Mixed-integer convex optimization: outer approximation algorithms and modeling power

by

Miles Lubin

B.S., The University of Chicago (2011)
M.S., The University of Chicago (2011)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2017

Author .....
Sloan School of Management
August 11, 2017

Certified by .....
Juan Pablo Vielma
Assistant Professor of Operations Research
Thesis Supervisor

Accepted by .....
Dimitris Bertsimas
Boeing Professor of Operations Research
Co-director, Operations Research Center

Mixed-integer convex optimization: outer approximation algorithms and modeling power

by

Miles Lubin

Submitted to the Sloan School of Management
on August 11, 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In this thesis, we study mixed-integer convex optimization, or mixed-integer convex programming (MICP), the class of optimization problems where one seeks to minimize a convex objective function subject to convex constraints and integrality restrictions on a subset of the variables. We focus on two broad and complementary questions on MICP.

The first question we address is, "what are efficient methods for solving MICP problems?" The methodology we develop is based on outer approximation, which allows us, for example, to reduce MICP to a sequence of mixed-integer linear programming (MILP) problems. By viewing MICP from the conic perspective of modern convex optimization as defined by Ben-Tal and Nemirovski, we obtain significant computational advances over the state of the art, e.g., by automating extended formulations by using disciplined convex programming. We develop the first finite-time outer approximation methods for problems in general mixed-integer conic form (which includes mixed-integer second-order-cone programming and mixed-integer semidefinite programming) and implement them in an open-source solver, Pajarito, obtaining competitive performance with the state of the art.

The second question we address is, "which nonconvex constraints can be modeled with MICP?" This question is important for understanding both the modeling power gained in generalizing from MILP to MICP and the potential applicability of MICP to nonconvex optimization problems that may not be naturally represented with integer variables. Among our contributions, we completely characterize the case where the number of integer assignments is bounded (e.g., mixed-binary), and to address the more general case we develop the concept of "rationally unbounded" convex sets. We show that under this natural restriction, the projections of MICP feasible sets are well behaved and can be completely characterized in some settings.

Thesis Supervisor: Juan Pablo Vielma
Title: Assistant Professor of Operations Research

Acknowledgments

I would like to thank my advisor, Juan Pablo Vielma, for his unending support and enthusiasm during my time as his student. He embodies the "Yes, and..." rule that I learned at a collaboration workshop; in our frequent meetings, his first response to any idea I propose has always been positive and followed by thoughtful and constructive critique. As a result, I was able to jump between numerous interesting topics and projects before converging to the work described in this thesis. I couldn't have asked for a better advisor. I would also like to thank Robert Freund and Stephen Boyd for serving on my thesis committee and for a number of stimulating discussions.

My path into optimization and operations research took a number of turns, and I would like to acknowledge and thank a number of people who helped me along the way before I arrived at MIT. My first class on optimization was a course on nonlinear programming taught by Elvio Pilotta at Universidad Nacional de Córdoba in Argentina during my semester abroad there. Elvio was a patient teacher, and my interest in the material led me to seek out research opportunities in optimization less than a year later. Andrew Siegel, whose class on High-Performance Computing (HPC) I was taking in Winter 2010, encouraged me to apply for an internship at Argonne National Lab and made the connections so that I could work under the supervision of Cosmin Petra and Mihai Anitescu developing methods to solve large-scale stochastic programming problems using HPC. My work at Argonne defined the start of my research career, and I thank Cosmin and Mihai for being great mentors and for making sure that I had no lack of challenging problems to work on. I would like to thank Kipp Martin, as a great teacher and valuable collaborator and mentor, together with Julian Hall, Victor Zavala, and Burhan Sandıkçı whom I also had the pleasure of working with during my time at Argonne. Julian graciously hosted me in Edinburgh for my first departmental seminar before I had even started my Ph.D.; I hope my presentations have improved since then because I noticed a few people sleeping in the audience.

This thesis would look very different without my collaborators Emre Yamangil,

Russell Bent, Chris Coey, and Ilias Zadik. It was during a barbecue at Los Alamos that Emre mentioned he would soon start working on convex MINLP, and based on my experience with extended formulations for MISOCP [88], I responded, "I have an idea for that." Emre's enthusiasm and Russell's support and insight helped refine the idea and prove that it was practically useful. Chris joined the project over a year ago and soon afterwards decided that the existing code was nonsense and needed to be rewritten. His insistence on making sure every line of code is justified has been inspiring and led us to develop a much better understanding of our own methods than we had before. I thank Ilias for giving me an opportunity to return to my mathematical roots and for being curious and excited about a problem that is outside of the scope of his main body of work in probability. Many of the key insights on MICP representability are due to him, and many of the results included here are based on my efforts to understand these insights.

My time at Los Alamos National Lab was one of the highlights of my graduate school experience. I was fortunate to be able to visit the Center for Nonlinear Studies (CNLS) for extended periods of time during the summers of 2014, 2015, and 2016. I thank my host, Russell Bent, for making this possible and my other collaborators there, Scott Backhaus, Misha Chertkov, Sidhant Misra, Harsha Nagarajan, Line Roald, Kaarthik Sundar, and Emre Yamangil as well as the all of the summer students, postdocs, and staff I interacted with for making the lab such a stimulating and friendly place. Special thanks go to hiking partners Jonas Kersulis and Deepjyoti Deka; to skiing partners Andrey Lokhov, Marc Vuffray, and Anatoly Zlotnik; to roommate and collaborator, Yury Dvorkin; and to wise remote collaborator Dan Bienstock.

Returning to MIT, I have to thank my friend and collaborator Iain Dunning for, among other things, turning our little side project into something that will have a lasting impact. I thank Joey Huchette for becoming an invaluable member of the JuliaOpt team and the whole world of Julia contributors for creating such a supportive community. Madeleine Udell deserves special recognition both for comments on earlier versions of this work and for creating Convex.jl, without which I likely would have

never learned about disciplined convex programming. From my colleagues at the ORC, all of whom I will miss, I would like to thank Michael Beeler and Nikita Korolko for their friendship, with special thanks again to hiking partners Frans de Ruiter, Jack Dunn, Sebastien Martin, and Yee Sian Ng. I thank Dimitris Bertsimas and Patrick Jaillet for keeping the ORC ship on course as co-directors. I thank Chris Coey, Ilias Zadik, and Nishanth Mundru for comments on drafts of the thesis.

I have had the opportunity to travel during my time as a student, and I thank Marcos Goycoolea, Eduardo Moreno, Tony Kelman, Aleksandr Kazachkov, Jack Poulson, Noemi Petra, Felipe Serrano, Robert Schwarz, Abel Siqueira, Ruth Misener, Victor Zavala, and Michael Friedlander for inviting me and hosting me at their respective institutions, and Line Roald for hosting me in Zurich.

I would like to acknowledge the Computational Science Graduate Fellowship for providing four years of support which without doubt changed the course of my research career, and in particular Aric Hagberg for setting up my first contacts with Los Alamos.

Finally, I am grateful to my family for supporting me on the long road to a Ph.D.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 MICP

In this thesis, we study mixed-integer convex optimization, or mixed-integer convex programming (MICP), the class of optimization problems where one seeks to minimize a convex objective function subject to convex constraints and integrality restrictions on a subset of the variables. A defining characteristic of MICP that distinguishes it from more general nonconvex optimization is that even though the integrality restrictions can be viewed as nonconvex constraints, they are discrete constraints. Specifically, in the typical case where the number of possible assignments to the integer variables is finite, in principle one can compute a globally optimal solution by solving a finite number of a convex optimization problems, which is not true for nonconvex optimization in general. Intuitively, we can think of convex optimization problems as easy to solve to global optimality, although some caveats apply. This observation may give us hope that there is sufficient structure in MICP problems to design practical algorithms that guarantee global optimality in finitely many steps. Note that although we focus on global optimality in our algorithmic efforts, it is reasonable to infer that by broadening the class of problems that we are able to solve to global optimality, we will also broaden the class of problems for which we can obtain (provably) good heuristic solutions in an appropriate amount of time, which in practice is perhaps what is actually useful.

MICP is a natural extension of mixed-integer linear optimization, or mixed-integer linear programming (MILP), where the objective function and constraints must be linear and polyhedral, respectively. Indeed, Kelley [50], in his widely cited paper that proposed the cutting-plane method for convex optimization, immediately noted how his approach could be combined with Gomory's algorithm [36] for pure integer linear programming to solve so-called integer convex programming problems, where all variables are constrained to integer values. Following early work in integer programming, of which Gomory's algorithm is just one piece, MILP has established itself as a practical framework for optimization problems in scheduling, logistics, planning, and many other areas. Although these problems are in general NP-Hard, almost 60 years of investment in MILP techniques has resulted in powerful commercial and open-source solvers that can solve MILP problems of practical interest within reasonable time limits [51]. Over the same period, convex optimization, or convex programming (CP), has grown into a well-developed field with solution methods that are both efficient in theory and widely used in practice [17, 48, 6].

Despite these advances in MILP and CP, our impression is that the more general MICP has not entered the mainstream of optimization in terms of both academic interest and applications in practice, with the exception of the special case of mixed-integer second-order-cone programming (MISOCP) that has recently become supported by commercial solvers. A critic might explain this situation by suggesting that more general MICP problems are either too difficult to solve or that there aren't sufficient real-world motivations to consider solving them in such generality.

We would respond to this explanation as follows. To date, the mainstream of MICP developments has been based on traditional nonlinear programming (NLP) under the name of convex mixed-integer nonlinear programming (convex MINLP). Ben-Tal and Nemirovski [7] draw a distinction between convex NLP and *modern convex optimization*, the former being based on smooth functions and the Karush-Kuhn-Tucker (KKT) conditions and the latter being based on conic duality with "nice" cones that we will later describe. In broad terms, our algorithmic techniques can be seen as modern-convex-optimization (i.e., conic) analogues of existing methods

for convex MINLP. The key insights we develop in Chapter 2 could be considered trivial by anyone familiar with extended formulations [45, 83], conic duality [6], and disciplined convex programming (DCP) [40], yet we were able to use these insights to build software that solved previously open benchmark instances. We suppose that this is the case both because there is little overlap in the MILP and CP research communities and because the computational infrastructure around modern convex optimization, namely modeling interfaces and solvers, has made significant progress in the past decade. For example, development of Bonmin [14], a leading convex MINLP solver, started in 2004, before the now widely-used DCP modeling package CVX [38] was released. Our subsequent work in Chapter 3 demonstrates that the additional structure provided by the conic approach can yield both better understanding of the structure of MICP problems and practical computational advances. We would argue, therefore, that it is at the very least premature and ill-informed to dismiss MICP on the basis of perceived difficulty because there remains fertile ground for additional advances by making connections between MILP and CP. As two examples of areas that could yield further significant advances, in this thesis we consider neither cuts that refine convex relaxations based on integrality information [52, 54], nor the effect of strong formulations, which have been crucial to MILP [86].

We would also argue that MICP has a demonstrated, yet not nearly fully realized, potential for significant real-world applications. Not attempting to be comprehensive, we briefly mention some of these applications. Bonami, Kilinç, and Linderoth [15] cite applications of convex MINLP and MISOCP in portfolio optimization, block layout design in manufacturing, network design, and design and control of chemical processes among others. SOCP arises from chance constraints [61] or robust optimization [8], on top of which it is natural to impose integrality restrictions, e.g., to make scheduling decisions [82]. Convex relaxations of the nonconvex power flow equations [22] can be combined with on/off decisions to enhance power grid operations [56]. Nonconvex obstacle avoidance constraints in robotics can be modeled with MICP [57]. Cardinality-constrained convex statistical estimation problems can be formulated as MICP problems [75, 69]. Finally, our preliminary work on MICP has already been

cited in a paper on genetic sequencing [21]. In a rigorous sense, the scope of MICP applicability is not known because there has been very little work on characterizing precisely which nonconvex optimization problems can be expressed exactly in MICP form. In Chapter 4 and our related publication, to our knowledge we are the first authors to consider the question of which nonconvex sets can serve as the feasible regions for MICP problems in full generality. Our results provide a new understanding of the scope and limitations of MICP.

Concluding our response to the critic, we would suggest that MICP is a promising research area for the academic community, and that more practical applications will follow once the technology is developed. In the remainder of the thesis, we hope to provide further support for this argument. We now introduce the three chapters which serve as the body of the thesis.

