin{pmatrix} q_a \\ \Delta p_a \end{pmatrix}, \quad z_a = \begin{pmatrix} z_{a,1} \\ z_{a,2} \end{pmatrix}. \tag{3.58}$$

Compressor Stations

From a technical point of view, compressor stations $a = ij \in \mathbb{A}_{\text{cs}}$ are the most complicated elements in gas transport networks. They are used to increase the ingoing gas pressure in order to overcome the pressure loss in pipes and resistors and thus to transport the gas over large distances.

Real-world stations consist of a finite set of compressor machines and drives. The compressor machines increase the gas pressure and the drives deliver the compressors with the power required for the compression process. Depending on the type of the compressor machines and drives, the operating ranges of the units are different. The complete description of all types that are relevant in practice is out of the scope of this thesis. For the moment, it is only remarked that a realistic description of a compressor station leads to a highly nonlinear and nonconvex mixed-integer model. It is referred to the upcoming publications [45, 67, 95, 107] for more detailed descriptions and concrete model formulations. A primal heuristic for solving the underlying MINLP of compressor stations with MPCC and NLP techniques is about to be published in [108].

For the purposes of this thesis, only a simplified model of compressor stations is considered. Here, compressor stations can also be in one of the states *closed*, *bypass* and *active*. The closed state and the bypass mode are modeled exactly as in the case of control valves. The active state is analogous to the corresponding active state of control valves (cf. (3.56)) but with the difference that the compressor machine *increases* the gas pressure;

$$p_j = p_i + \Delta p_a, \quad \Delta p_a \in \left[\Delta p_a^-, \Delta p_a^+\right], \quad q_a \in \left[q_a^-, q_a^+\right] \cap \mathbb{R}_+. \tag{3.59}$$

In analogy to the MINLP model of control valves one obtains the mixed-integer compressor station model

$$0 \leq c_a(x,z) = \begin{pmatrix} c_a^{\text{flow-lb-open}}(x,z) \\ c_a^{\text{flow-ub-open}}(x,z) \\ c_a^{\text{flow-lb-active}}(x,z) \\ c_a^{\text{p-coupl-1}}(x,z) \\ c_a^{\text{p-coupl-2}}(x,z) \\ c_a^{\text{consistent-states}}(x,z) \end{pmatrix}, \quad x_a = \begin{pmatrix} q_a \\ \Delta p_a \end{pmatrix}, \quad z_a = \begin{pmatrix} z_{a,1} \\ z_{a,2} \end{pmatrix}, \tag{3.60}$$

with constraints

$$0 \leq c_a^{\text{flow-lb-open}}(x,z) = q_a - z_{a,1} q_a^-, \tag{3.61a}$$

$$0 \leq c_a^{\text{flow-ub-open}}(x,z) = -q_a + z_{a,1} q_a^+, \tag{3.61b}$$

$$0 \leq c_a^{\text{flow-lb-active}}(x,z) = q_a - (1 - z_{a,2})\, q_a^-, \tag{3.61c}$$

$$0 \leq c_a^{\text{p-coupl-1}}(x,z) = M_{a,1}\,(1 - z_{a,1}) + \Delta p_a^+ z_{a,2} + (p_i - p_j)\,, \tag{3.61d}$$

$$0 \leq c_a^{\text{p-coupl-2}}(x,z) = M_{a,2}\,(1 - z_{a,1}) - \Delta p_a^- z_{a,2} + (p_j - p_i)\,, \tag{3.61e}$$

$$0 \leq c_a^{\text{consistent-states}}(x,z) = z_{a,1} - z_{a,2}. \tag{3.61f}$$

3.2.5 Model Summary

In the last sections a mixed-integer nonlinear model of the elements of gas transport networks is developed. Table 3.3 gives an overview of all network elements.

Network element	Symbol
Entries	$\mathbb{V}_+$
Exits	$\mathbb{V}_-$
Junctions	$\mathbb{V}_0$
Pipes	$\mathbb{A}_{\text{pi}}$
Short cuts	$\mathbb{A}_{\text{sc}}$
Resistors	$\mathbb{A}_{\text{re}} = \mathbb{A}_{\text{re}}^{\text{pwc}} \cup \mathbb{A}_{\text{re}}^{\text{pwq}}$
Valves	$\mathbb{A}_{\text{vl}}$
Control valves	$\mathbb{A}_{\text{cv}}$
Compressor stations	$\mathbb{A}_{\text{cs}}$

Table 3.3: Node types and network elements.

In the following, the component models are collected obtaining a complete MINLP model for gas transport networks that can be used as a model for the problem of validation of nominations. Since this problem is a feasibility problem, there is no objective function. The problem reads

$$\exists?\,(x,z) \in \mathbb{R}^{n_x} \times \{0,1\}^{n_z} : \quad c_{\mathcal{E}}\,(x) = 0, \quad c_{\mathcal{I}}\,(x,z) \geq 0, \tag{3.62}$$

where the sets of constraints are given by

$$c_{\mathcal{E}}(x) = \begin{pmatrix} c_{\mathbb{V}_+}(x) \\ c_{\mathbb{V}_-}(x) \\ c_{\mathbb{V}_0}(x) \\ c^{\text{smooth}}_{\mathbb{A}_{\text{pi}}}(x) \\ c^{\text{smooth}}_{\mathbb{A}^{\text{pwc}}_{\text{re}}}(x) \\ c^{\text{smooth}}_{\mathbb{A}^{\text{pwq}}_{\text{re}}}(x) \\ c_{\mathbb{A}_{\text{sc}}}(x) \end{pmatrix} \quad \text{and} \quad c_{\mathcal{I}}(x,z) = \begin{pmatrix} c_{\mathbb{A}_{\text{vl}}}(x,z) \\ c_{\mathbb{A}_{\text{cv}}}(x,z) \\ c_{\mathbb{A}_{\text{cs}}}(x,z) \end{pmatrix}. \tag{3.63}$$

The variable vectors are given by

$$x = \begin{pmatrix} x^{\text{smooth}}_{\mathbb{A}_{\text{pi}}} \\ x^{\text{smooth}}_{\mathbb{A}^{\text{pwc}}_{\text{pi}}} \\ x^{\text{smooth}}_{\mathbb{A}^{\text{pwq}}_{\text{pi}}} \\ x_{\mathbb{A}_{\text{sc}}} \\ x_{\mathbb{A}_{\text{vl}}} \\ x_{\mathbb{A}_{\text{cv}}} \\ x_{\mathbb{A}_{\text{cs}}} \end{pmatrix} \quad \text{and} \quad z = \begin{pmatrix} z_{\mathbb{A}_{\text{vl}}} \\ z_{\mathbb{A}_{\text{cv}}} \\ z_{\mathbb{A}_{\text{cs}}} \end{pmatrix}. \tag{3.64}$$

It should be remarked that this model is generic in the way that it can also be used for other problems in gas transport networks. For instance, by incorporating a reasonable objective function, the problem of *cost optimization* can be solved with almost the same set of constraints.

The model (3.62) of the problem of validation of nominations is a mixed-integer, nonsmooth, nonconvex and nonlinear feasibility problem. Nonlinear and nonconvex aspects mostly appear in constraints modeling the pressure loss in pipes and resistors. The compressor station model is drastically simplified. A complete model of the technical and physical behavior of these entities would lead to additional nonlinear and nonconvex constraints (see [107]). The problem description on which (3.62) is based on also contains a lot of nonsmooth aspects, e.g. the HPPC friction model or the model of resistors with piecewise constant pressure loss. Most of them are smoothed in (3.62). One remaining nonsmoothness is the second-order discontinuity of the piecewise quadratic pressure loss equation of resistors. In [107] it is also discussed that a non-isothermal model would require additional nonsmooth constraints. Furthermore, the reformulation technique described in the next section may introduce additional nonsmooth aspects when applied to (3.62). Finally, it is mentioned that all discrete aspects of the model appear in the active elements.

(3.62) is a significantly simplified model of the problem of validation of nominations. Nevertheless, it is a very hard mixed-integer, nonsmooth, nonconvex and nonlinear optimization problem and the practical experience shows that it is not solvable with state-of-the-art solvers for mixed-integer nonlinear programming. Numerical experiments will be published in [95].

3.3 An MPCC Approach for MINLPs in Gas Network Planning

Mixed-integer nonlinear optimization is a comfortable tool to model real-world problems from industry, finance, etc. However, state-of-the-art software packages for this class of optimization problems are often far away from solving problems in the size in which they appear in practice. In addition, even for small- or medium-scaled problems, nowadays software packages are also not able to compete with solver packages for mixed-integer linear problems (MIP) or nonlinear optimization problems (NLP) in terms of performance and robustness. Hence, a lot of real-world optimization projects develop problem-specific solution strategies that follow one of the two approaches: the *MIP-driven approach* or the *NLP-driven approach*. Unfortunately, both approaches often follow the *law of the hammer* (see [84]):

> "If all you have is a hammer, everything looks like a nail."

Experts in the MIP world often try to get rid of the nonlinearities in the MINLP under consideration. This approach leads to large-scale mixed-integer linear models. The number of discrete variables is often drastically increased due to the techniques required for linearizing the nonlinear constraints and the objective function (see [29, 79, 87, 129] and the references therein). In contrast to that, experts in the NLP world try to get rid of the discrete aspects of the problem so that they end up with a continuous model without discrete variables.

Obviously, both approaches have their specific advantages and drawbacks. The MIP approach benefits if the underlying MINLP is dominated by discrete variables and has only a few nonlinear aspects. In this case, the linearization techniques increase the number of discrete variables only slightly, yielding an MIP reformulation with admissible size. Furthermore, the MIP approach has the important advantage that it delivers global optimal solutions (of the linearized problem). The NLP approach only delivers local optimal solutions of the reformulated problem without discrete variables. It is in advantage if the underlying MINLP incorporates only a few discrete variables and contains a lot of nonlinear aspects. An additional pro for the NLP approach is that the reformulated NLPs can mostly be solved faster than the corresponding reformulated MIPs. However, this is often affected drastically by the choice of starting points and by the scaling of the problems. This is not the case (at least not to the same degree) for state-of-the-art MIP software. Finally, one of the main advantages of the MIP approach, namely its property of delivering global optimal solutions, is not as heavily weighted for feasibility problems (like the problem of validation of nominations) as for optimization problems.

In this section an alternative approach is presented that is referred to as the MPCC-based approach. For this, a definition of a certain subclass of MINLPs is given and it is shown how this

subclass can be tackled with MPCC-based modeling techniques in Section 3.3.1. Subsequently, it is shown in Section 3.3.2 that the problem of validation of nominations is contained in the considered subclass of MINLPs. Finally, the reformulation technique is applied to the model that is developed in the last section.

The techniques presented in this section are about to be published in [108]. In addition, [108] contains a heuristic solution approach for a nonsmooth MPCC model that incorporates a more detailed model of compressor stations.

3.3.1 A Reformulation Technique for 2-State-MINLPs

This section considers MINLPs of the form (2.8). In what follows, a subclass of these problems is defined and it is shown how problems of this subclass can be restated without requiring discrete variables. Without loss of generality, the MINLPs are given in binary form, i.e. all discrete variables are restricted to $\{0, 1\}$.

Recent approaches from the field of continuous reformulation of discrete-continuous problems can be found in [68, 116]. Therein, the authors make use of NCP-functions, in particular of the Fischer–Burmeister function (2.21). The Fischer–Burmeister function can be used to shrink the feasible set of a continuous variable x to $B := \{0, 1\}$ or $\tilde{B} := \{0, y \colon y \geq 1\}$. A different approach aims to model the feasible sets B or $\tilde{B}$ with complementarity constraints: A binary variable z can be restated by introducing an auxiliary continuous variable x subject to the constraints

$$x\,(1 - x) = 0, \quad x \in [0, 1]\,. \tag{3.65}$$

(3.65) is a poor formulation for two reasons. First of all, it is nonconvex. Secondly, its feasible set only consists of two disjoint points. Especially the latter makes it very hard for standard solvers to solve problems incorporating formulations like (3.65). With an additional continuous variable y that is subject to the constraint $x + y = 1$ one obtains

$$xy = 0, \quad x, y \geq 0, \tag{3.66}$$

and sees that (3.65) is equivalent to the MPCC standard formulation (cf. (2.13)). Here, the cases $x = 0, y > 0$ and $x > 0, y = 0$ correspond to $x = 0$ and $x = 1$, respectively. Unfortunately, the better numerical properties of (3.66) come at the price of an undecided third state, i.e. $x = y = 0$.

A lot of real-world applications share a feature that can be exploited in a way that is more useful in practice: The alternatives of the discrete aspects can be represented as subsets of a space of continuous variables. Formally, let A be a certain aspect of a model that consists of a set of states $A_1, \ldots, A_a$. Moreover, assume that there are finite sets of variables and constraints

$$x_A \in \mathbb{R}^{n_A}, \quad c_{\mathcal{E},A_i}(x_A) = 0, \quad c_{\mathcal{I},A_i}(x_A) \geq 0, \quad i = 1, \ldots, a, \tag{3.67}$$

that are used to model the states. Going further, let

$$z_A = (z_{A_i})_{i=1}^{a} \in \{0,1\}^a \tag{3.68}$$

be a vector of binary variables that are used to switch between the different states of the model aspect A. Finally, assume that the model is stated such that

$$\text{model aspect } A \text{ is in state } A_i \quad \Longleftrightarrow \quad z_{A_i} = 1, \quad z_{A_j} = 0 \quad \forall j \neq i. \tag{3.69}$$

A mixed-integer nonlinear formulation for the described situation using big-M constraints and a SOS-1 constraint reads

$$M_{\mathcal{E},A_i}\left(1 - z_{A_i}\right) - c_{\mathcal{E},A_i}\left(x_A\right) \geq 0, \quad i = 1,\ldots,a, \tag{3.70a}$$

$$M_{\mathcal{E},A_i}\left(1 - z_{A_i}\right) + c_{\mathcal{E},A_i}\left(x_A\right) \geq 0, \quad i = 1,\ldots,a, \tag{3.70b}$$

$$M_{\mathcal{I},A_i}\left(1 - z_{A_i}\right) + c_{\mathcal{I},A_i}\left(x_A\right) \geq 0, \quad i = 1,\ldots,a, \tag{3.70c}$$

$$\sum_{i=1}^{a} z_{A_i} = 1, \quad z_{A_i} \in \{0,1\}, \quad i = 1,\ldots,a. \tag{3.70d}$$

If desired, (3.70) can also be stated as an equivalent general disjunctive program (GDP) [50, 100];

$$\bigvee_{i=1}^{a} \begin{pmatrix} z_{A_i} = 1 \\ c_{\mathcal{E},A_i}\left(x_A\right) = 0 \\ c_{\mathcal{I},A_i}\left(x_A\right) \geq 0 \end{pmatrix}. \tag{3.71}$$

The following introduces the concept of *non-disjunctive* states, i.e. states for which the feasible sets are not disjoint. The formal definition requires the definition of *characteristic functions*.

Definition 11 (Characteristic Function)**.** *Let A be a model aspect with states $A_i, i = 1,\ldots,a$, represented by variables and constraints as in* (3.67) *and* (3.68)*. A function $\chi_{A_i} : \mathbb{R}^{n_A} \to \mathbb{R}$ is called a* characteristic function *of state A_i if*

$$\chi_{A_i}(x) = 0, \quad \text{if } c_{\mathcal{E},A_i}(x) = 0 \text{ and } c_{\mathcal{I},A_i}(x) \geq 0, \tag{3.72a}$$

$$\chi_{A_i}(x) \neq 0, \quad \text{else.} \tag{3.72b}$$

Definition 12 (Non-Disjunctive States)**.** *Two states A_i and A_j are called* non-disjunctive *if there exists $x \in \mathbb{R}^{n_A}$ such that*

$$\chi_{A_i}(x) = \chi_{A_j}(x) = 0. \tag{3.73}$$

With the last definitions, one can define the subclass of MINLPs that is considered in the following.

Definition 13 (2-State-MINLP)**.** *MINLPs in which all discrete aspects have two non-disjunctive states are called* 2-state-MINLPs.

As a direct consequence of Definition 12 one gets the following lemma:

Lemma 3. *Let A_1, A_2 be non-disjunctive states of a model aspect A that is modeled with variables (x_A, z_A) and sets of constraints $c_{\mathcal{E},A_i}, c_{\mathcal{I},A_i}, i = 1, 2$. Let χ_{A_i} denote corresponding characteristic functions. Then the MINLP model* (3.70) *of A can be equivalently replaced by the MPCC model*

$$\chi_{A_1}(x)\,\chi_{A_2}(x) = 0. \tag{3.74}$$

Proof. Let x_A^* be a solution of the reformulated MPCC model and let $\chi_{A_1}(x_A^*) = 0$. Then, using (3.72) it follows that $c_{\mathcal{E},A_1}(x_A^*) = 0$ and $c_{\mathcal{I},A_1}(x_A^*) \geq 0$. Setting $z_{A_1} = 1$ and $z_{A_2} = 0$ one gets a feasible solution of (3.70). The case $\chi_{A_2}(x_A^*) = 0$ and the reverse direction can be proven analogously. □

In the case of 2-state MINLPs, the undecided state of (3.66) does not lead to a problem since the main property of non-disjunctive states is that their continuous variables can be identical.

Since the problem of validation of nominations is a feasibility problem, one is mainly interested in the relationship between the feasible points of the 2-state-MINLP and the feasible points of the reformulated MPCC model. The last lemma directly implies the following corollary.

Corollary 1. *Let P be a 2-state-MINLP and let Q be an MPCC reformulation as described above. Then for every Q-feasible point x_Q^* there exists a P-feasible point (x_P^*, z_P^*). Conversely, if there is no Q-feasible point, P is also infeasible. Thus, there is a one-to-one correspondence of feasible points between 2-state-MINLP problems and their MPCC reformulations.*

3.3.2 The Problem of Validation of Nominations Revisited

With the reformulation technique presented in the last section, it is now possible to restate the MINLP formulation (3.62) of the model of the problem of validation of nominations. It turns out that (3.62) is a 2-state-MINLP according to Definition 13. As discussed in Section 3.2.5, all discrete aspects of the problem appear in the component models of active network elements. Thus, the only component models that have to be reformulated are the ones of valves, control valves and compressor stations.

Reformulation of the Valve Model

One can easily see that the valve model (3.55) fits into the concept of non-disjunctive model aspects. The model aspect A (= *valve*) has the two states $A_1 =$ *open* and $A_2 =$ *closed*. They are non-disjunctive if

$$0 \in \left[q_a^-, q_a^+\right] \quad \text{and} \quad \left[p_i^-, p_i^+\right] \cap \left[p_j^-, p_j^+\right] \neq \emptyset. \tag{3.75}$$

Notice that the state of the valve can be decided a-priori if (3.75) does not hold. Thus, the assumption (3.75) is without loss of generality. The characteristic functions are

$$\chi_a^{\text{open}}(x) = p_j - p_i, \quad \chi_a^{\text{closed}}(x) = q_a. \tag{3.76}$$

According to the last section the 2-state-MINLP (3.55) can be equivalently reformulated using the complementarity constrained model

$$0 = c_a^{\text{mpcc}}(x) = c_a^{\text{vl-state}}(x) = \chi_a^{\text{open}}(x)\,\chi_a^{\text{closed}}(x), \quad x_a^{\text{mpcc}} = q_a. \tag{3.77}$$

The latter formulation offers two advantages: No binary variables are required and the number of constraints reduces from four to one.

Reformulation of the Control Valve Model

The control valve model (3.58) is a 2-state-MINLP only if $\Delta p_a^- = 0$ holds. However, this appears to be a moderate restriction in practice: All control valves of the real-world gas transport network that is considered in Chapter 6 satisfy $\Delta p_a^- = 0$. With this, control valves can be modeled as a model aspect with two non-disjunctive states $A_1 = \textit{open}$ and $A_2 = \textit{closed}$. The characteristic functions are

$$\chi_a^{\text{open}}(x) = p_j - p_i + \Delta p_a, \quad \chi_a^{\text{closed}}(x) = q_a. \tag{3.78}$$

Hence, the state *open* can be further distinguished into the cases *active* or *bypass* depending on the value of the pressure decrease variable Δp_a. In summary, the MPCC formulation reads

$$0 = c_{\mathcal{E},a}^{\text{mpcc}}(x) = c_a^{\text{cv-state}}(x) = \chi_a^{\text{open}}(x)\,\chi_a^{\text{closed}}(x), \tag{3.79a}$$

$$0 \le c_{\mathcal{I},a}^{\text{mpcc}}(x) = c_a^{\text{cv-active-state}}(x) = \Delta p_a q_a. \tag{3.79b}$$

(3.79b) models that the control valve can only decrease the pressure for positive flow. The variable vector reduces to the continuous variable vector of the MINLP model of control valves;

$$x_a = \begin{pmatrix} q_a \\ \Delta p_a \end{pmatrix}. \tag{3.80}$$

Reformulation of the Compressor Station Model

The MINLP model (3.60) of compressor stations is the same as for control valves except for the sign of the variable Δp_a. Hence, the reformulation is the same except for this sign, too. In summary, one has

$$0 = c_{\mathcal{E},a}^{\text{mpcc}}(x) = c_a^{\text{cs-state}}(x) = \chi_a^{\text{open}}(x)\,\chi_a^{\text{closed}}(x), \tag{3.81a}$$

$$0 \le c_{\mathcal{I},a}^{\text{mpcc}}(x) = c_a^{\text{cs-active-state}}(x) = \Delta p_a q_a, \tag{3.81b}$$

$$x_a = \begin{pmatrix} q_a \\ \Delta p_a \end{pmatrix}, \tag{3.81c}$$

with the modified characteristic function

$$\chi_a^{\text{open}}(x) = p_j - p_i - \Delta p_a \tag{3.82}$$

for the state *open*.

Summary of the Reformulated MPCC Model

Collecting all reformulated component models of the problem of validation of nominations leads to the MPCC model

$$\exists ? x \in \mathbb{R}^{n_x} : \quad c_{\mathcal{E}}^{\text{mpcc}}(x) = 0, \quad c_{\mathcal{I}}^{\text{mpcc}}(x) \geq 0, \tag{3.83}$$

with the sets of constraints

$$c_{\mathcal{E}}^{\text{mpcc}}(x) = \begin{pmatrix} c_{\mathbb{V}_+}(x) \\ c_{\mathbb{V}_-}(x) \\ c_{\mathbb{V}_0}(x) \\ c_{\mathbb{A}_{\text{pi}}}^{\text{smooth}}(x) \\ c_{\mathbb{A}_{\text{re}}^{\text{pwc}}}^{\text{smooth}}(x) \\ c_{\mathbb{A}_{\text{re}}^{\text{pwq}}}^{\text{smooth}}(x) \\ c_{\mathbb{A}_{\text{sc}}}(x) \\ c_{\mathbb{A}_{\text{vl}}}^{\text{mpcc}}(x) \\ c_{\mathcal{E},\mathbb{A}_{\text{cv}}}^{\text{mpcc}}(x) \\ c_{\mathcal{E},\mathbb{A}_{\text{cs}}}^{\text{mpcc}}(x) \end{pmatrix} \quad \text{and} \quad c_{\mathcal{I}}^{\text{mpcc}}(x) = \begin{pmatrix} c_{\mathcal{I},\mathbb{A}_{\text{cv}}}^{\text{mpcc}}(x) \\ c_{\mathcal{I},\mathbb{A}_{\text{cs}}}^{\text{mpcc}}(x) \end{pmatrix}. \tag{3.84}$$

By taking a closer look to the MPCC model (3.83) one sees that it is not in the MPCC standard form (2.13) since the non-negativity constraints (2.13d) are not contained in (3.83). The reason for this is that the characteristic functions that are used to build up the complementarity constraints do not have to be non-negative. For instance, the characteristic functions

$$\chi_a^{\text{open}}(x) = p_j - p_i \quad \text{and} \quad \chi_a^{\text{closed}}(x) = q_a \tag{3.85}$$

for valves $a \in \mathbb{A}_{\text{vl}}$ can take negative values for negative mass flow q_a or decoupled pressures satisfying $p_i > p_j$.

Unfortunately, the standard regularization techniques (cf. Section 2.2) are not valid in this situation. To overcome this drawback, two different strategies may be employed:

1. Squaring all characteristic functions χ that are part of $c_{\mathcal{E}}^{\text{mpcc}}(x)$. One can easily see that this leads to non-negative complementarity pairings and does not harm property (3.72) in Definition 11.

2. Replacing all characteristic functions χ by their absolute values $|\chi|$. Again, the non-negativity constraints of the MPCC standard form are obviously satisfied and (3.72) in Definition 11 is still valid.

The first approach leads to ill-conditioned KKT systems, especially when a penalization scheme is used to regularize the MPCC. The second approach leads to better conditioned systems but incorporates much more nonsmooth aspects. Both approaches and their properties are discussed in detail in Chapter 6.

Chapter 4

Interior-Point Methods

Interior-point (or *barrier*) *methods* emerged in the 1980s. They came up due to the search for algorithms solving linear programs (LP) with a better theoretical complexity than the simplex method. It is well-known that the complexity of the simplex method can be exponential in the size of the problem (cf. [91]). The earliest proposed algorithm for solving LPs with a polynomial worst-case complexity was the *ellipsoid method* by Khachiyan [66]. However, this method turned out to be impractical and was not competitive with the simplex method. In 1984, Karmarkar [63] proposed a new method that shares the complexity property of Khachiyan's algorithm but also has a good practical performance. Since then a lot of research activity came up leading to a wide range of theoretical results and implementations. One decade after Karmarkar's publication a subclass of interior-point methods arose that are known as *primal-dual* methods. This subclass turned out to be the most efficient instantiation of interior-point methods and is the topic of this chapter.

A useful property of interior-point methods is that their basic algorithmic framework is almost the same for most of the standard classes of optimization problems. This motivates the main idea of this thesis – namely to develop a generic and highly flexible interior-point framework that can be used to solve different classes of optimization problems like LP, quadratic programming (QP), (nonconvex) NLP as well as MPCC and nonsmooth nonlinear problems.

The outline of this chapter is as follows. Section 4.1 describes a standard interior-point method for nonlinear programming including standard techniques like globalization strategies, reduction techniques for KKT systems, etc. This method is strongly oriented towards the interior-point code Ipopt of Andreas Wächter [132, 134]. Since it turned out in the past that Ipopt is a very efficient and reliable interior-point method its main algorithmic principles are chosen to be the backbone of the method developed for this thesis. Additionally, useful aspects of other interior-point methods like LOQO [127] are integrated into the framework. In Section 4.2 the subclass of problems with locatable and separable nonsmoothness is defined and techniques for solving this

practically important subclass are developed. To the best of the author's knowledge, the problem classification as well as the developed solution strategies are not considered in the literature before. In Section 4.3 it is stated how the generic interior-point framework can be extended to solve complementarity constrained problems. The presented method strongly relies on the paper [74] by Leyffer et al. Here, the focus is on a careful algorithmic design that allows to combine both the extensions for nonsmooth problems and the extensions for MPCCs. This combination is finally presented in Section 4.4.

Notation Throughout this chapter the following notation is used. A vector $v \in \mathbb{R}^n$ is denoted by a small letter and vector components are denoted by sub-indices, e.g. v_i is the i-th component of the vector v. A diagonal matrix build from a vector is denoted by the corresponding capital letter, i.e. $V = \operatorname{diag}(v_1, \ldots, v_n) \in \mathbb{R}^{n \times n}$. Vectors made up of sub-vectors are written as ordered lists of sub-vectors, e.g. $v = (x, y)$ stands for $v = (x^T, y^T)^T$. e is the vector of ones in appropriate dimension, i.e. $e = (1, \ldots, 1)^T$. f denotes the objective function or portions of it. As already introduced in Chapter 2, constraint vectors c are indexed by index sets $\mathcal{E}$ and $\mathcal{I}$ for equality and inequality constraints, respectively. Iteration numbers are denoted by braced super-indices. For example, $y^{(k)}$ stands for the vector y in iteration k. Finally, if $g(x) : \mathbb{R}^n \to \mathbb{R}^k$ is an arbitrary function and $v = (x, y) \in \mathbb{R}^{n+m}$ is a vector containing the sub-vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, the expression $g(v)$ is interpreted as $g(x)$ if the original argument x of g is clear from the context.

4.1 Interior-Point Methods for Nonlinear Optimization

This section describes an interior-point method for solving nonlinear and nonconvex constrained optimization problems of the form

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{4.1a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{4.1b}$$

$$c_{\mathcal{I}}(x) \in [r_l, r_u], \tag{4.1c}$$

$$x \in [b_l, b_u], \tag{4.1d}$$

where inequality constraints are split into the lower and upper range constraints (4.1c) and the lower and upper variable bounds (4.1d). The problem data consists of the objective function $f : \mathbb{R}^n \to \mathbb{R}$, m equality constraints

$$c_{\mathcal{E}} = (c_1(x), \ldots, c_m(x))^T : \mathbb{R}^n \to \mathbb{R}^m, \tag{4.2}$$

k inequality constraints

$$c_{\mathcal{I}} = (c_{m+1}(x), \ldots, c_{m+k}(x))^T : \mathbb{R}^n \to \mathbb{R}^k, \tag{4.3}$$

inequality ranges $r = (r_l, r_u) \in \mathbb{R}^{2k}$ and variable bounds $b = (b_l, b_u) \in \mathbb{R}^{2n}$. If it is not explicitly stated otherwise, $f, c_\mathcal{E}, c_\mathcal{I}$ are assumed to be twice continuously differentiable in this section.

The main idea of interior-point methods is to reformulate (4.1) by using slack variables to translate inequality into equality constraints. With slack variables $s = (s_l, s_u) \in \mathbb{R}^{2n}$ for the variable bounds and $t = (t_l, t_u) \in \mathbb{R}^{2k}$ for the range constraints one obtains

$$\begin{aligned}
\min_{(x,s,t)\in\mathbb{R}^{n_p}} \quad & f(x) && \text{(4.4a)}\\
\text{s.t.} \quad & \alpha := c_\mathcal{E}(x) = 0, && \text{(4.4b)}\\
& \rho_l := c_\mathcal{I}(x) - t_l - r_l = 0, && \text{(4.4c)}\\
& \rho_u := -c_\mathcal{I}(x) - t_u + r_u = 0, && \text{(4.4d)}\\
& \beta_l := x - s_l - b_l = 0, && \text{(4.4e)}\\
& \beta_u := -x - s_u + b_u = 0, && \text{(4.4f)}\\
& s_l, s_u, t_l, t_u \geq 0, && \text{(4.4g)}
\end{aligned}$$

with $n_p = 3n + 2k$. The remaining inequality constraints are the simple variable bounds (4.4g) for the slack variables s, t. The Lagrangian of (4.4) reads

$$\begin{aligned}
\mathcal{L}(y, \lambda, \xi) = {} & f(x)\\
& - z^T c_\mathcal{E}(x)\\
& - v_l^T \left(c_\mathcal{I}(x) - t_l - r_l\right) - v_u^T \left(-c_\mathcal{I}(x) - t_u + r_u\right)\\
& - u_l^T \left(x - s_l - b_l\right) - u_u^T \left(-x - s_u + b_u\right)\\
& - \lambda_l^T s_l - \lambda_u^T s_u - \xi_l^T t_l - \xi_u^T t_u.
\end{aligned} \tag{4.5}$$

$z \in \mathbb{R}^m$ is the vector of dual multipliers of the equality constraints, $v = (v_l, v_u) \in \mathbb{R}^{2k}$ is the vector of dual multipliers of the lower and upper range constraints and $u = (u_l, u_u) \in \mathbb{R}^{2n}$ is the vector of dual multipliers of the lower and upper variable bounds. $\lambda = (\lambda_l, \lambda_u) \in \mathbb{R}^{2n}$ and $\xi = (\xi_l, \xi_u) \in \mathbb{R}^{2k}$ are the vectors of dual multipliers of the positivity constraints (4.4g). For better reading, $y_\text{pri} := (x, s, t) \in \mathbb{R}^{n_p}$ denotes the vector of primal variables, $y_\text{dual} := (z, u, v) \in \mathbb{R}^{n_d}, n_d = m + 2n + 2k$, denotes the vector of dual variables and $y := (y_\text{pri}, y_\text{dual}) \in \mathbb{R}^N, N = n_p + n_d = 5n + m + 4k$.

