Adaptive Mixed-Integer Refinements for Solving Nonlinear Problems with Discrete Decisions

Adaptive gemischt-ganzzahlige Verfeinerungen zur Lösung nichtlinearer Probleme mit diskreten Entscheidungen

Der Naturwissenschaftlichen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

zur

Erlangung des Doktorgrades Dr. rer. nat.

vorgelegt von

Robert Burlacu
aus Iaşi

Als Dissertation genehmigt
von der Naturwissenschaftlichen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der mündlichen Prüfung:	26.09.2019
Vorsitzende/r der Promotionsorgans:	Prof. Dr. Georg Kreimer
Gutachter/in:	Prof. Dr. Alexander Martin
	Prof. Dr. Rüdiger Schultz

Acknowledgements

First an foremost, I am very thankful to my supervisor Prof. Dr. Alexander Martin for enabling me to pursue my Ph.D. in such an interesting field of mathematics. It was a pleasure to work in his group, which is surrounded by a very collegial, productive, and friendly atmosphere. His continuous support and advice over the years played a major role in the success of this thesis. The sympathy he showed in various situations and the conversations with him were very encouraging and motivated me constantly during my Ph.D. time.

I am also very grateful to Dr. Lars Schewe for always making time and for the numerous discussions of any depth on miscellaneous topics. His enthusiasm for mathematics was very enriching and inspiring.

I owe special thanks to Prof. Dr. Martin Schmidt for many discussions and help on various mathematical topics and for his well-structured way of writing a paper. Not only have I learned a lot from him, but I have also always enjoyed his relaxed and open-minded character.

I am particularly grateful to Prof. Dr. Frauke Liers for her reliable support as my mentor within the research project "SFB Transregio 154: Mathematical Modelling, Simulation and Optimization using the Example of Gas Networks"[1] supported by the German Research Foundation[2]. I thank her and Prof. Dr. Alexander Martin for creating such a cooperative and welcoming working environment at our chair.

My warmest gratitude goes to my colleagues at our chair. I cannot thank Dr. Mathias Sirvent enough for uncountable fruitful discussions on various optimization issues, for all his inspiring words, and his refreshing argentine mentality. I am very thankful to Manu Kapolke for all the mind-expanding philosophical discussions and for reviewing all my attempts to prove the simplex problem. Many thanks to Denis Aßmann, who always had an open ear for me and from whom I often borrowed his Python abilities. I am very grateful to Dr. Lena Hupp for the many exciting discussions about discrete optimization and for her enthusiasm, which has always transferred to me. At this point, I would also like to thank Dennis Adelhütte, Kevin-Martin Aigner, Denis Aßmann, Dr. Lena Hupp, Manu Kapolke, Prof. Dr. Martin Schmidt,

[1] http://trr154.fau.de
[2] http://www.dfg.de

Dr. Mathias Sirvent, Johannes Thürauf, Sebastian Tschuppik, and Dr. Dieter Weninger for proof-reading parts of this thesis. I am very grateful to all remaining colleagues at our chair for their mutual respect and entertaining coffee breaks. In addition, my thanks go to the experts for derivatives on the other side of the floor for the enjoyable time.

I would like to express my sincere thanks to my fellow Ph.D. students and the principal investigators within the SFB Transregio 154 project. The supportive and sociable team spirit was very enjoyable for me. In this context, I would also like to thank the German Research Foundation (DFG) for the corresponding financial support within subproject B07 of the SFB Transregio 154 project.

Special thanks go to Kevin-Martin Aigner for the pleasant working atmosphere and collaboration on Section 6.2 of this thesis. I would also like to express my sincere gratitude to Dr. Björn Geißler for his help with many algorithmic questions and his constructive cooperation on joint works. I am also very thankful for the joint works with many other co-authors within the SFB Transregio 154 project. Moreover, it was always a pleasure to collaborate with my colleagues from the computer science department. My thanks go in particular to Peter Wägemann, who came up with the idea of an interdisciplinary cooperation and whose enthusiastic attitude I have always greatly appreciated.

Finally, I wish to express my deepest gratitude to my friends, my girlfriend and my family. Thank you very much for your ongoing support and your strong belief in me.

Zusammenfassung

Viele Optimierungsprobleme der Wissenschaft und Technik werden anhand eines Systems nichtlinearer Restriktionen in Kombination mit diskreten Entscheidungen gelöst. Mathematisch lassen sich diese Probleme als gemischt-ganzzahlige nichtlineare Programme (MINLPs) modellieren, da diese sowohl nichtlineare Zusammenhänge als auch diskrete Entscheidungen darstellen können. Das Zusammenspiel von Ganzzahligkeit und Nichtlinearität stellt dabei eine große Herausforderung bei der Lösung dieser Probleme dar.

In dieser Arbeit behandeln wir eine Methode zur global optimalen Lösung von MINLPs durch Diskretisierung der auftretenden Nichtlinearitäten. Unser Ansatz erfordert nur kontinuierliche Nichtlinearitäten mit beschränktem Definitionsbereich und ist daher für eine Vielzahl von MINLP-Problemen geeignet. Die grundlegende Idee hierbei ist, ausgereifte und zuverlässige Technologie der gemischt-ganzzahligen Programmierung zur Lösung von MINLPs einzusetzen. Ähnlich wie das Lösen von gemischt-ganzzahligen linearen Programmen (MIPs) auf der Lösung linearer Programme, d.h. von Relaxierungen der MIPs, beruht, entwickeln wir einen Ansatz zur Lösung von MINLPs, der auf der Lösung von MIP-Relaxierungen beruht. Hierzu verwenden wir stückweise lineare Funktionen, um MIP-Relaxierungen des zugrunde liegenden MINLPs zu konstruieren. Der iterative Algorithmus konstruiert MIP-Relaxierungen, die abwechselnd verfeinert und gelöst werden, bis eine global optimale Lösung gefunden wird.

Die Verfeinerung der Nichtlinearitäten ist entscheidend für die Konvergenz und Korrektheit des Algorithmus. Aus diesem Grund untersuchen wir verschiedene Verfeinerungsstrategien mit dem Schwerpunkt auf deren Einbettung in unseren adaptiven MIP-basierten Ansatz. Wir liefern zunächst Konvergenzresultate für die vorgestellten Verfeinerungsmethoden. Darüber hinaus leiten wir erste Ergebnisse zur Größe einer MIP-Relaxierung her, die benötigt wird, um eine a priori gegebene Genauigkeit für die Nichtlinearitäten zu erreichen.

Abschließend veranschaulichen wir die Anwendbarkeit unseres Ansatzes in der Praxis durch numerische Ergebnisse für MINLPs, die mit den modernsten globalen MINLP-Lösern nur schwer zu lösen sind. Diese Probleme entstehen im Kontext der Gastransportoptimierung und der optimalen Lastflussberechnung. Sie kombinieren

nichtkonvexe Nichtlinearitäten, die spezifische physikalische Phänomene beschreiben, mit ganzzahligen Restriktionen, die diskrete Entscheidungen für schaltbare Elemente modellieren. Solche Elemente sind beispielsweise Kompressoren bei Gasnetzwerken oder Generatoreinheiten bei Stromnetzwerken. Anhand dieser MINLP-Instanzen zeigen wir den Mehrwert unseres Ansatzes gegenüber anderen globalen MINLP-Lösern auf.

Abstract

Many optimization problems in science and technology are subject to a system of nonlinear constraints combined with discrete decisions. Mathematically, these problems can be modeled as mixed-integer nonlinear programs (MINLPs), since they can represent both nonlinear correlations and discrete decisions. However, the interaction of integrality and nonlinearity poses a major challenge in solving these problems.

In this thesis, we propose a method for solving MINLPs to global optimality by discretization of the occurring nonlinearities. Our approach requires only continuous nonlinearities with bounded domains and is thus suitable for a wide range of MINLP problems. The emphasis is on using sophisticated and reliable mixed-integer linear programming technology to solve MINLPs. Similarly to the solution of mixed-integer linear programs (MIPs), which relies on solving linear programming relaxations, we develop a framework for solving MINLPs by MIP relaxations. To this end, we use piecewise linear functions to construct MIP relaxations of the underlying MINLP. An iterative algorithm constructs MIP relaxations that are subsequently refined and solved until a globally optimal solution is found.

The refinement of the nonlinearities is crucial for the outcome of the algorithm. For that reason, we study different refinement strategies with a focus on embedding them in our adaptive MIP-based framework. We prove convergence results for the presented refinement methods. In addition, we present first results on the size of an MIP relaxation that is required to achieve an a priori given accuracy for the nonlinearities.

Finally, we illustrate the practicalness of our approach by numerical results for MINLPs that are difficult to solve by state-of-the-art global MINLP solvers. These problems arise in the context of gas transport network optimization and optimal power flow. They combine non-convex nonlinearities that describe certain physical phenomena with integer restrictions that model discrete decisions for switchable elements. Such elements are, for instance, compressors in case of gas networks or generator units in case of power networks. On the basis of theses MINLP instances, we demonstrate the advantage of our approach over other global MINLP solvers.

Table of Contents

CHAPTER 1

Introduction

In many academic and real-world problems such as from engineering or economics, discrete decisions and nonlinear dependencies occur in interaction. The combination of these two aspects leads to the relatively general modeling class of mixed-integer nonlinear programs (MINLPs). The great challenge in solving MINLPs lies in the optimization over a mixed-integer feasible set while incorporating nonlinear functions at the same time.

In this thesis, we develop a solution approach that is suitable for a wide range of MINLP problems. The main idea is based on using piecewise linear functions to construct mixed-integer linear program (MIP) relaxations of the underlying MINLP. In order to find a global optimum of the given MINLP, an iterative algorithm constructs MIP relaxations of the MINLP that are then alternately refined and solved.

Our approach involves two different methodologies compared to standard approaches for solving MINLPs. First, while spatial branch-and-bound uses a single branch-and-bound tree, we solve multiple MIPs and therefore use multiple branch-and-bound trees. Second, in contrast to many methods that use only piecewise linear *approximations* to obtain approximate solutions to MINLPs, we extend such approximations to piecewise linear *relaxations* in our subproblems. This allows us both to obtain globally optimal solutions for feasible MINLPs and to prove infeasibility otherwise.

We investigate various refinement strategies for the MIP relaxations with regard to their incorporation into our adaptive MIP-based framework. We show convergence results for these refinement procedures, which are crucial for both the size of the MIP relaxations and the overall convergence of our algorithm. Moreover, we derive tight bounds on the size of the MIP relaxations for an a priori given accuracy.

Finally, we apply our approach to MINLPs that arise in the context of gas transport network optimization and optimal power flow. We solve both stationary and transient gas network optimization problems. These problems combine non-convex nonlinearities and discrete aspects. On the one hand, modeling the gas physics derived from the Euler equations results in a coupled system of nonlinear equations. On the other hand, switching active elements such as valves and compressors involves discrete

decisions. In the transient case, we furthermore introduce a new model for the gas flow in pipelines. The optimal power flow problems also involve non-convex nonlinear functions and discrete decisions. Here, the nonlinearities originate from an alternate current power flow model. The integer part models the switching of the generator units of the power network. The concluding numerical results demonstrate that our MIP-based approach outperforms state-of-the-art global MINLP solvers on several difficult MINLP problems.

Incorporation of Joint Work with other Authors

Some results of this thesis have been achieved in collaboration with other researchers. In particular, the author of this thesis was supervised by Prof. Dr. Alexander Martin and Dr. Lars Schewe within subproject B07 of the research project "SFB Transregio 154: Mathematical Modelling, Simulation and Optimization using the Example of Gas Networks"[1] supported by the German Research Foundation[2].

The results of this thesis are to a large extent based on the following two publications:

> R. Burlacu, H. Egger, M. Groß, A. Martin, M. E. Pfetsch, L. Schewe, M. Sirvent, and M. Skutella (2018). "Maximizing the storage capacity of gas networks: a global MINLP approach". In: *Optimization and Engineering*, pp. 1–31. DOI: 10.1007/s11081-018-9414-5.

> R. Burlacu, B. Geißler, and L. Schewe (2019). "Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes". In: *Optimization Methods and Software*, pp. 1–28. DOI: 10.1080/10556788.2018.1556661.

The author of this thesis made significant contributions to both publications. In the following, at the beginning of each chapter, we outline in detail the results achieved by the author.

The Outline of this Thesis

This thesis is structured as follows.

Chapter 2 gives a formal description of the MINLP problems that we consider in this thesis. It provides a literature survey and the most important approaches in the field of MINLP. Moreover, we conduct an experiment, in which MIPs are reformulated as nonlinear programs (NLPs) by replacing all binary variables using

[1] http://trr154.fau.de
[2] http://www.dfg.de

the constraint $x^2 - x = 0$. These NLPs are subsequently solved by state-of-the-art global NLP solvers.

In Chapter 3, we develop our MIP-based approach for solving MINLPs. We prove its convergence and investigate different procedures for the refinement of the relaxations of the nonlinearities. Additionally, we give tight bounds on the size of the MIP relaxations that are required to achieve an a priori given accuracy.

In Chapter 4, we derive MINLP models for stationary and transient gas network optimization problems. In the transient case, we introduce a new gas flow model that is based on the stationary model.

Chapter 5 describes the MINLP model for optimal power flow problems with additional switching of the generator units.

In Chapter 6, we compare the practical performance of our MIP-based approach with state-of-the-art global MINLP solvers. To this end, we use the previously introduced MINLP problems from the field of gas network optimization and optimal power flow as benchmark instances.

We finally summarize and conclude this thesis in Chapter 7.

CHAPTER 2

Mixed-Integer Nonlinear Programming

In this chapter, we present a formal description of the problems that we address in this thesis and give some theoretical background to the methods we develop in the subsequent chapters. In addition, we provide a literature overview and describe the most important approaches in the field of mixed-integer nonlinear programming.

Throughout this work, with $[n] \coloneqq \{1, \dots, n\}$ for $n \in \mathbb{N}$, we consider an MINLP problem as an optimization problem of the following type:

$$\begin{aligned} \min_{x} \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b, \\ & f_i(x) \leq 0 \qquad \text{for all } i \in [k], \\ & l \leq x \leq u, \\ & x \in \mathbb{R}^q \times \mathbb{Z}^p, \end{aligned} \tag{P}$$

where k, q, $p \in \mathbb{N}$. First, $Ax \leq b$ represents all linear constraints, while the nonlinear constraints are described using the continuous nonlinear real-valued functions $f_i \colon \mathbb{R}^{q+p} \to \mathbb{R}$ for $i = 1, \dots, k$. The variables x are bounded from below and above by the vectors $l, u \in \mathbb{R}^{q+p}$. Furthermore, we denote by $\mathcal{F}$ the set of all nonlinear functions $f_i(x)$. Let $D_f \subset \mathbb{R}^{q+p}$ be the domain of a nonlinear function $f \in \mathcal{F}$. Since each variable in (P) has lower and upper bounds, the domain D_f is a compact set. We consider it to be a d-dimensional box with its edges parallel to the coordinate axes, while $d \leq q + p$. Equality constraints, i.e., constraints of type $f_i(x) = 0$, are implicitly contained in (P) by adding the constraints $f_i(x) \leq 0$ and $-f_i(x) \leq 0$. Moreover, we are not restricted to a linear objective function $c^\top x$, because we can include any nonlinear objective function $f \colon \mathbb{R}^{q+p} \to \mathbb{R}$ simply by substituting $f(x)$ with a variable $y \in \mathbb{R}$ and adding $f(x) \leq y$ as a constraint to the MINLP problem. Due to $\max c^\top x = -\min -c^\top x$, any maximization problem can be transformed to a minimization problem. Thus, (P) represents a general formal description of MINLP problems.

If there are no integer variables in the optimization problem (P), we refer to it as a nonlinear program (NLP). If all functions in (P) are linear, we call the resulting

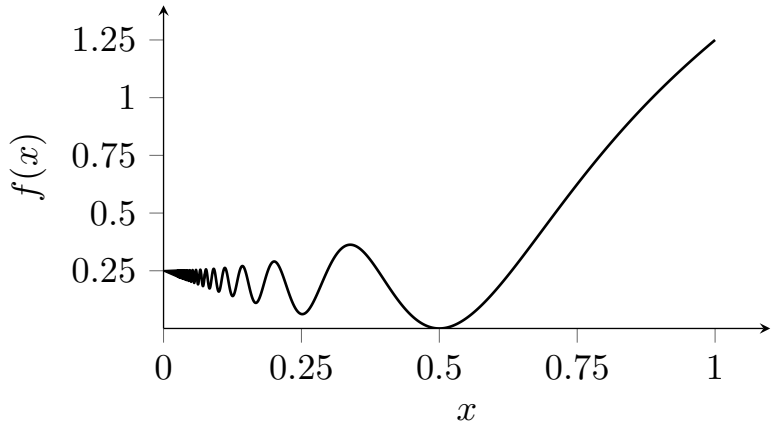


FIGURE 2.1. An example of a continuous nonlinear function $f(x)$ defined by (1) with infinitely many only local optima.

optimization problem a mixed-integer program (MIP). An MIP, where additionally all variables are allowed to attain non-integer values is called a linear program (LP).

In general, MINLPs are very hard to solve. In order to distinguish between "easy" and "hard" problems, we categorize MINLPs according to their nonlinear functions.

DEFINITION 2.1. We call the optimization problem (P) a convex MINLP, if all functions $f \in \mathcal{F}$ in (P) are convex. If any function $f \in \mathcal{F}$ is non-convex, then we refer to (P) as a non-convex MINLP. This transfers to NLPs equivalently.

Note that apart from the terminology in Definition 2.1, the feasible set of (P) is always non-convex if integer variables or nonlinear equality constraints are involved. In terms of global optimization, convex MINLPs are usually much easier to solve than non-convex MINLPs. This is mainly due to the fact that for convex optimization problems any local optimum is also a global optimum. In non-convex global optimization, however, it is often the case that solution algorithms either get stuck in local optima or they are unable to distinguishing between local and global optima. This is aggravated by non-convex functions with infinitely many only local optima; see Example 2.2.

EXAMPLE 2.2. *Consider the function* $f\colon [0,1] \to \mathbb{R}$ *defined as*

$$f(x) \coloneqq \begin{cases} x \sin^2\left(0.5\pi x^{-1}\right) + (x-0.5)^2 & \text{if } x \in (0,1], \\ 0.25 & \text{if } x = 0; \end{cases} \tag{1}$$

see Figure 2.1 for an illustration. The global minimum of $f(x)$ *on* $[0,1]$ *is attained at* $x = 0.5$*. Furthermore,* $f(x)$ *is continuous on the compact box domain* $[0,1]$ *and therefore fulfills the conditions of the nonlinear functions in* (P)*. Since* $\lim_{x\to 0} x^{-1} = \infty$*,*

there are infinitely many only local minima (and also maxima) of $f(x)$ *on* $[0, \epsilon]$ *for any* $\epsilon > 0$.

Drawing a line between "easy" and "hard" computational problems in a general and reasonable manner naturally leads to their categorization with respect to their intrinsic difficulty. Computational complexity theory attempts to undertake this categorization in a formally coherent way. We assume that the reader is familiar with the two most famous complexity classes P and NP, as well as the terms NP-hardness and NP-completeness. For further details we refer to the comprehensive overview by Garey and Johnson (1979).

At this point, we would like to point out that there are actually decision problems that are not in NP, e.g., the halting problem; see Goldreich (2010) for more details. Moreover, the decision problems corresponding to MINLPs with unbounded feasible sets are not decidable in general; see Jeroslow (1973). This result implies a definition of MINLPs as in (P), since it does not apply to MINLPs with compact feasible sets.

This chapter is structured as follows. Section 2.1 provides a description of the methods for LPs and MIPs contained in modern solvers. Afterward, we give a brief overview of approaches for solving convex MINLPs in Section 2.2 and non-convex MINLPs in Section 2.3. Typically, such solution approaches are based on procedures using search trees and can therefore be divided into *multi-tree* and *single-tree* methods; see Section 2.2 and Section 2.3 for more details. We conclude this chapter in Section 2.3 with an experiment, in which MIPs are reformulated as nonlinear programs (NLPs) by replacing all binary variables using the constraint $x^2 - x = 0$. We refer to Burer and Letchford (2012) and Belotti et al. (2013) for more detailed overviews and general concepts.

This chapter primarily summarizes known facts and methods from the literature. The contribution of the author of this thesis is the experiment at the end of the chapter, where we reformulate MIPs as NLPs using the nonlinear constraint $x^2 - x = 0$ and subsequently solve the NLPs.

2.1. Linear and Mixed-Integer Programming

Over the last decades, a tremendous progress has been made in the field of linear and mixed-integer programming. Solvers are now capable of solving huge LPs and even MIPs arising from practical applications. Achterberg and Wunderling (2013), R. E. Bixby (2002, 2012), and R. E. Bixby et al. (2004) give a history-charged overview on computations of LPs and MIPs.

2.1.1. Linear Programming. Linear programming started to attract public attention in 1947 when George Dantzig proposed his simplex algorithm to solve LPs.

Down to the present day, the simplex algorithm is used in many state-of-the-art solvers. The performance, however, has drastically improved. One of the first (in that time large-scale) LPs solved by the simplex method consisted of nine equations and 77 variables. It is reported by Dantzig (1963) that with the aid of hand-operated desk calculators, 120 man-days were needed to obtain a solution to this problem in 1948. Five years later, one of the earlier simplex implementations solved the problem in 2 minutes on an IBM 701 machine while printing each iteration required three quarters of the total run time. More than 60 years later, modern LP solvers are capable of solving problems modeling real-world applications with several million constraints and variables without difficulty. This is due both to machine and, to a significant degree, algorithmic improvements. These two factors multiply in addition, which leads to this remarkable increase in performance.

The simplex method, albeit its excellent performance in practice, is of exponential run time in worst-case; see Klee and Minty (1972). Nevertheless, Borgwardt (1987) showed in a probabilistic analysis that the *shadow-vertex* variant of the simplex algorithm is in average of polynomial run time. It was not until 1979 that a polynomial time algorithm appeared. Khachiyan (1979) proposed the ellipsoid method for solving LPs and proved its polynomial run time. Unfortunately, this method is only of theoretical value so far, as all implementations have failed in practice due to numerical instability. Another polynomial time approach is the interior point algorithm introduced by Karmarkar (1984). This method has a high performance in practice and is most widely integrated in modern LP solvers. To put it more generally, it is successfully applied even within the scope of convex NLPs.

There is a variety of LP software nowadays. Most commonly utilized state-of-the-art solvers are Gurobi, Cplex, and Xpress; see Gurobi (2019), Cplex (2019), and Xpress (2019), respectively. A performant open-source alternative is SoPlex; see Gleixner et al. (2012) and Wunderling (1996). All this solvers essentially rely on the use of both simplex methods and interior point algorithms.

2.1.2. Mixed-Integer Programming. Many methods in mixed-integer programming are built upon linear programming. Therefore, reliable and fast LP solvers establish a strong basis for a sound MIP code.

The first algorithms for solving MIPs appeared in the mid to late 1950s. Gomory (1958) proposed a cutting-plane method, which alternately solves an LP relaxation of the MIP and adds constraints that cut off non-integer solutions until the LP relaxation delivers an integer solution. Secondly, a branch-and-bound (BB) approach is given in Dakin (1965) and Land and Doig (1960) including theoretical ground work of Markowitz and Manne (1957a). It consists of a procedure for splitting the feasible set of the problem and a mechanism determining an objective bound

for all subproblems by solving an LP. Both techniques are then repeatedly utilized until a global optimal solutions is found. We point out that apart from mixed-integer programming, BB is an essential method for many (global) optimization problems. Combining the enumerative approach of BB with cutting-plane algorithms results in the highly performant branch-and-cut, which is commonly integrated in today's MIP solvers.

Another widely used method for solving MIPs is based on the reformulation and subsequent decomposition of the problem into a master and a subproblem. Thereafter, an iterative procedure alternately solves both the master and the subproblem. In each iteration, at first, the master problem is solved with either fewer variables or fewer constraints. The subproblem then verifies whether the solution of the master problem is optimal and provides the master problem with variables or constraints, respectively, in the event that optimality is not attained. From this perspective, all decomposition methods attempt to tackle an MIP by solving a series of smaller-sized MIPs. Two of the most famous approaches are the Dantzig-Wolfe decomposition as in Dantzig and Wolfe (1960) and Benders' decomposition as in Benders (1962). The former adds variables to the master problem while constraints are inserted in the latter.

The decomposition methodology is, to some extent, motivated by the NP-hardness of MIPs, since with linearly increasing size of the problem, the run time for solving MIPs grows exponentially. More precisely, Karp (1972) showed that binary integer programs, i.e., MIPs containing only binary (integer) variables, are NP-hard. Therefore, MIPs are NP-hard in general.

Finally, we mention two powerful tools implemented in all modern MIP solvers: preprocessing and warm-starting. Sophisticated preprocessing reduces an MIP to a significantly smaller-sized problem and is even able to deliver (optimal) solutions. Starting with Brearley et al. (1975) and Savelsbergh (1994), an enormous number of different techniques have been developed within the last four decades; see for instance the works by Achterberg et al. (2016) and R. Bixby and Rothberg (2007); Gamrath et al. (2015). Preprocessing is now a vital part of mixed-integer programming. On the other hand, warm-starting speeds up the solution process itself. We recall that many MIP methods are basically based on successively solving LPs, many of which arise from small modifications of pre-existing LPs. Due to linear program duality, the optimal solution of the pre-existing LP is still a feasible solution for the modified LP or its dual LP and often times even near to optimal. In this way, warm-starting drastically reduces the number of iterations of embedded LP algorithms; see Ladányi et al. (2001) for more details.

Exploiting problem-specific structures led to an advantageous combination of the aforementioned MIP approaches. In this regard, plenty of research has been carried

out, beginning with Dantzig et al. (1954) and continued by many others, e.g., Grötschel and Padberg (1979a,b), Crowder et al. (1983), and Van Roy and Wolsey (1987). At the present day, we have mature MIP solvers that are both in control of enumeration and highly numerically stable. We are therefore very confident to develop an MINLP method that is essentially based on MIP technology.

Since solving LPs is fundamental for MIPs, we naturally have Gurobi, Cplex, Xpress, like in the case of LPs, and the performant open-source alternative SCIP Maher et al. (2017), as state-of-the-art solvers for MIPs.

2.2. Convex Mixed-Integer Nonlinear Programming

Convex MINLPs (as in Definition 2.1) are typically solved by either addressing the MIP and NLP parts of the problem separately or by employing efficient methodologies from the field of MIPs. We draw a distinction between single-tree approaches, where a single BB tree is used, and multi-tree approaches, where multiple BB trees are used, and present the most important ones.

One single-tree method is the NLP-based BB mechanism that works analogously to the MIP case but relies on solving convex NLPs instead of LPs; see Gupta and Ravindran (1985). Again, the method can be improved by adding cutting planes, as for example in Stubbs and Mehrotra (2002). Moreover, even warm-starting is available, e.g., as in Mahajan et al. (2012). Therein, the authors approximate the underlying convex NLPs by convex quadratic programs (QPs), i.e., MINLPs containing only one nonlinear and quadratic function in the set $\mathcal{F}$. Their approach converges to the optimal solution of a given MINLP and uses warm-starting.

A common multi-tree method is outer approximation, which was proposed by Duran and Grossmann (1986) and further developed by Fletcher and Leyffer (1994). The idea here is to solve an alternating series of MIP relaxations of an MINLP in a master problem and NLPs obtained by fixing integer variables in a subproblem. The MIP relaxations are set up by omitting all nonlinearities of the MINLP and successively adding linear cuts constructed from the solution of the subproblem. At the same time, the integer fixation in the subproblem is derived from the solution of the master problem.

Another multi-tree approach in the same setting of master problem and subproblem is the generalized Benders' decomposition, which is an extension of Benders' decomposition for MIPs; see Geoffrion (1972). In fact, it was one of the first methods in the context of convex MINLPs.

As a last multi-tree approach, we mention the extended cutting plane method by Westerlund and Pettersson (1995), which is closely related to outer approximation. Here, instead of solving convex NLPs in the subproblem to obtain linear cuts for the

master problem, the cuts are deduced directly from the solution of the master problem. Unsurprisingly, this procedure leads to weaker cuts in general. However, in case of large convex MINLPs with comparatively few nonlinearities, the extended cutting plane method delivers fairly good results; see again Westerlund and Pettersson (1995).

To come full circle, we present a second single-tree strategy: the LP/NLP-based BB by Quesada and Grossmann (1992). Roughly speaking, it is a performant embedding of outer approximation into a single-tree BB framework. LP/NLP-based BB omits to repeatedly solve the MIP relaxations in the master problems to optimality. Instead, the approach initially solves an LP relaxation of the MIP master problem while branching on integer variables leads to integrality. Whenever a feasible solution is found in the BB tree, the algorithm adds linear cuts analogously to outer approximation.

As previously hinted, the interior point method, which is of polynomial run time, is implemented in many state-of-the-art solvers for solving convex NLPs. Nonetheless, convex (and non-convex) MINLPs are NP-hard in general as MIPs, a special case of MINLPs, are already NP-hard.

There is a considerable number of solvers for convex MINLPs. We only present two of the most commonly used ones and point as before to Belotti et al. (2013) and Bussieck and Vigerske (2010) for a more comprehensive survey. First, α-ECP is a solver for convex MINLPs using the extended cutting plane method; see Westerlund and Lundqvist (2001). An open-source alternative is BONMIN, which integrates the NLP-based BB, LP/NLP-based BB, and outer approximation methodologies; see Bonami et al. (2008). Finally, we indicate that any solver for general (and possibly non-convex) MINLPs is naturally also a solver for convex MINLPs.

2.3. Non-Convex Mixed-Integer Nonlinear Programming

When modeling an optimization problem as an MINLP, non-convexities can occur rather quickly. For example, a nonlinear equality constraint alone already results in a non-convex feasible set. Non-convex MINLPs are usually tackled by some sort of relaxation of the (non-convex) nonlinearities combined with a refinement mechanism. We first present the most important relaxation methods and afterward the refinement approaches.

Finding a convex relaxation of a nonlinear function is nontrivial in general; see Tawarmalani and Sahinidis (2002) for an extensive survey on this topic. For a wide range of practically interesting MINLPs, however, the nonlinear functions are factorable, i.e., they can be represented by a recursive combination of elementary operators contained in the set

$$\mathcal{E} = \{+, \times, /, \hat{}, \sin, \cos, \exp, \log, |\cdot|\};$$

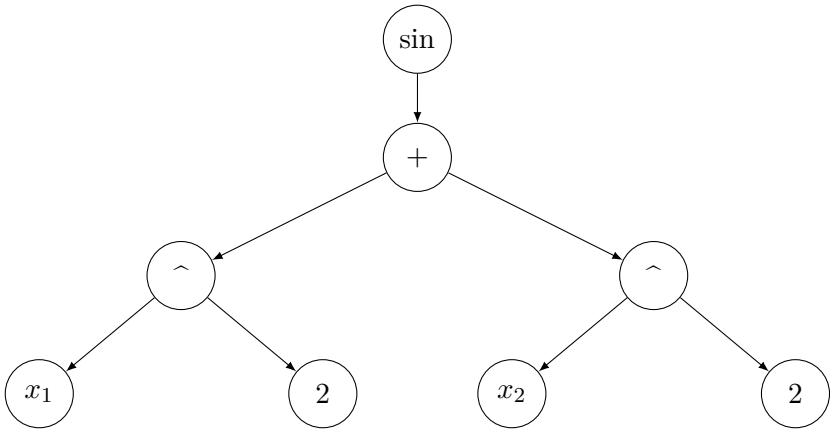


FIGURE 2.2. The expression tree of $f(x_1, x_2) = \sin(x_1^2 + x_2^2)$.

see Belotti et al. (2010). An illustration of such a representation by a so-called expression tree is given in Example 2.3.

EXAMPLE 2.3. *Consider the optimization problem*

$$\begin{aligned} \max_{x_1,x_2} \quad & x_1 + 0.5\,x_2 \\ \text{s.t.} \quad & \sin(x_1^2 + x_2^2) \leq 0.5, \\ & x_1 + x_2 \leq 1, \\ & 0 \leq x_1, x_2 \leq 1, \\ & x_1, x_2 \in \mathbb{R}. \end{aligned} \tag{S}$$

Let $f(x_1, x_2) = \sin(x_1^2 + x_2^2)$. Using the representation given by the expression tree as in Figure 2.2, we can write the function f as

$$f(x_1, x_2) = f_1(f_2(f_3(x_1), f_4(x_2))),$$

where

$$f_1(x) = \sin(x), \quad f_2(x_1, x_2) = x_1 + x_2, \quad \text{and} \quad f_3(x) = f_4(x) = x^2.$$

Introducing new variables x_3, x_4, and x_5, we can reformulate (S) *as*

$$\begin{aligned} \max_{x_1,x_2} \quad & x_1 + 0.5\,x_2 \\ \text{s.t.} \quad & \sin(x_5) \leq 0.5, \\ & x_5 = x_3 + x_4, \\ & x_3 = x_1^2, \\ & x_4 = x_2^2, \end{aligned} \tag{S$'$}$$

$$
\begin{aligned}
& x_1 + x_2 \leq 1, \\
& 0 \leq x_1, x_2, x_3, x_4 \leq 1, \\
& 0 \leq x_5 \leq 2, \\
& x_1, x_2, x_3, x_4, x_5 \in \mathbb{R}.
\end{aligned}
$$

The reformulated problem (S′) *is equivalent to problem* (S)*, but contains only univariate nonlinearities unlike* (S)*. To put it another way, we lift the problem to a higher dimension to be able to solve it more easily. After computing a solution, we omit the additionally introduced variables, i.e., we project the solution back onto the original variable space.*

Consequently, an MINLP with factorable nonlinear functions can be reformulated to contain only univariate nonlinearities derived from $\mathcal{E}$ and, if necessary, the coupling bivariate function $f(x_1, x_2) = x_1 x_2$. The benefit of this reduction is that (linear) convex relaxations of the nonlinearities in the reformulated MINLP are well studied in the literature, e.g., Liberti and Pantelides (2003) for monomials and Al-Khayyal and Falk (1983) and McCormick (1976) in case of $f(x_1, x_2) = x_1 x_2$. Although this reformulation is theoretically applicable to a wide range of MINLPs, it may not always be a good approach, since the relaxations that are obtained can be weak compared to the tightest possible ones. Moreover, this method excludes functions whose analytic expressions are not at hand, for example, integral functions with unknown primitive integrals, such as the logarithmic integral function $\int_0^x \mathrm{d}x / \ln x$, and simulation-based black-box functions. The latter arise for instance in the context of the design of nanophotonic devices; see Maria et al. (2009).

A more generic relaxation is obtained by the α-convexification as in Adjiman and Floudas (1996) and Androulakis et al. (1995), where the nonlinear functions only have to be twice differential instead of factorable. Therein, the authors propose a convexification of a non-convex function by adding an α-weighted sum of convex quadratic terms. With proper choice of the parameter α depending on the minimum eigenvalue of the non-convex function's Hessian, the resulting function is convex. Since computing minimum eigenvalues itself amounts to solving non-convex optimization problems, the authors further determine lower bounds on the minimum eigenvalues with the help of interval Hessians, which results in convex underestimators of the nonlinearities.

A third approach providing relaxations of non-convex functions is based on piecewise linearizations. This work fundamentally relies on using piecewise linear functions in order to obtain MIP relaxations of an MINLP. We therefore point to Chapter 3 for more details.

Endowed with these relaxation procedures, non-convex MINLPs are most commonly solved by spatial BB; see E. M. Smith and Pantelides (1997) and Horst and Tuy (1996). Initially, this method constructs linear or nonlinear convex relaxations of all non-convex functions. After additional relaxation of integrality, the resulting LP or convex NLP is solved. In the event that the solution of the relaxed optimization problem is infeasible for the MINLP, branching is performed. Accordingly, all subproblems that arise in spatial BB are either LPs or convex NLPs. In contrast to BB for MIPs, however, spatial BB branches on both continuous and integer variables. A great number of branching strategies in the context of MIPs are extended to MINLPs, e.g., strong branching, pseudocost branching, and reliability branching; see the works by Achterberg et al. (2005) and Belotti et al. (2009). With increasingly tight relaxations, the approach eventually converges to the optimal solution of the non-convex MINLP. Moreover, we point out that spatial BB is a single-tree method.

Another vital aspect of a performant spatial BB algorithm is bound tightening, since tight bounds are crucial for tight relaxations of the nonlinearities. The most popular bound reduction procedures are feasibility-based bound tightening (FBBT) and optimality-based bound tightening (OBBT); see Ryoo and Sahinidis (1995, 1996) and again Belotti et al. (2009). The main idea of FBBT is to reduce the bounds of variables that are directly coupled via constraints. As the bound of one of these variables is changed, the bounds of the other ones are tightened accordingly. This technique propagates throughout the variables of the optimization problem until no bound is reduced any further. On the other hand, OBBT tightens the bounds using LP relaxations of the MINLP. With n variables of some LP relaxation, OBBT solves at most $2n$ LPs minimizing and maximizing each of the n variables while considering the same feasible set as the LP relaxation. This technique provides fairly tight bounds in practice. In return, it is very costly. Generally, FBBT delivers weaker bounds than OBBT, although it can be applied efficiently at almost any spatial BB node. OBBT, however, is typically only performed at the root node or at nodes of small depth.

Over the last two decades, a large number of spatial BB implementations have emerged. Baron is a powerful MINLP solver requiring factorable nonlinearities; see Baron (2019). After decomposing all nonlinear functions as described previously, linear outer approximations of the known convex relaxations are solved at each node of the BB search tree. Another MINLP solver, which relies on α-convexifications, is α-BB; see α-BB (2019). In addition to convex underestimators for general twice differential functions, tight convex envelopes are available for certain commonly occurring, non-convex functions. Two open-source alternatives are again SCIP and Couenne; see Couenne (2019). Both solvers pursue a similar approach as Baron.

Finally, we bring to attention that categorizing MINLPs solely by convexity and

non-convexity may not be sufficient to capture the complexity of MINLPs that are hard to solve. For instance, we can reformulate any MIP as a non-convex NLP containing only one-dimensional nonlinearities. In particular, any binary variable x can be reduced to the constraint $x^2 - x = 0$, or more generally, any integer variable x to $\sin(\pi x) = 0$ with continuous x. In conclusion, non-convex NLPs with nonlinearities of type $x^2 - x = 0$ alone are in theory as hard as binary MIPs and therefore already NP-hard. These NLPs, however, are arguably one of the most modest non-convex NLPs.

One could even go so far as to formulate each MIP as an NLP and then solve the NLP instead of the MIP. In the following, we perform this experiment on the basis of the MIPLIB 2017; see MIPLIB (2018). We restrict ourselves to those instances of the benchmark set that have only binary variables as integer variables. In total, 164 instances are considered; see Table A.1 for an overview. We point out that 7 of these are infeasible. Each instance is solved on a cluster node with two Xeon 5650 "Westmere" chips (12 cores + SMT) running at 2.66 GHz with 12 MB Shared Cache per chip and 24 GB of RAM. The maximal total run time is 4 h. In order to neglect the impact of highly optimized multi-core implementations, we perform each computation on a single core. In this way, we get a slightly sharper comparison of the MIP and NLP specific methodologies. Furthermore, we utilize Gurobi 8.0.1 and SCIP 5.0 (denoted by SCIP-MIP) to solve the instances as MIPs and Baron 18.5.8 and again SCIP 5.0 (as NLP solver and denoted by SCIP-NLP) to solve the instances as NLPs, all within GAMS 25.1.2; see GAMS (2018).

We compare both relative optimality gaps and run times using performance profiles as proposed by Dolan and Moré (2002). With z as the primal bound corresponding to the best found feasible solution and d as dual bound, which is a bound on the best possible objective value, the relative optimality gap is defined as $|z - d|/|z|$. Let $g_{p,s}$ be the best gap obtained by solver s for problem p after a certain time limit. With the performance ratio $r_{p,s} = g_{p,s} / \min_s g_{p,s}$, the performance profile $\rho_s(\tau)$ is the percentage of problems solved by approach s such that the ratios $r_{p,s}$ are within a factor $\tau \in \mathbb{R}$ of the best possible ratios. Equivalent to this, we obtain the performance profile for the run times if we consider the run time $t_{p,s}$ that the solver s needs to solve problem p and form the ratios $r_{p,s} = t_{p,s} / \min_s t_{p,s}$. All performance profiles in this thesis are generated with the help of Perprof-py; see Siqueira et al. (2016).

First we look at the relative optimality gaps. An instance is considered to be solved as soon as the gap is less than 0.0001, which is the default value of Gurobi for instance. For the performance profile we set $g_{p,s} = 0.0001$ for solved instances to prevent a division by zero. An infeasible instance is also considered as solved if the infeasibility is detected. As we can see in Figure 2.3, the gaps obtained by

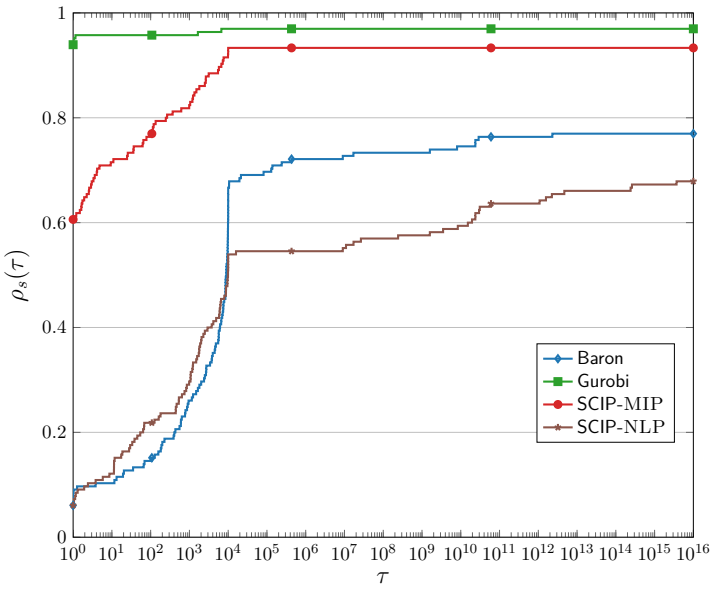


FIGURE 2.3. Performance profile for Gurobi, SCIP-MIP, Baron, and SCIP-NLP comparing relative optimality gaps obtained after a total run time limit of 4 h per instance.

solving the problems as MIPs are clearly tighter than the ones obtained by solving the problems as NLPs. Overall, for almost all problems, the gaps that are obtained by Gurobi are the tightest of all gaps. The optimality gaps provided by SCIP-MIP are in around 60 % of all cases the tightest. In addition, Gurobi is able to find a feasible solution for every feasible instance and SCIP-MIP for almost every feasible instance, with Gurobi generally finding solutions of higher quality. Infeasibility is detected by both Gurobi and SCIP-MIP for three of the 7 instances that are infeasible, while the two NLP solvers do not detect the infeasibility for any of the cases. In general, Baron and SCIP-NLP perform significantly worse than Gurobi and SCIP-MIP. In direct comparison, however, the two NLP solver are of similar quality. In about 10 % of all cases, both solvers are able to find solutions with optimality gaps that are as tight as the tightest. Baron, however, computes feasible solutions for 77 % of all instances as opposed to 67 % by SCIP-NLP. We point out that SCIP-NLP is the only solver that hits the memory limit of 24 GB RAM for 29 instances. In this case, we use the best solutions that are computed before the memory limit is reached.