1.2 Chapter 2: Polyhedral outer approximation, extended formulations, and disciplined convex programming

This chapter is based on work with Emre Yamangil, Russell Bent, and Juan Pablo Vielma. Preliminary work was published in conference proceedings as [63] with an extended journal article currently under review [64].

In order to employ MILP to solve MICP problems, we relax the convex constraints by representing them as an intersection of a finite number of half-spaces, that is, polyhedral constraints. Based on this idea, Duran and Grossman [31] and Leyffer [58] developed the *outer approximation* (OA) algorithm which solves a sequence of MILP and convex NLP subproblems to deliver a globally optimal solution for convex MINLP problems in a finite number of iterations; we present a generalized version of this algorithm which does not rely on NLP subproblems in Section 2.1.

Despite the fact that many MICP approaches, including the OA algorithm, build on MILP approaches, there remains a significant performance gap between the two

problem classes. Bonami, Kilinç, and Linderoth [15] note in a recent review that continued advances in MILP and NLP have translated into "far more modest" growth in the scale of problems which convex MINLP solvers can solve within reasonable time limits.

The cases in which the OA algorithm and others based on polyhedral approximation perform poorly are those in which the convex set of constraints is poorly approximated by a small number of half-spaces. In Section 2.2, we review a simple example identified by Hijazi et al. [45] where the OA algorithm requires 2^n iterations to solve an MICP instance with n decision variables. Fortunately, [45] also propose a solution based on ideas from Tawarmalani and Sahinidis [83] that can significantly improve the quality of a polyhedral approximation by constructing the approximation in a higher dimensional space. These constructions are known as *extended formulations*, which have also been considered by [88, 53]. Although Hijazi et al. demonstrate impressive computational gains by using extended formulations, implementing these techniques within traditional convex MINLP solvers requires more structural information than provided by value-and-gradient oracles through which these solvers typically interact with nonlinear functions. To our knowledge, MINOTAUR [59] is the only such solver to automate extended formulations. In Section 2.3 we identify the modeling concept of disciplined convex programming (DCP) [40], popularized in the CVX software package [38], as a practical solution to the problem of automatically generating extended formulations based on a user's algebraic representation of an MICP problem.

Our investigation of DCP leads us in Section 2.4 to consider conic optimization as a representation of convex constraints that compactly encodes all of the information needed to construct extended formulations. This key observation links together a number of streams in modern convex optimization and MICP research, and in particular explains the increasingly popular role of MISOCP and how it can be extended to cover "general" MICP. Pulling these pieces together, in Section 2.5 we develop the first finite-time OA algorithm for mixed-integer conic optimization problems based on conic duality. We note explicit failure cases when the assumptions of strong duality

or bounded number integer variables are not satisfied. In Section 2.6, we present a prototype of `Pajarito`, a new solver for MICP based on the conic OA algorithm and compare its efficiency with state-of-the-art convex MINLP solvers Bonmin, SCIP, and Artelys Knitro. We report the solution of a number of previously unsolved benchmark instances.

1.3 Chapter 3: $\mathcal{K}^*$ cuts, LP-based branch-and-bound, and `Pajarito`

This chapter is based on work with Chris Coey and Juan Pablo Vielma which is soon to be submitted for publication.

While in Chapter 2, conic form is motivated as a convenient encoding for automating extended formulations, in this chapter we take conic form as the starting point and develop a comprehensive computational framework for solving mixed-integer conic optimization problems by outer approximation. Our methodology treats standard problem classes like MISOCP and MISDP as well as problems involving the nonsymmetric exponential cone.

In Section 3.2 we develop "$\mathcal{K}^*$ cuts" as a unifying way to understand outer approximation of convex cones. We address the question of how to initialize a polyhedral outer approximation, making connections with known inner/outer approximations of the PSD cone [3]. We obtain cuts via conic duality from solving continuous relaxations and continuous subproblems with bound restrictions on the integer-constrained variables, generalizing results from Chapter 2 that were restricted to subproblems with all integer-constrained variables fixed. We show that separating hyperplanes based on subgradients, which are in use by some existing solvers, may also be interpreted as $\mathcal{K}^*$ cuts.

Following the introduction of $\mathcal{K}^*$ cuts, in Section 3.3 we present an LP-based branch-and-bound algorithm which is a conic analogue of the Quesada and Grossmann [76] algorithm for convex MINLP. It differs from the iterative OA approach

proposed in Chapter 2 by using a single branch-and-bound tree instead of solving a sequence of independent MILP subproblems and hence may be faster. The analysis of this algorithm assumes that LP subproblems are solved to exact solutions. In Section 3.4 we slightly relax this assumption, allowing the LP solver to enforce the $\mathcal{K}^*$ cuts with some positive tolerance on violations. In this setting, we can correct the $\mathcal{K}^*$ cuts by rescaling them and, with a modified branch-and-bound algorithm, guarantee that we find a globally optimal solution for any strictly positive relative optimality gap.

In Section 3.5 we consider extensions to the $\mathcal{K}^*$-cuts framework, obtaining multiple linear cuts or second-order cone outer approximations from a single cut for the PSD cone and cleanly handling lifting of the second-order cone [88].

In Section 3.6, we present our solver Pajarito and its architecture as the first MICP solver written in the Julia language [12]. Notably, Pajarito was almost completely rewritten following the preliminary results in Chapter 2. Finally, in our computational experiments in Section 3.7, we test a number of algorithmic variants to gain a better understanding of what works best in practice. When we put our best methods against other competitors on the MISOCP benchmark library in CBLIB [34], we find that Pajarito outperforms all open-source competitors and obtains competitive performance with CPLEX's specialized MISOCP algorithm when we use CPLEX purely as an MILP solver.

1.4 Chapter 4: The modeling power of MICP

This chapter is based on work with Ilias Zadik and Juan Pablo Vielma which has been submitted for publication [66]. It is an extension of the conference publication [65].

Early advances in solution techniques for mixed-integer linear programming (MILP) motivated studies by Jeroslow and Lowe [49] and others (recently reviewed in [86]) on understanding precisely which sets can be encoded as projections of points within a closed polyhedron satisfying integrality restrictions on a subset of the variables. These sets can serve as feasible sets in mixed-integer linear optimization problems

and hence an understanding of their structure yields a better understanding of what can be modeled with MILP. Jeroslow and Lowe, for example, proved that the epigraph of the piecewise linear function $f(x)$ which equals 1 if $x > 0$ and 0 if $x = 0$, is not representable over the domain $x \geq 0$. Such a function would naturally be used to model a fixed cost in production. It is now well known that an upper bound on x is required in order to encode such fixed costs in an MILP formulation.

Motivated by our progress in MICP solution techniques, in this chapter we address the analogous question of which nonconvex sets may be represented as projections of points within a convex set satisfying integrality restrictions on a subset of the variables, i.e., which sets may serve as feasible regions for MICP problems. To our knowledge, our work in [65] was the first to consider this general case. Related but more specific analysis has been developed by Del Pia and Poskin [24] where they characterized the case where the convex set is an intersection of a polyhedron with an ellipsoidal region and by Dey and Morán [25] where they studied the structure of integer points within convex sets but without allowing a mix of continuous and discrete variables.

In Section 4.2 we provide a complete characterization of what can be represented when the number of possible integer assignments is bounded, extending [65] with a revised formulation that can be proven to represent the convex hull of a union of convex sets.

The most substantial contributions of the chapter focus on the case with infinitely many possible integer assignments, for which we develop the notion of *rational MICP* (Section 4.3) as an analogue of rational MILP that reasonably restricts the directions of unboundedness. This notion has been revised and simplified from [65]. Using the additional structure of rational MICP, we can make a number of strong statements characterizing MICP representations of compact sets, epigraphs, subsets of the natural numbers, and graphs of piecewise linear functions.

In Section 4.4 we develop properties independent of the rationality assumption that can be used to put a lower bound on how many integer dimensions are needed to represent given nonconvex sets. These properties are based on a necessary criterion

for MICP representability presented in [65], which we used to prove, for example, that the set of rank-one matrices is not representable.

Our results on MICP representability enable us to answer, for the most part, a question posed by S. Boyd in a private discussion. The question was, to what extent can a DCP framework like CVX be extended to include basic operations which are MICP representable, i.e., *which functions may be represented as MIDCP atoms*? (See Section 2.4 for background on MIDCP.) A corollary of Theorem 2 is that functions defined on a compact domain whose epigraphs are rational MICP representable (with an additional regularity condition) must be piecewise convex with finitely many pieces. This observation suggests that it is reasonable to restrict one's attention to such piecewise convex functions when designing MIDCP atoms.

1.5 Notation

We use the notation $[\![k]\!]$ to denote the set $\{1, 2, \ldots, k\}$. We use boldface symbols to refer to matrices and vectors. We use subscript and drop boldface to refer to elements of a vector. We use superscripts to index over a collection of vectors. $\mathrm{cl}(A)$ is the closure of the set A. $\boldsymbol{e}(k)$ is vector of appropriate length with zero everywhere except for $(e(k))_k = 1$. $\mathbf{1}$ and $\mathbf{0}$ are the vectors of all 1 and 0, respectively, with dimension implied by the context. We denote the set of real numbers by $\mathbb{R}$ and the set of rational numbers by $\mathbb{Q}$.

Following [48, 17] we say that a set $S \subseteq \mathbb{R}^n$ is **convex** if for all $\boldsymbol{x}^1, \boldsymbol{x}^2 \in S$ and for all $\lambda \in [0, 1]$ we have $\lambda\boldsymbol{x}^1 + (1-\lambda)\boldsymbol{x}^2 \in S$. We say a set $S \subseteq \mathbb{R}^n$ is a **cone** if for all $\boldsymbol{x} \in S$ and for all $\lambda \in [0, \infty)$ we have $\lambda\boldsymbol{x} \in S$. Given a function $f : S \to \mathbb{R}$ for some $S \subseteq \mathbb{R}^n$ we define its **epigraph** to be the set $\mathrm{epi}(f) = \{(t, \boldsymbol{x}) \in \mathbb{R} \times S : t \geq f(\boldsymbol{x})\}$. If $\mathrm{epi}(f)$ is a convex set then we say f is a convex function. We define a **polyhedron** to be a finite intersection of closed halfspaces, i.e., a set of the form $\{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}$ for some matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$.

Notation used in a specific chapter or section will be introduced as appropriate. Note that we use both $\boldsymbol{c}^T\boldsymbol{x}$ and $\langle \boldsymbol{c}, \boldsymbol{x} \rangle$ to denote the inner product $\sum_{i \in [\![n]\!]} c_i x_i$ in

different chapters.

Chapter 2

Polyhedral approximation in mixed-integer convex optimization

2.1 Polyhedral outer approximation

We state a generic mixed-integer convex optimization problem as

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x} \in \mathcal{X}, \\ & x_i \in \mathbb{Z}, l_i \leq x_i \leq u_i \quad \forall i \in [\![I]\!], \end{aligned} \tag{MICONV}$$

where $\mathcal{X}$ is a closed, convex set, and the first I of N variables are constrained to take integer values with explicit finite bounds l_i and u_i. We assume that the objective function is linear. This assumption is without loss of generality because, given a convex, nonlinear objective function $f(\boldsymbol{x})$, we may introduce an additional variable t, constrain $(t, \boldsymbol{x})$ to belong to the epigraph of f, and then take t as the linear objective to minimize; see [15]. For concreteness, the convex set of constraints $\mathcal{X}$ could be specified as

$$\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^n : g_j(\boldsymbol{x}) \leq 0, j \in [\![J]\!]\}, \tag{2.1}$$

for some collection of J smooth, convex functions, in which case MICONV would typically be called a convex MINLP instance. We refer to the constraints $x_i \in \mathbb{Z}\ \forall\, i \in [\![I]\!]$ as integrality constraints. Note that when these integrality constraints are relaxed (i.e., removed), MICONV becomes a convex optimization problem.