As usual for barrier methods (cf. [91]), (4.4) is finally translated into the *μ-parameterized log-barrier problem*

$$\begin{aligned}
\min_{y_\text{pri}\in\mathbb{R}^{n_p}} \quad & \varphi_\mu := f(x) - \mu \left[\sum_{i=1}^{n} \left(\ln s_{l_i} + \ln s_{u_i}\right) + \sum_{i=1}^{k} \left(\ln t_{l_i} + \ln t_{u_i}\right)\right] && \text{(4.6a)}\\
\text{s.t.} \quad & \alpha(x) = \rho_l(x, t_l) = \rho_u(x, t_u) = \beta_l(x, s_l) = \beta_u(x, s_u) = 0, && \text{(4.6b)}
\end{aligned}$$

that is solved for a sequence of *barrier parameters* $\mu > 0$ converging to zero. It is well-known that this barrier approach can also be interpreted equivalently as applying a homotopy method to the

μ-perturbed KKT conditions of the reformulated problem (4.4) (while maintaining strict positivity of the slack variables s, t). These conditions include *primal feasibility* (4.4b)–(4.4g), *dual feasibility*

$$\nabla_x \mathcal{L} = g - \nabla c_{\mathcal{E}}(x)^T z - \nabla c_{\mathcal{I}}(x)^T (v_l - v_u) - (u_l - u_u) = 0, \tag{4.7a}$$

$$\nabla_{s_l} \mathcal{L} = u_l - \lambda_l = 0, \tag{4.7b}$$

$$\nabla_{s_u} \mathcal{L} = u_u - \lambda_u = 0, \tag{4.7c}$$

$$\nabla_{t_l} \mathcal{L} = v_l - \xi_l = 0, \tag{4.7d}$$

$$\nabla_{t_u} \mathcal{L} = v_u - \xi_u = 0, \tag{4.7e}$$

μ-perturbed complementarity

$$S_l \Lambda_l e = S_u \Lambda_u e = T_l \Xi_l e = T_u \Xi_u e = \mu e \tag{4.8}$$

and *non-negativity* of dual multipliers corresponding to the inequality constraints, i.e.

$$\lambda_l, \lambda_u, \xi_l, \xi_u \geq 0. \tag{4.9}$$

In (4.7a), $g := \nabla f(x) \in \mathbb{R}^n$ denotes the gradient of the objective function;

$$g(x) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right)^T, \tag{4.10}$$

and $\nabla c_{\mathcal{E}}(x) \in \mathbb{R}^{m \times n}$ and $\nabla c_{\mathcal{I}}(x) \in \mathbb{R}^{k \times n}$ are the Jacobians of the constraints;

$$\nabla c_{\mathcal{E}}(x) = \begin{bmatrix} \frac{\partial c_1}{\partial x_1} & \cdots & \frac{\partial c_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial c_m}{\partial x_1} & \cdots & \frac{\partial c_m}{\partial x_n} \end{bmatrix}, \quad \nabla c_{\mathcal{I}}(x) = \begin{bmatrix} \frac{\partial c_{m+1}}{\partial x_1} & \cdots & \frac{\partial c_{m+1}}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial c_{m+k}}{\partial x_1} & \cdots & \frac{\partial c_{m+k}}{\partial x_n} \end{bmatrix}. \tag{4.11}$$

Using dual feasibility (4.7b)–(4.7e), the dual variables λ, ξ corresponding to the non-negativity constraints (4.4g) can be eliminated. This leads to the restated versions of (4.8) and (4.9);

$$S_l U_l e = S_u U_u e = T_l V_l e = T_u V_u e = \mu e \tag{4.12}$$

and

$$u_l, u_u, v_l, v_u \geq 0, \tag{4.13}$$

respectively. Finally, the μ-perturbed KKT conditions are

$$(4.7a), (4.4b) - (4.4g), (4.12), (4.13). \tag{4.14}$$

With these preliminaries, one can describe an interior-point method as follows. Given an iterate $y \in \mathbb{R}^N$ and a barrier parameter μ, a search direction $\Delta y \in \mathbb{R}^N$ is computed with one step of Newton's method applied to the nonlinear system of equations (4.7a), (4.4b)–(4.4f), (4.12). With this search direction at hand, maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}$, $\bar{\alpha}_{\text{dual}}$ are determined

that maintain strict positivity of the primal and dual slacks s, t and u, v, respectively. This is done by the so-called *fraction-to-the-boundary rule*:

$$\bar{\alpha}_{\text{pri}} := \min(\alpha_s, \alpha_t), \tag{4.15a}$$

$$\alpha_s := \max(\alpha \in (0,1] : s + \alpha \Delta s \geq (1-\tau)\, s), \tag{4.15b}$$

$$\alpha_t := \max(\alpha \in (0,1] : t + \alpha \Delta t \geq (1-\tau)\, t), \tag{4.15c}$$

$$\bar{\alpha}_{\text{dual}} := \min(\alpha_u, \alpha_v), \tag{4.15d}$$

$$\alpha_u := \max(\alpha \in (0,1] : u + \alpha \Delta u \geq (1-\tau)\, u), \tag{4.15e}$$

$$\alpha_v := \max(\alpha \in (0,1] : v + \alpha \Delta v \geq (1-\tau)\, v). \tag{4.15f}$$

Here, $\tau \in (0,1)$ is the so-called *fraction-to-the-boundary parameter* that is computed as in the Ipopt code [134]:[1]

$$\tau := \max\left\{\tau_{\min}, 1 - \kappa_\tau \mu\right\}, \quad \kappa_\tau > 0. \tag{4.16}$$

$\tau_{\min} \in (0,1)$ is a lower bound of τ. In the following, all parameters denoted by κ are constants that can be chosen by the user. To ensure global convergence, the primal step length is possibly shortened again. This is done by a filter or merit function based backtracking line-search method leading to an actual primal step length α_{pri}. The latter is then used to compute the next iterate $y^+ = (y^+_{\text{pri}}, y^+_{\text{dual}})$ via

$$y^+_{\text{pri}} = y_{\text{pri}} + \alpha_{\text{pri}} \Delta y_{\text{pri}}, \tag{4.17a}$$

$$y^+_{\text{dual}} = y_{\text{dual}} + \bar{\alpha}_{\text{dual}} \Delta y_{\text{dual}}. \tag{4.17b}$$

Finally, it has to be determined when and how to update the barrier parameter μ. One mainly distinguishes between *monotone* or *adaptive* methods (see Section 4.1.4 for the details). The standard *monotone Fiacco-McCormick approach* [37] holds the barrier parameter fixed as long as an approximative solution of the current barrier problem is reached. More formally, the *(perturbed) KKT error* is defined as

$$e(\mu) := \max(\theta_{\text{pri}}(y_{\text{pri}}), \theta_{\text{dual}}(y), \theta_{\text{compl}}(y;\mu)). \tag{4.18}$$

Here,

$$\theta_{\text{pri}}(y_{\text{pri}}) := \max(\|\alpha\|, \|\rho_l\|, \|\rho_u\|, \|\beta_l\|, \|\beta_u\|) \tag{4.19}$$

measures primal infeasibility,

$$\theta_{\text{dual}}(y) := \|\nabla_x \mathcal{L}\| \tag{4.20}$$

measures dual infeasibility and

$$\theta_{\text{compl}}(y;\mu) := \max(\|SUe - \mu e\|, \|TVe - \mu e\|) \tag{4.21}$$

[1] To be exact, the constant κ_τ is always chosen to be 1 in the Ipopt code.

measures the (μ-perturbed) complementarity. If not otherwise stated, the infinity norm $\|\cdot\|_\infty$ is used. With this notation, the current barrier problem is approximately solved if

$$e(\mu) < \kappa_\mu \mu \tag{4.22}$$

holds with a constant $\kappa_\mu > 0$. Note that $e(0)$ measures the KKT error of the original NLP. Thus, an iterate with $e(0) = 0$ and $s, t, u, v \geq 0$ is a KKT point if a certain constraint qualification (like the LICQ condition) holds. This motivates the overall termination criterion of the algorithm: Given a user-specified tolerance $\varepsilon_{\text{tol}} > 0$, the algorithm terminates when

$$e(0) < \varepsilon_{\text{tol}} \tag{4.23}$$

is satisfied. (4.22) and (4.23) represent only one possibility of a termination criterion for the barrier problems and the NLP. In practice, especially for badly scaled problems, it is often reasonable to replace the KKT error e by a scaled variant:

$$e_s(\mu) := \max\left(\frac{\theta_{\text{pri}}(y_{\text{pri}})}{s_p}, \frac{\theta_{\text{dual}}(y)}{s_d}, \frac{\theta_{\text{compl}}(y;\mu)}{s_c}\right). \tag{4.24}$$

Here, s_p, s_d and s_c are positive scaling parameters.

Algorithm 1: Basic Interior-Point Framework

Input : Starting point $y^{(0)}$, initial barrier parameter $\mu^{(0)} > 0$ and a user-specified vector of algorithmic constants κ.

1 Set $k = 1$.
2 **while** NLP termination criterion (4.23) is not satisfied **do**
3 Compute the search direction $\Delta y^{(k)}$ by applying one step of Newton's method to the system (4.7a), (4.4b)–(4.4f), (4.12).
4 Compute maximum step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
5 Apply a globalization strategy to obtain an actual primal step length $\alpha_{\text{pri}}^{(k)}$.
6 Compute the new iterate $y^{(k+1)}$ using (4.17).
7 Compute a new barrier parameter $\mu^{(k+1)}$ or set $\mu^{(k+1)} = \mu^{(k)}$.
8 Set $k \leftarrow k + 1$.

Algorithm 1 states an interior-point framework that forms the basis of all concretizations that are presented in the following. It has a lot of degrees of freedom for algorithmic choices:

1. The choice of the initial point $y^{(0)}$.
2. The concrete formulation of the Newton system in line 3.
3. The globalization strategy delivering the actual primal step length in line 5.
4. The barrier parameter update method in line 7.

These algorithmic choices are the topic of the next sections.

4.1.1 Computation of the Search Direction

The barrier problem (4.6) with fixed μ is solved by applying Newton's method to the system of primal-dual equations (4.7a),(4.4b)–(4.4f), (4.12). Using the notation

$$\Phi_l := S_l^{-1} U_l, \qquad \phi_l := u_l - \mu S_l^{-1} e, \tag{4.25a}$$

$$\Phi_u := S_u^{-1} U_u, \qquad \phi_u := u_u - \mu S_u^{-1} e, \tag{4.25b}$$

$$\Psi_l := T_l^{-1} V_l, \qquad \psi_l := v_l - \mu T_l^{-1} e, \tag{4.25c}$$

$$\Psi_u := T_u^{-1} V_u, \qquad \psi_u := v_u - \mu T_u^{-1} e, \tag{4.25d}$$

and applying Newton's method one gets the large but highly structured symmetric KKT system $\Omega \Delta y = -\omega$ with

$$\Omega = \left[\begin{array}{ccccc|ccccc} H & & & & & \nabla c_{\mathcal{E}}^T & I & -I & \nabla c_{\mathcal{I}}^T & -\nabla c_{\mathcal{I}}^T \\ & \Phi_l & & & & & -I & & & \\ & & \Phi_u & & & & & -I & & \\ & & & \Psi_l & & & & & -I & \\ & & & & \Psi_u & & & & & -I \\ \hline \nabla c_{\mathcal{E}} & & & & & & & & & \\ I & -I & & & & & & & & \\ -I & & -I & & & & & & & \\ \nabla c_{\mathcal{I}} & & & -I & & & & & & \\ -\nabla c_{\mathcal{I}} & & & & -I & & & & & \end{array}\right] \in \mathbb{R}^{N \times N}, \tag{4.26a}$$

$$\Delta y = (\Delta x, \Delta s_l, \Delta s_u, \Delta t_l, \Delta t_u, -\Delta z, -\Delta u_l, -\Delta u_u, -\Delta v_l, -\Delta v_u)^T, \tag{4.26b}$$

$$\omega = (\nabla_x \mathcal{L}, \phi_l, \phi_u, \psi_l, \psi_u, \alpha, \beta_l, \beta_u, \rho_l, \rho_u)^T. \tag{4.26c}$$

The left upper block H in Ω stands for the Hessian of the Lagrangian function, i.e.

$$H = \nabla_{xx}^2 \mathcal{L}(x, z, v_l, v_u) = \nabla_{xx}^2 f(x) - \sum_{i \in \mathcal{E}} z_i \nabla_{xx}^2 c_i(x) - \sum_{i \in \mathcal{I}} (v_{l_i} - v_{u_i}) \nabla_{xx}^2 c_i(x), \tag{4.27}$$

where $\nabla_{xx}^2 \varphi$ denotes the Hessian of a function $\varphi : \mathbb{R}^n \to \mathbb{R}$, i.e.

$$\nabla_{xx}^2 \varphi(x) = \left[\begin{array}{ccc} \frac{\partial^2 \varphi}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 \varphi}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \varphi}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 \varphi}{\partial x_n \partial x_n} \end{array}\right] \mathbb{R}^{n \times n}. \tag{4.28}$$

Since $\mathcal{L}$ is twice continuously differentiable, its Hessian H is symmetric (Schwarz's theorem).

The matrix Ω in (4.26a) is called the *primal-dual* matrix, whereas one gets the so-called *primal* matrix with $\Phi_l, \Phi_u, \Psi_l, \Psi_u$ replaced by

$$\Phi_l = \mu S_l^{-2}, \quad \Phi_u = \mu S_u^{-2}, \quad \Psi_l = \mu T_l^{-2}, \quad \Psi_u = \mu T_u^{-2}. \tag{4.29}$$

One obtains the primal system by writing down the KKT conditions of the barrier reformulation (4.6), which contain the equations

$$-\mu S_l^{-1} e + u_l = 0, \qquad -\mu S_u^{-1} e + u_u = 0, \tag{4.30a}$$

$$-\mu T_l^{-1} e + v_l = 0, \qquad -\mu T_u^{-1} e + v_u = 0, \tag{4.30b}$$

and directly applying Newton's method. In contrast to that, the primal-dual system is obtained by multiplying (4.30) with S_l, S_u, T_l, T_u and afterwards applying Newton's method. In what follows, only the primal-dual system is considered.

From here on, the arguments of the problem data $c_\mathcal{E}$ and $c_\mathcal{I}$ and the Lagrangian $\mathcal{L}$ are omitted. Instead of solving the KKT system $\Omega \Delta y = -\omega$, first the size of the system is reduced to the original problem size, i.e. to a linear system in $\mathbb{R}^{n+m+k}$. For this, only simple operations with diagonal matrices are used (cf. [117]).

Elimination of bound slacks

First, the slack variables $s = (s_l, s_u)$ of the variable bounds are eliminated. Rows 7 and 8 of (4.26) yield

$$\Delta x - \Delta s_l = -\beta_l, \qquad \Delta s_l = \Delta x + \beta_l, \tag{4.31}$$

$$-\Delta x - \Delta s_u = -\beta_u, \qquad \Delta s_u = -\Delta x + \beta_u. \tag{4.32}$$

Substitution of Δs_l and Δs_u into rows 2 and 3 gives

$$\Phi_l \Delta s_l + \Delta u_l = -\phi_l, \qquad \Delta u_l = -\Phi_l \Delta x - \bar{\phi}_l, \qquad \bar{\phi}_l := \Phi_l \beta_l + \phi_l, \tag{4.33}$$

$$\Phi_u \Delta s_u + \Delta u_u = -\phi_u, \qquad \Delta u_u = \Phi_u \Delta x - \bar{\phi}_u, \qquad \bar{\phi}_u := \Phi_u \beta_u + \phi_u. \tag{4.34}$$

Finally, by substituting Δu_l and Δu_u into the first row of (4.26) one obtains

$$(H + \Phi)\, \Delta x - \nabla c_\mathcal{E}^T \Delta z - \nabla c_\mathcal{I}^T \Delta v_l + \nabla c_\mathcal{I}^T \Delta v_u = -\gamma \tag{4.35}$$

with

$$\Phi := \Phi_l + \Phi_u, \quad \gamma := \nabla_x \mathcal{L} + \bar{\phi}_l - \bar{\phi}_u. \tag{4.36}$$

Thus, one obtains the *partially reduced KKT system*

$$\left[\begin{array}{ccc|ccc} H+\Phi & & & \nabla c_\mathcal{E}^T & \nabla c_\mathcal{I}^T & -\nabla c_\mathcal{I}^T \\ & \Psi_l & & & -I & \\ & & \Psi_u & & & -I \\ \hline \nabla c_\mathcal{E} & & & & & \\ \nabla c_\mathcal{I} & -I & & & & \\ -\nabla c_\mathcal{I} & & -I & & & \end{array}\right] \left(\begin{array}{c} \Delta x \\ \Delta t_l \\ \Delta t_u \\ \hline -\Delta z \\ -\Delta v_l \\ -\Delta v_u \end{array}\right) = - \left(\begin{array}{c} \gamma \\ \psi_l \\ \psi_u \\ \hline \alpha \\ \rho_l \\ \rho_u \end{array}\right). \tag{4.37}$$

Elimination of range slacks

Next, the slack variables $t = (t_l, t_u)$ of the range constraints are eliminated. Rows 5 and 6 of (4.37) yield

$$\nabla c_{\mathcal{I}} \Delta x - \Delta t_l = -\rho_l, \qquad \Delta t_l = \nabla c_{\mathcal{I}} \Delta x + \rho_l, \qquad \nabla c_{\mathcal{I}} \Delta x = \Delta t_l - \rho_l, \tag{4.38}$$

$$-\nabla c_{\mathcal{I}} \Delta x - \Delta t_u = -\rho_u, \qquad \Delta t_u = -\Delta t_l + \rho, \qquad \rho := \rho_l + \rho_u. \tag{4.39}$$

By substituting Δt_u into row 3 of (4.37) one gets

$$\Delta v_u = \Psi_u \Delta t_l - \hat{\psi}_u, \quad \hat{\psi}_u := \Psi_u \rho + \psi_u \tag{4.40}$$

and substituting Δt_u into the difference of row 2 and 3 yields

$$\Psi \Delta t_l + \Delta \hat{v} = -\hat{\psi}, \quad \Psi := \Psi_l + \Psi_u, \quad \hat{\psi} = \psi_l - \hat{\psi}_u, \quad \Delta \hat{v} := \Delta v_l - \Delta v_u. \tag{4.41}$$

Solving the latter for Δt_l gives

$$\Delta t_l = \Psi^{-1} \left(-\Delta \hat{v} - \hat{\psi} \right), \tag{4.42}$$

which leads to

$$\nabla c_{\mathcal{I}} \Delta x + \Psi^{-1} \Delta \hat{v} = -\hat{\rho}, \quad \hat{\rho} := \rho_l + \Psi^{-1} \hat{\psi} \tag{4.43}$$

by substituting Δt_l into row 5 of (4.37). Finally, using $\Delta \hat{v}$ in the first row leads to the *reduced KKT system* $\Omega_{\mathrm{r}} \Delta y_{\mathrm{r}} = -\omega_{\mathrm{r}}$:

$$\begin{bmatrix} H + \Phi & \nabla c_{\mathcal{E}}^T & \nabla c_{\mathcal{I}}^T \\ \nabla c_{\mathcal{E}} & & \\ \nabla c_{\mathcal{I}} & & -\Psi^{-1} \end{bmatrix} \begin{pmatrix} \Delta x \\ -\Delta z \\ -\Delta \hat{v} \end{pmatrix} = - \begin{pmatrix} \gamma \\ \alpha \\ \hat{\rho} \end{pmatrix}. \tag{4.44}$$

In summary, the system $\Omega \Delta y = -\omega$ is first reduced to the system $\Omega_{\mathrm{r}} \Delta y_{\mathrm{r}} = -\omega_{\mathrm{r}}$ with $\Omega_{\mathrm{r}} \in \mathbb{R}^{N_{\mathrm{r}} \times N_{\mathrm{r}}}$ and $N_{\mathrm{r}} = n + m + k$. The smaller system is solved and its solution Δy_{r} is then expanded to a solution Δy of the full KKT system. When solving the reduced system one encounters two main difficulties in solving general nonconvex nonlinear optimization problems. Since the overall interior-point method is globalized by a filter line-search algorithm (cf. Section 4.1.2), one has to ensure that the search direction Δy is a descent direction with respect to the globalization strategy. In addition, a solution of (4.44) does not have to exist if the second block row is not regular. To guarantee that these two assumptions hold in every iteration one can incorporate a *convexification* and *regularization strategy* by modifying system (4.44) to

$$\begin{bmatrix} H + \Phi + \nu_c I & \nabla c_{\mathcal{E}}^T & \nabla c_{\mathcal{I}}^T \\ \nabla c_{\mathcal{E}} & -\nu_r I & \\ \nabla c_{\mathcal{I}} & & -\Psi^{-1} \end{bmatrix} \begin{pmatrix} \Delta x \\ -\Delta z \\ -\Delta \hat{v} \end{pmatrix} = - \begin{pmatrix} \gamma \\ \alpha \\ \hat{\rho} \end{pmatrix}, \tag{4.45}$$

where $\nu_c, \nu_r \geq 0$ are the *convexification* and *regularization parameters*. If ν_c is large enough, it can be shown that the Hessian block $H + \Phi + \nu_c I$ projected onto the null-space of the Jacobians $\nabla c = \left[\nabla c_{\mathcal{E}}^T, \nabla c_{\mathcal{I}}^T\right]^T$ of the constraints is positive definite. Additionally, a value $\nu_r \neq 0$ regularizes the second block row and thus guarantees the existence of a unique search direction. The solution algorithm for system (4.45) including adaptive choices of ν_c and ν_r is presented in the next sections.

Solution of the Reduced KKT System: The Dense Case

The solution of the step equation (4.44) may encounter the two problems of nonconvexity and singularity. The KKT solution algorithm presented in this section exploits the special optimization superstructure of the linear system (4.44) and modifies the system matrix in a way such that both positive definiteness of the projected Hessian and regularity of the Jacobians of the constraints are addressed by modifying the system matrix Ω_r. The techniques presented in this section, namely the null-space and the Schur complement method, are tailored for dense blocks of the reduced KKT matrix Ω_r.

Both techniques first compute the LQ factorization

$$\nabla c_{\mathcal{E}} = LQ = \begin{bmatrix} L_1 & 0 \end{bmatrix} \begin{bmatrix} Y^T \\ Z^T \end{bmatrix} = L_1 Y^T, \tag{4.46}$$

of the Jacobian of the equality constraints. Here, $L \in \mathbb{R}^{m \times n}, Q \in \mathbb{R}^{n \times n}, L_1 \in \mathbb{R}^{m \times m}, Y \in \mathbb{R}^{n \times m}, Z \in \mathbb{R}^{n \times (n-m)}$ holds and the zero block in L is of size $m \times (n-m)$. With this, one obtains an orthonormal basis Z of the null-space $\ker(\nabla c_{\mathcal{E}})$ of $\nabla c_{\mathcal{E}}$ and a matrix Y such that $[Y\,Z]$ is an orthonormal basis of $\mathbb{R}^n$.

If L_1 has zero columns at its right or, equivalently, if Z has dimension $n \times d$ with $d > n - m$, there is a rank deficiency in $\nabla c_{\mathcal{E}}$. In this case, Ω_r can be regularized with a negative multiple of the identity matrix $-\nu_r I, \nu_r > 0$, in the $(2,2)$-block. This yields the *regularized reduced KKT system* matrix

$$\tilde{\Omega} := \begin{bmatrix} \bar{H} & \nabla c_{\mathcal{E}}^T & \nabla c_{\mathcal{I}}^T \\ \nabla c_{\mathcal{E}} & -\nu_r I & \\ \nabla c_{\mathcal{I}} & & -\Psi^{-1} \end{bmatrix} =: \begin{bmatrix} \bar{H} & \nabla c^T \\ \nabla c & -M \end{bmatrix}, \tag{4.47a}$$

$$M := \begin{bmatrix} \nu_r I & \\ & \Psi^{-1} \end{bmatrix} \in \mathbb{R}^{(m+k)\times(m+k)}, \quad \bar{H} := H + \Phi. \tag{4.47b}$$

If the KKT matrix is regularized as in (4.47), the regularized and reduced KKT system is solved with the *Schur complement method*, whereas the original system (4.44) is solved with the *null-space method*. Both methods are described in detail in the following.

Null-Space Method By using (4.46) the primal search direction Δx is decomposed into

$$\Delta x = Y\Delta x_Y + Z\Delta x_Z, \quad \Delta x_Y \in \mathbb{R}^m, \quad \Delta x_Z \in \mathbb{R}^{n-m}. \tag{4.48}$$

Substitution of this decomposition into the second row of (4.44) yields

$$\nabla c_{\mathcal{E}}\Delta x = \nabla c_{\mathcal{E}}\left(Y\Delta x_Y + Z\Delta x_Z\right) = \left(\nabla c_{\mathcal{E}}Y\right)\Delta x_Y = -\alpha, \tag{4.49}$$

with $\nabla c_{\mathcal{E}}Y \in \mathbb{R}^{m\times m}$. System (4.49) can be efficiently solved using the already computed LQ factorization (4.46);

$$\left(\nabla c_{\mathcal{E}}Y\right)\Delta x_Y = L_1\Delta x_Y = -\alpha. \tag{4.50}$$

Next, $-\Delta\hat{v}$ is eliminated by using the third equation of (4.44);

$$\begin{aligned} -\Delta\hat{v} &= \Psi\left(\hat{\rho} + \nabla c_{\mathcal{I}}\Delta x\right) \\ &= \Psi\left(\hat{\rho} + \nabla c_{\mathcal{I}}\left(Y\Delta x_Y + Z\Delta x_Z\right)\right) \\ &= \bar{\rho} + \Psi\nabla c_{\mathcal{I}}Z\Delta x_Z. \end{aligned} \tag{4.51}$$

Here, $\bar{\rho} := \Psi\left(\hat{\rho} + \nabla c_{\mathcal{I}}Y\Delta x_Y\right)$. Substitution of $-\Delta\hat{v}$ into the first row of (4.44) and multiplication with Z^T yields

$$\hat{H}\Delta x_Z = -\bar{\gamma}, \tag{4.52}$$

where

$$\hat{H} := Z^T\left(\bar{H} + \nabla c_{\mathcal{I}}^T\Psi\nabla c_{\mathcal{I}}\right)Z \in \mathbb{R}^{(n-m)\times(n-m)} \tag{4.53}$$

is the *reduced Hessian* and the modified right-hand side is given by

$$\bar{\gamma} := Z^T\left(\gamma + \bar{H}Y\Delta x_Y + \nabla c_{\mathcal{I}}^T\bar{\rho}\right). \tag{4.54}$$

(4.52) is solved by a Cholesky factorization of the reduced Hessian $\hat{H}$. If $\hat{H}$ is not positive definite, $\bar{H}$ is convexified by successively adding an increasing multiple of the identity $\nu_c I, \nu_c > 0$, re-computing

$$\hat{H} = Z^T\left(\bar{H} + \nu_c I + \nabla c_{\mathcal{I}}^T\Psi\nabla c_{\mathcal{I}}\right)Z \tag{4.55}$$

and

$$\bar{\gamma} = Z^T(\gamma + [\bar{H} + \nu_c I]Y\Delta x_Y + \nabla c_{\mathcal{I}}^T\bar{\rho}), \tag{4.56}$$

and re-computing the Cholesky factorization until it succeeds. After solving (4.52), Δx is computed by using (4.48) and $-\Delta\hat{v}$ is computed by (4.51). Finally, the first equation of (4.44) multiplied with Y^T reads

$$Y^T\nabla c_{\mathcal{E}}^T\left(-\Delta z\right) = -Y^T\left(\gamma + \hat{H}\Delta x + \nabla c_{\mathcal{I}}^T\left(-\Delta\hat{v}\right)\right). \tag{4.57}$$

The latter can be solved using the LQ factorization (4.46) again;

$$L_1^T\left(-\Delta z\right) = -\hat{\gamma}, \quad \hat{\gamma} := Y^T\left(\gamma + \hat{H}\Delta x + \nabla c_{\mathcal{I}}^T\left(-\Delta\hat{v}\right)\right). \tag{4.58}$$

Schur Complement Method The Schur complement method is used to solve the KKT system if Ω_r has to be regularized due to a rank deficiency in its second block row. This leads to the regularized matrix as in (4.47). For this case, the right-hand side and the solution vector can be rewritten as follows:

$$\begin{pmatrix} \Delta x \\ -\Delta\eta \end{pmatrix} := \begin{pmatrix} \Delta x \\ -\Delta z \\ -\Delta\hat{v} \end{pmatrix}, \quad \begin{pmatrix} \gamma \\ \eta \end{pmatrix} := \begin{pmatrix} \gamma \\ \alpha \\ \hat{\rho} \end{pmatrix}, \quad \Delta\eta, \eta \in \mathbb{R}^{m+k}. \tag{4.59}$$

First, $-\Delta\eta$ is eliminated using the second row of (4.47);

$$-\Delta\eta = M^{-1}\left(\eta + \nabla c \Delta x\right). \tag{4.60}$$

Substitution of $-\Delta\eta$ into the first row of (4.47) yields

$$W\Delta x = -\bar{\eta}, \tag{4.61}$$

with

$$W := \bar{H} + \nabla c^T M^{-1} \nabla c, \quad \bar{\eta} := \gamma + \nabla c^T M^{-1}\eta. \tag{4.62}$$

W is the so-called *Schur complement matrix*. (4.61) is solved by a Cholesky factorization of W. If W is not positive definite, the same convexification strategy as in the null-space method is applied.

Algorithm 2 combines the presented strategies for solving (4.44) for dense matrix blocks. The last degree of freedom in Algorithm 2 is the update strategy for the convexification parameter. In the concrete implementation developed for this thesis, the inertia correction subprocedure proposed in [134] is used.

Solution of the Reduced KKT System: The Sparse Case

Algorithm 2 is not preferable if one wants to exploit the sparsity that often appears in the problem data matrices $H, \nabla c_\mathcal{E}, \nabla c_\mathcal{I}$ of NLP-type real-world problems. In this case, the implemented method uses external software packages for the factorization of symmetric indefinite linear systems. Most of these solvers provide the user with information about rank deficiencies and the inertia (see [91]) of the matrix to factorize. With these information, one can design algorithms for factorizing the reduced KKT matrix that address the problems of missing positive definiteness of the projected Hessian as well as rank deficiencies. See [134] for a detailed description of the used inertia correction subprocedure.

Iterative Refinement

In practice, the reduced KKT matrix Ω_r is often ill-conditioned. This may give rise to inaccurate solutions Δy_r. Since the quality of the search direction is a crucial aspect of interior-point methods,

Algorithm 2: Solution Algorithm for the Reduced KKT System with Dense Matrix Blocks

Input : Reduced KKT matrix Ω_r and right-hand side ω_r.

Compute LQ factorization $\nabla c_\mathcal{E} = LQ$ (see (4.46)).
if L_1 has no zero columns **then**
  Continue with line 6.
else
  Choose regularization parameter $\nu_r > 0$ and modify the KKT matrix as described in (4.47). Continue with line 18.
`// Null-Space Method`
Solve $L_1 \Delta x_Y = -\alpha$ to obtain Δx_Y (see (4.49) and (4.50)).
Compute $\Delta \tilde{x}_Y = Y \Delta x_Y$ and $\bar{\rho} = \Psi \left(\hat{\rho} + \nabla c_\mathcal{I} \Delta \tilde{x}_Y \right)$.
Initialize convexification counter $k_c = 0$.
repeat `/* convexification strategy */`
  Update ν_c, set $\tilde{H} = \bar{H} + \nu_c I$ and compute $\hat{H} = Z^T \left(\tilde{H} + \nabla c_\mathcal{I}^T \Psi \nabla c_\mathcal{I} \right) Z$.
  Try to compute the Cholesky factorization $L_{\hat{H}} L_{\hat{H}}^T = \hat{H}$ and set $k_c \leftarrow k_c + 1$.
until Cholesky factorization was successful
Compute $\bar{\gamma} = Z^T \left(\gamma + \hat{H} \Delta \tilde{x}_Y + \nabla c_\mathcal{I}^T \bar{\rho} \right)$ and solve $\hat{H} \Delta x_Z = -\bar{\gamma}$.
Compute $-\Delta \hat{v}$ using $\Delta \tilde{x}_Z = Z \Delta x_Z$ and $-\Delta \hat{v} = \bar{\rho} + \Psi \nabla c_\mathcal{I} \Delta \tilde{x}_Z$.
Compute $\Delta x = \Delta \tilde{x}_Y + \Delta \tilde{x}_Z$.
Compute $\hat{\gamma} = Y^T \left(\gamma + \hat{H} \Delta x + \nabla c_\mathcal{I}^T \left(-\Delta \hat{v} \right) \right)$ and solve $L_1^T \left(-\Delta z \right) = -\hat{\gamma}$.
return search direction $\Delta y_\mathrm{r} = (\Delta x, -\Delta z, -\Delta \hat{v})$.
`// Schur Complement Method`
Compute system matrix and right-hand side of (4.61) via $C = \nabla c^T M^{-1} \in \mathbb{R}^{n \times (m+k)}, \bar{\eta} = \gamma + C\eta, W = \bar{H} + C \nabla c$.
Initialize convexification counter $k_c = 0$.
repeat `/* convexification strategy */`
  Update ν_c, set $\hat{W} = W + \nu_c I$.
  Try to compute the Cholesky factorization $L_{\hat{W}} L_{\hat{W}}^T = \hat{W}$ and set $k_c \leftarrow k_c + 1$.
until Cholesky factorization was successful
Solve $\hat{W} \Delta x = -\bar{\eta}$ and compute $-\Delta \eta = M^{-1} \left(\eta + \nabla c \Delta x \right)$.
return search direction $\Delta y_\mathrm{r} = (\Delta x, -\Delta \eta)$.

the proposed algorithm applies an *iterative refinement* strategy in order to improve the accuracy of the solution of the KKT system.

Let k be the iteration counter of the iterative refinement algorithm and let $\Delta y_\mathrm{r}^{(0)}$ be the original solution of the reduced KKT system $\Omega_\mathrm{r} \Delta y_\mathrm{r} = -\omega_\mathrm{r}$. Then the iterative procedure

$$r^{(k)} = -\omega_\mathrm{r} - \Omega_\mathrm{r} \Delta y_\mathrm{r}^{(k)}, \quad \Omega_\mathrm{r} c^{(k)} = r^{(k)}, \quad \Delta y_\mathrm{r}^{(k+1)} = \Delta y_\mathrm{r}^{(k)} + c^{(k)}, \qquad k = 0, 1, \ldots, \tag{4.63}$$

is applied until the residual $r^{(k)}$ satisfies the stopping criterion

$$\| r^{(k)} \| < \varepsilon_r \tag{4.64}$$

for a user-specified tolerance $\varepsilon_r > 0$. For more details about iterative refinement see [48].

4.1.2 Globalization with a Filter Line-Search Algorithm

After computing the search direction Δy, primal and dual step lengths $\alpha_{\text{pri}}, \alpha_{\text{dual}}$ have to be chosen to determine the next iterate. The dual step length is chosen to be $\bar{\alpha}_{\text{dual}}$ as defined in (4.15). In contrast to that, it is required that the new primal iterate $y_{\text{pri}}^{(k+1)}$ satisfies additional conditions in order to ensure global convergence. There are two standard techniques of nonlinear programming algorithms to force global convergence: merit functions and filters. The first combines the aim of minimizing the objective function and reaching feasibility in a *merit function*. A standard merit function for problem (4.4) is the ℓ_1 penalty function

$$m\left(y_{\text{pri}}\right) = f\left(x\right) + \pi\left(\theta_{\text{pri}}\left(x\right) + \sum_{i=1}^{n}\left(\left[s_{l_i}\right]^- + \left[s_{u_i}\right]^-\right) + \sum_{i\in\mathcal{I}}\left(\left[t_{l_i}\right]^- + \left[t_{u_i}\right]^-\right)\right). \tag{4.65}$$

Here, for $x \in \mathbb{R}$ the function $[x]^-$ is defined as

$$[x]^- := \begin{cases} 0, & x \geq 0, \\ -x, & x < 0. \end{cases} \tag{4.66}$$

$\pi > 0$ is the so-called *penalty parameter*. For interior-point methods, the slack variable terms in (4.65) always vanish leading to the simplified merit function

$$m\left(y_{\text{pri}}\right) = f\left(x\right) + \pi\theta_{\text{pri}}\left(x\right). \tag{4.67}$$

Filter methods are first proposed by Fletcher and Leyffer in [41]. The idea of filter methods is to treat the goals of minimizing the objective function and reaching feasibility separately. This allows filter methods to accept step lengths that make progress only in the objective function *or* feasibility instead of requiring progress in a combination of both like in (4.65). As a consequence, filter methods can often take larger steps and tend to be more robust.

The following paragraphs review the filter line-search method for the interior-point method proposed by Wächter and Biegler in [134]. The general definitions and concepts can also be found in the book of Nocedal and Wright [91]. Some other ideas concerning filter methods are given by Benson et al. in [10].

In this section the filter method is described for the original objective function f and the measure of primal infeasibility θ_{pri}. Later, it is applied to the barrier problems with fixed barrier parameter μ. This is simply done by replacing all occurrences of f by φ_μ. The filter is then denoted by $\mathcal{F}_\mu$ for the sake of clarity.

The following definition is taken from [91].

Definition 14 (Filter)**.** *1. One says that a pair* $(f_i, \theta_{\text{pri},i})$ *is* dominated *by another pair* $(f_j, \theta_{\text{pri},j})$ *if* $f_j \leq f_i$ *and* $\theta_{\text{pri},j} \leq \theta_{\text{pri},i}$ *holds.*

2. A filter $\mathcal{F}$ *is a list of pairs* $(f_i, \theta_{\text{pri},i})$ *in which no pair dominates any other pair.*

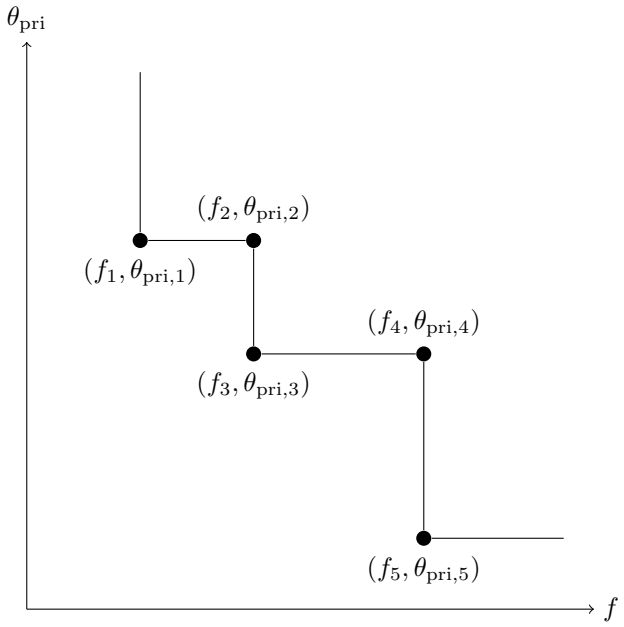

Figure 4.1: A filter with five entries.

Figure 4.1 gives a graphical illustration of a filter. The basic idea of a backtracking filter line-search is to generate a decreasing sequence of primal step lengths

$$\alpha_{\text{pri},l+1}^{(k)} = \kappa_b \alpha_{\text{pri},l}^{(k)}, \quad \kappa_b \in (0,1), \quad l = 0, 1, \ldots, \tag{4.68}$$

and with this, a sequence of primal *trial points*

$$y_{\text{pri}}^{+} = y_{\text{pri}}^{(k)} + \alpha_{\text{pri},l}^{(k)} \Delta y_{\text{pri}}^{(k)}, \quad l = 0, 1, \ldots, \tag{4.69}$$

until a trial point is found such that

$$\left(f^{+}, \theta_{\text{pri}}^{+}\right) := \left(f(y_{\text{pri}}^{+}), \theta_{\text{pri}}(y_{\text{pri}}^{+})\right) \tag{4.70}$$

is *acceptable* to the filter. The concept of *acceptability* is concretized in the following definition.