In terms of the run times, Figure 2.4 shows a similar picture as for the relative optimality gaps. Solving the problems as MIPs is orders of magnitude faster than solving them as NLPs. The performance of the solvers can even be ordered monotonously. Gurobi is by far the fastest solver, which solves more than 80 % of all problems to global optimality. Next comes SCIP-MIP, which finds an optimal solution in about 10 % of

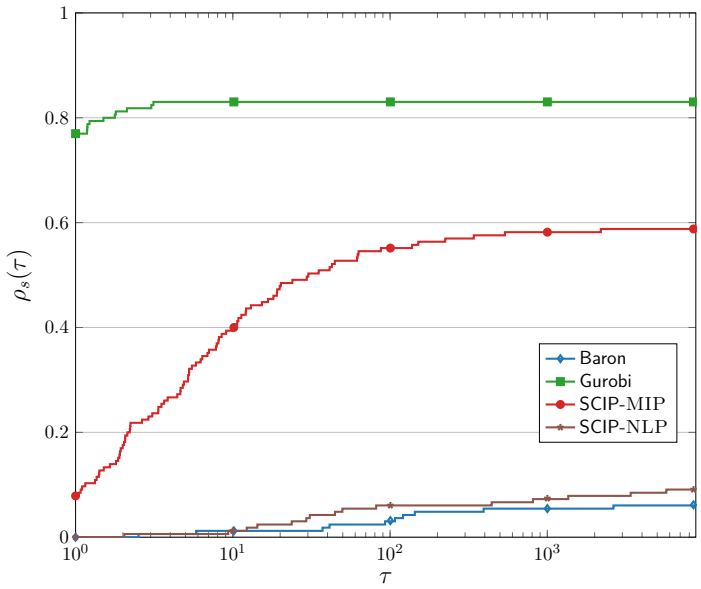


FIGURE 2.4. Performance profile for Gurobi, SCIP-MIP, Baron, and SCIP-NLP comparing run times needed to solve the benchmark problems to global optimality with a total run time limit of 4 h per instance.

the problems as quickly as the fastest solver. It is also able to find optimal solutions for almost 60 % of all instances within a run time factor of 10^3 compared to Gurobi. SCIP-NLP is never the quickest to find an optimal solution. In total, it can find an optimal solution in about 10 % of all cases, whereby the run times are up to a factor of 10^3 longer than those of the fastest solvers. As the last comes Baron, which is a little slower than SCIP-NLP. Moreover, it solves only about 6 % of the benchmark problems to global optimality.

The experiment of reformulating MIPs to NLPs shows that an intensive study of special structures and classes of constraints, such as $x^2 - x = 0$, is mandatory. To exaggerate, one could even say that the MIP community has specialized in the nonlinear constraint $x^2 - x = 0$. The exploitation of special structures that arise in MIPs, on the other hand, has led to MIP technology becoming capable of solving a large number of various MIPs nowadays. A finer categorization of MINLPs than only convex and non-convex may therefore be useful and even necessary to achieve a comparable development for MINLPs.

CHAPTER 3

Adaptive MIP Relaxations for MINLPs

In this chapter, we develop a multi-tree algorithm for solving MINLPs, which mainly relies on the solution of adaptively refined MIP relaxations of the MINLP. Our goal is to find an optimal solution for MINLP problems as in (P) such that no constraint of the problem is violated by more than an a priori (but arbitrarily) given error bound.

We therefore adaptively construct MIP relaxations of (P) and solve these to global optimality until no given error bound is violated. We indicate that in practice, the smallest attainable error bound is determined by the tolerance of the MIP solver. Although not all constraints of (P) may be fulfilled exactly by our approach, we refer to it as global optimization of MINLPs, since all a priori given error bounds are controllable and in fact all so-called global optimization algorithms for MINLP problems deal with feasibility tolerances, e.g., for integrality or constraint violations. For solving MIP and LP problems to arbitrarily high levels of precision, we refer to Cook et al. (2011) and Gleixner (2015).

In the following, we prove convergence of our MIP-based approach and investigate different refinement procedures. Furthermore, we present some results on piecewise linear approximations of nonlinear functions. More precisely, we show how to compute the maximal approximation error and for some specific functions, we locate the points that attain this error. Additionally, tight bounds on the size of MIP relaxations that are required to achieve an a priori given accuracy are obtained.

In the last decade, several publications on solving MINLPs by MIP relaxations and especially MIP approximations appeared, many of which are application-driven. Martin et al. (2006) describe one of the first approaches on this topic. The algorithmic groundwork for our adaptive multi-tree approach is laid by Geißler (2011) and Geißler et al. (2012a) introducing the idea of solving a series of adaptively refined MIP relaxations. Moreover, techniques to obtain MIP relaxations based on MIP approximations are presented. Very recently, Sirvent (2018) presented adaptive decomposition-based methodologies for solving MINLPs with simulation-based black-box nonlinearities using MIP relaxations. The focus therein lies on relaxation strategies, while in this work we place more emphasis on refinement strategies. Another adaptive approach based on α-convexifications for solving non-convex MINLPs with twice differential nonlinearities is given by Lundell et al. (2013). Here, the authors use piecewise linear

approximations of the α-convexifications to obtain tighter convex underestimators. Breakpoints are added iteratively until the optimal solutions of the resulting convex MINLP relaxations converge to the global solution of the non-convex MINLP. In contrast, we focus on MIP technology and therefore solve a series of MIP relaxations of the non-convex MINLP. However, in all these approaches two important problems must be addressed: the construction of good approximations of the nonlinear functions and the incorporation of these approximations into an MIP.

One way to obtain such approximations is to fix the error in advance and compute optimal linearizations for each function as in Rebennack and Kallrath (2015a,b). Complementary, Morsi (2013) shows how to construct polynomial relaxations of one-dimensional nonlinear functions with an a priori given approximation error and a minimal number of line segments. For up to three-dimensional functions, explicit approximation techniques for general nonlinear functions have been developed by Misener and Floudas (2010). The main drawback of all these methods, however, is that the number of simplices in the approximation grows exponentially with the dimension of the function. We point out that the approach by Rovatti et al. (2014) overcomes this problem by dropping the requirement that the piecewise linear function must interpolate the original nonlinear function at the vertices of the triangulation. In our case, we base the relaxations on piecewise linear approximations that interpolate the function at the vertices. The relaxations completely contain the graph of the function and are defined on simplices, which are defined by several vertices.

This chapter is structured as follows. First, in Section 3.1, we show in detail how to model piecewise linear functions by the generalized incremental method as an MIP. Furthermore, we give a brief overview of other MIP models for piecewise linear functions. In Section 3.2, we describe how to compute the maximal approximation error for piecewise linear approximations of nonlinear functions. For some specific functions, we characterize where points attaining this error are located. Section 3.3 presents our main algorithm for solving MINLPs, which is based on the adaptive refinement of MIP relaxations. Moreover, we classify the refinement procedures that guarantee convergence of the algorithm. In Section 3.4, we investigate different refinement strategies that we can apply in our adaptive MIP-based algorithm. More precisely, we analyze for each refinement method whether it is contained in the previously introduced class of refinements that lead to a convergent main algorithm. Finally, we give tight bounds on the size of MIP relaxations that is required to achieve an a priori given accuracy in Section 3.5.

Section 3.1 and 3.2 mostly summarizes known methods and facts from the literature. Many results in Section 3.3–3.5 have been published in

R. Burlacu, B. Geißler, and L. Schewe (2019). "Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes". In: *Optimization Methods and Software*, pp. 1–28. DOI: 10.1080/10556788.2018.1556661.

The author of this thesis made significant contributions to this publication. All other results in Section 3.3–3.5 that are not included in the publication from above as well as Lemma 3.3, have been achieved by the author and are presented for the first time in this thesis.

3.1. Modeling Piecewise Linear MIP Relaxations

In this section, we show how to model piecewise linear relaxations as an MIP. We start with two basic definitions.

DEFINITION 3.1. With affinely independent $\bar{x}_0, \dots, \bar{x}_k \in \mathbb{R}^d$, the set

$$S = \left\{ x \in \mathbb{R}^d \,:\, x = \sum_{i=0}^{k} \lambda_i \bar{x}_i \;\text{ with }\; \sum_{i=0}^{k} \lambda_i = 1 \;\text{ and }\; \lambda_i \geq 0 \;\text{ for all } i = 1, \dots, k \right\}$$

is called a k-dimensional *simplex*.

In this thesis, we consider a simplex $S \subset \mathbb{R}^d$ to be full-dimensional, i.e., $k = d$ in Definition 3.1, and described by its extreme points $\mathcal{V}(S) = \{\bar{x}_0, \dots, \bar{x}_d\}$. We recall that D_f is the compact domain of the nonlinear function $f \in \mathcal{F}$.

DEFINITION 3.2. The set $\mathcal{T} = \{S_1, \dots, S_n\}$ with (full-dimensional) simplices $S_i \subset \mathbb{R}^d$ for $i = 1, \dots, n$, where d is the dimension of D_f, is called a *triangulation* of D_f, if $D_f = \bigcup_{i=1}^{n} S_i$ and $\operatorname{int}(S_j) \cap \operatorname{int}(S_k) = \emptyset$ for every $j \neq k$. The notation $\operatorname{int}(S)$ is used to denote the relative interior of a set $S \subset \mathbb{R}^d$.

Since we are only interested in an optimal solution for the MINLP problem (P) such that no constraint is violated by more than an a priori given error bound, it suffices for our purposes if a piecewise linear approximation ϕ_f of $f \in \mathcal{F}$ is a set of continuous piecewise linear functions that completely cover the domain D_f. Therefore, ϕ_f is not necessarily a function itself, but more generally described as a set of continuous affine functions $\phi_{f;S_i}(x) \colon S_i \to \mathbb{R}$, together with a triangulation $\mathcal{T}(\phi_f) = \{S_1, \dots, S_n\}$ of D_f and auxiliary binary variables $z \in \{0,1\}^{n-1}$ indicating which of the n functions $\phi_{f;S_i}$ is chosen. Herewith, the aim is to formulate a mixed-integer linear model in which

$$y = \phi_f(x, z) \coloneqq \phi_{f;S_1}(x) + \sum_{i=1}^{n-1} z_i (\phi_{f;S_{i+1}}(x) - \phi_{f;S_i}(x)) \tag{2}$$

holds for $x \in D_f$. Additionally, if $z_i = 1$ then $z_{i-1} = 1$ also holds and if $z_0 = 0$ then the first approximation $\phi_{f;S_0}(x)$ is active. We note that the notation $\mathcal{T}(\phi_f)$ denotes a

triangulation $\mathcal{T}$ that is associated with the approximation ϕ_f, but does not depend on it.

In general, an approximation covers only a relatively small part of the graph of the nonlinear function f. This means that we cannot represent the entire feasible set of the MINLP by an MIP approximation. Consequently, an optimal solution of the MIP approximation does not deliver a dual bound for the MINLP. However, since a relaxation of f covers the graph of f completely, the corresponding MIP relaxation covers the entire feasible set of the MINLP. Thus, we can provide a dual bound for the MINLP by the objective function value of the optimal solution of the MIP relaxation. Furthermore, if the MIP relaxation is infeasible, the MINLP itself is infeasible as well. On the contrary, if an MIP approximation is infeasible, we cannot conclude that the MINLP itself is infeasible. With

$$\epsilon_u(f, S) \coloneqq \max_{x \in S} f(x) - \phi_{f;S}(x),$$
$$\epsilon_o(f, S) \coloneqq \max_{x \in S} \phi_{f;S}(x) - f(x)$$

as the maximum underestimation and the maximum overestimation of f by $\phi_{f;S}$, we obtain a relaxation on the basis of the approximation (2) by

$$y = \phi_f(x, z) + e_f,$$
$$e_f \leq \epsilon_u(f, S_1) + \sum_{i=1}^{n-1} z_i(\epsilon_u(f, S_{i+1}) - \epsilon_u(f, S_i)), \tag{3}$$
$$e_f \geq -\epsilon_o(f, S_1) - \sum_{i=1}^{n-1} z_i(\epsilon_o(f, S_{i+1}) - \epsilon_o(f, S_i)). \tag{4}$$

See Figure 3.1 for an illustration. The blue lines depict a piecewise linear approximation of the nonlinear function $f(x) = 0.5\,|x|\,x$ on the three simplices $S_1 = [-2, -1]$, $S_2 = [-1, 1]$, and $S_3 = [1, 2]$. In order to obtain a relaxation, we add the maximal approximation error on each simplex to the approximation or subtract it. On S_1 the function f is only underestimated, while on S_3 it is only overestimated. Since the approximation underestimates and overestimates f on S_2, we have to add and subtract the corresponding errors on this simplex. Please note that $\epsilon_u(f, S)$ and $\epsilon_o(f, S)$, respectively, are equal to zero if $\phi_{f;S}$ does not underestimate or overestimate f on the simplex S.

3.1.1. Generalized Incremental Model. There are many different ways to model piecewise linear functions as an MIP. We apply the classic incremental method of Markowitz and Manne (1957b), which has been extended to relaxations by Geißler et al. (2012b). Originally developed for one-dimensional functions, a generalization to higher dimensions is described by J. Lee and Wilson (2001) and Wilson (1998)

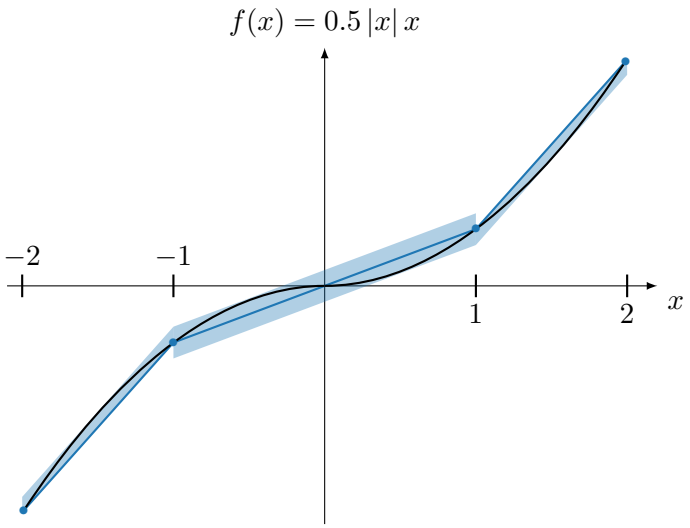


FIGURE 3.1. A relaxation (light blue area) of $f(x) = 0.5\,|x|\,x$, where the maximum underestimation and the maximum overestimation are added to or subtracted from the approximation of f (blue lines).

and Geißler (2011). The latter presents numerical experiments that show an advantage of the generalized incremental model over other models. Similar numerical results are presented by Correa-Posada and Sánchez-Martín (2014). Nevertheless, it is plausible that, depending on the nature of the MINLP problem and in particular the occurring nonlinearities, it is sensible to use different MIP models for the piecewise linear relaxations. So far, however, there are no rigorous numerical experiments that would confirm such an assumption. This has yet to be clarified in future research.

There are two main ideas of the generalized incremental model. At first, any point x^S inside a simplex S with $\mathcal{V}(S) = \{\bar{x}_0, \dots, \bar{x}_d\}$ can be expressed either as a convex combination of its vertices or equivalently as

$$x^S = \bar{x}_0^S + \sum_{j=1}^{d} (\bar{x}_j^S - \bar{x}_0^S)\delta_j^S \tag{5}$$

with $\sum_{j=1}^{d} \delta_j^S \leq 1$ and $\delta_j^S \geq 0$ for $j = 1, \dots, d$.

The other main idea is that all simplices of a triangulation are ordered in such a way that the last vertex of any simplex is equal to the first vertex of the next one. In this way, we can construct a Hamiltonian path and model the piecewise linear approximation along this path. It is now sufficient to show that an ordering of the simplices with the following properties is available:

(O1) The simplices in $\mathcal{T} = \{S_1, \dots, S_n\}$ are ordered such that $S_i \cap S_{i+1} \neq \emptyset$ for $i = 1, \dots, n-1$, and

(O2) for each simplex S_i its vertices $\bar{x}_0^{S_i}, \dots, \bar{x}_d^{S_i}$ can be labeled such that $\bar{x}_d^{S_i} = \bar{x}_0^{S_{i+1}}$ for $i = 1, \dots, n-1$.

Let $\bar{y}_j^{S_i} \coloneqq f(\bar{x}_j^{S_i})$ be the value of f at the vertex $\bar{x}_j^{S_i}$ of Simplex S_i. More formally, for any nonlinear function $f \in \mathcal{F}$, we describe the generalized incremental model by:

$$x = \bar{x}_0^{S_1} + \sum_{i=1}^{n} \sum_{j=1}^{d} (\bar{x}_j^{S_i} - \bar{x}_0^{S_i}) \delta_j^{S_i}, \tag{6a}$$

$$y = \bar{y}_0^{S_1} + \sum_{i=1}^{n} \sum_{j=1}^{d} (\bar{y}_j^{S_i} - \bar{y}_0^{S_i}) \delta_j^{S_i}, \tag{6b}$$

$$1 \geq \sum_{j=1}^{d} \delta_j^{S_i} \qquad \text{for all } i = 1, \dots, n, \tag{6c}$$

$$0 \leq \delta_j^{S_i} \qquad \text{for all } i = 1, \dots, n;\ j = 1, \dots, d, \tag{6d}$$

$$z_i \geq \sum_{j=1}^{d} \delta_j^{S_{i+1}} \qquad \text{for all } i = 1, \dots, n-1, \tag{6e}$$

$$z_i \leq \delta_d^{S_i} \qquad \text{for all } i = 1, \dots, n-1, \tag{6f}$$

$$z_i \in \{0, 1\} \qquad \text{for all } i = 1, \dots, n-1. \tag{6g}$$

Constraints (6e)–(6g) ensure that the δ-variables satisfy the filling condition, which states that if for any simplex S_i a variable $\delta_j^{S_i}$ is positive, then $\delta_j^{S_{i-1}} = 1$ must hold. This means that $\delta_j^{S_i}$ can only be positive if the variables $\delta_d^{S_k}$ are equal to one for all previous simplices $k = 1, \dots, i-1$. See Figure 3.2 for an illustration in case of a two-dimensional domain D_f together with a triangulation and its corresponding ordering of the simplices. All $\delta_d^{S_k}$ are equal to one for $k = 1, \dots, 3$ and equal to zero for $k = 5, \dots, 9$. Within the simplex S_4, the point x is obtained by $\delta_1^{S_4} = 0.2$ and $\delta_2^{S_4} = 0.6$. The point x is represented by the blue path within the triangulation, which corresponds to (5).

Another important characteristic of the generalized incremental method is the *local ideality*. An MIP model of a piecewise linear function is locally ideal if its LP relaxation is integral, i.e., all binary variables of the MIP model are integral in the optimal solution of the LP. Wilson (1998) showed that the MIP formulation (6) for a single nonlinear function f is locally ideal.

We can now obtain an MIP relaxation for (P) by replacing (6b) with

$$y = \bar{y}_0^{S_1} + \sum_{i=1}^{n} \sum_{j=1}^{d} (\bar{y}_j^{S_i} - \bar{y}_0^{S_i}) \delta_j^{S_i} + e_f,$$

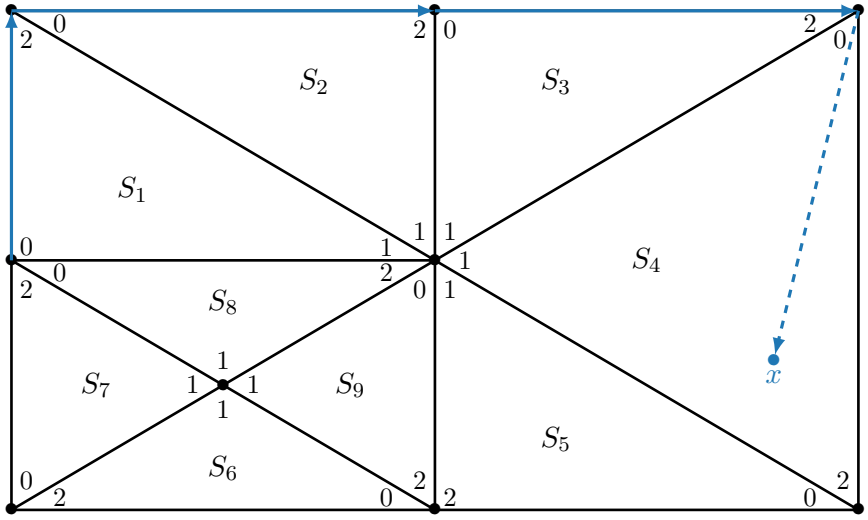


FIGURE 3.2. A triangulation of a two-dimensional domain D_f and its corresponding ordering of the simplices S_1–S_9. Each point x is represented by a path (thick blue arrows) through the triangulation.

and adding the inequalities (3) and (4). Moreover, we have lower and upper bounds

$$-\max_{S\in\mathcal{T}(\phi_f)} \epsilon_o(f,S) \;\le\; e_f \;\le\; \max_{S\in\mathcal{T}(\phi_f)} \epsilon_u(f,S)$$

for $e_f \in \mathbb{R}$. Concerning the computation of the approximation errors $\epsilon_u(f,S)$ and $\epsilon_o(f,S)$, we refer to Section 3.2.

Additionally, if the gradient $\nabla f(\hat{x})$ of $f \in \mathcal{F}$ at point $\hat{x}$ is available and f is convex or concave, we can tighten the relaxation by adding linearization cuts of the type

$$\nabla f(\hat{x})^\top (x - \hat{x}) \le y \qquad \text{or} \qquad \nabla f(\hat{x})^\top (x - \hat{x}) \ge y.$$

However, since we pursue an approach that solves multiple MIPs and thus uses multiple branch-and-bound trees, it can be too expensive to store these linearization cuts throughout all iterations. Hence, it may be sensible to use very few or even no linearization cuts at all.

3.1.2. Other MIP Models. For the sake of completeness, we mention further MIP models for piecewise linear functions, but do not go into much detail. As already pointed out, it is still unanswered which model is most suitable for the various MINLP problems and therefore still the subject of future research. A detailed overview of the following models is given by Vielma et al. (2010).

The generalized convex combination model expresses a point x as a convex combination of the vertices of the simplex that contains x. There are two variants of this model. The aggregated model introduces only one convex combination variable

for each vertex of the triangulation. The disaggregated model contains a convex combination variable for each vertex of the simplex. The first model consists of fewer variables, but is not locally ideal. The second one uses more variables, but is locally ideal.

Another MIP model for piecewise linear functions are the generalized logarithmic models. Based on the disaggregated convex combination model the idea is to use only a logarithmic number of binary variables, since choosing between n simplices can be modeled by $\log n$ binary variables. An aggregated version of this model is also available, but is restricted to specific triangulations. Both versions, however, are locally ideal.

3.2. Accuracy of Piecewise Linear Approximations

As we have already seen, the estimate of the maximal approximation error is a necessary part of our approach. The exact computation of this error is crucial to obtain tight MIP relaxations whose feasible set is just big enough to cover the graph of the nonlinear function f. There are many fairly general and more specific cases for which this is possible with relatively little effort.

First, Geißler (2011) showed how the maximal approximation error on a simplex and a point attaining the error can be determined by solving at most $d + 2$ convex optimization problems, if convex or concave envelopes of f are known. These optimization problems are rather small, since they contain only d variables. Although it is generally NP-hard to find convex envelopes Crama (1989), they are analytically given for a considerable number of nonlinear functions; see for example Geißler (2011), Jach et al. (2008), and McCormick (1976). In case exact convex envelopes are not available, it is possible to only use convex underestimators; see Androulakis et al. (1995) and Tawarmalani and Sahinidis (2004) for more details. Compared to envelopes, the quality of the MIP relaxation, however, decreases depending on the tightness of the underestimators.

In point of fact, we primarily need the maximal approximation errors on the respective simplices rather than the convex envelopes. For specific functions, we can explicitly indicate points where the maximal error is attained. For instance, in case of quadratic one-dimensional functions, like $f(x) = x^2$, such a point is given by the midpoint of the line-segment (one-dimensional simplex). Another example is $f(x, y) = xy$, for which a lot of research has been done regarding convex envelopes, starting with the above mentioned work by McCormick (1976). For this function, we know that on a simplex, a point with maximal approximation error lies on one of the midpoints of the three edges; see Pottmann et al. (2000) for example. We can more

generally specify when a point with maximal approximation error lies on one of the facets of the simplex.

LEMMA 3.3. *Let $f\colon D_f \to \mathbb{R}$ with (compact) $D_f \subset \mathbb{R}^d$ and a triangulation $\mathcal{T}$ of D_f. If for each $x \in D_f$ there is a line $L_x \subset \mathbb{R}^d$ containing x such that the nonlinear function f is linear along L_x, then for each simplex $S \in \mathcal{T}$ there is a point with maximal approximation error on one of the facets of S.*

PROOF. Let $S \in \mathcal{T}$ and $\phi_{f;S}$ be an affine linear approximation of f on the simplex S. Furthermore, let x be a point in the interior of the simplex S and L_x be a line along which f is linear. Naturally, $\phi_{f;S}$ is also linear along L_x. Thus, the approximation error $|\phi_{f;S} - f|$ is linear along L_x. A linear function, however, is either monotonically decreasing or increasing. Since we want to determine the maximal approximation error on the simplex S and S is a compact set, we only need to consider the linear function $|\phi_{f;S} - f|$ along L_x on a compact domain $[l, u]$, where l, u correspond to some points on the facets of S. It follows that the maximal error is either attained at l, u, or both l and u. Therefore, a point with maximal approximation error lies on one of the facets of S. □

With Lemma 3.3, for some nonlinear functions, it is easy to show that a point with maximal approximation error lies on one of the facets of a simplex of the triangulation.

EXAMPLE 3.4. *Let*

$$f(x) = \frac{\prod_{i \in I} a_i x_i}{\prod_{j \in J} a_j x_j} \tag{7}$$

with $D_f \subset \mathbb{R}^d$, $a_i, a_j \in \mathbb{R}$, $I \cap J = \emptyset$, and $I \cup J = \{1, \ldots, d\}$. For $\bar{x} \in D_f$, we consider the line $L_x = \bar{x} + \lambda e_k$, where $\lambda \in \mathbb{R}$, $k \in I$ and e_k is the kth unit vector. Along L_x, however, f is given by

$$(\bar{x}_k + \lambda)\, f(\bar{x}_1, \ldots, \bar{x}_{k-1}, 1, \bar{x}_{k+1}, \ldots, \bar{x}_d),$$

which is linear in λ. For each simplex of a triangulation of D_f, it follows by Lemma 3.3 that the maximal approximation error is attained at one of its facets.

Please note that the function $f(x, y) = xy$ is a special case of (7). Due to Lemma 3.3, a point with maximal error lies on one of the three edges of a simplex in case of $f(x, y) = xy$. Moreover, along the edges of the simplices, f can be described by a quadratic one-dimensional function, for which the maximal error is attained at the midpoint of the edges. This is another proof that for $f(x, y) = xy$ a point with maximal approximation error lies on one of the midpoints of the three edges.

EXAMPLE 3.5. *Let*

$$f = \left(\sum_{i=1}^{d} (a_i x_i^k) \right)^{\frac{1}{k}} \tag{8}$$

with $D_f \subset \mathbb{R}^d$, $a_i \in \mathbb{R}$, and $k \in \mathbb{R}^+$. Additionally, we assume that D_f is contained in the positive orthant of $\mathbb{R}^d$ if k is an even integer. For $\bar{x} \in D_f$, we consider the line $L_x = \lambda \bar{x}$, where $\lambda \in \mathbb{R}$. Along L_x, the nonlinear function f is given by

$$\left(\sum_{i=1}^{d}(a_i \bar{x}_i^k \lambda^k)\right)^{\frac{1}{k}} = \left(\sum_{i=1}^{d}(a_i \bar{x}_i^k)\right)^{\frac{1}{k}} \lambda,$$

which again is linear in λ. This also holds if k is an even integer due to our assumption that D_f is contained in the positive orthant of $\mathbb{R}^d$. It again follows by Lemma 3.3 that the maximal approximation error is attained at one of the facets for each simplex of a triangulation of D_f.

A special case of a function as in (8) is $f(x, y) = \sqrt{x^2 - y^2}$, which occurs in the context of gas transport optimization. This function essentially models the correlation between the gas flow q and pressure loss $p_{\text{in}} - p_{\text{out}}$ in a pipeline, which is given by the Weymouth equation

$$\beta |q| q = p_{\text{in}}^2 - p_{\text{out}}^2,$$

where β is some constant. We refer to Chapter 4 for more details on gas transport optimization.

As described in Chapter 2, it is possible to subdivide multi-dimensional functions into several lower-dimensional functions by expression trees. For the new functions, the domains and the a priori given error bounds must be converted from the original ones according to the propagation through the expression tree. This allows to split the problem of determining the maximal approximation error into several problems with lower dimensions. In most cases it is reasonable to follow such a strategy. MIP relaxations based on expression trees, however, are not always useful, for example if the variables of the function have a very large range. Since the errors of the lower-dimensional functions can amplify, the problem can quickly become numerically intractable. In this case, it is more sensible to approximate the multi-dimensional function directly and construct an MIP relaxation from it.

If the dimension of the function is comparatively low, it is also possible to determine the maximal error using a global NLP solver; see again the book by Horst and Tuy (1996) for extensive details on methods of global optimization. As the dimension increases, this problem becomes increasingly difficult to solve. Thus the NLP part of the MINLP becomes more and more dominant and we obtain more and more a pure NLP problem. In this thesis, however, we focus mainly on MINLPs that have a significant MIP part.

Finally, we point out that if the Lipschitz-constant of the nonlinear function is known, we can estimate the maximal approximation error by evaluation of the function over a grid on the respective simplex. The quality of the estimate can then be

controlled with the resolution of the grid. Naturally, depending on the dimension of the function, there is a trade-off between the costs of the evaluations and the accuracy of the estimate.

As pointed out before, for a notable variety of functions, optimal triangulations (with respect to the number of simplices) that satisfy a given approximation error, are known. Such triangulations can be used in our approach to find an initial MIP relaxation of the MINLP, which is then adaptively refined. In the next section, we clarify in more detail what such an algorithm looks like.

3.3. An Adaptive Refinement Algorithm for MINLPs

Our aim is to solve an MINLP problem, while we mainly solve MIP relaxations. For the remainder of this chapter, we always refer to globally optimal solutions whenever we speak of optimal solutions.

We follow a multi-tree approach, which solves several MIPs with increasing accuracy. Initially, we solve a coarse MIP relaxation and check whether all error bounds are satisfied in its optimal solution. Then, we locally refine those piecewise linear approximations, where the error bounds are violated. Afterward, from the refined approximation, a new MIP relaxation is constructed and the procedure is started over with the new relaxation until all error bounds are satisfied. A more formal description of this approach is given in Algorithm 1.

The reason for the adaptivity in our algorithm is that we are only interested in an optimal solution of an MIP relaxation that satisfies all a priori given error bounds. It is not necessary that all feasible points of the MIP relaxation comply with these errors bounds. The adaptive approach allows us to omit many binary variables, which has a significant impact on the run time. Another more theoretical reason is that fixing the accuracy can lead to bad outcomes when solving MINLPs as shown by Dey and Gupte (2015). Therein, the authors show how poorly even very fine piecewise linear relaxations scale for the specific example of the pooling problem. More precisely, they prove that for any $\epsilon > 0$ and for any piecewise linear relaxation, there exists an instance of the pooling problem such that the ratio of the optimal function value of the relaxation to the optimal function value is at least $n - \epsilon$ where n is the number of output nodes. This shows that it is necessary to not rely on an fixed discretization of the nonlinearities. Consequently, our adaptive approach circumvents the problems discussed by Dey and Gupte (2015).

In the following, we denote the projection of the optimal solution x of an MIP relaxation on D_f by x_f for all $f \in \mathcal{F}$. Moreover, the Euclidean norm is denoted by $\|\cdot\|$. The refinement procedure in Algorithm 1 is somewhat abstract for now. We investigate different possibilities for this in Section 3.4.

Algorithm 1 Global optimization of an MINLP by solving adaptively refined MIP relaxations

Input: An MINLP problem P of type (P), upper bounds $\epsilon_f^0 > 0$ for the absolute linearization errors in the piecewise linear approximations used to construct the initial relaxation and the maximal absolute linearization errors $\epsilon_f > 0$ for all $f \in \mathcal{F}$.

Output: If P is feasible, the algorithm returns an optimal solution x of an MIP relaxation Π of P with $|f(x_f) - y_f| \leq \epsilon_f$ for all $f \in \mathcal{F}$ and $c^\top x \leq c^\top x'$ for any feasible point x' of P. If no such MIP relaxation Π of P exists, this is reported by returning *infeasible*.

Set $\mathcal{F} \leftarrow$ all nonlinear functions that are contained after (optional) reformulation of P with the help of expression trees.
Set $D_f, \epsilon_f^0, \epsilon_f \leftarrow$ the domain D_f and the error bounds ϵ_f^0, ϵ_f that result from the propagation of the original domain and error bounds through the expression trees for all $f \in \mathcal{F}$.
Compute an initial piecewise linear approximation ϕ_f^0 of $f \in \mathcal{F}$ satisfying the upper bound ϵ_f^0 for all $f \in \mathcal{F}$.
Set $i \leftarrow 0$.
repeat
  Construct an MIP relaxation Π^i of P from ϕ_f^i for all $f \in \mathcal{F}$.
  Solve Π^i.
  if Π^i is *feasible* **then**
    Set $x^i \leftarrow$ optimal solution of Π^i.
  else
    return *infeasible*.
  end if
  Set stop $\leftarrow$ true.
  for all $f \in \mathcal{F}$ **do**
    Set $x_f^i \leftarrow$ projection of x^i on D_f.
    Set $y_f^i \leftarrow$ value of the variable for the approximated function value of f.
    if $|f(x_f^i) - y_f^i| > \epsilon_f$ **then**
      Set $z_f \leftarrow$ values of the binary variables in x^i used to model ϕ_f^i.
      Set $S_f \leftarrow$ simplex in $\mathcal{T}(\phi_f^i)$, which has been selected according to z_f.
      Set $\mathcal{T}(\phi_f^{i+1}) \leftarrow$ refinement of the triangulation $\mathcal{T}(\phi_f^i)$ (see Section 3.4).
      Set $\phi_f^{i+1} \leftarrow$ piecewise linear approximation according to $\mathcal{T}(\phi_f^{i+1})$.
      Set stop $\leftarrow$ false.
    else
      Set $\phi_f^{i+1} \leftarrow \phi_f^i$.
    end if
  end for
  Set $i \leftarrow i + 1$.
until stop.
return x^{i-1}.

We now prove the convergence of our algorithm under the assumption that the refining procedure has the following property. The refinement strategies that fulfill this property in particular are described in Section 3.4.

DEFINITION 3.6. The refinement procedure in Algorithm 1 is called *δ-precise*, if for an arbitrary sequence $(S^i) \in \mathcal{T}_i$ of simplices that are refined by the refinement procedure with initial triangulation $\mathcal{T}_0$ of D_f and given $\delta > 0$, there exists an index $N \in \mathbb{N}$, such that

$$\operatorname{diam}(S^N) < \delta \tag{9}$$

holds, where $\operatorname{diam}(S^N) \coloneqq \sup_{x',x'' \in S^N}\{\|x' - x''\|\}$.

THEOREM 3.7. *If the refinement procedure in Algorithm* 1 *is δ-precise for every $\delta > 0$, then Algorithm* 1 *is correct and terminates after a finite number of steps.*

PROOF. We first show that Algorithm 1 terminates after a finite number of steps and prove correctness afterward.

Let (x_f^i) be the sequence of (projected) optimal solutions of Π^i for $f \in \mathcal{F}$. Since the refinement procedure is δ-precise for every $\delta > 0$, there is an index $N \in \mathbb{N}$ with $\|\bar{x} - \bar{x}_i\| < \delta$ for every $\delta > 0$, $\bar{x} \in S_f^N$ and $\bar{x}_i \in \mathcal{V}(S_f^N)$. Furthermore, because D_f is compact, f is uniformly continuous on D_f and therefore there is a $\delta_f > 0$ such that $|f(\bar{x}) - \phi_f^N(\bar{x})| < \epsilon_f/2$. From the construction of the MIP relaxation it follows that

$$y_f^N \in [\phi_f^N(x_f^N) - (\epsilon_f/2), \phi_f^N(x_f^N) + (\epsilon_f/2)].$$

Thus, we have

$$|f(x_f^N) - y_f^N| \leq |f(x_f^N) - \phi_f^N(x_f^N)| + \frac{\epsilon_f}{2} \leq \epsilon_f$$

and Algorithm 1 terminates after a finite number of steps.

If, on the one hand, P is feasible, we can assure that Algorithm 1 does not return *infeasible* in any iteration. Because the algorithm terminates after a finite number of steps, there must be an MIP relaxation Π^N which has an optimal solution x^N with $|f(x_f^N) - y_f^N| \leq \epsilon_f$ for all $f \in \mathcal{F}$. Moreover, Π^N is a relaxation of P so that $c^\top x^N \leq c^\top x'$ for all feasible points x' of P.

If, on the other hand, there exists no MIP relaxation Π of P which has a feasible point $\tilde{x}$ satisfying $|f(\tilde{x}_f) - y_f| \leq \epsilon_f$ for all $f \in \mathcal{F}$, we can easily conclude that Algorithm 1 must return *infeasible*, since it terminates after a finite number of steps. □

THEOREM 3.8. *Let the refinement procedure in Algorithm* 1 *be δ-precise for every $\delta > 0$. If $\epsilon_f \to 0$ for all $f \in \mathcal{F}$, then the (global) optimal solution obtained by Algorithm* 1 *converges to a (global) optimal solution of the* MINLP *problem P if and only if P is feasible.*

PROOF. First, it is clear that if Algorithm 1 converges to an optimal solution of the MINLP problem P, then P must be feasible.

Let us now assume that P is feasible. Then, any MIP relaxation of P is feasible. We consider a sequence $(\epsilon_f/2^i)$ of constantly decreasing error bounds for all $f \in \mathcal{F}$ with limit 0. For each of these sequence elements, we obtain an optimal solution of an MIP relaxation that satisfies the corresponding error bound due to the δ-preciseness of the refinement procedure and Theorem 3.7. This yields a sequence of optimal solutions (x_f^i) of some MIP relaxations that satisfy the error bounds $\epsilon_f/2^i$. Since D_f is compact, there is a convergent subsequence of (x_f^i) with a limit $\tilde{x}_f$ and corresponding error bound 0. This means, however, that $\tilde{x}_f$ is feasible for P, because the approximation error is 0 and P is feasible. Moreover, $\tilde{x}_f$ is the optimal solution of an MIP relaxation of P, from which it follows that $\tilde{x}_f$ is optimal for P due to the relaxation property. □

3.4. Refinement Procedures

In this section, we investigate different refinement strategies that we can apply in Algorithm 1. We primarily focus on the δ-preciseness of the methods. Furthermore, if the refinement procedure is δ-precise, we show how to obtain a new triangulation of the domain D_f after a refinement step. Finally, we give a brief comparison of the described methods using an example of a two-dimensional nonlinear function.

3.4.1. Limited Accuracy by Adding Points with Maximal Error. Intuitively, the method of adding a point with maximal approximation error as a new vertex to the triangulation and constructing a new triangulation from it, seems very promising. We can proceed as follows: If a simplex S_f of triangulation $\mathcal{T}(\mathcal{V}_f)$ is chosen for refinement, we add a point with maximal approximation error on S_f to the set $\mathcal{V}_f$ of linearization points obtaining a new triangulation $\mathcal{T}(\mathcal{V}_f \cup \{v\})$. Unfortunately, an approximation with arbitrary accuracy is not always attainable by such a procedure, as we will show in the following.

Indeed, let ϕ_f be a piecewise linear approximation of a nonlinear function $f \in \mathcal{F}$ on a box domain D_f. Moreover, let its triangulation $\mathcal{T}(\mathcal{V}_f)$ correspond to a set $\mathcal{V}_f$ of linearization points obtained by successively adding solely points with maximal approximation error on a simplex. As initial linearization points we consider the set of extreme points of D_f. We show that there are nonlinear functions f that cannot be approximated by ϕ_f with arbitrary accuracy.

THEOREM 3.9. *Let $g\colon \mathbb{R} \to \mathbb{R}$ be a continuous nonlinear function with at least three roots x_0, x_1, and x_2. Additionally, let $g(x) > 0$ for $x \in (x_0, x_1)$ and $g(x) < 0$ for $x \in (x_1, x_2)$ or vice versa. Then, g can be extended to a two-dimensional function f*

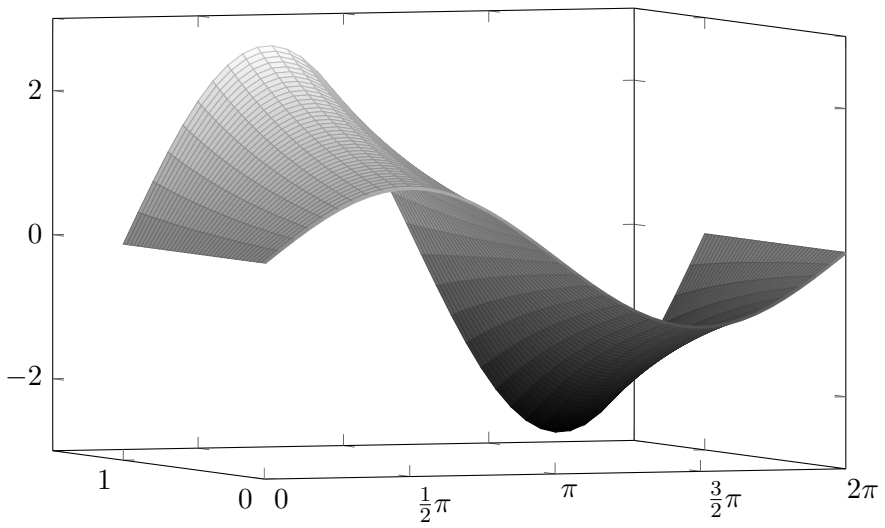


FIGURE 3.3. Extension of $g(x) = \sin(x)$ and $0 \leq x \leq 2\pi$ by multiplying with $h(y) = e^y$.

with domain D_f, such that there exists a fixed $\epsilon > 0$ with $|\phi_f(x,y) - f(x,y)| > \epsilon$ for any approximation ϕ_f and some $(x,y) \in D_f$ depending on ϕ_f.

PROOF. Let $h\colon \mathbb{R} \to \mathbb{R}$ be a continuous and strictly increasing function with $h(0) > 0$ and $f\colon D_f \to \mathbb{R}, (x,y) \mapsto g(x)h(y)$ with

$$D_f = \{(x,y) \in \mathbb{R}^2 \,:\, x_0 \leq x \leq x_2,\ 0 \leq y \leq \bar{y}\}$$

and arbitrary $\bar{y} > 0$. See Figure 3.3 for an illustration of such a function. We now prove that

$$\tilde{\epsilon} \coloneqq \min\left\{ \max_{x\in(x_0,x_1)} |g(x)|, \max_{x\in(x_1,x_2)} |g(x)| \right\} h(0)$$

is a lower bound for the approximation error of any piecewise linear approximation ϕ_f defined as above.