A straightforward approach for finding the global solution of MICONV is branch and bound. Branch and bound is an enumerative algorithm where lower bounds derived from relaxing the integrality constraints in MICONV are combined with recursively partitioning the space of possible integer solutions. The recursive partition is based on "branches" such as $x_i \leq k$ and $x_i \geq k+1$ for some integer-constrained index $i \in [\![I]\!]$ and some value k chosen between the lower bound l_i and the upper bound u_i of x_i. In the worst case, branch and bound requires enumerating all possible assignments of the integer variables, but in practice it can perform much better by effectively pruning the search tree. Gupta and Ravindran [42] describe an early implementation of branch-and-bound for MICP, and Bonami et al. [16] more recently revisit this approach.

On many but certainly not all problems, however, the branch-and-bound algorithm is not competitive with an alternative family of approaches based on *polyhedral outer approximation*. Driven by the availability of effective solvers for LP and MILP, it was observed in the early 1990s by Leyffer and others [58] that it is often more effective to avoid solving convex, nonlinear relaxations, when possible, in favor of solving polyhedral relaxations using MILP. This idea has influenced a majority of the solvers recently reviewed and benchmarked by Bonami et al. [15].

In this section, we will provide a sketch of an OA algorithm. We derive the algorithm in a more general way than most authors that will later be useful in the discussion of mixed-integer conic problems in Section 2.5, although for intuition and concreteness of the discussion we illustrate the key points of the algorithm for the case of the smooth, convex representation (2.1), which is the traditional setting. We refer readers to [14, 31, 1] for a more rigorous treatment of the traditional setting and Section 2.5 for more on the conic setting (i.e., when $\mathcal{X}$ is an intersection of convex cone and an affine subspace). We begin by defining polyhedral outer approximations.

Definition 1. *A set P is a polyhedral outer approximation of a convex set $\mathcal{X}$ if P is a polyhedron and P contains $\mathcal{X}$, i.e., $\mathcal{X} \subseteq P$.*

Note that we have not specified the explicit form of the polyhedron. While the traditional OA algorithm imagines P to be of the form $\{\boldsymbol{x} \in \mathbb{R} : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}$ for some $\boldsymbol{A}$ and $\boldsymbol{b}$, it is useful to not tie ourselves, for now, to a particular representation of the polyhedra.

Polyhedral outer approximations of convex sets are quite natural in the sense that every closed convex set can be represented as an intersection of an *infinite* number of closed half-spaces [48]. For instance, when $\mathcal{X}$ is given in the functional form (2.1) and each function $g_j : \mathbb{R}^n \to \mathbb{R}$ is smooth and finite-valued over $\mathbb{R}^n$ then the following equivalence holds:

$$\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^n : g_j(\boldsymbol{x}') + \nabla g_j(\boldsymbol{x}')^T(\boldsymbol{x} - \boldsymbol{x}') \leq 0 \ \forall \, \boldsymbol{x}' \in \mathbb{R}^n, j \in [\![J]\!]\}, \tag{2.2}$$

where $\nabla g_j(\boldsymbol{x}')$ is the gradient of g_j. When some g_j functions are not defined (or do not take finite values) over all of $\mathbb{R}^n$ then these "gradient inequalities" plus additional linear constraints enforcing the domain of each g_j provide a representation of $\mathcal{X}$ as an intersection of halfspaces; see [48] for further discussion.

Hence, in the most basic case, a polyhedral approximation of $\mathcal{X}$ can be derived by picking a finite number of points $S \subset \mathbb{R}^n$ and collecting the half-spaces in (2.2) for $\boldsymbol{x}' \in S$ instead of for all $\boldsymbol{x}' \in \mathbb{R}^n$. What is perhaps surprising is that a finite number of half-spaces provides a sufficient representation of $\mathcal{X}$ in order to solve MICONV to global optimality, under some assumptions which we later prove are necessary. This idea is encompassed by the OA algorithm which we now describe.

Given a polyhedral outer approximation P of the constraint set $\mathcal{X}$, we define the following mixed-integer linear *relaxation* of MICONV:

$$
\begin{aligned}
r_P = \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T \boldsymbol{x} \\
\text{s.t.} \quad & \boldsymbol{x} \in P, \\
& x_i \in \mathbb{Z},\, l_i \leq x_i \leq u_i \quad \forall i \in [\![I]\!]\,.
\end{aligned}
\tag{MIOA(P))}
$$

Note that MIOA(P) is a relaxation because any $\boldsymbol{x}$ feasible to MICONV must be feasible to MIOA(P). Therefore the optimal value of MIOA(P) provides a lower bound on the optimal value of MICONV. This bound *may* be NP-Hard to compute, since it requires solving a mixed-integer linear optimization problem; nevertheless we may use existing, powerful MILP solvers for these relaxations.

We refer to the integer-constrained components of a solution $\boldsymbol{x}$ as $\boldsymbol{x}_{[\![I]\!]}$. Given a solution $\boldsymbol{x}^*$ to MIOA(P), the OA algorithm proceeds to solve the continuous, convex problem CONV($\boldsymbol{x}^*_{[\![I]\!]}$) that results from fixing the integer-constrained variables $\boldsymbol{x}_{[\![I]\!]}$ to their values in $\boldsymbol{x}^*_{[\![I]\!]}$:

$$
\begin{aligned}
v_{\boldsymbol{x}^*_I} = \min \quad & \boldsymbol{c}^T \boldsymbol{x} \\
\text{s.t.} \quad & \boldsymbol{x} \in \mathcal{X}, \\
& \boldsymbol{x}_{[\![I]\!]} = \boldsymbol{x}^*_{[\![I]\!]}.
\end{aligned}
\tag{CONV($\boldsymbol{x}^*_{[\![I]\!]}$))}
$$

If CONV($\boldsymbol{x}^*_{[\![I]\!]}$) is feasible, let $\boldsymbol{x}'$ be the optimal solution. Then $\boldsymbol{x}'$ is a feasible solution to MICONV and provides a corresponding upper bound on the best possible objective value. If the objective value of this convex subproblem equals the objective value of MIOA(P) (i.e., $\boldsymbol{c}^T\boldsymbol{x}' = \boldsymbol{c}^T\boldsymbol{x}^*$), then $\boldsymbol{x}'$ is a globally optimal solution of MICONV. If there is a gap, then the OA algorithm must update the polyhedral outer approximation P and re-solve MIOA(P) with a tighter approximation, yielding a nondecreasing sequence of lower bounds.

To ensure finite termination of OA it is sufficient to prevent repetition of unique assignments of the integer-valued components $\boldsymbol{x}^*_{[\![I]\!]}$, because by assumption there is only a finite number of possible values. The following lemma states a condition on

the polyhedral outer approximation P that helps prove finite convergence.

Lemma 1. *Fixing $\boldsymbol{z} \in \mathbb{Z}^I$, if $\boldsymbol{x} \in P$ implies $\boldsymbol{c}^T\boldsymbol{x} \geq v_{\boldsymbol{z}}$ for all $\boldsymbol{x} \in \mathbb{R}^N$ with $\boldsymbol{x}_{[\![I]\!]} = \boldsymbol{z}$ where $v_{\boldsymbol{z}}$ is the optimal value of CONV($\boldsymbol{z}$) then the OA algorithm must terminate if MIOA(P) returns an optimal solution $\boldsymbol{x}^*$ with integer components matching $\boldsymbol{x}^*_{[\![I]\!]} = \boldsymbol{z}$.*

Proof. Assume we solve MIOA(P) and obtain a solution $\boldsymbol{x}^*$. If the integer part of $\boldsymbol{x}^*$ matches $\boldsymbol{z}$, by our assumptions we have $r_P = \boldsymbol{c}^T\boldsymbol{x}^* \geq v_{\boldsymbol{z}}$, where r_P is the optimal value of MIOA(P). Since MIOA(P) is a relaxation and $v_{\boldsymbol{z}}$ is the objective value of a feasible solution, then we must have $r_P = v_{\boldsymbol{z}}$. Thus, we have proven global optimality of this feasible solution and terminate. □

Note that Lemma 1 provides a general condition that does not assume any particular representation of the convex constraints $\mathcal{X}$. In the traditional setting of the smooth, convex representation (2.1), if $\boldsymbol{x}'$ is an optimal solution to CONV($\boldsymbol{x}^*_{[\![I]\!]}$) *and strong duality holds*, e.g., as in Prop 5.1.5 of Bertsekas [11], then the set of constraints

$$g_j(\boldsymbol{x}') + \nabla g_j(\boldsymbol{x}')^T(\boldsymbol{x} - \boldsymbol{x}') \leq 0 \;\forall\, j \in [\![J]\!] \tag{2.3}$$

are sufficient to enforce the condition in Lemma 1 for finite convergence. In other words, within the OA loop after solving CONV($\boldsymbol{x}^*_{[\![I]\!]}$), updating P by adding the constraints (2.3) is sufficient to ensure that the integer solution $\boldsymbol{x}^*_I$ does not repeat, except possibly at termination. Intuitively, strong duality in CONV($\boldsymbol{x}^*_{[\![I]\!]}$) implies that there are no descent directions (over the continuous variables) from $\boldsymbol{x}'$ which are feasible to a first-order approximation of the constraints $g_j(\boldsymbol{x}) \leq 0$ for $j \in [\![J]\!]$ [11]. Hence a point $\boldsymbol{x}$ sharing the integer components $\boldsymbol{x}_{[\![I]\!]} = \boldsymbol{x}^*_{[\![I]\!]}$ must satisfy $\boldsymbol{c}^T(\boldsymbol{x} - \boldsymbol{x}') \geq 0$ or precisely $\boldsymbol{c}^T\boldsymbol{x} \geq v_{\boldsymbol{x}^*_I}$. See [58, 31, 1] for further discussion.

If CONV($\boldsymbol{x}^*_{[\![I]\!]}$) is infeasible, then to ensure finite convergence it is important to refine the polyhedral approximation P to exclude the corresponding integer point. That is, we update P so that

$$\{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x}_{[\![I]\!]} = \boldsymbol{x}^*_{[\![I]\!]}\} \cap P = \emptyset. \tag{2.4}$$

In the traditional smooth setting, it is possible in the infeasible case to derive a set of constraints analogous to (2.3), e.g., by solving an auxiliary feasibility problem where we also assume strong duality holds [14, 1].

To review, the OA algorithm proceeds in a loop between the MILP relaxation MIOA(P) and the continuous subproblem with integer values fixed CONV($\boldsymbol{x}^*_{[\![I]\!]}$). The MILP relaxation provides lower bounds and feeds integer assignments to the continuous subproblem. The continuous subproblem yields feasible solutions *and* sufficient information to update the polyhedral approximation in order to avoid repeating the same assignment of integer values. The algorithm is stated more formally in Algorithm 1 and illustrated in Figure 2-1.