Definition 15 (Acceptability)**.** *A trial point y^{+} is* acceptable *to the filter if the corresponding pair $(f^{+}, \theta_{\text{pri}}^{+})$ is not dominated by any pair in the filter, i.e. if*

$$f^{+} < f_j \quad \textit{or} \quad \theta_{\text{pri}}^{+} < \theta_{\text{pri},j} \tag{4.71}$$

holds for all filter entries $(f_j, \theta_{\text{pri},j})$.

To improve the practical performance of the algorithm, this acceptance criterion is strengthened to

$$f^{+} \leq f_j - \kappa_m \theta_{\text{pri},j} \quad \text{or} \quad \theta_{\text{pri}}^{+} \leq (1 - \kappa_m)\, \theta_{\text{pri},j} \tag{4.72}$$

for a given *filter margin* $\kappa_m > 0$. The idea of a strengthened filter acceptance criterion is already mentioned in the original proposal of filter methods [41].

Sufficient Decrease Conditions

In addition to the filter acceptance criterion a trial point has to satisfy certain sufficient decrease conditions. According to [134], two cases are distinguished. If the constraint violation is small, i.e. $\theta_{\text{pri}}(y_{\text{pri}}^{(k)}) =: \theta_{\text{pri}}^{(k)} \leq \theta_{\text{pri}}^{\min}$, and the so-called *f-type switching condition*

$$d^{(k)} < 0 \quad \text{and} \quad \alpha_{\text{pri},l}^{(k)} \left(-d^{(k)}\right)^{\kappa_{s_1}} > \kappa_{s_2} \left(\theta_{\text{pri}}^{(k)}\right)^{\kappa_{s_3}} \tag{4.73}$$

holds with $d^{(k)} := \left(g^{(k)}\right)^T \Delta y_{\text{pri}}^{(k)}$ and constants $\kappa_{s_1} \geq 1, \kappa_{s_2} > 0, \kappa_{s_3} > 1$, a trial point has to fulfill the *Armijo condition*:

$$f^+ \leq f^{(k)} + \kappa_a \alpha_{\text{pri},l}^{(k)} d^{(k)}, \quad \kappa_a \in (0, 1/2). \tag{4.74}$$

Note that the first condition in (4.73) only depends on the current iterate and not on the current trial point. Hence, this condition has to be evaluated only once in every iteration k of the main interior-point method. If $\theta_{\text{pri}}^{(k)} > \theta_{\text{pri}}^{\min}$ or (4.73) does not hold, the trial point has to satisfy one of the following sufficient decrease conditions with respect the current iterate:

$$f^+ \leq f^{(k)} - \kappa_m \theta_{\text{pri}}^{(k)} \quad \text{or} \quad \theta_{\text{pri}}^+ \leq (1 - \kappa_m)\, \theta_{\text{pri}}^{(k)}. \tag{4.75}$$

Maximum Constraint Violation

Additionally, one wants to ensure that a trial point with a constraint violation larger than a given threshold $\theta_{\text{pri}}^{\max}$ is never accepted by the filter (cf. [41]). This is easily done by initializing the filter as follows:

$$\mathcal{F} = \left\{(-\infty, \theta_{\text{pri}}^{\max})\right\}. \tag{4.76}$$

Mostly, $\theta_{\text{pri}}^{\max}$ is computed in dependence of the primal infeasibility at the starting point, e.g.

$$\theta_{\text{pri}}^{\max} = \kappa_{i_1} \max\left(1, \kappa_{i_2} \theta_{\text{pri}}^{(0)}\right), \quad \kappa_{i_1}, \kappa_{i_2} > 0. \tag{4.77}$$

Second-Order Correction

Second-order corrections (SOC) are used to avoid the *Maratos effect* (see [91]) and to improve the overall robustness of the algorithm. In the method developed for this thesis, an SOC step is computed if the first trial point is rejected by the filter line-search procedure.

First, a definition of a second-order correction step is given. Afterwards, it is described how SOC steps can be computed efficiently in an interior-point framework. A more detailed discussion of second-order correction can be found in the books of Conn et al. [28] or of Nocedal and Wright [91].

In the following, the set of equality constraints of (4.4) is abbreviated by

$$c(y_{\text{pri}}) := \begin{pmatrix} \alpha(x) \\ \rho_l(x,t_l) \\ \rho_u(x,t_u) \\ \beta_l(x,s_l) \\ \beta_u(x,s_u) \end{pmatrix} = \begin{pmatrix} c_{\mathcal{E}}(x) \\ c_{\mathcal{I}}(x) - t_l - r_l \\ -c_{\mathcal{I}}(x) - t_u + r_u \\ x - s_l - b_l \\ -x - s_u + b_u \end{pmatrix}. \tag{4.78}$$

With Taylor's approximation one has

$$c(y_{\text{pri}}^+) = c(y_{\text{pri}}) + \nabla c(y_{\text{pri}})\Delta y_{\text{pri}} + O(\|\Delta y_{\text{pri}}\|^2) \tag{4.79}$$

for $y_{\text{pri}}^+ := y_{\text{pri}} + \Delta y_{\text{pri}}$. The KKT system (4.26) implies

$$c(y_{\text{pri}}) + \nabla c(y_{\text{pri}})\Delta y_{\text{pri}} = 0, \tag{4.80}$$

leading to

$$c(y_{\text{pri}}^+) = O(\|\Delta y_{\text{pri}}\|^2). \tag{4.81}$$

As usual (see [28]), an SOC step Δy_{soc} is defined by the conditions

$$c(y_{\text{pri}}^+ + \Delta y_{\text{soc}}) = o(\|\Delta y_{\text{pri}}\|^2) \quad \text{and} \quad \Delta y_{\text{soc}} = o(\|\Delta y_{\text{pri}}\|). \tag{4.82}$$

To compute an SOC step one employs Taylor's formula again;

$$c(y_{\text{pri}}^+ + \Delta y_{\text{soc}}) = c(y_{\text{pri}}^+) + \nabla c(y_{\text{pri}}^+)\Delta y_{\text{soc}} + O(\|\Delta y_{\text{soc}}\|^2). \tag{4.83}$$

If the LICQ condition holds, ∇c has full rank at least in a neighborhood of a solution x^*. A solution Δy_{soc} of the linear system

$$c(y_{\text{pri}}^+) + \nabla c(y_{\text{pri}}^+)\Delta y_{\text{soc}} = 0 \tag{4.84}$$

is then an SOC step (see [28, pages 643 ff.]). Here, $\nabla c \in \mathbb{R}^{n_d \times n_p}$ with $n_d < n_p$ holds and hence, the solution of (4.84) is not unique. As a remedy, one often computes the minimum ℓ_2 norm solution of the system. This solution can either be computed by the Moore-Penrose pseudoinverse ∇c^+ (see [48] for details) via

$$\Delta y_{\text{soc}} = -\nabla c^+ c(y_{\text{pri}}^+), \quad \nabla c^+ = \nabla c^T \left(\nabla c \nabla c^T\right)^{-1}, \tag{4.85}$$

or by solving the convex quadratic problem

$$\min_{p \in \mathbb{R}^{n_p}} \frac{1}{2}\|p\|_2^2 \quad \text{s.t.} \quad c(y_{\text{pri}}^+) + \nabla c(y_{\text{pri}}^+)p = 0. \tag{4.86}$$

The latter is equivalent to the solution of the corresponding KKT system

$$\begin{bmatrix} I & \nabla c^T(y_{\text{pri}}^+) \\ \nabla c(y_{\text{pri}}^+) & 0 \end{bmatrix} \begin{pmatrix} p \\ -\lambda_{\text{soc}} \end{pmatrix} = - \begin{pmatrix} 0 \\ c(y_{\text{pri}}^+) \end{pmatrix} \tag{4.87}$$

with some Lagrange multipliers $\lambda_{\text{soc}} \in \mathbb{R}^{n_d}$. Equations (4.84)–(4.87) have the significant numerical drawback that the Jacobians of the constraints have to be re-computed at y_{pri}^+. Luckily, it is possible to prove that a minimum ℓ_2 norm solution of

$$c(y_{\text{pri}}^+) + \nabla c(y_{\text{pri}})\Delta y_{\text{soc}} = 0 \tag{4.88}$$

is an SOC step, too. Here, (4.88) reads

$$\begin{bmatrix} \nabla c_{\mathcal{E}}(x) & & & & \\ I & -I & & & \\ -I & & -I & & \\ \nabla c_{\mathcal{I}}(x) & & & -I & \\ -\nabla c_{\mathcal{I}}(x) & & & & -I \end{bmatrix} \begin{pmatrix} \Delta x_{\text{soc}} \\ \Delta s_{l,\text{soc}} \\ \Delta s_{u,\text{soc}} \\ \Delta t_{l,\text{soc}} \\ \Delta t_{u,\text{soc}} \end{pmatrix} = - \begin{pmatrix} \alpha^+ \\ \beta_l^+ \\ \beta_u^+ \\ \rho_l^+ \\ \rho_u^+ \end{pmatrix}. \tag{4.89}$$

System (4.89) is underdetermined, too. To avoid additional matrix factorizations, Δy_{soc} can be computed by solving system (4.26) with a different upper right-hand side in which the upper block is set to zero;

$$\nabla_x \mathcal{L} = \phi_l = \phi_u = \psi_l = \psi_u = 0. \tag{4.90}$$

Applying the same reduction and expansion techniques as described in Section 4.1.1 but without executing zero-operations coming from the modified right-hand side gives the following simplified reduction and expansion schemes:

1. Modified reduction steps for solving the SOC KKT system:

$$\gamma = \Phi_l\beta_l - \Phi_u\beta_u, \quad \hat{\rho} = \rho_l + \Psi^{-1}\hat{\psi}, \quad \hat{\psi} = -\hat{\psi}_u, \quad \hat{\psi}_u = \Psi_u\rho, \quad \rho = \rho_l + \rho_u. \tag{4.91}$$

2. Modified expansion steps for solving the SOC KKT system:

$$\Delta t_l = \Psi^{-1}\left(-\Delta\hat{v} - \hat{\psi}\right), \quad \Delta t_u = -\Delta t_l + \rho. \tag{4.92}$$

Algorithm 3 states the complete filter line-search algorithm.

Algorithm 3: Filter Line-Search Algorithm

Input : Primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$, maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$, filter margin $\kappa_m > 0$, minimum primal step length $\alpha_{\text{pri}}^{\min} > 0$.

1 Initialize backtracking line-search counter $l = 0$, set $\alpha_{\text{pri},l}^{(k)} = \bar{\alpha}_{\text{pri}}^{(k)}$.
2 **if** $\alpha_{\text{pri},l}^{(k)} < \alpha_{\text{pri}}^{\min}$ **then**
3 Go to the feasibility restoration phase (see Sect. 4.1.3).
4 Compute trial point $y_{\text{pri}}^{+} = y_{\text{pri}}^{(k)} + \alpha_{\text{pri},l}^{(k)} \Delta y_{\text{pri}}^{(k)}$.
5 **if** y_{pri}^{+} is acceptable to $\mathcal{F}$ (cf. (4.72)) **then**
6 **if** $\theta_{\text{pri}}^{(k)} \leq \theta_{\text{pri}}^{\min}$ **and** (4.73) holds **then**
7 **if** (4.74) holds **then**
8 Accept y_{pri}^{+}, **return** step length $\alpha_{\text{pri},l}^{(k)}$ and search direction $\Delta y_{\text{pri}}^{(k)}$.
9 **else if** $l = 0$ **then**
10 Go to SOC step computation (line 22).
11 **else**
12 **if** (4.75) holds **then**
13 Accept y_{pri}^{+}.
14 **if** (4.73) or (4.74) does not hold **then**
15 Add pair $(f^{+}, \theta_{\text{pri}}^{+})$ to $\mathcal{F}$ and remove all entries from the filter that are dominated by $(f^{+}, \theta_{\text{pri}}^{+})$.
16 **return** step length $\alpha_{\text{pri},l}^{(k)}$ and search direction $\Delta y_{\text{pri}}^{(k)}$.
17 **else if** $l = 0$ **then**
18 Go to SOC step computation (line 22).
19 **else if** $l = 0$ **then**
20 Go to SOC step computation (line 22).
21 Set $\alpha_{\text{pri},l+1}^{(k)} = \kappa_b \alpha_{\text{pri},l}^{(k)}$, $l \leftarrow l + 1$ and go to line 2.
`// Second-Order Correction`
22 Compute SOC step Δy_{soc} as described in Section 4.1.2, the corrected search direction $\Delta y_{\text{cor}}^{+} = \alpha_{\text{pri},l}^{(k)} \Delta y_{\text{pri}}^{(k)} + \Delta y_{\text{soc}}$ and the new step length α_{soc}^{+} using (4.15) with $\Delta y_{\text{cor}}^{+}$.
23 Compute new SOC trial point $y_{\text{soc}}^{+} = y_{\text{pri}}^{(k)} + \alpha_{\text{soc}}^{+} \Delta y_{\text{cor}}^{+}$.
24 **if** y_{soc}^{+} is acceptable to $\mathcal{F}$ (cf. (4.72)) **then**
25 **if** $\theta_{\text{pri}}^{(k)} \leq \theta_{\text{pri}}^{\min}$ **and** (4.73) holds for $\theta_{\text{pri}}^{+} = \theta_{\text{pri}}(y_{\text{soc}}^{+})$ and $\Delta y_{\text{pri}}^{(k)}$ replaced by $\Delta y_{\text{cor}}^{+}$ **then**
26 **if** (4.74) holds for $f^{+} = f(y_{\text{soc}}^{+})$ and $\Delta y_{\text{pri}}^{(k)}$ replaced by $\Delta y_{\text{cor}}^{+}$ **then**
27 Accept y_{soc}^{+}, **return** step length α_{soc}^{+} and corrected search direction $\Delta y_{\text{cor}}^{+}$.
28 **else if** (4.75) holds for $f^{+} = f(y_{\text{soc}}^{+})$ and $\theta_{\text{pri}}^{+} = \theta_{\text{pri}}(y_{\text{soc}}^{+})$ **then**
29 Accept y_{soc}^{+}.
30 **if** (4.73) or (4.74) (modified as above) does not hold **then**
31 Add pair $(f^{+}, \theta_{\text{pri}}^{+})$ to $\mathcal{F}$ and remove all entries from the filter that are dominated by $(f^{+}, \theta_{\text{pri}}^{+})$.
32 **return** step length α_{soc}^{+} and corrected search direction $\Delta y_{\text{cor}}^{+}$.
33 Discard α_{soc}^{+} and $\Delta y_{\text{cor}}^{+}$, set $\alpha_{\text{pri},l+1}^{(k)} = \kappa_b \alpha_{\text{pri},l}^{(k)}$, $l \leftarrow l + 1$ and go to line 2.

4.1.3 Feasibility Restoration Phase

If the backtracking filter line-search procedure (Algorithm 3) succeeds, it returns a primal step length and, possibly, a corrected search direction. Both are used to compute the new iterate. Nevertheless, it is possible that Algorithm 3 does not succeed. In this case, it is not possible to compute a sufficiently large primal step length $\alpha_{\text{pri}}^{(k)} > \alpha_{\text{pri}}^{\min}$ that is acceptable and satisfies all other required conditions. Before terminating the overall interior-point algorithm, it is possible to revert to a so-called *feasibility restoration phase*. Here, the goal is to compute a new iterate that is acceptable to the filter due to a decreased primal infeasibility. To achieve this goal, the same interior-point algorithm is applied to the feasibility restoration phase problem

$$\min_{(x,s^+,s^-)} \quad \pi_{\mathcal{E}} \sum_{i\in\mathcal{E}} \left(s_i^+ + s_i^-\right) + \pi_{\mathcal{I}} \sum_{i\in\mathcal{I}} \left(s_i^+ + s_i^-\right) + \pi_{\mathcal{V}} \sum_{i\in\mathcal{V}} \left(s_i^+ + s_i^-\right) \tag{4.93a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) + s_{\mathcal{E}}^+ - s_{\mathcal{E}}^- = 0, \tag{4.93b}$$

$$c_{\mathcal{I}}(x) + s_{\mathcal{I}}^+ - s_{\mathcal{I}}^- \in [r_l, r_u], \tag{4.93c}$$

$$x + s_{\mathcal{V}}^+ - s_{\mathcal{V}}^- \in [b_l, b_u], \tag{4.93d}$$

$$s_{\mathcal{E}}^+, s_{\mathcal{E}}^-, s_{\mathcal{I}}^+, s_{\mathcal{I}}^-, s_{\mathcal{V}}^+, s_{\mathcal{V}}^- \geq 0. \tag{4.93e}$$

$s^+ = (s_{\mathcal{E}}^+, s_{\mathcal{I}}^+, s_{\mathcal{V}}^+)$ and $s^- = (s_{\mathcal{E}}^-, s_{\mathcal{I}}^-, s_{\mathcal{V}}^-)$ are vectors of slack variables (both in $\mathbb{R}^{n+m+k}$). To avoid a confusion in the notation, $\mathcal{V}$ denotes the set of variables, i.e. $x = (x_i)_{i\in\mathcal{V}}$. Obviously, (4.93) is always feasible and the original NLP (4.1) is feasible if (4.93) has a solution with objective value 0. $\pi_{\mathcal{E}}, \pi_{\mathcal{I}}, \pi_{\mathcal{V}} > 0$ are weighting factors. It is reasonable to avoid that the solution of the feasibility restoration problem is too far away from the last iterate $x^{(k)}$ at which the feasibility restoration phase is invoked. This can be done by incorporating an additional term like

$$\pi_d \left\| x - x^{(k)} \right\|_2^2, \quad \pi_d > 0, \tag{4.94}$$

in the objective function (4.93a) and by initializing the algorithm for solving (4.93) with the current iterate $x^{(k)}$. Problem (4.93) is then solved until an iterate is found that is acceptable to the filter of the original NLP. More details on feasibility restoration phases within filter line-search methods can be found in [41, 133, 134].

4.1.4 Updating the Barrier Parameter

The question *when* and *how* to update the barrier parameter μ is one of the most crucial algorithmic options for interior-point methods. One can distinguish between *adaptive* and *monotone* strategies. Adaptive strategies update the barrier parameter in every iteration. In practice, these methods are often more efficient but lack global convergence properties. Monotone strategies update the barrier parameter only after approximately solving the current barrier problem (cf. [91]). This approach,

known as the *Fiacco-McCormick approach*, provides a global convergence theory for decreasing sequences of barrier parameters $\mu \searrow 0$ [37].

In the interior-point method developed for this thesis, a *mixed* strategy is used that inherits both the practical performance of adaptive methods as well as the global convergence properties of the monotone Fiacco-McCormick approach. This is realized by following the ideas of Nocedal et al. [90]: The algorithm stays in the adaptive mode as long as it makes progress towards a solution of the NLP. If the algorithm fails to make progress it switches to a monotone mode where it stays until the algorithm recognizes that it can return to the adaptive mode.

This section first describes the idea of the overall globalization framework for a mixed strategy of barrier parameter updates and afterwards states some well-known update rules for the barrier parameter.

A Globalization Framework for Mixed Updates of the Barrier Parameter

The globalization framework for adaptive updates of the barrier parameter involves a second filter line-search algorithm using another filter $\mathcal{F}_{\text{NLP}}$ made up of the objective function f and the constraint violation θ_{pri} for the NLP (4.4). The corresponding filter line-search algorithm is less complicated than the one for the barrier problems described in Section 4.1.2. After computing the search direction as described in Section 4.1.1, a backtracking line-search is applied until a trial point is found that is acceptable to the filter $\mathcal{F}_{\text{NLP}}$. If no acceptable trial point can be found, the algorithm resets the barrier parameter μ to

$$\mu^{(k+1)} = \kappa_r \delta^{(k)}, \quad \delta^{(k)} := \frac{\left(s^{(k)}\right)^T u^{(k)} + \left(t^{(k)}\right)^T v^{(k)}}{2n + 2k}, \quad \kappa_r \in (0,1)\,, \tag{4.95}$$

and switches to the monotone mode. Now, the barrier parameter is fixed and the barrier problem is approximately solved using the filter line-search procedure described in Section 4.1.2. If the barrier problem is approximately solved, the barrier parameter is decreased and the next barrier problem is solved. During the solution of the barrier problems it is instantly checked if a new iterate is acceptable to the filter $\mathcal{F}_{\text{NLP}}$. If this is the case, the algorithm directly switches back to the adaptive mode without solving the current barrier problem. $\delta^{(k)}$ in (4.95) is called *duality measure* in the following. Algorithm 4 formally states the method described above.

The rest of this section describes several methods for updating the barrier parameter.

Mehrotra's Predictor-Corrector Method

Mehrotra's predictor-corrector (MPC) algorithm is proposed in [85] for interior-point methods for linear programming. Here, the idea is directly extended to the nonlinear case (cf. [90, 91]).

The method is embedded in the computation of search directions for the interior-point method. First, the KKT system (4.26) is solved with $\mu = 0$ yielding the so-called *affine scaling direction*

Algorithm 4: Globalization Filter Line-Search Method for Mixed Barrier Parameter Updates

Input : Primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$, constants $\kappa_b \in (0,1), \kappa_{m_1} > 0, \kappa_{m_2} > 0$.

1 Set backtracking line-search counter $l = 0$ and initialize $\alpha_{\text{pri},l}^{(k)} = \bar{\alpha}_{\text{pri}}^{(k)}$.
2 Compute filter margin $\kappa_m = \kappa_{m_1} \min(\kappa_{m_2}, (\theta_{\text{dual}}^{(k)})^2 + (\theta_{\text{pri}}^{(k)})^2 + (\theta_{\text{compl}}^{(k)})^2)$
3 **if** $\alpha_{\text{pri},l}^{(k)} < \alpha_{\text{pri}}^{\min}$ **then**
4 Stop the algorithm; no acceptable step length can be found.
5 Compute trial point $y_{\text{pri}}^{+} = y_{\text{pri}}^{(k)} + \alpha_{\text{pri},l}^{(k)} \Delta y_{\text{pri}}^{(k)}$.
6 **if** $(f(y_{\text{pri}}^{+}) + \kappa_m, \theta_{\text{pri}}(y_{\text{pri}}^{+}) + \kappa_m)$ is acceptable to $\mathcal{F}_{\text{NLP}}$ **then**
7 Update $\mathcal{F}_{\text{NLP}}$, **return** step length $\alpha_{\text{pri},l}^{(k)}$.
8 **else**
9 Set $\alpha_{\text{pri},l+1}^{(k)} = \kappa_b \alpha_{\text{pri},l}^{(k)}$.
10 Increase backtracking line-search counter $l \leftarrow l + 1$ and go to line 3.

$\Delta y_{\text{aff}}^{(k)}$. After temporarily using this direction with a maximum step length $\alpha_{\text{aff}}^{(k)}$ (determined by the fraction-to-the-boundary rule (4.15) with $\tau = 1$), i.e.

$$y_{\text{aff}}^{(k)} = y^{(k)} + \alpha_{\text{aff}}^{(k)} \Delta y_{\text{aff}}^{(k)}, \tag{4.96}$$

the so-called *affine duality measure*

$$\delta_{\text{aff}}^{(k)} := \frac{\left(s_{\text{aff}}^{(k)}\right)^T u_{\text{aff}}^{(k)} + \left(t_{\text{aff}}^{(k)}\right)^T v_{\text{aff}}^{(k)}}{2n + 2k} \tag{4.97}$$

can be computed. Together with the current duality measure $\delta^{(k)}$ the *centering parameter* $\sigma^{(k)}$ is determined by Mehrotra's heuristic (cf. [85, 91]);

$$\sigma^{(k)} = \left(\frac{\delta_{\text{aff}}^{(k)}}{\delta^{(k)}}\right)^3. \tag{4.98}$$

Finally, the update of the barrier parameter is computed by

$$\mu^{(k+1)} = \sigma^{(k)} \mu^{(k)}. \tag{4.99}$$

Next, the *corrector step* is computed. For this, system (4.26) is solved with $\mu = \mu^{(k+1)}$ and additional *corrector contributions* to the right-hand side quantities $\phi_l, \phi_u, \psi_l, \psi_u$ (cf. [91]). Thus, system (4.26) is solved with the modified right-hand sides

$$\phi_{l,\text{cor}}^{(k)} = \phi_l^{(k)} + \left(S_l^{(k)}\right)^{-1} \Delta S_{l,\text{aff}}^{(k)} \Delta U_{l,\text{aff}}^{(k)} e, \tag{4.100a}$$

$$\phi_{u,\text{cor}}^{(k)} = \phi_u^{(k)} + \left(S_u^{(k)}\right)^{-1} \Delta S_{u,\text{aff}}^{(k)} \Delta U_{u,\text{aff}}^{(k)} e, \tag{4.100b}$$

$$\psi_{l,\text{cor}}^{(k)} = \psi_l^{(k)} + \left(T_l^{(k)}\right)^{-1} \Delta T_{l,\text{aff}}^{(k)} \Delta V_{l,\text{aff}}^{(k)} e, \tag{4.100c}$$

$$\psi_{u,\text{cor}}^{(k)} = \psi_u^{(k)} + \left(T_u^{(k)}\right)^{-1} \Delta T_{u,\text{aff}}^{(k)} \Delta V_{u,\text{aff}}^{(k)} e. \tag{4.100d}$$

The corrector step is finally used as the search direction for the current iteration and $\mu^{(k+1)}$ from (4.99) is used as the new barrier parameter.

There are no additional matrix factorizations required in the corrector step. Only the barrier parameter μ changes, influencing only the right-hand side of the KKT system. This is a significant advantage of primal-dual methods that solve the KKT system in the form of (4.26). For primal methods, μ also appears in the KKT matrix (cf. Section 4.1.1) leading to separate matrix factorizations for the predictor and the corrector step, respectively.

The LOQO Rule

The LOQO rule

$$\mu^{(k+1)} = \kappa_{L_1} \min\left((1-\kappa_{L_2})\frac{1-\xi}{\xi}, 2\right)^3 \delta^{(k)}, \tag{4.101}$$

with

$$\kappa_{L_1} > 0, \quad \kappa_{L_2} \in (0,1) \tag{4.102}$$

and

$$\xi = \frac{1}{\delta^{(k)}} \min\left(\min_{i=1,\ldots,n} s_i^{(k)} u_i^{(k)}, \min_{i\in\mathcal{I}} t_i^{(k)} v_i^{(k)}\right) \tag{4.103}$$

is published by Vanderbei and Shanno in [128] and is implemented in the interior-point code LOQO [127].

The Ipopt Rule

In Ipopt [134], the barrier parameter is updated using the formula

$$\mu^{(k+1)} = \min\left(\kappa_{I_1}\mu^{(k)}, \left(\mu^{(k)}\right)^{\kappa_{I_2}}\right), \tag{4.104}$$

with constants $\kappa_{I_1} \in (0,1), \kappa_{I_2} \in (1,2)$.

Decreasing Sequences of Barrier Parameters and a Numerical Safeguard

To apply the barrier method convergence theory of Fiacco and McCormick, one has to ensure that a sequence of barrier problems with decreasing barrier parameters $\mu \searrow 0$ is solved. The Ipopt rule generates a decreasing sequence of barrier parameters since from (4.104) directly follows $\mu^{(k+1)} < \mu^{(k)}$. This is not the case for the LOQO rule and Mehrotra's predictor-corrector method. For these two methods a decrease of μ is guaranteed by additionally setting

$$\mu^{(k+1)} \leftarrow \min\left(\mu^{(k+1)}, \kappa_d \mu^{(k)}\right), \quad \kappa_d \in (0,1)\,. \tag{4.105}$$

Finally, an additional numerical safeguard is incorporated. If μ becomes very small, the barrier term diagonal matrices Φ and Ψ (cf. (4.36) and (4.41)) often become very small, too, leading

to ill-conditioned KKT systems. To avoid this situation, it is ensured that μ does not become unnecessary small by setting

$$\mu^{(k+1)} \leftarrow \max\left(\kappa_n \varepsilon_{\text{tol}}, \mu^{(k+1)}\right), \quad \kappa_n \in (0,1). \tag{4.106}$$

4.1.5 Starting Point Strategies

The performance of interior-point methods depends largely on the quality of the starting point. Two different situations can be distinguished depending on whether the user provides an initial primal estimate ($y_{\text{pri}}^{(0)}$) or a primal and dual estimate ($y_{\text{pri}}^{(0)}, y_{\text{dual}}^{(0)}$). If the user does not supply any starting point, $y_{\text{pri}}^{(0)} = 0$ is set. The main aspects for determining the quality of the starting point are the following:

Primal Feasibility Obviously, it is helpful to start the algorithm with an almost feasible point, i.e. with a starting point $y^{(0)}$ with $\theta_{\text{pri}}^{(0)}$ not being too large.

Interiority The proposed interior-point method guarantees that the primal and dual slack variables s, t and u, v stay strictly positive for each iterate (cf. (4.15)). Hence, the starting point should not be too close to its bounds. Otherwise, the practical experience shows that the first step lengths are often very short and thus do not make any significant progress towards a solution.

Centrality The concept of centrality refers to that of a *central path* (see [91]). An iterate is well centered if its distance to the central path, i.e. $\theta_{\text{compl}}(y;\mu)$, is not too large. This goal may conflict with the goal of interiority, especially for small barrier parameters μ.

Dual Feasibility Intuitively, a good starting point should not be too far away from dual feasibility, i.e. $\theta_{\text{dual}}^{(0)}$ should not be too large.

Since θ_{pri} only depends on primal variables, no dual quantities have to be given to reach the goal of small primal infeasibility. All other aspects depend on primal and dual estimates.

Standard initialization schemes try to address at least some of the aspects discussed above. Algorithm 5 states the default mode of the implementation developed for this thesis. Primal infeasibility concerning variable bounds and range constraints is addressed in lines 2, 3, 10 and 11. The perturbations p_{l_i} and p_{u_i} are chosen as in the Ipopt code [134]. Centrality and interiority is tried to achieve in lines 4–7 and 10–13. Finally, dual feasibility is addressed in line 14.

Algorithm 5: Default Interior-Point Method Initialization Scheme

Input : User provided primal starting point $\bar{x}$, initial barrier parameter $\mu^{(0)}$, minimum distance to the boundary $\kappa_d > 0$, constants $\kappa_{p_1}, \kappa_{p_2} > 0$.

for $i = 1, \ldots, n$ **do**
- Set $x_i^{(0)} = \max(\bar{x}_i, b_{l_i} + p_{l_i})$ with $p_{l_i} := \min(\kappa_{p_1} \max(1, |b_{l_i}|), \kappa_{p_2} (b_{u_i} - b_{l_i}))$.
- Set $x_i^{(0)} = \min(\bar{x}_i, b_{u_i} - p_{u_i})$ with $p_{u_i} := \min(\kappa_{p_1} \max(1, |b_{u_i}|), \kappa_{p_2} (b_{u_i} - b_{l_i}))$.
- Set $s_{l_i}^{(0)} = \max(x_i^{(0)} - b_{l_i}, \kappa_d)$.
- Set $s_{u_i}^{(0)} = \max(b_{u_i} - x_i^{(0)}, \kappa_d)$.
- Set $u_{l_i}^{(0)} = \max(\mu^{(0)}/s_{l_i}^{(0)}, \kappa_d)$.
- Set $u_{u_i}^{(0)} = \max(\mu^{(0)}/s_{u_i}^{(0)}, \kappa_d)$.

Evaluate $c_{\mathcal{I}}^{(0)} := c_{\mathcal{I}}(x^{(0)})$.

for all $i \in \mathcal{I}$ **do**
- Set $t_{l_i} = \max(c_i^{(0)} - r_{l_i}, \kappa_d)$.
- Set $t_{u_i} = \max(r_{u_i} - c_i^{(0)}, \kappa_d)$.
- Set $v_{l_i}^{(0)} = \max(\mu^{(0)}/t_{l_i}^{(0)}, \kappa_d)$.
- Set $v_{u_i}^{(0)} = \max(\mu^{(0)}/t_{u_i}^{(0)}, \kappa_d)$.

Evaluate $c_{\mathcal{E}}^{(0)} := c_{\mathcal{E}}(x^{(0)})$, compute

$$b = g^{(0)} - \left(\nabla c_{\mathcal{I}}^{(0)}\right)^T \left(v_l^{(0)} - v_u^{(0)}\right) - \left(u_l^{(0)} - u_u^{(0)}\right) \tag{4.107}$$

and solve the linear least-squares problem

$$z^{(0)} := \arg\min_z \frac{1}{2} \left\| \left(\nabla c_{\mathcal{E}}^{(0)}\right)^T z - b \right\|_2^2 \tag{4.108}$$

to initialize z.

4.1.6 Problem Scaling

The overall interior-point method is significantly affected by the scaling of the problem. However, it is not clear how to scale a nonlinear problem in a unique way leading to improved efficiency and robustness. Moreover, problem scaling mainly depends on the concrete model formulation. Nevertheless, an automatic problem scaling is implemented in a way such that the problem data $f, c_{\mathcal{E}}$ and $c_{\mathcal{I}}$ are replaced by

$$f \leftarrow \sigma_f f, \quad c_{\mathcal{E}} \leftarrow \Sigma_{\mathcal{E}} c_{\mathcal{E}}, \quad c_{\mathcal{E}} \leftarrow \Sigma_{\mathcal{I}} c_{\mathcal{I}}. \tag{4.109}$$

Here, $\sigma_f > 0$ is a positive real number and $\Sigma_{\mathcal{E}} = \operatorname{diag}((\sigma_i)_{i \in \mathcal{E}}) > 0 \in \mathbb{R}^{m \times m}$ and $\Sigma_{\mathcal{I}} = \operatorname{diag}((\sigma_i)_{i \in \mathcal{I}}) > 0 \in \mathbb{R}^{k \times k}$ are positive definite diagonal matrices. In the concrete implementation for this thesis, the scaling factors are computed following the ideas of the Ipopt code [134];

$$\sigma_f = \min\left(1, \frac{\kappa_g}{\|g(x^{(0)})\|_\infty}\right), \tag{4.110a}$$

$$\sigma_i = \min\left(1, \frac{\kappa_g}{\|c_i^{(0)}\|_\infty}\right), \quad i \in \mathcal{E}, \tag{4.110b}$$

$$\sigma_i = \min\left(1, \frac{\kappa_g}{\|c_i^{(0)}\|_\infty}\right), \quad i \in \mathcal{I}. \tag{4.110c}$$

$\kappa_g > 0$ is a constant such that all components of the scaled gradients are less than or equal to κ_g at the starting point.

4.1.7 Heuristics and Algorithmic Details

This section deals with small enhancements of the presented interior-point method. These enhancements are designed to improve the method on some difficult instances but do not harm the performance and robustness of the method on other instances.