Let ϕ_f be such a piecewise linear approximation. Due to the construction of f, if $S_e \in \mathcal{T}(\mathcal{V}_f)$ is a simplex containing the edge $e = \operatorname{conv}\{(x_0,0),(x_2,0)\}$, then for all $(x,y) \in S_e$ either $\phi_f(x,y) \geq 0$ or $\phi_f(x,y) \leq 0$ holds. We now choose X to be one of the two intervals (x_0,x_1) and (x_1,x_2), such that with $x \in X$ either $g(x) > 0$ and $\phi_f(x,y) \leq 0$ or $g(x) < 0$ and $\phi_f(x,y) \geq 0$ holds. This is always possible due to the assumption that $g(x)$ has at least three roots x_0, x_1, and x_2 with $g(x) > 0$ for $x \in (x_0,x_1)$ and $g(x) < 0$ for $x \in (x_1,x_2)$ or vice versa. Then, for any $x \in X$ it follows that

$$|0 - g(x)h(0)| = |\phi_f(x,0) - f(x,0)| < |\phi_f(x,y) - f(x,y)| \tag{10}$$

for all $y > 0$, as $h(y)$ is strictly increasing and that a point with maximal approximation

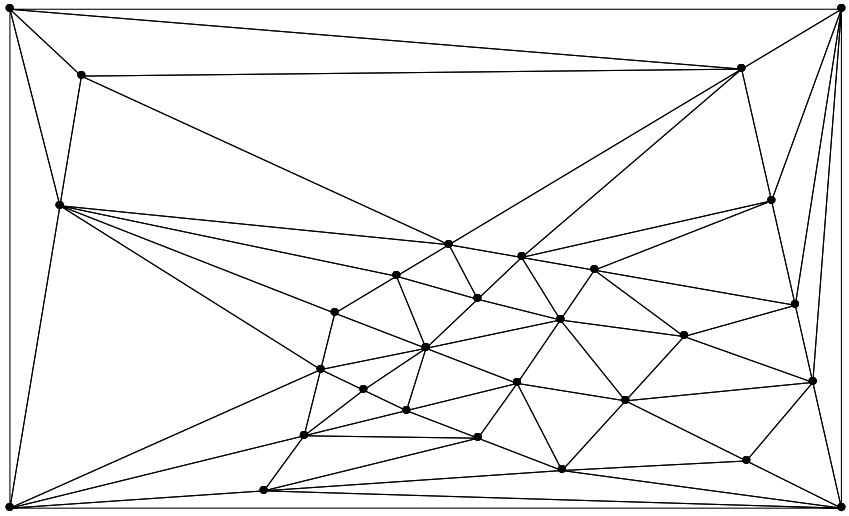


FIGURE 3.4. Delaunay triangulation of the two-dimensional domain $D_f = [0, 2\pi] \times [0, 2\pi]$ of the nonlinear function $f(x, y) = \sin(x)e^y$ with flat simplices obtained by adding points with maximal approximation error to the triangulation.

error can never be contained on edge e. Any initial triangulation of ϕ_f contains a simplex S_e with edge e, since the triangulation corresponds to set of extreme points of D_f. Thus, there is a simplex S_e with edge e in any approximation ϕ_f, because only points with maximal approximation error are added to the triangulation. Furthermore, considering that $\phi_f(x, 0) = 0$ in any ϕ_f, with (10) it follows that the maximal approximation error on S_e, and therefore on any piecewise linear approximation ϕ_f, is larger than $\tilde{\epsilon}$. □

Although a refinement strategy adding solely points with maximal approximation error on a simplex often works well in practice according to Geißler (2011); Geißler et al. (2013), it is due to Theorem 3.9 not necessarily δ-precise for every $\delta > 0$.

Another disadvantage is that the simplices of the triangulation can become very flat with such a procedure, even if we use a Delaunay triangulation. Flat simplices, however, lead to high numerical instability. Delaunay triangulations have the advantage that they maximize the smallest angle of all triangles; see Berg et al. (2008) and Delaunay (1934). Hence, they offer the best possibility to avoid flat simplices. We show an example for the domain $D_f = [0, 2\pi] \times [0, 2\pi]$ of the nonlinear function $f(x, y) = \sin(x)e^y$ from Figure 3.3. We consecutively generate 25 random points in the lower half ($y \leq \pi$) of the domain D_f. For each of these points, we determine a point with maximal approximation error on the simplex containing the point and subsequently construct a new Delaunay triangulation. The final triangulation after 25

refinement steps is depicted in Figure 3.4. We see in this example that the routine of adding points with maximal approximation error can lead to very flat simplices, which is in general highly unfavorable.

Finally, many functions $g(x)$ with the properties of Theorem 3.9 exist, e.g., polynomials of the kind $(x-a_1)(x-a_2)(x-a_3)$, and therefore many corresponding functions $f(x,y)$. In order to circumvent Theorem 3.9, the refinement procedure has to add linearization points on the edge e from the theorem. Hence, in terms of global optimization and controlling the approximation error, the refinement strategy adding linearization points on the longest edge of a simplex, such as the longest-edge bisection, is to some extent naturally motivated by Theorem 3.9.

3.4.2. Longest-Edge Bisection. We first show that a refinement procedure that performs a longest-edge bisection as in Algorithm 2 is δ-precise for every $\delta > 0$. We then specify how to obtain a new triangulation from the old one after a refinement step using the generalized incremental method. Please note that with Algorithm 2 hanging nodes can occur, i.e., nodes that are contained in the vertex set of a simplex S, but not in all vertex sets of the simplices that are adjacent to S. A classical longest-edge bisection does not allow for hanging nodes. As pointed out before, we only need an MIP approximation as in (2). This, however, allows for hanging nodes, since we only require that the resulting approximation is continuous on each simplex of the triangulation, but not on the entire triangulation.

Now, let $\widetilde{\mathcal{T}}_k$ be the refined triangulation of an initial triangulation $\mathcal{T}_0$ of D_f obtained by applying Algorithm 2 in such a way that in every iteration $i \leq k$ all simplices of $\widetilde{\mathcal{T}}_{i-1}$ are refined.

LEMMA 3.10. *Let $S \in \mathbb{R}^d$ be a simplex of $\mathcal{T}_0$ and e the longest edge of S. Then, the longest edge of any simplex of $\widetilde{\mathcal{T}}_{l\binom{d+1}{2}}$ contained in (the set) S is bounded by $\left(\frac{\sqrt{3}}{2}\right)^l \|e\|$ with $l \in \mathbb{N}$.*

PROOF. In a refinement step of Algorithm 2, the newly introduced midpoint of the longest edge is connected to all opposing $d-1$ vertices. Let $\tilde{e}$ be one of these $d-1$ edges that are added in the first refinement step. Since $\tilde{e}$ can also be considered as the median of the triangle T described by $\bar{x}_a, \bar{x}_b$, where $\operatorname{conv}(\{\bar{x}_a, \bar{x}_b\}) = e$ (as in Algorithm 2) and a vertex $\bar{x}_c$ of the remaining $d-1$ vertices of S, with Apollonius' theorem it follows that

$$\|\tilde{e}\| \leq \frac{\sqrt{3}}{2}\|e\|. \tag{11}$$

Because a longest edge is halved in a refinement step and there are at most $\binom{d+1}{2}$ longest edges in S, the length of the longest edge of any simplex of $\widetilde{\mathcal{T}}_{\binom{d+1}{2}}$ contained in S is bounded by (11). Applying this argument recursively, we see that after $l\binom{d+1}{2}$

Algorithm 2 Longest-edge bisection on a simplex S

Input: A simplex S and a scalar δ.
Output: If the longest edge of S is greater than δ, then a set of two simplices $\{S', S''\}$ with $S = S' \cup S''$ and $\operatorname{int}(S') \cap \operatorname{int}(S'') = \emptyset$ is returned. Otherwise no refinement is performed.

```
Set e ← longest edge of S with endpoints x̄_a, x̄_b ∈ V(S).
if ‖e‖ ≤ δ then
    return V(S).
else
    Set x̃ ← midpoint of the longest edge e.
    Set V(S') ← (V(S) \ x̄_b) ∪ x̃ ;  S' ← conv(V(S')).
    Set V(S'') ← (V(S) \ x̄_a) ∪ x̃ ;  S'' ← conv(V(S'')).
    return V(S'), V(S'').
end if
```

refinement steps any edge of a simplex of $\widetilde{\mathcal{T}}_{l\binom{d+1}{2}}$ that is contained as a set in the set S is bounded from above by $\left(\frac{\sqrt{3}}{2}\right)^l \|e\|$. □

We point out that Definition 3.6 is a parametrized version of the notion of exhaustiveness from Horst and Tuy (1996, Definition IV.10) and formally more convenient in our case. The same applies to Lemma 3.10, which is similar to Horst and Tuy (1996, Proposition IV.2) and adapted to our context. In Horst and Tuy (1996) the authors use the concept of exhaustiveness of a refinement procedure to prove convergence of their simplicial branch-and-bound methods. In a slightly different way, we prove convergence of Algorithm 2, which, unlike the simplicial branch-and-bound methods in Horst and Tuy (1996), allows for integer variables and non-convex nonlinear functions that are not necessarily Lipschitz-continuous. Moreover, our approach uses multiple branch-and-bound trees while the methods in Horst and Tuy (1996) are based on a single branch-and-bound tree.

THEOREM 3.11. *Let $\delta > 0$, then there is an $\tilde{N} \in \mathbb{N}$ such that $\widetilde{\mathcal{T}}_{\tilde{N}}$ is a refinement of every triangulation obtained by applying Algorithm 2 to $\mathcal{T}_0$ with δ as input parameter.*

PROOF. We focus only on $d \geq 2$, because in the case $d = 1$ the theorem obviously holds by interval bisection. With $l = \max\left\{0, \left\lceil \ln\left(\frac{\|e_0\|}{\delta}\right) / \ln\left(\frac{2}{\sqrt{3}}\right) \right\rceil\right\}$, applying Lemma 3.10, we conclude that after at most

$$\tilde{N} := \binom{d+1}{2} \max\left\{0, \left\lceil \frac{\ln\left(\frac{\|e_0\|}{\delta}\right)}{\ln\left(\frac{2}{\sqrt{3}}\right)} \right\rceil\right\}$$

refinement steps the longest edge of any simplex of $\widetilde{\mathcal{T}}_{\tilde{N}}$ is bounded by δ, where $\delta > 0$ and e_0 is the longest edge of all simplices of $\mathcal{T}_0$. Since Algorithm 2 only refines simplices with a longest edge larger than δ and no simplex in $\widetilde{\mathcal{T}}_{\tilde{N}}$ has an edge longer than δ, it

follows by the pigeonhole principle that no refinement of $\mathcal{T}_0$ obtained by Algorithm 2 can be finer than $\widetilde{\mathcal{T}}_{\tilde{N}}$. □

THEOREM 3.12. *Algorithm* 2 *as refinement procedure in Algorithm 1 is* δ*-precise for every* $\delta > 0$*. With* $\tilde{N}$ *as in Theorem* 3.11*, the number of refinement steps* N *as in Definition* 3.6 *is bounded from above by*

$$N \coloneqq m(2^{\tilde{N}} - 1) + 1, \tag{12}$$

where m *is the number of simplices contained in* $\mathcal{T}_0$*.*

PROOF. Counting every single simplex that has to be refined in order to obtain the triangulation $\widetilde{\mathcal{T}}_{\tilde{N}}$ from $\mathcal{T}_0$, we get

$$m(1 + 2 + 4 + \ldots + 2^{\tilde{N}-1}) = m(2^{\tilde{N}} - 1)$$

refinements in total. Again, by the pigeonhole principle, it follows that every sequence $(S^i) \in \mathcal{T}_i$ of simplices has an element S^k with index $k \leq m(2^{\tilde{N}} - 1) + 1$, such that $S^k \in \widetilde{\mathcal{T}}_{\tilde{N}}$, since $\widetilde{\mathcal{T}}_{\tilde{N}}$ is a refinement of every triangulation obtained by Algorithm 2 with parameter δ. Therefore, simplex S^k has property (9), as no simplex in $\widetilde{\mathcal{T}}_{\tilde{N}}$ has an edge longer than δ. □

We now show that a piecewise linear approximation obtained by Algorithm 2 can be modeled by the generalized incremental model.

THEOREM 3.13. *Let* $\mathcal{T}$ *be a triangulation with the properties* (O1) *and* (O2)*. Then, any triangulation* $\mathcal{T}'$ *obtained by applying Algorithm* 2 *to* $\mathcal{T}$ *maintains the properties* (O1) *and* (O2)*.*

PROOF. With $\mathcal{T} = \{S_1, \ldots, S_n\}$, let S_k be the simplex that has to be refined and $\bar{x}_0^{S_k}, \ldots, \bar{x}_d^{S_k}$ its labeled vertices. One of the two simplices S'_k and S''_k (as in Algorithm 2) contains $\bar{x}_0^{S_k}$, whereas the other one contains $\bar{x}_d^{S_k}$, since $\mathcal{V}(S'_k)$ and $\mathcal{V}(S''_k)$ only differ in one vertex. Without loss of generality, let $\bar{x}_0^{S_k}$ be contained in S'_k and $\bar{x}_d^{S_k}$ in S''_k. We now order the simplices of $\mathcal{T}'$ as

$$(S_1, \ldots, S_{k-1}, S'_k, S''_k, S_{k+1}, \ldots, S_n). \tag{13}$$

With the midpoint $\tilde{x}$, we apply the following labeling:

$$\bar{x}_0^{S'_k} = \bar{x}_0^{S_k},\ \bar{x}_d^{S'_k} = \tilde{x}; \qquad \bar{x}_0^{S''_k} = \tilde{x},\ \bar{x}_d^{S''_k} = \bar{x}_d^{S_k}; \tag{14}$$

see Figure 3.5 for an illustration in case of $d = 2$. Furthermore, S_{k-1} and S'_k are linked by $\bar{x}_d^{S_{k-1}} = \bar{x}_0^{S'_k}$, and S''_k and S_{k+1} by $\bar{x}_d^{S''_k} = \bar{x}_0^{S_{k+1}}$. Therefore, the ordering (13) of the simplices of $\mathcal{T}'$ has the properties (O1) and (O2), because $S'_k \cap S''_k \neq \emptyset$ trivially holds, $\bar{x}_d^{S'_k} = \bar{x}_0^{S''_k}$ as in (14) and the rest is inherited from $\mathcal{T}$. □

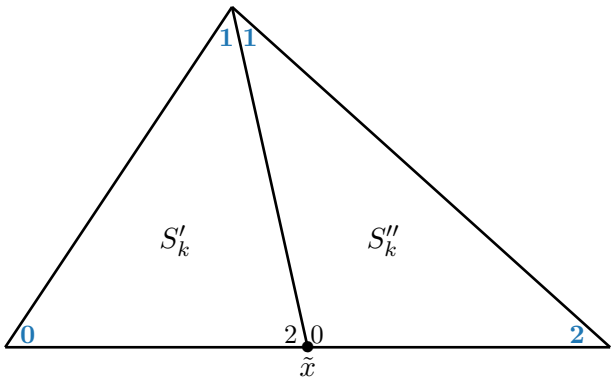


FIGURE 3.5. A refinement of a two-dimensional simplex by the longest-edge bisection with corresponding ordering and labeling of the two sub-simplices. The labeled vertices of the original simplex that is refined are marked in bold blue.

Please note that if no initial triangulation is given, we can simply choose a standard triangulation with vertices equal to the set of extreme points of D_f. We refer to the book by O'Rourke (1998) for more details on this topic. As shown in Geißler (2011), such a triangulation has always an ordering satisfying (O1) and (O2). With higher dimension d of D_f, however, even the triangulation of D_f itself becomes intractable. Let $T(d)$ be the number of simplices in a minimum-cardinality triangulation of the d-cube. Known results are for instance

$$T(2) = 2, \quad T(3) = 5, \quad T(4) = 16, \quad T(5) = 67, \quad T(6) = 308, \quad T(7) = 1493$$

and more generally, at least

$$T(d) \geq \left\lceil \frac{6^{\frac{d}{2}} d!}{2(d+1)^{\frac{d+1}{2}}} \right\rceil$$

simplices are needed for a triangulation of the d-cube; see W. D. Smith (2000). It is therefore imperative to use expression trees even at low dimensions to keep the MINLP problem computationally within reach.

REMARK 3.14. We easily obtain a triangulation $\mathcal{T}(\phi_f^{i+1})$ of D_f by replacing a simplex S in $\mathcal{T}(\phi_f^i)$ with S' and S'', since $S = S' \cup S''$. Hence, there is a piecewise linear approximation $\phi_f^{i+1}(x)$ for every $x \in D_f$.

The next corollary follows directly from Theorem 3.7 and Theorem 3.12.

COROLLARY 3.15. *Algorithm* 1 *together with* Algorithm 2 *as refinement procedure is correct and terminates after a finite number of steps.*

We point out that for a continuous function $f \in \mathcal{F}$, in general, its corresponding δ_f

Algorithm 3 Generalized red refinement of a simplex S

Input: A simplex S with $\mathcal{V}(S) = \{\bar{x}_0, \dots, \bar{x}_d\}$ and a scalar δ.

Output: If the longest edge of S is greater than δ, a set of 2^d simplices $\{S^0, \dots, S^{2^d-1}\}$ with $S = \cup_{i=0}^{2^d-1} S^i$ and $\operatorname{int}(S^i) \cap \operatorname{int}(S^j) = \emptyset$ for all $i \neq j$ is returned. Otherwise no refinement is performed.

Set $e \leftarrow$ longest edge of S.
if $\|e\| \leq \delta$ **then**
  return $\mathcal{V}(S)$.
else
  Set $i \leftarrow -1$.
  for $0 \leq k \leq d$ **do**
    Set $v_0 \leftarrow \frac{1}{2}(\bar{x}_0 + \bar{x}_k)$.
    for $\tau \in \mathrm{Sym}_d$ **do**
      if $\tau^{-1}(1) < \cdots < \tau^{-1}(k)$ **and** $\tau^{-1}(k+1) < \cdots < \tau^{-1}(d)$ **then**
        for $1 \leq l \leq d$ **do**
          Set $v_l \leftarrow v_{l-1} + \frac{1}{2}(\bar{x}_{\tau(l)} - \bar{x}_{\tau(l)-1})$.
        end for
        Set $i \leftarrow i + 1$.
        Set $\mathcal{V}(S^i) \leftarrow \{v_0, \dots, v_d\}$; $S^i \leftarrow \operatorname{conv}(\mathcal{V}(S^i))$.
      end if
    end for
  end for
  return $\mathcal{V}(S^0), \dots, \mathcal{V}(S^{2^d-1})$.
end if

as in the proof of Theorem 3.7 might not be computable. Since such a $\delta_f > 0$ exists for every continuous $f \in \mathcal{F}$, it suffices to set $\delta_f := 0$ as input parameter for Algorithm 2. In practice, however, the lower bound of an attainable δ is determined by the capabilities of the MIP solver.

3.4.3. Generalized Red Refinement. Another well-known refinement procedure is the red refinement. In this thesis, we consider the simplex refinement algorithm 3 by Bey (2000) and Freudenthal (1942). It is a generalization of the red refinement strategy, which originally was only developed for triangles. It is shown by the authors that the generalized red refinement procedure always delivers a triangulation of the simplex that has to be refined by 2^d sub-simplices. Moreover, the triangulation is *consistent*, i.e., the intersection of any two sub-simplices is either empty or a common lower-dimensional simplex with respect to the vertex sets. Consequently, a consistent triangulation does not allow for hanging nodes.

First, we illustrate the refinement by Algorithm 3 using an example in dimension two.

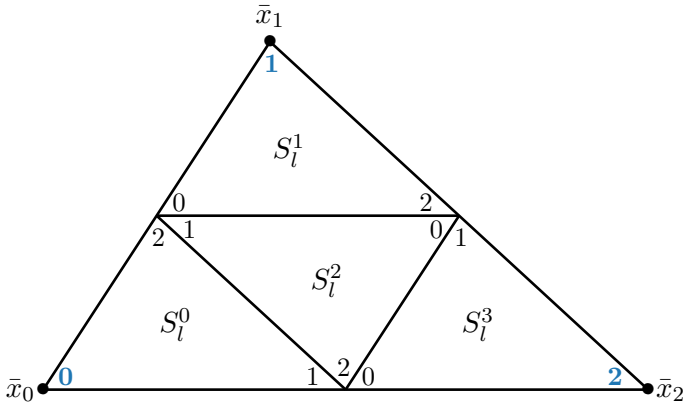


FIGURE 3.6. A red refinement of a two-dimensional simplex with corresponding ordering and labeling of the four sub-simplices. The labeled vertices of the original simplex that is refined are marked in bold blue.

EXAMPLE 3.16. *We consider a simplex S_l of some triangulation of a two-dimensional nonlinear function with vertex set $\mathcal{V}(S_l) = \{\bar{x}_0, \bar{x}_1, \bar{x}_2\}$ that has to be refined. Let the scalar δ be sufficiently large such that a refinement is performed by Algorithm 3. Note that for $k \leq 1$ the condition $\tau^{-1}(1) < \cdots < \tau^{-1}(k)$ is considered to be fulfilled. The same applies for the condition $\tau^{-1}(k+1) < \cdots < \tau^{-1}(d)$ in case of $k \geq d-1$.*

First, the symmetry group Sym_2 has two permutations: the identity $\tau_1 = \mathrm{id}$ and the permutation $\tau_2 \colon \{1,2\} \to \{2,1\}$. The identity fulfills the condition in Line 9 for all $0 \leq k \leq 2$.

Since τ_2 does not satisfy Line 9 for $k = 0$, we obtain the first corner sub-simplex S_l^0 for τ_1. The vertices of S_l^0 are

$$v_0 = \bar{x}_0, \quad v_1 = v_0 + \frac{1}{2}(\bar{x}_1 - \bar{x}_0), \quad v_2 = v_1 + \frac{1}{2}(\bar{x}_2 - \bar{x}_1); \tag{15}$$

see Figure 3.6 for an illustration.

For $k = 1$ both permutations τ_1 and τ_2 comply with Line 9. Equivalent to $k = 0$, with τ_1 we obtain the simplex S_l^1 as in (15), while now $v_0 = \bar{x}_0 + \frac{1}{2}(\bar{x}_1 - \bar{x}_0)$. We thus obtain the corner sub-simplex S_l^1 simply by translating the corner sub-simplex S_l^0 by the vector $\frac{1}{2}(\bar{x}_1 - \bar{x}_0)$. For τ_2, we compute the vertices of the simplex S_l^2 as

$$v_0 = \frac{1}{2}(\bar{x}_1 - \bar{x}_0), \quad v_1 = v_0 + \frac{1}{2}(\bar{x}_2 - \bar{x}_1), \quad v_2 = v_1 + \frac{1}{2}(\bar{x}_1 - \bar{x}_0). \tag{16}$$

Finally, for $k = 1$ again only the identity τ_1 fulfills the condition in Line 9. We obtain the simplex S_l^3 as in (15) with $v_0 = \bar{x}_0 + \frac{1}{2}(\bar{x}_2 - \bar{x}_0)$. The corner sub-simplex S_l^3 again corresponds to a translation of S_l^0 by the vector $\frac{1}{2}(\bar{x}_2 - \bar{x}_0)$.

We now proceed as in the previous subsection. First, we prove the δ-preciseness

of the refinement procedure and show how to model a refined triangulation by the generalized incremental method afterward.

Again, let $\widetilde{\mathcal{T}}_k$ be the refined triangulation of an initial triangulation $\mathcal{T}_0$ of D_f obtained by applying Algorithm 3 such that in every iteration $i \leq k$ all simplices of $\widetilde{\mathcal{T}}_{i-1}$ are refined.

LEMMA 3.17. *Let $S \in \mathbb{R}^d$ be a simplex of $\mathcal{T}_0$ and e the longest edge of S. Then, the longest edge of any simplex of $\widetilde{\mathcal{T}}_l$ contained in (the set) S is bounded by $\frac{1}{2^l}\|e\|$ with $l \in \mathbb{N}$.*

PROOF. The lemma follows directly from Line 11 of Algorithm 3, since

$$\left\| \frac{1}{2}(\bar{x}_{\tau(l)} - \bar{x}_{\tau(l)-1}) \right\| \leq \frac{1}{2}\|e\|$$

are the lengths of the edges of the sub-simplices that are constructed during the first refinement step. Applying this recursively finishes the proof. □

With Lemma 3.17 and the results from Section 3.4.2 the δ-preciseness of the red refinement follows directly:

COROLLARY 3.18. *Let $\delta > 0$, then there is an $\tilde{N} \in \mathbb{N}$, such that $\widetilde{\mathcal{T}}_{\tilde{N}}$ is a refinement of every triangulation obtained by applying Algorithm 3 to $\mathcal{T}_0$ with δ as input parameter.*

PROOF. With $\tilde{N} \coloneqq l = \max\{0, \lceil \ln(\frac{\|e_0\|}{\delta})/\ln(2) \rceil\}$ and applying Lemma 3.17, the proof works equivalently to the one of Theorem 3.11. □

COROLLARY 3.19. *Algorithm 3 as refinement procedure in Algorithm 1 is δ-precise for every $\delta > 0$. With $\tilde{N}$ as in Corollary 3.18, the number of refinement steps N as in Definition 3.6 is bounded from above by*

$$N \coloneqq m(2^{\tilde{N}d} - 1) + 1,$$

where m is the number of simplices contained in $\mathcal{T}_0$.

PROOF. We again count every single simplex that has to be refined and obtain

$$m(1 + 2^d + 2^{2d} + \ldots + 2^{\tilde{N}d-1}) = m(2^{\tilde{N}d} - 1)$$

refinements in total. The rest of the proof follows by the pigeonhole principle as in the proof of Theorem 3.12. □

We now show that a piecewise linear approximation that results from applying Algorithm 3 can also be modeled with the generalized incremental method. We first prove two lemmata that are used afterward to prove the main result of this subsection.

LEMMA 3.20. *Let $\mathcal{S} = \{S^0, \ldots, S^{2^d-1}\}$ be a refinement of a simplex S by Algorithm 3 with $\mathcal{V}(S) = \{\bar{x}_0, \ldots, \bar{x}_d\}$. Then, each simplex of the subset of the corner sub-simplices*

$\mathcal{S}' = \{S^{i_0}, \dots, S^{i_d}\}$ of $\mathcal{S}$ contains a vertex of the simplex S, i.e., $\bar{x}_j \in \mathcal{V}(S^{i_j})$ for all $j = 0, \dots, d$. Moreover, for each pair of simplices $S^{i_j}, S^{i_k} \in \mathcal{S}'$ the midpoint m_{jk} of the edge with endpoints $\bar{x}_j$ and $\bar{x}_k$ is contained in both vertex sets of the simplices.

PROOF. First, the identity $\mathrm{id} \in \mathrm{Sym}_d$ always fulfills the conditions from Line 9 of Algorithm 3. Let S^{i_j} be the simplex that is constructed using the starting vertex $v_0 = \frac{1}{2}(\bar{x}_0 + \bar{x}_j)$ and $\tau = \mathrm{id}$, where $j = 0, \dots, d$. Due to the telescoping sum in Line 11, it follows that

$$v_j = \underbrace{\underbrace{\underbrace{\frac{1}{2}(\bar{x}_0 + \bar{x}_j)}_{v_0} + \frac{1}{2}(\bar{x}_1 - \bar{x}_0)}_{v_1} + \frac{1}{2}(\bar{x}_2 - \bar{x}_1)}_{\vdots} + \cdots + \frac{1}{2}(\bar{x}_j - \bar{x}_{j-1}) = x_j \tag{17}$$

is contained in the vertex set of S^{i_j}.

Furthermore, due to the telescoping sum in (17), we can rewrite Line 11 as

$$v_l \leftarrow \frac{1}{2}(\bar{x}_0 + \bar{x}_j) + \frac{1}{2}(\bar{x}_l - \bar{x}_0) = \frac{1}{2}(\bar{x}_j + \bar{x}_l)$$

for $\tau = \mathrm{id}$. Since $m_{jk} = \frac{1}{2}(\bar{x}_j + \bar{x}_k)$, we conclude that the vertices v_k and v_j that occur during the construction of S^{i_j} and S^{i_k}, respectively, are equal to m_{jk}. □

LEMMA 3.21. *Let $\mathcal{S}'' = \{S^0, \dots, S^{2^d - 1 - (d+1)}\}$ be a refinement of a simplex S by Algorithm 3 without the $d+1$ corner sub-simplices of $\mathcal{S}'$ as in Lemma 3.20. Then, the union of all simplices of $\mathcal{S}''$ is a (convex) polytope and the triangulation of $\mathcal{S}''$ has an ordering with the properties* (O1) *and* (O2).

PROOF. Alternatively to the vertex description, we can describe the simplex S by its half-space representation

$$S = \left\{ x \in \mathbb{R}^d \,:\, a_j^\top x \leq b_j \text{ with } a_j \in \mathbb{R}^d \text{ and } b_j \in \mathbb{R} \text{ for } j = 0, \dots, d \right\}. \tag{18}$$

We now describe the set $\mathcal{S}''$ by adding the inequalities that separate all corner sub-simplices from the set S as in (18). For a vertex of S exactly d inequalities in (18) are tight. Due to Lemma 3.20, the vertex set of the corner sub-simplex $S^{i_j} \in \mathcal{S}'$ consists of the vertex $\bar{x}_j$ and all midpoints m_{jk} with $k = 0, \dots, d$. Let $a_j^\top x \leq b_j$ be the inequality that is not tight for $\bar{x}_j$. Naturally, it is tight for all other vertices of S and it follows that

$$a_j^\top m_{jk} = \frac{1}{2}\, a_j^\top (\bar{x}_k + \bar{x}_j) = \frac{1}{2}(b_j + a_j^\top \bar{x}_j). \tag{19}$$

Moreover, since the red refinement also delivers a triangulation of S, where all interiors of the sub-simplices are disjunct, we can describe S^{i_j} by substituting $a_j^\top x \leq b_j$ with $a_j^\top x \leq \frac{1}{2}(b_j + a_j^\top \bar{x}_j)$ in (18). Therefore, by separating all corner sub-simplices, we

obtain the half-space description

$$\mathcal{S}'' = \Big\{ x \in \mathbb{R}^d \,:\, a_j^\top x \le b_j,\ a_j^\top x \ge \frac{1}{2}(b_j + a_j^\top \bar{x}_j) \\ \text{with } a_j \in \mathbb{R}^d \text{ and } b_j \in \mathbb{R} \text{ for } j = 0, \dots, d \Big\}.$$

It follows that the union of all simplices of $\mathcal{S}''$ is a convex polytope.

The consistency of the triangulation of S translates to the triangulation of $\mathcal{S}''$, because only the corner sub-simplices are omitted. It is shown by Geißler (2011) that a triangulation with the property that for each nonempty subset $\tilde{\mathcal{S}} \subsetneq \mathcal{T}$ of a triangulation $\mathcal{T}$ there exist simplices $S_1 \in \tilde{\mathcal{S}}$ and $S_2 \in \mathcal{T} \setminus \tilde{\mathcal{S}}$ such that S_1, S_2 have d common vertices, always yields a triangulation with the properties (O1) and (O2). Therefore, we only have to prove that the triangulation of $\mathcal{S}''$ has this property. To this end, we consider a nonempty subset $\tilde{\mathcal{S}}$ of $\mathcal{S}''$. Each facet of a d-dimensional simplex consists of d vertices of the simplex. Let $\tilde{\mathcal{S}}_\mathrm{F}$ be the set of all facets of the simplices of $\tilde{\mathcal{S}}$, where the simplices are described by the convex hull of its vertices. Then, each facet in $\tilde{\mathcal{S}}_\mathrm{F}$ is either a facet of $\mathcal{S}''$ (in the sense of convex hulls) or a common facet of two simplices of $\tilde{\mathcal{S}}$. This is due to the consistency of the triangulation. Since $\tilde{\mathcal{S}} \subsetneq \mathcal{S}''$, however, there must be a facet in $\tilde{\mathcal{S}}_\mathrm{F}$ that is a common facet of two simplices S_1, S_2 such that $S_1 \in \tilde{\mathcal{S}}$ and $S_2 \in \mathcal{S}'' \setminus \tilde{\mathcal{S}}$. □

Endowed with Lemma 3.20 and 3.21, we are now ready to prove the main result of this subsection.

THEOREM 3.22. *Let $\mathcal{T}$ be a triangulation with the properties* (O1) *and* (O2)*. Then, any triangulation $\mathcal{T}'$ obtained by applying Algorithm* 3 *to $\mathcal{T}$ maintains the properties* (O1) *and* (O2)*.*

PROOF. For the following proof, we first show that there is an ordering of the sets $\mathcal{S}'$ and $\mathcal{S}''$ from Lemma 3.20 and 3.21 with the properties (O1) and (O2). The second part of the proof merges these two orderings to obtain an overall ordering with the properties (O1) and (O2).

Let $\mathcal{T} = \{S_1, \dots, S_n\}$ and S_l be the simplex that has to be refined, while $\bar{x}_0^{S_l}, \dots, \bar{x}_d^{S_l}$ are its labeled vertices. We first consider $d \ge 4$ and $d = 2, 3$ afterward. The 2^d simplices $S_l^0, \dots, S_l^{2^d-1}$, into which S_l is divided by the red refinement, have due to Lemma 3.20 the property that the corner sub-simplices contain the vertices of S_l. Without loss of generality, let $\bar{x}_j^{S_l} \in \mathcal{V}(S_l^j)$.

We first show that the set $\mathcal{S}'$ of the corner sub-simplices has an ordering with the required properties. The corner sub-simplices $S_l^j \in \mathcal{S}'$ yield a complete graph $G = (V, E)$ with the simplices as the node set V and the midpoints m_{jk} as the edge set E connecting the simplices S_l^j and S_l^k. Due to $m_{jk} = m_{kj}$, we assume in the following for

the notation m_{jk} that $j < k$ holds. For each Hamiltonian path in G, we can use the path itself as an ordering of the simplices that correspond to the nodes of G. An edge connecting two consecutive nodes of the path corresponds to a common midpoint of two consecutive simplices. Therefore, the ordering naturally has property (O1), which indicates that two consecutive simplices have at least one common vertex. The ordering has the property (O2) as well, which states that the last vertex of any simplex is equal to the first vertex of the next one: With two consecutive simplices S_l^j and S_l^k that correspond to two consecutive nodes of the Hamiltonian path, we only have to set $\bar{x}_d^{S_l^j} = m_{jk}$ and $\bar{x}_0^{S_l^k} = m_{jk}$.

Moreover, it follows from Lemma 3.21 that there is an ordering

$$(S_l^{i_0}, \dots, S_l^{i_{2^d-1-(d+1)}}) \tag{20}$$

of the simplex set $\mathcal{S}''$ with the properties (O1) and (O2). Since the vertices of the simplices of $\mathcal{S}''$ are the midpoints m_{jk}, there must be two midpoints m_{jk} and m_{st} with

$$m_{jk} = \bar{x}_0^{S_l^{i_0}} \qquad \text{and} \qquad m_{st} = \bar{x}_d^{S_l^{i_{2^d-1-(d+1)}}}.$$

We now link the orderings of $\mathcal{S}'$ and $\mathcal{S}''$ to obtain an overall ordering with the properties (O1) and (O2). Let R be a Hamiltonian path in the sub-graph of G that consists of the vertices $V \setminus \{0, j, k, s, t, d\}$. Such a path is always attainable, because any sub-graph of a complete graph is also complete. With $j \neq k$ one of the following three cases applies to the node j: $j = s$, $j = t$, or $j \neq s \wedge j \neq t$. With $s \neq t$, we have the following three cases for the node s: $s = j$, $s = k$, or $s \neq j \wedge s \neq k$. Consequently, due to $j \neq k$ and $s \neq t$ only the following five cases are possible for the nodes j, k, s, and t:

$$j = s \wedge k \neq t, \quad j = s \wedge k = t, \quad j \neq s \wedge k = t, \quad k = s, \quad j \neq s \wedge k \neq t. \tag{21}$$

The case $k = s$ is equivalent to the case $j = t$, since the inverse of the ordering (20) also has the properties (O1) and (O2).

Keeping the cases in (21) in mind, we define the path

$$H = \begin{cases} (0, j, t, R, k, d), & \text{if } j = s \wedge k \neq t, & \text{(22a)} \\ (0, R, j, k, d), & \text{if } j = s \wedge k = t, & \text{(22b)} \\ (0, j, s, R, t, d), & \text{if } k = s, & \text{(22c)} \\ (0, j, s, R, k, t, d), & \text{otherwise.} & \text{(22d)} \end{cases}$$

Please note that if $t = d$ in (22a), we can again invert the ordering (20) such that $t \neq d$ and $k = d$. The same can be applied in case of $k = d$ in (22c) and (22d). Moreover, any permutation of the labeling $\bar{x}_i$ of the simplex $S_l^{i_0}$, where $i = 0, \dots, d-1$ and of the simplex $S_l^{i_{2^d-1-(d+1)}}$, where $i = 1, \dots, d$, is permissible. Thus, we can assume that

$m_{st} \neq m_{0d}$ and that $m_{jk} \neq m_{0u}$ if $m_{st} = m_{ud}$. This guarantees us that the path H is always a Hamiltonian path.

The path H corresponds to an ordering, where the vertices in H are the corner sub-simplices $S_l^j \in \mathcal{S}'$. As showed above, this ordering has the properties (O1) and (O2). We now insert the ordering (20) of the simplex set $\mathcal{S}''$ into the one of H after the simplex that corresponds to the vertex j. The union of $\mathcal{S}'$ and $\mathcal{S}''$ corresponds to the set of all 2^d sub-simplices into which S_l is divided by the red refinement. Therefore, the resulting ordering covers all 2^d sub-simplices. The merging of the two orderings of $\mathcal{S}'$ and $\mathcal{S}''$ via the midpoints m_{jk} and m_{st} finally leads to an overall ordering that has the properties (O1) and (O2). The remainder of the proof works equivalently to the one of Theorem 3.13.

In case of $d = 2$, we use the ordering $(S_l^0, S_l^1, S_l^2, S_l^3)$, where S_l^0, S_l^1, and S_l^3 are the three corner sub-simplices and S_l^2 the remaining center sub-simplex; see again Figure 3.6 for an illustration.

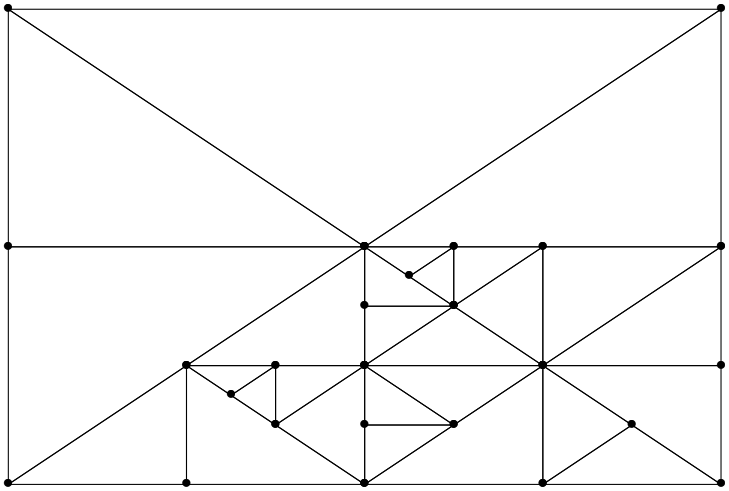
For $d = 3$, we assume without loss of generality that the vertex $\bar{x}_3$ of the simplex that has to be refined is contained in S_l^7. We use the ordering $(S_l^0, S_l^1, S_l^2, S_l^3, S_l^4, S_l^5, S_l^6, S_l^7)$, where S_l^0, S_l^1, S_l^2, and S_l^7 are the four corner sub-simplices contained in $\mathcal{S}'$ and S_l^3–S_l^6 the four center sub-simplex of $\mathcal{S}''$. Finally, the orderings of $\mathcal{S}'$ and $\mathcal{S}''$ are linked via $m_{jk} = m_{02}$ and $m_{st} = m_{27}$.

The rest of the proof again works equivalently to the one of Theorem 3.13. □

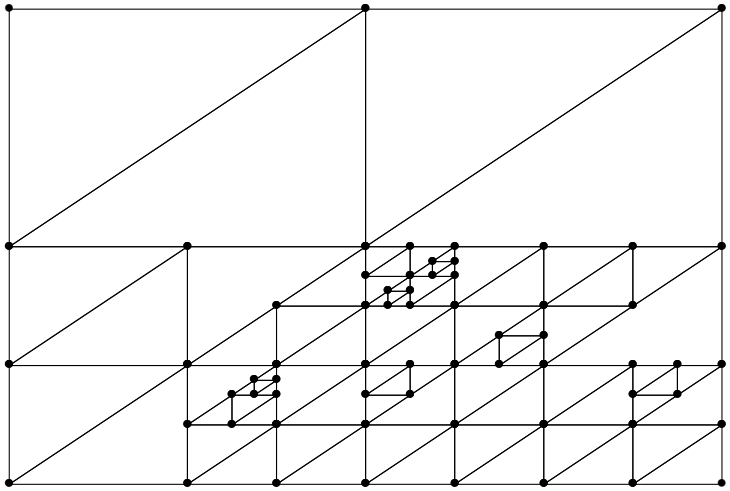
3.4.4. Comparison of the Methods. In this section, we compare the longest-edge bisection and red refinement with the procedure that adds points with maximal approximation error from Section 3.4. To this end, we again use the two-dimensional nonlinear function $f(x, y) = \sin(x)e^y$ and the domain $D_f = [0, 2\pi] \times [0, 2\pi]$ and generate 25 random points in the lower half ($y \leq \pi$) of the domain D_f. We refine the simplices that contain these points and subsequently construct new triangulations.

First, we can see in Figure 3.7a and 3.7b that both the longest-edge bisection and the red refinement yield triangulations that prevent flat simplices. In contrast, the procedure adding points with maximal approximation error can lead to very flat simplices; see Figure 3.4 again.

Moreover, we give more insight into the quality of the triangulations using the maximal approximation errors ϵ_i on the simplices of the triangulation. Table 3.1 shows a comparison of common error measures. The first column indicates the median of the errors ϵ_i, while the arithmetic mean of ϵ_i is given in the second column. The third and fourth column contain the minimum and maximum of ϵ_i. We conclude that in our example $f(x, y) = \sin(x)e^y$ after 25 refinement steps the red refinement delivers the triangulation with the smallest approximation errors. The longest-edge bisection



(A) Triangulation using the longest-edge bisection.



(B) Triangulation using the red refinement.

FIGURE 3.7. Triangulation of the domain $D_f = [0, 2\pi] \times [0, 2\pi]$ of the nonlinear function $f(x, y) = \sin(x)e^y$ obtained by applying the longest-edge bisection (above) and red refinement (below).

and the procedure adding points with maximal approximation error perform very similarly with respect to the approximation errors.

The number of simplices of a triangulation, however, is crucial for the applicability of our MIP-based approach, since the run time for solving an MIP grows exponentially with its size and each simplex corresponds to a binary variable. In our example $f(x, y) = \sin(x)e^y$, the red refinement produces 77 simplices, while the procedure adding points with maximal approximation error and the longest-edge bisection only need 52 simplices and 27 simplices, respectively. In this regard, the longest-edge

TABLE 3.1. Various error measures using the maximal approximation errors on the simplices of the triangulations that are obtained by applying the longest-edge bisection, the red refinement, and the procedure adding points with maximal approximation error.

refinement procedure	median	mean	min	max
longest-edge bisection	1.01	34.71	0.18	534.43
red refinement	0.21	19.03	0.02	535.49
points with maximal error	0.95	30.30	0.18	534.43

bisection is the best procedure. More generally, independent of the dimension of the nonlinear function, the longest-edge bisection increases the number of simplices of a triangulation by only one per refinement step. The other procedures, however, lead to higher increases. In the following, we therefore use the longest-edge bisection as refinement procedure in Algorithm 1.

Finally, we point out that the triangulations in Figure 3.7a and 3.7b both yield a Delaunay triangulation. More specific, each simplex of these triangulations is a Delaunay simplex, i.e., there exists a Delaunay triangulation for the set of vertices in Figure 3.7a and 3.7b such that the simplex is contained in the Delaunay triangulation. A more precise characterization of this phenomenon remains the subject of future research.

3.5. Complexity of MIP Relaxations

In this section, we analyze the complexity of the resulting MIP relaxations in Algorithm 1 in terms of their sizes. In our context, complexity is narrowed down to the maximal number of continuous and binary variables. In the following, we consider a fixed dimension of the box domain D_f of a nonlinear function f. With higher dimension, the number of simplices that are needed for the triangulation of D_f itself drastically increases and therefore dominates the number of refinements performed by Algorithm 1.