Algorithm 1 The polyhedral outer approximation (OA) algorithm

Initialize: $z_U \leftarrow \infty, z_L \leftarrow -\infty$, polyhedron $P \supset \mathcal{X}$ such that MIOA(P) is bounded. Fix convergence tolerance ϵ.
while $z_U - z_L \geq \epsilon$ **do**
  Solve MIOA(P).
  if MIOA(P) is infeasible **then**
    MICONV is infeasible, so terminate.
  end if
  Let $\boldsymbol{x}^*$ be the optimal solution of MIOA(P) with objective value w_T.
  Update lower bound $z_L \leftarrow w_T$.
  Solve CONV($\boldsymbol{x}^*_{[\![I]\!]}$).
  if CONV($\boldsymbol{x}^*_{[\![I]\!]}$) is feasible **then**
    Let $\boldsymbol{x}'$ be an optimal solution of CONV($\boldsymbol{x}^*_{[\![I]\!]}$) with objective value $v_{\boldsymbol{x}^*_{[\![I]\!]}}$.
    Derive polyhedron Q satisfying $\boldsymbol{x} \in Q$ with $\boldsymbol{x}_{[\![I]\!]} = \boldsymbol{x}^*_{[\![I]\!]}$ implies $\boldsymbol{c}^T\boldsymbol{x} \geq v_{\boldsymbol{x}^*_{[\![I]\!]}}$ by using strong duality (e.g., (2.3)).
    if $v_{\boldsymbol{x}^*_{[\![I]\!]}} < z_U$ **then**
      Update upper bound $z_U \leftarrow v_{\boldsymbol{x}^*_{[\![I]\!]}}$.
      Record $\boldsymbol{x}'$ as the best known solution.
    end if
  else if CONV($\boldsymbol{x}^*_{[\![I]\!]}$) is infeasible **then**
    Derive polyhedron Q satisfying $\{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x}_{[\![I]\!]} = \boldsymbol{x}^*_{[\![I]\!]}\} \cap Q = \emptyset$.
  end if
  Update $P \leftarrow P \cap Q$.
end while

The efficiency of the OA algorithm is derived from the speed of solving the MIOA(P) problem by using state-of-the-art MILP solvers. Indeed, in 2014 benchmarks by Hans Mittelman, the OA algorithm implemented within Bonmin using CPLEX as the MILP

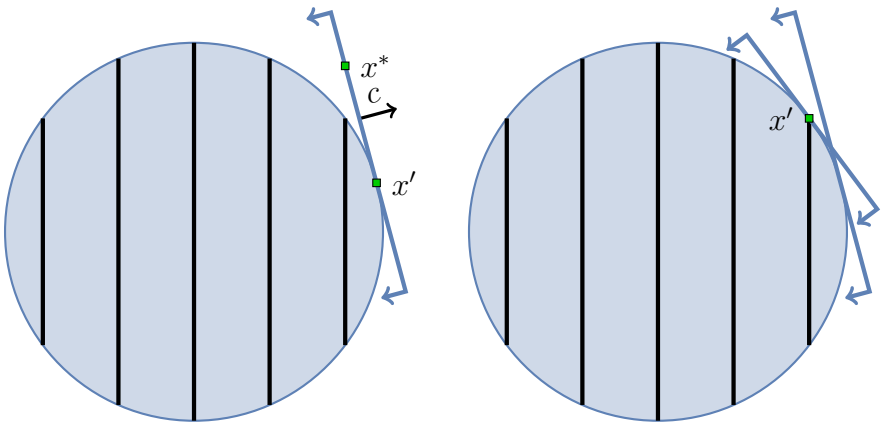


Figure 2-1: An illustration of the outer approximation algorithm. Here, we minimize a linear objective c over the ball $x_1^2 + x_2^2 \leq 2.5$ with x_1 integer constrained. On the left, the point $\boldsymbol{x}'$ is the solution of the continuous relaxation, and we initialize the polyhedral outer approximation with the tangent at $\boldsymbol{x}'$. We then solve the MIOA(P) subproblem, which yields $\boldsymbol{x}^*$. Fixing $x_1 = 2$, we optimize over the circle and update the polyhedral approximation with the tangent at $\boldsymbol{x}'$ (on the right). In the next iteration of the OA algorithm (not shown), we will prove global optimality of $\boldsymbol{x}'$.

solver was found to be the fastest among convex MINLP solvers [70]. In spite of taking advantage of MILP solvers, the traditional OA algorithm suffers from the fact that the gradient inequalities (2.3) may not be sufficiently strong to ensure fast convergence. In the following section, we identify when these conditions may occur and how to work around them within the framework of OA.

2.2 Outer approximation enhancements

The OA algorithm is powerful but relies on polyhedral outer approximations serving as good approximations of convex sets. The assumptions of the OA algorithm guarantee that there exists a polyhedron P such that the optimal objective value of MIOA(P) matches the optimal objective value of MICONV, precisely at convergence. In the case that MICONV has no feasible solution, these assumptions furthermore guarantee that there exists an outer approximating polyhedron P such that MIOA(P) has no feasible solution. In Section 2.5, we discuss in more detail what may happen when the assumptions fail, although even in the typical case when they are satisfied, these

polyhedra may have exponentially many constraints. Indeed, there are known cases where the OA algorithm requires 2^n iterations to converge in $\mathbb{R}^n$. In this section, we review an illustrative case where the OA algorithm performs poorly and the techniques from the literature that have been proposed to address this issue.

Figure 2-2 illustrates an example developed by Hijazi et al. [45], specifically the problem,

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t.} \quad & \sum_{i \in [\![n]\!]} \left(x_i - \frac{1}{2} \right)^2 \leq \frac{n-1}{4}, \\ & \boldsymbol{x} \in \mathbb{Z}^n, \boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{1}, \end{aligned} \tag{2.5}$$

which, regardless of the objective vector $\boldsymbol{c}$, has no feasible solutions. Any polyhedral approximation of the single convex constraint, a simple ball, requires 2^n half-spaces until the corresponding outer approximation problem MIOA(P) has no feasible solution. At this point the OA algorithm terminates reporting infeasibility.

Hijazi et al. propose a simple yet powerful reformulation that addresses this poor convergence behavior. To motivate their reformulation, we recall a basic example from linear programming. The ℓ_1 unit ball, i.e., $\{\boldsymbol{x} \in \mathbb{R}^n : \sum_{i \in [\![n]\!]} |x_i| \leq 1\}$, is representable as an intersection of half spaces in $\mathbb{R}^n$, namely the 2^n half spaces of the form $\sum_{i \in [\![n]\!]} s_i x_i \leq 1$ where $s_i = \pm 1$. This exponentially large representation of the ℓ_1 ball is seldom used in practice, however. Instead, it is common to introduce extra variables z_i with constraints

$$z_i \geq x_i, z_i \geq -x_i \text{ for } i \in [\![n]\!] \text{ and } \sum_{i \in [\![n]\!]} z_i \leq 1. \tag{2.6}$$

It is not difficult to show that $||\boldsymbol{x}||_1 \leq 1$ if and only if there exist $\boldsymbol{z}$ satisfying the constraints (2.6). Note that these $2n + 1$ constraints define a polyhedron in $\mathbb{R}^{2n}$, which we call an *extended formulation* of the ℓ_1 ball because the ℓ_1 ball is precisely the projection of this polyhedron defined in $(\boldsymbol{x}, \boldsymbol{z})$ space onto the space of $\boldsymbol{x}$ variables. It is

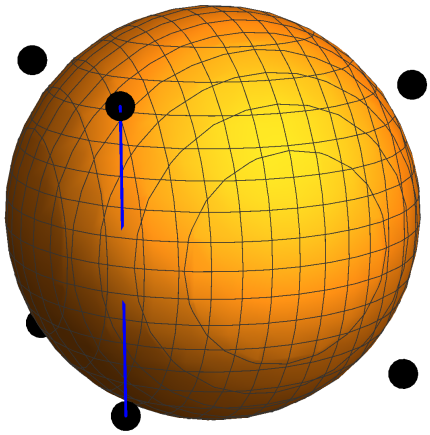


Figure 2-2: The example developed by Hijazi et al. [45] demonstrating the case where the outer approximation algorithm requires 2^n iterations to converge in dimension n. The intersection of the ball with the integer lattice points (in black) is empty, yet any polyhedral outer approximation of the ball in $\mathbb{R}^n$ requires 2^n hyperplanes before it has an empty intersection with the integer lattice, because the line segments between any two lattice points (one of which is drawn) intersect the ball. Hence, any hyperplane can separate at most one lattice point from the ball, and we require 2^n of these to prove infeasibility.

well known that polyhedra, such as the ℓ_1 ball, that require a large description as half-spaces in $\mathbb{R}^n$ might have a representation with many fewer half-spaces if additional variables are introduced [68]. Note, in this case, that the extended formulation is derived by introducing a variable z_i to represent the epigraph $\{(z, x) : |x| \leq z\}$ of each $|x_i|$ term, taking advantage of the fact that the ℓ_1 ball can be represented as a constraint on a sum of these univariate functions.

The solution proposed by Hijazi et al. and earlier by Tawarmalani and Sahinidis [83] follows this line of reasoning by introducing an extended formulation for the polyhedral representation of the smooth ℓ_2 ball. Analogously to the case of the ℓ_1 ball, Hijazi et al. construct an outer-approximating polyhedron in $\mathbb{R}^{2n}$ with $2n + 1$ constraints which contains no integer points. By the previous discussion, we know that the projection of this small polyhedron in $\mathbb{R}^{2n}$ must have at least 2^n inequalities in $\mathbb{R}^n$. Their solution precisely exploits the separability structure in the definition of

the ℓ_2 ball, introducing an extra variable z_i for each term and solving instead,

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{z}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t.} \quad & \sum_{i\in[\![n]\!]} z_i \leq \frac{n-1}{4}, \\ & z_i \geq \left(x_i - \frac{1}{2}\right)^2, \quad \forall\, i \in [\![n]\!] \\ & \boldsymbol{x} \in \mathbb{Z}^n, \mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1}. \end{aligned} \tag{2.7}$$

The OA algorithm applied to (2.7) proves infeasibility in 2 iterations because it constructs polyhedral approximations (based on gradient inequalities (2.3)) to the constraints in the $(\boldsymbol{x}, \boldsymbol{z})$ space. More generally, Hijazi et al. and Tawarmalani and Sahinidis propose to reformulate any convex constraint of the form $\sum_i f_i(x_i) \leq k$ as $\sum_i z_i \leq k$ and $z_i \geq f_i(x_i)$ for each i where f_i are univariate convex functions. Just by performing this simple transformation before providing the problem to the OA algorithm, they are able to achieve impressive computational gains in reducing the time to solution and number of iterations of the algorithm.

Building on the ideas of Hijazi et al. and Tawarmalani and Sahinidis, Vielma et al. [88] propose an extended formulation for the *second-order cone* $\{(r, \boldsymbol{t}) \in \mathbb{R}^{1+n} : ||\boldsymbol{t}||_2 \leq r\}$, which is not immediately representable as a sum of univariate convex functions. They recognize that the second-order cone is indeed representable as a sum of bivariate convex functions, i.e., $\sum_{i\in[\![n]\!]} t_i^2/r \leq r$, after squaring both sides and dividing by r. They obtain an extended formulation by introducing auxiliary variables $z_i \geq t_i^2/r$ and constrain $\sum_{i\in[\![n]\!]} z_i \leq t$. This simple transformation was subsequently implemented by commercial solvers for MISOCP like Gurobi [13], CPLEX [84], and Xpress [5], yielding significant improvements on their internal and public benchmarks.

In spite of the promising computational results of Hijazi et al. first reported in 2011 and the more recent extension by Vielma et al., to our knowledge, MINOTAUR [59] is the only convex MINLP solver which has (very recently) implemented these techniques in an automated way. To understand why others like Bonmin [14] have not done so, it is important to realize that convex MINLP solvers historically have had no

concept of the mathematical or algebraic structure behind their constraints, instead viewing them through black-box oracles to query first-order and possibly second-order derivative values. The summation structure we exploit, which is algebraic in nature, is simply not available when viewed through this form, making it quite difficult to retrofit this functionality into the existing architectures of convex MINLP solvers. In the following section, we will propose a substantially different representation of MICP problems that is a natural fit for extended formulations.

2.3 Disciplined Convex Programming (DCP) as a solution

In order to implement the extended formulation proposal of [45] in an automated way, one may be led to attempt a direct analysis of a user's algebraic representation of the convex constraints in a problem. However, this approach is far from straightforward. First of all, the problem of *convexity detection* is necessary as a subroutine, because it is only correct to exploit summation structure of a convex function $h(\boldsymbol{x}) = f(\boldsymbol{x})+g(\boldsymbol{x})$ when *both* f and g are convex. This is not a necessary condition for the convexity of h; consider $f(\boldsymbol{x}) = x_1^2 - x_2^2$ and $g(\boldsymbol{x}) = 2x_2^2$. Convexity detection of algebraic expressions is NP-Hard [4], which poses challenges for implementing such an approach in a reliable and scalable way. *Ad-hoc* approaches [32] are possible but are highly sensitive to the form in which the user inputs the problem; for example, approaches based on composition rules fail to recognize convexity of $\sqrt{x_1^2 + x_2^2}$ and $\log(\exp(x_1) + \exp(x_2))$ [83].

Instead of attempting such analyses of arbitrary algebraic representations of convex functions, we propose to use the modeling concept of disciplined convex programming (DCP) first proposed by Grant, Boyd, and Ye [40, 39]. In short, DCP solves the problem of convexity detection by asking users to express convex constraints in such a way that convexity is proven by composition rules, which are sufficient but not necessary. These composition rules are those from basic convex analysis, for example,

the sum of convex functions is convex, the point-wise maximum of convex functions is convex, and the composition $f(g(x))$ is convex when f is convex and nondecreasing and g is convex. The full set of DCP rules is reviewed in [40, 81].