Jamming and Shifting of Slack Variables

In some cases the problem appears that some of the primal slack variables are very close to their bounds and that the corresponding search directions are negative with large absolute values. This effect is called *jamming*. To handle it, Benson et al. [9] suggest to shift the bound slack variables $s_{l_i}, i \in I_{s_l} \subset \{1, \ldots, n\}$, and $s_{u_i}, i \in I_{s_u} \subset \{1, \ldots, n\}$, as well as the range slack variables $t_{l_i}, i \in I_{t_l} \subset \mathcal{I}$, and $t_{u_i}, i \in I_{t_u} \subset \mathcal{I}$. Here,

$$I_{s_l} := \left\{ i \in \{1, \ldots, n\} : s_{l_i} < \kappa_{s_1} \text{ and } \frac{\Delta s_{l_i}}{s_{l_i}} < -\kappa_{s_2} \right\}, \tag{4.111a}$$

$$I_{s_u} := \left\{ i \in \{1, \ldots, n\} : s_{u_i} < \kappa_{s_1} \text{ and } \frac{\Delta s_{u_i}}{s_{u_i}} < -\kappa_{s_2} \right\}, \tag{4.111b}$$

$$I_{t_l} := \left\{ i \in \mathcal{I} \colon t_{l_i} < \kappa_{s_1} \text{ and } \frac{\Delta t_{l_i}}{t_{l_i}} < -\kappa_{s_2} \right\}, \tag{4.111c}$$

$$I_{t_u} := \left\{ i \in \mathcal{I} \colon t_{u_i} < \kappa_{s_1} \text{ and } \frac{\Delta t_{u_i}}{t_{u_i}} < -\kappa_{s_2} \right\}, \tag{4.111d}$$

with constants $\kappa_{s_1}, \kappa_{s_2} > 0$. Typically, κ_{s_1} is chosen to be small and κ_{s_2} is chosen to be large. The shifting is then defined by

$$s_{l_i} \leftarrow s_{l_i} + \frac{n - |I_{s_l}|}{\sum_{i \in \{1,\ldots,n\} \setminus I_{s_l}} s_{l_i}}, \qquad s_{u_i} \leftarrow s_{u_i} + \frac{n - |I_{s_u}|}{\sum_{i \in \{1,\ldots,n\} \setminus I_{s_u}} s_{u_i}}, \tag{4.112a}$$

$$t_{l_i} \leftarrow t_{l_i} + \frac{k - |I_{t_l}|}{\sum_{i \in \mathcal{I} \setminus I_{t_l}} t_{l_i}}, \qquad t_{u_i} \leftarrow t_{u_i} + \frac{k - |I_{t_u}|}{\sum_{i \in \mathcal{I} \setminus I_{t_u}} t_{u_i}}. \tag{4.112b}$$

Resetting of Dual Variables

For some instances diverging dual iterates and search directions can be noticed in practice, whereas primal search directions are very short and primal feasibility is already reached. This situation can mostly be observed for

1. badly scaled instances,
2. instances without strict relative interior or
3. instances not satisfying standard constraint qualifications like the LICQ condition.

In some of these cases it might be helpful to reset the dual variables to $z = 0$ and

$$u_{l_i} = \begin{cases} \kappa_d, & \text{if } s_{l_i} < \kappa_{e_1} \\ \min(u_{l_i}, \kappa_d), & \text{else} \end{cases}, \qquad u_{u_i} = \begin{cases} \kappa_d, & \text{if } s_{u_i} < \kappa_{e_1} \\ \min(u_{u_i}, \kappa_d), & \text{else} \end{cases}, \tag{4.113a}$$

$$v_{l_i} = \begin{cases} \kappa_d, & \text{if } t_{l_i} < \kappa_{e_1} \\ \min(v_{l_i}, \kappa_d), & \text{else} \end{cases}, \qquad v_{u_i} = \begin{cases} \kappa_d, & \text{if } t_{u_i} < \kappa_{e_1} \\ \min(v_{u_i}, \kappa_d), & \text{else} \end{cases}, \tag{4.113b}$$

whenever

$$\theta_{\text{pri}} < \kappa_{e_2}, \quad \theta_{\text{dual}} > \kappa_{e_3} \quad \text{and} \quad \|y_{\text{dual}}\| > \kappa_{e_4} \tag{4.114}$$

holds. κ_d is the same parameter as in Algorithm 5 and $\kappa_{e_1}, \kappa_{e_2}, \kappa_{e_3}$ and κ_{e_4} can be chosen arbitrarily. Typically, κ_{e_1} and κ_{e_2} are chosen to be small and κ_{e_3} and κ_{e_4} are chosen to be large.

The Special Case of Bound-Feasible Starting Points

For *huge*-scale applications (see Section 6.2 for an example) it can be crucial to scale down the size of the vectors y and Δy. This can be realized within the so-called *bound-feasible* case. If the algorithm starts with a bound-feasible set of primal variables, i.e. $x \in [b_l, b_u]$, the algorithm can be modified so that the feasibility with respect to the variable bounds is conserved from iteration to iteration. For this, identifying

$$s_l = x - b_l \quad \text{and} \quad s_u = b_u - x, \tag{4.115}$$

yields

$$\beta_l = 0, \qquad \Delta s_l = \Delta x, \qquad -\Delta u_l = \Phi_l \Delta x + \phi_l, \qquad \bar{\phi}_l = \phi_l, \tag{4.116}$$

$$\beta_u = 0, \qquad \Delta s_u = -\Delta x, \qquad -\Delta u_u = -\Phi_u \Delta x + \phi_u, \qquad \bar{\phi}_u = \phi_u. \tag{4.117}$$

In this case, s_l and s_u are not required as iteration variables. Thus, s_l, s_u, Δs_l and Δs_u do not have to be stored explicitly. Instead, simple operations using $x, \Delta x$, b_l and b_u are performed as stated above. With these modifications, the KKT system (4.26) reduces to a system in $\mathbb{R}^{3n+m+4k}$.

The Special Case of Quadratic and Linear Problems

The presented interior-point method can be applied to linear and (convex) quadratic optimization problems, too. These problems are given as

$$\min_{x \in \mathbb{R}^n} \quad \frac{1}{2} x^T H x + c^T x \tag{4.118a}$$

$$\text{s.t.} \quad Ax - a = 0, \tag{4.118b}$$

$$Bx \in [r_l, r_u], \tag{4.118c}$$

$$x \in [b_l, b_u], \tag{4.118d}$$

with $x, c, b_l, b_u \in \mathbb{R}^n, H \in \mathbb{R}^{n \times n}, A \in \mathbb{R}^{m \times n}, a \in \mathbb{R}^m, B \in \mathbb{R}^{k \times n}$ and $r_l, r_u \in \mathbb{R}^k$. For linear problems, $H = 0$ holds. The KKT system (4.26) as well as the according reduction and expansion techniques stay the same if one identifies

$$f(x) = \frac{1}{2} x^T H x + c^T x, \qquad c_{\mathcal{E}}(x) = Ax - a, \qquad c_{\mathcal{I}}(x) = Bx. \tag{4.119a}$$

$$\nabla^2_{xx} \mathcal{L} = H, \qquad \nabla c_{\mathcal{E}} = A, \qquad \nabla c_{\mathcal{I}} = B. \tag{4.119b}$$

In addition, a globalization strategy like a filter line-search is not required for linear and convex quadratic problems.

IPM Reduction and Expansion

Solving the KKT system (4.26) and storing the iterates as well as the right-hand sides of the KKT system and the search directions are the most time and memory consuming tasks in interior-point methods. Thus, it is reasonable to employ efficient solution techniques and vector memory management. As it is already mentioned, the operations for solving the complete KKT system (4.26) consist of a *reduction*, a *solution* and an *expansion* step. Table 4.1 gives an overview of the memory management within a complete iteration of the interior-point method. The number above the arrows defines the computation order for each step. All operations with the same number can be done in parallel because they are independent of each other. The notation $a \to b$ means that the memory of vector a is overwritten by b and that the computation of b requires a. Downward ($\searrow$) or upward arrows ($\nearrow$) stand for a swapping of the corresponding memory blocks.

For the bound-feasible case, the full KKT system shrinks to $\mathbb{R}^{3n+m+4k}$. The vector management of the reduction, solution and expansion steps changes as it is shown in Table 4.2.

4.1.8 The Complete Interior-Point Algorithm

By now, the complete interior-point method can be stated (see Algorithm 6).

Block size	IPM reduction			KKT solution			IPM expansion		
n	$\nabla_x \mathcal{L}$	$\xrightarrow{2}$	γ	γ	$\longrightarrow$	Δx	Δx	$\longrightarrow$	Δx
n	ϕ_l	$\xrightarrow{1}$	$\bar{\phi}_l$	$\bar{\phi}_l$	$\searrow$	β_l	β_l	$\xrightarrow{5}$	Δs_l
n	ϕ_u	$\xrightarrow{1}$	$\bar{\phi}_u$	$\bar{\phi}_u$	$\searrow$	β_u	β_u	$\xrightarrow{5}$	Δs_u
k	ψ_l	$\xrightarrow{5}$	$\hat{\psi}$	$\hat{\psi}$	$\longrightarrow$	$\hat{\psi}$	$\hat{\psi}$	$\xrightarrow{1}$	Δt_l
k	ψ_u	$\xrightarrow{4}$	$\hat{\psi}_u$	$\hat{\psi}_u$	$\searrow$	ρ	ρ	$\xrightarrow{2}$	Δt_u
m	α	$\longrightarrow$	α	α	$\longrightarrow$	$-\Delta z$	$-\Delta z$	$\longrightarrow$	$-\Delta z$
n	β_l	$\longrightarrow$	β_l	β_l	$\nearrow$	$\bar{\phi}_l$	$\bar{\phi}_l$	$\xrightarrow{5}$	$-\Delta u_l$
n	β_u	$\longrightarrow$	β_u	β_u	$\nearrow$	$\bar{\phi}_u$	$\bar{\phi}_u$	$\xrightarrow{5}$	$-\Delta u_u$
k	ρ_l	$\xrightarrow{6}$	$\hat{\rho}$	$\hat{\rho}$	$\longrightarrow$	$-\Delta \hat{v}$	$-\Delta \hat{v}$	$\xrightarrow{4}$	$-\Delta v_l$
k	ρ_u	$\xrightarrow{3}$	ρ	ρ	$\nearrow$	$\hat{\psi}_u$	$\hat{\psi}_u$	$\xrightarrow{3}$	$-\Delta v_u$

Table 4.1: Vector management for IPM reduction, reduced KKT system solution and IPM expansion.

Block size	IPM reduction			KKT solution			IPM expansion		
n	$\nabla_x \mathcal{L}$	$\xrightarrow{1}$	γ	γ	$\xrightarrow{1}$	Δx	Δx	$\longrightarrow$	Δx
k	ψ_l	$\xrightarrow{3}$	$\hat{\psi}$	$\hat{\psi}$	$\longrightarrow$	$\hat{\psi}$	$\hat{\psi}$	$\xrightarrow{1}$	Δt_l
k	ψ_u	$\xrightarrow{2}$	$\hat{\psi}_u$	$\hat{\psi}_u$	$\overset{2}{\searrow}$	ρ	ρ	$\xrightarrow{2}$	Δt_u
m	α	$\longrightarrow$	α	α	$\xrightarrow{1}$	$-\Delta z$	$-\Delta z$	$\longrightarrow$	$-\Delta z$
n	ϕ_l	$\longrightarrow$	ϕ_l	ϕ_l	$\longrightarrow$	ϕ_l	ϕ_l	$\xrightarrow{1}$	$-\Delta u_l$
n	ϕ_u	$\longrightarrow$	ϕ_u	ϕ_u	$\longrightarrow$	$\bar{\phi}_u$	$\bar{\phi}_u$	$\xrightarrow{1}$	$-\Delta u_u$
k	ρ_l	$\xrightarrow{4}$	$\hat{\rho}$	$\hat{\rho}$	$\xrightarrow{1}$	$-\Delta \hat{v}$	$-\Delta \hat{v}$	$\xrightarrow{4}$	$-\Delta v_l$
k	ρ_u	$\xrightarrow{1}$	ρ	ρ	$\overset{2}{\nearrow}$	$\hat{\psi}_u$	$\hat{\psi}_u$	$\xrightarrow{3}$	$-\Delta v_u$

Table 4.2: Vector management for IPM reduction, reduced KKT system solution and IPM expansion in the bound-feasible case.

Algorithm 6: Filter Line-Search Interior-Point Algorithm with Mixed Barrier Parameter Updates

Input : User provided starting point $\bar{x}$ for the original NLP (4.1), initial barrier parameter $\mu^{(0)}$, vector of algorithmic constants κ.

Set iteration counter $k = 0$, initialize filter $\mathcal{F}_{\text{NLP}}$ using (4.76) and (4.77), set μ-mode = `adaptive`.
Call Algorithm 5 with $\bar{x}$ and $\mu^{(0)}$ to obtain the starting point $y^{(0)}$.
while NLP termination criterion (4.23) does not hold **do**
    Increase iteration counter $k \leftarrow k + 1$.
    if μ-mode = `adaptive` **then** `/* adaptive mode */`
        Compute $\mu^{(k)}$ by any rule (cf. Section 4.1.4) and update τ using (4.16).
        Compute search direction $\Delta y^{(k)}$ as described in Section 4.1.1.
        Compute maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
        Call Algorithm 4 with primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$ to obtain $\alpha_{\text{pri}}^{(k)}$.
        if Algorithm 4 succeeds **then** `/* stay adaptive */`
            Compute new primal and dual iterates $y_{\text{pri}}^{(k+1)}, y_{\text{dual}}^{(k+1)}$ using (4.17) with primal step length $\alpha_{\text{pri}}^{(k)}$ and dual step length $\bar{\alpha}_{\text{dual}}^{(k)}$.
        else `/* switch to monotone mode */`
            Set μ-mode $\leftarrow$ `monotone`, reset barrier parameter $\mu^{(k+1)}$ using (4.95), update τ using (4.16), reset barrier problem filter $\mathcal{F}_\mu$ using (4.76) and (4.77).
    else `/* monotone mode */`
        if barrier problem termination criterion (4.22) holds **then**
            Compute $\mu^{(k)}$ by any rule (cf. Section 4.1.4) and update τ using (4.16).
            Reset barrier problem filter $\mathcal{F}_\mu$ using (4.76) and (4.77).
        Compute search direction $\Delta y^{(k)}$ as described in Section 4.1.1.
        Compute maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
        Call Algorithm 3 with primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$ to obtain $\alpha_{\text{pri}}^{(k)}$.
        if Algorithm 3 succeeds **then**
            Compute new primal and dual iterates $y_{\text{pri}}^{(k+1)}, y_{\text{dual}}^{(k+1)}$ using (4.17) with primal step length $\alpha_{\text{pri}}^{(k)}$ and dual step length $\bar{\alpha}_{\text{dual}}^{(k)}$.
            if $y_{\text{pri}}^{(k+1)}$ is acceptable to the filter $\mathcal{F}_{\text{NLP}}$ **then**
                Set μ-mode $\leftarrow$ `adaptive`.
        else
            Go to feasibility restoration phase.
return optimal solution $y^{(k)}$.

4.1.9 Convergence Analysis

The described interior-point method is strongly oriented towards the implementation described in [134] and the convergence results given in [133] for the interior-point code Ipopt. By this reason, the proofs of global and local convergence are not given here again but only the main assumptions and results are stated.

The following assumptions are taken directly from [133] and allow to prove global convergence results for the barrier problem (4.6) for a fixed barrier parameter μ. By driving μ to zero it is clear that the algorithm finally reaches an optimal solution of the original NLP (4.1).

Assumption 1. *Let $y^{(0)}$ be the starting point and $(y_{\text{pri}}^{(k)})$ the sequence of iterates generated by Algorithm 6 (with $\kappa_{s_1} = 1$ in (4.73)). Moreover, assume that the feasibility restoration phase always terminates successfully and that Algorithm 6 does not stop with a KKT point at line 3.*

(A1) There exists an open set $\mathcal{O} \subset \mathbb{R}^{n_p}$ with $[y_{\text{pri}}^{(k)}, y_{\text{pri}}^{(k)} + \bar{\alpha}_{\text{pri}}^{(k)} \Delta y_{\text{pri}}^{(k)}] \subset \mathcal{O}$ for all iterations k and $f, c_{\mathcal{E}}, c_{\mathcal{I}}$ are differentiable and bounded on $\mathcal{O}$ and their derivatives are bounded and Lipschitz-continuous on $\mathcal{O}$.

(A2) The Hessians $H^{(k)}$ of the Lagrangian of the original NLP (4.1) or, if an approximation is used, the matrices $W^{(k)}$ approximating this Lagrangian are uniformly bounded.

(A3) The matrices $\Omega_H^{(k)} = \operatorname{diag}(H^{(k)}, \Phi_l^{(k)}, \Phi_u^{(k)}, \Psi_l^{(k)}, \Psi_u^{(k)})$ with $\Phi_l^{(k)}, \Phi_u^{(k)}, \Psi_l^{(k)}, \Psi_u^{(k)}$ defined by (4.29) are uniformly positive definite on the null-space of the Jacobian of the constraints of (4.6), i.e. on $\ker(\nabla c^{(k)})$ with

$$\nabla c^{(k)} = \begin{bmatrix} \nabla c_{\mathcal{E}}^{(k)} & & & & \\ I & -I & & & \\ -I & & -I & & \\ \nabla c_{\mathcal{I}}^{(k)} & & & -I & \\ -\nabla c_{\mathcal{I}}^{(k)} & & & & -I \end{bmatrix}. \tag{4.120}$$

Equivalently, one may assume that there exists a constant $C_1 > 0$ such that

$$\lambda_{\min}((Z^{(k)})^T \Omega_H^{(k)} Z^{(k)}) \geq C_1 \tag{4.121}$$

holds for all iterations k. Here, $Z^{(k)}$ is a matrix whose columns build an orthonormal basis of the null-space of $\nabla c^{(k)}$ and $\lambda_{\min}(M)$ denotes the smallest eigenvalue of the symmetric matrix M.

(A4) There exists a constant $C_2 > 0$ such that $\sigma_{\min}(\nabla c^{(k)}) \geq C_2$ for all iterations k. Here, $\sigma_{\min}(M)$ denotes the smallest singular value of the matrix M.

(A5) There exists a constant $C_3 > 0$ such that Algorithm 6 does not enter the feasibility restoration phase if $\theta_{\text{pri}}^{(k)} \leq C_3$.

(A6) The slack variables $s^{(k)}$ and $t^{(k)}$ are bounded for all iterations k.

(A7) The gradients of the active constraints, i.e. $\nabla c(y^*)$ and

$$e_{i+n} \text{ for all } i \text{ with } s_{l_i}^* = 0, \qquad e_{i+2n} \text{ for all } i \text{ with } s_{u_i}^* = 0, \tag{4.122a}$$

$$e_{i+3n} \text{ for all } i \text{ with } t_{l_i}^* = 0, \qquad e_{i+3n+k} \text{ for all } i \text{ with } t_{u_i}^* = 0 \tag{4.122b}$$

are linearly independent for all feasible limit points y^* of $(y^{(k)})$.

(A8) There exist constants $\delta_1, \delta_2 > 0$ such that whenever the feasibility restoration phase is called in an iteration k with $\theta_{\text{pri}}^{(k)} < \delta_2$, it returns a new iterate with $s_i^{(k+1)} \geq s_i^{(k)}$ and $t_i^{(k+1)} \geq t_i^{(k)}$ for all components $s_i^{(k)}$ and $t_i^{(k)}$ with $s_i^{(k)} \leq \delta_1$ and $t_i^{(k)} \leq \delta_1$.

These assumptions are slightly stronger than the ones used by Wächter and Biegler in [133]. More precisely, Wächter and Biegler additionally consider the case in which the feasibility restoration phase may be called after an unsuccessful computation of the search direction.

Note that it is remarked in [133] that the "primal Hessian" in (A3) can also be replaced by the "primal-dual Hessian" (i.e. by using $\Phi_l^{(k)}, \Phi_u^{(k)}, \Psi_l^{(k)}, \Psi_u^{(k)}$ as defined in (4.25)) under additional assumptions.

In [133], the following global convergence theorems are proved.

Theorem 7 (Feasibility: [133], Theorem 1)**.** *Suppose Assumption 1 holds. Then* $\lim_{k\to\infty} \theta_{\text{pri}}^{(k)} = 0$.

Theorem 8 (Optimality: [133], Theorem 2)**.** *Suppose Assumption 1 holds and let $\chi(y_{\text{pri}})$ be the first-order criticality measure defined by*

$$\chi(y_{\text{pri}}) = \left\| - \left(Z^T H Z\right)^{-1} Z^T \left(g - H\left(\left(\nabla c Y\right)^{-1} c\right)\right)\right\|_2 . \tag{4.123}$$

Here, the columns of Z build an orthonormal basis of the null-space of ∇c and Y is a matrix such that $[Y\,Z]$ is an orthonormal basis of $\mathbb{R}^{n_p}$. Then $\liminf_{k\to\infty} \chi^{(k)} = 0$.

In [133] it is also shown that χ is a first-order criticality measure under Assumption 1. Additionally, the following theorem holds, guaranteeing that the barrier method is well-defined. This means that the objective function φ_μ of the barrier problem (4.6) is well-defined.

Theorem 9 (Well-Posedness of Algorithm 6: [133], Theorem 3)**.** *Suppose Assumption 1 holds. Then there exists a constant $\varepsilon > 0$ such that $(s^{(k)}, t^{(k)}) \geq \varepsilon e$ holds in every iteration k.*

Finally, Nocedal et al. proved a global convergence result in [90] for the mixed barrier parameter update strategy.

Theorem 10. *Suppose Assumption 1 holds and that the monotone mode in Algorithm 6 always terminates successfully. Then the KKT error $\theta_{\text{dual}}^2 + \theta_{\text{pri}}^2 + \theta_{\text{compl}}^2$ converges to zero.*

4.2 An Interior-Point Method for Nonsmooth Nonlinear Problems

In this section an extended interior-point algorithm for a special subclass of nonsmooth nonlinear optimization problems is presented.

Most of the existing methods for nonsmooth optimization with convex objective function and convex constraints are so-called *bundle-methods*. This type of methods tries to approximate the nonsmooth functions by a bundle of hyperplanes. See the books of Hiriart-Urruty and Lemaréchal [55, 56] for the details. A first attempt to use filters to force global convergence for nonsmooth optimization algorithms is made by Fletcher and Leyffer in [40]. A nice overview of software for nonsmooth optimization can be found in [64].

The method proposed in the following is based on Algorithm 6. To be more specific, only those algorithmic building blocks of Algorithm 6 are replaced or modified that are required to handle nonsmooth constrained problems.

The section is organized as follows. In Section 4.2.1 the problem class is defined that is considered in the following. The next section describes the main algorithmic strategy. Here, it should be remarked that some aspects of the presented strategy are influenced by discussions with Andreas Wächter. The algorithmic building blocks that are especially designed to handle nonsmooth constrained problems are discussed in Section 4.2.3. Finally, Section 4.2.4 states the complete algorithm.

4.2.1 Definition of the Problem Class

The aim of this section is to define a special subclass of nonsmooth constrained optimization problems. This subclass consists of problems whose constraints $c = (c_{\mathcal{E}}, c_{\mathcal{I}})$ are piecewise smooth (i.e. piecewise $\mathcal{C}^2$) and locally Lipschitz-continuous. Furthermore, the constraints have to satisfy two additional properties. These properties are the *separable nonsmoothness property* and the existence of so-called *localization functions*. Both will be defined and illustrated in the following.

Definition 16 (Separable Nonsmoothness Property)**.** *Let $c_i(x) = 0$ be an equality constraint with piecewise smooth and locally Lipschitz-continuous c_i. The constraint $c_i(x) = 0$ satisfies the* separable nonsmoothness property *if there exist*

1. *a single variable $x_{i_\nu}, i_\nu \in \{1, \ldots, n\}$, and a univariate, piecewise smooth and locally Lipschitz-continuous, convex function $\theta_i : \mathbb{R} \to \mathbb{R}$ depending on $x_{i_\nu} \in \mathbb{R}$ (i.e. $\theta_i = \theta_i(x_{i_\nu})$),*

2. *a smooth function $\tilde{c}_i : \mathbb{R}^{n+1} \to \mathbb{R}$ depending on $x \in \mathbb{R}^n$ and on an additional auxiliary variable $x_{i_a} \in \mathbb{R}$ (i.e., $i_a \notin \{1, \ldots, n\}$),*

such that $c_i(x) = 0$ *can be equivalently restated as*

$$\tilde{c}_i(x, x_{i_a}) = 0 \quad \textit{with} \quad x_{i_a} \quad \textit{subject to} \quad x_{i_a} \pm \theta_i(x_{i_\nu}) = 0. \tag{4.124}$$

$\tilde{c}_i$ *is called a* lifting *of* c_i*. The analogous definition also applies to inequality constraints.*

The motivation of this definition is to formalize the situation in which the nonsmoothness of a constraint can be shifted into a univariate, piecewise smooth and locally Lipschitz-continuous function. This makes it possible to construct a modified stationarity test for piecewise smooth and locally Lipschitz-continuous constrained problems (cf. Theorem 6) that can be implemented in an efficient way. The following example illustrates the definition.

Example 2. *Consider the piecewise smooth and locally Lipschitz-continuous constraint* $c_i(x_1, x_2) = 0$ *with*

$$c_i(x_1, x_2) = x_1^2 + \min(x_2, 0) - 42. \tag{4.125}$$

With

$$x_3 + \theta_i(x_2) = 0, \quad \theta_i(x_2) := -\min(x_2, 0), \tag{4.126}$$

and

$$\tilde{c}_i(x_1, x_2, x_3) := x_1^2 + x_3 - 42, \tag{4.127}$$

one can see that c_i *satisfies the separable nonsmoothness property:*

$$\begin{aligned} & \tilde{c}_i(x_1, x_2, x_3) = x_1^2 + x_3 - 42 = 0 && \text{(4.128a)} \\ \iff \quad & x_1^2 + \min(x_2, 0) - 42 = 0 && \text{(4.128b)} \\ \iff \quad & c_i(x_1, x_2) = 0. && \text{(4.128c)} \end{aligned}$$

In this example $i_\nu = 2$ *and* $i_a = 3$ *holds.*

The next example lists some functions that are often used for modeling of nonsmooth aspects and that can be used as the function θ_i in Definition 16.

Example 3. *The functions* $\min(x_1, 0), \max(x_1, 0)$ *and the absolute value function* $|x_1|$ *are univariate, piecewise smooth and locally Lipschitz-continuous functions.* $\max(x_1, 0)$ *and* $|x_1|$ *are convex.* $\min(x_1, 0)$ *is concave but fits into the situation of Definition 16 due to the arbitrary sign of* θ_i *in* (4.124) *because* $-\min(x_1, 0)$ *is convex.*

Definition 16 is not only applicable to constraints in which the nonsmooth part depends on only one variable. The next example presents reformulations for two multivariate nonsmooth constraints that often appear in practice.

Example 4. *1. Consider the bivariate absolute value constraint* $c(x_1, x_2) = |x_1 - x_2| = 0$. *By introducing an auxiliary variable* x_3 *subject to the constraint*

$$c_{\text{aux}}(x_1, x_2, x_3) = x_1 - x_2 - x_3 = 0 \tag{4.129}$$

one can rewrite $c(x_1, x_2) = 0$ *equivalently by*

$$c_{\text{aux}}(x_1, x_2, x_3) = x_1 - x_2 - x_3 = 0 \quad \textit{and} \quad \hat{c}(x_3) = |x_3| = 0. \tag{4.130}$$

$\hat{c}(x_3)$ *fits into the framework of Definition 16.*

2. Consider the bivariate constraint $c(x_1, x_2) = \max(x_1, x_2) = 0$. *One can easily see that*

$$c(x_1, x_2) = \frac{x_1 + x_2 + |x_1 - x_2|}{2} \tag{4.131}$$

holds. Using the reformulation of the preceding example one sees that this fits into the framework of Definition 16, too.

In the following, smooth and nonsmooth constraints of a problem are explicitly distinguished. For this, the set of nonsmooth constraints is denoted by the index set $\mathcal{N} \subset (\mathcal{E} \cup \mathcal{I})$. The second property that the constraints under consideration have to satisfy is that there exist so-called localization functions for all nonsmooth constraints $c_i, i \in \mathcal{N}$.

Definition 17 (Localization Functions)**.** *Let* $c_i, i \in \mathcal{N}$, *be a piecewise smooth and locally Lipschitz-continuous function. A function* $\ell_i(x) : \mathbb{R}^n \to \mathbb{R}$ *is called a* localization function *for* c_i *if and only if*

$$c_i \in \mathcal{C}^2(\mathcal{S}(\ell_i)), \tag{4.132}$$

where

$$\mathcal{S}(\ell_i) := \{x \in \mathbb{R}^n : \ell_i(x) \neq 0\} \tag{4.133}$$

denotes the set of points at which ℓ_i *does not vanish. The set*

$$\mathcal{K}_{c_i} = \{x \in \mathbb{R}^n : \ell_i(x) = 0\} \tag{4.134}$$

is the set of points at which c_i *fails to be differentiable. The set*

$$\mathcal{K} = \{x \in \mathbb{R}^n : \exists i \in \mathcal{N} \text{ with } \ell_i(x) = 0\} = \bigcup_{i \in \mathcal{E} \cup \mathcal{I}} \mathcal{K}_{c_i} \tag{4.135}$$

is the set of points at which at least one constraint c_i *fails to be differentiable.*

Note that it is not possible to use the support $\operatorname{supp}(\ell_i)$ in (4.132), because $\operatorname{supp}(\ell_i)$ is defined to be the closure of $\mathcal{S}$. The purpose of localization functions is that one can easily check whether a constraint c_i is smooth or not at a given point $x \in \mathbb{R}^n$. The next example shows that it is easy to find localization functions for a lot of nonsmooth functions that appear in practice.

Example 5. *1. The function $\ell(x_1) = x_1$ is a localization function for the constraints*

$$c_1(x_1) = |x_1| = 0, \quad c_2(x_1) = \min(x_1, 0) = 0, \quad c_3(x_1) = \max(x_1, 0) = 0. \tag{4.136}$$

2. The function $\ell(x_1, x_2) = x_1 - x_2$ is a localization function for the constraints

$$c_4(x_1, x_2) = |x_1 - x_2| = 0, \tag{4.137a}$$

$$c_5(x_1, x_2) = \min(x_1, x_2) = 0, \tag{4.137b}$$

$$c_6(x_1, x_2) = \max(x_1, x_2) = 0. \tag{4.137c}$$

The rest of this section treats problem (4.1) where $f \in \mathcal{C}^2$ and the constraints $c = (c_\mathcal{E}, c_\mathcal{I})$ satisfy the following conditions:

(N1) c is piecewise smooth, i.e. piecewise $\mathcal{C}^2$, and locally Lipschitz-continuous.

(N2) All nonsmooth constraints $c_i, i \in \mathcal{N}$, satisfy the separable nonsmoothness property.

(N3) There exist localization functions for all nonsmooth constraints $c_i, i \in \mathcal{N}$.

Definition 18. *A problem of type* (4.1) *satisfying (N1)–(N3) is called an* optimization problem with locatable and separable nonsmoothness.

For the description of the interior-point algorithm that solves optimization problems with locatable and separable nonsmoothness it is assumed that the problem is already given in the form of (4.124). Thus, the problem is already given with auxiliary variables x_{i_a} and auxiliary constraints $x_{i_a} \pm \theta_i(x_{i_\nu}) = 0$ for all nonsmooth constraints c_i. In particular, all nonsmooth constraints are already replaced by their liftings. This leads to the following general problem formulation:

$$\min_{(x, x_a)} \quad f(x) \tag{4.138a}$$

$$\text{s.t.} \quad c_i(x) = 0 \quad \forall i \in \mathcal{E} \setminus \mathcal{N}, \tag{4.138b}$$

$$c_i(x) \in [r_l, r_u] \quad \forall i \in \mathcal{I} \setminus \mathcal{N}, \tag{4.138c}$$

$$\tilde{c}_i(x, x_{i_a}) = 0 \quad \forall i \in \mathcal{E} \cap \mathcal{N}, \tag{4.138d}$$

$$\tilde{c}_i(x, x_{i_a}) \in [r_l, r_u] \quad \forall i \in \mathcal{I} \cap \mathcal{N}, \tag{4.138e}$$

$$\vartheta_i(x_{i_a}, x_{i_\nu}) = x_{i_a} \pm \theta_i(x_{i_\nu}) = 0 \quad \forall i \in \mathcal{N}, \tag{4.138f}$$

$$x \in [b_l, b_u], \tag{4.138g}$$

$$x \in \mathbb{R}^n, x_a = (x_{i_a})_{i \in \mathcal{N}} \in \mathbb{R}^{|\mathcal{N}|}. \tag{4.138h}$$

In the following, the variable vector is abbreviated by $\hat{x} := (x, x_a) \in \mathbb{R}^{n_\mathcal{N}}$ with $n_\mathcal{N} = n + |\mathcal{N}|$. If the distinction between x and x_a is not required, the variable vector is also abbreviated by x again. Notice that the assumption of a smooth objective function is without loss of generality:

A nonsmooth objective function $f(x)$ can easily be substituted by an auxiliary variable x_f that is minimized and that is subject to the additional nonsmooth equality constraint $c_f(x, x_f) = x_f - f(x) = 0$. Obviously, the introduced constraint $c_f(x, x_f)$ has to satisfy the conditions (N1)–(N3).

4.2.2 Basic Algorithmic Strategy

The main idea of the following is to modify the basic interior-point method discussed in Section 4.1 as little as possible and as much as necessary such that it can handle optimization problems with locatable and separable nonsmoothness. Thus, only those algorithmic building blocks of Algorithm 6 are replaced that are faced with nonsmooth aspects of the problem.

The main idea of the modification of the method is that the algorithm tries to classify the region in which an iterate lies. With this classification the algorithm decides if it tries to handle nonsmooth aspects of the problem explicitly or if it tries to avoid to handle them. Based on this general strategy, the status of the algorithm is split into three modes:

No convergence If an iterate $y^{(k)}$ has a stationarity measure $e(0)^{(k)}$ (see (4.18)) with

$$e\,(0)^{(k)} > \kappa_m \varepsilon_{\text{tol}}, \quad \kappa_m > 1, \tag{4.139}$$

the algorithm is in the *no-convergence* mode. In this mode the algorithm tries to avoid points y with $\ell_i(y) = 0$ for all nonsmooth constraints $c_i, i \in \mathcal{N}$. This is realized within modified backtracking line-search algorithms (see below). The rest of the algorithm stays the same.

Convergence in a smooth region If an iterate $y^{(k)}$ is reached with

$$\varepsilon_{\text{tol}} < e\,(0)^{(k)} < \kappa_m \varepsilon_{\text{tol}} \tag{4.140}$$

and

$$\left|\ell_i\left(y^{(k)}\right)\right| > \varepsilon_{\mathcal{N}} \quad \forall i \in \mathcal{N}, \quad \varepsilon_{\mathcal{N}} > 0, \tag{4.141}$$

the algorithm is in the *smooth-region-convergence* mode. In this case it is assumed that the current iterate is in a region of local convergence and that there is no point y near $y^{(k)}$ with $\ell_i(y) = 0$ for all nonsmooth constraints $c_i, i \in \mathcal{N}$. In the smooth-region-convergence mode only the backtracking line-search is modified as it is the case in the no-convergence mode.

Convergence in a nonsmooth region If an iterate $y^{(k)}$ is reached satisfying (4.140) and

$$\exists i \in \mathcal{N} \quad \text{with} \quad \left|\ell_i\left(y^{(k)}\right)\right| \leq \varepsilon_{\mathcal{N}}, \tag{4.142}$$

the algorithm is in the *nonsmooth-region-convergence* mode. Here, it is assumed that the current iterate is in a region of local convergence and that it is likely that the limit point

to which the algorithm may converge is a point $y^* \in \mathcal{K}$. The algorithmic strategy is then modified in a way such that the algorithm avoids to cross over points at which some problem data fails to be differentiable. Thus, if there is a point of non-differentiability $y \in \mathcal{K}$ in the search direction $\Delta y^{(k)}$, i.e.

$$\exists y \in \mathcal{K} \cap \bar{R}\left(y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{pri}}^{(k)}\right), \tag{4.143}$$

with

$$\bar{R}\left(y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{pri}}^{(k)}\right) := \left\{y_{\text{pri}}^{(k)} + \alpha \Delta y_{\text{pri}}^{(k)} \colon \alpha \in \left(0, \bar{\alpha}_{\text{pri}}^{(k)}\right]\right\}, \tag{4.144}$$

the algorithm "visits" y and checks a modified stationarity criterion for nonsmooth problems. If the modified stationarity test passes, the algorithm stops and returns y as a local solution of the nonsmooth problem. Otherwise, the algorithm proceeds with special problem-tailored generalized gradients for those constraints that fail to be differentiable at the iterate. These generalized gradients are used in the Jacobians and the Hessian that are part of the KKT matrix of the next iteration.

4.2.3 Modified Building Blocks

The Modified Line-Search

If the algorithm is in the *no-convergence* mode or in the *smooth-region-convergence* mode, the overall goal is to avoid points at which some constraints are not differentiable. For this, the backtracking line-search procedures that are invoked in lines 9 and 20 of Algorithm 6 are modified. The main design of these algorithms stays the same but line 5 of Algorithm 4 and lines 4 and 23 of Algorithm 3 are replaced by the subprocedure given in Algorithm 7.