As we have seen in the previous section and due to Theorem 3.12, using the generalized incremental method the number of both continuous and binary variables is of size $\mathcal{O}(2^{\tilde{N}})$ or, with $\tilde{N} = \mathcal{O}(d^2|\ln(\operatorname{diam}(D_f)/\delta_f)|)$, of size

$$\mathcal{O}((\operatorname{diam}(D_f)/\delta_f)^{d^2}).$$

In order to give more exact bounds, we assume for the rest of this section that $f \in \mathcal{F}$ is Lipschitz-continuous with Lipschitz-constant L_f. This is a common assumption in terms of practically interesting MINLP problems. With regard to the convergence of Algorithm 1, we get the following result.

THEOREM 3.23. *Let $f \in \mathcal{F}$ be Lipschitz-continuous with Lipschitz-constant L_f. Then, it suffices to set $\delta_f := \epsilon_f/(2L_f)$ as input parameter in Algorithm 2, such that Algorithm 1 together with Algorithm 2 as refinement procedure is correct and terminates after a finite number of steps.*

PROOF. The proof works almost analogously to the proof of Theorem 3.7. With $\delta_f = \epsilon_f/(2L_f)$, now $|f(\bar{x}) - \phi_f^N(\bar{x})| < \epsilon_f/2$ holds by applying Lipschitz-continuity of f. □

Regarding the number of refinements performed in Algorithm 1, we now have

$$\tilde{N} = \mathcal{O}\left(d^2 \left| \ln\left(\frac{L_f \operatorname{diam}(D_f)}{\epsilon_f}\right)\right|\right), \tag{23}$$

where $\tilde{N}$ is defined as in Theorem 3.11, and a maximal number of variables of size

$$\mathcal{O}\left((L_f \operatorname{diam}(D_f)/\epsilon_f)^{d^2}\right). \tag{24}$$

These numbers are scale-invariant, since with the scaling of the nonlinear function f, the Lipschitz-constant L_f is also scaled accordingly. A classical result on the complexity of computing approximations of f to within ϵ_f is $\Theta((1/\epsilon_f)^d)$ by Traub et al. (1988). Compared to this, we see that a longest-edge bisection might not yield an approximation of f with minimal cost, since every binary variable corresponds to a refinement step and therefore an evaluation of f. However, because we only need an optimal solution of an MIP relaxation that satisfies all a priori given approximation errors, locally fine approximations are sufficient in our case. The adaptivity in Algorithm 1 aims at obtaining good locally fine approximations. The longest-edge bisection rule increases the number of binary variables by only one per refinement step for each nonlinear function and is thus very convenient with respect to the adaptive refinements.

As shown by Vielma et al. (2010) and Vielma and Nemhauser (2011), a modeling of the MIP relaxations obtained by Algorithm 1 with a logarithmic number of binary variables is possible. Their so-called logarithmic disaggregated convex combination model is only based on a triangulation of D_f as in Definition 3.2 and does not impose any other special requirements. Therefore, with Remark 3.14 we can easily use the model for our MIP relaxations. Hence, concerning the number of binary variables, we get $\mathcal{O}(\tilde{N})$ with $\tilde{N}$ as in (23), whereas the number of continuous variables equals the one of the generalized incremental method and is thus of size $\mathcal{O}(2^{\tilde{N}})$. However, we point out that in practice the generalized incremental method might be superior in terms of the run time; see again Correa-Posada and Sánchez-Martín (2014) and Geißler (2011) for more detailed discussions.

The logarithmic branching convex combination model described by Vielma and Nemhauser requires a smaller amount of continuous variables than the logarithmic

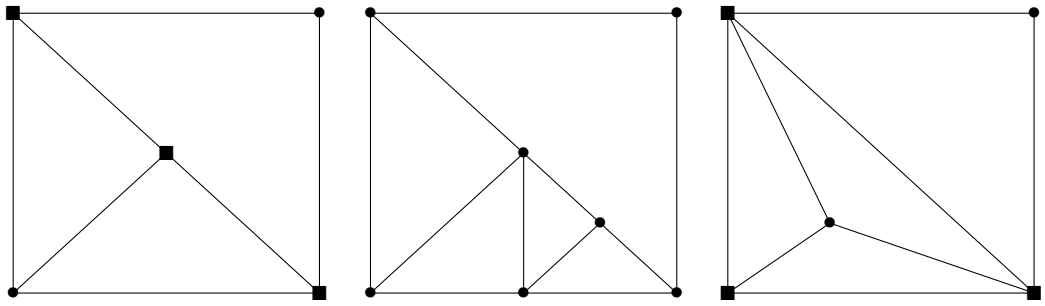


FIGURE 3.8. Triangulations of a two-dimensional box for which only a 3-way IB scheme (left and right), and a 2-way IB scheme (middle) exist. In case of 3-way IB schemes, a minimal infeasible set of size three is marked by squares.

disaggregated convex combination model. This is accomplished by using branching dichotomies, which decide if a variable corresponding to a vertex of the triangulation is set to zero or not. In this way, only one continuous variable is needed for each vertex of a triangulation, instead of $d+1$ variables for each d-dimensional simplex of a triangulation as in the logarithmic disaggregated convex combination model. Very recently, the approach of Vielma and Nemhauser has been generalized in the preprint Huchette and Vielma (2016) introducing the notion of k-way independent branching (IB) schemes. Thereby, a 2-way IB scheme is needed for a triangulation in order to utilize the logarithmic branching convex combination model. Furthermore, the existence of k-way IB schemes has been characterized as follows. With $\mathcal{V}$ as the set of all vertices of the triangulation, a subset $V \in \mathcal{V}$ is called infeasible, if V is not contained in the set of all vertices describing some simplex $S \in \mathcal{S}$ of the triangulation, i.e., $V \not\subseteq \mathcal{V}(S)$ for all $S \in \mathcal{S}$.

PROPOSITION 3.24 (Theorem 1, Huchette and Vielma (2016)). *A k-way* IB *scheme exists if and only if each minimal infeasible set $V \in \mathcal{V}$ has $|V| \leq k$.*

Unfortunately, this model is unsuitable for our purpose. We can see in Figure 3.8 that there can be triangulations in Algorithm 1 for which only 3-way IB schemes exist:

PROPOSITION 3.25. *Triangulations occurring in Algorithm 1 do not always have* 2*-way* IB *schemes*

We remark that there is no direct impact of hanging nodes of a triangulation to the existence of 2-way IB schemes.

CHAPTER 4

Optimization of Gas Transport Networks

In this chapter, we derive MINLP problems that arise in the optimization of gas transport networks. For these MINLPs, we demonstrate the applicability of our approach from Chapter 3 in Chapter 6.

Natural gas is one of the most important energy sources worldwide. According to the BP Statistical Review of World Energy 2018, almost a quarter of the world's total energy consumption was covered by natural gas in 2017; see BP Review (2018) and Figure 4.1 for an overview of the distribution of the various energy sources. With constantly increasing energy consumption and the associated increase in gas amounts that are traded, the transport of gas is also becoming more and more of a challenge.

In Germany, natural gas is gaining even more importance in the context of the "turnaround in energy policy". The main goal is the transition from the unsustainable use of fossil fuels and especially nuclear energy towards a sustainable supply with renewable energy. In order to enable an energy supply during the transition to nuclear-free energy, gas will play a decisive role as an energy source in the coming decades. Partially motivated by this, the academic research project "SFB Transregio 154: Mathematical Modelling, Simulation and Optimization using the Example of Gas Networks" supported by the German Research Foundation was initiated in 2014. This

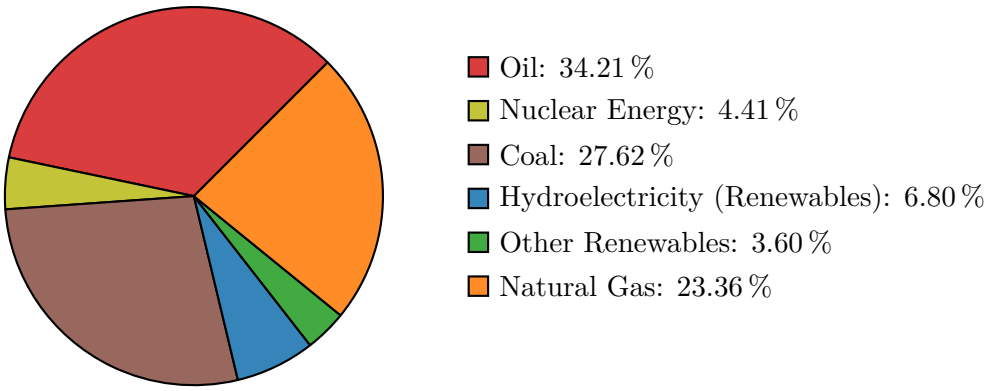


FIGURE 4.1. Overview of the distribution of the various energy sources covering the total energy consumption worldwide according to BP Review (2018).

thesis is to a large extent the result of the subproject B07 of the SFB Transregio 154 project and focuses on one of its central aspects: the optimization of gas transport networks.

Mathematically, the problems related to gas transport are challenging due to the interaction of gas flow and pressure. First, from this coupling of flow and pressure, it follows that there is generally no fixed capacity of a pipe. Therefore, classical network flow theory, e.g., Ford and Fulkerson (1956), cannot describe the characteristics of gas networks. Moreover, gas networks are highly sensitive, since changing the pressures on a small part of the network can have a significant impact on a much larger part of the network.

The chapter is structured as follows. First, in Section 4.1, we motivate the gas optimization problems discussed in this thesis and give an overview of the literature on approaches in the field of gas network optimization and related problems. In Section 4.2, we describe the mathematical model for the transport of gas in a network. Then, we provide a discretization of the partial differential equations (PDEs) of this model in Section 4.3. Endowed with this discretization, we model in Section 4.4 two gas optimization problems as MINLP problems that are the basis for the computational results in Chapter 6.

This chapter is mainly based on the book Koch et al. (2015) and the publication

> R. Burlacu, H. Egger, M. Groß, A. Martin, M. E. Pfetsch, L. Schewe, M. Sirvent, and M. Skutella (2018). "Maximizing the storage capacity of gas networks: a global MINLP approach". In: *Optimization and Engineering*, pp. 1–31. DOI: 10.1007/s11081-018-9414-5.

The author of this thesis made important contributions to the elaboration of the MINLP models and implementation of the approaches in the latter.

4.1. Optimization Issues

In this section, we present two important optimization tasks in the field of gas transport that we will model as MINLP problems in the remainder of this chapter. In addition, we provide a literature overview of various approaches to these and similar optimization problems.

In Europe, on the basis of legislation, gas trading and gas transport are managed by different companies. Gas transport is typically performed by transmission system operators (TSOs). These are supplied with gas by the producers at so-called entries and transport it through a pipeline (or short pipe) network to companies that distribute the gas at so-called exits. A TSO is therefore responsible for ensuring that a certain amount of gas can be transported reliably. In this thesis, we highlight two major optimization tasks of a TSO: the validation of a so-called nomination and the

maximization of the storage capacity of a gas network. We formulate this problems as MINLPs, which we subsequently solve by our adaptive MIP-based approach.

First, since gas trading and gas transport are decoupled, the TSOs sell rights to inject the gas into or withdraw it from the network up to a maximal amount. These maximal amounts are also called a booking. The right holders, for their part, must specify exactly the amount of gas they wish to inject or withdraw at least one day in advance. This specification is called a nomination. Moreover, each nomination must be balanced, i.e., the sum of the injection at all entries must be equal to the sum of the withdrawal at all entries. The nomination validation problem now is to satisfy all right holders with regard to the nominated gas amounts. Here, we are dealing with planning problems on large time scales, where neglecting short-time transient effects is reasonable. We can therefore use a stationary model for the gas physics.

The second optimization issue we address is the problem of maximizing the storage capacity of a gas network, which is of particular interest for TSOs. Especially with regard to Power-to-Gas, this problem becomes essential. In this context, the gas network serves as a storage system by converting electrical power into gas, e.g., by generating hydrogen or methane and feeding it into the gas network. In a typical Power-to-Gas scenario, the generated gas is fed into the network only during a specified period of time, for instance when solar or wind energy is available and is discharged later when there is a high demand. To model these dynamics, such scenarios require a transient model of the gas physics, which makes the problem much more complex compared to the stationary case.

From a mathematical point of view, these problems result in hard optimization problems that are challenging because they combine nonlinear and discrete aspects. First, the nonlinear part originates to a large extent from the physics of the gas, which is generally described by the Euler equations, i.e., a coupled system of nonlinear hyperbolic PDEs. On the other hand, switching active elements, i.e., controllable facilities such as valves and compressors that can have different modes, involves discrete decisions.

To be more precise, in our setting, the nomination validation problem is as follows: Find an admissible configuration of the controllable elements satisfying all physical and technical constraints. Additionally, due to the friction-induced pressure drop on long pipes, we also incorporate compressors increasing gas pressures into the model, which leads to a goal of minimizing the compressor energy. Now, the problem of maximizing the storage capacity of a gas network can be summarized as the following: Maximize the amount of gas that can be fed into the network such that there is an admissible time-dependent control of the active elements satisfying all physical and technical constraints. Furthermore, a specific control has to be computed. We tackle

these global optimization problems by a first-discretize-then-optimize approach. To this end, we discretize the system of nonlinear PDEs that describes the gas physics. Incorporating active elements such as valves and compressors, we subsequently obtain non-convex MINLPs. Our focus lies in particular on globally optimal solutions of these problems. In order to solve these MINLPs in Chapter 6, we utilize the approach from Chapter 3.

In recent years, a lot of literature has been published on such gas optimization problems. We again refer to the book by Koch et al. (2015) that provides a comprehensive survey in case of a stationary gas physics. Recently, alternating direction methods have been used in Geißler et al. (2015b, 2018) to combine MIP and NLP techniques in order to solve stationary gas optimization problems for large-scale real-world instances. Moreover, Gugat et al. (2018) proposed a decomposition-based approach that solves an MIP in the master problem and deals with the gas physics in the subproblem using a separation problem. Therein, the gas physics is described by a system of ordinary differential equations (ODEs), whereby convexity and monotony of the constraint functions is required for their method. Similarly to this approach, Schmidt et al. (2018) developed a method that replaces the necessity of convexity and monotony with Lipschitz continuity. Beyond that, the authors were able to prove that only approximate Lipschitz constants are sufficient for a finite termination and deduced additional conditions under which infeasibility can be detected. Ríos-Mercado and Borraz-Sánchez (2015) as well as Hante et al. (2017) present general information on modeling and solution methods for gas transport.

The global optimization of transient mixed-integer gas transport has been tackled with related methods by Mahlke et al. (2010) and Domschke et al. (2011). Therein, the authors focus on minimizing the fuel gas consumption of the compressors while gas injection and discharge are given a priori and fixed at each entry and exit. In contrast, we maximize in the transient case the amount of gas that can be fed into the network, where some entries and exits have a variable gas injection and discharge. Recently, Gugat et al. (2017) presented an instantaneous control approach for solving transient gas transport problems, where an MIP needs to be solved for each time step. Global optimality, however, is only guaranteed for each time step and not for the entire time horizon. Moreover, Hahn et al. (2017) proposed several heuristics for a transient optimal control model for gas transport networks.

Finally, to complete the overview, we give a brief review of the extensive literature on several general approaches to optimization problems that combine PDEs and discrete decisions. This is typically referred to as optimal control. In Gerdts (2006) and H. Lee et al. (1999), variable time transformation methods that result in a continuous formulation are proposed. However, these are specifically tailored to ODEs. Other

approaches switch at discrete time points between different systems of ODEs. Recently, a generalization to PDEs was analyzed in Rüffler and Hante (2016), whereby only local optimality can be guaranteed. In addition, methods have been developed that use complementarity-based reformulations, see for example the works by Baumrucker and L. Biegler (2009), Hante and Schmidt (2017), and Schmidt (2013), while they depend on special nonlinear solvers or supplementary relaxation techniques. Recently, Buchheim et al. (2015) presented a global solution method for certain semi-linear elliptic mixed-integer PDE problems using outer approximation. A first-discretize-then-optimize approach has been used in the following articles. Sager et al. (2009) developed the so-called convexification method, in which discrete decisions are used to derive a convex relaxation of mixed-integer optimal control problems. This method is limited to ODEs only and is extended to PDEs by Hante and Sager (2013). Moreover, Jung et al. (2015) and Sager et al. (2011) applied this convexification method to handle discrete decisions over time and propose an efficient way to compute feasible solutions. Bock et al. (2018) studied cases in which discrete decisions depend on the state variables and studied a reformulation as well as solution method for such problems. Another direction that has been intensively investigated, especially in the chemical engineering community, is mixed-integer dynamic optimization. The dynamic system is typically described by a series of differential-algebraic equations (DAEs). Allgor and Barton (1999) propose a general decomposition-based approach for the problems that arise in this regard. For further surveys in this field of research, we refer to the references contained therein.

4.2. Mathematical Model of Gas Transport

In this section, we present the basic equations that describe the transport of gas in a network consisting of pipes and active elements, i.e., elements such as valves and compressors that can be controlled in various ways. First, we describe the network and its elements. We subsequently introduce the partial differential and algebraic equations that model the conservation of mass and momentum across the entire network.

We model the gas network using a directed finite graph $G = (V, A)$ with node set V and arc set A. Every arc $a \in A$ corresponds to a specific element of the network, i.e., a pipe or an active element. Correspondingly, we split $A = A_{\mathrm{pi}} \cup A_{\mathrm{ae}}$ into subsets of pipes and active elements. We consider the set of active elements in this and the succeeding section from a general perspective and do not state it more precisely. For a detailed description of the various active elements, see Section 4.4. The nodes $v \in V$ model the endpoints of specific elements and correspond to junctions of several elements or to the entries and exits of the network, where gas can be injected or discharged.

For the subsequent modeling, we need the notation for the set of ingoing and outgoing arcs

$$\delta^{\text{in}}(v) \coloneqq \{a = (v_1, v_2) \in A : v_2 = v\},$$
$$\delta^{\text{out}}(v) \coloneqq \{a = (v_1, v_2) \in A : v_1 = v\}$$

of a node $v \in V$. Herewith, we denote by $A(v) \coloneqq \delta^{\text{in}}(v) \cup \delta^{\text{out}}(v)$ the set of arcs that are incident on v.

We now split the set of nodes $V = V_0 \cup V_\partial$ into a set of interior nodes $V_0 \coloneqq \{v \in V : |A(v)| > 1\}$ and a set of boundary nodes $V_\partial = \{v \in V : |A(v)| = 1\}$. Moreover, we partition $V_\partial = V_q \cup V_p$ into boundary nodes where either the mass flow q or the pressure p, both specified more precisely in the next subsection, is prescribed.

4.2.1. Gas Flow in Pipes. Gas is transported through the network via pipes. The gas transport in a one-dimensional pipe $a \in A_{\text{pi}}$ itself is generally described by the Euler equations, which are given as a system of nonlinear hyperbolic PDEs. They are formed by the continuity equation, the balance of moments, and the energy equation; see for example the works by Feistauer (1994) and Lurie (2008):

$$\partial_t \rho + \partial_x(\rho v) = 0, \tag{25a}$$
$$\partial_t(\rho v) + \partial_x(p + \rho v^2) = -\frac{\lambda}{2D}\rho v\,|v| - g\rho h', \tag{25b}$$
$$\partial_t\left(\rho\left(\tfrac{1}{2}v^2 + e\right)\right) + \partial_x\left(\rho v\left(\tfrac{1}{2}v^2 + e\right) + pv\right) = -\frac{k_w}{D}\left(T - T_w\right). \tag{25c}$$

The three equations (25a)–(25c) describe the conservation of mass, momentum, and energy, respectively. Here, $\rho = \rho(x,t) \in \mathbb{R}_{>0}$, $v = v(x,t) \in \mathbb{R}$, $p = p(x,t) \in \mathbb{R}_{>0}$, and $T = T(x,t) \in \mathbb{R}$ are the unknown density, velocity in the direction of the pipe, pressure, and temperature, respectively. The constants λ, g, and k_w are the friction coefficient, the gravitational constant, and the heat coefficient. Furthermore, we denote the slope of the pipe by $h' = h'(x) \in [-1, 1]$, the diameter by D, and the temperature at the pipe wall surface by $T_w = T_w(x) \in \mathbb{R}$. The internal energy is given by the variable $e = c_v T + gh$ with the specific heat constant c_v and height h of the pipe. System (25) consists of three equations with four unknowns. In order to complete the system, an additional fourth equation is needed. To this end, we use the equation of state for real gases

$$p = R_{\text{s}}\rho T z(p, T), \tag{26}$$

where $z(p,T)$ is the compressibility factor and R_s the specific gas constant. As for the spatial coordinate x and the time coordinate t, we have $x \in [0, l]$ with the pipe length l and $t \in [0, \mathcal{T}]$ with an end of the time horizon at $\mathcal{T}$.

In our case, we describe the friction factor λ of a pipe by the formula of Nikuradse

$$\lambda = \left(2\log_{10}\left(\frac{D}{k}\right) + 1.138\right)^{-2},$$

where k is the roughness of the pipe. This is a reasonable approximation for turbulent flows and very large Reynolds numbers, which we assume in the following; see Fügenschuh et al. (2015) for more details.

In this thesis, we consider gas optimization problems including discrete decisions that result from the switching of active elements. Additionally, we put particular emphasis on global optimal solutions for these problems. In order to get computationally tractable problems, we have to simplify system (25) as a trade-off, which is common practice.

First, we assume that the temperature T is constant, which allows us to neglect the energy equation (25c). The resulting system is often referred to as the isothermal Euler equations. We further assume that the compressibility factor z is constant as well. As a consequence, the speed of sound c is also constant and can be computed by

$$c^2 = \sqrt{\frac{p}{\rho}} = R_s T z.$$

This transforms the equation of state (26) to

$$p = c^2 \rho. \tag{27}$$

Using this simplified equation of state, we can reformulate the second term on the left-hand side of the balance of moments (25b) as

$$p + \rho v^2 = p\left(1 + \frac{v^2}{c^2}\right).$$

In practice, the velocity of gas is usually around $10\,\mathrm{m\,s^{-1}}$, which is significantly lower than the speed of sound in the gas at about $340\,\mathrm{m\,s^{-1}}$. We therefore fix the gas velocity to $10\,\mathrm{m\,s^{-1}}$ in order to reduce noise and vibrations in the pipe. As a result, v^2/c^2 is very small and can thus be neglected in our model.

We furthermore consider $\partial_t(\rho v)$ to be very small as well. This assumption leads to a model comparable to the friction-dominated models discussed by J. Brouwer et al. (2011), which are widely used, e.g., in the engineering literature; see again the work by J. Brouwer et al. (2011) and the references therein.

For cylindric pipes and one-dimensional flow in the direction of the pipe, which we consider in the following, the mass flow q is related to ρ and v by

$$q = A\rho v,$$

where $A = D^2\pi/4$ is the cross-sectional area of the pipe. In this way, we obtain a

simplified formulation of the continuity equation (25a) and balance of moments (25b):

$$A\partial_t \rho + \partial_x q = 0, \tag{28}$$

$$\partial_x p = -\frac{\lambda}{2D}\frac{|q|q}{A^2\rho} - g\rho h'. \tag{29}$$

In summary, the system (27)–(29) of nonlinear parabolic PDEs is a friction-dominated approximation of the system consisting of the isothermal Euler equations and the equation of state (26); see Domschke et al. (2017) for additional details. In the following, we refer to this friction-dominated and transient model as (FDT) for short.

In this thesis, we also consider stationary gas optimization problems. In this case, the gas is in a steady state and we can omit all time derivatives. Applying this to the (FDT) model now results in the following system:

$$\partial_x q = 0, \tag{30}$$

$$\partial_x p = -\frac{\lambda}{2D}\frac{|q|q}{A^2\rho} - g\rho h'. \tag{31}$$

We refer to this friction-dominated and stationary model as (FDS) for short. The first equation simply states that the mass flow q along the pipe is constant. The second equation can be solved analytically for p using the simplified equation of state (27); see Fügenschuh et al. (2015) for more details. In the case of horizontal pipes, for which $h' = 0$, this corresponds to the well-known Weymouth equation

$$p_{\text{out}}^2 - p_{\text{in}}^2 = -\frac{l\lambda c^2}{DA^2}\,|q|\,q, \tag{32}$$

where $p_{\text{in}} = p(0)$ and $p_{\text{out}} = p(l)$; see Weymouth (1912). An overview of the models derived in this subsection with all simplifications is shown in Figure 4.2.

In the following, all introduced physical entities associated with a pipe $a = (v_1, v_2) \in A$ are labeled by a corresponding index a. Moreover, we identify a pipe a with its spatial coordinate interval $[0, l_a]$, whereby the node v_1 corresponds to the spatial coordinate 0 and the node v_2 corresponds to the coordinate l_a. This allows us to define differentiation for functions defined on a. For the remainder of Section 4.2, we consider the transient pipe model (FDT).

4.2.2. Active Elements. In this section, we discuss active elements in more general terms and refer to Section 4.4 for a more specific treatment. An active Element $a = (v_1, v_2) \in A_{\text{ae}}$ can be switched on or off. It can be interpreted as a pipe of length $l_a = 0$, in which the pressures $p_a(v_1)$, $p_a(v_2)$ and mass flows $q_a(v_1)$, $q_a(v_2)$ at the pipe ends v_1 and v_2 are related in a certain algebraic manner. First, switched-off elements block gas from passing. For switched-on elements, however, flow through the

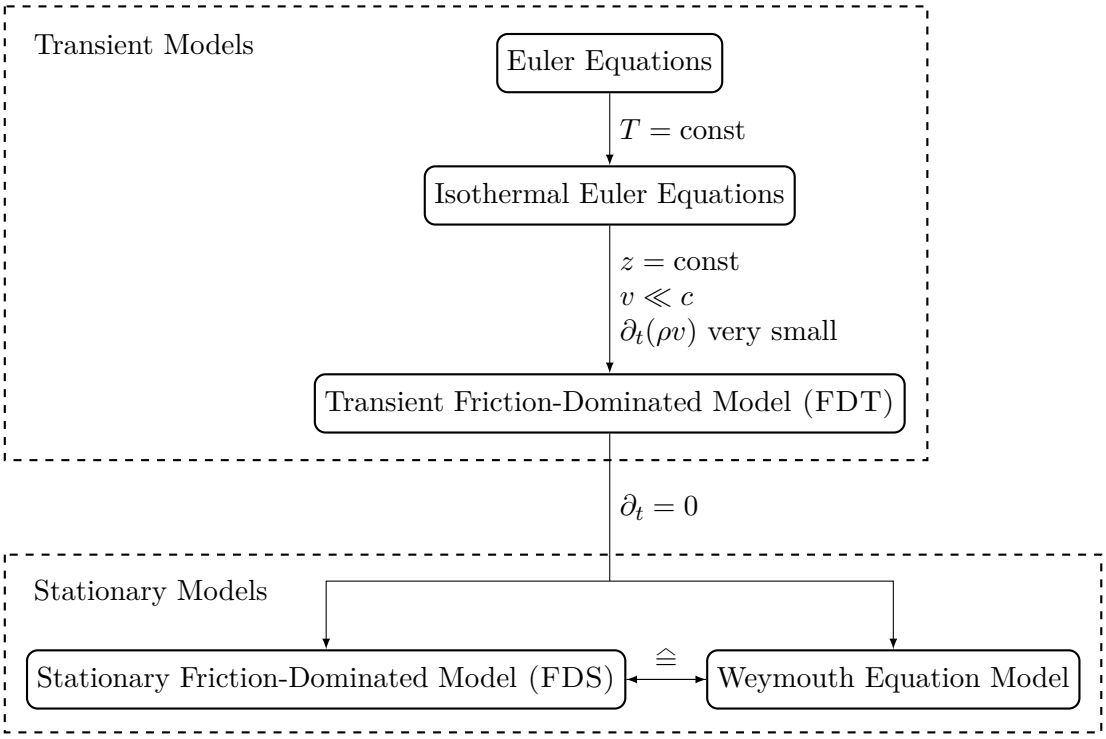


FIGURE 4.2. Overview of the stationary and transient models that are used to represent gas flow in a pipe in this thesis.

element is permitted and we require that

$$p_a(v_2) = p_a(v_1) + \triangle\hat{p}_a,$$

where the pressure difference $\triangle\hat{p}_a$ can be negative, zero, or positive and has to be prescribed. From now on, the hat symbol indicates that the corresponding function is considered as input data for the corresponding gas transport model.

In order to model the flow through an active element $a \in A_{\text{ae}}$, we now replace the equations (28), (29) and (30), (31), respectively, by

$$q_a(v_1) - q_a(v_2) = 0, \tag{33}$$

$$\hat{s}_a(p_a(v_2) - p_a(v_1)) + (1 - \hat{s}_a)q_a(v_1) = \hat{s}_a\triangle\hat{p}_a, \tag{34}$$

where the switching variable $\hat{s}_a \in \{0, 1\}$ and $\triangle\hat{p}_a$ have to be prescribed.

We point out that in this and the following section, we will discuss the gas flow models with regard to numerical simulation. In Section 4.4, we specifically address the optimization part of our first-discretize-then-optimize approach and consider $\hat{s}_a$ and $\triangle\hat{p}_a$ as free variables that are to be optimized.

4.2.3. Coupling Conditions. In order to comply with the basic principles of mass and momentum conservation also at an interior node $v \in V_0$ of the gas network, we require that

$$\sum_{a \in \delta^{\text{out}}(v)} q_a(v) - \sum_{a \in \delta^{\text{in}}(v)} q_a(v) = \hat{q}_v, \tag{35}$$

$$p_a(v) = p_v \qquad \text{for all } a \in A(v). \tag{36}$$

Here, $q_a(v)$ again indicates the mass flow in the element a at the spatial coordinate that is associated with the node v. The variable p_v represents the value of the pressure at the node v, which is determined automatically when solving the system. We point out that for interior nodes in fact $\hat{q}_v = 0$ and that we use $\hat{q}_v$ in (35) instead of 0 because of a uniform notation.

At the boundary nodes $v \in V_\partial = V_p \cup V_q$, we prescribe either the mass flow or the pressure. We state this by

$$\sum_{a \in \delta^{\text{out}}(v)} q_a(v) - \sum_{a \in \delta^{\text{in}}(v)} q_a(v) = \hat{q}_v \qquad \text{for all } v \in V_q, \tag{37}$$

$$p_a(v) = \hat{p}_v \qquad \text{for all } v \in V_p,\ a \in A(v). \tag{38}$$

Now, $\hat{q}_v$ is the prescribed mass flow entering or leaving the system at the node v and $\hat{p}_v$ is the prescribed pressure at the node v. Note that by definition of the boundary nodes, the two sums in (37) only lead to exactly one summand, i.e., there is also only one element $a \in A(v)$ for each $v \in V_p$ in (38). As before, the hat symbol denotes prescribed data for the gas flow model.

4.2.4. Initial Data. In order to describe the evolution of the gas network completely, we have to additionally prescribe the initial density distribution on all elements $a \in A$ by

$$\rho_a(x, 0) = \hat{\rho}_{a,0}(x). \tag{39}$$

The model (30)–(39) fully describes the stationary gas flow in a network. On the other hand, the model (27)–(29), (33)–(39) is our complete mathematical model for the transient gas flow in the network and forms the starting point for the subsequent discretization.

4.3. Discretization of the PDEs

In this section, we derive a discretization method for the transient gas flow model established in the previous section. We begin with a change of variables and a subsequent reformulation of the system (27)–(29), (33)–(39) that describes the gas

flow in a network consisting of pipes and active elements. This results in a system that is better suited for a systematic discretization.

Thereafter, we discuss the discretization in space by a finite volume approach. With regard to time discretization, we use an implicit Euler method and a time-expanded graph for modeling purposes. This idea originates from Martin Skutella, Marc E. Pfetsch, and Martin Groß within subproject A07 of the previously mentioned SFB Transregio 154 project.

We focus in this section on the simulation of the gas flow in the network, i.e., we deal with the discretization part of our first-discretize-then-optimize approach. For the optimization part, we refer to Section 4.4.

4.3.1. Change of Variables. For the succeeding discretization, we reformulate the model (27)–(29), (33)–(39) using the squared pressure $\pi := p^2$. The equations (27)–(29), which model the gas transport for a single pipe $a \in A_{\mathrm{pi}}$ can be reformulated equivalently as follows. First, the equation of state (27) reads as

$$\pi_a = c^4 \rho_a^2. \tag{40}$$

For ease of notation, we again state the continuity equation

$$A_a \partial_t \rho_a + \partial_x q_a = 0. \tag{41}$$

Using (27) and multiplying by p_a, the balance of moments (29) corresponds to

$$p_a \partial_x p_a = -\frac{\lambda_a c^2}{2 D_a A_a^2} |q_a| q_a - \frac{g h_a'}{c^2} p_a^2 \iff \left(1 + \frac{g h_a'}{c^2}\right) \partial_x \pi_a = -\frac{\lambda_a c^2}{D_a A_a^2} |q_a| q_a. \tag{42}$$

Note that $p_a \partial_x p_a = \frac{1}{2} \partial_x \pi_a$ is used in (42), which follows by the product rule. Hence, we can omit the factor $\frac{1}{2}$ on both sides of the equation.

Since we only have to consider the cases $\hat{s}_a = 0$ and $\hat{s}_a = 1$, the algebraic equations (33) and (34), which model the gas transport through active elements $a \in A_{\mathrm{ae}}$, can be rewritten accordingly as

$$q_a(v_1) - q_a(v_2) = 0, \tag{43}$$

$$\hat{s}_a(\pi_a(v_2) - \pi_a(v_1)) + (1 - \hat{s}_a) q_a(v_1) = \hat{s}_a \triangle \hat{\pi}_a, \tag{44}$$

where $\triangle \hat{\pi}_a$ is directly related to $\triangle \hat{p}_a$ via (34).

In a similar manner, we rewrite the coupling condition (36) at the interior vertices $v \in V_0$, while the coupling condition (35) remains unchanged:

$$\sum_{a \in \delta^{\mathrm{out}}(v)} q_a(v) - \sum_{a \in \delta^{\mathrm{in}}(v)} q_a(v) = \hat{q}_v, \tag{45}$$

$$\pi_a(v) = \pi_v \qquad \text{for all } a \in A(v). \tag{46}$$

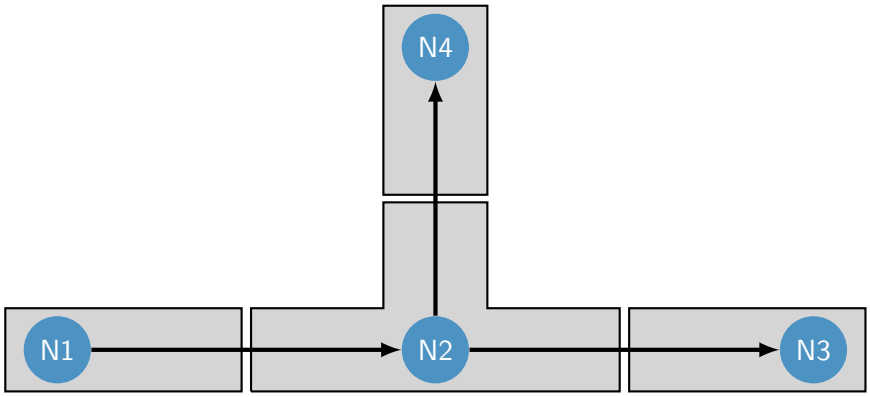


FIGURE 4.3. A gas network described by a graph $G = (V, A)$ with node set $V = \{\mathsf{N1}, \mathsf{N2}, \mathsf{N3}, \mathsf{N4}\}$ and arc set $A = \{(\mathsf{N1}, \mathsf{N2}), (\mathsf{N2}, \mathsf{N3}), (\mathsf{N2}, \mathsf{N4})\}$. All arcs of the network correspond to pipes. The control volumes vol_v for all nodes $v \in V$ are shown in gray.

Again, the variable π_v represents the value of the squared pressure at the node v, which is determined automatically during the solution process.

Using (27), we reformulate the boundary conditions (37) and (38) as

$$\sum_{a \in \delta^{\mathrm{out}}(v)} q_a(v) - \sum_{a \in \delta^{\mathrm{in}}(v)} q_a(v) = \hat{q}_v \qquad \text{for all } v \in V_q, \tag{47}$$

$$\rho_a(v) = \hat{\rho}_v \qquad \text{for all } v \in V_p,\ a \in A(v), \tag{48}$$

where $\hat{\rho}_v$ can again be obtained from the prescribed pressure $\hat{p}_v$ via (27).

For ease of notation, we also recall the initial condition

$$\rho_a(x, 0) = \hat{\rho}_{a,0}(x). \tag{49}$$

The system (40)–(49) in the three variables ρ, q, and π forms the starting point for the discretization approach outlined in the following.

4.3.2. Space Discretization. As already mentioned, we identify a pipe $a \in A_{\mathrm{pi}}$ with its spatial coordinate interval $[0, l_a]$. In addition, we now associate to each node $v \in V$ a control volume vol_v made up of the pipes $a \in A(v)$ cut to half lengths; see Figure 4.3 for an illustration. We refer to the physical volume of this control volume as

$$|\mathrm{vol}_v| := \sum_{a \in A(v)} \frac{1}{2} A_a l_a,$$

which is the sum of half the volumes of the incident pipes. Moreover, we require that $\mathrm{vol}_v > 0$ for all $v \in V$, i.e., every node is endpoint of at least one pipe $a \in A_{\mathrm{pi}}$, which is trivially fulfilled.

Let $\tilde{v}_a$ be the midpoint of the pipe $a \in A_{\mathrm{pi}}$. By integration of the continuity

equation (41) over the control volume vol_v and use of the coupling condition (45) at the vertex v, we obtain

$$\int_{\mathrm{vol}_v} \partial_t \rho_a \,\mathrm{d}x + \sum_{a \in \delta^{\mathrm{out}}(v)} q_a(\tilde{v}_a) - \sum_{a \in \delta^{\mathrm{in}}(v)} q_a(\tilde{v}_a) = \hat{q}_v. \tag{50}$$

Note that the cross-sectional area A_a from (41) appears implicitly in the definition of the control volume vol_v. The equation (50) expresses the conservation of mass on the control volume vol_v.

Integration of the momentum equation (42) over a pipe $a = (v_1, v_2)$ leads to

$$\int_a \partial_x \pi_a \,\mathrm{d}x + \frac{g}{c^2} \int_a h_a' \partial_x \pi_a \,\mathrm{d}x = -\frac{\lambda_a c^2}{D_a A_a^2} \int_a |q_a| q_a \,\mathrm{d}x, \tag{51}$$

which expresses the integral balance of pressure forces and friction over a pipe.

We now replace ρ_a and q_a by their respective averages over the control volumes vol_v for all $v \in V$ and over all arcs $a \in A$. To this end, we use piecewise constant functions and denote the averages by $\tilde{\rho}_v$ and $\tilde{q}_a$, respectively. As approximation for (50), we then obtain

$$|\mathrm{vol}_v| \, \partial_t \tilde{\rho}_v + \sum_{a \in \delta^{\mathrm{out}}(v)} \tilde{q}_a - \sum_{a \in \delta^{\mathrm{in}}(v)} \tilde{q}_a = \hat{q}_v \qquad \text{for all } v \in V_0 \cup V_q. \tag{52}$$

For the boundary nodes $v \in V_p$, where the pressure is prescribed, we instead require that

$$\tilde{\rho}_v = \hat{\rho}_v. \tag{53}$$

With the slope at the midpoint of the pipe $\tilde{h}_a'$ and the approximations

$$h_a' \approx l_a \tilde{h}_a',$$

$$\int_a |q_a| q_a \,\mathrm{d}x \approx l_a |\tilde{q}_a| \tilde{q}_a,$$

the balance of moments is approximately described by

$$-\left(\frac{D_a A_a^2}{\lambda_a c^2 l_a} + \frac{g \tilde{h}_a' D_a A_a^2}{\lambda_a c^4} \right) (\pi_{v_2} - \pi_{v_1}) = |\tilde{q}_a| \tilde{q}_a \tag{54}$$

for all pipes $a \in A_{\mathrm{pi}}$.

For an active element $a = (v_1, v_2) \in A_{\mathrm{ae}}$, we can directly use equation (44) and replace (54) by

$$\hat{s}_a (\pi_{v_2} - \pi_{v_1}) + (1 - \hat{s}_a) \tilde{q}_a = \hat{s}_a \triangle \hat{\pi}_a. \tag{55}$$

This equations resembles the balance of momentum for an active element $a \in A_{\mathrm{ae}}$. Moreover, we set $l_a = 0$ in case of active elements for the calculation of the volume $|\mathrm{vol}_v|$ assigned to the vertex v.

To complete the system, we finally require that

$$\pi_v = c^4 \tilde{\rho}_v^2 \qquad \text{for all } v \in V. \tag{56}$$

Further, we prescribe the initial density distribution on every control volume vol_v by

$$\tilde{\rho}_v(0) = \hat{\rho}_{v,0}, \tag{57}$$

while $\hat{\rho}_{v,0}$ is defined as the average of the initial density distribution $\hat{\rho}_{a,0}$ in (49) over the control volume vol_v. We finally point out that the resolution of this space discretization can be controlled by artificially subdividing a pipe into several smaller pipes. This also allows a non-conforming discretization.

The system (52)–(57) is the numerical model for the gas transport in a network after semi-discretization in space. In order to obtain a computational model, we still have to discretize in time.

4.3.3. Time Discretization. For the discretization in time, we utilize the implicit Euler method. Let $\tau > 0$ be the time step and $t_n = n\tau$ with $n = 0, \ldots, N$ the discrete time points, where t_N corresponds to the end of the time horizon $\mathcal{T}$. For a function $u(t)$ of time, we denote by u_n the approximation for $u(t_n)$.

By integration of the space-discretized equation (52) in time from t_{n-1} to t_n, we obtain

$$|\mathrm{vol}_v| \left(\tilde{\rho}_{v,n} - \tilde{\rho}_{v,n-1}\right) + \int_{t_{n-1}}^{t_n} \left(\sum_{a \in \delta^{\mathrm{out}}(v)} \tilde{q}_a - \sum_{a \in \delta^{\mathrm{in}}(v)} \tilde{q}_a \right) = \tau \hat{q}_{v,n}.$$

By further approximating

$$\int_{t_{n-1}}^{t_n} \tilde{q}_a \approx \tau \tilde{q}_{a,n},$$

we then have

$$\frac{|\mathrm{vol}_v|}{\tau} \tilde{\rho}_{v,n} + \sum_{a \in \delta^{\mathrm{out}}(v)} \tilde{q}_{a,n} - \sum_{a \in \delta^{\mathrm{in}}(v)} \tilde{q}_{a,n} = \frac{|\mathrm{vol}_v|}{\tau} \tilde{\rho}_{v,n-1} + \hat{q}_{v,n} \quad \text{for all } v \in V_0 \cup V_q. \tag{58}$$

To obtain a more intuitional and graph-oriented modeling, we use a time-expanded graph that can be used equivalently to incorporate equation (58). The discretization pursued in this subsection is a time-expansion technique, in which time-dependent properties change only at discrete time points. It can therefore be modeled by a time-expanded graph. The benefit of this technique is that we can now basically use a stationary flow model for each time step. The time steps themselves are then linked using time expansion by linear constraints. This leads to an MINLP model whose nonlinear functions have the same dimension as in the stationary case, which is very favorable with regard to piecewise linear approximations of the nonlinearities. Roughly speaking, with time extension, we solve a transient problem by solving a large stationary problem.