Even though it is possible to write down convex functions which do not satisfy these composition rules, the DCP philosophy is to disallow them and instead introduce new *atoms* (or basic operations) which users must use when writing down their model. For example, $\text{logsumexp}(x_1, x_2)$ replaces $\log(\exp(x_1) + \exp(x_2))$ and $\text{norm}(x_1, x_2)$ replaces $\sqrt{x_1^2 + x_2^2}$. Although asking users to express their optimization problems in this form breaks away from the traditional setting of convex MINLP, DCP also formalizes the folklore within the convex MINLP community that the way in which you write down the convex constraints can have a significant impact on the solution time; see, e.g., Hijazi et al. [45] and our example later discussed in Equation 2.17.

The success over the past decade of the CVX software package [38] which implements DCP has demonstrated that this modeling concept is practical. Users are willing to learn the rules of DCP in order to gain access to powerful (continuous, convex) solvers, and furthermore the number of basic atoms needed to cover nearly all convex optimization problems of practical interest is relatively small.

Although we motivated DCP as a solution to the subproblem of convexity detection, it in fact provides a complete solution to the problem of automatically generating an extended formulation and encoding it in a computationally convenient form given a user's algebraic representation of a problem. All DCP-valid expressions are compositions of basic operations (atoms); for example the expression $\max\{\exp(x^2), -2x\}$ is DCP-valid because the basic composition rules prove its convexity. A lesser-known aspect of DCP is that these rules of composition have a 1-1 correspondence with extended formulations based on the epigraphs of the atoms. Observe, for example, that

$$t \geq \max\{\exp(x^2), -2x\} \tag{2.8}$$

if and only if

$$t \geq \exp(x^2), t \geq -2x \tag{2.9}$$

if and only if there exists s such that

$$s \geq x^2, t \geq \exp(s), t \geq -2x, \tag{2.10}$$

where the validity of the latter transformation holds precisely because $\exp(\cdot)$ is increasing and therefore $s \geq x^2$ implies $\exp(s) \geq \exp(x^2)$. Furthermore, the constraints $s \geq x^2$ and $t \geq \exp(s)$ are convex because square and exp are convex functions; hence (2.10) is a convex extended formulation of (2.8). Note that while we previously discussed extended formulations derived only from disaggregating sums, disaggregating compositions of functions in this form also yields stronger polyhedral approximations [83]. The existence of this extended formulation is no coincidence. Grant and Boyd [39] explain that a tractable representation of the epigraph of an atom is sufficient to incorporate it into a DCP modeling framework. That is, if an implementation of DCP knows how to optimize over a model with the constraint $t \geq f(x)$ for some convex function f, then f can be incorporated as an atom within the DCP framework and used within much more complex expressions so long as they follow the DCP composition rules.

Our analysis of DCP has led us to the conclusion that DCP provides the means to automate the generation of extended formulations in a way that has never been done in the context of MICP. Users need only express their MICP problem by using a DCP modeling language like CVX or more recent implementations like CVXPY [26] (in Python), or Convex.jl [85] (in Julia). Any DCP-compatible model is convex by construction and emits an extended formulation that can safely disaggregate sums and more complex compositions of functions.

We do note that in some cases it may not be obvious how to write a known convex function in DCP form. In our initial work translating convex MINLP benchmark instances into DCP form, we were unable to find a DCP representation of the univariate concave function $\frac{x}{x+1}$ which is not in DCP form because division of affine expressions is neither convex nor concave in general. Fortunately, a reviewer suggested rewriting $\frac{x}{x+1} = 1 - \frac{1}{x+1}$ where $\frac{1}{x+1}$ is a DCP-recognized convex function so long as $x + 1 \geq 0$.

With this trick we were able to translate *all* of the benchmark instances we considered into DCP form, as we discuss in more details in the following section.

2.4 MIDCP and conic representability

While DCP modeling languages have traditionally supported only convex problems, CVX recently added support for mixed-integer convex problems under the name of MI**D**CP, and the subsequently-developed DCP modeling languages also support integrality constraints. We will use the terminology MIDCP to refer to MICP models expressed in DCP form. In the previous section we argued that an MIDCP representation of an MICP problem provides sufficient information to construct an extended formulation, which in turn could be used to accelerate the convergence of the outer approximation algorithm by providing strong polyhedral approximations. However, an MIDCP representation is quite complex, much more so than the "black-box" derivative-based representation that traditional convex MINLP solvers work with. Handling the MIDCP form requires understanding each atom within the DCP library and manipulating the *expression graph* data structures which are used to represent the user's algebraic expressions.

It turns out that there is a representation of MIDCP models which is much more compact and convenient for use as an input format for an MICP solver, and this is as *mixed-integer conic* optimization problems. Before stating the form of these problems, we first consider the standard continuous conic optimization problem:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \in \mathcal{K}, \end{aligned} \tag{CONE}$$

where $\mathcal{K} \subseteq \mathbb{R}^n$ is a closed convex cone. A simple example of a cone is the nonnegative orthant $\mathbb{R}^n_+ = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x} \geq \boldsymbol{0}\}$. When $\mathcal{K} = \mathbb{R}^n_+$ then (CONE) reduces to a linear programming problem. Typically, $\mathcal{K}$ is a product of cones $\mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_r$, where

each $\mathcal{K}_i$ is one of a small number of recognized cones.

One of Grant et al.'s original motivations for developing the DCP framework was to provide access to powerful solvers for the second-order cone (SOC) [60],

$$\mathcal{L}^{1+n} = \left\{(r, \boldsymbol{t}) \in \mathbb{R}^{1+n} : r \geq \|\boldsymbol{t}\|_2\right\}, \tag{2.11}$$

and the cone of positive semidefinite matrices,

$$\text{PSD}_n = \{\boldsymbol{A} \in \mathbb{R}^{n \times n} : \boldsymbol{A} = \boldsymbol{A}^T, \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} \geq 0 \, \forall \, \boldsymbol{x} \in \mathbb{R}^n\}. \tag{2.12}$$

CVX, for example, does *not* use smooth, derivative-based representations of the epigraphs of atoms but instead uses a conic representation of each of its atoms. For instance, for $x, y \geq 0$ the epigraph of the negated geometric mean $f(x, y) = -\sqrt{xy}$ is a convex set representable as $t \geq -\sqrt{xy}$ iff $\exists \, z \geq 0$ such that $-t \leq z \leq \sqrt{xy}$ iff

$$-t \leq z \text{ and } z^2 \leq xy \text{ iff } \; -t \leq z \text{ and } (x/\sqrt{2}, y/\sqrt{2}, z) \in \mathcal{V}^3, \tag{2.13}$$

where

$$\mathcal{V}^{2+n} = \left\{(r, s, \boldsymbol{t}) \in \mathbb{R}^{2+n} : r, s \geq 0, 2rs \geq \|\boldsymbol{t}\|_2^2\right\} \tag{2.14}$$

is the n-dimensional *rotated* second-order cone, a common cone useful for modeling (e.g., functions like x^2) which itself is representable as a transformation of the second-order cone [6]. While this conic representation of the geometric mean is known in the literature [6], it is arguably unnecessarily complex for modelers to understand, and CVX, for example, provides a `geo_mean` atom which transparently handles this transformation.

Subsequent to the second-order and positive semidefinite cones, researchers have investigated the exponential cone [80],

$$\mathcal{E} = \text{cl}\left(\left\{(r, s, t) \in \mathbb{R}^3 : s > 0, r \geq s \exp\left(\frac{t}{s}\right)\right\}\right), \tag{2.15}$$

Table 2.1: A categorization of the 333 MICP instances in the MINLPLIB2 library according to conic representability. Over two thirds are pure MISOCP problems and nearly one third is representable by using the exponential (EXP) cone alone. All instances are representable by using standard cones.

SOC only	EXP only	SOC and EXP	POW only	Not representable	Total
217	107	7	2	0	333

and the power cone [44],

$$\mathrm{POW}_\alpha = \{(x, y, z) \in \mathbb{R}^3 : |z| \leq x^\alpha y^{1-\alpha}, x \geq 0, y \geq 0\}, \tag{2.16}$$

which can be used to represent functions like entropy ($-x\log(x)$) and fractional powers, respectively. This small collection of cones is sufficient to represent any convex optimization problem which you may input within existing DCP implementations, including CVX.

In the context of MICP, these cones are indeed sufficient from our experience. We classified all 333 convex MINLP instances from the MINLPLIB2 benchmark library [89] and found that 217 are representable by using purely second-order cones (and so fall under the previously mentioned MISOCP problem class), 107 are representable by using purely exponential cones, and the remaining by some mix of second-order, exponential, and power cones; see Table 2.1. No instances require the PSD cone, because mixed-integer semidefinite programming (MISDP) falls outside of the scope of traditional convex MINLP. Of particular note are the trimloss [43] family of instances which have constraints of the form,

$$\sum\nolimits_{k\in[\![q]\!]} -\sqrt{x_k y_k} \leq \boldsymbol{c}^T \boldsymbol{z} + b. \tag{2.17}$$

Prior to our report in [63], the `tls5` and `tls6` instances had been unsolved since 2001. By directly rewriting these problems into MIDCP form, we obtained an MISOCP representation because all constraints are representable by using second-order cones, precisely by using the transformation of the geometric mean discussed above. Once in MISOCP form, we provided the problem to Gurobi 6.0, which was able to solve

them to global optimality within a day, indicating the value of conic formulations.

Given that DCP provides an infrastructure to translate convex problems into conic form, we may consider mixed-integer conic problems as a compact representation of MIDCP problems. Below, we state our standard form (in the scope of this chapter) for mixed-integer conic problems,

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{z}} \quad & \boldsymbol{c}^T\boldsymbol{z} \\ \text{s.t.} \quad & \boldsymbol{A}^x\boldsymbol{x} + \boldsymbol{A}^z\boldsymbol{z} = \boldsymbol{b} \\ & \boldsymbol{L} \leq \boldsymbol{x} \leq \boldsymbol{U}, \boldsymbol{x} \in \mathbb{Z}^n, \boldsymbol{z} \in \mathcal{K}, \end{aligned} \tag{MICONE}$$

where $\mathcal{K} \subseteq \mathbb{R}^k$ is a closed convex cone. Without loss of generality, we assume integer-constrained variables are not restricted to belong to cones, since we may introduce corresponding continuous variables by equality constraints. In Section 2.5 we discuss solving MICONE via polyhedral outer approximation.

2.5 Outer approximation algorithm for mixed-integer conic problems

The observations of the previous section motivated the development of a solver for problems of the form MICONE. We note that the traditional convergence theory is generally insufficient because it assumes differentiability, while conic problems have nondifferentiability that is sometimes intrinsic to the model. Nonsmooth perspective functions like $f(x, y) = x^2/y$, for example, which are used in disjunctive convex optimization [19], have been particularly challenging for derivative-based convex MINLP solvers and have motivated smooth approximations [41]. On the other hand, conic form can handle these nonsmooth functions in a natural way, so long as there is a solver capable of solving the continuous conic relaxations.

In this section, we present the first OA algorithm with finite-time convergence guarantees for problems of the form MICONE. The convergence guarantees depend

on two necessary assumptions, first on the existence of finite bounds on the integer-constrained variables, and second on strong duality holding in certain convex subproblems. We provide convergence counterexamples when each of these assumptions does not hold.

We begin with the definition of dual cones.

Definition 2. *Given a cone $\mathcal{K}$, we define $\mathcal{K}^* := \{\boldsymbol{\beta} \in \mathbb{R}^k : \boldsymbol{\beta}^T \boldsymbol{z} \geq 0 \ \forall \boldsymbol{z} \in \mathcal{K}\}$ as the dual cone of $\mathcal{K}$.*

Dual cones provide an equivalent outer description of any closed, convex cone, as the following lemma states. We refer readers to [6] for the proof.

Lemma 2. *Let $\mathcal{K}$ be a closed, convex cone. Then $\boldsymbol{z} \in \mathcal{K}$ iff $\boldsymbol{z}^T \boldsymbol{\beta} \geq 0 \ \forall \boldsymbol{\beta} \in \mathcal{K}^*$.*

We note that the second-order cone $\mathcal{L}$, the rotated second order cone $\mathcal{V}$ (2.14), and the cone of positive semidefinite matrices are *self-dual*, which means that the dual cone and the original cone are the same [6]. While the exponential and power cones are not self-dual, the discussions that follow are valid for them and other general cones.