Remark 1. *1. A "forth-tracking" is allowed in the modified line-search in order to avoid small step lengths that result from visited points in the line-search procedure at which some constraints fail to be differentiable. By this, the algorithm tries to avoid unnecessary small step lengths. The boolean flag* `allow-incr` *states whether a "forth-tracking" is allowed or not.*

2. The boolean flag `allow-incr` *in Algorithm 7 is set to* `false` *in the beginning of Algorithm 3 and Algorithm 4 and set to* `true` *after every backtracking step in the original algorithms (i.e. in lines 21 and 33 of Algorithm 3 and in line 9 of Algorithm 4).*

This yields the following lemma:

Lemma 4. *Consider Algorithm 3 and Algorithm 4 with the extension given in Algorithm 7.*

(i) The primal step length is never increased two times consecutively.

Algorithm 7: Modified Filter Line-Search for Nonsmooth Problems

Input : Primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$, primal step length $\alpha_{\text{pri},l}^{(k)}$, constants $\kappa_{b^-} = \kappa_b$ (see Algorithm 3 and Algorithm 4) and $\kappa_{b^+} > 1$ with $\kappa_{b^+}\kappa_{b^-} < 1$, $\bar{k}_m \in \mathbb{N}$.

1 Initialize `trial-found` = `false` and $k_m = 0$.
2 **while** not `trial-found` **do**
3 **if** $k_m > \bar{k}_m$ **then**
4 Stop the algorithm; no trial point $y^+ \notin \mathcal{K}$ could be found.
5 Compute a new trial point $y_{\text{pri}}^+ = y_{\text{pri}}^{(k)} + \alpha_{\text{pri},l}^{(k)} \Delta y_{\text{pri}}^{(k)}$.
6 **if** $\exists i \in \mathcal{N}$ with $\ell_i(y_{\text{pri}}^+) = 0$ **then**
7 **if** `allow-incr` = `true` **then**
8 Set $\alpha_{\text{pri},l}^{(k)} \leftarrow \kappa_{b^+} \alpha_{\text{pri},l}^{(k)}$.
9 Set `allow-incr` = `false`.
10 **else**
11 Set $\alpha_{\text{pri},l}^{(k)} \leftarrow \kappa_{b^-} \alpha_{\text{pri},l}^{(k)}$.
12 Set `allow-incr` = `true`.
13 Set $k_m \leftarrow k_m + 1$.
14 **else**
15 Set `trial-found` = `true`.
16 **return** trial point y_{pri}^+ and primal step length $\alpha_{\text{pri},l}^{(k)}$.

(ii) The step length computed by the modified line-search is not greater than $\bar{\alpha}_{\text{pri}}$. In other words, the line-search procedures extended by the subprocedure given in Algorithm 7 are still backtracking algorithms.

Proof. Both parts of the lemma are proved for the case that the subprocedure given in Algorithm 7 is called from line 4 in Algorithm 3. All other cases can be proved analogously.

(i) The `if`-part in line 8 of Algorithm 7 can only be reached if $k_m = 0$ and $l > 1$ (i.e. after a backtracking in the original line-search algorithm) or if $k_m > 0$ and the `else`-block in line 11 of Algorithm 7 is reached in the last sub-iteration $k_m - 1$. In the former case the `allow-incr` flag is set to `true` after a backtracking step in Algorithm 3 (see lines 21 and 33). For the latter case, the step length is decreased in the last sub-iteration $k_m - 1$. In both cases, the assertion holds.

(ii) First, let $l = 0$. Because the `allow-incr` flag is set to `false` in the beginning of Algorithm 3 (Remark 1), the first modification of the primal step length can only be a decrease in line 11 of Algorithm 7. The rest follows directly from part (i) and $\kappa_{b^+}\kappa_{b^-} < 1$.

□

Unfortunately, it is not possible to prove that Algorithm 7 terminates after a finite number of iterations without the safeguard in line 4. There might be pathological examples of piecewise

smooth constraints c with $\mathcal{T} \subset \mathcal{K}$, where $\mathcal{T}$ is an infinite set of trial points generated by Algorithm 7. In this case, the algorithm would never stop without the safeguard in line 4. On the one hand, it is not likely in practice that the algorithm is confronted with such constraints. On the other hand, it is possible that a primal search direction Δy_{pri} and a maximum primal step length $\bar{\alpha}_{\text{pri}}$ with small norm $\|\bar{\alpha}_{\text{pri}} \Delta y_{\text{pri}}\|$ lead to a sequence of trial points for which the smoothness test in line 6 always passes *numerically*. The latter might be the case because the criterion

$$\exists i \in \mathcal{N} \quad \text{with} \quad \ell_i \left(y_{\text{pri}}^+ \right) = 0 \tag{4.145}$$

is usually implemented as

$$\exists i \in \mathcal{N} \quad \text{with} \quad \left| \ell_i \left(y_{\text{pri}}^+ \right) \right| < \varepsilon_\ell \tag{4.146}$$

for a given tolerance $\varepsilon_\ell > 0$. In these cases, the safeguard in line 4 gets active and the subprocedure returns a point at which some constraints fail to be differentiable. If the complete line-search method finds an acceptable point anyway, the main algorithm proceeds with a problem-tailored nonsmooth stationarity test (see below).

The Step Length Truncation Rule

If the algorithm is in *nonsmooth-region-convergence* mode the main algorithmic strategy changes. Now, the goal is not to avoid to handle the nonsmoothness of the problem but to handle it explicitly. For this, a test that checks whether there is a nonsmooth point on the set $\bar{R} := \bar{R}(y^{(k)}, \Delta y^{(k)}, \bar{\alpha}_{\text{pri}}^{(k)})$ is included. Thus, it is distinguished whether

$$\bar{R} \cap \mathcal{K} = \emptyset \tag{4.147}$$

holds or not. If (4.147) holds, then all constraints are smooth on $\bar{R}$ and the maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$ can be used as the initial primal step length in the filter line-search procedure. Otherwise, the step length is truncated again, obtaining $\hat{\alpha}_{\text{pri}}^{(k)} < \bar{\alpha}_{\text{pri}}^{(k)}$ with

$$R \left(y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}, \hat{\alpha}_{\text{pri}}^{(k)} \right) \cap \mathcal{K} = \emptyset. \tag{4.148}$$

Here, R is defined by

$$R \left(y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}, \hat{\alpha}_{\text{pri}}^{(k)} \right) := \left\{ y_{\text{pri}}^{(k)} + \alpha \Delta y_{\text{pri}}^{(k)} \colon \alpha \in \left(0, \hat{\alpha}_{\text{pri}}^{(k)} \right) \right\}. \tag{4.149}$$

Notice that $\bar{R}$ and R only differ in the property if the interval α belongs to is closed at its right end or not. In order to determine $\hat{\alpha}_{\text{pri}}^{(k)}$, consider the one-dimensional optimization problems

$$\min \quad \alpha_i \tag{4.150a}$$

$$\text{s.t.} \quad \ell_i \left(y_{\text{pri}}^{(k)} + \alpha_i \Delta y_{\text{pri}}^{(k)} \right) = 0, \tag{4.150b}$$

$$\alpha_i \in \left[0, \bar{\alpha}_{\text{pri}}^{(k)} \right] \tag{4.150c}$$

for all $i \in \mathcal{N}$.

Lemma 5. *If* (4.150) *is infeasible, then* $c_i \in \mathcal{C}^2(\bar{R})$. *If* (4.150) *is feasible and has the global solution* $\hat{\alpha}_i$, *then* $c_i \in \mathcal{C}^2(\hat{R}_i)$ *with* $\hat{R}_i := \{y_{\text{pri}}^{(k)} + \alpha \Delta y_{\text{pri}}^{(k)} \colon \alpha \in (0, \hat{\alpha}_i)\}$.

Proof. (4.150) is infeasible if there is no α_i with $\ell_i \left(y_{\text{pri}}^{(k)} + \alpha_i \Delta y_{\text{pri}}^{(k)} \right) = 0$ and $\alpha_i \in [0, \bar{\alpha}_{\text{pri}}^{(k)}]$. Thus, the localization function has no root $y \in \bar{R}$, giving $c_i \in \mathcal{C}^2(\bar{R})$.

If (4.150) is feasible with optimal value $\hat{\alpha}_i$, there is no $\tilde{\alpha}_i \in [0, \hat{\alpha}_i)$ with $\ell_i \left(y_{\text{pri}}^{(k)} + \tilde{\alpha}_i \Delta y_{\text{pri}}^{(k)} \right) = 0$. Thus, $c_i \in \mathcal{C}^2(\hat{R}_i)$. □

Algorithm 8: Step Length Truncation Rule for Nonsmooth Problems

Input : Primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$.

1 **for** all $i \in \mathcal{N}$ **do**
2 Compute $\hat{\alpha}_i$ by solving (4.150) to global optimality.
3 If (4.150) is infeasible, set $\hat{\alpha}_i = \bar{\alpha}_{\text{pri}}^{(k)}$.
4 **return** $\hat{\alpha}_{\text{pri}}^{(k)} := \min_{i \in \mathcal{N}}(\hat{\alpha}_i)$.

Algorithm 8 states the complete step length truncation rule. By construction of the algorithm and Lemma 5, the following assertion holds.

Lemma 6. *Assume that the maximum step length* $\hat{\alpha}_{\text{pri}}^{(k)}$ *is computed by Algorithm 8 and that all optimization problems* (4.150) *are solved to global optimality in line 2 of Algorithm 8. Then all constraints* c *are smooth on* $\hat{R} := \{y_{\text{pri}}^{(k)} + \alpha \Delta y_{\text{pri}}^{(k)} \colon \alpha \in (0, \hat{\alpha}_{\text{pri}}^{(k)})\}$.

In the following, an iteration in which the maximum primal step length is shortened by Algorithm 8, i.e. an iteration for which $\hat{\alpha}_{\text{pri}}^{(k)} < \bar{\alpha}_{\text{pri}}^{(k)}$ holds, is called a *$\mathcal{K}$-iteration*.

Lemma 7. *Assume* $\Delta y_{\text{pri},i}^{(k)} \neq 0$ *for all* i *and* $\bar{\alpha}_{\text{pri}}^{(k)} > 0$. *Then Algorithm 8 always returns positive step lengths* $\hat{\alpha}_{\text{pri}}^{(k)} > 0$.

Proof. The following two cases are distinguished:

(i) Consider $y_{\text{pri}}^{(k)} \in \mathcal{K}$. Thus, at least one c_i is not differentiable at $y_{\text{pri}}^{(k)}$. Since all c_i are piecewise smooth and $\bar{\alpha}_{\text{pri}}^{(k)} \Delta y_{\text{pri}}^{(k)} \neq 0$, there exists an $\varepsilon > 0$ such that all c_i are smooth on $\{y_{\text{pri}}^{(k)} + \alpha \Delta y_{\text{pri}}^{(k)} \colon \alpha \in (0, \varepsilon)\}$. Hence, $0 < \hat{\alpha}_{\text{pri}}^{(k)}$.

(ii) Assume $y_{\text{pri}}^{(k)} \notin \mathcal{K}$. Since all c_i are piecewise smooth, there exists an $\varepsilon > 0$ such that all c_i are smooth on the open ε-ball $B_\varepsilon(y_{\text{pri}}^{(k)}) := \{y \in \mathbb{R}^{n_p} \colon \|y - y_{\text{pri}}^{(k)}\|_2 < \varepsilon\}$. Thus, $\hat{\alpha}_{\text{pri}}^{(k)} \geq \min(\varepsilon, \bar{\alpha}_{\text{pri}}^{(k)}) > 0$.

□

Some Remarks on Problem (4.150) To ensure that Lemma 6 is valid one has to solve the one-dimensional optimization problems (4.150) to global optimality. Unfortunately, this is practically impossible if there are no additional requirements on the functions $\ell_i, i \in \mathcal{N}$. For a lot of problems that are relevant in practice (cf. Example 5), the localization functions are linear. In this case, the problems (4.150) for $i \in \mathcal{N}$ can be rewritten as the single linear optimization problem

$$\begin{aligned} \max \quad & \alpha && \text{(4.151a)} \\ \text{s.t.} \quad & s_i \ell_i \left(y_{\text{pri}}^{(k)} + \alpha \Delta y_{\text{pri}}^{(k)} \right) \geq 0 \quad \forall i \in \mathcal{N}, && \text{(4.151b)} \\ & \alpha \in \left[0, \bar{\alpha}_{\text{pri}}^{(k)} \right], && \text{(4.151c)} \end{aligned}$$

with

$$s_i := \ell_i \left(y_{\text{pri}}^{(k)} \right). \tag{4.152}$$

Problem (4.151) contains one variable with simple bounds and $|\mathcal{N}|$ linear inequality constraints. It can be solved to global optimality by any LP solver or an LP-tailored version of the interior-point method described in Section 4.1. In contrast to (4.150), (4.151) has the additional advantage that it is always feasible. If none of the constraints (4.151b) is active in the solution (except for those for which $s_i = 0$), there is no point of non-differentiability for any c_i on the set $R(y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{pri}}^{(k)})$ and the optimal solution of (4.151) is $\bar{\alpha}_{\text{pri}}^{(k)}$.

Nonsmooth Stationarity Test

Interior-point algorithms try to compute a KKT point of the problem at hand. Hence, the proposed interior-point algorithm for nonsmooth constrained problems tries to compute a KKT point with respect to the KKT conditions (2.26) stated in Theorem 6. A termination criterion has to check whether a KKT point is approximately reached or not. To state such a termination criterion for piecewise smooth and locally Lipschitz-continuous but not necessarily smooth problems one especially has to consider the dual feasibility condition (2.26a) for nonsmooth problems.

In the following theorem, e_i denotes the i-th unit vector.

Theorem 11. *Consider problem* (4.138) *with locatable and separable nonsmoothness and let y be a primal-dual iterate. Furthermore, let $\mathcal{K}(y) \subset (\mathcal{E} \cup \mathcal{I})$ be the set of indices of constraints that fail to be differentiable at y and*

$$\begin{aligned} \delta = g - (u_l - u_u) - \sum_{i \in \mathcal{E} \setminus \mathcal{K}(y)} z_i \nabla c_i(x) - \sum_{i \in \mathcal{I} \setminus \mathcal{K}(y)} (v_{l_i} - v_{u_i}) \nabla c_i(x) \\ - \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i e_{i_a} - \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} (v_{l_i} - v_{u_i}) e_{i_a}. \end{aligned} \tag{4.153}$$

Define

$$\tilde{I}_i := z_i I_i, \qquad i \in \mathcal{E} \cap \mathcal{K}(y), \tag{4.154a}$$

$$\tilde{I}_i := (v_{l_i} - v_{u_i}) I_i, \quad i \in \mathcal{I} \cap \mathcal{K}(y), \tag{4.154b}$$

where the I_i are the subdifferentials at y of the corresponding univariate, piecewise smooth and locally Lipschitz-continuous functions θ_i (cf. Definition 16). In (4.154)*, the multiplication of a scalar α with an interval $I := [I^-, I^+]$ is defined as*

$$\alpha I := \begin{cases} [\alpha I^-, \alpha I^+], & \alpha \geq 0, \\ [\alpha I^+, \alpha I^-], & \alpha < 0. \end{cases} \tag{4.155}$$

Moreover, define

$$\mathcal{X} := \{k \in \{1, \ldots, n\} : \exists i \in \mathcal{K}(y) \ \text{ with } k = i_\nu\}. \tag{4.156}$$

Finally, set $\hat{I}_j := [\hat{I}_j^-, \hat{I}_j^+]$ for $j \in \mathcal{X}$ with

$$\hat{I}_j^- = \sum_{i \in \mathcal{K}(y): i_\nu = j} \tilde{I}_i^-, \qquad \hat{I}_j^+ = \sum_{i \in \mathcal{K}(y): i_\nu = j} \tilde{I}_i^+. \tag{4.157}$$

Then dual feasibility (2.26a) *holds at y if there exist scalars $\gamma_j \in \hat{I}_j$ such that*

$$\delta_j \mp \gamma_j = 0 \quad \forall j \in \mathcal{X} \tag{4.158}$$

and

$$\delta_j = 0 \quad \forall j \notin \mathcal{X}. \tag{4.159}$$

Proof. First, it is known from Lemma 2 that subdifferentials of univariate functions (like θ_i in Definition 16) are intervals. Then dual feasibility (2.26a) with $\lambda_f = 1$ is given by

$$0 \in \partial_x \mathcal{L}(x, z, v_l, v_u, u_l, u_u) \tag{4.160}$$

$$= \partial f(x) - \sum_{i \in \mathcal{E}} z_i \partial c_i(x) - \sum_{i \in \mathcal{I}} (v_{l_i} - v_{u_i}) \partial c_i(x) - (u_l - u_u) \tag{4.161}$$

$$\begin{aligned} &= \partial f(x) - (u_l - u_u) \\ &\quad - \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i \partial c_i(x) - \sum_{i \in \mathcal{E} \setminus \mathcal{K}(y)} z_i \partial c_i(x) \\ &\quad - \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} (v_{l_i} - v_{u_i}) \partial c_i(x) - \sum_{i \in \mathcal{I} \setminus \mathcal{K}(y)} (v_{l_i} - v_{u_i}) \partial c_i(x) \end{aligned} \tag{4.162}$$

$$\begin{aligned} &= g - (u_l - u_u) - \sum_{i \in \mathcal{E} \setminus \mathcal{K}(y)} z_i \nabla c_i(x) - \sum_{i \in \mathcal{I} \setminus \mathcal{K}(y)} (v_{l_i} - v_{u_i}) \nabla c_i(x) \\ &\quad - \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i \partial c_i(x) - \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} (v_{l_i} - v_{u_i}) \partial c_i(x) \end{aligned} \tag{4.163}$$

$$=: \tilde{\delta} - \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i \partial c_i(x) - \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} (v_{l_i} - v_{u_i}) \partial c_i(x). \tag{4.164}$$

In (4.162), the sums are split into sums of constraints that are differentiable at y and sums of constraints that fail to be differentiable at y. (4.163) follows because the generalized gradient of a differentiable function is the gradient. Using the fact that all constraints satisfy the separable nonsmoothness property one can rewrite (4.164) as

$$\tilde{\delta} - \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i \partial \left[x_{i_a} \pm \theta_i \left(x_{i_\nu} \right) \right] - \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} \left(v_{l_i} - v_{u_i} \right) \partial \left[x_{i_a} \pm \theta_i \left(x_{i_\nu} \right) \right] \tag{4.165}$$

$$= \tilde{\delta} - \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i e_{i_a} \mp \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i \partial \theta_i \left(x_{i_\nu} \right) \tag{4.166}$$

$$- \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} \left(v_{l_i} - v_{u_i} \right) e_{i_a} \mp \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} \left(v_{l_i} - v_{u_i} \right) \partial \theta_i \left(x_{i_\nu} \right)$$

$$= \delta \mp \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i \partial \theta_i \left(x_{i_\nu} \right) \mp \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} \left(v_{l_i} - v_{u_i} \right) \partial \theta_i \left(x_{i_\nu} \right) \tag{4.167}$$

$$= \delta \mp \sum_{i \in \mathcal{E} \cap \mathcal{K}(y)} z_i I_i e_{i_\nu} \mp \sum_{i \in \mathcal{I} \cap \mathcal{K}(y)} \left(v_{l_i} - v_{u_i} \right) I_i e_{i_\nu} \tag{4.168}$$

$$= \delta \mp \sum_{i \in \mathcal{K}(y)} \tilde{I}_i e_{i_\nu} \tag{4.169}$$

$$= \delta \mp \sum_{j \in \mathcal{X}} \hat{I}_j e_j. \tag{4.170}$$

The theorem follows directly from the last equation. □

In general, it is not easy to evaluate the dual feasibility condition (2.26a) in practice. However, Theorem 11 yields a useful termination criterion for problems with locatable and separable nonsmoothness. The theorem asserts that only subdifferentials, i.e. the intervals I_i, of the nonsmooth univariate functions θ_i are demanded of the user. The rest can be computed easily using the equations stated in the theorem.

Generalized Gradients

If the algorithm is at an iterate $y^{(k)} \in \mathcal{K}$ and the nonsmooth stationarity test established in the last section does not pass, the algorithm proceeds. Now, the main problem is that not all gradients of the problem data exist at $y^{(k)}$. Thus, one cannot build the KKT matrix

$$\begin{bmatrix} H^{(k)} + \Phi^{(k)} & \left(\nabla c_{\mathcal{E}}^{(k)} \right)^T & \left(\nabla c_{\mathcal{I}}^{(k)} \right)^T \\ \nabla c_{\mathcal{E}}^{(k)} & & \\ \nabla c_{\mathcal{I}}^{(k)} & & -\left(\Psi^{(k)} \right)^{-1} \end{bmatrix} \tag{4.171}$$

of the next iteration since it would contain gradients that are not defined. To handle this situation problem-tailored generalized gradients for those constraints that fail to be differentiable at $y^{(k)}$ are used. The concrete choice of generalized gradients is motivated in the following.

In most of the cases in which no gradient $\nabla c_i(x^{(k)})$ exists, the last iteration was a *$\mathcal{K}$-iteration* in which the step length was truncated in order to avoid a step over a point at which some constraints fail to be differentiable. Thus, without the step length truncation rule, the algorithm would have taken a larger step into the last search direction $\Delta y^{(k-1)}$. In the current iteration, the algorithm "remembers" this fact and uses the first-order and second-order information belonging to the (smooth) region the last search direction $\Delta y^{(k-1)}$ points to. The idea of the problem-tailored generalized gradients is motivated in the following example and is defined formally afterwards.

Example 6 (The Absolute Value Function)**.** *Consider the absolute value function $c(y) = |y|$. c is smooth on $\mathbb{R} \setminus \{0\}$ and locally Lipschitz-continuous. The localization function is $\ell(y) = y$. Moreover, consider an iterate $y^{(k-1)} < 0$, a search direction $\Delta y^{(k-1)} > |y^{(k-1)}|$ and a maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k-1)} = 1$. In this situation, the primal step length is truncated to $\hat{\alpha}_{\text{pri}}^{(k-1)} = -y^{(k-1)}/\Delta y^{(k-1)} < 1$ by Algorithm 8. Under the assumption that $y^{(k-1)} + \hat{\alpha}_{\text{pri}}^{(k-1)} \Delta y^{(k-1)}$ was accepted by the globalization strategy, it follows that $y^{(k)} = 0$ and $\ell(y^{(k)}) = 0$. Hence, $y^{(k)} \in \mathcal{K}$ so that $\nabla c(y^{(k)})$ does not exist. This situation is interpreted in a way that the algorithm "would like" to take a longer step in the direction $\Delta y^{(k-1)}$, i.e. towards the right half plane $\mathbb{R}_+$, which was prohibited by the step length truncation rule. Thus, the non-existing gradient $\nabla c(y^{(k)})$ is replaced by the derivative of a smooth continuation of c restricted to the positive half space $\mathbb{R}_+$. More precisely, the (later defined) generalized gradient $\tilde{\nabla} c(0, y^{(k-1)}) = 1$ is used. Figure 4.2 illustrates the situation.*

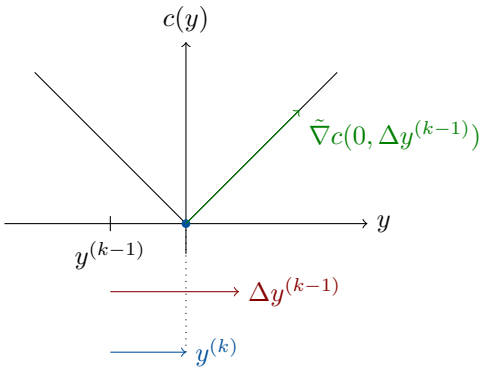

Figure 4.2: A generalized gradient for the absolute value function.

Next, a formal definition for what is described in the last example is given.

Definition 19. *Consider a constraint $\vartheta_i : \mathbb{R}^{n_\mathcal{N}} \to \mathbb{R}$ of the problem* (4.138)*, i.e.*

$$\vartheta_i(\hat{x}) = \vartheta_i(x, x_a) = x_{i_a} \pm \theta_i(x_{i_\nu}). \tag{4.172}$$

Furthermore, let $\hat{d} \in \mathbb{R}^{n_\mathcal{N}}$ and $\tilde{d}_{i_\nu} \in \mathbb{R}^{n_\mathcal{N}}$ the vector of zeros except for $\hat{d}_{i_\nu}/|\hat{d}_{i_\nu}|$ at the i_ν-th component. If $\hat{d}_{i_\nu} = 0$, $\tilde{d}_{i_\nu}$ is the vector of zeros. The generalized signed one-sided partial directional

derivative w.r.t. x_{i_ν} *of* ϑ_i *at* $\hat{x}$ *in the direction* $\hat{d}$ *is defined as*

$$\widetilde{\partial}_{i_\nu}\vartheta_i\left(\hat{x};\hat{d}\right) := \operatorname{sign}\left(\hat{d}_{i_\nu}\right)\lim_{t\downarrow 0}\frac{\vartheta_i\left(\hat{x}+t\tilde{d}_{i_\nu}\right)-\vartheta_i\left(\hat{x}\right)}{t}. \tag{4.173}$$

Here, $\operatorname{sign}(0)$ *is defined to be* 0.

Lemma 8. *The generalized signed one-sided partial directional derivative w.r.t.* x_{i_ν} *of the constraint* ϑ_i *of problem* (4.138) *is well-defined.*

Proof. The difference quotient in (4.173) is bounded by the local Lipschitz constant $L = L(\hat{x})$ and the existence of the limit follows directly from the piecewise smoothness of ϑ_i. □

Notice that the last definition is similar to the definition of standard one-sided directional derivatives. Both coincide for the one-dimensional case. Nevertheless, Definition 19 is more general in the sense that it allows the definition of the *generalized signed one-sided directional gradient*:

Definition 20. *Let* $\vartheta_i, \hat{x}$ *and* $\hat{d}$ *be as in Definition 19. The* generalized signed one-sided directional gradient *of* ϑ_i *at* $\hat{x}$ *in the direction* $\hat{d}$ *is defined as*

$$\widetilde{\nabla}\vartheta_i\left(\hat{x};\hat{d}\right) := \left(\widetilde{\partial}_j\vartheta_i\left(\hat{x};\hat{d}\right)\right)_{j=1}^{n_{\mathcal{N}}} \in \mathbb{R}^{n_{\mathcal{N}}}, \tag{4.174}$$

where $\widetilde{\partial}_j\vartheta_i(\hat{x};\hat{d}) := \partial_{\hat{x}_j}\vartheta_i(\hat{x})$, *i.e. the standard partial derivative of* ϑ_i, *for all* $j \neq i_\nu$ *and* $\widetilde{\partial}_{i_\nu}$ *is the generalized signed one-sided partial directional derivative w.r.t.* x_{i_ν}.

Example 7 (The Absolute Value Function Revisited)**.** *The last two definitions formalize exactly what is described in Example 6. Consider the constraint*

$$\vartheta_i\left(x_1,x_2\right) = x_2 + \theta_i\left(x_1\right) = x_2 + |x_1| = 0. \tag{4.175}$$

Here, $i_a = 2$ *and* $i_\nu = 1$. *Let* $\hat{x} = (0,0)^T$ *and* $\hat{d} = (1,\hat{d}_2)^T$ *for arbitrary* $\hat{d}_2$. *Thus,* $\tilde{d}_{i_\nu} = (1,0)^T$. *Definition 19 implies*

$$\widetilde{\partial}_{i_\nu}\vartheta_i\left(\hat{x};\hat{d}\right) = \operatorname{sign}\left(\hat{d}_{i_\nu}\right)\lim_{t\downarrow 0}\frac{\vartheta_i\left(\hat{x}+t\tilde{d}_{i_\nu}\right)-\vartheta_i\left(\hat{x}\right)}{t} \tag{4.176a}$$

$$= \lim_{t\downarrow 0}\frac{\theta_i\left(t\right)}{t} \tag{4.176b}$$

$$= 1. \tag{4.176c}$$

Analogously, one obtains $\widetilde{\partial}_{i_\nu}\vartheta_i\left(\hat{x};\hat{d}\right) = -1$ *for* $\hat{d} = (-1,\hat{d}_2)^T$. *Thus,* $\widetilde{\nabla}\vartheta_i\left(\hat{x};\hat{d}\right) = (1,1)^T$ *holds for* $\hat{d} = (1,\hat{d}_2)^T$ *and* $\widetilde{\nabla}\vartheta_i\left(\hat{x};\hat{d}\right) = (-1,1)^T$ *holds for* $\hat{d} = (-1,\hat{d}_2)^T$.

The following theorem shows that the generalized signed one-sided directional gradient belongs to Clarke's generalized gradient.

Theorem 12. *Consider a constraint* $\vartheta_i : \mathbb{R}^{n_\mathcal{N}} \to \mathbb{R}$ *of the problem* (4.138). *Furthermore, let* $\hat{x} \in \mathbb{R}^{n_\mathcal{N}}$ *and* $\hat{d} \in \mathbb{R}^{n_\mathcal{N}}$ *with* $\hat{d}_{i_\nu} \neq 0$. *Then*

$$\widetilde{\nabla}\vartheta_i\left(\hat{x};\hat{d}\right) \in \partial\vartheta_i\left(\hat{x}\right) \tag{4.177}$$

holds.

Proof. Using Lemma 1, it is to show that there exists a sequence $(\hat{x}_k)$ with $\hat{x}_k \to \hat{x}$, $\hat{x}_k \in \mathbb{R}^{n_\mathcal{N}} \setminus \mathcal{K}$ and $\widetilde{\nabla}\vartheta_i(\hat{x};\hat{d}) = \lim_{\hat{x}_k \to \hat{x}} \nabla\vartheta_i(\hat{x}_k)$. Furthermore, it is enough to show that the theorem holds component-wise, i.e. it remains to prove

$$\widetilde{\partial}_j\vartheta_i\left(\hat{x};\hat{d}\right) = \lim_{\hat{x}_k \to \hat{x}} \partial_{\hat{x}_j}\vartheta_i\left(\hat{x}_k\right). \tag{4.178}$$

Let $\hat{x}_k := \hat{x} + \alpha_k\hat{d}$ with positive numbers $\alpha_k \to 0$. For all variables $\hat{x}_j$ with $j \neq i_\nu$, $\widetilde{\partial}_j\vartheta_i\left(\hat{x};\hat{d}\right)$ is the standard gradient and thus (4.178) holds because of the piecewise smoothness of ϑ_i. By this reason, it is enough to show the convergence for the i_ν-th component. Starting with the right-hand side, one has

$$\begin{aligned}
\lim_{\hat{x}_k \to \hat{x}} \partial_{\hat{x}_{i_\nu}}\vartheta_i\left(\hat{x}_k\right) &= \lim_{\hat{x}_k \to \hat{x}} \lim_{t \to 0} \frac{\vartheta_i\left(\hat{x}_k + te_{i_\nu}\right) - \vartheta_i\left(\hat{x}_k\right)}{t} && \text{(4.179a)}\\
&= \lim_{\hat{x}_k \to \hat{x}} \operatorname{sign}\left(\hat{d}_{i_\nu}\right) \lim_{s \downarrow 0} \frac{\vartheta_i\left(\hat{x}_k + s\tilde{d}_{i_\nu}\right) - \vartheta_i\left(\hat{x}_k\right)}{s} && \text{(4.179b)}\\
&= \lim_{\hat{x}_k \to \hat{x}} \widetilde{\partial}_{i_\nu}\vartheta_i\left(\hat{x}_k;\hat{d}\right) && \text{(4.179c)}\\
&= \widetilde{\partial}_{i_\nu}\vartheta_i\left(\hat{x};\hat{d}\right). && \text{(4.179d)}
\end{aligned}$$

The first equality holds due to the definition of the partial derivative that exists for $\hat{x}_k$ since $\hat{x}_k \in \mathbb{R}^{n_\mathcal{N}} \setminus \mathcal{K}$ for sufficiently small α_k. Since ϑ_i is differentiable at all $\hat{x}_k$ one can switch to the directional derivative in the direction $\tilde{d}_{i_\nu}$ in the second equation. A possible alternation in the sign is addressed by the factor $\operatorname{sign}(\hat{d}_{i_\nu})$ that is independent of the limit. The penultimate equation is exactly the definition of $\widetilde{\partial}_{i_\nu}\vartheta_i(\hat{x};\hat{d})$ and the last equation holds because ϑ_i is piecewise smooth, i.e. it is smooth on the interior of the set $\left\{\hat{x} + \alpha\hat{d} \colon \alpha > 0\right\}$ for sufficiently small $\|\alpha\hat{d}\|$. □

The assumption "$\hat{d}_{i_\nu} \neq 0$" of the last theorem is of special importance in the algorithm. The algorithm checks, if a starting point given by the user is a point at which some constraints fail to be differentiable. If there are constraints that are non-differentiable at this point, the starting point is perturbed such that $\ell_i(x^{(0)}) \neq 0$ for all $i \in \mathcal{N}$. Thereby, it is guaranteed that $\Delta x_{i_\nu}^{(k-1)} \neq 0$ holds if an iterate $x^{(k)}$ is reached with $\ell_i(x^{(k)}) = 0$. Since $\hat{d}$ is always chosen to be the last search direction in the algorithm, the assumption is not restricting in practice.

In summary, the proposed method chooses the ordinary gradients if they exist and the generalized signed one-sided directional gradients at those points where some constraints are not

differentiable. In analogy, second-order information is constructed by applying the same ideas to the (generalized) gradients. This leads to a well-defined system matrix (4.171) for the Newton step.

4.2.4 An Extended Interior-Point Method for Nonsmooth Nonlinear Optimization

In the last sections the problem class under consideration is introduced and the modified building blocks of an interior-point algorithm that is able to solve this problem class are described.

Algorithm 9 states the complete method. It is strongly based on Algorithm 6. The extensions and modifications are emphasized with blue colored sans-serif fonts. For better reading, some textual descriptions in the algorithm are abbreviated. In these cases, the full text can be found in Algorithm 6.

Algorithm 9: Extended Interior-Point Method for Nonsmooth Problems

Input : User provided starting point $\bar{x}$ for the original nonsmooth problem, initial barrier parameter $\mu^{(0)}$, vector of algorithmic constants κ.