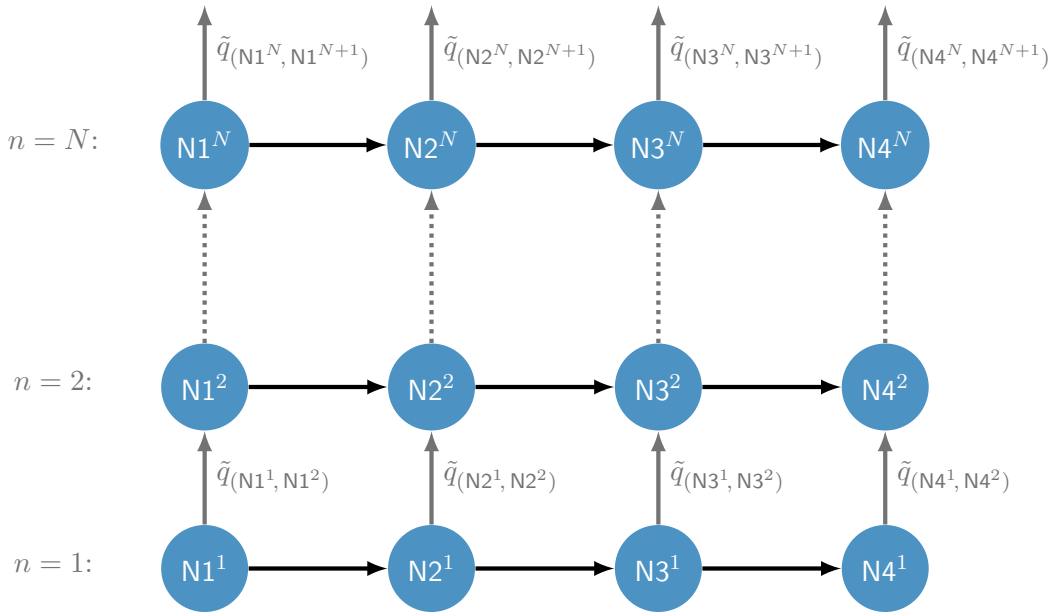


FIGURE 4.4. A gas network described by a graph $G = (V, A)$ with node set $V = \{\mathsf{N1}, \mathsf{N2}, \mathsf{N3}, \mathsf{N4}\}$ and arc set $A = \{(\mathsf{N1}, \mathsf{N2}), (\mathsf{N2}, \mathsf{N3}), (\mathsf{N3}, \mathsf{N4})\}$. All arcs of the network correspond to pipes, while each copy of a node is linked to the previous and subsequent copies by corresponding arcs (dark gray).

We copy the gas network for each time point and introduce for each node $v \in V$ a holdover arc (v^n, v^{n+1}) connecting the copy v^n of the node v in time step n with the one in time step $n+1$. Moreover, we introduce a mass flow variable

$$\tilde{q}_{(v^n, v^{n+1})} \coloneqq \frac{|\mathrm{vol}_v|}{\tau} \tilde{\rho}_{v,n},$$

which loosely speaking indicates how much gas remains in the node v from time point t_n to time point t_{n+1}. See Figure 4.4 for an illustration.

We further define the parameters

$$\alpha_v \coloneqq \frac{|\mathrm{vol}_v|}{\tau} \qquad \text{for all } v \in V,$$

$$\beta_a \coloneqq -\left(\frac{D_a A_a^2}{\lambda_a c^2 l_a} + \frac{g \tilde{h}'_a D_a A_a^2}{\lambda_a c^4}\right)^{-1} \qquad \text{for all } a \in A_{\mathrm{pi}}$$

and assume that $\hat{s}_{a,n}$, $\hat{\rho}_{v,0}$, $\hat{\rho}_{v,n}$, $\hat{q}_{v,n}$, and $\triangle\hat{\pi}_{a,n}$ are prescribed. The fully discrete approximation of the transient gas transport model (27)–(29), (33)–(39) is now given

by

$$\tilde{q}_{(v^n,v^{n+1})} + \sum_{a\in\delta^{\mathrm{out}}(v)} \tilde{q}_{a,n} - \sum_{a\in\delta^{\mathrm{in}}(v)} \tilde{q}_{a,n} = \tilde{q}_{(v^{n-1},v^n)} + \hat{q}_{v,n} \qquad \text{for all } v \in V_0 \cup V_q, \tag{59a}$$

$$\tilde{q}_{(v^n,v^{n+1})} = \alpha_a \hat{\rho}_{v,n} \qquad \text{for all } v \in V_p, \tag{59b}$$

$$\tilde{q}^2_{(v^n,v^{n+1})} = \frac{\alpha_a^2}{c^4}\pi_{v,n} \qquad \text{for all } v \in V, \tag{59c}$$

$$\pi_{v_2,n} - \pi_{v_1,n} = \beta_a |\tilde{q}_{a,n}|\tilde{q}_{a,n} \qquad \text{for all } a \in A_{\mathrm{pi}}, \tag{59d}$$

$$\hat{s}_{a,n}(\pi_{v_2,n} - \pi_{v_1,n}) + (1-\hat{s}_{a,n})\tilde{q}_{a,n} = \hat{s}_{a,n}\triangle\hat{\pi}_{a,n} \qquad \text{for all } a \in A_{\mathrm{ae}}, \tag{59e}$$

for all $n \in \{1,\dots,N\}$,. With $\tilde{q}_{(v^0,v^1)} = \alpha_a \hat{\rho}_{v,0}$ for all $v \in V$ and the variable vectors $\left(\tilde{q}_{(v^n,v^{n+1})}\right)_{v\in V}$, $(\tilde{q}_{a,n})_{a\in A}$, and $(\pi_{v,n})_{v\in V}$, where $n = 1,\dots,N$, the problem (59) is a simulation problem. It does not involve integer variables, because the controls $\hat{s}_{a,n}$ and $\triangle\hat{\pi}_{a,n}$ are assumed to be prescribed here. In the further course of our first-discretize-then-optimize approach, we will deal with gas optimization problems, where $s_{a,n}$ and $\triangle\pi_{a,n}$ are included as free variables (without the hat symbol) in Section 4.4. Since the number of unknowns and equations in (59) match, we can prove that there is a unique solution once the input and control variables are set appropriately.

We also point out that the time-expanded graph can be considered as an extension of a stationary gas transport model to a transient gas transport model via the mass flow on the holdover arcs. Indeed, if we omit all mass flow variables $\tilde{q}_{(v^n,v^{n+1})}$ and accordingly the equations (59b) and (59c), we obtain for each time step n the stationary model that is based on the Weymouth equation; see model (FDS) and (32).

Finally, we remark that the well-posedness of the simulation problem (59) is proven in Burlacu et al. (2018). It is not part of this thesis, since the author of this thesis did not contribute to its proof.

4.4. MINLP Models

Endowed with the discretization from Section 4.3, we continue in this section with the optimization part of our first-discretize-then-optimize approach. To this end, we show how to model gas optimization problems that are discussed at the beginning of this chapter and incorporate the previously derived flow models as MINLPs. More precisely, we consider the stationary gas transport optimization problem of minimizing the compressor energy costs and the transient gas optimization problem of maximizing the storage capacity of gas networks. Furthermore, we give more specific descriptions for the various active elements $a \in A_{\mathrm{ae}} = A_{\mathrm{vl}} \cup A_{\mathrm{cv}} \cup A_{\mathrm{rs}} \cup A_{\mathrm{cm}}$, where A_{vl}, A_{cv}, A_{rs}, and A_{cm} correspond to the set of all valves, control valves, resistors, and compressors.

As mentioned in Subsection 4.3.3, the control variables $s_{a,n}$ and $\triangle\pi_{a,n}$ are now free and to optimize.

In addition, as pointed out in the same subsection, the stationary flow model can basically be considered as the transient model in a single time step. We therefore always only describe the transient modeling of a specific gas network element if the stationary modeling is equivalent to the transient one in a single time step. Otherwise, we explicitly state the different models. To be consistent with the implementation regarding the MINLP problems described in this section, we now replace the pressure square variable π with the square of the pressure variable p^2. For ease of notation, we also write q instead of $\tilde{q}$ for the mass flow variables.

First, for each node $v \in V$ and time step n, we assume lower and upper bounds $p^-_{v,n}$ and $p^+_{v,n}$ for the pressure variable $p_{v,n}$ and lower and upper bounds $q^-_{a,n}$ and $q^+_{a,n}$ for the mass flow variable $q_{a,n}$ for each arc $a \in A$ and time step n. Regarding the stationary flow model, we simply omit the second index n for the pressure and mass flow variables and corresponding bounds. Consequently, with the bounds of the pressure variable $p_{v,n}$, the lower and upper bounds $q^-_{(v^n,v^{n+1})}$ and $q^+_{(v^n,v^{n+1})}$ for the flow variable $q_{(v^n,v^{n+1})}$ are implicitly given via (59c) for each node $v \in V$ and time step n. Finally, we assume that $V_\partial = V_q$, i.e., the mass flow is prescribed for each boundary node $n \in V_\partial$. Also recall that $q_{(v^0,v^1)}$ are given for all $v \in V$ by $q_{(v^0,v^1)} = \alpha_a \hat{\rho}_{v,0}$, where $\hat{\rho}_{v,0}$ is the prescribed average of the initial density distribution.

4.4.1. Mass Flow Conservation. We distinguish between the transient and stationary flow model.

4.4.1.1. *Transient Flow Model.* With $[N] \coloneqq \{1, \dots, N\}$, we have the mass flow conservation constraints

$$q_{(v^n,v^{n+1})} + \sum_{a \in \delta^{\text{out}}(v)} q_{a,n} - \sum_{a \in \delta^{\text{in}}(v)} q_{a,n} = q_{(v^{n-1},v^n)} + \hat{q}_{v,n} \qquad \text{for all } v \in V,\ n \in [N]. \tag{60}$$

Additionally, we have the constraints

$$q_{(v^n,v^{n+1})} = \frac{\alpha_a}{c^2} p_{v,n} \qquad \text{for all } v \in V,\ n \in [N] \tag{61}$$

coupling the two consecutive time points n and $n+1$.

4.4.1.2. *Stationary Flow Model.* In case of the stationary flow model, we simply have

$$\sum_{a \in \delta^{\text{out}}(v)} q_a - \sum_{a \in \delta^{\text{in}}(v)} q_a = \hat{q}_v \qquad \text{for all } v \in V \tag{62}$$

for the mass flow conservation.

4.4.2. Pipes. Pipes are the most frequent element in our gas networks and are essential for the transport of the gas. They can be considered as passive elements, i.e.,

unlike active elements, there is no switching between different modes. With $\pi = p^2$, we incorporate the pressure loss equations (59d) as

$$p_{v_2,n}^2 - p_{v_1,n}^2 = \beta_a |q_{a,n}| q_{a,n} \qquad \text{for all } a = (v_1, v_2) \in A_{\text{pi}},\ n \in [N] \tag{63}$$

in case of the transient flow model. For the stationary case, we only have one time step in (63) and omit the index n.

4.4.3. Shortcuts. Another type of passive elements are shortcuts. They are solely used for modeling purposes, such as the incorporation of hybrid points like gas storage facilities, and do not exist in reality. They have no influence on the physics of the gas. We denote by A_{sc} the set of all shortcuts. In the transient case, we model shortcuts simply by adding the constraints

$$p_{v_1,n} = p_{v_2,n} \qquad \text{for all } a = (v_1, v_2) \in A_{\text{sc}},\ n \in [N]. \tag{64}$$

In the stationary case, we again consider only one time step in (64) and omit the index n.

4.4.4. (Control) Valves. The first active element that we describe in more detail is the valve. It is a controllable element in the gas network that can be switched between two modes: closed or open. The following model is largely based on the stationary model by Geißler et al. (2015a). A closed valve $a = (v_1, v_2) \in A_{\text{vl}}$, where $A_{\text{vl}} \subset A_{\text{ae}}$ is the set of all valves, impedes gas from passing, which leads to decoupled pressures at the nodes v_1 and v_2. On the contrary, for open valves, we have $p_{v_1,n} = p_{v_2,n}$ and no pressure drop. We model valves using the binary switching variables $s_{a,n} \in \{0, 1\}$, whereby $s_{a,n}$ is equal to one, if and only if the valve is open and equal to zero if it is closed. In case of the transient flow model, this is described by

$$q_{a,n} \le q_{a,n}^+ s_{a,n}, \tag{65a}$$

$$q_{a,n} \ge q_{a,n}^- s_{a,n}, \tag{65b}$$

$$(p_{v_2,n}^+ - p_{v_1,n}^-) s_{a,n} + p_{v_2,n} - p_{v_1,n} \le p_{v_2,n}^+ - p_{v_1,n}^-, \tag{65c}$$

$$(p_{v_1,n}^+ - p_{v_2,n}^-) s_{a,n} + p_{v_1,n} - p_{v_2,n} \le p_{v_1,n}^+ - p_{v_2,n}^- \tag{65d}$$

for all $a = (v_1, v_2) \in A_{\text{vl}}$ and $n \in [N]$. As before, the consideration of a single time step in (65) and the omission of the index n corresponds to the model for the stationary case.

In the same way as a valve, a control valve can either be closed or open. Again, with $A_{\text{cv}} \subset A_{\text{ae}}$ as the set of all control valves, a closed control valve $a = (v_1, v_2) \in A_{\text{cv}}$ impedes gas from passing, resulting in decoupled pressures at the nodes v_1 and v_2. An open control valve, however, can reduce pressure within a given range $[\triangle p_{a,n}^-, \triangle p_{a,n}^+]$.

To model a control valve, we use (65) from above and replace the equations (65c) and (65d) by

$$(p^+_{v_2,n} - p^-_{v_1,n} + \triangle p^-_{a,n})s_{a,n} + p_{v_2,n} - p_{v_1,n} \leq p^+_{v_2,n} - p^-_{v_1,n}, \tag{66a}$$

$$(p^+_{v_1,n} - p^-_{v_2,n} - \triangle p^+_{a,n})s_{a,n} + p_{v_1,n} - p_{v_2,n} \leq p^+_{v_1,n} - p^-_{v_2,n} \tag{66b}$$

for all $a = (v_1, v_2) \in A_{\text{cv}}$ and $n \in [N]$. We again take into account a single time step in (66) and omit the index n to obtain the stationary model for a control valve.

We point out that a control valve is a uni-directional element, i.e., we have $q^-_{a,n} \geq 0$ for the lower mass flow bound and that pressure can only be reduced in the direction of the flow. Furthermore, a control valve is always located in a control valve station, which additionally provides the possibility of a bypass, leading to identical pressures at the nodes v_1 and v_2; see again Geißler et al. (2015a).

4.4.5. Resistors. To model gas network elements that cause a pressure drop, e.g., measuring stations and filtration plants, we use (fictitious) resistors, which are contained in the set $A_{\text{rs}} \subset A_{\text{ae}}$. As before, the following model is largely based on the stationary model by Geißler et al. (2015a). Unlike control valves, resistors operate in both directions. We model this with the binary variables $s_{a,n}$, where $s_{a,n}$ equals to zero, if and only if gas flows in the direction of the arc $a \in A_{\text{rs}}$, i.e., we have $q_{a,n} \geq 0$. On the other hand, $s_{a,n}$ equals to one, if and only if gas flows against the direction of the arc a, i.e., $q_{a,n} \leq 0$. This is described by

$$q^-_{a,n} s_{a,n} \leq q_{a,n} \leq q^+_{a,n}(1 - s_{a,n}) \tag{67}$$

for the transient case and again by considering only one time step in (67) and omission of the index n for the stationary case. In the remaining part, we distinguish for additional constraints between the transient and the stationary model of the resistor.

4.4.5.1. *Transient Flow Model.* In the transient case, each resistor $a \in A_{\text{rs}}$ causes a constant pressure drop $\triangle \hat{p}_{a,n} > 0$. This is modeled by

$$p_{v_1,n} - p_{v_2,n} + 2\triangle \hat{p}_{a,n} s_{a,n} = \triangle \hat{p}_{a,n} \qquad \text{for all } a = (v_1, v_2) \in A_{\text{rs}},\ n \in [N]. \tag{68}$$

4.4.5.2. *Stationary Flow Model.* For the stationary model, we split the set of resistors $A_{\text{rs}} = A_{\text{rs-c}} \cup A_{\text{rs-v}}$ an differentiate between two types of resistors: resistors $a \in A_{\text{rs-c}}$ with constant pressure drop and resistors $a \in A_{\text{rs-v}}$ with variable pressure drop.

First, we model a resistor $a \in A_{\text{rs-c}}$ with constant pressure drop by (68) while we consider only one time step and omit the index n.

A resistor $a \in A_{\text{rs-v}}$ with variable pressure drop is determined by its drag factor ξ_a

and diameter D_a. We model it by

$$p_{v_1}^2 - p_{v_2}^2 + |\triangle p_a| \triangle p_a = \gamma_a |q_a| q_a, \tag{69a}$$

$$\triangle p_a = p_{v_1} - p_{v_2}, \tag{69b}$$

$$p_{v_1} - p_{v_2} \leq (p_{v_1}^+ - p_{v_2}^-)(1 - s_a), \tag{69c}$$

$$p_{v_2} - p_{v_1} \leq (p_{v_2}^+ - p_{v_1}^-) s_a \tag{69d}$$

for all $a = (v_1, v_2) \in A_{\text{rs-v}}$. The pressure drop is now given by the variable $\triangle p_a$ and

$$\gamma_a = \frac{16\, \xi_a c^2}{\pi^2 D_a^4} \tag{70}$$

is a resistor-specific constant.

4.4.6. Compressors. In order to compensate for pressure drop on large gas networks, compressors that increase the inlet pressure to a higher outlet pressure, are installed. We consider different compressor models for the transient and stationary case. The following models are again largely based on the stationary model by Geißler et al. (2015a).

4.4.6.1. *Transient Flow Model.* A compressor $a = (v_1, v_2) \in A_{\text{cm}}$ is modeled using the binary variables $s_{a,n}$, whereby $s_{a,n}$ is equal to one, if and only if the compressor is operating, i.e., increasing the pressure $p_{v_1,n}$. In the following, we restrict ourselves to a simplified compressor model for the transient case, where an operating compressor increases the pressure $p_{v_1,n}$ such that

$$1 < r_a^- \leq \frac{p_{v_2,n}}{p_{v_1,n}} \leq r_a^+ \tag{71}$$

holds for given lower and upper bounds r_a^- and r_a^+ of the compression ratio. Note that the flow is only allowed in the direction of the arc (from v_1 to v_2) for an operating compressor with $0 \leq (q_{a,n}^{\text{op}})^- \leq (q_{a,n}^{\text{op}})^+$ as the corresponding bounds. Otherwise, if $s_{a,n}$ is equal to zero, the compressor is in bypass mode, i.e., we have $p_{v_1,n} = p_{v_2,n}$ while flow in both directions is allowed with the bounds $(q_{a,n}^{\text{by}})^- \leq (q_{a,n}^{\text{by}})^+$. We model this by

$$(q_{a,n}^{\text{by}})^- (1 - s_{a,n}) + (q_{a,n}^{\text{op}})^- s_{a,n} \leq q_{a,n}, \tag{72a}$$

$$(q_{a,n}^{\text{by}})^+ (1 - s_{a,n}) + (q_{a,n}^{\text{op}})^+ s_{a,n} \geq q_{a,n}, \tag{72b}$$

$$\triangle p_{a,n}^- \, s_{a,n} \leq p_{v_2,n} - p_{v_1,n}, \tag{72c}$$

$$\triangle p_{a,n}^+ \, s_{a,n} \geq p_{v_2,n} - p_{v_1,n}, \tag{72d}$$

$$r_a^- \, p_{v_1,n} - (r_a^- \, p_{v_1,n}^+ - p_{v_2,n}^-)(1 - s_{a,n}) \leq p_{v_2,n}, \tag{72e}$$

$$r_a^+ \, p_{v_1,n} - (r_a^+ \, p_{v_1,n}^- - p_{v_2,n}^+)(1 - s_{a,n}) \geq p_{v_2,n} \tag{72f}$$

for all $a = (v_1, v_2) \in A_{\text{cm}}$ and $n \in [N]$, where $\triangle p^-_{a,n}$ and $\triangle p^+_{a,n}$ are the bounds for the pressure increase. Finally, we point out that only compressors that are in either operating or bypass mode are considered in the transient case. The possibility to close a compressor can be modeled with additional inlet and outlet valves. We use the above modeling since closing a compressor is not useful for the instances of the transient optimization problem that we consider in Chapter 6.

4.4.6.2. *Stationary Flow Model.* In the stationary case, we use a more realistic compressor model. Each compressor $a \in A_{\text{cm}}$ is specified by its adiabatic efficiency $\eta_{\text{ad},a}$ and its energy cost coefficient c_a. Again, we use the binary variable s_a, whereby s_a is equal to one, if and only if the compressor is operating. For an operating compressor, only flow in the direction of the arc a is allowed. Now, however, we incorporate the possibility to close a compressor, i.e., to decouple the pressures at the inlet and outlet nodes and to block gas from passing through the compressor. This is done by switching s_a to zero. Similar to control valves, (usually several) compressors are located in a compressor station that allows for an additional bypass mode and corresponding flow in both directions; see again Geißler et al. (2015a) for details.

Furthermore, the operating range of a compressor is described by its non-convex characteristic diagram. We subdivide the set $A_{\text{cm}} = A_{\text{cm-t}} \cup A_{\text{cm-p}}$ distinguishing between turbo compressors $a \in A_{\text{cm-t}}$ that are usually driven by gas turbines and piston compressors $a \in A_{\text{cm-p}}$ that are typically shipped with electric or gas driven motors. We illustrate their corresponding characteristic diagrams in Figure 4.5. The gray area describes the feasible operating range of the compressor. We refer to Fügenschuh et al. (2015) for more details on how to obtain these characteristic diagrams. Since the description of the operating range requires a volumetric flow Q_a and we restrict ourselves to a mass flow q_a, we first add

$$Q_a = \frac{q_a}{\rho_0} \qquad \text{for all } a \in A_{\text{cm}}, \tag{73}$$

where ρ_0 is the gas density under normal conditions.

Moreover, the adiabatic process of compression leads to a change in the specific enthalpy $H_{\text{ad},a}$. We model this by

$$H_{\text{ad},a} = \frac{\kappa}{\kappa - 1} c^2 \left(\left(\frac{p_{v_2}}{p_{v_1}} \right)^{\frac{\kappa-1}{\kappa}} - 1 \right) \qquad \text{for all } a = (v_1, v_2) \in A_{\text{cm}}, \tag{74}$$

where κ is the adiabatic exponent which we assume to be constant. Here, considering a mass flow of q_a, we describe the power P_a the compressor consumes to increase the inlet pressure p_{v_1} to a higher outlet pressure p_{v_2} by

$$P_a = \frac{H_{\text{ad},a}}{\eta_{\text{ad},a}} q_a. \tag{75}$$

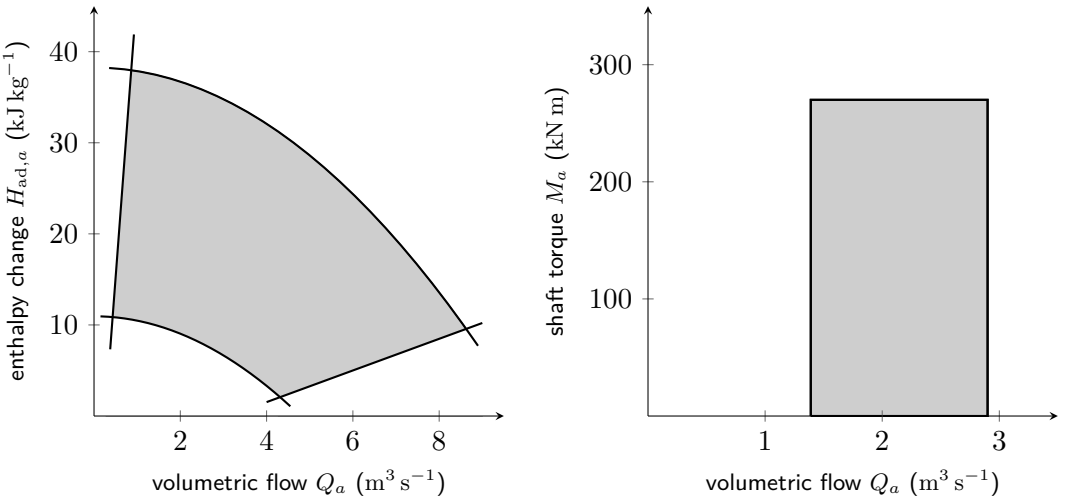


FIGURE 4.5. Characteristic diagrams for turbo compressors (left) and piston compressors (right). The feasible operating ranges are marked gray.

To complete the modeling of the different modes of a compressor, we add for all $a = (v_1, v_2) \in A_{\mathrm{cm}}$ the constraints

$$q_a \leq q_a^+ \, s_a, \tag{76a}$$

$$q_a \geq q_a^- \, s_a, \tag{76b}$$

$$P_a \leq P_a^+ \, s_a, \tag{76c}$$

$$P_a \geq P_a^- \, s_a, \tag{76d}$$

$$p_{v_2} - p_{v_1} \leq \triangle p_a^+ s_a + (p_{v_2}^+ - p_{v_1}^-)(1 - s_a), \tag{76e}$$

$$p_{v_2} - p_{v_1} \geq \triangle p_a^- s_a + (p_{v_2}^- - p_{v_1}^+)(1 - s_a), \tag{76f}$$

$$p_{v_2} \leq r_a^+ p_{v_1} - (r_a^+ p_{v_1}^- - p_{v_2}^+)(1 - s_a), \tag{76g}$$

$$p_{v_2} \geq r_a^- p_{v_1} - (r_a^- p_{v_1}^+ - p_{v_2}^-)(1 - s_a), \tag{76h}$$

where P_a^- and P_a^+ are the bounds for the power consumption and $\triangle p_a^-$ and $\triangle p_a^+$ are the bounds for the pressure increase. As in the transient case, the compression ratio is bounded by $1 < r_a^- \leq p_{v_2}/p_{v_1} \leq r_a^+$. We remark, that whenever a compressor is operating, power consumption occurs while increasing the pressure of the gas that flows in the direction of the arc, i.e., $p_{v_2} \geq p_{v_1}$ and $q_a \geq 0$. Therefore, flow against the arc direction is impossible. As pointed out before, a closed compressor results in decoupled pressures at the nodes v_1 and v_2 and there is no mass flow through the compressor.

For the characteristic diagrams, we use the convex outer-approximation approach

described by Geißler et al. (2015a) in case of a turbo compressor. For piston compressors $a = (v_1, v_2) \in A_{\text{cm-p}}$, we obtain a relaxation of the operating range by the constraints (76e)–(76h), together with

$$\frac{Q_a^-}{c^2} p_{v_1} \leq q_a \leq \frac{Q_a^+}{c^2} p_{v_1}, \tag{77}$$

where Q_a^- and Q_a^+ are given lower and upper bounds of the volumetric flow rate Q_a. Furthermore, the shaft torque M_a and $H_{\text{ad},a}$ are connected via

$$M_a = \frac{V_o \rho_0}{2\pi \eta_{\text{ad},a}} H_{\text{ad},a}, \tag{78}$$

where V_o is the operating volume of the piston compressor.

4.4.7. Switching Restrictions. Finally, in case of the transient flow model, we add constraints to limit switching operations of active elements to predetermined time intervals. These constraints imply that if the mode of an active element is changed, then it must stay in this mode for a specified time. This is motivated by the practice where the time between changing the settings of the active elements is usually long.

We thus require that an active element stays closed for S seconds if the mode is changed from open to closed (or bypass in case of a compressor), and vice versa. We model this for all $a \in A_{\text{vl}} \cup A_{\text{cv}} \cup A_{\text{cm}}$ by

$$\sum_{i=n}^{n+W_n-1} s_{a,i} \leq W_n + W_n(s_{a,n} - s_{a,n-1}) \qquad \text{for all } n \in [N], \tag{79a}$$

$$\sum_{i=n}^{n+W_n-1} s_{a,i} \geq W_n(s_{a,n} - s_{a,n-1}) \qquad \text{for all } n \in [N], \tag{79b}$$

where

$$W_n \coloneqq \min\left\{ \left\lceil \frac{S_a}{\tau} \right\rceil,\ N - n + 1 \right\} \tag{80}$$

considering that the size of a time step τ is given in seconds.

These min-up/min-down constraints are also used in other optimization problems, e.g., unit commitment problems, where power generating machines have specific min-up and min-down times; see the works by Frangioni et al. (2008) and Muckstadt and Koenig (1977) for example. Moreover, Saravanan et al. (2013) provide a review on solution methods for unit commitment problems that were developed in the last decades. Investigations on the polyhedral structure of constraints of type (79) can be found in J. Lee et al. (2004).

4.4.8. Stationary Compressor Energy Cost Minimization. As mentioned before, in the stationary case, we cover an essential problem in gas transport optimization: the minimization of the compressor energy costs while validating a nomination.

We therefore consider a gas transport network and a nomination, i.e., a prescribed amount of gas $\hat{q}_v$ for each entry and each exit $v \in V_\partial = V_s \cup V_t$, where V_s and V_t are the sets of all entries and exits. We assume that this nomination is balanced, i.e., we have

$$\sum_{v \in V_s} \hat{q}_v = \sum_{v \in V_t} \hat{q}_v.$$

First, we aim to find an admissible configuration of the active elements that satisfies all physical and technical constraints. Due to friction-induced pressure drop on long pipes, however, compressors that increase certain gas pressures have to be taken into account. This leads to the objective of minimizing the compressor energy costs.

Considering the various elements of a gas network that are described in the previous subsections, we now formally obtain the following MINLP problem:

$$\begin{aligned}
\min \quad & \sum_{a \in A_{\mathrm{cm}}} c_a P_a && \text{(81a)}\\
\text{s.t.} \quad & \text{mass flow conservation (62)}, && \text{(81b)}\\
& \text{pipes constraints (63)}, && \text{(81c)}\\
& \text{shortcuts constraints (64)}, && \text{(81d)}\\
& \text{(control) valves constraints (65) and (66)}, && \text{(81e)}\\
& \text{resistors constraints (67)–(69)}, && \text{(81f)}\\
& \text{compressors constraints (73)–(78)}, && \text{(81g)}\\
& p_v^- \le p_v \le p_v^+ \qquad \text{for all } v \in V, && \text{(81h)}\\
& q_a^- \le q_a \le q_a^+ \qquad \text{for all } a \in A, && \text{(81i)}\\
& s_a \in \{0,1\} \qquad \text{for all } a \in A_{\mathrm{ae}}. && \text{(81j)}
\end{aligned}$$

For all equations that are referenced in (81) and also apply to the transient case, we assume that only one time step is considered and that the index n is omitted. Moreover, as pointed out before, the variables s_a and $\triangle p_a$ are now free and to optimize. Although the variable $\triangle p_a$ for the pressure difference is not explicitly contained in the MINLP model (81) for each $a = (v_1, v_2) \in A_{\mathrm{cv}} \cup A_{\mathrm{rs}} \cup A_{\mathrm{cm}}$, it is implicitly specified by

$$\triangle p_a = p_{v_2} - p_{v_1}.$$

The nonlinear parts of the MINLP model (81) are the pipes constraints (63), the resistors constraints (69a), and the compressor constraints (74) and (75). Due to these constraints, this MINLP problem is non-convex.

4.4.9. Transient Storage Capacity Maximization. For the transient case, we focus on the optimization problem of maximizing the storage capacity of a

gas network. To this end, we consider the following situation: Let a nomination $q^{\text{nom}} \in \mathbb{R}^{N|V_s \cup V_t|}$ be given, i.e., supply and demand for each time step $n \in [N]$ and each entry and exit $v \in V_s \cup V_t$.

We assume that the supply and demand vector consisting of the elements of q^{nom} that correspond only to the first time step $n = 1$ are feasible for the stationary case, i.e.,, there exists an admissible configuration of the active elements satisfying all physical and technical constraints for $n = 1$. This problem is equivalent to the MINLP problem (81), where we set $c_a = 0$ and omit all constraints that include the power variable P_a for all $a \in A_{\text{cm}}$. We further assume that such a feasible solution is available, which we will use as a starting solution for the first time step of our optimization problem.

Moreover, there are time windows

$$[N_s^{\text{extra}}] \coloneqq \{1 \leq n_s^- , \dots , n_s^+ \leq N\},$$
$$[N_t^{\text{extra}}] \coloneqq \{1 \leq n_t^- , \dots , n_t^+ \leq N\},$$

in which, additionally to the nomination, a positive amount of gas $q^{\text{extra}} \in \mathbb{R}^{N|V_{s'} \cup V_{t'}|}$ with the bounds

$$0 \leq (q_{v,n}^{\text{extra}})^- \leq q_{v,n}^{\text{extra}} \leq (q_{v,n}^{\text{extra}})^+ \qquad \text{for all } v \in V_{s'} \cup V_{t'},\ n \in [N],$$

can be injected at selected entries $V_{s'} \subset V_s$ and withdrawn at selected exits $V_{t'} \subset V_t$, respectively. With

$$(q_{v,n}^{\text{extra}})^- = (q_{v,n}^{\text{extra}})^+ = 0 \qquad \text{for all } v \in V_{s'},\ n \notin [N_s^{\text{extra}}],$$
$$(q_{v,n}^{\text{extra}})^- = (q_{v,n}^{\text{extra}})^+ = 0 \qquad \text{for all } v \in V_{t'},\ n \notin [N_t^{\text{extra}}],$$

we replace the mass flow conservation constraints (60) for all $v \in V_{s'} \cup V_{t'}$ by

$$q_{(v^n,v^{n+1})} + \sum_{a \in \delta^{\text{out}}(v)} q_{a,n} - \sum_{a \in \delta^{\text{in}}(v)} q_{a,n} = q_{(v^{n-1},v^n)} + \hat{q}_{v,n} + q_{v,n}^{\text{extra}} \quad \text{for all } v \in V_{s'}, \tag{82a}$$

$$q_{(v^n,v^{n+1})} + \sum_{a \in \delta^{\text{out}}(v)} q_{a,n} - \sum_{a \in \delta^{\text{in}}(v)} q_{a,n} = q_{(v^{n-1},v^n)} + \hat{q}_{v,n} - q_{v,n}^{\text{extra}} \quad \text{for all } v \in V_{t'}, \tag{82b}$$

for all $n \in [N]$. Furthermore, we balance q^{extra} by

$$\sum_{v \in V_{s'}, n \in [N]} q_{v,n}^{\text{extra}} \ - \sum_{v \in V_{t'}, n \in [N]} q_{v,n}^{\text{extra}} \ = \ 0. \tag{83}$$

Our goal is to maximize this additional amount of gas that can be stored in the network.

This gas optimization problem can have different globally optimal solutions. We

therefore take a simplified version of the minimization of the compressor energy into account, however, with such low costs that it is almost negligible. This gives us optimal solutions that tend to switch on compressors only when necessary. A possible simple formulation for the energy minimization of all compressors $a = (v_1, v_2) \in A_{\mathrm{cm}}$ is

$$\min \sum_{a \in A_{\mathrm{cm}}} \gamma_1 \int (p_{v_2}(t) - p_{v_1}(t))\,\mathrm{d}t + \gamma_2 \left|\partial_t (p_{v_2}(t) - p_{v_1}(t))\right|, \tag{84}$$

where $\gamma_1,\ \gamma_2 > 0$ can be considered as costs and are small in our case. With the pressure increase $\triangle p_{a,n} = p_{v_2,n} - p_{v_1,n}$ for all compressors $a = (v_1, v_2) \in A_{\mathrm{cm}}$ and the change of the pressure increase over time $|\triangle p_{a,n} - \triangle p_{a,n-1}|$, the formulation (84) corresponds to

$$\min \sum_{\substack{a \in A_{\mathrm{cm}},\\ n \in [N]}} \gamma_1 \triangle p_{a,n} + \gamma_2 \left|\triangle p_{a,n} - \triangle p_{a,n-1}\right| \tag{85}$$

in our MINLP setting. We eliminate the absolute values in (85) by applying a common LP technique. To this end, we introduce a new variable $\triangle P_{a,n} \in \mathbb{R}_{>0}$ for all $a \in A_{\mathrm{cm}}$ and all $n \in [N]$ and substitute $\triangle P_{a,n} \coloneqq |\triangle p_{a,n} - \triangle p_{a,n-1}|$. Additionally, we add the constraints

$$\triangle P_{a,n} \geq \triangle p_{a,n} - \triangle p_{a,n-1}, \tag{86a}$$

$$\triangle P_{a,n} \geq \triangle p_{a,n-1} - \triangle p_{a,n}. \tag{86b}$$

Incorporating (85) into the objective function, we are now able to provide the complete problem of maximizing the storage capacity of a gas network as the MINLP problem:

$$\max \sum_{v \in V_{s'}, n \in [N]} q_{v,n}^{\mathrm{extra}} - \sum_{\substack{a \in A_{\mathrm{cm}},\\ n \in [N]}} \gamma_1 \triangle p_{a,n} + \gamma_2 \triangle P_{a,n} \tag{87a}$$

$$\text{s.t. mass flow conservation (60) and (82)–(83)}, \tag{87b}$$

$$\text{time coupling constraints (61)}, \tag{87c}$$

$$\text{pipes constraints (63)}, \tag{87d}$$

$$\text{shortcuts constraints (64)}, \tag{87e}$$

$$\text{(control) valves constraints (65) and (66)}, \tag{87f}$$

$$\text{resistors constraints (67)–(68)}, \tag{87g}$$

$$\text{compressors constraints (71)–(72)}, \tag{87h}$$

$$\text{switching conditions (79)}, \tag{87i}$$

$$\text{absolute values elimination (86)}, \tag{87j}$$

$$p_{v,n}^{-} \leq p_{v,n} \leq p_{v,n}^{+} \qquad \text{for all } v \in V,\ n \in [N], \tag{87k}$$

$$q^{-}_{a,n} \leq q_{a,n} \leq q^{+}_{a,n} \qquad \text{for all } a \in A,\, n \in [N], \tag{87l}$$

$$(q^{\text{extra}}_{v,n})^{-} \leq q^{\text{extra}}_{v,n} \leq (q^{\text{extra}}_{v,n})^{+} \qquad \text{for all } v \in V_{s'} \cup V_{t'},\, n \in [N], \tag{87m}$$

$$0 \leq \triangle P_{a,n} \qquad \text{for all } a \in A_{\text{cm}},\, n \in [N], \tag{87n}$$

$$s_{a,n} \in \{0, 1\} \qquad \text{for all } a \in A_{\text{ae}},\, n \in [N]. \tag{87o}$$

As in the stationary case, the variables $s_{a,n}$ and $\triangle p_{a,n}$ are now free and to optimize. The nonlinear parts of the MINLP model (87) are the pipes constraints (63) for all $n \in [N]$. Again, due to these constraints, this MINLP problem is non-convex. However, these nonlinear functions are univariate and therefore only depend one a single variable, which simplifies the piecewise linear approximations of the nonlinearities.

CHAPTER 5

AC Optimal Power Flow with Generator Switching

In this short chapter, we show how to model a fundamental problem in power system analysis as an MINLP. We use this MINLP to demonstrate the applicability of our adaptive MIP-based approach in Chapter 6.

In power system analysis, the focus is on the numerical evaluation of the electrical power flow in a power network. One of the core tasks is to determine an optimal power flow (OPF), i.e., the condition with the lowest cost per kW h that is generated. This can be used to plan future extensions of power supply grids and to identify optimal operating conditions for present systems.

Moreover, we extend the classical OPF problem by the possibility of switching power generation units on and off. Typically, these units must generate a minimal (positive) amount of power and therefore cannot be controlled continuously. The additional component of generation unit switching thus allows us a more comprehensive optimization of the OPF problem. We consider an alternating current (AC) power flow model.

AC OPF has been studied in the literature for a long time. Frank et al. (2012) provide a comprehensive literature overview of the methods that have been applied in the context of OPF. Carpentier (1962) was the first to model the AC OPF problem as a non-convex NLP. The first solution approach delivering a local optimal solution for this model is proposed by Dommel and Tinney (1968).

Many of the optimization approaches developed for AC OPF in recent years use semi-definite programming (SDP). Lavaei and Low (2012) show that the dual problem to AC OPF is an SDP problem and that if a certain "rank 1" condition for the optimal solution of the SDP is fulfilled, then the duality gap is zero. However, the "rank 1" condition can only be verified after solving the SDP. In addition, the condition depends on the parameters of the power network and the network itself. The approach therefore cannot provide satisfactory results in all cases. Another possibility is to reformulate the AC OPF using conic quadratic constraints resulting in a model with nonlinearities that are easier to handle; see Jabr (2008). This reformulation and the method using the dual SDP problem are combined by Kocuk et al. (2018). Based on the dual SDP problem, the authors construct second-order cone programming (SOCP) relaxations that are embedded in a branch-and-cut framework. Additional cutting

planes, convex envelopes, and bound tightening techniques complement the approach that is the current state-of-the-art solver for AC OPF. Moreover, Coffrin et al. (2016) give an overview of relaxation methods for AC OPF and present a comparison of the methods.

We point out that a lot of research has been carried out that incorporates direct current (DC) flow models, which are a linear approximation of AC models. The advantage is that DC models circumvent the nonlinear parts of the AC model. Thus, very large power networks can be considered. Electricity market models, for instance, often use DC models; see the works by Krebs et al. (2018), Piao et al. (2017), Schweppe et al. (1988), and Stigler and Todem (2005). For electrical distribution networks, which we consider in our computational part in Chapter 6, the DC approximation is not accurate enough. Purchala et al. (2005) analyze for which power networks the DC approximation is useful.

The combination of AC OPF and discrete decisions has rarely been discussed in the literature so far. Bai et al. (2016) present a two-level approach for OPF with additional switching of the transmission lines of the power network. First, the authors compute optimal discrete decisions with a DC model and then deduce feasible decisions from the DC decisions with an AC model. The same problem is addressed by Kocuk et al. (2017), who propose a method that constructs mixed-integer SOCP relaxations based on the SOCP relaxations for AC OPF introduced by Jabr (2008) and solves the problem by branch-and-cut.

Another class of OPF problems with discrete decisions are the time-dependent unit commitment optimal power flow problems. These are usually solved using decomposition methods with a discrete master problem and an AC OPF subproblem. The drawback here, however, is that the AC OPF subproblem can often not be solved to global optimality; see for example Castillo et al. (2016). Schultz and Wollenberg (2017) investigate such unit commitment problems under uncertainty of load and power infeed from renewable energies. Furthermore, we refer to Subsection 4.4.7 for more references on the unit commitment problem.

AC OPF problems with switching of the generator units are even less studied in literature. In the Ph.D. thesis of Kuhn (2011) heuristic solutions for this problem are developed. Very recently, Salgado et al. (2018) have proposed a mixed-integer method for solving this problem. Based on a mixed-integer SDP relaxation of the problem, they derive two MIP approximations that are subsequently solved: an inner and an outer MIP approximation. The inner MIP approximation improves feasible solutions, while the outer MIP approximation provides a relaxation and thus delivers dual bounds.

On the one hand, the mathematical challenges of the AC OPF with generator

switching arise from the non-convexity of the AC power flow equations. On the other hand, switching generation units entails discrete decisions. Altogether, this leads to MINLP problems that are hard to solve in practice by state-of-the-art solvers. This is aggravated by the fact that many AC models contain trigonometric functions, which are not supported by many MINLP solvers such as Baron and SCIP.

This chapter primarily summarizes known facts and models from the literature. As mentioned before, we solve the MINLP problems that are introduced in this chapter in Chapter 6.

We now describe the MINLP model for the AC OPF with generator unit switching. The model is based on the power flow models of Kocuk et al. (2017). We represent a power network by an undirected graph $N = (B, L)$, where the node set B denotes the buses and the edge set L the transmission lines. The subset $U \subset B$ denotes the generation units of the system. Our goal is to minimize the production costs of the generators such that all physical and technical constraints are satisfied. The physical restrictions are essentially described by Ohm's and Kirchoff's Law, while the technical restrictions model limits of the transmission lines and of the production quantities of the generators.