Based on the above lemma, we state the analogue of the MILP relaxation MIOA(P) for (MICONE) as

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{z}} \quad & \boldsymbol{c}^T \boldsymbol{z} \\ \text{s.t.} \quad & \boldsymbol{A}^x \boldsymbol{x} + \boldsymbol{A}^z \boldsymbol{z} = \boldsymbol{b} \\ & \boldsymbol{L} \leq \boldsymbol{x} \leq \boldsymbol{U}, \boldsymbol{x} \in \mathbb{Z}^n, \\ & \boldsymbol{\beta}^T \boldsymbol{z} \geq 0 \ \forall \beta \in T. \end{aligned} \tag{MICONEOA(T))}$$

Note that if $T = \mathcal{K}^*$, MICONEOA(T) is an equivalent semi-infinite representation of (MICONE). If $T \subset \mathcal{K}^*$ and $|T| < \infty$ then MICONEOA(T) is an MILP outer approximation of (MICONE) whose objective value is a lower bound on the optimal value of (MICONE). In the context of the discussion in Section 2.1, given T, our polyhedral approximation of $\mathcal{K}$ is $P_T = \{\boldsymbol{z} : \boldsymbol{\beta}^T \boldsymbol{z} \geq 0 \ \forall \boldsymbol{\beta} \in T\}$, and we explicitly treat the linear equality constraints separately.

In the conic setting, we state the continuous subproblem CONV($\boldsymbol{x}^*_{[\![I]\!]}$) with integer values fixed as

$$\begin{aligned} v_{\boldsymbol{x}^*} = \min_{\boldsymbol{z}} \quad & \boldsymbol{c}^T\boldsymbol{z} \\ \text{s.t.} \quad & \boldsymbol{A}^z\boldsymbol{z} = \boldsymbol{b} - \boldsymbol{A}^x\boldsymbol{x}^*, \\ & \boldsymbol{z} \in \mathcal{K}. \end{aligned} \tag{CONE($\boldsymbol{x}^*$))}$$

Using conic duality, we obtain the dual of CONE($\boldsymbol{x}^*$) as

$$\begin{aligned} \max_{\boldsymbol{\beta},\boldsymbol{\lambda}} \quad & \boldsymbol{\lambda}^T(\boldsymbol{b} - \boldsymbol{A}^x\boldsymbol{x}^*) \\ \text{s.t.} \quad & \boldsymbol{\beta} = \boldsymbol{c} - (\boldsymbol{A}^z)^T\boldsymbol{\lambda} \\ & \boldsymbol{\beta} \in \mathcal{K}^*. \end{aligned} \tag{2.18}$$

Under the assumptions of strong duality, the optimal solutions $\boldsymbol{\beta}$ to the dual problem (2.18) correspond precisely to the half-spaces which ensure the conditions in Lemma 1 when CONE($\boldsymbol{x}^*$) is feasible; hence, we add these solutions to the set T. When CONE($\boldsymbol{x}^*$) is infeasible and (2.18) is unbounded, the rays of (2.18) provide solutions that satisfy (2.4), guaranteeing finite convergence of the OA algorithm. The following two lemmas prove these statements.

Lemma 3. *Given $\boldsymbol{x}^*$, assume CONE($\boldsymbol{x}^*$) is feasible and strong duality holds at the optimal primal-dual solution $(\boldsymbol{z}^*, \boldsymbol{\beta}^*, \boldsymbol{\lambda}^*)$. Then for any $\boldsymbol{z}$ with $\boldsymbol{A}^z\boldsymbol{z} = \boldsymbol{b} - \boldsymbol{A}^x\boldsymbol{x}^*$ and $(\boldsymbol{\beta}^*)^T\boldsymbol{z} \geq 0$, we have $\boldsymbol{c}^T\boldsymbol{z} \geq v_{\boldsymbol{x}^*}$.*

Proof.

$$(\boldsymbol{\beta}^*)^T\boldsymbol{z} = (\boldsymbol{c} - (\boldsymbol{A}^z)^T\boldsymbol{\lambda}^*)^T\boldsymbol{z} = \boldsymbol{c}^T\boldsymbol{z} - (\boldsymbol{\lambda}^*)^T(\boldsymbol{b} - \boldsymbol{A}^x\boldsymbol{x}^*) = \boldsymbol{c}^T\boldsymbol{z} - v_{\boldsymbol{x}^*} \geq 0. \tag{2.19}$$

□

Lemma 4. *Given $\boldsymbol{x}^*$, assume CONE($\boldsymbol{x}^*$) is infeasible and* (2.18) *is unbounded, such that we have a ray $(\boldsymbol{\beta}^*, \boldsymbol{\lambda}^*)$ satisfying $\boldsymbol{\beta}^* \in \mathcal{K}^*$, $\boldsymbol{\beta}^* = -(\boldsymbol{A}^z)^T\boldsymbol{\lambda}^*$, and $(\boldsymbol{\lambda}^*)^T(\boldsymbol{b} -$*

$\boldsymbol{A}^x\boldsymbol{x}^*) > 0$. *Then for any* $\boldsymbol{z}$ *satisfying* $\boldsymbol{A}^z\boldsymbol{z} = \boldsymbol{b} - \boldsymbol{A}^x\boldsymbol{x}^*$ *we have* $(\boldsymbol{\beta}^*)^T\boldsymbol{z} < 0$.

Proof.

$$(\boldsymbol{\beta}^*)^T\boldsymbol{z} = -(\boldsymbol{\lambda}^*)^T\boldsymbol{A}^z\boldsymbol{z} = -(\boldsymbol{\lambda}^*)^T(\boldsymbol{b} - \boldsymbol{A}^x\boldsymbol{x}^*) < 0. \tag{2.20}$$

□

We restate Algorithm 1 specialized for the conic case as Algorithm 2. In Chapter 3 we develop methods for initializing T, in particular so that MICONEOA(T) is bounded.

Algorithm 2 The conic polyhedral outer approximation (OA) algorithm

Initialize: $z_U \leftarrow \infty, z_L \leftarrow -\infty, T \leftarrow \emptyset$. Fix convergence tolerance ϵ.
while $z_U - z_L \geq \epsilon$ **do**
  Solve MICONEOA(T).
  if MICONEOA(T) is infeasible **then**
    MICONE is infeasible, so terminate.
  end if
  Let $(\boldsymbol{x}^*, \boldsymbol{z}')$ be the optimal solution of MICONEOA(T) with objective value w_T.
  Update lower bound $z_L \leftarrow w_T$.
  Solve CONE($\boldsymbol{x}^*$).
  if CONE($\boldsymbol{x}^*$) is feasible **then**
    Let $(\boldsymbol{z}^*, \boldsymbol{\beta}^*, \boldsymbol{\lambda}^*)$ be an optimal primal-dual solution with objective value $v_{\boldsymbol{x}^*}$.
    if $v_{\boldsymbol{x}^*} < z_U$ **then**
      $z_U \leftarrow v_{\boldsymbol{x}^*}$
      Record $(\boldsymbol{x}^*, \boldsymbol{z}^*)$ as the best known solution.
    end if
  else if CONE($\boldsymbol{x}^*$) is infeasible **then**
    Let $(\boldsymbol{\beta}^*, \boldsymbol{\lambda}^*)$ be a ray of (2.18).
  end if
  $T \leftarrow T \cup \{\boldsymbol{\beta}^*\}$
end while

In contrast to the OA algorithm developed by Drewes and Ulbrich [29] for the special case of MISOCP, Algorithm 2 is arguably much simpler because it is based on conic duality instead of subgradients and the KKT conditions. Drewes and Ulbrich also propose a second subproblem when infeasibility occurs even though rays of the dual would suffice.

2.5.1 Failures of outer approximation

When the assumption of strong duality fails, it may be impossible for the OA algorithm to converge in a finite number of iterations.

Consider the problem adapted from [46],

$$\begin{aligned} \min \quad & z \\ \text{s.t.} \quad & x = 0, \\ & (x, y, z) \in \mathcal{V}^3. \end{aligned} \tag{2.21}$$

Note that $(0, y, z) \in \mathcal{V}^3$ implies $z = 0$, so the optimal value is trivially zero.

The conic dual of this problem is

$$\begin{aligned} \max \quad & 0 \\ \text{s.t.} \quad & (\beta, 0, 1) \in \mathcal{V}^3, \\ & \beta \in \mathbb{R}. \end{aligned} \tag{2.22}$$

The dual is infeasible because there is no β satisfying $0\beta \geq 1$. So there is no strong duality in this case. The following lemma demonstrates that polyhedral approximations fail *entirely*. The proof uses only basic results from linear programming and conic duality.

Lemma 5. *There is no polyhedral outer approximation $P_{\mathcal{V}} \supset \mathcal{V}^3$ such that the following relaxation of* (2.21) *is bounded:*

$$\begin{aligned} \min \quad & z \\ \text{s.t.} \quad & x = 0, \\ & (x, y, z) \in P_{\mathcal{V}}. \end{aligned} \tag{2.23}$$

Proof. Let us assume that $\mathcal{V}^3 \subset P_{\mathcal{V}} := \{(x, y, z) : \boldsymbol{a}^x x + \boldsymbol{a}^y y + \boldsymbol{a}^z z \geq 0\}$ for some vectors $\boldsymbol{a}^x, \boldsymbol{a}^y, \boldsymbol{a}^z$. The right-hand side can be taken to be zero because $\mathcal{V}^3$ is a cone. Specifically, positive right-hand-side values are invalid because they would cut off the point $(0, 0, 0)$, and negative values can be strengthened to zero. Since (2.23) is a linear

programming problem invariant to positive rescaling, it is bounded iff there exists a feasible dual solution $(\beta, \boldsymbol{\alpha})$ satisfying

$$\boldsymbol{\alpha}^T \boldsymbol{a}^x = \beta, \tag{2.24}$$

$$\boldsymbol{\alpha}^T \boldsymbol{a}^y = 0, \tag{2.25}$$

$$\boldsymbol{\alpha}^T \boldsymbol{a}^z = 1, \tag{2.26}$$

$$\boldsymbol{\alpha} \geq \mathbf{0}. \tag{2.27}$$

Suppose, for contradiction, that there exist (β, α) satisfying these dual feasibility conditions. Let (a_i^x, a_i^y, a_i^z) denote the coefficients of the ith linear inequality in $P_{\mathcal{V}}$. Since $P_{\mathcal{V}}$ is a valid outer approximation, we have that

$$a_i^x x + a_i^y y + a_z^i z \geq 0, \, \forall (x, y, z) \in \mathcal{V}^3, \tag{2.28}$$

hence $(a_i^x, a_i^y, a_i^z) \in (\mathcal{V}^3)^* = \mathcal{V}^3$, recalling that $\mathcal{V}^3$ is self-dual. Therefore we have

$$(\boldsymbol{\alpha}^T \boldsymbol{a}^x, \boldsymbol{\alpha}^T \boldsymbol{a}^y, \boldsymbol{\alpha}^T \boldsymbol{a}^z) \in \mathcal{V}^3 \tag{2.29}$$

for $\boldsymbol{\alpha} \geq \mathbf{0}$. This follows from the fact that the vector, $(\boldsymbol{\alpha}^T \boldsymbol{a}^x, \boldsymbol{\alpha}^T \boldsymbol{a}^y, \boldsymbol{\alpha}^T \boldsymbol{a}^z)$, is a non-negative linear combination of elements of $\mathcal{V}^3$ and $\mathcal{V}^3$ is a convex cone. However, the duality conditions imply $(\beta, 0, 1) \in \mathcal{V}^3$, i.e., $0 \geq 1$, which is a contradiction. □

Lemma 5 implies that the following MISOCP instance cannot be solved by the OA algorithm:

$$\begin{aligned} \min \quad & z \\ \text{s.t.} \quad & x = 0, \\ & (x, y, z) \in \mathcal{V}^3, \\ & x \in \{0, 1\}, \end{aligned} \tag{2.30}$$

because the optimal value of *any* MILP relaxation will be $-\infty$ while the true optimal objective is 0, hence the convergence conditions cannot be satisfied.

This example strengthens the observation by [46] that MISOCP solvers may fail when certain constraint qualifications do not hold. In fact, no approach based on straightforward polyhedral approximation can succeed. Very recently, Gally et al. [35] have studied conditions in the context of mixed-integer semidefinite optimization which ensure that strong duality holds when integer values are fixed.