Set iteration counter $k = 0$, initialize filter $\mathcal{F}_{\text{NLP}}$ using (4.76) and (4.77), set μ-mode = `adaptive`.
Set conv-mode = `no-convergence`.
If required, perturb the starting point $\bar{x}$ such that $\ell_i(\bar{x}) \neq 0$ holds for all $i \in \mathcal{N}$.
Call Algorithm 5 with $\bar{x}$ and $\mu^{(0)}$ to obtain initial point $y^{(0)}$.
If required, perturb the starting point $y^{(0)}$ such that $\ell_i(y^{(0)}) \neq 0$ holds for all $i \in \mathcal{N}$.
while nonsmooth termination criterion does not hold (see Theorem 11) **do**
  Set conv-mode according to (4.139)–(4.142).
  Increase iteration counter $k \leftarrow k + 1$.
  if μ-mode = `adaptive` **then** `/* adaptive mode */`
    Compute $\mu^{(k)}$ by any rule (cf. Section 4.1.4) and update τ using (4.16).
    Compute search direction $\Delta y^{(k)}$ as described in Section 4.1.1.
    (Use the generalized one-sided directional gradients with $d = \Delta y^{(k-1)}$ to set up the KKT matrix if $y^{(k)} \in \mathcal{K}$.)
    Compute maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
    if conv-mode = `nonsmooth-region-convergence` **then**
      Call Algorithm 8 with $y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}$ and $\bar{\alpha}_{\text{pri}}^{(k)}$ to obtain $\hat{\alpha}_{\text{pri}}^{(k)}$.
    else
      Set $\hat{\alpha}_{\text{pri}}^{(k)} = \bar{\alpha}_{\text{pri}}^{(k)}$.
    Call Algorithm 4 (modified by Algorithm 7 if conv-mode $\neq$ `nonsmooth-region-convergence`) with $y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\hat{\alpha}_{\text{pri}}^{(k)}$ to obtain $\alpha_{\text{pri}}^{(k)}$.
    if Algorithm 4 succeeds **then** `/* stay adaptive */`
      Compute new primal and dual iterates $y_{\text{pri}}^{(k+1)}, y_{\text{dual}}^{(k+1)}$ using (4.17) with primal step length $\alpha_{\text{pri}}^{(k)}$ and dual step length $\bar{\alpha}_{\text{dual}}^{(k)}$.
    else `/* switch to monotone mode */`
      Set μ-mode $\leftarrow$ `monotone`, reset barrier parameter $\mu^{(k+1)}$ using (4.95), update τ using (4.16), reset barrier problem filter $\mathcal{F}_\mu$ using (4.76) and (4.77).
  else `/* monotone mode */`
    if nonsmooth barrier problem termination criterion (4.22) holds **then**
      Compute $\mu^{(k)}$ by any rule (cf. Section 4.1.4) and update τ using (4.16).
      Reset barrier problem filter $\mathcal{F}_\mu$ using (4.76) and (4.77)
    Compute search direction $\Delta y^{(k)}$ as described in Section 4.1.1.
    (Use the generalized one-sided directional gradients with $d = \Delta y^{(k-1)}$ to set up the KKT matrix if $y^{(k)} \in \mathcal{K}$.)
    Compute maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
    if conv-mode = `nonsmooth-region-convergence` **then**
      Call Algorithm 8 with $y_{\text{pri}}^{(k)}, \Delta y_{\text{pri}}^{(k)}$ and $\bar{\alpha}_{\text{pri}}^{(k)}$ to obtain $\hat{\alpha}_{\text{pri}}^{(k)}$.
    else
      Set $\hat{\alpha}_{\text{pri}}^{(k)} = \bar{\alpha}_{\text{pri}}^{(k)}$.
    Call Algorithm 3 (modified by Algorithm 7 if conv-mode $\neq$ `nonsmooth-region-convergence`) with $y_{\text{pri}}^{(k)}$, $\Delta y^{(k)}$ and maximum primal step length $\hat{\alpha}_{\text{pri}}^{(k)}$ to obtain $\alpha_{\text{pri}}^{(k)}$.
    if Algorithm 3 succeeds **then**
      Compute new primal and dual iterates $y_{\text{pri}}^{(k+1)}, y_{\text{dual}}^{(k+1)}$ using (4.17) with primal step length $\alpha_{\text{pri}}^{(k)}$ and dual step length $\bar{\alpha}_{\text{dual}}^{(k)}$.
      if $y_{\text{pri}}^{(k+1)}$ is acceptable to the filter $\mathcal{F}_{\text{NLP}}$ **then**
        Set μ-mode $\leftarrow$ `adaptive`.
    else
      Go to feasibility restoration phase.
return optimal solution $y^{(k)}$.

4.3 An Interior-Point Method for MPCCs

This section describes the extensions of the interior-point method for nonlinear optimization problems in order to solve MPCC-type problems. The essential idea of these extensions is that the original MPCC is replaced by a sequence of regularized nonlinear optimization problems. The proposed method solves a sequence of regularized problems and ensures that the regularization parameter is driven to zero in the limit. In the following, the focus will be on regularization by *relaxation* and *penalization*. The basic concepts and results of this field are given in Section 2.2. The following reviews the regularized formulations and discusses when and how to update the regularization parameter.

First attempts of applying NLP-based techniques to MPCCs are made by Fletcher and Leyffer in [42], where they use the active-set SQP method FilterSQP to solve the MPCCs. In [42], Fletcher and Leyffer also compare SQP-type codes with interior-point methods (LOQO and KNITRO) applied to MPCCs. They come to the result that active-set SQP methods perform better than interior-point methods. MPCC-tailored interior-point algorithms based on relaxation are published by Liu and Sun in [75] and by Raghunathan and Biegler in [99]. In [11], Benson et al. use the interior-point code LOQO to solve MPCCs. In [74], Leyffer et al. develop a penalty interior-point framework for solving MPCCs and give two concrete algorithmic instantiations of their framework. The following description is mainly based on the latter paper. In practice, most of the used software that is able to solve MPCC problems is NLP-based and implements several regularization schemes. For example, today's versions of LOQO [11] and KNITRO [20] are able to handle complementarity constrained problems. Notice that both codes are commercial and not open source. In [99] Raghunathan and Biegler extend the open source code Ipopt such that it can handle MPCCs. Unfortunately, this version is not under active development anymore. Fletcher and Leyffer extend their SQP code FilterSQP in order to solve MPCCs [42]. Again (and to the best of the author's knowledge), the extended method filtermpec is not under active development anymore.

4.3.1 MPCC Regularization by Relaxation

As described in Section 2.2, relaxation schemes replace the original MPCC by a sequence of relaxed problems

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{4.180a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{4.180b}$$

$$c_{\mathcal{I}}(x) \geq 0, \tag{4.180c}$$

$$\phi(x) \geq 0, \psi(x) \geq 0, \tag{4.180d}$$

$$\phi_i(x)\,\psi_i(x) \leq \xi, \quad i = 1, \ldots, p. \tag{4.180e}$$

Compared to the standard form (4.1) of NLPs, the additional problem data of (4.180) are

1. additional inequality constraints for the non-negativity of the complementarity pairings ϕ_i, ψ_i and
2. additional inequality constraints $\phi_i(x)\psi_i(x) \leq \xi$ for the relaxation of the complementarity conditions.

As a consequence, the user has to provide function evaluations as well as first- and, possibly, second-order evaluations for the complementarity pairings ϕ_i, ψ_i. Whether there is a second-order evaluation required or not depends on the nonlinearity of ϕ and ψ.

When applying the extended relaxation scheme of DeMiguel et al. (see [31]), the only difference to (4.180) is the right-hand side of the inequality constraints for the single complementarity constraints. Now, $\phi(x) \geq 0$ and $\psi(x) \geq 0$ are replaced by $\phi(x) \geq -\theta$ and $\psi(x) \geq -\theta$. This only affects the KKT data but not the KKT structure.

The developed algorithm solves MPCCs as a regularized sequence of NLPs. For this, it takes the complete MPCC problem data, constructs the corresponding data for the regularized NLPs and additionally handles the modification of the regularization parameters ξ and θ.

4.3.2 MPCC Regularization by Penalization

Penalization schemes completely remove the complementarity conditions $\phi_i(x)\psi_i(x) = 0$ from the set of constraints and penalize their violation by an additional penalty term in the objective function (see Section 2.2):

$$\min_{x \in \mathbb{R}^n} \quad f(x) + \frac{1}{\xi}\Pi(\phi(x), \psi(x)) \tag{4.181a}$$

$$\text{s.t.} \quad c_{\mathcal{E}}(x) = 0, \tag{4.181b}$$

$$c_{\mathcal{I}}(x) \geq 0, \tag{4.181c}$$

$$\phi(x) \geq 0, \psi(x) \geq 0. \tag{4.181d}$$

As it is the case for relaxation schemes, the additional MPCC problem data are the additional inequality constraints $\phi(x) \geq 0$ and $\psi(x) \geq 0$. The remaining differences only appear in the extended objective function

$$f(x) + \frac{1}{\xi}\Pi(\phi(x), \psi(x)), \tag{4.182}$$

leading to the modified gradient

$$\nabla f(x) + \frac{1}{\xi}\nabla\Pi(\phi(x), \psi(x)) \tag{4.183}$$

and to the modified Hessian

$$\nabla^2_{xx} f(x) + \frac{1}{\xi}\nabla^2_{xx}\Pi(\phi(x), \psi(x)) \tag{4.184}$$

of the objective function. Depending on the nonlinearity of Π, ϕ and ψ this extension changes the sparsity structure of the Hessian of the Lagrangian in the upper left block of the KKT matrix.

As for the relaxation scheme, the algorithm requires the complete MPCC data and constructs the regularized NLPs (4.181). In addition, the modification of ξ is handled. This is the topic of the next section.

4.3.3 Updating the Regularization Parameter

Beside the structural modifications according to the additional data of the original MPCC, the question arises when and how to update the regularization parameter ξ.

Interior-point methods compute a KKT point of the given problem by decreasing the primal and dual infeasibility as well as the values of the KKT complementarity conditions (2.7e). During the algorithm the latter are relaxed yielding the so-called μ-perturbed complementarity conditions (4.8). In the developed algorithmic framework, the updates of the regularization parameter ξ are directly coupled with the updates of the barrier parameter μ. This coupling has to satisfy that ξ goes to zero if μ does. Whenever the interior-point method tries to decrease the barrier parameter, it first checks if a certain *MPCC-complementarity measure* has a lower value than a given tolerance. The measure depends on the current values of the complementarity pairings $\phi_i(x), \psi_i(x)$ and the required tolerance may depend on the current barrier parameter.

There is a wide range of possible MPCC-complementarity measures. The only requirements that should be satisfied by MPCC-complementarity measures m are that they should be non-negative and that they tend to zero if the complementarity conditions $\phi_i(x)\psi_i(x)$ do. In analogy, the tolerance should tend to zero if the barrier parameter tends to zero.

For instance, Leyffer et al. [74] use the measure

$$m = \max_{i=1,\ldots,p} \{\min(\phi_i(x), \psi_i(x))\} \tag{4.185}$$

and the tolerance

$$\varepsilon_m = \mu^{0.4}. \tag{4.186}$$

Another possible MPCC-complementarity measure is the MPCC penalty objective function term

$$m = \sum_{i=1}^{p} \phi_i(x)\, \psi_i(x). \tag{4.187}$$

Independent of the chosen regularization scheme, the regularization parameter is updated whenever the barrier parameter should be updated and

$$m < \varepsilon_m \tag{4.188}$$

does not hold. In these cases, the barrier parameter stays the same and the regularization parameter is updated such that $\xi^{(k+1)} < \xi^{(k)}$ holds. A standard update scheme is

$$\xi^{(k+1)} = \kappa_\xi \xi^{(k)}, \quad \kappa_\xi \in (0,1)\,. \tag{4.189}$$

If the barrier parameter should be updated and (4.188) is satisfied, the regularization parameter ξ stays the same and the barrier parameter is updated.

This leads to an interior-point method for solving MPCCs that is given in Algorithm 10. This method is strongly oriented towards the method proposed in [74]. However, Algorithm 10 is more generic since the algorithm is independent of the concrete regularization scheme.

Like the extended method for solving nonsmooth problems (Algorithm 9), Algorithm 10 is based on Algorithm 6. The modifications and extensions are emphasized with blue sans-serif fonts.

Algorithm 10: Extended Interior-Point Method for MPCCs

Input : User provided starting point $\bar{x}$ for the MPCC, initial barrier parameter $\mu^{(0)}$, initial regularization parameter $\xi^{(0)}$, vector of algorithmic constants κ and a tolerance $\varepsilon_{m_{\text{tol}}} > 0$ for the MPCC-complementarity measure

Set iteration counter $k = 0$, initialize filter $\mathcal{F}_{\text{NLP}}$ using (4.76) and (4.77), set μ-mode = `adaptive`.
Call Algorithm 5 with $\bar{x}$ and $\mu^{(0)}$ to obtain initial point $y^{(0)}$.
while NLP termination criterion (4.23) does not hold or $m > \varepsilon_{m_{\text{tol}}}$ **do**
    Increase iteration counter $k \leftarrow k + 1$.
    if μ-mode = `adaptive` **then** `/* adaptive mode */`
        if (4.188) holds **then**
            Compute $\mu^{(k)}$ by any rule (cf. Section 4.1.4) and update τ using (4.16).
            Set $\xi^{(k)} = \xi^{(k-1)}$.
        else `/* update regularization parameter */`
            Update the regularization parameter using (4.189) to obtain $\xi^{(k)}$.
            Re-evaluate the problem data.
        Compute search direction $\Delta y^{(k)}$ as described in Section 4.1.1.
        Compute maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
        Call Algorithm 4 with primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$ to obtain $\alpha_{\text{pri}}^{(k)}$.
        if Algorithm 4 succeeds **then** `/* stay adaptive */`
            Compute new primal and dual iterates $y_{\text{pri}}^{(k+1)}, y_{\text{dual}}^{(k+1)}$ using (4.17) with primal step length $\alpha_{\text{pri}}^{(k)}$ and dual step length $\bar{\alpha}_{\text{dual}}^{(k)}$.
        else `/* switch to monotone mode */`
            Set μ-mode $\leftarrow$ `monotone`, reset barrier parameter $\mu^{(k+1)}$ using (4.95), update τ using (4.16), reset barrier problem filter $\mathcal{F}_\mu$ using (4.76) and (4.77).
    else `/* monotone mode */`
        if barrier problem termination criterion (4.22) holds **then**
            if (4.188) holds **then**
                Compute $\mu^{(k)}$ by any rule (cf. Section 4.1.4) and update τ (4.16).
                Set $\xi^{(k)} = \xi^{(k-1)}$.
                Reset barrier problem filter $\mathcal{F}_\mu$ using (4.76) and (4.77).
            else `/* update regularization parameter */`
                Update the regularization parameter using (4.189) to obtain $\xi^{(k)}$.
                Re-evaluate the problem data.
        Compute search direction $\Delta y^{(k)}$ as described in Section 4.1.1.
        Compute maximum primal and dual step lengths $\bar{\alpha}_{\text{pri}}^{(k)}, \bar{\alpha}_{\text{dual}}^{(k)}$ using (4.15).
        Call Algorithm 3 with primal iterate $y_{\text{pri}}^{(k)}$, primal search direction $\Delta y_{\text{pri}}^{(k)}$ and maximum primal step length $\bar{\alpha}_{\text{pri}}^{(k)}$ to obtain $\alpha_{\text{pri}}^{(k)}$.
        if Algorithm 3 succeeds **then**
            Compute new primal and dual iterates $y_{\text{pri}}^{(k+1)}, y_{\text{dual}}^{(k+1)}$ using (4.17) with primal step length $\alpha_{\text{pri}}^{(k)}$ and dual step length $\bar{\alpha}_{\text{dual}}^{(k)}$.
            if $y_{\text{pri}}^{(k+1)}$ is acceptable to the filter $\mathcal{F}_{\text{NLP}}$ **then**
                Set μ-mode $\leftarrow$ `adaptive`.
        else
            Go to feasibility restoration phase.
return optimal solution $y^{(k)}$.

4.4 An Interior-Point Method for Nonsmooth and Complementarity Constrained Nonlinear Optimization

The two preceding sections deal with extensions and modifications of the basic interior-point method (Algorithm 6). These extensions and modifications enable the method to handle

1. optimization problems with locatable and separable nonsmoothness (Algorithm 9) and
2. complementarity constrained problems (Algorithm 10).

By taking a closer look to both methods one can see that the corresponding modifications are carefully designed in an *orthogonal* way. That means that there are no building blocks of the basic algorithm that have to be modified in different ways in Algorithm 9 and Algorithm 10.

This fact allows to combine both extensions in order to obtain an algorithm that is able to handle both nonsmooth and complementarity constrained problems. The modification of the basic algorithm using both modifications is straightforward so that the resulting algorithm is not explicitly stated here.

Chapter 5

Software Design

In this chapter some aspects of software design are discussed that strongly guided the implementation of the interior-point methods described in Chapter 4.

First, a brief overview of some general design concepts is given in Section 5.1. In the subsequent section the high-level architecture of the implementation of the interior-point methods is discussed. Finally, Section 5.3 briefly lists the used external libraries and presents techniques for increasing the software quality of the developed framework.

5.1 General Concepts of Software Design

In Chapter 4 the property of interior-point methods that they allow to solve different kinds of classes of mathematical optimization problems without significantly changing the algorithmic framework is pinpointed frequently. Moreover, extended or modified versions of a basic interior-point method are developed that are able to solve the challenging problem classes of complementarity constrained and nonsmooth problems.

In computational mathematics, the situation is often as follows. There exist a lot of software packages for a lot of different classes of optimization problems. Furthermore, these software packages are quite monolithic. That means, they are especially tailored for the concrete problem class they are developed for and thus, behave very well in terms of performance and robustness. But they often reinvent the wheel and implement things that were implemented many times before. The goal of the software framework that is developed for this thesis is to stop reinventing the wheel. It provides an almost complete algorithmic skeleton of interior-point methods and an adequate number of algorithmic building blocks that can be plugged into the skeleton.

Such a framework has two main advantages. First of all, applied computational mathematicians in optimization are often involved in industrial projects. In a lot of these projects there is a phase

of *proof of concept*. In this phase, the mathematicians have to prove that their methodology can be applied to the industrial problem at hand. To give this proof it is often the case that several approaches of *modeling* and *solving* the problem are tested. The modeling part in this phase greatly benefits from algebraic modeling languages for mathematical optimization like GAMS [103] or AMPL [44]. The effort spent for trying different solution strategies greatly depends on the knowledge of the mathematician about the used solvers. Obviously, this part would benefit from a generic software framework that can be instantiated in different versions leading to different algorithms with the same mathematical superstructure.

Secondly, the phase after giving the proof of concept comes into play. One is often confronted with the situation that it is known which basic algorithm is preferable; e.g. if an SQP method or an interior-point method should be used. Nevertheless, it is often desirable to modify some (few) parts of the algorithm in order to increase its robustness or performance.

To make the last point clear, have a look at a prominent example: A lot of optimization problems have a special KKT structure. If this structure is known it can be exploited in the interior-point building block that solves the KKT system in every iteration. For instance, this is especially the case for time-dependent control problems or for problems from stochastic programming. For the latter, some numerical results of a very problem-specific solution approach for portfolio optimization are discussed in Chapter 6. The approach exploits the special structure of the problem yielding a proper parallelization scheme and a highly efficient parallel solution algorithm for the KKT system.

In order to be able to specialize the interior-point method such that it can incorporate highly problem-specific solution approaches, the software framework has to satisfy some important concepts from the field of software engineering. Here, the most important concepts are the ones of *orthogonality*, *cohesion* and *coupling*.

Orthogonality Every mathematician knows the concept of orthogonal vectors from geometry: Two vectors are orthogonal in the Euclidean plane if they meet at right angles. The latter is formally defined by a zero scalar product of the two vectors. Obviously, both vectors can be changed *independently* in length and sign without destroying the property of orthogonality. Translated to software design, the vectors can be interpreted as modules of the software (cf. Figure 5.1). Thus, an orthogonal software design satisfies the property that their modules are as independent as possible. This means, that the functionality or implementation details of a module can change without harming the functionality of other modules. See [60] for a more detailed description of the concept of orthogonal software design.

For the development of an interior-point framework orthogonality means that certain algorithmic building blocks can be exchanged or modified in their implementation details without affecting the implementation and the design of other algorithmic building blocks. In terms of the

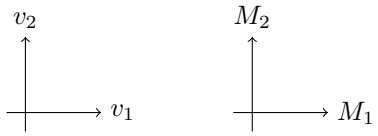

Figure 5.1: Orthogonal vectors v_1 and v_2 and software modules M_1 and M_2.

example given above, an orthogonal software design for interior-point methods allows to replace a general-purpose KKT system solution algorithm (that might be the default) by a problem-specific algorithm without affecting the implementation of other algorithmic building blocks (e.g. algorithms implementing the barrier parameter update rules). According to this, the right part of Figure 5.1 may be concretized like it is shown in Figure 5.2

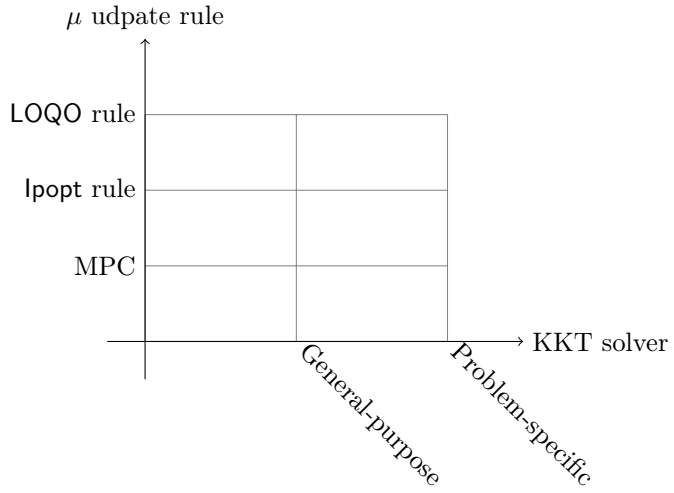

Figure 5.2: Orthogonal algorithmic building blocks; barrier parameter update rules and KKT system solution algorithms.

Cohesion The concept of orthogonality is strongly connected with the concept of *cohesion*. According to [144], cohesion is the "degree to which the elements of a module belong together". Obviously, it is desirable to implement software modules with high cohesion. The advantages of software modules with high cohesion are easy to see. The readability of the code as well as its maintainability increases with the cohesion of its modules. As a consequence, a highly cohesive interior-point framework simplifies the task of exchanging several algorithmic building blocks of the method. For instance, a new user does not have to know anything about the KKT solution algorithm if he only wants to implement a new update rule for the barrier parameter.

Coupling The last important concept that should be mentioned here is the one of *coupling*. Coupling describes the degree to which a software module depends on any other module (see [144]). To achieve the goal of orthogonal code, one has to realize software modules that are as loosely coupled as possible. Obviously, loosely coupled modules often correlate with high cohesive modules and are a precondition for orthogonal software.

C++ Generic Programming Techniques

The implementation of the interior-point method is done in C++. C++ (especially in its current version C++11) offers a wide range of techniques that allow to implement an efficient and generic interior-point framework satisfying the general concepts of software design discussed above.

The interior-point framework implemented for this thesis will be referred to as Clean::IPM in the following. It is part of the software framework Clean, which is an acronym for *A C++ Library of Efficient Algorithms in Numerics*. Clean is a generic library that is developed in the working group *Algorithmic Optimization* of Marc Steinbach at the Leibniz Universität Hannover. It is intended to become public domain when it is considered to be sufficiently mature. The application of most of the discussed software concepts in the context of numerics goes back to Marc Steinbach and Clean.

To realize the concepts of software design discussed above in Clean::IPM, techniques of generic programming with C++ templates are extensively used to implement specializations of the design patterns of *policy-based class design* and *traits*. Policies and policy-based class design are introduced by Alexandrescu in [3]. Already one of the first sentences in his book makes clear, what policy-based class design is about:

> "In brief, policy-based class design fosters assembling a class with complex behavior out of many little classes (called *policies*), each of which takes care of only one behavioral or structural aspect."

This quotation directly suggests that policy-based class design may be a good candidate for realizing orthogonal, highly cohesive and loosely coupled code. The paradigm of policy-based class design can be interpreted as compile-time variant of the *strategy pattern* (see [46]). The main idea is to design classes that take several template parameters as input. These template parameters are instantiated in dependence on types given by the user. The latter types are the policies that specify a well-defined "behavioral or structural aspect" of the class.

The last point that should be mentioned concerning policies addresses the combinatorial explosion of design choices. By taking a closer look at Figure 5.2 one sees that a so-called *do-it-all interface* (see [3]) has to implement $6 = 2 \cdot 3$ variants of the interior-point method by combining only two instantiations of KKT system solution algorithms and three rules for the barrier parame-

ter updates. In policy-based class design, one follows Alexandrescu's phrase "Never use brute force in fighting an exponential". Using policy classes, the number of aspects in this example is two and one has to implement only $2 + 3 = 5$ single aspects. Of course, five is not that much less than six, but it is obvious that the policy approach yields a *linear* increase with respect to the number of aspects or modules in contrast to an *exponential* increase for the do-it-all interface.

The second C++ generic programming technique that is frequently used in Clean::IPM are *traits*. Traits are one of the key features of the *C++ standard template library* [122]. They allow to make compile-time decisions depending only on types (e.g. policy classes) instead of making a run-time decision depending on values (see [2]). For instance, traits are used in order to automatically choose (at compile-time) between implementation details that depend on some user-given choices, e.g. on the class of the optimization problem.

A more detailed description of policies and traits is out of the scope of this thesis. It is referred to the references within the last paragraphs or the Boost pages on generic programming [13] and the references therein. The latter also introduces some other techniques like *tag dispatching* or *adaptors* that are used in the implementation of Clean::IPM, too.

5.2 The Software Architecture of Clean::IPM

This section describes the high-level software architecture of Clean::IPM. The main components can be distinguished into the following groups:

Main Algorithm The main interior-point algorithm encapsulates the algorithmic logic that is stated in the basic framework (see Algorithm 1).

Sub-Algorithms The sub-algorithms encapsulate all free choices that are left in Algorithm 1. Thus, for every task in interior-point methods (like updating the barrier parameter, solving the KKT system, etc.) there has to be a sub-algorithm that is responsible for this task. As usual, there are a lot of different possibilities how a certain task can be done (see e.g. the various rules for updating the barrier parameter in Section 4.1.4). Here, the user has the free choice. Either he uses an existing sub-algorithm or he implements a new one. If a new sub-algorithm is implemented, the only requirement that the new implementation has to satisfy is the fixed interface between the main algorithm, the sub-algorithm and their server.

Servers Servers are used to collect all relevant data structures and to delegate the operations on these data structures that are required by the main algorithm as well as by the sub-algorithms. The server used by the main algorithm and the sub-algorithms has to be the same because all algorithms have to operate on the same data, e.g. on the vector of iterates or on the vector of search directions.

Since the algorithms for solving complementarity constrained (Algorithm 10) and nonsmooth constrained (Algorithm 9) problems not only exchange certain algorithmic building blocks but also add some new ones, there are different servers for different problem classes. The specialized servers are generated by inheritance of the main server that is constructed for standard nonlinear problems.

Data Structures The main data structures of interior-point methods are on the one hand the vectors for the iterates y, for the right-hand sides ω of the KKT system and for the search directions Δy. On the other hand there are the problem matrices $\nabla c_{\mathcal{E}}, \nabla c_{\mathcal{I}}$ and H as well as the barrier term diagonal matrices Φ and Ψ. Like it is the case for sub-algorithms, the user can employ existing data structures or may implement new ones. Again, the only precondition for new implementations is the satisfaction of the fixed interfaces.

At this point, an aspect of strong but desired coupling has to be mentioned. The data structures of the problem matrices are not determined by the problem itself but by the sub-algorithm that solves the KKT system. Since this sub-algorithm is one of the most important ingredients of interior-point methods, this component has to determine the data structures on which it operates. As a consequence, the user who implements the problem has to work with these data structures. A special but frequently appearing case is the one in which the person implementing the problem is the person designing and implementing the KKT system solution algorithm. For instance, the example from stochastic programming mentioned in Section 5.1 fits into this situation.

Problem Adaptors Another component of the Clean::IPM software architecture are so-called problem adaptors. These adaptors mainly follow the adaptor pattern (see [13] or [46]) and wrap a user-given implementation of a problem in order to replace it, e.g. by a scaled or regularized version. For instance, when solving MPCC problems the user itself only has to implement the original MPCC formulation and to choose a certain regularization strategy. The corresponding adaptor then wraps the MPCC formulation such that the main algorithm only sees a regularized (NLP-type) version of the MPCC.

Figure 5.3 illustrates the above described components and their relationships. Loose coupling is represented by that most of the nodes of the components in Figure 5.3 are not connected. The special aspects that introduce stronger coupling to the design are represented by the black connections. Orthogonality of the software design is also represented by component nodes that are not directly connected. For instance, implementing a different line-search sub-algorithm for the barrier problems does not affect and is not affected by the chosen data structures for the barrier matrices or the scaling of the problem. There are no concrete possibilities of instantiations given in Figure 5.3. All leaves of the graph represent algorithmic (sub-)tasks for which a concrete

instantiation still has to be chosen. For instance, the leaf "μ-Update Rule" denotes that there has to be a choice for the barrier parameter updates. The set of concrete choices depends on what is implemented in the framework and what is extended by the user. See Figure 5.4 for a set of concrete instantiations of the update rule for barrier parameter.

5.3 Used External Libraries and Code Quality

The implemented method uses some external software libraries for sparse and dense linear algebra. For dense matrix-matrix, matrix-vector and all vector-vector operations, the default implementations of the vector data structures of Clean::IPM use LAPACK routines [5]. All operations with sparse matrices are self-implemented or use routines from the HSL Mathematical Software Library [58]. In addition, some libraries from Boost (see [124]) are used. The code documentation is done using Doxygen [126].

To achieve a high quality code some standard techniques are used. In addition to a high rate of documentation (see Table 5.1) the technique of *programming by contract* is applied (cf. [86]). By this, almost every method is encapsulated by so-called *pre-* and *postconditions* that try to check the correctness of the method.

Finally, the code is tested in nightly regression tests by a self-implemented regression test library consisting of (currently) 95 regression tests.

It seems to be an axiom of scientific software development that a library is never complete – and it seems to be true for Clean::IPM, too. Thus, lines of code or other quality measures like documentation-code-lines ratios are in the flow. However, Table 5.1 states the current numbers for (total) lines of code and comments.

Total lines of code	37 735
Lines of documentation & comments	12 894
Lines of programming by contract	815

Table 5.1: Code statistics for Clean::IPM.

Figure 5.3: A schematic overview of the software architecture of Clean::IPM.

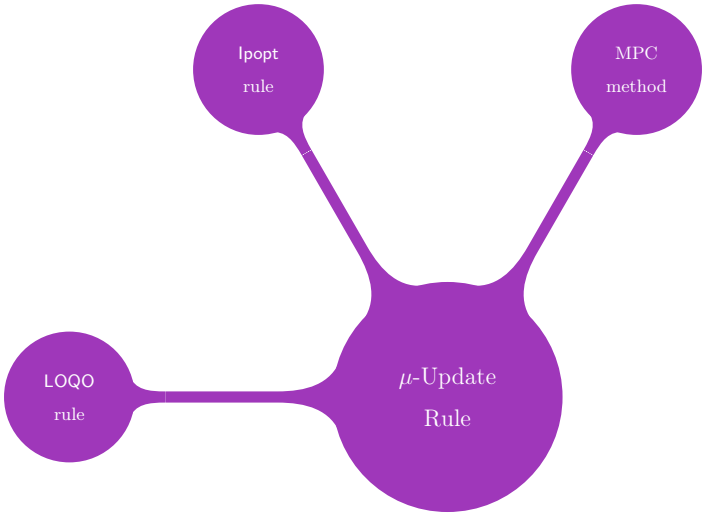

Figure 5.4: The current set of update rules for the barrier parameter implemented in Clean::IPM.

Chapter 6

Numerical Results

The main goal of this chapter is to document the generality of the Clean::IPM framework. First, in Section 6.1 and Section 6.2 two use cases of Clean::IPM are presented that include applications in stochastic programming and mathematical biology. Both are part of a diploma thesis or a Phd project. To solve these problems, some non-standard techniques are required and it is shown that the Clean::IPM framework can be used to develop these techniques and to integrate them in Clean::IPM. Section 6.3 presents some preliminary results on a large NLP test set, namely the Hock–Schittkowski test set. By this, the robustness of the basic interior-point algorithm of Clean::IPM is documented. Finally, Section 6.4 discusses numerical results concerning the problem of validation of nominations in gas transport networks (see Chapter 3) and shows that Clean::IPM can be used in real-world projects.

All numerical results presented in this chapter are produced with an instantiation of Clean::IPM without a feasibility restoration phase. An orthogonal design of the feasibility restoration phase within the interior-point framework is not easy to determine if nonsmooth and MPCC-type problems should be solved, too. Thus, the work on this part of the implementation is not yet finished. However, the integration of a feasibility restoration phase increases the robustness of the algorithm and will be realized in the future.

6.1 Computation of Recombination Probabilities

In his diploma thesis [98] Probst uses Clean::IPM in the field of mathematical biology. He computes estimations of recombination probabilities based on monitoring of a population. In contrast to the standard methods in this field, e.g. stochastic simulation using Monte-Carlo techniques, he uses a Markov process model to achieve an initial value problem for the dynamics of the joint distributions of the population. For a given time interval $[0, T]$ and fixed initial and boundary values, this leads

to the constrained nonlinear least-squares problem

$$\min_{x\in\mathbb{R}^n} \quad \frac{1}{2}\left\|D\left(x,T\right)-\hat{D}\right\|_2^2 \tag{6.1a}$$

$$\text{s.t.} \quad \frac{\mathrm{d}\,D\left(x,t\right)}{\mathrm{d}\,t} = D\left(x,t\right)G^T + \lambda N\left(\Theta\left(x\right)-I\right)D\left(x,t\right), \quad D\left(x,0\right) \text{ given}, \tag{6.1b}$$

$$\sum_{i=1}^{n} x_i = 1, \tag{6.1c}$$

$$x \in [0,1]^n. \tag{6.1d}$$

The used model is highly complicated. Here, only a brief description is given. The components of the vector $x \in \mathbb{R}^n$ are the recombination probabilities. $\hat{D}$ is the given distribution data and D denotes the vector of partial distributions which jointly characterize allelic states at a number of loci at the same or at different chromosomes. The matrix multiplication with G^T models the impact of mutation on the distributions. This mutation is modeled by a Markov semi-group operator. λ is the parameter of the exponential distribution that models the waiting (lifetime) period and $\Theta(x)$ denotes the transition matrix of the Markov semi-group operator of the evolutionary process at the moment of recombination. Finally, N is half of the size of the considered population. See [98] for the details.

It turns out that (6.1) is ill-conditioned and thus, the initial value problem (6.1b) has to be solved with high accuracy in every iteration. Unfortunately, the solution of the initial value problem is the most expensive computation in solving (6.1). As a remedy, Probst uses the generic Clean::IPM framework to develop an adaptive accuracy control for the solution of the initial value problems. This adaptive control depends on the progress made by the interior-point algorithm with respect to the objective function (6.1a). By this, Probst shows that it is possible to control the accuracy of the evaluation of the problem data depending on quantities of the algorithm. This can also be crucial for a lot of other applications, especially those incorporating differential equations.

In his computational experiments, Probst uses the KKT system solution algorithm for dense matrix blocks (Algorithm 2) and compares different update rules for the barrier parameter. The LOQO rule (see Section 4.1.4) turns out to be the most robust. Finally, he compares this concrete instantiation of the Clean::IPM framework with an extended Levenberg–Marquardt method. Using adaptive accuracy control, both methods are able to solve the problems in the magnitude of minutes on a desktop computer with 2.6 GHz and 12 GB RAM.

6.2 Large-Scale Stochastic Programming

In his PhD project, Hübner[1] develops a highly problem-specific parallel solution algorithm for KKT systems arising in interior-point methods for multistage stochastic programming. By additionally developing problem-specific parallel vector data structures, he is able to solve *huge*-scale instances of a certain class of multistage stochastic programs with Clean::IPM.

The background of the application is to manage a portfolio with n assets optimally over a given time horizon $[0, T]$. The objective is to minimize the risk with respect to a fixed outcome. The concrete modeling of future and uncertainty consists of a discrete time horizon $[0, 1, \ldots, T]$ and a discrete event space depending on the number of assets. Both together lead to a so-called event tree, i.e. a tree with depth T and branching $n+1$ for every inner node in the tree. For more details on the topic of portfolio optimization see [119].

Steinbach [118, 120] considers these multistage stochastic convex quadratic programs as control problems over a tree. Following his ideas, let V be the set of nodes of the tree and let L be the set of leaves, i.e. the set of scenarios. For a node $j \in V$, $\pi(j)$ denotes its predecessor. With the state variables $x_j \in \mathbb{R}^{n+1}$ modeling the value of the assets and the control variables $u_j \in \mathbb{R}^{2n}$ modeling the buys and sells, one obtains a so-called *tree-sparse program with incoming control* (see [120] for the details);

$$\min_{(x,u)} \quad \sum_{j \in L} \frac{1}{2} x_j^T H_j x_j \tag{6.2a}$$

$$\text{s.t.} \quad x_j - G_j x_{\pi(j)} - E_j u_j - h_j = 0 \quad \forall j \in V, \tag{6.2b}$$

$$u_j \in \left[u_j^-, u_j^+\right] \quad \forall j \in V, \tag{6.2c}$$

$$x_j \in \left[x_j^-, x_j^+\right] \quad \forall j \in V, \tag{6.2d}$$

$$\sum_{j \in V} (D_j u_j + F_j x_j) + e_V = 0. \tag{6.2e}$$

(6.2b) models the dynamics of the program, (6.2e) is incorporated to fix the outcome. (6.2c) and (6.2d) are simple control and state variable bounds, respectively. Finally, notice that the objective function is only defined on the leaves of the tree.