A power supply system can be characterized by a nodal admittance matrix $Y \in \mathbb{C}^{|B|\times|B|}$ that describes the nodal admittance of the buses, i.e., roughly speaking, it contains information about the network topology and transmission parameters. For each transmission line $(k, l) \in L$, this matrix has a component $Y_{kl} = G_{kl}+iB_{kl}$ where $i = \sqrt{-1}$. With the shunt conductance g_{kk} and susceptance b_{kk} at bus $k \in B$, we also have $G_{kk} = g_{kk} - \sum_{k\neq l} G_{kl}$ and $B_{kk} = b_{kk} - \sum_{k\neq l} B_{kl}$. We denote by p^g_k and q^g_k the real and reactive power output of the generator for all $k \in B$ and assume lower and upper bounds $(p^g_k)^-, (q^g_k)^-$ and $(p^g_k)^+, (q^g_k)^+$. If $k \in B \setminus U$, then we simply set $p^g_k = 0$ and $q^g_k = 0$. The corresponding demand at bus k is given by $\hat{p}^d_k$ and $\hat{q}^d_k$. As in Chapter 4, the hat symbol again indicates input data. Moreover, we denote the real and reactive power on a transmission line $(k, l) \in L$ by p_{kl} and q_{kl}.

In the following, we introduce two variants of the AC OPF problem, both of which are related to the form the complex voltage V_k is described in. First, the rectangular form specifies the complex voltage by $V_k = e_k + if_k$. Second, with the polar form, we obtain $V_k = |V_k|(\cos\theta_i + i\sin\theta_i)$, while

$$|V_k| = \sqrt{e_k^2 + f_k^2} \tag{88}$$

is the voltage magnitude with lower and upper bounds $|V_k|^-$ and $|V_k|^+$, and θ_i is the phase angle.

The rectangular model of the AC OPF problem is given by:

$$\min \sum_{k\in U} C_k(p_k^g, s_k) \tag{89a}$$

$$\text{s.t. } p_k^g - \hat{p}_k^d = g_{kk}(e_k^2 + f_k^2) + \sum_{l\in\delta(k)} p_{kl} \qquad \text{for all } k \in B, \tag{89b}$$

$$q_k^g - \hat{q}_k^d = -b_{kk}(e_k^2 + f_k^2) + \sum_{l\in\delta(k)} q_{kl} \qquad \text{for all } k \in B, \tag{89c}$$

$$(|V_k|^-)^2 \le e_k^2 + f_k^2 \le (|V_k|^+)^2 \qquad \text{for all } k \in B, \tag{89d}$$

$$p_{kl} = -G_{kl}(e_k^2 + f_k^2 - e_k e_l - f_k f_l) - B_{kl}(e_k f_l - e_l f_k) \quad \text{for all } (k,l) \in L, \tag{89e}$$

$$q_{kl} = B_{kl}(e_k^2 + f_k^2 - e_k e_l - f_k f_l) - G_{kl}(e_k f_l - e_l f_k) \quad \text{for all } (k,l) \in L, \tag{89f}$$

$$p_{kl}^2 + q_{kl}^2 \le (d_{kl}^+)^2 \qquad \text{for all } (k,l) \in L, \tag{89g}$$

$$(p_k^g)^- s_k \le p_k^g \le (p_k^g)^+ s_k \qquad \text{for all } k \in U, \tag{89h}$$

$$(q_k^g)^- \le q_k^g \le (q_k^g)^+ \qquad \text{for all } k \in U, \tag{89i}$$

$$p_{kl}^- \le p_{kl} \le p_{kl}^+ \qquad \text{for all } (k,l) \in L, \tag{89j}$$

$$q_{kl}^- \le q_{kl} \le q_{kl}^+ \qquad \text{for all } (k,l) \in L, \tag{89k}$$

$$-|V_k|^- \le e_k, f_k \le |V_k|^+ \qquad \text{for all } k \in B, \tag{89l}$$

$$s_k \in \{0,1\} \qquad \text{for all } k \in U. \tag{89m}$$

We compute the lower and upper bounds of e_k, f_k in (89l) via (88). The lower and upper bounds p_{kl}^-, q_{kl}^- and p_{kl}^+, q_{kl}^+ of p_{kl}, q_{kl} in (89h) and (89i) are computed using (89e) and (89f) and the bounds of e_k and f_k.

Each sum $C_k(p_k^g, s_k)$ of the objective function is either linear or quadratic in p_k^g. Additionally, each constant term of a cost function of a generator is multiplied in the objective function with the binary variable s_k. The constraints (89b) and (89c) ensure the conservation of active and reactive power flows at each bus of the network. As usually, the real and reactive power p_{kl} and q_{kl} are computed as in (89e) and (89f). Additionally, we restrict the apparent power d_{kl} on line $(k,l) \in L$ by d_{kl}^+. Since $d_{kl}^2 = p_{kl}^2 + q_{kl}^2$, this restriction is enforced by (89g). We model the switching of the generator units by (89h). The reactive power output q_k^g of the generator unit k is required for the voltage maintenance and can be continuously controlled starting at zero. In addition, the control of q_k^g is independent of the control of p_k^g in our model, which is consistent with the models of Vittal and Bergen (2000). Therefore, no binary variables are involved in (89i). Finally, we point out that one of the variables f_k must be set to zero, because the voltage angle at a reference node is fixed at zero. This prevents the optimization problem (89) from having an infinite number of globally optimal solutions.

We can use the polar form of the complex voltage to obtain a model that contains nonlinearities that are easier to handle. For each nonlinear expression in (89), we then obtain the following alternative:

$$e_k^2 + f_k^2 = |V_k|^2, \tag{90a}$$

$$e_k e_l + f_k f_l = |V_k|\,|V_l| \cos(\theta_k - \theta_l), \tag{90b}$$

$$e_k f_l - e_l f_k = -|V_k|\,|V_l| \sin(\theta_k - \theta_l). \tag{90c}$$

We now introduce the variable substitutions

$$c_{kk} \coloneqq e_k^2 + f_k^2, \qquad c_{kl} \coloneqq e_k e_l + f_k f_l, \qquad t_{kl} \coloneqq e_k f_l - e_l f_k \tag{91}$$

for each bus $k \in B$ and line $(k,l) \in L$. With (90) and (91), we consequently obtain the polar model of the AC OPF problem:

$$\min \sum_{k \in U} C_k(p_k^g, s_k) \tag{92a}$$

$$\text{s.t. } p_k^g - \hat{p}_k^d = g_{kk}(c_{kk}) + \sum_{l \in \delta(k)} p_{kl} \qquad \text{for all } k \in B, \tag{92b}$$

$$q_k^g - \hat{q}_k^d = -b_{kk}(c_{kk}) + \sum_{l \in \delta(k)} q_{kl} \qquad \text{for all } k \in B, \tag{92c}$$

$$(|V_k|^-)^2 \le c_{kk} \le (|V_k|^+)^2 \qquad \text{for all } k \in B, \tag{92d}$$

$$p_{kl} = -G_{kl}(c_{kk} - c_{kl}) - B_{kl} t_{kl} \qquad \text{for all } (k,l) \in L, \tag{92e}$$

$$q_{kl} = B_{kl}(c_{kk} - c_{kl}) - G_{kl} t_{kl} \qquad \text{for all } (k,l) \in L, \tag{92f}$$

$$c_{kl} = c_{lk}, \quad t_{kl} = -t_{lk} \qquad \text{for all } (k,l) \in L, \tag{92g}$$

$$p_{kl}^2 + q_{kl}^2 \le (d_{kl}^+)^2 \qquad \text{for all } (k,l) \in L, \tag{92h}$$

$$c_{kl}^2 + t_{kl}^2 = c_{kk} c_{ll} \qquad \text{for all } (k,l) \in L, \tag{92i}$$

$$\theta_l - \theta_k = \operatorname{arctan2}(t_{kl}, c_{kl}) \qquad \text{for all } (k,l) \in L, \tag{92j}$$

$$(p_k^g)^- s_k \le p_k^g \le (p_k^g)^+ s_k \qquad \text{for all } k \in U, \tag{92k}$$

$$(q_k^g)^- \le q_k^g \le (q_k^g)^+ \qquad \text{for all } k \in U, \tag{92l}$$

$$p_{kl}^- \le p_{kl} \le p_{kl}^+ \qquad \text{for all } (k,l) \in L, \tag{92m}$$

$$q_{kl}^- \le q_{kl} \le q_{kl}^+ \qquad \text{for all } (k,l) \in L, \tag{92n}$$

$$c_{kl}^- \le c_{kl} \le c_{kl}^+ \qquad \text{for all } (k,l) \in L, \tag{92o}$$

$$t_{kl}^- \le t_{kl} \le t_{kl}^+ \qquad \text{for all } (k,l) \in L, \tag{92p}$$

$$s_k \in \{0,1\} \qquad \text{for all } k \in U. \tag{92q}$$

The nonlinear part of the MINLP model (92) is (92h)–(92j). Finally, we would like to remark that if $-\pi/2 < \theta_l - \theta_k < \pi/2$ holds, which is often the case in practice,

we can reformulate the constraint (92j) by

$$c_{kl} \tan(\theta_l - \theta_k) = t_{kl}. \tag{93}$$

This is equivalent to (92j), but much easier to handle in terms of nonlinearity. In the remainder of this thesis, we always refer to real power production of a generator, whenever we only indicate the power production of a generator.

CHAPTER 6

Computational Results

In this chapter, we present numerical results that demonstrate the practicability of our adaptive approach from Chapter 3 to application-driven MINLPs with a considerable discrete aspect. To this end, we consider MINLP problems that are difficult to solve by state-of-the-art MINLP solvers. First, we solve both stationary and transient gas network optimization problems. As a second application, we solve optimal power flow problems that include the switching of generator units. Moreover, we compare our adaptive MIP-based approach with state-of-the-art MINLP solvers on several MINLP instances.

This chapter is structured as follows. Section 6.1 addresses problems in the field of gas transport network optimization. We apply our adaptive approach on both stationary and transient gas transport optimization problems. Moreover, we discuss occurring implementation issues. In Section 6.2, we solve optimal power flow problems with additional generator switching. As in the previous section, we address occurring implementation issues. We conclude the chapter in Section 6.3 with some final remarks.

Most of the computational results in Section 6.1 have been carried out in the two publications

> R. Burlacu, H. Egger, M. Groß, A. Martin, M. E. Pfetsch, L. Schewe, M. Sirvent, and M. Skutella (2018). "Maximizing the storage capacity of gas networks: a global MINLP approach". In: *Optimization and Engineering*, pp. 1–31. DOI: 10.1007/s11081-018-9414-5.

> R. Burlacu, B. Geißler, and L. Schewe (2019). "Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes". In: *Optimization Methods and Software*, pp. 1–28. DOI: 10.1080/10556788.2018.1556661.

The author of this thesis has made a significant contribution to the conception and elaboration of the implementation and to the computations in these publications. Section 6.2 is a joint work of the author of this thesis and Kevin-Martin Aigner. The latter implemented Algorithm 1 and the details discussed in Subsection 6.2.1 in a Python framework under the guidance of the author of this thesis. Furthermore, the results in this section are part of a working paper with contributions by the author

and Kevin-Martin Aigner, Frauke Liers, and Alexander Martin. All computational results in Section 6.2 are presented for the first time in this thesis.

6.1. Optimization of Gas Transport Networks

In this section, we consider both stationary and transient gas network optimization problems, which are introduced in Chapter 4. For the stationary case, we construct all MINLP instances from the gas network GasLib-582 and for the transient case from the network GasLib-11; see Schmidt et al. (2017).

The computations are carried out using the C++ software framework Lamatto++, see LaMaTTO++ (2015), on a cluster using 12 cores of a machine with two Xeon 5650 "Westmere" chips (12 cores + SMT) running at 2.66 GHz with 12 MB Shared Cache per chip and 24 GB of RAM. Furthermore, we utilize Gurobi 6.0.4 as MIP solver, using the 12 cores mentioned above for the parallel solution of each MIP. The state-of-the-art MINLP solvers Baron 17.1.2 and SCIP 3.2 are both used within GAMS 24.8.3; see GAMS (2017). Note that for a fair comparison, Baron and SCIP also have 12 cores available and run in parallel.

6.1.1. Stationary Compressor Energy Cost Minimization. In this subsection, we present computational results for the GasLib-582 gas network, which consists of 582 nodes, 278 pipes, 5 compressor stations, 23 control valves, 8 resistors, 26 valves and 269 shortcuts. The subsequent computations are based solely on a subset of all scaled (95 % of flow amount) nominations provided by GasLib-582 that consists of more than 4000 nominations. Each of these nominations corresponds in our case to a separate instance, i.e., a separate MINLP problem. First, we provide an insight into implementation issues that arise in the context of stationary compressor energy cost minimization. Finally, we show both a detailed example computation and a comparison with the state-of-the-art MINLP solvers Baron and SCIP.

6.1.1.1. *Implementation Issues.* The implementation details we discuss now can be thought of as either preprocessing techniques or parameter tuning, based on empirical values, for Algorithm 1 concerning MINLP problems in the fashion of (81).

Since Algorithm 1 iteratively solves MIP relaxations of the MINLP (81), the run time of the algorithm mainly depends on the size of the MIP relaxations. Based on (23), the complexity of a relaxation of a nonlinear function drastically increases with the function's dimension. Hence, we reformulate the MINLP (81) by constructing expression trees in order to reduce the dimension of the nonlinear functions; see again Geißler (2011) and Belotti et al. (2013) for general details.

In our case, we add variables f_v and the nonlinear constraints $f_v = p_v^2$ for each node v that is incident to an arc $a \in A_{\mathrm{pi}} \cup A_{\mathrm{rs}}$, i.e., for which we have $a \in \delta^-(v)$ or

$a \in \delta^+(v)$. For each node v that is incident to an arc $a \in A_{\mathrm{cm}}$, we add $f_v^{\kappa^+} = p_v^{(\kappa-1)/\kappa}$ if $a \in \delta^+(v)$ and $f_v^{\kappa^-} = p_v^{-(\kappa-1)/\kappa}$ if $a \in \delta^-(v)$ and the corresponding variables.

Furthermore, we add $f_a^{\mathrm{pi}} = |q_a|q_a$ for each $a \in A_{\mathrm{pi}}$, $f_a^{\mathrm{rs}^+} = |\triangle p_a|\triangle p_a$ and $f_a^{\mathrm{rs}^-} = |q_a|q_a$ for each $a \in A_{\mathrm{rs}}$, and $f_a^{\mathrm{cm}} = f_{v_2}^{\kappa^+} f_{v_1}^{\kappa^-}$ for each $a = (v_1, v_2) \in A_{\mathrm{cm}}$ together with the corresponding variables.

Herewith, we reformulate for each $a = (v_1, v_2) \in A_{\mathrm{pi}}$ the pressure loss equations (63) by the linear constraint

$$f_{v_2} - f_{v_1} = \beta_a f_a^{\mathrm{pi}}. \tag{94}$$

Similarly, we reformulate the equation (69a) that model the pressure drop for each resistor $a = (v_1, v_2) \in A_{\mathrm{rs}}$ by

$$f_{v_1} - f_{v_2} + f_a^{\mathrm{rs}^+} = \gamma_a f_a^{\mathrm{rs}^-}. \tag{95}$$

As third reformulation, we replace the equations (74) describing the change in the specific enthalpy $H_{\mathrm{ad},a}$ for each compressor $a = (v_1, v_2) \in A_{\mathrm{cm}}$ by the linear constraint

$$H_{\mathrm{ad},a} = \frac{\kappa}{\kappa - 1} c^2 (f_a^{\mathrm{cm}} - 1). \tag{96}$$

Moreover, we perform the preprocessing methods described by Geißler et al. (2015a) both tightening the bounds of the flow and pressure variables and reducing the amount of nonlinear functions that are taken into account for relaxation.

Unlike the formal description of Algorithm 1, in which in every refinement step, every nonlinear function $f \in \mathcal{F}$ with an approximation error larger than ϵ_f is refined, we only refine a certain amount of all nonlinear functions that are worst regarding to a specific score. We build this score upon the set

$$M^i := \left\{ \frac{\dim(f)^2 |f(x_f^i) - y_f^i|}{\epsilon_f} : f \in \mathcal{F} \text{ and } |f(x_f^i) - y_f^i| > \epsilon_f \right\}, \tag{97}$$

in every iteration of Algorithm 1. In order to obtain the elements of M^i with corresponding nonlinear functions that have to be refined, we pursue marking strategies applied in adaptive finite elements methods; see Verfürth (1996) and Dörfler (1996). With η^i as the maximum score contained in M^i, we refine any $f \in \mathcal{F}$, which has a score larger than $\theta\eta^i$. Typically, $\theta = 0.5$ is chosen. However, since the run time of an MIP grows exponentially with its size, we set $\theta = 0.75$ in our case. In this way, we can speed up the algorithm by solving many smaller-sized MIP relaxations instead of few bigger-sized ones. Since we use the longest-edge bisection as refinement procedure in Algorithm 1 and therefore $|f(x_f^i) - y_f^i| \leq \epsilon_f$ holds for any $f \in \mathcal{F}$ after a finite number of refinement steps (see Chapter 3), it follows that this approach still is convergent. Note that the square of the dimension of f in (97) corresponds to (23).

Beyond that, whenever a feasible solution of an MIP relaxation is found, we fix all

discrete variables of the underlying MINLP problem according to the MIP solution obtaining an NLP problem, which we solve to local optimality. However, we use the NLP solver's standard feasibility tolerance of 10^{-6} instead of the error bounds ϵ_f. With this relatively inexpensive primal heuristic, we are often able to find feasible solutions for the MINLP problem quite rapidly. In the subsequent computations, we use CONOPT3, which in our case performs better than CONOPT4, as local NLP solver within GAMS 24.8.3.

Finally, since we are only interested in a single globally optimal MIP solution satisfying some given error bounds ϵ_f for all $f \in \mathcal{F}$, it is not necessary to solve every MIP relaxation to global optimality. Instead, it suffices to solve only every kth MIP relaxation (and the first and last one, respectively) to global optimality and to use bigger relative MIP gaps otherwise. We chose $k = 50$ in our case. The relative MIP gaps depend on whether feasible solutions for the MINLP problems are found or not. With u as upper bound corresponding to the incumbent solution found by the local NLP solver and l as lower bound obtained by MIP relaxations as in Algorithm 1, the relative MIP gap is set to $(u - l)/(2u)$, if an upper bound is available. Otherwise, a relative MIP gap of 10 % is chosen.

Concerning the error bounds in Algorithm 1, we choose 20.0 bar for the pressure loss equations and 20.0 MW for the equations describing the power consumption as initial approximation errors. Due to the complexity of the underlying MINLP problems, final approximation errors for the example computation are set to 2.0 bar and 0.2 MW, respectively. For the rest, we choose 1.0 bar and 0.1 MW as final approximation errors for practical reasons, because even in this case all computations run into time limit.

Finally, we point out that Baron and SCIP are given the same reformulations (94)–(96) of the nonlinear constraints.

6.1.1.2. *Example Computation.* We start with an example instance showing the practical performance of Algorithm 1 and that even with coarse final approximation errors good solutions can be computed.

The result presented in Table 6.1 is obtained by solving a randomly chosen nomination of GasLib-582 (with ID "nomination_warm_95_2051") that could be solved within a time limit of 10 h. The first column indicates the iteration number of Algorithm 1. The next three columns contain the size of the corresponding MIP relaxation, split into the number of continuous and binary variables and the number of constraints. Subsequently, the best lower bound l and upper bound u are given, followed by the run time (in s) of the actual MIP relaxation and the NLPs that are solved to local optimality. The next column contains the relative gap computed by $(u - l)/u$. In the last two columns, the number of violated constraints and the number

TABLE 6.1. Example computation of the GasLib-582 instance with ID "nomination_warm_95_2051" by Algorithm 1 combined with CONOPT3 as local NLP solver.

iter	cont	bin	cons	lower	upper	t_{MIP}	t_{NLP}	gap	viol	ref
0	2927	593	5714	114.40	416.02	1.85	3.77	72.50 %	77	1
1	2928	594	5716	114.40	316.80	1.12	2.62	63.89 %	74	4
5	2937	602	5733	114.40	316.80	1.43	3.06	63.89 %	75	1
25	3002	653	5849	135.59	316.80	5.88	8.57	57.20 %	79	5
49	3097	722	6013	218.59	316.80	1.91	5.60	31.00 %	77	2
50	3101	725	6020	243.67	316.80	15.64	7.84	23.08 %	72	1
51	3103	726	6023	243.67	316.80	6.44	3.17	23.08 %	73	1
75	3224	826	6244	265.82	316.80	10.93	24.34	16.09 %	65	2
99	3313	896	6403	294.40	316.80	3.08	10.68	7.07 %	61	2
100	3315	898	6407	301.61	316.80	284.05	20.19	4.79 %	54	1
149	3477	1051	6722	301.61	316.80	13.34	5.44	4.79 %	14	2
150	3479	1053	6726	305.89	316.80	233.16	1.55	3.44 %	14	1
194	3592	1154	6940	305.89	316.80	771.75	3.89	3.44 %	4	1
195	3592	1154	6940	306.88	316.80	126.78	4.93	3.13 %	4	1
total						4274.53	1156.66			537

of constraints which are chosen for refinement are indicated. Finally, the last row gives an overview of the total run time and number of constraints chosen for refinement.

After about 1.5 h, Algorithm 1 is able to find an optimal solution for the MINLP problem such that no constraint is violated by more than 2.0 bar and 0.2 MW, respectively. Combined with CONOPT3, even a feasible solution for the MINLP problem could be found, which is optimal up to a relative gap of almost 3 %.

Note that the final MIP relaxation consists of 3592 continuous and 1154 binary variables, and 6940 constraints only, whereas an MIP relaxation constructed by piecewise linear approximations satisfying the final approximation errors everywhere, consists of 39 193 continuous, 21 735 binary variables, and 61 709 constraints. The first feasible solution for the latter MIP is found after a run time of almost 8 h, whereas after a total run time of 10 h lower and upper bounds are 131.35 and 594.50, respectively, resulting in a relative MIP gap of more than 77 %. We can see that although final approximation errors are relatively high, adaptivity in Algorithm 1 is crucial for a reasonable overall run time. Since the run time to solve an MIP problem exponentially increases with the size of the MIP, adaptivity becomes even more important if final approximation errors are tighter; see Geißler (2011) for a more detailed discussion.

Since the upper bound in Table 6.1 remains unchanged after the first iteration,

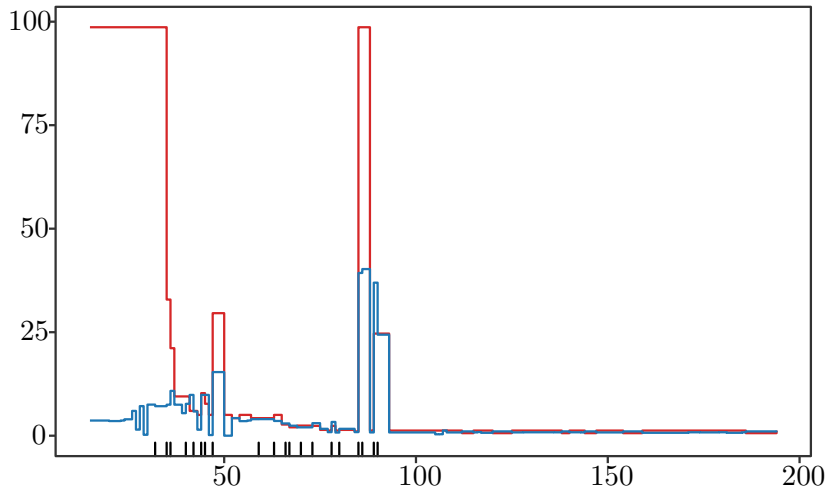


FIGURE 6.1. Iteration log for a nonlinear function $f = xy$ that shows the maximal approximation error on the simplex containing the incumbent solution of the corresponding MIP relaxation (red lines), the approximation error of the solution (blue lines) and the iterations in which the approximation is refined (black dash).

we conclude that with our approach even very coarse initial approximations can lead to solutions that are feasible for the MINLP and optimal within a relative gap of almost 3 %, in a couple of seconds only.

We conclude the example presentation of our approach with Figure 6.1 that shows an iteration log of a nonlinear function $f\colon \mathbb{R}^2 \to \mathbb{R}, (x, y) \mapsto xy$ with domain $D_f = [1.028, 1.206] \times [30.350, 1139.280]$ and Lipschitz-constant $L_f = \operatorname{diam}(D_f)$. This function corresponds to the power consumption of a compressor; see (75). The final approximation error for f is set to 1.265. We plot the maximal approximation error on the simplex that contains the incumbent solution of the corresponding MIP relaxation by red lines, whereas the approximation error for the solution itself is given by blue lines. Moreover, whenever the function is marked for refinement a black dash is drawn.

As expected, the error of the MIP solution tends to a value smaller than 1.265 as more refinement steps are performed on f. The staircase-shaped descending of the error is characteristic for Algorithm 1. Often, solutions of consecutive MIP relaxations have a local affinity, i.e., the projections of the solutions on the domain D_f are close to each other. Hence, depending on the specific triangulation of the piecewise linear approximation, in some cases we need more than one refinement per nonlinear function in order to scale the error down to the next level. We point out, that exploiting this phenomenon, e.g., by refining not only the simplex containing the MIP solution, but also adjacent simplices, may lead to further improvements.

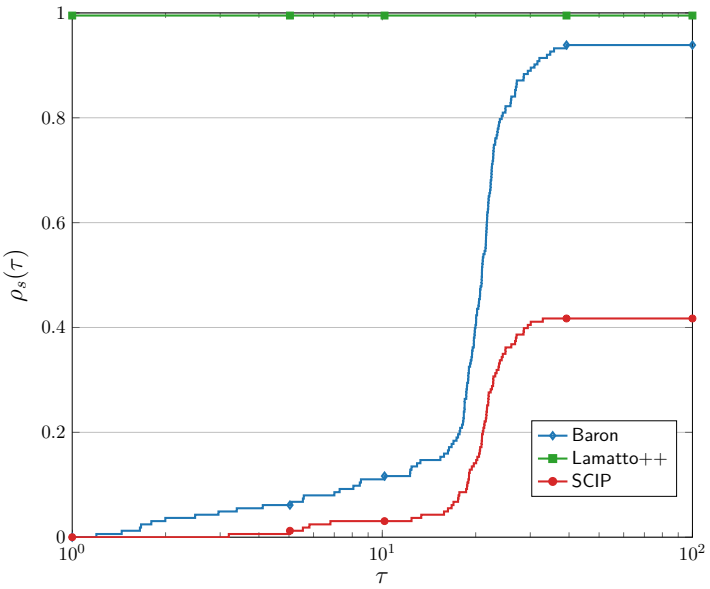


FIGURE 6.2. Performance profiles for Algorithm 1 combined with CONOPT3 (Lamatto++), Baron, and SCIP comparing relative gaps obtained after a time limit of 4 h.

With 20 refinements from iteration 0 to 90 and 2 refinements performed on f for the initial approximation, in total, 22 refinements and 195 iterations are enough to obtain an optimal MIP solution satisfying the final error bounds both for f and any other nonlinear function occurring in the MINLP. Regarding the required number of refinement steps, compared to the worst case estimation of $2^{\tilde{N}}$ as in (12), where $\tilde{N} = 3\lceil \ln\left(\frac{2\,\mathrm{diam}(D_f)\,\mathrm{diam}(D_f)}{1.265}\right)/\ln\left(\frac{2}{\sqrt{3}}\right)\rceil = 303$, a far less amount is needed in this case.

Finally, we remark that although the total run time of 1.5 h of Algorithm 1 and CONOPT3 appears to be long in order to obtain an MINLP solution that is optimal within a relative gap of 3 %, it is quite short compared to other MINLP solvers.

6.1.1.3. *Comparison with State-of-the-Art* MINLP *Solvers.* We demonstrate the advantage of Algorithm 1 combined with the local NLP solver CONOPT3 over the state-of-the-art global MINLP solvers Baron and SCIP in Figure 6.2 by comparing relative gaps computed by $(u - l)/u$, with the aid of performance profiles; see again Dolan and Moré (2002) and Chapter 2 for more details.

In order to calculate those profiles, we randomly choose 200 out of roughly 4000 nominations provided by GasLib-582; see Table A.2 in Appendix A for their IDs. Considering only those nominations for which at least one solver was able to compute a feasible solution within the total time limit of 4 h, 163 nominations remain for the performance profiles. Moreover, due to the complexity of the problems, each of the 163

TABLE 6.2. Frequency of different solution statuses given by Baron, Algorithm 1 combined with CONOPT3 (Lamatto++), and SCIP for a set of 200 randomly chosen nominations provided by GasLib-582.

	l and u	only u	only l	infeasible	none	error
Baron	49	104	0	35	12	0
Lamatto++	163	0	0	34	3	0
SCIP	13	55	0	19	108	5

nominations ran into timeout, i.e., the relative gap could not be closed. A second reason for a timeout in our case is that there is no optimal solution for an MIP relaxation that satisfied all final approximation errors.

We remark that the nomination with ID "nomination_warm_95_2051" from the previous subsection is also included in the 200 nominations. However, we choose 1.0 bar and 0.1 MW instead of 2.0 bar and 0.2 MW as final approximation errors. After a total run time of 4 h, the best lower bound for this specific nomination is now 307.95, while the best upper bound is again 316.80 as in Table 6.1. This results in a relative gap of about 2.8 %, which is slightly smaller than the one obtained in the previous subsection.

As we can see in Figure 6.2, our approach is clearly preferable to Baron and SCIP applied to gas network optimization problems described in GasLib-582. In all cases, Algorithm 1 combined with CONOPT3 computes the smallest gap, while Baron and SCIP in no case were capable to compute the smallest gap. Additionally, in most cases the relative gaps obtained by our approach are smaller than 10 % and differ from the relative gaps computed by Baron and SCIP by a magnitude of almost 10, which can be deduced from Figure 6.2.

In order to gain an in-depth look at Figure 6.2, we compare both lower bounds l and upper bounds u in Figure 6.3. The upper bounds computed by CONOPT3 after fixing all discrete variables corresponding to an MIP solution obtained by Algorithm 1 are slightly tighter than the ones attained by Baron and clearly tighter than the ones computed by SCIP. The main benefit of our approach, however, derives from the fact that MIP relaxations obtained by Algorithm 1 yield notably tighter lower bounds than the lower bounds computed by Baron and SCIP. Moreover, the MIP relaxations can be solved both reliably and fast utilizing mature MIP technology. Baron and SCIP both struggle to deliver reasonable lower bounds and perform roughly the same in this respect.

We summarize the comparison presenting the different solution statuses given by every solver and their frequency in Table 6.2. The first column indicates nominations with both an upper bound u and a lower bound l greater than zero, whereas in the

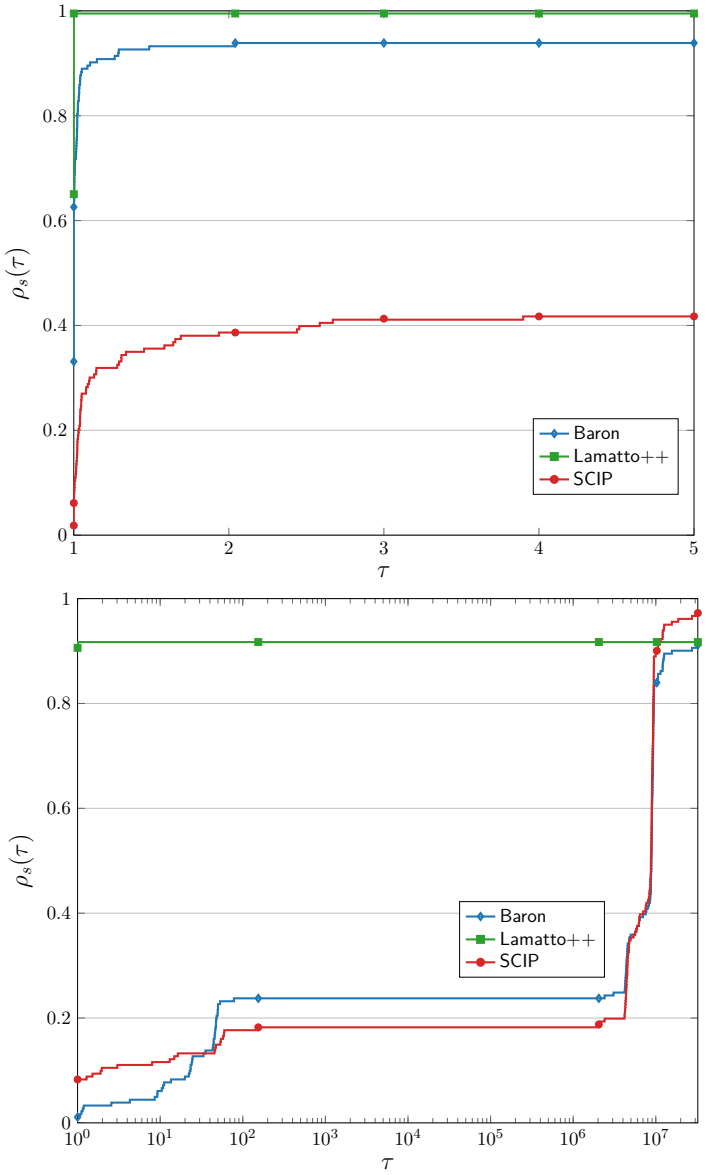


FIGURE 6.3. Performance profiles for Algorithm 1 combined with CONOPT3 (Lamatto++), Baron, and SCIP comparing upper bounds u (above) and lower bounds l (below) obtained after a time limit of 4 h.

second and third column only those nominations are taken into account, for which only an upper bound u, and a lower bound l, respectively, is available. The next column contains the number of nominations that are detected as infeasible, followed by a column in which nominations without any solution status are considered. The last column gives the number of nominations that are declared as infeasible, and for which at least one of the other solvers is able to compute a feasible solution for the

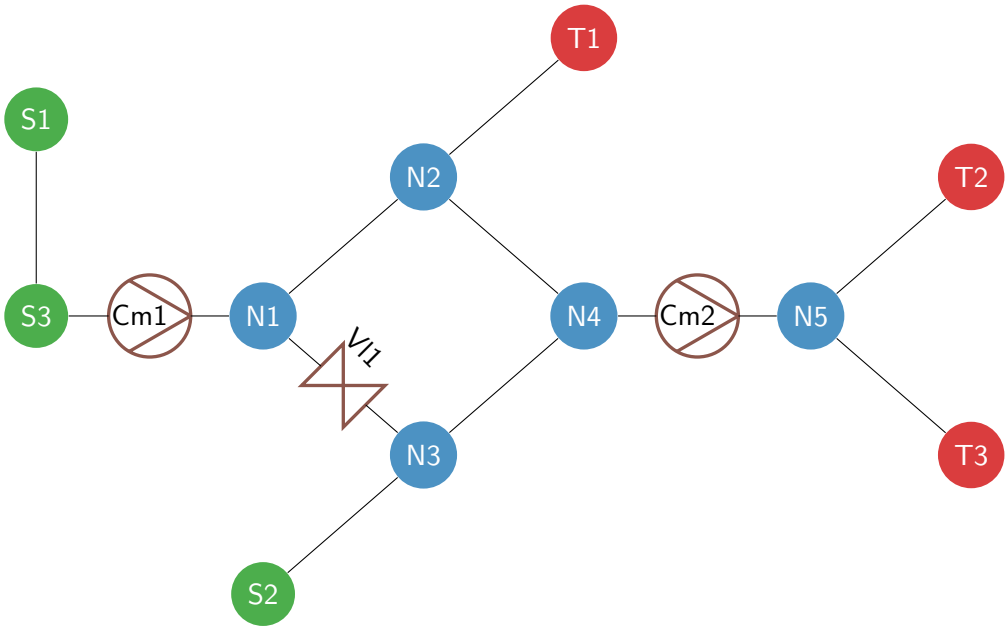


FIGURE 6.4. The GasLib-11 network.

corresponding nomination. Algorithm 1 combined with CONOPT3 is able to find feasible solutions for all 163 nominations with additional non-trivial lower bounds. As already pointed out, Baron struggles with computing lower bounds, as Baron is able to find feasible solutions for 153 nominations, but only in 49 cases a non-trivial lower bound is found as well. In this regard, SCIP is clearly inferior. Moreover, in 5 cases SCIP detects infeasibility of the MINLP, which presumably is false, since our approach and Baron are able to find feasible solutions. In this matter, our approach and Baron are almost coherent. For more results concerning the detection of infeasibility we refer to Joormann et al. (2015) and Hiller et al. (2015).

Finally, we point out that our approach is also suitable for obtaining high-quality solutions at short run time. In fact, for 161 out of 163 nominations, Algorithm 1 is able to compute discrete decisions proven to be feasible for the corresponding MINLP by the local NLP solver CONOPT3 after a total run time limit of just 10 min. Furthermore, as we can see in Table 6.1, Algorithm 1 delivers reasonable lower bounds after only a few iterations, even with coarse initial approximations.

6.1.2. Transient Storage Capacity Maximization. In this subsection, we present computational results for the GasLib-11 gas network. The GasLib-11 network depicted in Figure 6.4 consists of three entries S1–S3, five interior vertices N1–N5, and three exits T1–T3. Two compressors Cm1 and Cm2 are installed between S3 and N1, and N4 and N5, respectively. We have lower and upper bounds $r_a^- = 1.0895$ and

$r_a^+ = 1.6009$ for the compression ratio of both compressors. All eight pipes have a length ℓ_a of 55 km, a diameter D_a of 0.5 m, and a roughness of 0.1 mm resulting in a friction factor $\lambda_a = 0.0137$. The two vertices T1 and T2 have a lower pressure bound of 40 bar and an upper pressure bound of 60 bar. All other vertices have lower and upper pressure bounds of 40 bar and 70 bar. The network also includes a valve Vl1 between N1 and N3.

Again, we first provide an insight into the implementation details in the context of transient storage capacity maximization. We then illustrate the applicability of our transient flow model by investigating pressure and flow waves within pipes. Thereafter, we solve an MINLP problem in the fashion of (87) based on GasLib-11. Finally, we give a short comparison with the state-of-the-art global MINLP solvers Baron and SCIP and local MINLP solvers α-ECP (see again Westerlund and Lundqvist (2001)), BONMIN (see again Bonami et al. (2008)), and Knitro by Byrd et al. (2006), all within GAMS 24.8.3.

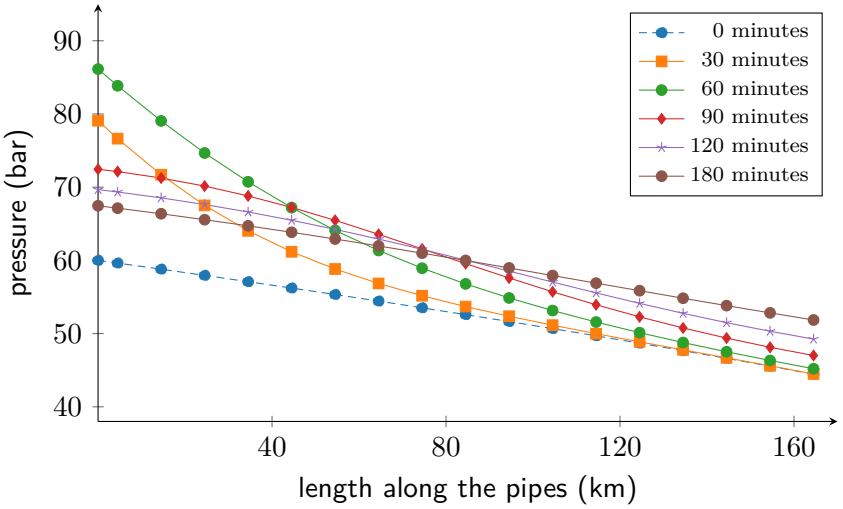
6.1.2.1. *Implementation Issues.* We again reformulate the MINLP (87) using expression trees to obtain a model with nonlinear functions of lower dimensions. For each $a = (v_1, v_2) \in A_{\mathrm{pi}}$ we substitute each pressure loss equation of type (63) by the linear constraint (94) and the corresponding nonlinear constraints $f_{v_1} = p_{v_1}^2$, $f_{v_2} = p_{v_2}^2$, and $f_a^{\mathrm{pi}} = |q_a| q_a$ as in Subsection 6.1.1.1.

As in the stationary case, we only refine a certain amount of all nonlinear functions that are worst regarding the score M^i. Here we select $\theta = 0.85$. We again only solve every 50th MIP relaxation to global optimality and proceed as in the stationary case regarding the relative MIP gaps.

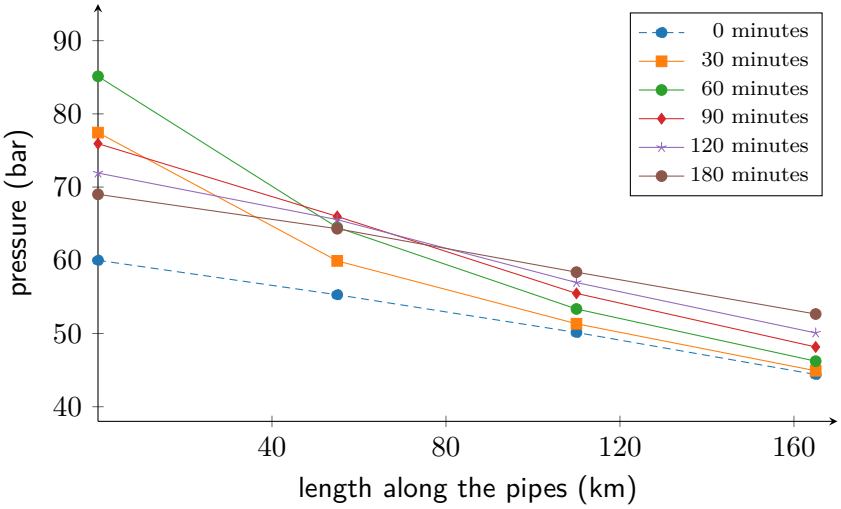
Concerning the error bounds in Algorithm 1, we use 50.0 bar for the pressure loss equations in the initial MIP relaxation. The final approximation errors are set to 2.0 bar.

As in the stationary case, if a feasible solution of an MIP relaxation is found, we fix all discrete variables of the underlying MINLP problem according to the MIP solution and solve the resulting NLP problem to local optimality. The NLPs are again solved using CONOPT3.