Another failure case is when we do not have finite bounds on the integer-constrained variables. Consider the problem

$$\min_{p,q,s} \quad s : \tag{2.31}$$

$$(p+q, s, 1) \in \mathcal{V}^3, \tag{2.32}$$

$$\frac{1}{2} - p \in \mathbb{R}_+, \tag{2.33}$$

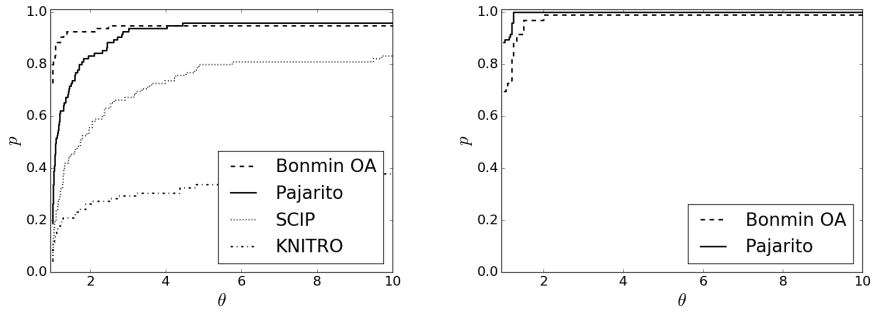
$$(p, q, s) \in \mathbb{Z}^3. \tag{2.34}$$

One may verify that the optimal value of the problem is 1. Yet, for any polyhedral outer approximation of $\mathcal{V}^3$, the solution $(0, q, 0)$ is feasible to the MILP relaxation for q sufficiently large. Therefore the optimal value of MICONEOA(T) is at most 0. Furthermore, strong duality for CONE(p, q, s) (in the form of either Lemma 3 or Lemma 4) holds for any $(p, q, s) \in \mathbb{Z}^3$.

2.6 Computational experiments

In this section we present numerical experiments with a prototype of Pajarito, an open-source solver for MICP publicly released at `https://github.com/JuliaOpt/Pajarito.jl`. The results here are based on Pajarito version 0.1. Subsequent to the work we report here, Pajarito has been almost completely rewritten with significant algorithmic advances; this will be described in Chapter 3.

We translated 194 convex instances of MINLPLIB2 [89] into Convex.jl [85], a DCP algebraic modeling language in Julia which performs automatic transformation into conic form. Our main points of comparison are Bonmin [14] using its OA algorithm, SCIP [2] using its default LP-based branch-and-cut algorithm, and Artelys



(a) Solution time

(b) Number of OA iterations

Figure 2-3: Comparison performance profiles [27] (solver performs within a factor of θ of the best on proportion p of instances) over all instances we tested from the MINLPLIB2 benchmark library. Higher is better. Bonmin is faster than Pajarito often within a small factor, yet Pajarito is able to solve a few more instances overall and with significantly fewer iterations.

Knitro [18] using its default nonlinear branch-and-bound algorithm; all three can be considered state-of-the-art academic or commercial solvers. We further compare our results with CPLEX for MISOCP instances only. All computations were performed on a high-performance cluster at Los Alamos National Laboratory with Intel® Xeon® E5-2687W v3 @3.10GHz 25.6MB L3 cache processors and 251GB DDR3 memory installed on every node. CPLEX v12.6.2 is used as a MILP and MISOCP solver. Because conic solvers supporting exponential cones were not sufficiently reliable in our initial experiments, we use Artelys Knitro v9.1.0 to solve all conic subproblems via traditional derivative-based methods.

Bonmin v1.8.3 and SCIP v3.2.0 are both compiled with CPLEX v12.6.2 and Ipopt v3.12.3 using the HSL linear algebra library MA97. All solvers are set to a relative optimality gap of 10^{-5}, are run on a single thread (both CPLEX and Artelys Knitro), and are given 10 hours of wall time limit (with the exception of `gams01`, a previously unsolved benchmark instance, where we give 32 threads to CPLEX for the MILP relaxations). The scripts to run these experiments can be found online at `https://github.com/mlubin/MICPExperiments`.

Numerical experiments indicate that the extended formulation drastically reduces the number of polyhedral OA iterations as expected. In aggregate across the instances

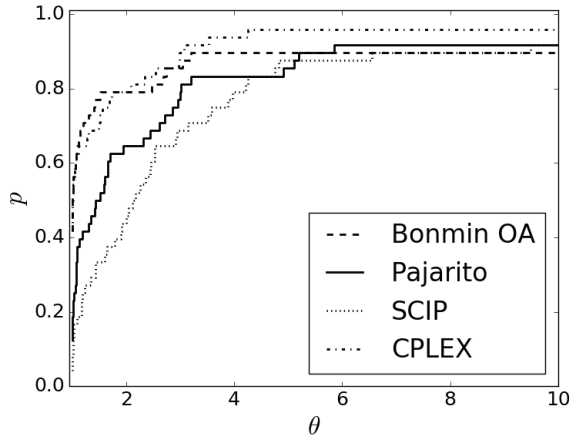


Figure 2-4: Performance profile [27] (solver is the fastest within a factor of θ of the best on proportion p of instances) over the instances representable as mixed-integer second-order cone problems where we can compare with the commercial CPLEX solver. Higher is better. CPLEX is the best overall, since notably it already implements the extended formulation proposed by Vielma et al. [88].

we tested, Bonmin requires 2685 iterations while Pajarito requires 994. We list the full results in Table A.1 and summarize them in Figure 2-3. In Figure 2-4 we present results for the subset of SOC-representable instances, where we can compare with commercial MISOCP solvers. In our performance profiles, all times are shifted by 10 seconds to decrease the influence of easy instances, and iteration counts are similarly shifted by 2.

Notably, Pajarito is able solve a previously unsolved instance, `gams01`, whose conic representation requires a mix of SOC and EXP cones and hence was not a pure MISOCP problem. The best known bound was 1735.06 and the best known solution was 21516.83. Pajarito solved the instance to optimality with an objective value of 21380.20 in 6 iterations.

Chapter 3

A conic framework for solving mixed-integer convex problems via outer approximation

3.1 Mixed-integer conic form and outer approximation

3.1.1 Mixed-integer conic (MICONE) general form $\mathfrak{M}$

We denote an MICONE problem in general form as $\mathfrak{M}$:

$$\mathfrak{M} \equiv \min_{\boldsymbol{x}\in\mathbb{R}^N} \quad \langle \boldsymbol{c}, \boldsymbol{x}\rangle : \tag{3.1}$$

$$\boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!]\,, \tag{3.2}$$

$$x_i \in \mathbb{Z}, \qquad \forall i \in [\![I]\!]\,. \tag{3.3}$$

We assume throughout that if $\mathfrak{M}$ is feasible, then its optimal value is attained, hence we write min instead of inf in the objective. In this chapter we use the notation $\langle\cdot,\cdot\rangle$ for the standard inner product of two vectors. The objective (3.1) therefore minimizes $\langle \boldsymbol{c}, \boldsymbol{x}\rangle = \sum_{i\in[\![N]\!]} c_i x_i$ over $\boldsymbol{x} \in \mathbb{R}^N$, subject to (denoted by ':') the M conic

constraints (3.2) and the I integrality constraints (3.3).

The conic constraints (3.2) express that for each $k \in [\![M]\!]$, the vector-valued affine transformation $\boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x} \in \mathbb{R}^{n_k}$ of $\boldsymbol{x} \in \mathbb{R}^N$ is restricted to belong to the cone $\mathcal{C}_k \subset \mathbb{R}^{n_k}$. The cone $\mathcal{C}_k$ must be a closed and convex cone. Definitions and example cones will be given in Section 3.1.2. For consistency, affine constraints that would normally be expressed with equality and inequality notation are instead encoded conically using the basic polyhedral cones $\mathbb{R}^n_+ = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} \geq \boldsymbol{0}\}$ (nonnegative cone), $\mathbb{R}_- = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} \leq \boldsymbol{0}\}$ (nonpositive cone), or $\{0\}^n$ (zero cone).

The set of integers is denoted $\mathbb{Z}$, so the integrality constraints (3.3) restrict the variables $x_1, \ldots, x_I$ to integer values. The variables $x_{I+1}, \ldots, x_N$ are continuous-valued. If $I = 0$ (no integrality constraints), $\mathfrak{M}$ is a convex optimization problem because it minimizes the linear objective (3.1) over continuous variables $\boldsymbol{x}$ subject to convex constraints (3.2).

The form $\mathfrak{M}$ is also general. A maximization problem may be converted to a minimization problem. Variable bounds on $\boldsymbol{x}$ can be incorporated into the conic constraints using $\mathbb{R}_+$ or $\mathbb{R}_-$. Binary restrictions on variables can be encoded using both integer restrictions and variable bounds. $\mathfrak{M}$ expresses established problem classes such as MILP, MISOCP, and MISDP.

3.1.2 Four versatile cones for convex nonlinear modeling

We have already mentioned the basic polyhedral cones, $\mathbb{R}_+$, $\mathbb{R}_-$, and $\{0\}$. Any mixed-integer linear program (MILP) can be written in the form $\mathfrak{M}$ using these basic polyhedral cones. Beyond these polyhedral cones, we reintroduce (from Chapter 2) several additional cones that are particularly versatile for convex nonlinear optimization modeling. We often omit the dimension parameter of the cone when it is implied by the context.

The second-order $\mathcal{L}$ and rotated second-order $\mathcal{V}$ Cones

The second-order cone is useful for modeling the ℓ_2-norm and many other convex functions; see [6]. It is typically defined as:

$$\mathcal{L}^{1+n} = \left\{(r, \boldsymbol{t}) \in \mathbb{R}^{1+n} : r \geq \|\boldsymbol{t}\|_2\right\}. \tag{3.4}$$

The rotated second-order cone is useful for modeling functions like x^2 and $\sqrt{x_1 x_2}$; see [6]:

$$\mathcal{V}^{2+n} = \left\{(r, s, \boldsymbol{t}) \in \mathbb{R}^{2+n} : r, s \geq 0, 2rs \geq \|\boldsymbol{t}\|_2^2\right\}. \tag{3.5}$$

As its name suggests, the rotated second-order cone $\mathcal{V}^{2+n}$ is an invertible linear transformation of the second-order cone $\mathcal{L}^{2+n}$ [71]:

$$(r, s, \boldsymbol{t}) \in \mathcal{V}^{2+n} \Leftrightarrow \left(\frac{r+s}{\sqrt{2}}, \frac{r-s}{\sqrt{2}}, \boldsymbol{t}\right) \in \mathcal{L}^{2+n}. \tag{3.6}$$

Thus for $\mathfrak{M}$ we can restrict attention to the second-order cone by applying this transformation to any rotated second-order cone constraint $\boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x} \in \mathcal{V}$. However, we will use the definition of $\mathcal{V}$ again when we address disaggregation in Section 3.5.3 and $\mathcal{V}$ cuts for outer approximation in Section 3.5.2.