A further investigation of (6.2) shows that the Hessian and the Jacobian of the constraints possess a highly structured sparsity pattern that is determined by the tree of the multistage stochastic program. Hübner exploits this structure in order to develop a parallel solution algorithm for the KKT system. He uses Clean::IPM as the main solution algorithm for the QP and modifies all vector data structures such that every matrix-vector, vector-vector and scalar-vector operation of the interior-point algorithm is parallelized in a problem-specific way. In addition, the algorithmic

[1] Dipl.-Math. Jens Hübner, huebner@ifam.uni-hannover.de, Institute of Applied Mathematics, Leibniz Universität Hannover

building block for solving the KKT system is replaced by a highly problem-specific parallel solution algorithm. This results in a completely parallelized interior-point method for distributed memory systems.

The according numerical results are highly encouraging. Table 6.1 shows the model statistics of the largest problems Hübner is able to solve using Clean::IPM. The nonzero elements are the nonzero elements of the KKT matrix. The problem size of the largest instance of (6.2) is up to almost 11 billion variables and 4 billion equality constraints. The computations are done on the "tane" cluster of the RRZN[2]. This cluster consists of 96 compute nodes with 12 processors per node. Every processor is an Intel Xeon X5670 with 2.93 GHz and every node has 48 GB RAM.

n (in mill.)	$\lvert\mathcal{E}\rvert$ (in mill.)	Nonzero entries (in mill.)	Processors	Required memory (in GB)	Wallclock time (in s)
110	39	332	12	34	44
311	111	833	36	103	88
421	153	997	60	134	31
1 210	435	3 050	120	413	49
1 872	648	7 454	180	837	51
3 111	1 111	8 333	420	1 104	36
10 896	3 922	27 458	1 080	3 760	48

Table 6.1: Model statistics and solution times of *huge*-scale tree-sparse convex programs.

This application shows two important advantages of the Clean::IPM framework. First, Hübner gives a proof of concept that the generic concept of Clean::IPM can be exploited to achieve a highly problem-specific instantiation of the interior-point framework. With this, it is possible to solve extremely large instances very efficiently. Secondly, solving problems of the sizes given in Table 6.1 also confirms the robustness and quality of the implementation.

6.3 The Hock–Schittkowski Test Set

The Hock–Schittkowski test set [57] is a traditional test set containing low-dimensional nonlinear optimization problems. Many of them are nonconvex and possess different local minima.

The Clean::IPM framework is tested on the Hock–Schittkowski test set as a part of the CUTEr test set [49]. It contains 126 problems. 9 problems are excluded from the test set due to CUTEr-specific compilation problems. The remaining 117 problems are solved by 6 different instantiations of the Clean::IPM interior-point framework. These instantiations differ

[2]Regionales Rechenzentrum Niedersachsen, Hannover, `http://www.rrzn.uni-hannover.de`

1. in the algorithmic aspect, if a monotone or mixed strategy for the barrier parameter updates is used and

2. in the concrete update rule for the barrier parameter (LOQO rule, Ipopt rule or Mehrotra's predictor-corrector method).

The stopping criterion presented in Section 4.1 is slightly modified such that (4.23) is replaced by

$$\theta_{\text{pri}} < \varepsilon_{\theta_{\text{pri}}}, \quad \theta_{\text{dual}} < \varepsilon_{\theta_{\text{dual}}}, \quad \theta_{\text{compl}} < \varepsilon_{\theta_{\text{compl}}}, \tag{6.3}$$

with constants

$$\varepsilon_{\theta_{\text{pri}}} := 10^{-4}, \quad \varepsilon_{\theta_{\text{dual}}} := 1, \quad \varepsilon_{\theta_{\text{compl}}} := 10^{-4}. \tag{6.4}$$

In addition, the scaled optimality measure (cf. (4.24)) as used in Ipopt [134] has to satisfy the tolerance 10^{-6}. The maximum number of iterations is set to 10^5.

The numerical results in the rest of this chapter are presented using performance profiles in the form proposed by Dolan and Moré in [32]: For a set of problems $\mathcal{P}$ and a set of solvers (or solver options) $\mathcal{S}$, the *performance measure*

$$t_{p,s} := \text{iterations required to solve problem } p \in \mathcal{P} \text{ by solver } s \in \mathcal{S} \tag{6.5}$$

is used. Here, $\mathcal{P}$ is the subset of the Hock–Schittkowski test set with $|\mathcal{P}| = 117$ and $\mathcal{S}$ is the set of different instantiations of the Clean::IPM framework with $|\mathcal{S}| = 6$. The *performance ratio* is defined by

$$r_{p,s} := \frac{t_{p,s}}{\min\{t_{p,s} \colon s \in \mathcal{S}\}}. \tag{6.6}$$

Then

$$\rho_s(\tau) := \frac{1}{|\mathcal{P}|} \left|\{p \in \mathcal{P} \colon r_{p,s} \leq \tau\}\right| \tag{6.7}$$

is the fraction of problems that are solved by solver s within a factor $\tau \geq 1$ of the performance of the best solver for problem p. The performance profile $\rho_s(\tau) : \mathbb{R} \to [0,1]$ is non-decreasing and piecewise constant (cf. [32]). Its basic interpretation is as follows: The value of $\rho_s(1)$ is the probability that solver s is the best of all tested solvers (with respect to the performance measure $t_{p,s}$). Moreover, set $r_M \geq r_{p,s}$ for all p and all s and $r_{p,s} = r_M$ if and only if solver s does not solve problem p. Thus, $r_{p,s} \in [1, r_M]$ and

$$\rho_s^* := \lim_{\tau \nearrow r_M} \rho_s(\tau) \tag{6.8}$$

can be interpreted as the probability that solver s solves a problem. In other words, $\rho_s(1)$ is a measure of efficiency whereas ρ_s^* is a measure of robustness.

Figure 6.1 shows the performance profiles for Clean::IPM with a mixed strategy for the barrier parameter updates on the Hock–Schittkowski test set using a $\log_2$-scale, i.e.

$$\rho_s(\tau) = \frac{1}{|\mathcal{P}|} \left|\{p \in \mathcal{P} \colon \log_2(r_{p,s}) \leq \tau\}\right| \tag{6.9}$$

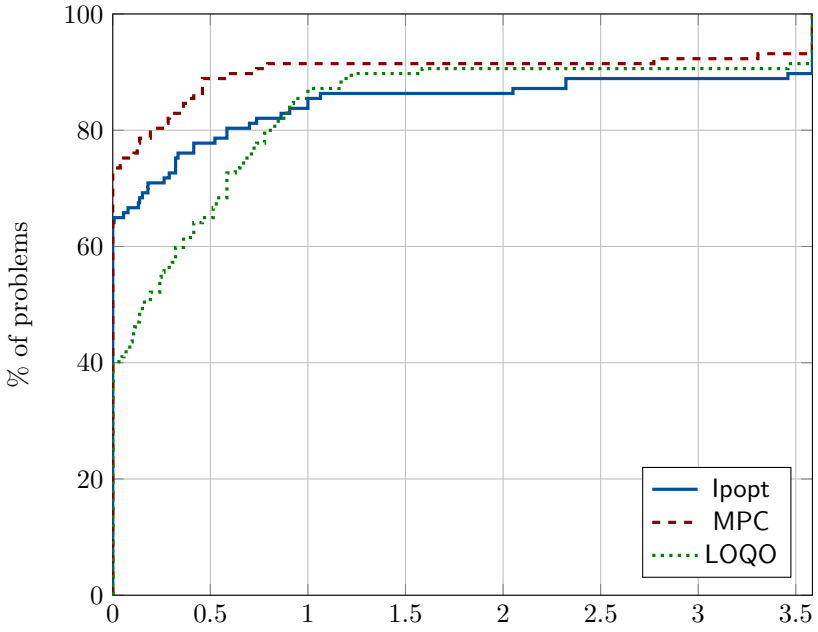

Figure 6.1: Clean::IPM with mixed μ-strategies on the Hock–Schittkowski test set.

is plotted instead of (6.7). The best mixed instantiation of the Clean::IPM framework for this test set is the one using Mehrotra's predictor-corrector method (denoted by MPC in the figures). It outperforms both the Ipopt and the LOQO update rule in terms of efficiency and robustness. Using the Ipopt rule leads to a more efficient algorithm than using the LOQO rule. All three instantiations are comparable in terms of robustness. Figure 6.2 shows the performance profiles for instantiations of Clean::IPM with a monotone update strategy for the barrier parameter. These monotone versions of the algorithm are as robust as the mixed versions. Here, using the LOQO rule or Mehrotra's predictor-corrector algorithm is more efficient than using the Ipopt rule.

If one considers both the solutions of the monotone and the mixed instantiations, it turns out that 7 problems remain unsolved. These problems are HS15, HS27, HS62, HS84, HS87, HS99 and HS102. By tuning the parameters of the algorithm or modifying the starting point, it is possible to achieve optimal solutions for HS15 (using the reduced vector management described in Section 4.1.7 with the LOQO rule and the modified starting point $0 \in \mathbb{R}^n$), HS27 (using the modified starting point $0 \in \mathbb{R}^n$) and HS102 (using the monotone version and the initial barrier parameter 10^4). This corresponds to a success rate of 96.6 %. The remaining 4 problems HS62, HS84, HS87 and HS99 are all solved to primal feasibility without satisfying the dual feasibility tolerance.

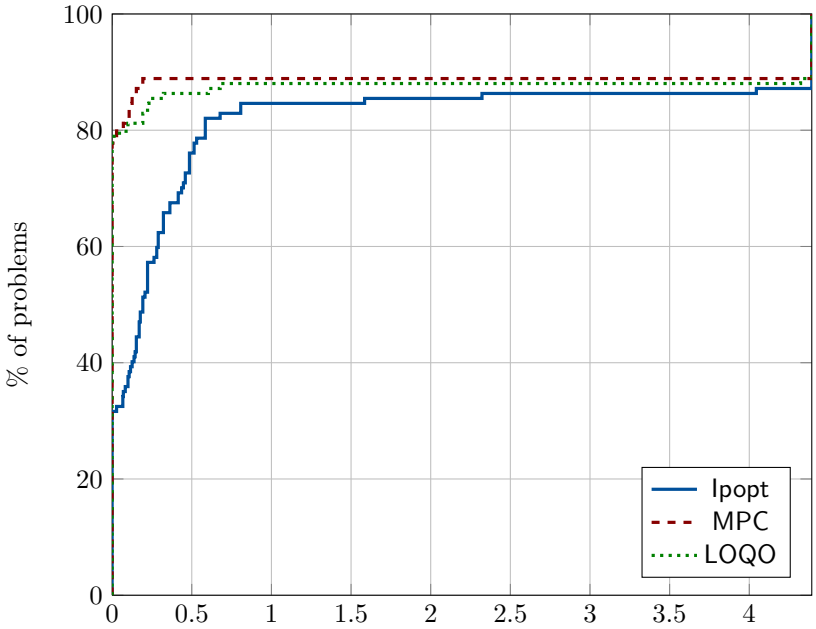

Figure 6.2: Clean::IPM with monotone μ-strategies on the Hock–Schittkowski test set.

6.4 Gas Network Planning

In Chapter 3, a nonsmooth, nonconvex and nonlinear MPCC model of the problem of validation of nominations is developed. Here, numerical results for an implementation of this model are presented.

The data used in this section is real-world data from the industry partner Open Grid Europe GmbH (OGE) within the ForNe project. The considered network is the northern high-calorific gas transport network of OGE. Figure 6.3 shows a schematic plot of the network.[3] The colored edges correspond to active elements or resistors (compressor stations are red, control valves are blue, valves are green and resistors are cyan). This network ranges from the North Sea in the north to the Ruhr area in the south and from the Netherlands in the west to Saxony-Anhalt in the east. It consists of more than 1200 km of pipelines and several compressor stations, control valves and valves. A statistic about the network can be found in Table 6.2. Although the northern high-calorific gas transport network is the smallest transport network of OGE, it is a large-scale instance with respect to the considered problem class. To the best of the author's knowledge, no other work beside the ForNe project exists that solves mixed-integer, nonconvex and nonlinear problems on networks of this size. As a part of the project ForNe, tools have been developed to automatically generate nominations. These nominations are based on historical data and the

[3]The plot is generated with LaMaTTO++, a framework for modeling and solving mixed-integer nonlinear programming problems on networks. It was originally developed by the working group of Alexander Martin, Friedrich-Alexander Universität Erlangen-Nürnberg, and is now used and extended in the research project ForNe.

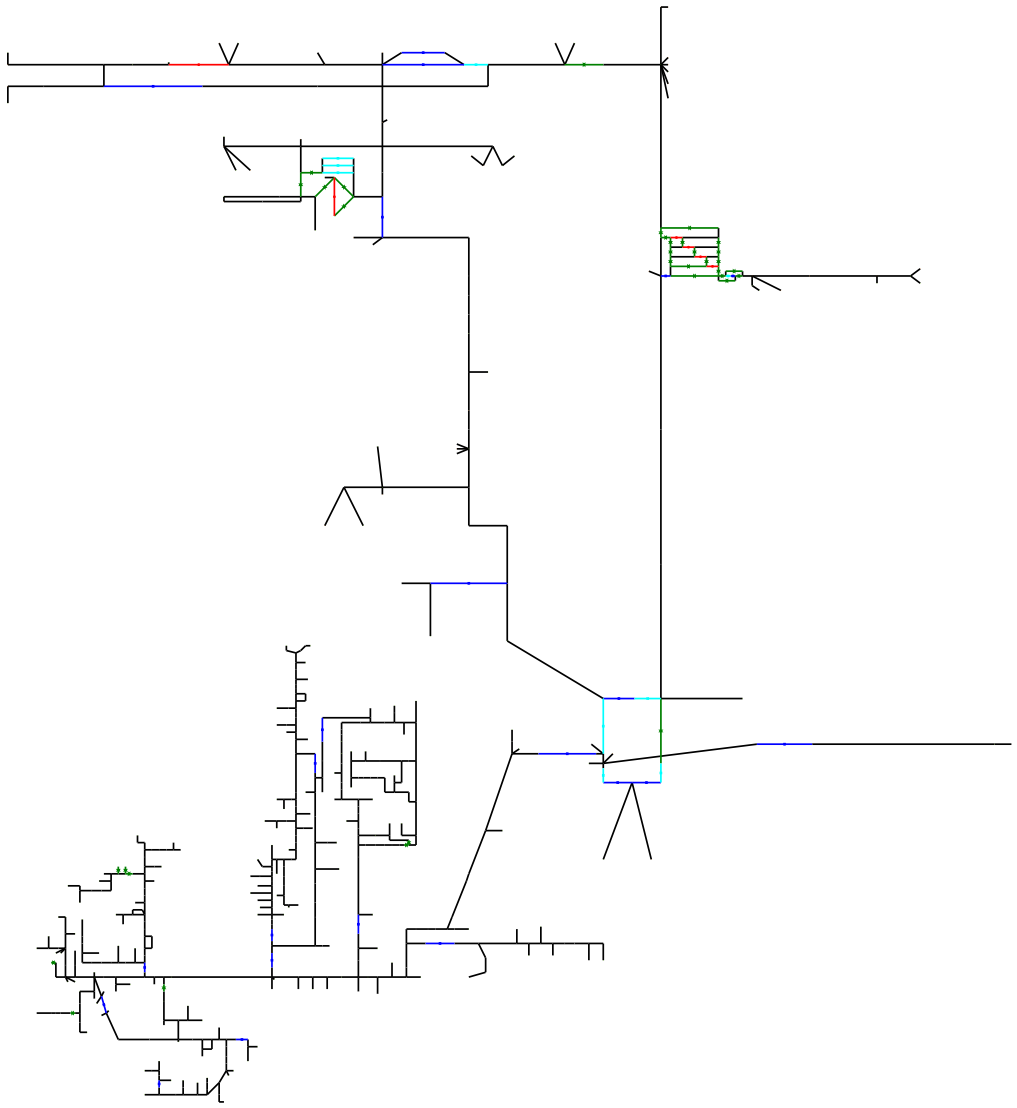

Figure 6.3: Schematic plot of the northern high-calorific gas transport network of Open Grid Europe GmbH.

current contract situation of OGE. The details will be published in the upcoming book [67].

To be able to solve the problem of validation of nominations an additional interface for (nonlinear) optimization models on networks is developed for this thesis. In the following, this interface is referred to as iGNO[4]. It is implemented in C++ and uses the Boost graph library BGL [113] as a basis for all network-specific components of the interface. In addition, iGNO provides parsers for handling the network and the nomination data. This data is given in XML and all parsers are based on the C++-based XML parser Xerces [1]. It should be mentioned that the developed interface is not restricted to gas networks and thus, can also be used for other optimization appli-

[4] iGNO is an acronym for *Interface for Gas Network Optimization*.

Network elements	Count
Pipes	452
Short cuts	99
Resistors ($\mathbb{A}_{\text{re}}^{\text{pwc}}$)	1
Resistors ($\mathbb{A}_{\text{re}}^{\text{pwq}}$)	8
Valves	35
Control valves	23
Compressor stations	6
Sources	31
Sinks	129
Junctions	432

Table 6.2: Network elements in the northern high-calorific gas transport network of Open Grid Europe GmbH.

cations on networks like drinking water or sewer network optimization. During the development of iGNO, techniques for increasing the software quality are applied as it is described for Clean::IPM in Section 5.3.

The numerical results presented in the following are based on the reformulated nonsmooth and complementarity constrained model discussed in Section 3.3. The size of the reformulated model is stated in Table 6.3. As it is already mentioned in Section 3.3 the used reformulation technique

Variables	1 963
Equality constraints	1 872
Inequality constraints	7
Complementarity constraints	64

Table 6.3: Size of the reformulated MPCC model (for the northern high-calorific gas network).

does not lead to an MPCC model in standard form. The reason for this is that the characteristic functions of the non-disjunctive states of active network elements do not have to be non-negative. For being able to apply standard regularization techniques like penalization or relaxation, the characteristic functions can be squared or they can be replaced by their absolute values. The model sizes given in Table 6.3 are the ones for the model without applying additional modifications of the characteristic functions. Both squaring the characteristic functions and using their absolute values lead to n_χ additional equality constraints and n_χ additional auxiliary variables. Here, n_χ is equal to the number of characteristic functions in the model that are used to set up the complementarity

constraints. For the northern high-calorific gas network, n_χ is given by

$$n_\chi = 2\left(|\mathbb{A}_{\mathrm{vl}}| + |\mathbb{A}_{\mathrm{cv}}| + |\mathbb{A}_{\mathrm{cs}}|\right) = 128. \tag{6.10}$$

If the characteristic functions are squared, the additional equality constraints read

$$\left(\chi_a^s\right)^2 - s_{\chi_a^s} = 0, \tag{6.11}$$

where χ_a^s is the characteristic function of state s and arc $a \in \mathbb{A}$. $s_{\chi_a^s} \geq 0$ is a new auxiliary variable that is used in the complementarity constraints instead of χ_a^s. In analogy, if the absolute values of the characteristic functions are used, the additional equality constraints read

$$|\chi_a^s| - s_{\chi_a^s} = 0. \tag{6.12}$$

Obviously, this formulation increases the nonsmoothness of the problem and it is easy to see that every MPCC-feasible point is a point at which at least the auxiliary constraints (6.12) fail to be differentiable. This property increases the hardness of the problem significantly. Nevertheless, it might be useful to try to solve the formulation using (6.12) instead of the one using (6.11). This aspect is discussed in more detail at the end of this chapter. In the following, the nonsmooth MPCC model using (6.11) is referred to as nMPCC-s and the model using (6.12) is called nMPCC-a.

In what follows, results for a set of 1000 randomly chosen nominations are presented. These nominations are generated within the ForNe project. All computations are done with an Intel Core i7 CPU 920 with 2.67 GHz and 12 GB RAM. The operating system is openSUSE 12.1. All executables are generated without any code optimization with the GCC compiler version 4.7.1.

The accomplished computational experiments show that the penalization scheme for regularizing the complementarity constraints leads to better numerical results than the relaxation scheme. Thus, the following presentation concentrates on the penalization method. As the MPCC-complementarity measure, (4.185) is used together with the adaptive tolerance defined in (4.186). The update parameter κ_ξ in (4.189) is set to $\kappa_\xi := 0.1$. The initial value of the penalization parameter ξ is 10^4. Moreover, the computational experiments show that it increases the robustness of the method if a lower bound on the penalization parameter is incorporated. This lower bound is set to 10^{-8}.

Figure 6.4 shows the performance profile for different instantiations of Clean::IPM applied to the nMPCC-s model. The results are generated with a monotone strategy for the updates of the barrier parameter and the Ipopt rule, the LOQO rule and Mehrotra's predictor-corrector method. In addition, two different initial values for the barrier parameter are tested ($\mu^{(0)} = 10^{-1}$ and $\mu^{(0)} = 10^2$). It can be seen that all described instantiations lead to a comparable performance but the larger initial value for the barrier parameter leads to slightly more robust algorithms. Since the update of the regularization parameter is directly coupled with the barrier parameter (cf.

(4.188)), a larger initial value $\mu^{(0)}$ may lead to earlier updates of the barrier parameter and thus, to earlier updates of the penalization parameter. These larger penalization parameters accentuate the complementarity constraints in the earlier barrier problems. This might be the reason for the increased robustness.

In addition to the performance profiles of the 6 instantiations of Clean::IPM, Figure 6.4 also shows the *best-of* profile, i.e. the performance profile that is achieved if one takes the best of all 6 solver instantiations for a given nomination. Obviously, this line is constant for all τ and its value (≈ 90 % in Figure 6.4) is the percentage of instances that is solved by at least one of the algorithmic instantiations. By this, it can be seen that the solution process for these instances significantly depends on the chosen algorithmic options. This is a well-known fact for nonlinear optimization in general and seems to be particularly true for the considered class of nonsmooth MPCCs. Since the problem of validation of nominations is a feasibility problem, a strong suggestion based on these results is to test a few algorithmic options in order to achieve a feasibility certificate.

Table 6.4 shows a statistic of the minimum, maximum and average iteration numbers of all successfully finished runs. The analogous information about the solution times can be found in Table 6.5. It should be remarked, that approximately 2/3 of the solution time is used for the evaluation of the problem data.

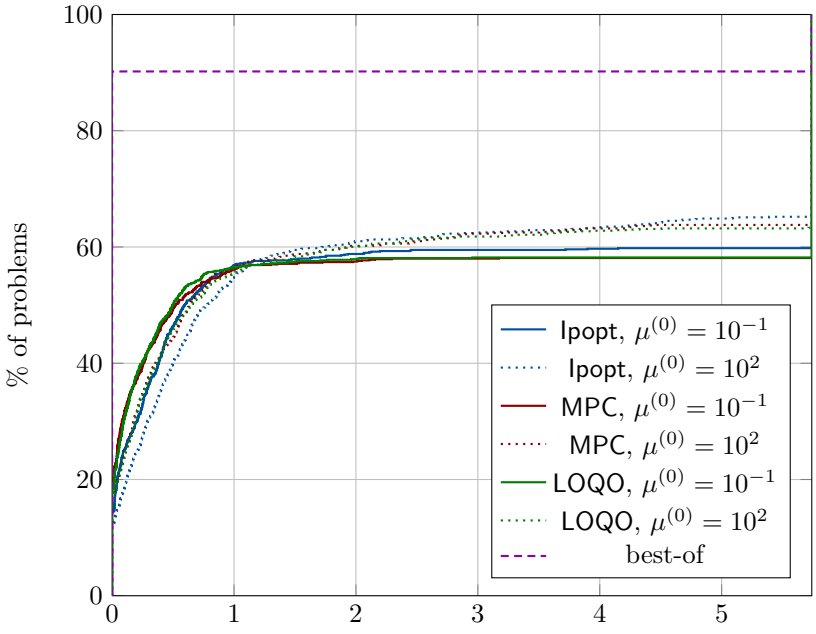

Figure 6.4: Performance profiles for Clean::IPM applied to the nMPCC-s model.

As it is already mentioned above, the nMPCC-a model is much harder to solve than the nMPCC-s model. The nMPCC-s model only contains the nonsmooth constraints of the component model of resistors $a \in \mathbb{A}_{\mathrm{re}}^{\mathrm{pwq}}$ with a piecewise quadratic pressure loss (cf. Section 3.2.4). In its original

Solver option	min. k	max. k	avg. k
Ipopt rule, $\mu^{(0)} = 10^{-1}$	41	1 866	112
Ipopt rule, $\mu^{(0)} = 10^{2}$	40	2 807	213
MPC rule, $\mu^{(0)} = 10^{-1}$	47	703	95
MPC rule, $\mu^{(0)} = 10^{2}$	43	2 086	168
LOQO rule, $\mu^{(0)} = 10^{-1}$	47	1 479	95
LOQO rule, $\mu^{(0)} = 10^{2}$	45	5 241	167

Table 6.4: Statistics of the iteration numbers (k) for Clean::IPM applied to the nMPCC-s model.

Solver option	min. t	max. t	avg. t
Ipopt rule, $\mu^{(0)} = 10^{-1}$	6	442	17
Ipopt rule, $\mu^{(0)} = 10^{2}$	6	425	31
MPC rule, $\mu^{(0)} = 10^{-1}$	7	130	15
MPC rule, $\mu^{(0)} = 10^{2}$	7	325	27
LOQO rule, $\mu^{(0)} = 10^{-1}$	5	217	14
LOQO rule, $\mu^{(0)} = 10^{2}$	6	300	25

Table 6.5: Statistics of the solution times (t, in s) for Clean::IPM applied to the nMPCC-s model.

version, the component model of resistors only contains a second-order discontinuity due to the term $|q|q$ in (3.48). Notice that this constraint can also be re-written in a completely smooth way. This can be done by a variable splitting for the mass flow variable q_a, i.e.

$$q_a = q_a^+ - q_a^-, \quad q_a^+, q_a^- \geq 0, \quad q_a^+ q_a^- = 0. \tag{6.13}$$

This reformulation only shifts the difficulty of the problem of nonsmoothness to another place because it introduces additional complementarity constraints. Moreover, the presented computational results show that this kind of nonsmooth constraints does not disturb the algorithm significantly. However, the reformulation that is required such that the model fits the standard form (4.138) of an optimization problem with locatable and separable nonsmoothness leads to a first-order discontinuity. Moreover, it is not necessarily the case that a feasible point has to be directly at this first-order discontinuity. In contrast to that, all MPCC-feasible points of the nMPCC-a model are points of first-order discontinuities.

The increasing difficulty can be seen in the corresponding performance profiles in Figure 6.5. The *best-of* profile states that only slightly more than 20 % of the nominations are solved by any of the tested algorithmic instantiations. Furthermore, it can be seen that smaller initial values of the barrier parameter lead to more robust algorithms in this case. Only a few instances can be

solved with the larger initial barrier parameter.

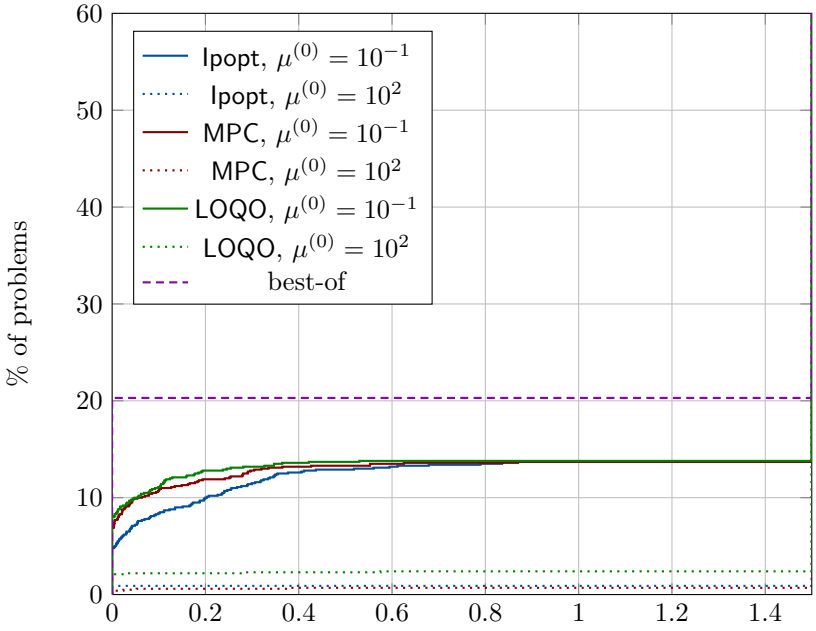

Figure 6.5: Performance profiles for Clean::IPM applied to the nMPCC-a model.

Solver option	min. k	max. k	avg. k
Ipopt rule, $\mu^{(0)} = 10^{-1}$	153	852	358
MPC rule, $\mu^{(0)} = 10^{-1}$	165	2 450	348
LOQO rule, $\mu^{(0)} = 10^{-1}$	162	2 450	344

Table 6.6: Statistics of the iteration numbers (k) for Clean::IPM applied to the nMPCC-a model.

Solver option	min. t	max. t	avg. t
Ipopt rule, $\mu^{(0)} = 10^{-1}$	24	147	57
MPC rule, $\mu^{(0)} = 10^{-1}$	25	427	56
LOQO rule, $\mu^{(0)} = 10^{-1}$	26	429	56

Table 6.7: Statistics of the solution times (t, in s) for Clean::IPM applied to the nMPCC-a model.

In the following, solution processes of Clean::IPM for an exemplary nMPCC-a instance and an exemplary nMPCC-s instance are analyzed. Comparing the statistics of the iteration number of the nMPCC-s model (Table 6.4) and the nMPCC-a model (Table 6.6) it is obvious that the model with the larger amount of nonsmooth aspects tends to require more iterations. Since the solution time is almost proportional to the number of iterations, this can also be seen in Table 6.5 and Table 6.7.

Figure 6.6 and Figure 6.7 show the progress in primal and dual infeasibility for an exemplary nMPCC-s and an exemplary nMPCC-a instance. One can see that the progress in primal feasibility is comparable except for the number of iterations that are required to achieve this progress. This is additionally illustrated in Figure 6.8 and Figure 6.9. There are much more primal step lengths in $[0.1, 1]$ for the nMPCC-s instance than for the nMPCC-a instance, where most of the primal step lengths are less than 0.1. The reason for this is that both the step length truncation rule as well as the modified filter line-search procedures (cf. Section 4.2.3 for both) lead to shorter step lengths. Since the primal search directions are in the same order of magnitude for both instances, this leads to shorter steps and slow convergence for the nMPCC-a instances. In most of the cases for which Clean::IPM fails to solve the nMPCC-a model, the combination of the step length truncation rule and the subsequent line-search leads to very short primal step lengths that are finally rejected by the line-search.

Nevertheless, it is possible to solve about 20 % of the nMPCC-a instances for the real-world and large-scale instances on the northern high-calorific gas transport network of OGE. Moreover, it turns out that some of the instances that cannot be solved for the nMPCC-s model can be solved for the nMPCC-a model. Combining both the results for the nMPCC-s model and the nMPCC-a model, approximately 93 % of the randomly chosen nominations are decided to be feasible. Thus, the Clean::IPM framework can be successfully used to solve the challenging problem of validation of nominations for real-world instances.

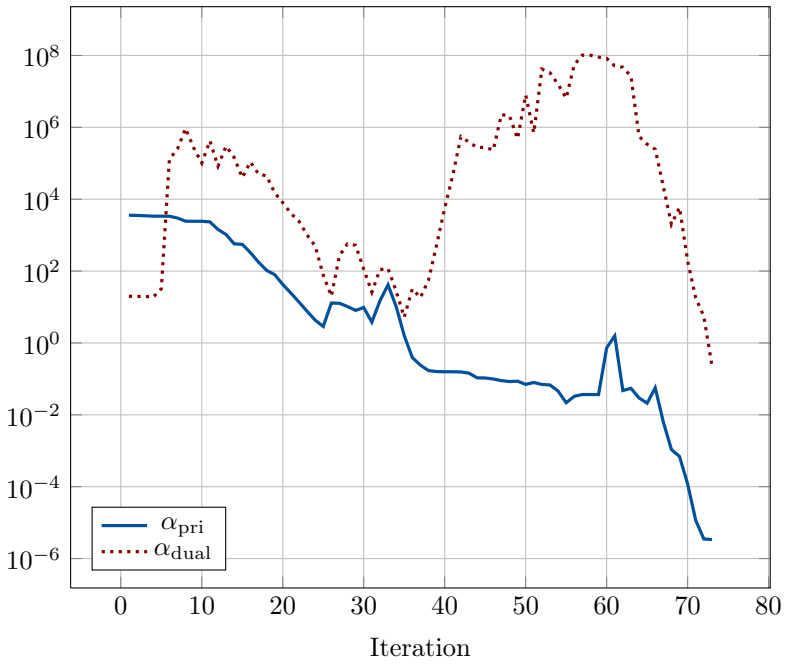

Figure 6.6: Primal and dual infeasibility during the solution of an exemplary nMPCC-s instance.

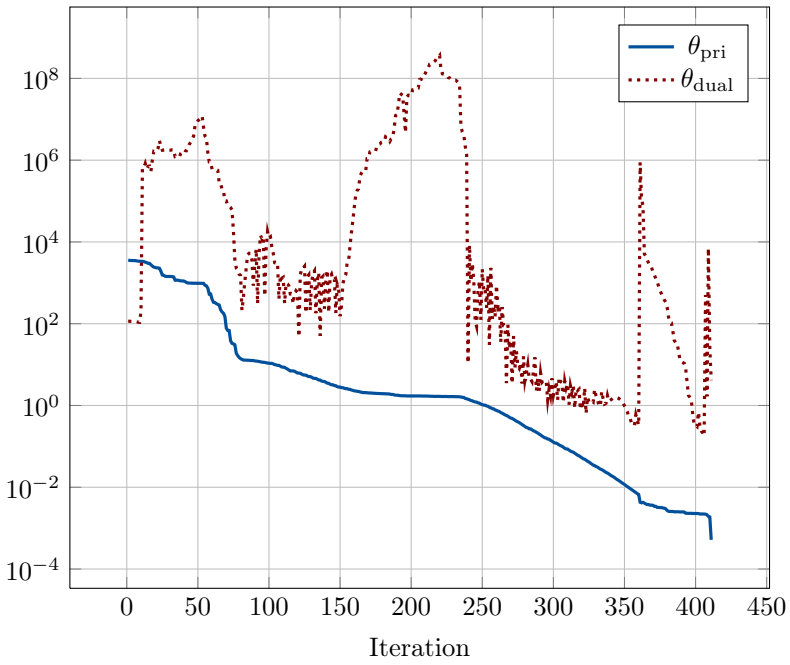

Figure 6.7: Primal and dual infeasibility during the solution of an exemplary nMPCC-a instance.

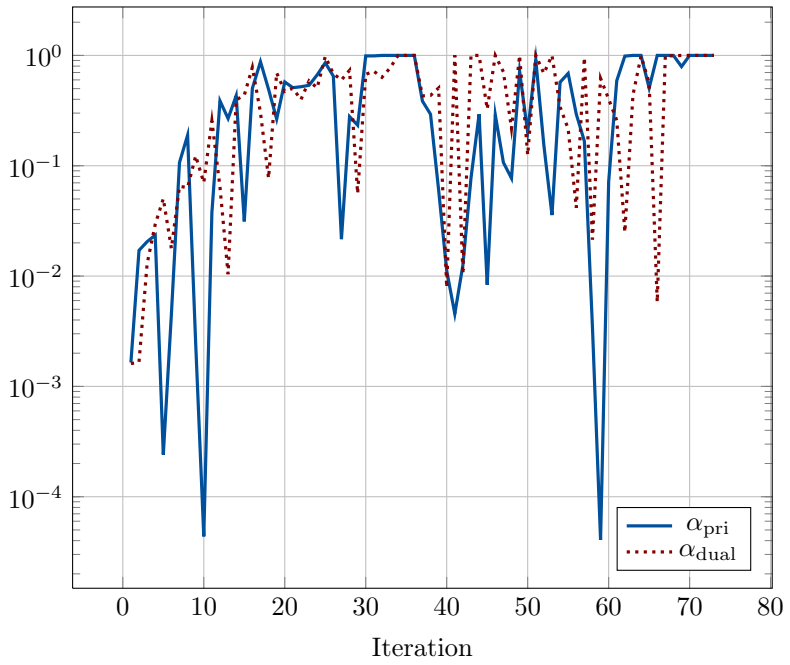

Figure 6.8: Primal and dual step lengths during the solution of an exemplary nMPCC-s instance.