We highlight two algorithmic extensions to the stationary case. First, whenever a feasible solution of the MINLP is obtained, we transform it into a feasible starting solution of the MIP relaxation. This can sometimes reduce the run time needed to solve the MIP. Furthermore, any objective value of a solution of an MIP relaxation provides a dual bound for the MINLP problem. At the same time, the dual bounds that we obtain while solving an MIP relaxation are also dual bounds for the MINLP. We exploit this by solving a very fine MIP relaxation parallel to our main algorithm. Although the MIP solver is unlikely to solve the fine MIP relaxation within reasonable



(A) Values for the fine discretization

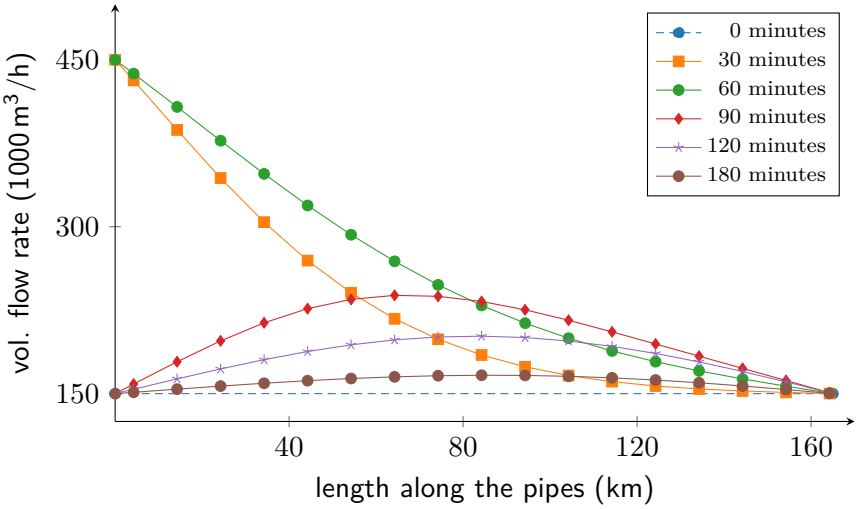


(B) Values for the coarse discretization

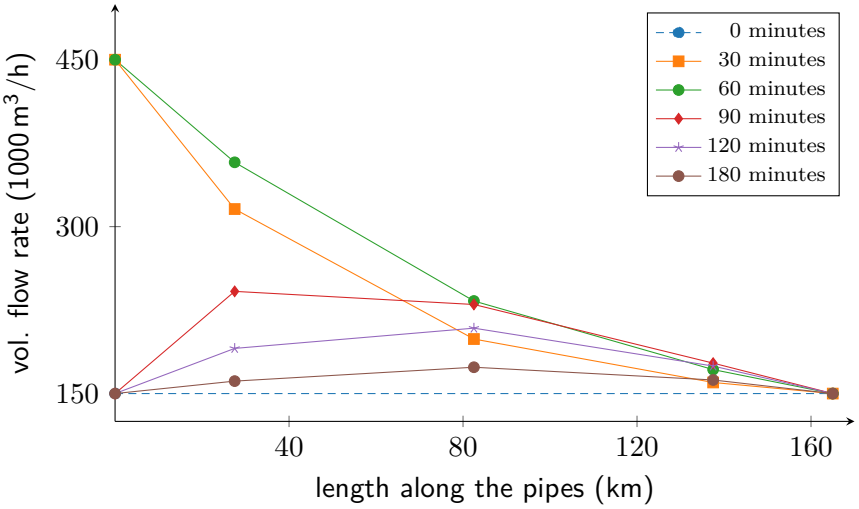
FIGURE 6.5. Pressure values (y-axis) for a fine (above) and coarse (below) discretization in three consecutive pipes of the GasLib-11 network over their accumulated length (x-axis) for different time points.

run time limits, if a tighter dual bound is found, we use it as dual bound for the MINLP.

6.1.2.2. *Pressure and Flow Waves within Pipes.* As a first case study, we investigate the propagation of pressure and flow waves within pipes. For this purpose, we consider the three consecutive pipes S2–N3–N4–N2 from the GasLib-11 network, which add up to a total length of 165 km. Initially, we assume a stationary state, where 150 $m^3\,h^{-1}$ are injected in S2 and discharged at N2. We immediately increase the supply at S2



(A) Values for the fine discretization



(B) Values for the coarse discretization

FIGURE 6.6. Flow values (y-axis) for a fine (above) and coarse (below) discretization in three consecutive pipes of the GasLib-11 network over their accumulated length (x-axis) for different time points.

to $450\,\mathrm{m}^3\,\mathrm{h}^{-1}$ and inject this amount up to minute 60. From minute 60 on, $150\,\mathrm{m}^3\,\mathrm{h}^{-1}$ are injected again. We choose a time discretization of 5 s and a spatial discretization of 500 m. Please note that in the following all flow values are given as volumetric flow values ($\mathrm{m}^3\,\mathrm{h}^{-1}$) instead of mass flow values as in Chapter 4. The volumetric flow Q and the mass flow q are related by $q = \rho_0 Q$, where $\rho_0 \approx 0.8304\,\mathrm{kg\,m}^{-3}$ is the density under normal conditions.

Figures 6.5 and 6.6 show that our time-expanded graph model is capable of

detecting parabolic pressure and flow waves and their propagation within pipes. Furthermore, we observe that these wave effects are quickly smoothed and tend towards a stationary state. This behavior is typical for parabolic PDEs.

We now give some insight into the discretization that we use in the subsequent subsection. Therein, we consider a time discretization of 10 min and a two-point space discretization of 55 km for the pipes. This coarse discretization is due to the fact that non-convex MINLPs as in (87) are hard to solve in general. Hence, we are forced to use a coarse discretization in order to keep the MINLP computationally tractable.

We run the same simulation as before, albeit with the coarse discretization. Now, we depict the result for the coarse discretization at different time points in Figures 6.5b and 6.6b. Please note that flow values between two discretization points in space are given by a piecewise constant function; see Section 4.3.2. Hence, the respective flow values in Figure 6.6b are marked at the midpoints between two consecutive space discretization points. We therefore obtain points at lengths of 27.5 km, 82.5 km, and 137.5 km. Pressure values, however, are given for single discretization points in space, leading to points at lengths of 0 km, 55 km, 110 km, and 165 km in Figure 6.5b. Comparing both discretizations, we can observe that the characteristic behavior of the fine discretization is maintained by the coarser one. In addition, the difference between the two discretizations vanishes with large time horizons. With the goal of global optimization, we are therefore confident to use the coarse discretization on time horizons of several hours.

TABLE 6.3. The prescribed nomination q^{nom} (given in $\mathrm{m}^3\,\mathrm{h}^{-1}$) for one time step and all entries and exits of the GasLib-11 network.

S1	S2	S3	T1	T2	T3
140.00	160.00	0.00	90.00	150.00	60.00

6.1.2.3. *Storage Capacity Maximization.* As a second case study, we solve the storage capacity maximization problem (87) using the GasLib-11 network shown in Figure 6.4. We choose the parameters $\gamma_1 = 0.0015$, $\gamma_2 = 0.02$, $S_a = 7200$ for all $a \in A_{\mathrm{cm}}$, and $S_a = 3600$ for all $a \in A_{\mathrm{vl}}$, which are introduced in Section 4.4. For the prescribed nomination $q^{\mathrm{nom}} \in \mathbb{R}^{N|V_s \cup V_t|}$, we use the values that are given in Table 6.3 for all time steps.

Both compressors Cm1 and Cm2 run in bypass mode at the beginning while the valve Vl1 is closed. This results in a tree-shaped network. Thus, we can compute an initial stationary solution by fixing the pressure for S1 to 58 bar and propagate the flow through the network. The resulting initial pressure values for all eleven vertices are given in Table 6.4.

TABLE 6.4. Initial pressure values (given in bar) for all eleven vertices of the GasLib-11 network as in Figure 6.4.

S1	S2	S3	N1	N2	N3	N4	N5	T1	T2	T3
58.00	59.94	53.77	53.77	49.18	54.55	48.56	48.56	47.15	42.60	47.66

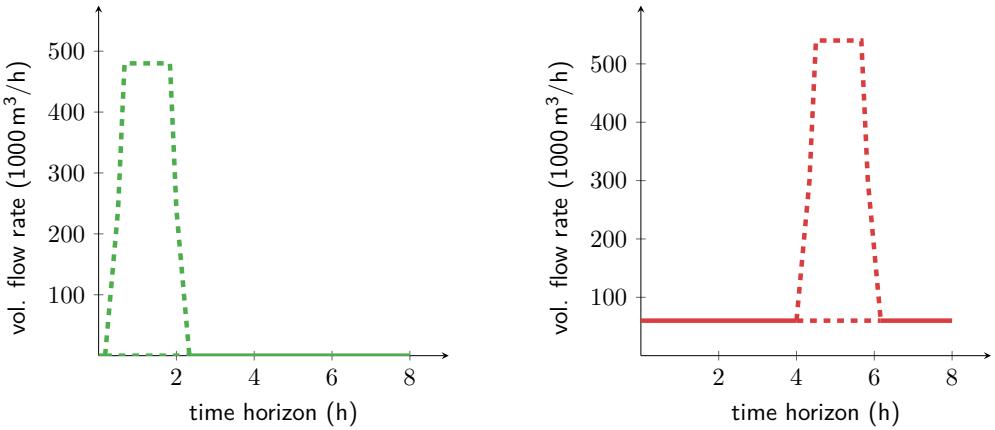


FIGURE 6.7. Given volumetric flow rate profiles for the additional amount of gas that can be injected at S3 (left) and must be discharged accordingly at T3 (right). Based on q^{extra} in (87), the dashed lines indicate the additional amount of gas that can be injected at S3 (green lines) and must be discharged accordingly at T3 (red lines) with corresponding upper and lower bounds.

We now consider a time horizon of 8 h, with a time discretization of 10 min leading to a total amount of 48 time steps and a two-point space discretization for the pipes. From minute 20 on, in addition to the nomination, a positive amount of gas can be injected for two hours at S3. We allow a maximal additional amount that corresponds to $500\,\mathrm{m}^3\,\mathrm{h}^{-1}$. Moreover, we stipulate a linear increase (and decrease) in the additionally injectable gas amount up to (and from) the maximum within 20 min. From the fourth hour onwards, the same additional amount of gas must be discharged at T3 within two hours, whereby the same conditions apply as in the case of the additional supply. See Figure 6.7 for an illustration.

The resulting MINLP problem consists of 2311 variables, of which 2164 are continuous and 147 are binary, and 2785 constraints, of which 2393 are linear and 392 are nonlinear.

After a total run time limit of 4 h, our MIP-based approach delivers a feasible solution for the storage capacity maximization problem (87) that is globally optimal within a relative gap of almost 5 %. The corresponding profiles of the solution are

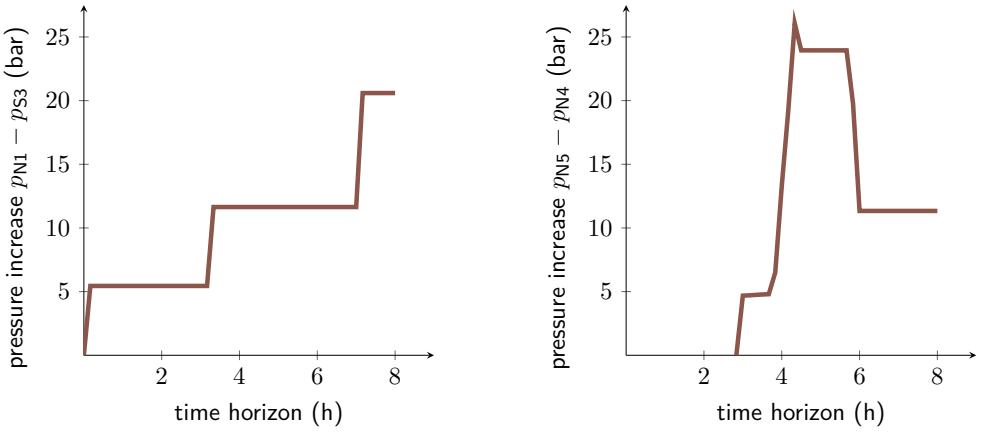


FIGURE 6.8. Profile for the pressure increase of compressor Cm1 (left) and compressor Cm2 (right) corresponding to the best solution for the storage capacity maximization problem (87) found after a total run time of 4 h.

shown in Figure 6.8 for the compressors Cm1 and Cm2, in Figure 6.9 for the pressures of all eleven vertices, in Figure 6.10 for the valve Vl1, and in Figure 6.11 for the additional amount of gas q^{extra}.

The compressor Cm1 is immediately switched on and operates throughout the whole time horizon. From minute 50 onwards, almost as much additional Gas q^{extra} is injected at S3 as possible. As a consequence, all pressure values rise with a higher amount of gas until no additional gas is injected in S3 anymore. Moreover, the valve Vl1 is opened, with approximately half of q^{extra} passing through it. About an hour before the additionally injected amount of gas is discharged at T3, the Compressor Cm2 is also switched on. At the same time, a small amount of gas passes through the valve again before it is closed. Due to the coincident compression of both compressors, the pressure at T3 remains within the pressure bounds while discharging the additional amount of gas.

Returning to the objective of maximizing the storage capacity, around 74.17 % of the possible additional amount of gas in q^{extra} is attainable according to our solution; see Figure 6.11. Due to the chosen parameters γ_1 and γ_2, the cost of compression is almost negligible in our MINLP problem. Furthermore, our solution is globally optimal within a gap of less than 6 %. Hence, we conclude that taking into account the model from above, approximately 78.62 % of the possible additional amount of gas in q^{extra} can be injected at S3.

Finally, we show an iteration log of Lamatto++ for the storage capacity maximization problem (87) in Table 6.5. After a total run time of less than 3 h, Lamatto++

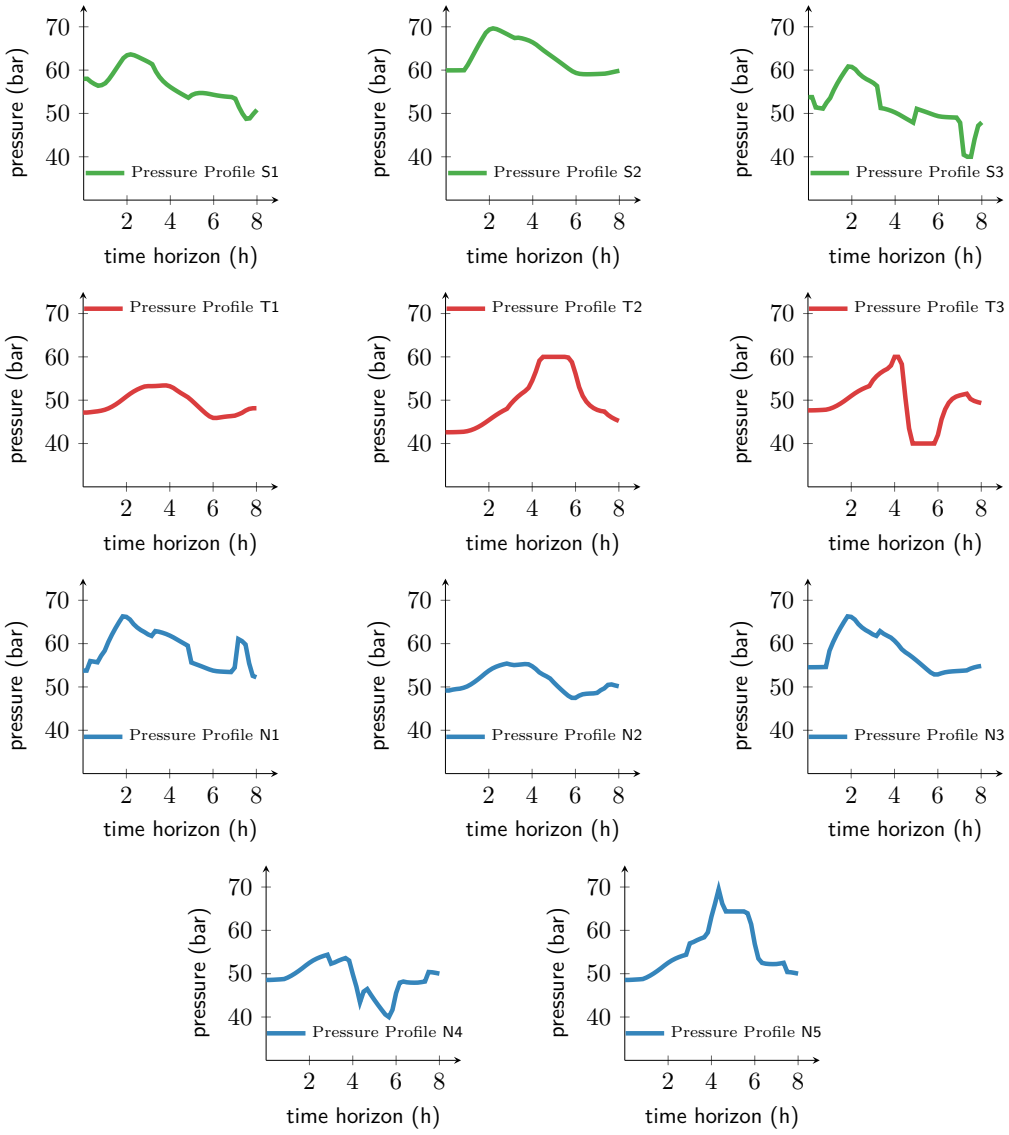


FIGURE 6.9. Profiles for the pressures of all eleven vertices of the GasLib-11 network in Figure 6.4. The values correspond to the best solution for the storage capacity maximization problem (87) found after a total run time of 4 h.

is able to find a solution that is feasible for the storage capacity maximization problem (87) and globally optimal within a relative gap of almost 5 %.

The first column in Table 6.5 indicates the corresponding iteration in Algorithm 1. The best dual bound d of the MINLP and the objective value z of the incumbent (feasible) solution are given in column two and three, respectively. The next column contains the relative gap $|(z - d)/z|$, which is given in percent. The last column

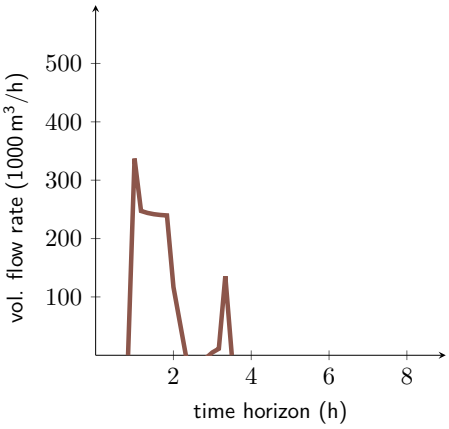


FIGURE 6.10. Volumetric flow rate profile for the valve VI1 showing the amount of gas passing through the valve in case that it is open. The values correspond to the best solution for the storage capacity maximization problem (87) found after a total run time of 4 h.

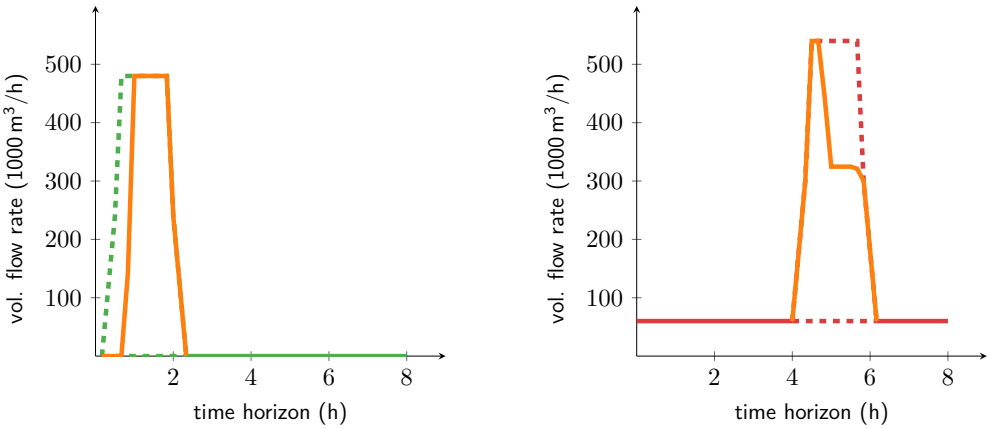


FIGURE 6.11. Volumetric flow rate profiles for the entry S1 (left) and the exit T2 (right) showing the additional amount of gas q^{extra} (orange) that is injected (S1) and discharged (T3). The values correspond to the best solution for the storage capacity maximization problem (87) found after a total run time of 4 h.

presents the run time in seconds that Lamatto++ spent until the current iteration. We observe that Lamatto++ produces feasible solutions with small relative gaps even in a short run time.

For the sake of comparison, the state-of-the-art global MINLP solvers Baron and SCIP solve the same MINLP on the same cluster using all 12 cores and a run time limit of 4 h. After the time limit has been reached, the best feasible solution Baron finds has an objective value of 332.991, while the best dual bound is 630.965. This

TABLE 6.5. Iteration log of Lamatto++ for the storage capacity maximization problem (87) using the gas network in Figure 6.4 with a total run time limit of 4 h.

iteration	dual	primal	gap	elapsed time
1	498.342	449.080	10.97 %	9.64
6	498.342	460.562	8.20 %	38.36
201	497.297	460.562	7.98 %	1726.50
218	494.919	460.562	7.46 %	1949.03
236	491.733	460.562	6.77 %	2216.00
237	490.160	460.562	6.43 %	5517.05
238	490.160	464.529	5.52 %	10 656.86
239	490.160	464.529	5.52 %	14 103.06

translates into a relative gap of about 89 %. The best feasible solution SCIP finds has an objective value of 296.724, where the best dual bound is 622.838. This corresponds to a relative gap of about 109 %. Our approach thus delivers significantly better results in this case. Sometimes, in the case of application-related problems, a good feasible solution that is obtained in a short time is also useful. To this end, we solve the same MINLP with the local MINLP solvers α-ECP, BONMIN, and Knitro. Within a run time of 30 min, BONMIN and Knitro are not able to find any feasible solution. α-ECP, however, is able to find a feasible solution with an objective value of 464.885, within only 2 min. It is slightly better than our solution found after 38.36 s and comparable to our best feasible solution. α-ECP is a local MINLP solver and thus unable to provide a dual bound. Moreover, it can only confirm the feasibility of the solution, but no local optimality. With regard to global optimality, our approach is therefore preferable, as it provides feasible solutions of high quality and tight dual bounds.

In conclusion, we see that our time-expanded graph method can be successfully applied in the context of the storage capacity maximization of gas networks. In addition, the approach delivers solutions within reasonable run time that are both physically plausible and near-global optimal.

6.2. AC Optimal Power Flow with Generator Switching

In this section, we present computational results for the two standard OPF test cases Case22Loop by Bukhsh et al. (2013) and Case39 by Zimmerman et al. (2011). As in the previous subsections, we first provide an insight into the implementation issues in the context of AC OPF with generator switching. Afterward, we present numerical results for slightly modified versions of the test cases Case22Loop and Case39, followed by a short comparison with the global MINLP solver Couenne 0.5.6. The medium-scale Case22Loop power network consists of 22 buses of which 11 buses are generator units,

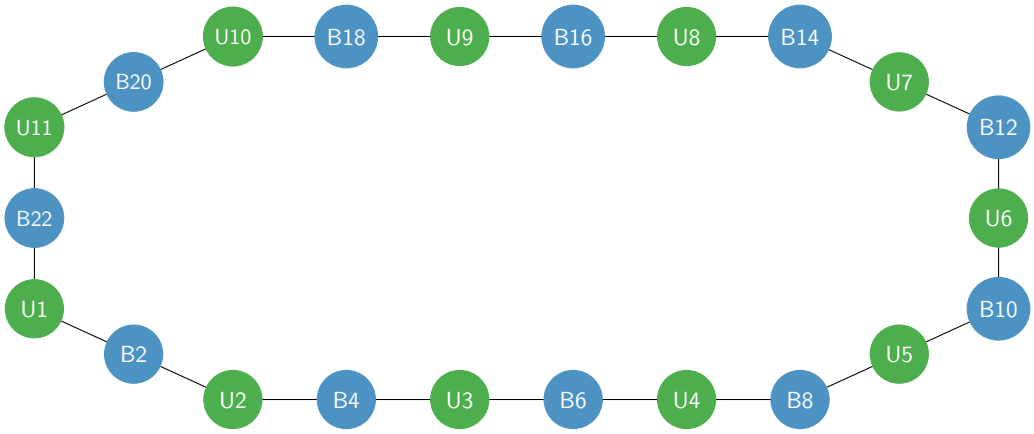


FIGURE 6.12. The Case22Loop power network.

and 22 transmission lines; see Figure 6.12 for an illustration. The medium-scale Case39 power network consists of 39 buses of which 10 buses are generator units, and 46 transmission lines; see Figure 6.13 for an illustration.

The computations are carried out on a laptop using two cores of a dual-core i7-7600U CPU at 2.8 GHz with 4 MB Shared Cache and 16 GB of RAM. Furthermore, we utilize Gurobi 8.0.1 as MIP solver, using the two cores mentioned above for the parallel solving of each MIP. Note that for a fair comparison, Couenne also has two cores available and runs in parallel.

6.2.1. Implementation Issues. As in the previous MINLP problems, we reduce the dimension of the nonlinear functions of the MINLP (92) using expression trees. For each transmission line $e = (k, l) \in L$, we add variables $f_{e_1}, f_{e_2}, \ldots, f_{e_7}$ and the nonlinear constraints

$$f_{e_1} = p_{kl}^2, \qquad f_{e_2} = q_{kl}^2, \qquad f_{e_3} = c_{kl}^2, \qquad f_{e_4} = t_{kl}^2, \qquad f_{e_5} = c_{kk}c_{ll},$$
$$f_{e_6} = \tan(\theta_l - \theta_k), \qquad f_{e_7} = c_{kl}f_{e_6}.$$

Herewith, we reformulate the equations (92h) by the linear constraints

$$f_{e_1} + f_{e_2} \leq (d_{kl}^+)^2.$$

Similarly, we reformulate (92i) with

$$f_{e_3} + f_{e_4} = f_{e_5}.$$

As the last reformulation, we replace (93) by the linear constraint

$$f_{e_7} = t_{kl}.$$

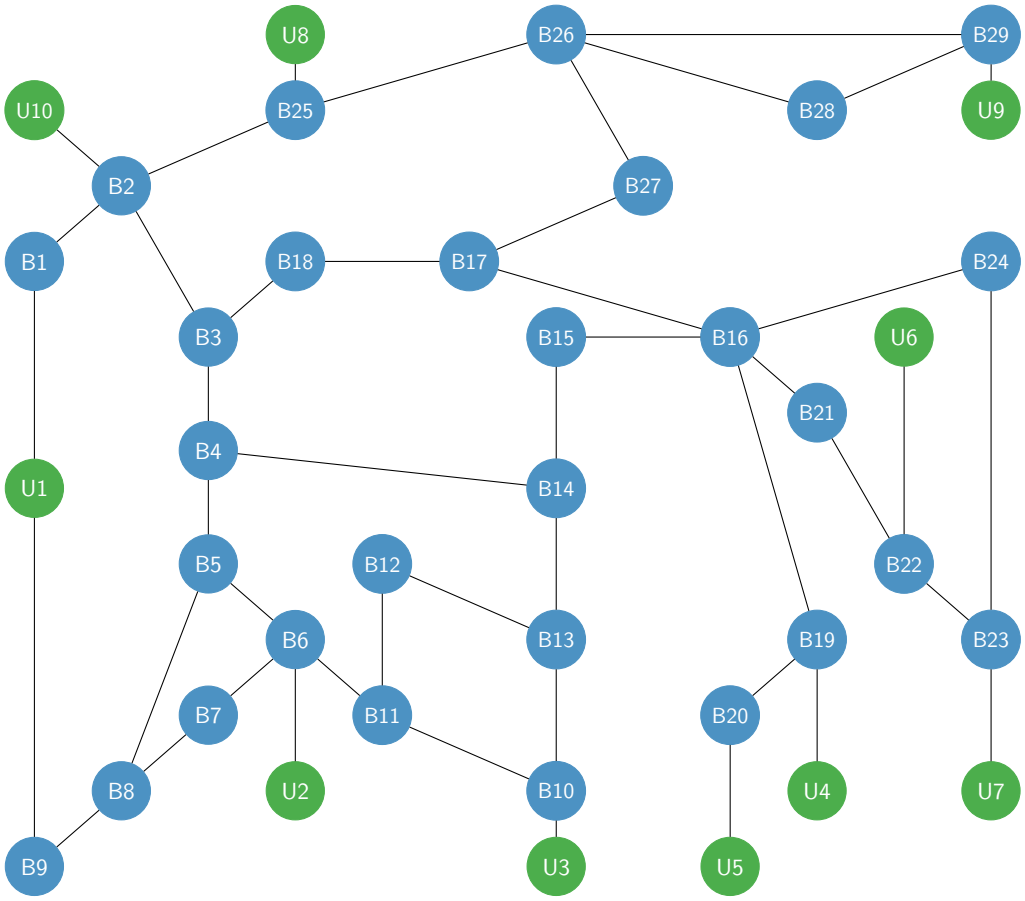


FIGURE 6.13. The Case39 power network.

For quadratic objective functions, we introduce a variable f_{C_k} substituting each square $(p_k^g)^2$ of the objective function and add the nonlinear constraints $f_{C_k} = (p_k^g)^2$ to the model.

As in the stationary case of the gas network optimization, we only refine a certain amount of all nonlinear functions that are worst regarding the score M^i and chose again $\theta = 0.75$. We solve only every 20th MIP relaxation to global optimality and proceed as before regarding the relative MIP gaps, where we use half of the incumbent relative gap of the MINLP as relative gap for the MIP. If no gap is available for the MINLP, we set 10 % as MIP gap.

As in previous computations, if a feasible solution of an MIP relaxation is found, we fix all discrete variables of the underlying MINLP problem according to the MIP solution and solve the resulting NLP problem to local optimality. The NLPs are now solved using IPOPT within GAMS 25.1.2. Whenever a feasible solution of the MINLP is obtained, we transform it into a feasible starting solution of the MIP relaxation. In

addition, we again compute tight dual bounds of the MINLP by solving a fine MIP relaxation and using its dual bounds.

Concerning the error bounds in Algorithm 1, we now simply use two equidistant interpolation points in case of one-dimensional nonlinear functions for the initial MIP relaxation. In case of a two-dimensional nonlinear function, we use the four extreme points of the function's domain. The final approximation errors are set to zero, since we are now mainly interested in the relative optimality gaps that we are able to obtain by Algorithm 1 combined with IPOPT as NLP solver.

Finally, we tighten the bounds of all variables in a preprocessing step. To this end, we solve for each variable two optimization problems minimizing and maximizing the variable, while we consider the LP relaxation of the initial MIP relaxation as the feasible set. We then set the bounds of the variables to the obtained optimal objective values. This bound tightening method is essentially the same as an optimality-based bound tightening; see again Section 2.3 for more details. Note that due to this preprocessing, in the case of Case22Loop and Case39, we obtain bounds for the variables θ_l, θ_k such that $-\pi/2 < \theta_l - \theta_k < \pi/2$ always holds. Hence, we can use (93) instead of (92j) in the MINLP model (92).

6.2.2. Generator Production Cost Minimization. We now solve the MINLP problem (92), which minimizes the production costs of the generators. The MINLP instances are based on the Case22Loop and Case39 power network.

6.2.2.1. *The Case22Loop Power Network.* We begin with the instance that is based on the Case22Loop power network. The 22 transmission lines and the generators of the Case22Loop network are identical and the objective function is linear with the same coefficients. The original lower bounds of the generator production are zero for all generators. With these bounds, all generators are switched on in the optimal solution, while each generator produces 206.31 MW. In real-world applications, however, the minimal amount of power production is usually greater than zero. Hence, we set all lower bounds to 230 MW, which is roughly an increase of 206.31 MW by 10 %. This prevents all generators from being switched on and thus brings combinatorics into play. The upper bounds for the production of the generator are all 10 000 MW. Furthermore, the buses $\mathsf{B2}, \mathsf{B4}, \ldots, \mathsf{B22}$ have all a demand of 204.25 MW.

The resulting MINLP problem consists of 143 variables, of which 132 are continuous and 11 are binary, and 176 constraints, of which 110 are linear and 66 are nonlinear.

After a total run time limit of 4 h, Algorithm 1 in combination with IPOPT delivers a feasible solution that is globally optimal within a relative gap of almost 0.4 %. From a practical point of view, we can regard this solution as essentially globally optimal.

TABLE 6.6. Production of the generator units of the Case22Loop power network (given in MW) corresponding to the best feasible solution found after a total run time of 4 h.

U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11
0	287.95	230	230	287.95	0	276.26	230	230	230	276.26

The production of the generator units corresponding to this solution is depicted in Table 6.6. As expected, only a subset of all generators are switched on.

TABLE 6.7. Iteration log of Algorithm 1 for the AC OPF problem with additional generator switching (92). The model is based on the power network of Case22Loop, while the total run time limit is 4 h.

iteration	dual	primal	gap	elapsed time
1	4426.000	-	-	210.15
2	4426.000	4564.020	3.024 %	222.67
6	4426.000	4556.839	2.871 %	235.98
21	4467.917	4556.839	1.951 %	358.94
41	4488.315	4556.839	1.504 %	485.09
61	4536.082	4556.839	0.456 %	698.56
921	4536.147	4556.839	0.454 %	9245.92
941	4536.518	4556.839	0.446 %	9573.24
1061	4536.803	4556.839	0.440 %	14 306.65
1071	4536.803	4556.839	0.440 %	14 397.98

Finally, we show an iteration log of Algorithm 1 in Table 6.7. The first column in Table 6.7 indicates the iteration in Algorithm 1. In column two and three, the best dual bound d of the MINLP and the objective value z of the incumbent (feasible) solution are given. The next column contains the relative gap $|(z-d)/z|$ in percent. The last column shows the run time in seconds that Algorithm 1 combined with IPOPT spent until the corresponding iteration. After a total run time of less than 4 min, our approach is able to find a solution that is feasible for the MINLP and globally optimal within a relative gap of almost 0.4 %. We see that even with shorter run time, our approach is able to find feasible solutions with small relative gaps.

We have the same MINLP solved by Couenne with the same run time limit of 4 h. We compare our approach with Couenne based on the iteration log in Table 6.8, which reads in the same manner as Table 6.7, except for the iteration numbers. Both in terms of the run time that is needed to find feasible solutions, as well as in terms of the quality of the solutions, Couenne and our method perform very similarly. However, our approach delivers tighter dual bounds and is therefore able to prove the optimality

TABLE 6.8. Iteration log of Couenne for the AC OPF problem with additional generator switching (92). The model is based on the power network of Case22Loop, while the total run time limit is 4 h.

dual	primal	gap	elapsed time
4222.544	4740.556	10.927 %	0.00
4429.462	4740.556	6.562 %	0.81
4429.462	4672.227	5.196 %	3.69
4429.462	4590.228	3.502 %	4.60
4429.462	4564.129	2.951 %	5.66
4438.267	4564.129	2.758 %	7.53
4441.981	4559.075	2.568 %	31.86
4448.090	4557.586	2.403 %	57.35
4449.893	4556.839	2.347 %	82.69
4452.677	4556.839	2.286 %	3600.78
4454.209	4556.839	2.252 %	7200.35
4456.600	4556.839	2.200 %	14 399.79

of the solution within a relative gap of almost 0.4 %. After a run time of 4 h, Couenne is not able to lower the relative gap any further than 2.2 %. We conclude that in case of the MINLP (92), based on the Case22Loop power network, Algorithm 1 combined with IPOPT is preferable to Couenne.

TABLE 6.9. Upper bounds for the production of the generator units of the Case39 power network (given in MW).

U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
2200	1292	1450	1304	1016	1374	1160	1128	1730	2080

TABLE 6.10. Demands for all buses of the Case39 power network (given in MW) including the generator units.

B1	B3	B4	B7	B8	B9	B12	B15	B16	B18	B20
97.60	322	500	233.80	522	6.50	8.53	320	329	158	680

B21	B23	B24	B25	B26	B27	B28	B29	U1	U2
274	247.50	308.60	224	139	281	206	283.50	1104	9.20

6.2.2.2. *The Case39 Power Network.* The second test instance is based on the Case39 power network. Now, the 46 transmission lines and the 10 generator units have different parameters. In addition, the objective function is quadratic with equal coefficients. We pursue the same strategy for the lower bounds of the generator

production as in the case of Case22Loop. Since the original bounds are again zero, all generators are switched on in the optimal solution. Here, each generator produces a different amount with an average production of 630.175 MW. We set the lower bounds to 700 MW, which as before approximately corresponds to an increase of 630.175 MW by 10 %. Moreover, we double the upper bounds for the generator production to prevent infeasibility; see Table 6.9 for the corresponding values. Table 6.10 contains all buses of the network that have a demand and the corresponding values.

The resulting MINLP problem consists of 515 variables, of which 505 are continuous and 10 are binary, and 630 constraints, of which 354 are linear and 276 are nonlinear.

TABLE 6.11. Production of the generator units of the Case39 power network (given in MW) corresponding to the best feasible solution found after a total run time of 4 h.

U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
807.37	792.14	786.47	0	782.85	781.07	774.70	0	784.93	792.13

After a total run time limit of 4 h, Algorithm 1 in combination with IPOPT delivers a feasible solution that is globally optimal within a relative gap of almost 10 %. We depict the production of the generator units corresponding to this solution in Table 6.11. Again, only a subset of all generators are switched on.

TABLE 6.12. Iteration log of Algorithm 1 for the AC OPF problem with additional generator switching (92). The model is based on the power network of Case39, while the total run time limit is 4 h.

iteration	dual	primal	gap	elapsed time
1	44 189.366	51 703.889	14.534 %	818.69
5	44 212.102	51 570.032	14.268 %	887.74
13	44 334.092	51 537.445	13.977 %	1091.17
14	45 990.397	51 537.445	10.763 %	1130.00
68	45 990.397	51 537.445	10.763 %	13 927.53

As for the previous instance, we show an iteration log of Algorithm 1 in Table 6.12. After a total run time of less than 20 min, our approach is able to find a solution that is feasible for the MINLP and globally optimal within a relative gap of almost 10 %. In conclusion, we see that our approach is able to find near-global feasible solutions even with shorter run time.

We again have the same MINLP solved by Couenne with the same run time limit of 4 h and compare our approach with Couenne based on the iteration log in Table 6.13.

TABLE 6.13. Iteration log of Couenne for the AC OPF problem with additional generator switching (92). The model is based on the power network of Case39, while the total run time limit is 4 h.

dual	primal	gap	elapsed time
38 390.162	-	-	0.00
38 390.162	58 711.450	34.612 %	47.89
42 502.945	58 711.450	27.607 %	60.29
45 991.800	58 711.450	21.665 %	88.44
45 991.800	58 711.450	21.665 %	14 397.75

In contrast to the instance based on Case22Loop, Couenne and our approach perform very similarly regarding the dual bounds. However, Algorithm 1 combined with IPOPT is able to find feasible solutions that are significantly better than the one obtained by Couenne. The best solution found by Couenne is globally optimal within a relative gap of more than 21 %, while our approach delivers a feasible solution with a relative gap of almost 10 %. We conclude that in case of the MINLP (92), based on the Case39 network, Algorithm 1 in combination with IPOPT again is superior to Couenne.

6.3. Final Remarks

In this chapter, we demonstrated the applicability of Algorithm 1 to real-world MINLP problems. To this end, we solved instances from the field of gas network transport optimization and optimal power flow. For these instances, our current implementation of Algorithm 1 is already superior to state-of-the-art global MINLP solvers such as Baron, SCIP, and Couenne. However, there is still plenty of room for improvements.

First, the preprocessing so far is only integrated into our implementation in a very basic version. As pointed out in Chapter 2, preprocessing is a vital aspect of MINLP (and MIP) solvers. A more sophisticated preprocessing, e.g., a feasibility-based bound tightening in each iteration, will therefore significantly reduce the size of the MIP relaxations in our algorithm, since they mainly depend on the bounds of the variables.

Nevertheless, the most promising idea is to refine the nonlinearities such that we can exploit warm-starting procedures implemented in modern MIP solvers. The main goal is to embed Algorithm 1 into a single-tree framework. As mentioned in Subsection 6.1.1.2, the projections of the solutions of consecutive MIP relaxations are often locally close to each other. A first step towards a single-tree approach, would be to use the solution of an MIP relaxation to obtain a starting solution for the subsequent MIP relaxation. With the longest-edge bisection, for each function the triangulations of two consecutive MIP relaxations differ only in at most one simplex, if a refinement is performed. Otherwise, the triangulations are identical. Thus, we

can simply fix all binary and continuous variables according to the solution of the last MIP relaxation, except for all simplices that are refined. This results in an MIP of comparatively small size. Based on the observation that solutions of consecutive MIP relaxations are often locally close to each other, we believe that the optimal solution of this small-sized MIP is in many cases a good starting solution for the subsequent MIP relaxation. With good starting solutions, however, the run time of an MIP relaxation can be drastically reduced.

Furthermore, no problem-specific heuristics are incorporated yet. One possibility is to use an NLP relaxation of the MINLP, where we reformulate the binary variables as complementary constraints and obtain a feasible solution by rounding; see for instance the works by Baumrucker and L. T. Biegler (2010) and Schewe and Schmidt (2018).

Finally, we remark that although a primal algorithm, which in our case is constituted by CONOPT3 and IPOPT, is necessary to compute valid upper bounds, it is not necessary for Algorithm 1 to terminate. In fact, with tight final error bounds, we can skip the primal algorithm entirely, because in this case Algorithm 1 eventually yields an optimal MIP solution that we can consider as an optimal solution for the corresponding MINLP.

CHAPTER 7

Summary and Conclusion

In this thesis, we developed a global solution approach for MINLPs that combines adaptive refinement strategies for the nonlinearities with MIP relaxations for the integer part of the MINLP.

We motivated our adaptive method, which is largely based on solving MIPs, in Chapter 2 by an experiment, in which we reformulated MIPs as NLPs by replacing all binary variables using the constraint $x^2 - x = 0$. These NLPs are subsequently solved by state-of-the-art global NLP solvers within SCIP and Baron. The numerical results of the experiment demonstrated the capability of modern MIP solvers in comparison to modern NLP solvers.

We laid the theoretical groundwork for our adaptive MINLP approach in Chapter 3. We classified the refinement procedures that guarantee the convergence of our adaptive MIP-based approach. In particular, we proved that the longest-edge bisection and the red refinement rule belong to this class of refinement procedures that lead to the convergence of our algorithm. However, a surprising result is that the intuitive refinement strategy, where points with maximal approximation error are added as new vertices cannot always yield approximations with arbitrary accuracy. Moreover, we presented first results on the size of an MIP relaxation that is required to achieve an a priori given accuracy.

We applied our approach to MINLP problems that are difficult to solve by state-of-the-art MINLP solvers. To this end, we solved both stationary and transient gas network optimization problems, while in the transient case we introduced a new model for the gas flow in pipelines. As a second application, we solved optimal power flow problems with additional switching of the generator units. The numerical results in Chapter 6 demonstrate that our MIP-based approach outperforms state-of-the-art global MINLP solvers on several difficult MINLP instances. Furthermore, our method even computes high-quality feasible solutions in a considerable short run time.

As pointed out before, our method strongly relies on MIP technology. The integer part of the MINLP problem is tackled by sophisticated MIP solvers that act as black-box oracles and quickly provide feasible solutions for the MIP relaxations with reliable quality guarantees. Hence, we believe that the higher the integer part of the

MINLP becomes, the more advantageous our approach is compared to other state-of-the-art global MINLP solvers. The numerical results in this thesis also support this notion. The gas network optimization instances contain a notably larger number of integer variables compared to the optimal power flow instances. In the former case, our approach clearly outperforms state-of-the-art MINLP solvers, whereas in the latter case our method is less superior.

In addition, our approach is suitable for a wide range of MINLP problems, since we only require the nonlinear functions of the MINLP to be continuous. In contrast to the solvers Baron and SCIP, for instance, our algorithm is capable of handling trigonometric functions that appear, e.g., in MINLPs arising from optimal power flow.

Building on this thesis, several directions exist for future research. Exemplary numerical results on triangulations in Section 3.4 suggest that the longest-edge and the red refinement rule yield Delaunay triangulations in the sense that the simplices are contained in a Delaunay triangulation of the set of the corresponding vertices. A more general characterization of this observation remains the subject of future research. At this point, we would also like to point out that in general it is still open which triangulations are optimal with respect to the number of sampling points for a given nonlinear function.

On the algorithmic side, there is still potential for improvements. Besides a more sophisticated preprocessing, the most promising course is the further development towards a warm-starting single-tree approach. A first step in this direction would be to use the solution of an MIP relaxation of the MINLP to obtain a starting solution for the subsequent MIP relaxation as described in Section 6.3.

Finally, more empirical studies will give a deeper insight into the MINLP classes for which our method is suitable. The next natural step is to build a global MINLP solver based on our approach and solve various MINLP problems, e.g., instances provided by the MINLPLIB; see Bussieck et al. (2003).