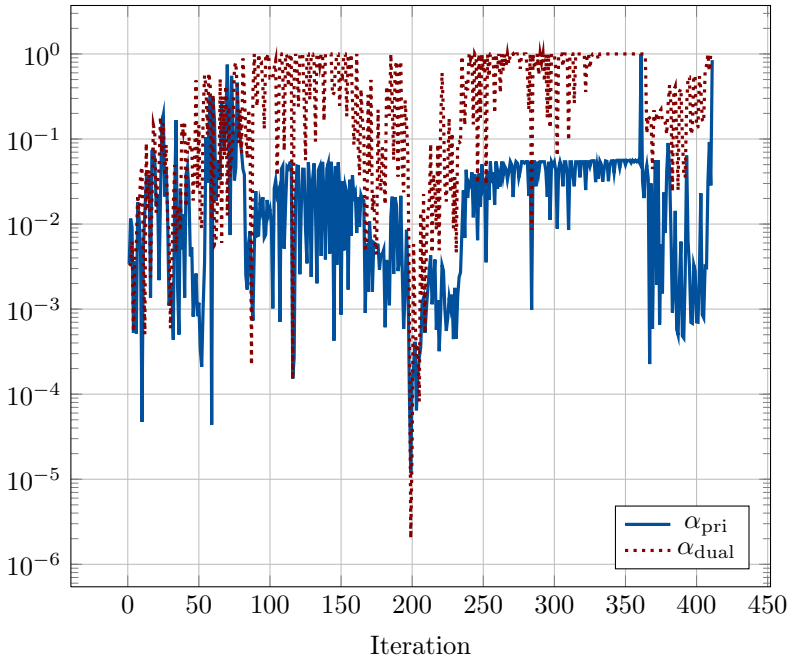

Figure 6.9: Primal and dual step lengths during the solution of an exemplary nMPCC-a instance.

Chapter 7

Conclusions and Outlook

In this thesis the generic interior-point framework Clean::IPM for nonlinear optimization is presented. As the main application, the problem of validation of nominations in gas transport networks is considered. This problem leads to a mixed-integer, nonsmooth, nonconvex and nonlinear feasibility problem that is intractable for state-of-the-art optimization solvers. Thus, a reformulation technique for a certain class of mixed-integer nonlinear problems is developed and applied to the considered problem. The result is a nonsmooth MPCC model. It is then described how the generic Clean::IPM framework can be extended in order to solve a certain class of nonsmooth problems as well as complementarity constrained problems. By reasons of the carefully chosen algorithmic design it is possible to combine both extensions leading to a solution approach for the reformulated nonsmooth MPCC model of the problem of validation of nominations.

The presented computational results first discuss some use cases of the Clean::IPM framework in the fields of mathematical biology and stochastic programming. The variety of these use cases documents the generality of the framework and gives a proof of concept for the design goals of Clean::IPM. In addition, preliminary results for Clean::IPM are presented on the Hock–Schittkowski test set in order to illustrate the robustness and performance of the basic interior-point method.

Finally, numerical results for Clean::IPM applied to the problem of validation of nominations in gas transport networks are given. It is shown that the extended interior-point method can be used to solve the nonsmooth and complementarity constrained model of the problem of validation of nominations. Nevertheless, a number of improvements are possible. As the computational experiments for strongly nonsmooth constrained problems show, there is room for improvement in the implementation of the extensions of Clean::IPM that handle the nonsmooth aspects of the problems. It can be seen that problems occur if many constraints fail to be differentiable in the limit of the iterates. It is thinkable that more carefully chosen parameters and tolerances of the algorithm can improve its robustness significantly. Apart from that, completely different strategies

for handling these situations are imaginable. One promising approach that will be investigated in the future is the detection of constraints that fail to be differentiable in the limit, fixation of the corresponding variables and projection of the problem. Beside these algorithmic developments a more realistic model, in particular of compressor stations, of the problem of validation of nominations should be implemented. This leads to larger models but the problem class stays the same. Thus, it is likely that even these more realistic models can be solved by the Clean::IPM framework.

In addition, the theoretical properties of the interior-point algorithm for nonsmooth problems should be extended. Previous achievements are given in this thesis but the questions concerning convergence properties are still open.

Finally, the basic interior-point method is going to be tested on larger test sets (like CUTEr) and the extensions for solving MPCC problems should be tested on general test sets (e.g. the MacMPEC collection [73]). After realizing these tests, it is the aim of the author to make the Clean::IPM framework available as open source. The author thinks that this can be beneficial for the mathematical optimization community since the generality of the framework can be exploited in order to develop and test new algorithmic ideas in the field of interior-point methods and their applications.

Bibliography

[1] Xerces-C++ XML Parser. `http://xerces.apache.org/xerces-c/`.

[2] Andrei Alexandrescu. Traits: The else-if-then of types. C++ Report, April 2000.

[3] Andrei Alexandrescu. *Modern C++ Design: Generic Programming and Design Patterns Applied*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[4] E. D. Andersen and K. D. Andersen. *The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm*, pages 197–232. Kluwer Academic Publishers, 1999.

[5] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, 1992. `http://www.netlib.org/lapack`.

[6] Pia Bales. Hierarchische Modellierung der Eulerschen Flussgleichungen in der Gasdynamik. Master's thesis, Technische Universität Darmstadt, 2005.

[7] Mapundi K. Banda and Michael Herty. Multiscale modeling for gas flow in pipe networks. *Mathematical Methods in the Applied Sciences*, 31:915–936, August 2008.

[8] Mapundi K. Banda, Michael Herty, and Axel Klar. Gas flow in pipeline networks. *Networks and Heterogeneous Media*, 1(1):41–56, March 2006.

[9] Hande Y. Benson, David F. Shanno, and Robert J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Jamming and comparative numerical testing. Technical Report ORFE-00-02, Princeton University, 2000.

[10] Hande Y. Benson, David F. Shanno, and Robert J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Filter methods and merit functions. Technical report, Princeton University, 2001. Revised Version, September 2001.

[11] Hande Y. Benson, David F. Shanno, and Robert J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Complementarity constraints. Technical Report ORFE 02-02, Princeton University, 2002.

[12] Hans Georg Bock, Ekaterina Kostina, Hoang Xuan Phu, and Rolf Rannacher, editors. *Modeling, Simulation and Optimization of Complex Processes*. Springer, Berlin, 2005.

[13] Boost. Generic Programming Techniques. `http://www.boost.org/community/generic_programming.html`.

[14] Conrado Borraz-Sánchez and Roger Z. Ríos-Mercado. A procedure for finding initial feasible solutions on cyclic natural gas networks. In *Proceedings of the 2004 NSF Design, Service and Manufacturing Grantees and Research Conference*, Dallas, USA, January 2004.

[15] Conrado Borraz-Sánchez and Roger Z. Ríos-Mercado. A Hybrid Meta-Heuristic Approach for Natural Gas Pipeline Network Optimization. In María Blesa, Christian Blum, Andrea Roli, and Michael Sampels, editors, *Hybrid Metaheuristics*, volume 3636 of *Lecture Notes in Computer Science*, pages 54–65. Springer, 2005.

[16] E. Andrew Boyd, L. Ridgway Scott, and Suming Wu. Evaluating the quality of pipeline optimization algorithms. In *PSIG 29th Annual Meeting, Tucson, Arizona*. Pipeline Simulation Interest Group, 1997. Paper 9709.

[17] Jens Brouwer, Ingenuin Gasser, and Michael Herty. Gas pipeline models revisited: Model hierarchies, nonisothermal models, and simulations of networks. *Multiscale Model. Simul.*, 9(2):601–623, 2011.

[18] Jens Burgschweiger, Bernd Gnädig, and Marc C. Steinbach. Optimization models for operative planning in drinking water networks. *Optim. Eng.*, 2008. Online First.

[19] Richard H. Byrd, Mary E. Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM J. Optim.*, 9(4):877–900, 2000.

[20] Richard H. Byrd, Jorge Nocedal, and Richard A. Waltz. KNITRO: An integrated package for nonlinear optimization. In *Large Scale Nonlinear Optimization, 35–59, 2006*, pages 35–59. Springer Verlag, 2006.

[21] Richard G. Carter. Compressor station optimization: Computational accuracy and speed. In *28th Annual Meeting*. Pipeline Simulation Interest Group, 1996. Paper 9605.

[22] Richard G. Carter. Pipeline optimization: Dynamic Programming after 30 years. In PSIG [96]. Paper 9803.

[23] Richard G. Carter, Don W. Schroeder, and Tim D. Harbick. Some Causes and Effects of Discontinuities in Modeling and Optimizing Gas Transmission Networks. Technical report, Stoner Associates, Carlisle, PA, USA, April 1994.

[24] Francis H. Clarke, Yuri S. Ledyaev, Ronald J. Stern, and Peter R. Wolenski. *Nonsmooth Analysis and Control Theory*, volume 178 of *Graduate Texts in Mathematics*. Springer, 1998.

[25] Frank H. Clarke. *Optimization and Nonsmooth Analysis*. Society for Industrial and Applied Mathematics, 1990.

[26] Diana Cobos-Zaleta and Roger Z. Ríos-Mercado. A MINLP model for a problem of minimizing fuel consumption on natural gas pipeline networks. In *Proc. XI Latin-Ibero-American Conference on Operations Research*, pages 1–9, 2002. Paper A48-01.

[27] Cyril Frank Colebrook. Turbulent flow in pipes with particular reference to the transition region between smooth and rough pipe laws. *Journal of the Institution of Civil Engineers*, 11:133–156, February 1939.

[28] Andrew R. Conn, Nicholas I. M. Gould, and Philippe Toint. *Trust-Region Methods*. MPS-SIAM series on optimization. SIAM, 2000.

[29] George B. Dantzig. On the significance of solving linear programming problems with some integer variables. *Econometrica*, 28(1):pp. 30–44, 1960.

[30] Daniel de Wolf and Yves Smeers. The gas transmission problem solved by an extension of the simplex algorithm. *Management Sci.*, 46(11):1454–1465, 2000.

[31] Angel-Victor DeMiguel, Michael P. Friedlander, Francisco J. Nogales, and Stefan Scholtes. A two-sided relaxation scheme for mathematical programs with equilibrium constraints. *SIAM J. Optim.*, 16(1):587–609, 2005.

[32] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91:201–213, 2002.

[33] Pia Domschke, Björn Geißler, Oliver Kolb, Jens Lang, Alexander Martin, and Antonio Morsi. Combination of nonlinear and linear optimization of transient gas networks. *INFORMS Journal of Computing*, 23(4):605–617, 2011.

[34] Klaus Ehrhardt and Marc C. Steinbach. KKT systems in operative planning for gas distribution networks. *Proc. Appl. Math. Mech.*, 4(1):606–607, 2004.

[35] Klaus Ehrhardt and Marc C. Steinbach. Nonlinear optimization in gas networks. In Bock et al. [12], pages 139–148.

[36] Miloslav Feistauer. *Mathematical Methods in Fluid Dynamics*, volume 67 of *Pitman Monographs and Surveys in Pure and Applied Mathematics Series*. Longman Scientific & Technical, Harlow, 1993.

[37] Anthony V. Fiacco and Garth P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. Wiley, New York, 1968. Reprinted by SIAM Publications, 1990.

[38] E. John Finnemore and Joseph E. Franzini. *Fluid Mechanics with Engineering Applications*. McGraw-Hill, 10th edition, 2002.

[39] A. Fischer. A special Newton-type optimization method. *Optimization*, 24(3-4):269–284, 1992.

[40] Roger Fletcher and Sven Leyffer. A bundle filter method for nonsmooth nonlinear optimization. Numerical Analysis Report NA/195, University of Dundee, December 1999.

[41] Roger Fletcher and Sven Leyffer. Nonlinear programming without a penalty function. *Math. Program.*, 91:239–269, 2000.

[42] Roger Fletcher and Sven Leyffer. Solving mathematical programs with complementary constraints as nonlinear programs. *Optim. Methods Software*, 19(1):15–40, 2004.

[43] Christodoulos A. Floudas. *Nonlinear and Mixed Integer Optimization: Fundamentals and Applications*, volume 37 of *Nonconvex Optimization and Its Applications*. Oxford University Press, New York, 1995.

[44] Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press/Brooks/Cole Publishing Company, 2nd edition, 2002.

[45] Armin Fügenschuh, Björn Geißler, Ralf Gollmer, Christine Hayn, René Henrion, Benjamin Hiller, Jesco Humpola, Thorsten Koch, Thomas Lehmann, Alexander Martin, Radoslava Mirkov, Antonio Morsi, Werner Römisch, Jessica Rövekamp, Lars Schewe, Martin Schmidt, Robert Schwarz, Rüdiger Schultz, Jonas Schweiger, Claudia Stangl, Marc C. Steinbach, and Bernhard M. Willert. Mathematical optimization for challenging network planning problems in unbundled liberalized gas markets. In preparation.

[46] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

[47] Björn Geißler. *Towards Globally Optimal Solutions for MINLPs by Discretization Techniques with Applications in Gas Network Optimization.* Ph. D. dissertation, University of Erlangen-Nuremberg, Germany, 2011.

[48] Gene H. Golub and Charles F. van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, USA, 2nd edition, 1989.

[49] Nicholas I. M. Gould, Dominique Orban, and Philippe L. Toint. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Softw.*, 29(4):373–394, December 2003.

[50] Ignacio E. Grossmann. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optim. Eng.*, 3(3):227–252, 2002. Special issue on mixed-integer programming and its applications to engineering.

[51] Martin Gugat. Boundary controllability between sub- and supercritical flow. *SIAM J. Control Optim.*, 42:1056–1070, 2003.

[52] Martin Gugat and Günter Leugering. Global boundary controllability of the de St. Venant equations between steady states. *Annales de l'Institut Henri Poincare (C) Non Linear Analysis*, 20(1):1 – 11, 2003.

[53] Martin Gugat, Günter Leugering, Klaus Schittkowski, and E. J. P. Georg Schmidt. Modelling, stabilization, and control of flow in networks of open channels. In Martin Grötschel, Sven O. Krumke, and Jörg Rambau, editors, *Online Optimization of Large Scale Systems*, pages 251–270. Springer, Berlin, 2001.

[54] Peter Hackländer. *Integrierte Betriebsplanung von Gasversorgungssystemen.* Ph. D. dissertation, Universität Wuppertal, 2002.

[55] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms I: Fundamentals*, volume 305 of *Grundlehren der mathematischen Wissenschaften.* Springer, 1993.

[56] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods*, volume 306 of *Grundlehren der mathematischen Wissenschaften.* Springer, 1993.

[57] Willi Hock and Klaus Schittkowski. *Test Examples for Nonlinear Programming Codes*, volume 187 of *Springer Lecture Notes in Economics and Mathematical Systems.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1981.

[58] The HSL Mathematical Software Library. *HSL MA27 Package Specification*, March 2003.

[59] X. M. Hu and Daniel Ralph. Convergence of a penalty method for mathematical programming with complementarity constraints. *J. Optim. Theory Appl.*, 123:365–390, 2004.

[60] Andrew Hunt and David Thomas. *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[61] Tomáš Jeníček. Steady-state optimization of gas transport. In SIMONE [114], pages 26–38.

[62] Tomáš Jeníček, Jaroslav Králik, J. Štěrba, Zdeněk Vostrý, and Jiři Záworka. Study to analyze the possibilities and features of an optimization system (optimum control system) to support the dispatching activities of Ruhrgas. Vertrauliche Dokumentation, LIWACOM Informationstechnik GmbH Essen, 1993.

[63] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[64] Napsu Karmitsa. NonSmooth Optimization (NSO) Software. `http://napsu.karmitsa.fi/nsosoftware/`.

[65] Donald La Verne Katz. *Handbook of natural gas engineering*. McGraw-Hill series in chemical engineering. McGraw-Hill, 1959.

[66] Leonid Genrikhovich Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[67] Thorsten Koch, Dagmar Bargmann, Mirko Ebbers, Armin Fügenschuh, Björn Geißler, Nina Geißler, Ralf Gollmer, Uwe Gotzes, Christine Hayn, Holger Heitsch, René Henrion, Benjamin Hiller, Jesco Humpola, Imke Joormann, Veronika Kühl, Thomas Lehmann, Hernan Leövey, Alexander Martin, Radoslava Mirkov, Andris Möller, Antonio Morsi, Djamal Oucherif, Antje Pelzer, Marc E. Pfetsch, Lars Schewe, Werner Römisch, Jessica Rövekamp, Martin Schmidt, Rüdiger Schultz, Robert Schwarz, Jonas Schweiger, Klaus Spreckelsen, Claudia Stangl, Marc C. Steinbach, Ansgar Steinkamp, Isabel Wegner-Specht, Bernhard M. Willert, and Stefan Vigerske. From simulation to optimization: Evaluating gas network capacities. In preparation.

[68] Korbinian Kraemer and Wolfgang Marquardt. Continuous reformulation of MINLP problems. In Moritz Diehl, Francois Glineur, Elias Jarlebring, and Wim Michiels, editors, *Recent Advances in Optimization and its Applications in Engineering*, pages 83–92. Springer Berlin Heidelberg, 2010.

[69] Jaroslav Králik. Compressor stations in SIMONE. In SIMONE [114], pages 93–117.

[70] Jaroslav Králik, Petr Stiegler, Zdeněk Vostrý, and Jiří Závorka. A universal dynamic simulation model of gas pipeline networks. *IEEE Trans. Syst., Man, Cybern.*, 14(4):597–606, 1984.

[71] Jaroslav Králik, Petr Stiegler, Zdeněk Vostrý, and Jiři Záworka. Modeling the dynamics of flow in gas pipelines. *IEEE Trans. Syst., Man, Cybern.*, SMC-14(4):586–596, 1984.

[72] Günter Leugering and E. J. P. Georg Schmidt. On the modelling and stabilization of flows in networks of open canals. *SIAM J. Control Optim.*, 41(1):164–180, 2002.

[73] Sven Leyffer. MacMPEC: AMPL collection of MPECs. `www.mcs.anl.gov/~leyffer/MacMPEC/`.

[74] Sven Leyffer, Gabriel López-Calva, and Jorge Nocedal. Interior methods for mathematical programs with complementarity constraints. *SIAM J. Optim.*, 17:52–77, 2004.

[75] Xinwei Liu and Jie Sun. A robust primal-dual interior-point algorithm for nonlinear programs. *SIAM J. Optim.*, 14(4):1163–1186, 2004.

[76] LIWACOM Informations GmbH and SIMONE Research Group s.r.o. *Gleichungen und Methoden*, 2004. Benutzerhandbuch.

[77] Zhi-Quan Luo, Jong-Shi Pang, and Daniel Ralph. *Mathematical programs with equilibrium constraints*. Cambridge University Press, 1996.

[78] Mikhail V. Lurie. *Modeling of Oil Product and Gas Pipeline Transportation*. Wiley-VCH, 2008.

[79] Harry M. Markowitz and Alan S. Manne. On the solution of discrete programming problems. *Econometrica*, 25(1):pp. 84–110, 1957.

[80] Alexander Martin, Björn Geißler, Claudia Hayn, Benjamin Hiller, Jesco Humpola, Thorsten Koch, Thomas Lehmann, Antonio Morsi, Marc Pfetsch, Lars Schewe, Martin Schmidt, Rüdiger Schultz, Robert Schwarz, Jonas Schweiger, Marc C. Steinbach, and Bernhard M. Willert. Optimierung Technischer Kapazitäten in Gasnetzen. In *Optimierung in der Energiewirtschaft*, volume 2157 of *VDI-Berichte*, pages 105–114, 2011.

[81] Alexander Martin, Debora Mahlke, and Susanne Moritz. A simulated annealing algorithm for transient optimization in gas networks. *Math. Methods Oper. Res.*, 66(1):99–115, 2007.

[82] Alexander Martin and Markus Möller. Cutting planes for the optimization of gas networks. In Bock et al. [12], pages 307–329.

[83] Alexander Martin, Markus Möller, and Susanne Moritz. Mixed integer models for the stationary case of gas network optimization. *Math. Program.*, 105(2-3, Ser. B):563–582, 2006.

[84] Abraham H. Maslow. *The psychology of science; a reconnaissance.* Harper & Row New York, 1st edition, 1966.

[85] Sanjay Mehrotra. On the implementation of a primal-dual interior point method. *SIAM J. Optim.*, 2(4):575–601, 1992.

[86] Bertrand Meyer. Applying "Design by Contract". *Computer*, 25(10):40–51, October 1992.

[87] R. R. Meyer. Mixed integer minimization models for piecewise-linear functions of a single variable. *Discrete Mathematics*, 16(2):163 – 171, 1976.

[88] Markus Möller. *Mixed Integer Models for the Optimisation of Gas Networks in the Stationary Case.* Ph. D. dissertation, Technische Universität Darmstadt, 2004.

[89] Susanne Moritz. *A Mixed Integer Approach for the Transient Case of Gas Network Optimization.* Ph. D. dissertation, Technische Universität Darmstadt, 2007.

[90] Jorge Nocedal, Andreas Wächter, and Richard A. Waltz. Adaptive barrier update strategies for nonlinear interior methods. *SIAM J. Optim.*, 19(4):1674–1693, 2009.

[91] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization.* Springer, Berlin, 2nd edition, 2006.

[92] Fred M. Odom and Gordon L. Muster. Tutorial on modeling of gas turbine driven centrifugal compressors. Technical Report 09A4, Pipeline Simulation Interest Group, 2009.

[93] A. Osiadacz. Nonlinear programming applied to the optimum control of a gas compressor station. *Int. J. Numer. Methods Eng.*, 15(9):1287–1301, 1980.

[94] I. Papay. OGIL Musz. Tud. Kozl., 1968.

[95] Marc E. Pfetsch, Armin Fügenschuh, Björn Geißler, Nina Geißler, Ralf Gollmer, Benjamin Hiller, Jesco Humpola, Thorsten Koch, Thomas Lehmann, Alexander Martin, Antonio Morsi, Jessica Rövekamp, Lars Schewe, Martin Schmidt, Rüdiger Schultz, Robert Schwarz, Jonas Schweiger, Claudia Stangl, Marc C. Steinbach, Stefan Vigerske, and Bernhard M. Willert. Validation of nominations in gas network optimization: Models, methods, and solutions. Submitted.

[96] Pipeline Simulation Interest Group. *PSIG 30th Annual Meeting, Denver, Colorado*, 1998.

[97] K. F. Pratt and J. G. Wilson. Optimization of the operation of gas transmission systems. *Transactions of the Institute of Measurement and Control*, 6(5):261–269, 1984.

[98] Sebastian Probst. Numerische Berechnung genetischer Rekombinationswahrscheinlichkeiten. Master's thesis, Leibniz Universität Hannover, 2012.

[99] Arvind U. Raghunathan and Lorenz T. Biegler. An interior point method for mathematical programs with complementarity constraints (MPCCs). *SIAM J. Optim.*, 15:720–750, March 2005.

[100] R. Raman and Ignacio E. Grossmann. Modeling and computational techniques for logic based integer programming. *Comput. Chem. Eng.*, 18(7):563–578, 1994.

[101] Roger Z. Ríos-Mercado, Seongbae Kim, and Andrew E. Boyd. Efficient operation of natural gas transmission systems: A network-based heuristic for cyclic structures. *Computers & Operations Research*, 33(8):2323–2351, 2006.

[102] Roger Z. Ríos-Mercado, Suming Wu, L. Ridgway Scott, and Andrew E. Boyd. A reduction technique for natural gas transmission network optimization problems. *Ann. Oper. Res.*, 117:217–234, 2002.

[103] Richard E. Rosenthal. *GAMS - A User's Guide*. GAMS Development Corporation, 2008.

[104] Andrzej Ruszczyński. *Nonlinear Optimization*. Princeton University Press, 2006.

[105] Jamal Saleh, editor. *Fluid Flow Handbook*. McGraw-Hill Handbooks. McGraw-Hill, 2002.

[106] Holger Scheel and Stefan Scholtes. Mathematical programs with complementarity constraints: Stationarity, optimality and sensitivity. *Math. Oper. Res.*, 25:1–22, 2000.

[107] Martin Schmidt, Marc C. Steinbach, and Bernhard M. Willert. High detail stationary optimization models for gas networks — Part 1: Model components. IfAM Preprint 94, Inst. of Applied Mathematics, Leibniz Universität Hannover, 2012. Submitted.

[108] Martin Schmidt, Marc C. Steinbach, and Bernhard M. Willert. A primal heuristic for nonsmooth mixed integer nonlinear optimization. IfAM Preprint 95, Inst. of Applied Mathematics, Leibniz Universität Hannover, 2012. Submitted.

[109] Martin Schmidt, Marc C. Steinbach, and Bernhard M. Willert. High detail stationary optimization models for gas networks — Part 2: Validation and results. In preparation, 2013.

[110] Stefan Scholtes. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM J. Optim.*, 11(4):918–936, 2001.

[111] Erwin Sekirnjak. Mixed integer optimization for gas transmission and distribution systems. Presentation manuscript, INFORMS-Meeting, Seattle, October 1998.

[112] Erwin Sekirnjak. Transiente Technische Optimierung (TTO-Prototyp). Vertrauliche Dokumentation, PSI AG, Berlin, 1999.

[113] Jeremy G. Siek, Lie-Quan Lee, and Andrew Lumsdaine. *The Boost Graph Library: User Guide and Reference Manual*. C++ In-Depth Series. Addison-Wesley, 2001.

[114] *Proceedings of 2nd International Workshop SIMONE on Innovative Approaches to Modeling and Optimal Control of Large Scale Pipeline Networks*, Prague, 1993.

[115] K. E. Starling and J. L. Savidge. *Compressibility factors of natural gas and other related hydrocarbon gases*. Transmission Measurement Committee report. American Gas Association, New York, 1992.

[116] Oliver Stein, Jan Oldenburg, and Wolfgang Marquardt. Continuous reformulations of discrete-continuous optimization problems. *Comput. Chem. Eng.*, 28(10):1951–1966, 2004.

[117] Marc C. Steinbach. *Fast Recursive SQP Methods for Large-Scale Optimal Control Problems*. Ph. D. dissertation, Universität Heidelberg, 1995.

[118] Marc C. Steinbach. Hierarchical sparsity in multistage convex stochastic programs. In Stanislav P. Uryasev and Panos M. Pardalos, editors, *Stochastic Optimization: Algorithms and Applications*, pages 385–410, Dordrecht, The Netherlands, 2001. Kluwer Academic Publishers.

[119] Marc C. Steinbach. Markowitz revisited: Mean-variance models in financial portfolio analysis. *SIAM Rev.*, 43(1):31–85, 2001.

[120] Marc C. Steinbach. Tree-sparse convex programs. *Math. Methods Oper. Res.*, 56(3):347–376, 2002.

[121] Marc C. Steinbach. On PDE solution in transient optimization of gas networks. *J. Comput. Appl. Math.*, 203(2):345–361, 2007.

[122] Alexander Stepanov and Meng Lee. The standard template library. Technical report, WG21/N0482, ISO Programming Language C++ Project, 1994.

[123] Defeng Sun and Liqun Qi. On NCP-functions. *Comput. Optim. Appl.*, 13(1-3):201–220, 1999. Computational Optimization - a Tribute to Olvi Mangasarian, Part II.

[124] Boost C++ Libraries. `http://www.boost.org/`, 1998-2013.

[125] Tom van der Hoeven. *Math in Gas and the art of linearization.* PhD thesis, Rijksuniversiteit Groningen, 2004.

[126] Dimitri van Heesch. Doxygen. `http://www.stack.nl/~dimitri/doxygen/`, 1997-2013.

[127] Robert J. Vanderbei. *LOQO User's Manual – Version 4.05.* Princeton University, School of Engineering and Applied Science, Department of Operations Research and Financial Engineering, Princeton, New Jersey, September 2006.

[128] Robert J. Vanderbei and David F. Shanno. An interior-point algorithm for nonconvex nonlinear programming. *Comput. Optim. Appl.*, 13:231–252, 1997.

[129] Juan Pablo Vielma and George L. Nemhauser. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. In Andrea Lodi, Alessandro Panconesi, and Giovanni Rinaldi, editors, *Integer Programming and Combinatorial Optimization*, volume 5035 of *Lecture Notes in Computer Science*, pages 199–213. Springer Berlin Heidelberg, 2008.

[130] Y. Villalobos-Morales, D. Cobos-Zaleta, H. J. Flores-Villarreal, Conrado Borraz-Sánchez, and Roger Z. Ríos-Mercado. On NLP and MINLP Formulations and Preprocessing for Fuel Cost Minimization of Natural Gas Transmission Networks. In *Proceedings of the 2003 NSF Design, Service and Manufacturing Grantees and Research Conference*, Birmingham, USA, January 2003.

[131] Zdeněk Vostrý. Transient optimization of gas transport and distribution. In SIMONE [114], pages 53–62.

[132] Andreas Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering.* PhD thesis, Carnegie Mellon University, 2002.

[133] Andreas Wächter and Lorenz T. Biegler. Line search filter methods for nonlinear programming: Motivation and global convergence. *SIAM J. on Optimization*, 16(1):1–31, May 2005.

[134] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1):25–57, 2006.

[135] Richard A. Waltz, José Luis Morales, Jorge Nocedal, and Dominique Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Math. Program.*, 107(3):391–408, July 2006.

[136] Andreas Weimann. *Modellierung und Simulation der Dynamik von Gasnetzen im Hinblick auf Gasnetzführung und Gasnetzüberwachung.* Ph. D. dissertation, Technische Universität München, 1978.

[137] T. R. Weymouth. Problems in Natural Gas Engineering. *Transactions of the American Society of Mechanical Engineers*, 34:185–231, 1912.

[138] Peter J. Wong and Robert E. Larson. Optimization of natural-gas pipeline systems via dynamic programming. *IEEE Trans. Automat. Contr.*, 13:475–481, October 1968.

[139] Peter J. Wong and Robert E. Larson. Optimization of tree-structured natural-gas transmission networks. *J. Math. Anal. Appl.*, 24:613–626, 1968.

[140] Shaun Wright, Mahesh Somani, and Chris Ditzel. Compressor station optimization. In PSIG [96]. Paper 9805.

[141] Suming Wu. *Steady-State Simulation and Fuel Cost Minimization of Gas Pipeline Networks.* ProQuest LLC, Ann Arbor, MI, 1998. Thesis (Ph.D.)-University of Houston.

[142] Suming Wu, Roger Z. Ríos-Mercado, Andrew E. Boyd, and L. Ridgway Scott. Model relaxations for the fuel cost minimization of steady-state gas pipeline networks. Technical Report TR-99-01, University of Chicago, January 1999.

[143] J. J. Ye and D. L. Zhu. Optimality conditions for bilevel programming problems. *Optimization*, 33:9–27, 1995.

[144] Edward Yourdon and Larry L. Constantine. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1st edition, 1979.

[145] Jiři Záworka. Project SIMONE—Achievements and running development. In SIMONE [114], pages 1–24.

Lebenslauf – Dipl.-Math. Martin Schmidt

Angaben zur Person

Geburtsdatum:	11.03.1983
Geburtsort:	Gifhorn, Deutschland
Adresse:	Große Barlinge 41, 30171 Hannover
E-Mail:	mschmidt@ifam.uni-hannover.de

Bildungsweg

1989–1993	Grundschule Groß Schwülper
1993–1995	Orientierungsstufe Groß Schwülper
1995–2002	Lessinggymnasium Braunschweig-Wenden
6/2002	Abitur am Lessinggymnasium Braunschweig-Wenden
10/2003–10/2008	Studium des Diplom-Studiengangs Mathematik mit der Studienrichtung Informatik und Anwendungsfach Biologie, Schwerpunkt Mikrobiologie, an der Leibniz Universität Hannover
04/2007	Studienarbeit: *Approximation der medialen Achse polygonal berandeter Gebiete in der euklidischen Ebene* an der Fakultät für Elektrotechnik und Informatik, Institut für Mensch-Maschine-Kommunikation, Fachgebiet Graphische Datenverarbeitung, Prof. Dr. Wolter
10/2008	Diplomarbeit: *Über Aspekte des Designs symmetrischer Verschlüsselungsverfahren mit einer Anwendung auf ein neues Kryptosystem* an der Fakultät für Mathematik und Physik, Institut für Algebra, Zahlentheorie und Diskrete Mathematik, Prof. Dr. Elsner
10/2008	Diplom im o. g. Studiengang
seit 4/2009	Promotionsstudent an der Leibniz Universität Hannover

Beruflicher Werdegang

8/2002–5/2003	Zivildienst im Krankenhaus Marienstift Braunschweig
4/2005–9/2005	Wissenschaftliche Hilfskraft bei Prof. Dr. Bessenrodt am Institut für Mathematik, Lehrgebiet Algebra und Zahlentheorie an der Universität Hannover
10/2007–12/2008	Wissenschaftliche Hilfskraft bei Prof. Dr. Elsner an der Fachhochschule für die Wirtschaft (FHDW) Hannover
11/2008–5/2010	Wissenschaftlicher Mitarbeiter am Konrad-Zuse-Zentrum für Informationstechnik Berlin, Abteilung Optimierung (Prof. Dr. Dr. h.c. mult. Grötschel)
01/2009–5/2010	Gastwissenschaftler am Institut für Angewandte Mathematik, AG Algorithmische Optimierung (Prof. Dr. Steinbach), der Leibniz Universität Hannover
seit 6/2010	Wissenschaftlicher Mitarbeiter am Institut für Angewandte Mathematik, AG Algorithmische Optimierung (Prof. Dr. Steinbach), der Leibniz Universität Hannover

Publikationen

1. *A Primal Heuristic for Nonsmooth Mixed Integer Nonlinear Optimization.* Mit M. C. Steinbach und B. M. Willert. Technical Report, IfAM Preprint 95, Inst. of Applied Mathematics, Leibniz Universität Hannover, 2012.

2. *Validation of Nominations in Gas Network Optimization: Models, Methods, and Solutions.* Mit M. E. Pfetsch, A. Fügenschuh, B. Geißler, N. Geißler, R. Gollmer, B. Hiller, J. Humpola, T. Koch, T. Lehmann, A. Martin, A. Morsi, J. Rövekamp, L. Schewe, R. Schultz, R. Schwarz, J. Schweiger, C. Stangl, M. C. Steinbach, S. Vigerske und B. M. Willert. ZIB Report 12-41, 2012.

3. *High detail stationary optimization models for gas networks — Part 1: Model components.* Mit M. C. Steinbach und B. M. Willert. Technical Report IfAM Preprint 94, Inst. of Applied Mathematics, Leibniz Universität Hannover, 2012.

4. *Optimierung Technischer Kapazitäten in Gasnetzen.* Mit A. Martin, B. Geißler, C. Hayn, B. Hiller, J. Humpola, T. Koch, T. Lehmann, A. Morsi, M. Pfetsch, L. Schewe, R. Schultz, R. Schwarz, J. Schweiger, M. C. Steinbach und B. M. Willert, . In: Optimierung in der Energiewirtschaft, VDI-Berichte 2157, 2011

5. *Using the Inhomogeneous Simultaneous Approximation Problem for Cryptographic Design.* Mit F. Armknecht and C. Elsner. In: A. Nitaj, D. Pointcheval (Eds.), Progress in Cryptology - AFRICACRYPT 2011, 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings. Lecture Notes in Computer Science Vol. 6737, Springer, 2011

6. *KronCrypt - A New Symmetric Cryptosystem Based on Kronecker's Approximation Theorem.* Mit C. Elsner. Cryptology ePrint Archive (http://eprint.iacr.org), Report 2009/416. 2009

7. *Über Aspekte des Designs symmetrischer Verschlüsselungsverfahren mit einer Anwendung auf ein neues Kryptosystem.* Forschungsberichte der FHDW Hannover (ISSN 1863-7043), Bericht Nr.: 02008/02, Dezember 2008