APPENDIX A

IDs of Instances

A.1. IDs of the MIPLIB 2017 Benchmark Problems

TABLE A.1. IDs of all 164 MIPLIB 2017 benchmark problems (infeasible ones are marked by *) that have only binary variables as integer variables and are chosen in order to calculate the corresponding performance profiles in Figure 2.3 and 2.4.

academictimetablesmall	app1-2	assign1-5-8	b1c1s1
bab2	bab6	beasleyC3	binkar10_1
blp-ar98	blp-ic98	bnatt400	bnatt500*
bppc4-08	cbs-cta	chromaticindex1024-7	chromaticindex512-7
cmflsp50-24-8-8	CMS750_4	co-100	cod105
cost266-UUE	csched007	csched008	cvs16r128-89
dano3_3	dano3_5	decomp2	drayage-100-23
drayage-25-23	dws008-01	eil33-2	eilA101-2
exp-1-500-5-5	fast0507	fastxgemm-n2r6s0t2	fhnw-binpack4-4*
fhnw-binpack4-48	glass-sc	glass4	gmu-35-40
gmu-35-50	graph20-20-1rand	h80x6320d	irish-electricity
irp	istanbul-no-cutoff	leo1	leo2
lotsize	mad	map10	map16715-04
markshare2	markshare_4_0	mas74	mas76
mc11	mcsched	milo-v12-6-r2-40-1	momentum1
n2seq36q	n3div36	neos-1122047	neos-1171448
neos-1171737	neos-1445765	neos-2075418-temuka*	neos-2978193-inde
neos-3216931-puriri	neos-3402294-bobin	neos-3402454-bohle*	neos-3555904-turama
neos-3627168-kasai	neos-3754224-navua*	neos-3754480-nidda	neos-3988577-wolgan*
neos-4300652-rahue	neos-4387871-tavua	neos-4413714-turia	neos-4532248-waihi
neos-4647030-tutaki	neos-4763324-toguru	neos-4954672-berkel	neos-5049753-cuanza
neos-5052403-cygnet	neos-5075914-elvire	neos-5093327-huahum	neos-5104907-jarama
neos-5107597-kakapo	neos-5114902-kasavu	neos-5188808-nattai	neos-5195221-niemur
neos-631710	neos-787933	neos-827175	neos-848589
neos-860300	neos-873061	neos-911970	neos-933966
neos-957323	neos-960392	neos17	neos5
net12	netdiversion	nexp-150-20-8-5	ns1116954
ns1208400	ns1644855	ns1760995	ns1830653
nw04	opm2-z10-s4	p200x1188c	peg-solitaire-a3
pg	pg5_34	physiciansched3-3	physiciansched6-2
pk1	qap10	rail01	rail02
rail507	ran14x18-disj-8	rd-rplusc-21	reblock115
rmatr100-p10	rmatr200-p5	rocII-5-11	roi2alpha3n4
roi5alpha10n8	s100	s250r10	satellites2-40
satellites2-60-fs	savsched1	sct2	seymour
seymour1	sing326	sing44	sorrell3
sp150x300d	sp97ar	sp98ar	supportcase10
supportcase18	supportcase22*	supportcase26	supportcase40
swath1	swath3	tbfp-network	thor50dday
toll-like	tr12-30	trento1	uccase12
uccase9	uct-subprob	unitcal_7	var-smallemery-m6j6

A.2. IDs of the GasLib-582 Nominations

TABLE A.2. IDs of all 200 GasLib-582 nominations that are chosen randomly in order to calculate the corresponding performance profiles in Section 6.1.1.3.

nomination_cold_95_									
101	1048	1089	1184	123	1321	1326	1642	1660	1729
1768	1911	2008	21	2120	2121	2158	2164	2187	229
2381	2393	2503	268	2720	3007	3111	33	3375	3539
3769	3970	4054	4127	4194	4209	468	676	82	954
957	971								
nomination_cool_95_									
1012	1032	1119	1130	1143	1153	1181	1291	1423	145
1576	1592	1665	1957	201	2040	2050	2168	2178	2196
2275	2284	2321	2364	2398	2453	25	2674	2682	2721
2770	2774	2794	2854	295	3103	312	3148	316	3201
3283	344	3509	3518	3567	3578	3642	3656	3658	3683
3756	378	379	3791	3858	3885	4072	4223	604	663
81	830	979							
nomination_freezing_95_									
1001	1035	1193	1243	1268	1328	1371	1434	1525	1558
1591	160	1889	1962	2308	2385	2408	2596	2685	2877
2964	3045	3068	3205	3260	3263	3291	3362	3417	3483
3550	3597	3635	3886	3911	3994	400	4008	4091	4211
426	448	454	466	52	56	639	645	741	818
836	881								
nomination_mild_95_									
1433	1455	1479	1661	1750	1922	1942	2052	2081	2450
2502	2568	2581	2612	2668	2817	3239	3309	3524	3541
3781	3795	4078	4178	4184	679	746	865		
nomination_warm_95_									
1095	1316	1385	1691	2051	2247	2314	2718	2824	3020
3579	360	450	637	786					

Bibliography

Achterberg, T., R. E. Bixby, Z. Gu, E. Rothberg, and D. Weninger (2016). *Presolve Reductions in Mixed Integer Programming*. eng. Tech. rep. 16-44. Takustr. 7, 14195 Berlin: ZIB.

Achterberg, T., T. Koch, and A. Martin (2005). "Branching rules revisited". In: *Operations Research Letters* 33.1, pp. 42–54. DOI: 10.1016/j.orl.2004.04.002.

Achterberg, T. and R. Wunderling (2013). "Mixed Integer Programming: Analyzing 12 Years of Progress". In: *Facets of Combinatorial Optimization: Festschrift for Martin Grötschel*. Ed. by M. Jünger and G. Reinelt. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 449–481. DOI: 10.1007/978-3-642-38189-8_18.

Adjiman, C. S. and C. A. Floudas (1996). "Rigorous convex underestimators for general twice-differentiable problems". In: *Journal of Global Optimization* 9.1, pp. 23–40. DOI: 10.1007/BF00121749.

Allgor, R. and P. Barton (1999). "Mixed-integer dynamic optimization I: problem formulation". In: *Computers & Chemical Engineering* 23.4, pp. 567–584. DOI: 10.1016/S0098-1354(98)00294-4.

Androulakis, I. P., C. D. Maranas, and C. A. Floudas (1995). "αBB: A global optimization method for general constrained nonconvex problems". In: *Journal of Global Optimization* 7.4, pp. 337–363. DOI: 10.1007/BF01099647.

Bai, Y., H. Zhong, Q. Xia, and C. Kang (2016). "A Two-Level Approach to AC Optimal Transmission Switching with an Accelerating Technique". In: *IEEE Transactions on Power Systems* 32, pp. 1–1. DOI: 10.1109/TPWRS.2016.2582214.

Baron (2019). *Branch and Reduce Optimization Navigator*. URL: http://archimedes.cheme.cmu.edu/?q=baron (visited on 03/01/2019).

Baumrucker, B. and L. T. Biegler (2010). "MPEC strategies for cost optimization of pipeline operations". In: *Computers & chemical engineering* 34.6, pp. 900–913.

Baumrucker, B. and L. Biegler (2009). "MPEC strategies for optimization of a class of hybrid dynamic systems". In: *Journal of Process Control* 19.8. Special Section on Hybrid Systems: Modeling, Simulation and Optimization, pp. 1248–1256. DOI: 10.1016/j.jprocont.2009.02.006.

α-BB (2019). *Alpha Branch-and-Bound*. URL: http://titan.engr.tamu.edu/tools/ (visited on 03/01/2019).

Belotti, P., J. Lee, L. Liberti, F. Margot, and A. Wächter (2009). "Branching and Bounds Tightening techniques for Non-convex MINLP". In: *Optimization Methods Software* 24.4-5, pp. 597–634. DOI: 10.1080/10556780903087124.

Belotti, P., S. Cafieri, J. Lee, and L. Liberti (2010). "Feasibility-Based Bounds Tightening via Fixed Points". In: *Combinatorial Optimization and Applications*. Ed. by W. Wu and O. Daescu. Springer Berlin Heidelberg, pp. 65–76.

Belotti, P., C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan (2013). "Mixed-integer nonlinear optimization". In: *Acta Numerica* 22, pp. 1–131. DOI: 10.1017/S0962492913000032.

Benders, J. F. (1962). "Partitioning Procedures for Solving Mixed-variables Programming Problems". In: *Numerische Mathematik* 4.1, pp. 238–252. DOI: 10.1007/BF01386316.

Berg, M. d., O. Cheong, M. v. Kreveld, and M. Overmars (2008). *Computational Geometry: Algorithms and Applications*. 3rd ed. Santa Clara, CA, USA: Springer-Verlag TELOS.

Bey, J. (2000). "Simplicial grid refinement: on Freudenthal's algorithm and the optimal number of congruence classes". In: *Numerische Mathematik* 85.1, pp. 1–29. DOI: 10.1007/s002110050475.

Bixby, R. E. (2002). "Solving Real-World Linear Programs: A Decade and More of Progress". In: *Operations Research* 50.1, pp. 3–15. DOI: 10.1287/opre.50.1.3.17780.

— (2012). "A Brief History of Linear and Mixed-Integer Programming Computation". In: pp. 107–121.

Bixby, R. E., M. Fenelon, Z. Gu, E. Rothberg, and R. Wunderling (2004). "18. Mixed-Integer Programming: A Progress Report". In: *The Sharpest Cut*, pp. 309–325. DOI: 10.1137/1.9780898718805.ch18.

Bixby, R. and E. Rothberg (2007). "Progress in computational mixed integer programming—A look back from the other side of the tipping point". In: *Annals of Operations Research* 149.1, pp. 37–41. DOI: 10.1007/s10479-006-0091-y.

Bock, H. G., C. Kirches, A. Meyer, and A. Potschka (2018). "Numerical solution of optimal control problems with explicit and implicit switches". In: *Optimization Methods and Software* 33.3, pp. 450–474. DOI: 10.1080/10556788.2018.1449843.

Bonami, P., L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter (2008). "An algorithmic

framework for convex mixed integer nonlinear programs". In: *Discrete Optimization* 5.2, pp. 186–204. DOI: 10.1016/j.disopt.2006.10.011.

Borgwardt, K. H. (1987). *The Average Number of Pivot Steps*. Berlin, Heidelberg: Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-61578-8_3.

BP Review (2018). *BP Statistical Review of World Energy June 2018*. URL: https://www.bp.com/content/dam/bp/en/corporate/pdf/energy-economics/statistical-review/bp-stats-review-2018-full-report.pdf (visited on 03/01/2019).

Brearley, A. L., G. Mitra, and H. P. Williams (1975). "Analysis of mathematical programming problems prior to applying the simplex algorithm". In: *Mathematical Programming* 8.1, pp. 54–83. DOI: 10.1007/BF01580428.

Buchheim, C., R. Kuhlmann, and C. Meyer (2015). *Combinatorial Optimal Control of Semilinear Elliptic PDEs*. Tech. rep. Optimization Online. URL: http://www.optimization-online.org/DB_HTML/2015/10/5161.html.

Bukhsh, W. A., A. Grothey, K. I. M. McKinnon, and P. A. Trodden (2013). "Local Solutions of the Optimal Power Flow Problem". In: *IEEE Transactions on Power Systems* 28.4, pp. 4780–4788. DOI: 10.1109/TPWRS.2013.2274577.

Burer, S. and A. N. Letchford (2012). "Non-convex mixed-integer nonlinear programming: A survey". In: *Surveys in Operations Research and Management Science* 17.2, pp. 97–106. DOI: 10.1016/j.sorms.2012.08.001.

Burlacu, R., H. Egger, M. Groß, A. Martin, M. E. Pfetsch, L. Schewe, M. Sirvent, and M. Skutella (2018). "Maximizing the storage capacity of gas networks: a global MINLP approach". In: *Optimization and Engineering*, pp. 1–31. DOI: 10.1007/s11081-018-9414-5.

Burlacu, R., B. Geißler, and L. Schewe (2019). "Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes". In: *Optimization Methods and Software*, pp. 1–28. DOI: 10.1080/10556788.2018.1556661.

Bussieck, M. R., A. S. Drud, and A. Meeraus (2003). "MINLPLIB – A Collection of Test Models for Mixed-Integer Nonlinear Programming". In: *INFORMS Journal on Computing* 15.1, pp. 114–119.

Bussieck, M. R. and S. Vigerske (2010). "MINLP Solver Software". In: *Wiley Encyclopedia of Operations Research and Management Science*. Ed. by J. J. Cochran, L. A. Cox, P. Keskinocak, J. P. Kharoufeh, and J. C. Smith. John Wiley & Sons, Inc. DOI: 10.1002/9780470400531.eorms0527.

Byrd, R. H., J. Nocedal, and R. A. Waltz (2006). "Knitro: An Integrated Package for Nonlinear Optimization". In: *Large-scale nonlinear optimization*. Springer, pp. 35–59. DOI: 10.1007/0-387-30065-1_4.

Carpentier, J. (1962). "Contribution a l'etude du dispatching economique". In: *Bulletin de la Societe Francaise des Electriciens* 3.1, pp. 431–447.

Castillo, A., C. Laird, C. A. Silva-Monroy, J. Watson, and R. P. O'Neill (2016). "The Unit Commitment Problem With AC Optimal Power Flow Constraints". In: *IEEE Transactions on Power Systems* 31.6, pp. 4853–4866. DOI: 10.1109/TPWRS.2015.2511010.

Coffrin, C., H. L. Hijazi, and P. Van Hentenryck (2016). "The QC Relaxation: A Theoretical and Computational Study on Optimal Power Flow". In: *IEEE Transactions on Power Systems* 31.4, pp. 3008–3018. DOI: 10.1109/TPWRS.2015.2463111.

Cook, W., T. Koch, D. E. Steffy, and K. Wolter (2011). "An Exact Rational Mixed-Integer Programming Solver". In: *Integer Programming and Combinatoral Optimization* 6655, pp. 104–116. DOI: 10.1007/978-3-642-20807-2_9.

Correa-Posada, C. M. and P. Sánchez-Martín (2014). "Gas Network Optimization: A comparison of Piecewise Linear Models". URL: http://www.optimization-online.org/DB_HTML/2014/10/4580.html.

Couenne (2019). *Convex Over and Under ENvelopes for Nonlinear Estimation*. URL: https://projects.coin-or.org/Couenne (visited on 03/01/2019).

Cplex (2019). *IBM ILOG CPLEX Optimization Studio*. URL: https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/ (visited on 03/01/2019).

Crama, Y. (1989). "Recognition problems for special classes of polynomials in 0–1 variables". In: *Mathematical Programming* 44.1, pp. 139–155. DOI: 10.1007/BF01587085.

Crowder, H., E. L. Johnson, and M. Padberg (1983). "Solving Large-Scale Zero-One Linear Programming Problems". In: *Oper. Res.* 31.5, pp. 803–834. DOI: 10.1287/opre.31.5.803.

Dakin, R. J. (1965). "A tree-search algorithm for mixed integer programming problems". In: *The Computer Journal* 8.3, pp. 250–255. DOI: 10.1093/comjnl/8.3.250.

Dantzig, G. B., R. Fulkerson, and S. Johnson (1954). "Solution of a Large-Scale Traveling-Salesman Problem". In: *Journal of the Operations Research Society of America* 2.4, pp. 393–410. DOI: 10.1287/opre.2.4.393.

Dantzig, G. B. (1963). *Linear programming and extensions*. Princeton Univ. Press.

Dantzig, G. B. and P. Wolfe (1960). "Decomposition Principle for Linear Programs". In: *Operations Research* 8.1, pp. 101–111. DOI: 10.1287/opre.8.1.101.

Delaunay, B. N. (1934). "Sur la sphère vide". In: *Bulletin of Academy of Sciences of the USSR* 7, pp. 793–800.

Dey, S. S. and A. Gupte (2015). "Analysis of MILP Techniques for the Pooling Problem". In: *Operations Research* 63.2, pp. 412–427. DOI: 10.1287/opre.2015.1357.

Dolan, E. D. and J. J. Moré (2002). "Benchmarking optimization software with performance profiles". In: *Mathematical Programming* 91.2, pp. 201–213. DOI: 10.1007/s101070100263.

Dommel, H. W. and W. F. Tinney (1968). "Optimal Power Flow Solutions". In: *IEEE Transactions on Power Apparatus and Systems* PAS-87.10, pp. 1866–1876. DOI: 10.1109/TPAS.1968.292150.

Domschke, P., B. Geißler, O. Kolb, J. Lang, A. Martin, and A. Morsi (2011). "Combination of Nonlinear and Linear Optimization of Transient Gas Networks". In: *INFORMS Journal on Computing* 23.4, pp. 605–617. DOI: 10.1287/ijoc.1100.0429.

Domschke, P., B. Hiller, J. Lang, and C. Tischendorf (2017). *Modellierung von Gasnetzwerken: Eine Übersicht*. Tech. rep. URL: https://opus4.kobv.de/opus4-trr154/frontdoor/index/index/docId/191 (visited on 03/01/2019).

Dörfler, W. (1996). "A Convergent Adaptive Algorithm for Poisson's Equation". In: *SIAM Journal on Numerical Analysis* 33.3, pp. 1106–1124. DOI: 10.1137/0733054.

Duran, M. A. and I. E. Grossmann (1986). "An outer-approximation algorithm for a class of mixed-integer nonlinear programs". In: *Mathematical Programming* 36.3, pp. 307–339. DOI: 10.1007/BF02592064.

Feistauer, M. M. (1994). *Mathematical Methods in Fluid Dynamics*. Pitman monographs and surveys in pure and applied mathematics. Harlow, Essex, England: Longman Scientific & Technical New York. DOI: 10.1002/zamm.19940741108.

Fletcher, R. and S. Leyffer (1994). "Solving mixed integer nonlinear programs by outer approximation". In: *Mathematical Programming* 66.1, pp. 327–349. DOI: 10.1007/BF01581153.

Ford, L. R. and D. R. Fulkerson (1956). "Maximal Flow through a Network". In: *Canadian Journal of Mathematics* 8.3, pp. 399–404. DOI: 10.4153/CJM-1956-045-5.

Frangioni, A., C. Gentile, and F. Lacalandra (2008). "Solving unit commitment problems with general ramp constraints". In: *International Journal of Electrical Power & Energy Systems* 30.5, pp. 316–326. DOI: 10.1016/j.ijepes.2007.10.003.

Frank, S., I. Steponavice, and S. Rebennack (2012). "Optimal power flow: a bibliographic survey I". In: *Energy Systems* 3.3, pp. 221–258. DOI: 10.1007/s12667-012-0056-y.

Freudenthal, H. (1942). "Simplizialzerlegungen von Beschränkter Flachheit". In: *Annals of Mathematics* 43.3, pp. 580–582. DOI: 10.2307/1968813.

Fügenschuh, A., B. Geißler, R. Gollmer, A. Morsi, M. E. Pfetsch, J. Rövekamp, M. Schmidt, K. Spreckelsen, and M. C. Steinbach (2015). "Physical and technical fundamentals of gas networks". In: *Evaluating Gas Network Capacities*. Ed. by T. Koch, B. Hiller, M. E. Pfetsch, and L. Schewe. SIAM-MOS series on Optimization. SIAM. Chap. 2, pp. 17–43. DOI: 10.1137/1.9781611973693.ch2.

Gamrath, G., T. Koch, A. Martin, M. Miltenberger, and D. Weninger (2015). "Progress in presolving for mixed integer programming". In: *Mathematical Programming Computation* 7.4, pp. 367–398. DOI: 10.1007/s12532-015-0083-5.

GAMS (2017). *General Algebraic Modeling System (GAMS) Release 24.8.3*. Washington, DC, USA. URL: http://www.gams.com/.

— (2018). *General Algebraic Modeling System (GAMS) Release 25.1.2*. Washington, DC, USA. URL: http://www.gams.com/.

Garey, M. R. and D. S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co.

Geißler, B. (2011). "Towards Globally Optimal Solutions of MINLPs by Discretization Techniques with Applications in Gas Network Optimization". PhD thesis. FAU Erlangen-Nürnberg.

Geißler, B., A. Martin, A. Morsi, and L. Schewe (2012a). "Using Piecewise Linear Functions for Solving MINLPs". In: *Mixed Integer Nonlinear Programming*. Ed. by J. Lee and S. Leyffer. Springer New York, pp. 287–314.

— (2012b). "Using Piecewise Linear Functions for Solving MINLPs". In: *Mixed Integer Nonlinear Programming*. Ed. by J. Lee and S. Leyffer. Vol. 154. The IMA Volumes in Mathematics and its Applications. Springer New York, pp. 287–314. DOI: 10.1007/978-1-4614-1927-3_10.

— (2015a). "The MILP-relaxation approach". In: *Evaluating Gas Network Capacities*. Ed. by T. Koch, B. Hiller, M. E. Pfetsch, and L. Schewe. SIAM-MOS series on Optimization. SIAM. Chap. 6, pp. 103–122. DOI: 10.1137/1.9781611973693.ch6.

Geißler, B., A. Morsi, and L. Schewe (2013). "A New Algorithm for MINLP Applied to Gas Transport Energy Cost Minimization". In: *Facets of Combinatorial Optimization*. Ed. by M. Jünger and G. Reinelt. Berlin, Heidelberg: Springer, pp. 321–353. DOI: 10.1007/978-3-642-38189-8_14.

Geißler, B., A. Morsi, L. Schewe, and M. Schmidt (2015b). "Solving power-constrained gas transportation problems using an MIP-based alternating direction method". In: *Computers & Chemical Engineering* 82, pp. 303–317. DOI: 10.1016/j.compchemeng.2015.07.005.

— (2018). “Solving Highly Detailed Gas Transport MINLPs: Block Separability and Penalty Alternating Direction Methods”. In: *INFORMS Journal on Computing* 30.2, pp. 309–323. DOI: 10.1287/ijoc.2017.0780.

Geoffrion, A. M. (1972). “Generalized Benders decomposition”. In: *Journal of Optimization Theory and Applications* 10.4, pp. 237–260. DOI: 10.1007/BF00934810.

Gerdts, M. (2006). “A variable time transformation method for mixed-integer optimal control problems”. In: *Optimal Control Applications and Methods* 27.3, pp. 169–182. DOI: 10.1002/oca.778.

Gleixner, A. M. (2015). “Exact and fast algorithms for mixed-integer nonlinear programming”. PhD thesis. Technische Universität Berlin. DOI: 10.14279/depositonce-4938.

Gleixner, A. M., D. Steffy, and K. Wolter (2012). *Improving the Accuracy of Linear Programming Solvers with Iterative Refinement*. eng. Tech. rep. 12-19. Takustr. 7, 14195 Berlin: ZIB.

Goldreich, O. (2010). *P, NP, and NP-Completeness: The Basics of Computational Complexity*. 1st. Cambridge University Press.

Gomory, R. E. (1958). “Outline of an algorithm for integer solutions to linear programs”. In: *Bulletin of the American Mathematical Society* 64.5, pp. 275–278.

Grötschel, M. and M. W. Padberg (1979a). “On the symmetric travelling salesman problem I: Inequalities”. In: *Mathematical Programming* 16.1, pp. 265–280. DOI: 10.1007/BF01582116.

— (1979b). “On the symmetric travelling salesman problem II: Lifting theorems and facets”. In: *Mathematical Programming* 16.1, pp. 281–302. DOI: 10.1007/BF01582117.

Gugat, M., G. Leugering, A. Martin, M. Schmidt, M. Sirvent, and D. Wintergerst (2017). “MIP-Based Instantaneous Control of Mixed-Integer PDE-Constrained Gas Transport Problems”. In: *Computational Optimization and Applications*. DOI: 10.1007/s10589-017-9970-1.

— (2018). “Towards simulation based mixed-integer optimization with differential equations”. In: *Networks*. DOI: 10.1002/net.21812.

Gupta, O. K. and A. Ravindran (1985). “Branch and Bound Experiments in Convex Nonlinear Integer Programming”. In: *Management Science* 31.12, pp. 1533–1546. DOI: 10.1287/mnsc.31.12.1533.

Gurobi (2019). *Gurobi Optimization Inc.* URL: http://www.gurobi.com (visited on 03/01/2019).

Hahn, M., S. Leyffer, and V. M. Zavala (2017). *Mixed-Integer PDE-Constrained Optimal Control of Gas Networks*. Tech. rep. URL: https://wiki.mcs.anl.gov/leyffer/images/2/27/GasNetMIP.pdf (visited on 03/01/2019).

Hante, F., G. Leugering, A. Martin, L. Schewe, and M. Schmidt (2017). "Challenges in optimal control problems for gas and fluid flow in networks of pipes and canals: From modeling to industrial applications". In: Industrial and Applied Mathematics. Ed. by P. Manchanda, R. Lozi, and A. H. Siddiqi, pp. 77–122. DOI: 10.1007/978-981-10-3758-0_5.

Hante, F. and S. Sager (2013). "Relaxation methods for mixed-integer optimal control of partial differential equations". In: *Computational Optimization and Applications* 55.1, pp. 197–225. DOI: 10.1007/s10589-012-9518-3.

Hante, F. and M. Schmidt (2017). *Complementarity-Based Nonlinear Programming Techniques for Optimal Mixing in Gas Networks*. Tech. rep. Friedrich-Alexander-Universität Erlangen-Nürnberg, p. 19.

Hiller, B., J. Humpola, T. Lehmann, R. Lenz, A. Morsi, M. E. Pfetsch, L. Schewe, M. Schmidt, R. Schwarz, J. Schweiger, C. Stangl, and B. M. Willert (2015). "Computational results for validation of nominations". In: *Evaluating Gas Network Capacities*. Ed. by T. Koch, B. Hiller, M. E. Pfetsch, and L. Schewe. SIAM-MOS series on Optimization. SIAM. Chap. 12, pp. 233–270. DOI: 10.1137/1.9781611973693.ch12.

Horst, R. and H. Tuy (1996). *Global Optimization. Deterministic Approaches*. 3rd. Springer Verlag, p. 730. DOI: 10.1007/978-3-662-03199-5.

Huchette, J. and J. P. Vielma (2016). "Small independent branching formulations for unions of V-polyhedra". URL: http://www.optimization-online.org/DB_HTML/2016/05/5454.html.

J. Brouwer, I. Gasser, and M. Herty (2011). "Gas pipeline models revisited: Model hierarchies, nonisothermal models, and simulations of networks". In: *Multiscale Modeling and Simulation* 9, pp. 601–623. DOI: 10.1137/100813580.

Jabr, R. A. (2008). "Optimal Power Flow Using an Extended Conic Quadratic Formulation". In: *IEEE Transactions on Power Systems* 23.3, pp. 1000–1008. DOI: 10.1109/TPWRS.2008.926439.

Jach, M., D. Michaels, and R. Weismantel (2008). "The Convex Envelope of (N-1)-Convex Functions". In: *SIAM J. on Optimization* 19.3, pp. 1451–1466. DOI: 10.1137/07069359X.

Jeroslow, R. G. (1973). "There Cannot be any Algorithm for Integer Programming with Quadratic Constraints". In: *Operations Research* 21.1, pp. 221–224.

Joormann, I., M. Schmidt, M. C. Steinbach, and B. M. Willert (2015). "What does "feasible" mean?" In: *Evaluating Gas Network Capacities*. Ed. by T. Koch, B. Hiller, M. E. Pfetsch, and L. Schewe. SIAM-MOS series on Optimization. SIAM. Chap. 11, pp. 211–232. DOI: 10.1137/1.9781611973693.ch11.

Jung, M. N., G. Reinelt, and S. Sager (2015). "The Lagrangian relaxation for the combinatorial integral approximation problem". In: *Optimization Methods and Software* 30.1, pp. 54–80. DOI: 10.1080/10556788.2014.890196.

Karmarkar, N. B. (1984). "A new polynomial-time algorithm for linear programming". In: *Combinatorica* 4.4, pp. 373–395. DOI: 10.1007/BF02579150.

Karp, R. M. (1972). "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations*. Ed. by R. E. Miller, J. W. Thatcher, and J. D. Bohlinger. Springer US, pp. 85–103. DOI: 10.1007/978-1-4684-2001-2_9.

Khachiyan, L. G. (1979). "A polynomial algorithm in linear programming". In: *Doklady Akademiia Nauk SSSR* 244, pp. 1093–1096.

Al-Khayyal, F. A. and J. E. Falk (1983). "Jointly Constrained Biconvex Programming". In: *Mathematics of Operations Research* 8.2, pp. 273–286. DOI: 10.1287/moor.8.2.273.

Klee, V. and G. J. Minty (1972). "How Good Is the Simplex Algorithm?" In: *Inequalities* III. Ed. by O. Shisha, pp. 159–175.

Koch, T., B. Hiller, M. E. Pfetsch, and L. Schewe, eds. (2015). *Evaluating Gas Network Capacities*. SIAM-MOS series on Optimization. SIAM. xvii + 364. DOI: 10.1137/1.9781611973693.

Kocuk, B., S. S. Dey, and X. A. Sun (2017). "New Formulation and Strong MISOCP Relaxations for AC Optimal Transmission Switching Problem". In: *IEEE Transactions on Power Systems* 32.6, pp. 4161–4170. DOI: 10.1109/TPWRS.2017.2666718.

Kocuk, B., S. S. Dey, and X. A. Sun (2018). "Matrix minor reformulation and SOCP-based spatial branch-and-cut method for the AC optimal power flow problem". In: *Mathematical Programming Computation* 10.4, pp. 557–596. DOI: 10.1007/s12532-018-0150-9.

Krebs, V., L. Schewe, and M. Schmidt (2018). "Uniqueness and multiplicity of market equilibria on DC power flow networks". In: *European Journal of Operational Research* 271.1, pp. 165–178. DOI: https://doi.org/10.1016/j.ejor.2018.05.016.

Kuhn, S. (2011). "Betriebsoptimierung von elektrischen Energieerzeugungsanlagen und Übertragungssystemen bei unvollständiger Information". PhD thesis. Universität Duisburg-Essen.

Ladányi, L., T. K. Ralphs, and L. E. Trotter Jr. (2001). "Branch, Cut, and Price: Sequential and Parallel". In: *Computational Combinatorial Optimization, Optimal or Provably Near-Optimal Solutions [Based on a Spring School]*. London, UK, UK: Springer-Verlag, pp. 223–260.

LaMaTTO++ (2015). *A Framework for Modeling and Solving Mixed-Integer Nonlinear Programming Problems on Networks*. URL: www.mso.math.fau.de/edom/projects/lamatto.html (visited on 03/01/2019).

Land, A. H. and A. G. Doig (1960). "An Automatic Method of Solving Discrete Programming Problems". In: *Econometrica* 28.3, pp. 497–520.

Lavaei, J. and S. H. Low (2012). "Zero Duality Gap in Optimal Power Flow Problem". In: *IEEE Transactions on Power Systems* 27.1, pp. 92–107. DOI: 10.1109/TPWRS.2011.2160974.

Lee, H., K. Teo, V. Rehbock, and L. Jennings (1999). "Control parametrization enhancing technique for optimal discrete-valued control problems". In: *Automatica* 35.8, pp. 1401–1407. DOI: 10.1016/S0005-1098(99)00050-3.

Lee, J., J. Leung, and F. Margot (2004). "Min-up/min-down polytopes". In: *Discrete Optimization* 1.1, pp. 77–85. DOI: 10.1016/j.disopt.2003.12.001.

Lee, J. and D. Wilson (2001). "Polyhedral methods for piecewise-linear functions. I. The lambda method". In: *Discrete Appl. Math.* 108.3, pp. 269–285. DOI: 10.1016/S0166-218X(00)00216-X.

Liberti, L. and C. C. Pantelides (2003). "Convex Envelopes of Monomials of Odd Degree". In: *Journal of Global Optimization* 25.2, pp. 157–168. DOI: 10.1023/A:1021924706467.

Lundell, A., A. Skjäl, and T. Westerlund (2013). "A reformulation framework for global optimization". In: *Journal of Global Optimization* 57.1, pp. 115–141. DOI: 10.1007/s10898-012-9877-4.

Lurie, M. V. (2008). *Modeling of Oil Product and Gas Pipeline Transportation*. Wiley-VCH.

Mahajan, A., S. Leyffer, and C. Kirches (2012). "Solving Mixed-Integer Nonlinear Programs by QP-Diving". In: *Preprint ANL/MCS-2071-0312, Argonne National Laboratory, Mathematics and Computer Science Division*.

Maher, S. J., T. Fischer, T. Gally, G. Gamrath, A. Gleixner, R. L. Gottwald, G. Hendel, T. Koch, M. E. Lübbecke, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, S. Schenker, R. Schwarz, F. Serrano, Y. Shinano, D. Weninger, J. T. Witt, and J. Witzig (2017). *The SCIP Optimization Suite 4.0*. Tech. rep. 17-12. Takustr. 7, 14195 Berlin: ZIB. URL: https://opus4.kobv.de/opus4-zib/frontdoor/index/index/docId/6217 (visited on 03/01/2019).

Mahlke, D., A. Martin, and S. Moritz (2010). "A mixed integer approach for time-dependent gas network optimization". In: *Optimization Methods and Software* 25.4, pp. 625–644. DOI: 10.1080/10556780903270886.

Maria, J., T. T. Truong, J. Yao, T.-W. Lee, R. G. Nuzzo, S. Leyffer, S. K. Gray, and J. A. Rogers (2009). "Optimization of 3D Plasmonic Crystal Structures

for Refractive Index Sensing". In: *The Journal of Physical Chemistry C* 113.24, pp. 10493–10499. DOI: 10.1021/jp9024552.

Markowitz, H. M. and A. S. Manne (1957a). "On the Solution of Discrete Programming Problems". In: *Econometrica* 25.1, pp. 84–110.

— (1957b). "On the Solution of Discrete Programming Problems". In: *Econometrica* 25.1, pp. 84–110.

Martin, A., M. Möller, and S. Moritz (2006). "Mixed Integer Models for the Stationary Case of Gas Network Optimization". In: *Mathematical Programming* 105.2, pp. 563–582. DOI: 10.1007/s10107-005-0665-5.

McCormick, G. P. (1976). "Computability of global solutions to factorable nonconvex programs: Part I — Convex underestimating problems". In: *Mathematical Programming* 10.1, pp. 147–175. DOI: 10.1007/BF01580665.

MIPLIB (2018). *MIPLIB 2017*. URL: http://miplib.zib.de.

Misener, R. and C. A. Floudas (2010). "Piecewise-linear approximations of multidimensional functions". In: *J. Optim. Theory Appl.* 145.1, pp. 120–147. DOI: 10.1007/s10957-009-9626-0.

Morsi, A. (2013). "Solving MINLPs on Loosely-Coupled Networks with Applications in Water and Gas Network Optimization". PhD thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU).

Muckstadt, J. A. and S. A. Koenig (1977). "An Application of Lagrangian Relaxation to Scheduling in Power-Generation Systems". In: *Operations Research* 25.3, pp. 387–403. DOI: 10.1287/opre.25.3.387.

O'Rourke, J. (1998). *Computational Geometry in C*. Cambridge University Press. DOI: 10.1017/CBO9780511804120.

Piao, L., M. de Weerdt, and L. de Vries (2017). "Electricity market design requirements for DC distribution systems". In: *2017 IEEE Second International Conference on DC Microgrids (ICDCM)*, pp. 95–101. DOI: 10.1109/ICDCM.2017.8001028.

Pottmann, H., R. Krasauskas, B. Hamann, K. Joy, and W. Seibold (2000). "On piecewise linear approximation of quadratic functions". In: *Journal for Geometry and Graphics Volume* 4, pp. 31–53.

Purchala, K., L. Meeus, D. V. Dommelen, and R. Belmans (2005). "Usefulness of DC power flow for active power flow analysis". In: *IEEE Power Engineering Society General Meeting, 2005*, 454–459 Vol. 1. DOI: 10.1109/PES.2005.1489581.

Quesada, I. and I. E. Grossmann (1992). "An LP NLP based branch and bound algorithm for convex MINLP optimization". In: *Computers & Chemical Engineering* 16, pp. 937–947.

Rebennack, S. and J. Kallrath (2015a). "Continuous piecewise linear delta-approximations for bivariate and multivariate functions". In: *J. Optim. Theory Appl.* 167.1, pp. 102–117. DOI: 10.1007/s10957-014-0688-2.

— (2015b). "Continuous piecewise linear delta-approximations for univariate functions: computing minimal breakpoint systems". In: *J. Optim. Theory Appl.* 167.2, pp. 617–643. DOI: 10.1007/s10957-014-0687-3.

Ríos-Mercado, R. Z. and C. Borraz-Sánchez (2015). "Optimization problems in natural gas transportation systems: A state-of-the-art review". In: *Applied Energy* 147, pp. 536–555. DOI: 10.1016/j.apenergy.2015.03.017.

Rovatti, R., C. D'Ambrosio, A. Lodi, and S. Martello (2014). "Optimistic MILP modeling of non-linear optimization problems". In: *European J. Oper. Res.* 239.1, pp. 32–45. DOI: 10.1016/j.ejor.2014.03.020.

Rüffler, F. and F. Hante (2016). "Optimal switching for hybrid semilinear evolutions". In: *Nonlinear Analysis: Hybrid Systems* 22, pp. 215–227. DOI: 10.1016/j.nahs.2016.05.001.

Ryoo, H. S. and N. V. Sahinidis (1995). "Global optimization of nonconvex NLPs and MINLPs with applications in process design". In: *Computers & Chemical Engineering* 19.5, pp. 551–566. DOI: 10.1016/0098-1354(94)00097-2.

— (1996). "A branch-and-reduce approach to global optimization". In: *Journal of Global Optimization* 8.2, pp. 107–138. DOI: 10.1007/BF00138689.

Sager, S., H. G. Bock, and G. Reinelt (2009). "Direct methods with maximal lower bound for mixed-integer optimal control problems". In: *Mathematical Programming* 118.1, pp. 109–149. DOI: 10.1007/s10107-007-0185-6.

Sager, S., M. Jung, and C. Kirches (2011). "Combinatorial integral approximation". In: *Mathematical Methods of Operations Research* 73.3, pp. 363–380. DOI: 10.1007/s00186-011-0355-4.

Salgado, E., A. Scozzari, F. Tardella, and L. Liberti (2018). "Alternating Current Optimal Power Flow with Generator Selection". In: *Combinatorial Optimization*. Ed. by J. Lee, G. Rinaldi, and A. R. Mahjoub. Springer International Publishing, pp. 364–375.

Saravanan, B., S. Das, S. Sikri, and D. P. Kothari (2013). "A solution to the unit commitment problem—a review". In: *Frontiers in Energy* 7.2, pp. 223–236. DOI: 10.1007/s11708-013-0240-3.

Savelsbergh, M. W. P. (1994). "Preprocessing and Probing Techniques for Mixed Integer Programming Problems". In: *INFORMS Journal on Computing* 6.4, pp. 445–454. DOI: 10.1287/ijoc.6.4.445.

Schewe, L. and M. Schmidt (2018). "Computing Feasible Points for MINLPs with MPECs". In: *Mathematical Programming Computation*. DOI: 10.1007/s12532-018-0141-x. optonline: 2016/12/5778. Forthcoming.

Schmidt, M. (2013). "A generic interior-point framework for nonsmooth and complementarity constrained nonlinear optimization". PhD thesis. Leibniz Universität Hannover.

Schmidt, M., D. Aßmann, R. Burlacu, J. Humpola, I. Joormann, N. Kanelakis, T. Koch, D. Oucherif, M. E. Pfetsch, L. Schewe, R. Schwarz, and M. Sirvent (2017). "GasLib—A Library of Gas Network Instances". In: *Data* 2.4. DOI: 10.3390/data2040040.

Schmidt, M., M. Sirvent, and W. Wollner (2018). "A decomposition method for MINLPs with Lipschitz continuous nonlinearities". In: *Mathematical Programming*. DOI: 10.1007/s10107-018-1309-x.

Schultz, R. and T. Wollenberg (2017). "Unit commitment under uncertainty in AC transmission systems via risk averse semidefinite stochastic Programs". In: *RAIRO-Oper. Res.* 51.2, pp. 391–416. DOI: 10.1051/ro/2016031.

Schweppe, F. C., M. C. Caramanis, R. D. Tabors, and R. E. Bohn (1988). "Spot Pricing of Electricity". In: Boston, MA: Springer US. DOI: 10.1007/978-1-4613-1683-1.

Siqueira, A., R. Gaia Costa da Silva, and L.-R. Santos (2016). "Perprof-py: A Python Package for Performance Profile of Mathematical Optimization Software". In: *Journal of Open Research Software* 4. DOI: 10.5334/jors.81.

Sirvent, M. (2018). "Incorporating Differential Equations into Mixed-Integer Programming for Gas Transport Optimization". PhD thesis. FAU Erlangen-Nürnberg.

Smith, E. M. and C. C. Pantelides (1997). "Global optimisation of nonconvex MINLPs". In: *Computers & Chemical Engineering* 21, S791–S796. DOI: 10.1016/S0098-1354(97)87599-0.

Smith, W. D. (2000). "A Lower Bound for the Simplexity of the n-cube via Hyperbolic Volumes". In: *European Journal of Combinatorics* 21.1, pp. 131–137.

Stigler, H. and C. Todem (2005). "Optimization of the Austrian Electricity Sector (Control Zone of VERBUND APG) by Nodal Pricing". English. In: *Central European journal of operations research* 13.2, pp. 105–125.

Stubbs, R. A. and S. Mehrotra (2002). "Generating Convex Polynomial Inequalities for Mixed 0–1 Programs". In: *Journal of Global Optimization* 24.3, pp. 311–332. DOI: 10.1023/A:1020351410169.

Tawarmalani, M. and N. V. Sahinidis (2002). *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming*. Kluwer Academic Publishers, p. 478. DOI: 10.1007/978-1-4757-3532-1.

Tawarmalani, M. and N. V. Sahinidis (2004). "Global optimization of mixed-integer nonlinear programs: A theoretical and computational study". In: *Mathematical Programming* 99.3, pp. 563–591. DOI: 10.1007/s10107-003-0467-6.

Traub, J. F., G. W. Wasilkowski, and H. Woźniakowski, eds. (1988). *Information-Based Complexity*. Boston, San Diego, and New York: Academic Press.

Van Roy, T. J. and L. A. Wolsey (1987). "Solving Mixed Integer Programming Problems Using Automatic Reformulation". In: *Oper. Res.* 35.1, pp. 45–57. DOI: 10.1287/opre.35.1.45.

Verfürth, R. (1996). *A Review of a Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*. John Wiley & Sons Inc and B. G. Teubner Publishers.

Vielma, J. P., S. Ahmed, and G. L. Nemhauser (2010). "Mixed-Integer Models for Nonseparable Piecewise-Linear Optimization: Unifying Framework and Extensions". In: *Operations Research* 58.2, pp. 303–315. DOI: 10.1287/opre.1090.0721.

Vielma, J. P. and G. L. Nemhauser (2011). "Modeling disjunctive constraints with a logarithmic number of binary variables and constraints". In: *Mathematical Programming* 128.1–2, pp. 49–72.

Vittal, V. and A. Bergen (2000). *Power Systems Analysis*. 2nd.

Westerlund, T. and K. Lundqvist (2001). *Alpha-ECP, version 5.01: An interactive MINLP-solver based on the extended cutting plane method*. Tech. rep. 01-178-A.

Westerlund, T. and F. Pettersson (1995). "An extended cutting plane method for solving convex MINLP problems". In: *Computers & Chemical Engineering* 19, pp. 131–136. DOI: 10.1016/0098-1354(95)87027-X.

Weymouth, T. (1912). "Problems in natural gas engineering". In: *Transactions of the American Society of Mechanical Engineers* 34.1349, pp. 185–231.

Wilson, D. (1998). "Polyhedral methods for piecewise-linear functions". PhD thesis. University of Kentucky.

Wunderling, R. (1996). "Paralleler und objektorientierter Simplex-Algorithmus". PhD thesis. Technische Universität Berlin.

Xpress (2019). *FICO Xpress Optimization*. URL: http://www.fico.com/en/products/fico-xpress-optimization (visited on 03/01/2019).

Zimmerman, R. D., C. E. Murillo-Sanchez, and R. J. Thomas (2011). "MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education". In: *IEEE Transactions on Power Systems* 26.1, pp. 12–19. DOI: 10.1109/TPWRS.2010.2051168.

List of Figures

List of Tables