Positive semidefinite cone in "svec" $\mathcal{P}$ and "smat" $\mathbb{S}_+$ forms

Modeling with the positive semidefinite (PSD) matrix cone is described extensively in [6]. This cone is typically thought of as a subset of the space of either square matrices $\mathbb{R}^{n\times n}$ or symmetric matrices $\mathbb{S}^n = \left\{\boldsymbol{T} \in \mathbb{R}^{n\times n} : \boldsymbol{T} = \boldsymbol{T}^T\right\}$. We use the latter convention for good reason: in the space of symmetric matrices, we do not need to enforce symmetry constraints, and the cone has effective dimension $\frac{1}{2}n(n+1)$ rather than n^2. The matrix cone in symmetric space can be defined in multiple equivalent

ways, for example using the minimum eigenvalue function $\lambda_{\min}$:

$$\mathbb{S}^n_+ = \{\boldsymbol{T} \in \mathbb{S}^n : \lambda_{\min}(\boldsymbol{T}) \geq 0\}. \tag{3.7}$$

This symmetric space definition is called the 'smat' PSD cone and we write $\boldsymbol{T} \in \mathbb{S}^n_+ \subset \mathbb{S}^n$ and use the standard matrix/trace inner product $\langle \boldsymbol{W}, \boldsymbol{T} \rangle = \sum_{i,j \in [n]} W_{ij} T_{ij}$. The equivalent vector space definition is called the 'svec' PSD cone and we write $\boldsymbol{t} \in \mathcal{P}^{\frac{1}{2}n(n+1)} \subset \mathbb{R}^{\frac{1}{2}n(n+1)}$ and use the vector inner product as described in Section 3.1.1. The transformation is described by [71] as follows, for $\boldsymbol{T} \in \mathbb{S}^n$ and $\boldsymbol{t} \in \mathbb{R}^{\frac{1}{2}n(n+1)}$:

$$\operatorname{svec}(\boldsymbol{T}) = \left(T_{1,1}, \sqrt{2}T_{2,1}, \ldots, \sqrt{2}T_{n,1}, T_{2,2}, \sqrt{2}T_{3,2}, \ldots, T_{n,n}\right) \tag{3.8}$$

$$\operatorname{smat}(\boldsymbol{t}) = \begin{bmatrix} t_1 & \frac{t_1}{\sqrt{2}} & \cdots & \frac{t_n}{\sqrt{2}} \\ \frac{t_2}{\sqrt{2}} & t_{n+1} & \cdots & \frac{t_{2n-1}}{\sqrt{2}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{t_n}{\sqrt{2}} & \frac{t_{n-1}}{\sqrt{2}} & \cdots & t_{\frac{1}{2}n(n+1)} \end{bmatrix} \tag{3.9}$$

It is often convenient to treat the PSD cone uniformly with other vector cones, so in the general form $\mathfrak{M}$ so we use the svec PSD cone $\mathcal{P}$. We convert from svec space to smat space when notationally appropriate, such as when we want an eigendecomposition.

The Exponential Cone $\mathcal{E}$

We use the following definition of the exponential cone:

$$\mathcal{E} = \operatorname{cl}\left(\left\{(r,s,t) \in \mathbb{R}^3 : s > 0, r \geq s \exp\left(\frac{t}{s}\right)\right\}\right) \tag{3.10}$$

$$= \{(r,0,t) : r \geq 0, t \leq 0\} \cup \left\{(r,s,t) : s > 0, r \geq s \exp\left(\frac{t}{s}\right)\right\}. \tag{3.11}$$

This cone can be used to model functions such as e^x (used interchangeably with $\exp(x)$) and $-\log(x)$, the negative entropy function $-x\log(x)$, and the log-sum-exp

function; see [80]. Chandrasekaran and Shah [20] recently showed that the $\mathcal{L}$ cone is representable by using the $\mathcal{E}$ cone, and hence one could argue that the exponential cone is more general than the second-order cone.

3.1.3 Outer approximation using dual cones

We assume the first K of M cones are nonlinear and the remaining are basic polyhedral cones. We now recall how to outer approximate the nonlinear cones using polyhedra.

We begin with the definition of the dual cone $\mathcal{K}^* \subset \mathbb{R}^n$ of a cone $\mathcal{K} \subset \mathbb{R}^n$:

$$\mathcal{K}^* = \{\boldsymbol{z} \in \mathbb{R}^n : \langle \boldsymbol{y}, \boldsymbol{z} \rangle \geq 0, \forall \boldsymbol{y} \in \mathcal{K}\}. \tag{3.12}$$

The dual cone of a product of cones $\mathcal{K}_1 \times \mathcal{K}_2$ is the product of the individual dual cones $\mathcal{K}_1^* \times \mathcal{K}_2^*$. An important property of dual cones is that if $\mathcal{K}$ is closed and convex then so is $\mathcal{K}^*$, and the following equivalence holds,

$$\boldsymbol{y} \in \mathcal{K} \Leftrightarrow \langle \boldsymbol{y}, \boldsymbol{z} \rangle \geq 0, \qquad \forall \boldsymbol{z} \in \mathcal{K}^*. \tag{3.13}$$

Often we will denote a vector of primal cone variables by $\boldsymbol{y}^k = \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k$, and a vector of dual cone variables by $\boldsymbol{z}^k \in \mathcal{C}_k^*$. All of the example cones we offered, except for the exponential cone $\mathcal{E}$, are 'self-dual', meaning that $\mathcal{K}^* = \mathcal{K}$. For $\mathcal{L}$, we often write the dual variables as $(u, \boldsymbol{w}) \in \mathcal{L}^* = \mathcal{L}$, and for $\mathcal{V}$, we write $(u, v, \boldsymbol{w}) \in \mathcal{V}^* = \mathcal{V}$. For the smat PSD cone, we write a symmetric matrix of dual variables as $\boldsymbol{W} \in (\mathbb{S}_+)^* = \mathbb{S}_+$, and for the svec PSD cone, we write a vector of dual variables as $\boldsymbol{w} \in \mathcal{P}^* = \mathcal{P}$. For the exponential cone, the dual cone can be defined by:

$$\mathcal{E}^* = \mathrm{cl}\left\{(u, v, w) \in \mathbb{R}^3 : w < 0, u \geq -w \exp\left(\frac{v}{w} - 1\right)\right\} \tag{3.14}$$

$$= \{(u, v, 0) : u, v \geq 0\} \cup \left\{(u, v, w) : w < 0, u \geq -w \exp\left(\frac{v}{w} - 1\right)\right\}. \tag{3.15}$$

For some $k \in [\![K]\!]$, consider choosing a finite subset of the points in the dual

cone and denote this subset $\mathcal{Z}_k \subset \mathcal{C}_k^*$. We can then construct a valid polyhedral outer approximation of $\mathcal{C}_k$ by enforcing the linear constraints corresponding to $\boldsymbol{z} \in \mathcal{Z}_k$, i.e.,

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z} \right\rangle \in \mathbb{R}_+, \qquad \forall \boldsymbol{z} \in \mathcal{Z}_k. \tag{3.16}$$

Definition 3. *For a given $\boldsymbol{z} \in \mathcal{C}_k^*$, we call a constraint of the form* (3.16) *a $\mathcal{K}^*$ cut induced by $\boldsymbol{z}$.*

In the next section we investigate how these cuts may be chosen.

3.2 Four simple classes of $\mathcal{K}^*$ cuts

3.2.1 Initial fixed $\mathcal{K}^*$ cuts

In this section we propose initial fixed $\mathcal{K}^*$ cuts which depend only on the dimensions n_k of the cones $\mathcal{C}_k, k \in [\![K]\!]$. Our goal here is not to define theoretically efficient initial outer approximations, but instead to show that well-known outer approximations can be interpreted in the framework of $\mathcal{K}^*$ cuts.

Initial Fixed $\mathcal{L}^*$ Cuts

Ben-Tal and Nemirovski [9] describe polyhedral outer approximations of $\mathcal{L}$ with provably good approximation guarantees. However, this family of polyhedral outer approximations requires additional auxiliary variables in a way that makes it difficult to refine dynamically [87], so we do not consider them. The two simple classes of cuts we consider are the variable bound $r \geq 0$ which is a $\mathcal{K}^*$ cut because $(1, \mathbf{0}) \in \mathcal{L}^*$ and secondly, from the inequality

$$\|\boldsymbol{t}\|_\infty = \max_{i \in [\![n]\!]} |t_i| \geq \sqrt{\sum_{i \in [\![n]\!]} t_i^2} = \|\boldsymbol{t}\|_2, \tag{3.17}$$

we see that $\|\boldsymbol{t}\|_\infty \leq r$ is a valid polyhedral outer approximation of $\mathcal{L}$. We can enforce this initial polyhedral outer approximation with a set of $2n$ initial fixed $\mathcal{L}^*$ cuts:

$$(1, \pm \boldsymbol{e}(i)) \in \mathcal{L}^*, \qquad \forall i \in [\![n]\!], \tag{3.18}$$

which impose the conditions $r \geq |t_i|, \forall i \in [\![n]\!]$.

Initial Fixed $\mathcal{P}^*$ Cuts

We impose the variable bounds that ensure the diagonal elements of $\boldsymbol{T} \in \mathbb{S}^n$ are nonnegative: $T_{i,i} \in \mathbb{R}_+, i \in [\![n]\!]$ (these are again $\mathcal{K}^*$ cuts).

Ahmadi and Hall [3] discuss polyhedral outer approximations of the PSD cone. They define the cone of diagonally dominant matrices, an important subset of the PSD cone, and use its dual cone as a fixed outer approximation of the PSD cone. This dual cone is polyhedral and has $n(n-1)$ extreme rays. We add a $\mathcal{K}^*$ cut for each such extreme ray:

$$(\boldsymbol{e}(i) + \boldsymbol{e}(j))(\boldsymbol{e}(i) + \boldsymbol{e}(j))^T, (\boldsymbol{e}(i) - \boldsymbol{e}(j))(\boldsymbol{e}(i) - \boldsymbol{e}(j))^T \in \left(\mathbb{S}^n_+\right)^*, \quad \forall i, j \in [\![n]\!] : i > j. \tag{3.19}$$

Initial Fixed $\mathcal{E}^*$ Cuts

We first impose the variable bounds $r, s \geq 0$ that are $\mathcal{K}^*$ cuts because $(1, 0, 0) \in \mathcal{E}^*$ and $(0, 1, 0) \in \mathcal{E}^*$. We then pick points from $\mathcal{E}^*$ as follows.

Fix $w = -1$ and consider the function $\exp(-v - 1)$. For the variable v, choose a set of L linearization points $\hat{v}_l \in \mathbb{R}, l \in [\![L]\!]$ and obtain the L $\mathcal{K}^*$ cuts defined by:

$$(\exp(-\hat{v}_l - 1), \hat{v}_l, -1) \in \mathcal{E}^*, \qquad \forall l \in [\![L]\!]. \tag{3.20}$$

The linearization points $\hat{v}_l, l \in [\![L]\!]$ could be chosen in various ways, for example, according to the curvature of the function $\exp(-v - 1)$ in order to guarantee a worst case "distance" from the $\mathcal{E}$ cone of any point feasible for the initial fixed polyhedral outer approximation.

3.2.2 $\mathcal{K}^*$ cuts from the continuous relaxation

We present here a family of cuts which, like the initial fixed cuts, can be computed before solving a finite outer approximation model. Unlike the initial fixed cuts, however, these cuts can be shown to provide a guarantee on the objective value of the finite outer approximation model. The cuts and corresponding proofs make use of conic duality which we now briefly review; see [6, 17] for a fuller treatment.

Consider relaxing the integrality constraints of $\mathfrak{M}$ to get the continuous relaxation $\mathfrak{R}$:

$$\mathfrak{R} \equiv \min_{\boldsymbol{x}} \quad \langle \boldsymbol{c}, \boldsymbol{x} \rangle : \tag{3.21}$$

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!], \tag{3.22}$$

$$\boldsymbol{x} \in \mathbb{R}^N. \tag{3.23}$$

The standard conic dual of $\mathfrak{R}$ is $\mathfrak{R}^*$:

$$\mathfrak{R}^* \equiv \max_{(\boldsymbol{z}^k)_{k \in [\![M]\!]}} \quad -\sum_{k \in [\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle : \tag{3.24}$$

$$\boldsymbol{c} + \sum_{k \in [\![M]\!]} (\boldsymbol{A}^k)^T \boldsymbol{z}^k \in \{0\}^N, \tag{3.25}$$

$$\boldsymbol{z}^k \in \mathcal{C}_k^*, \qquad \forall k \in [\![M]\!]. \tag{3.26}$$

Friberg [34] provides a mutually exclusive and exhaustive list of five possible statuses that may result from a conic primal-dual pair, all of which can be verified by simple certificates. For the discussions that follow, we will assume for any conic problem we consider that either (i) there exists an optimal primal-dual pair, i.e., a pair of solutions feasible to $\mathfrak{R}$ and $\mathfrak{R}^*$ with matching objective values, or (ii) the primal problem is infeasible and there exists a ray of the dual which certifies infeasibility.

For $\mathfrak{R}$ this infeasibility certificate has the form:

$$-\sum_{k\in[\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle < 0, \tag{3.27}$$

$$\sum_{k\in[\![M]\!]} (\boldsymbol{A}^k)^T \boldsymbol{z}^k \in \{0\}^N, \tag{3.28}$$

$$\boldsymbol{z}^k \in \mathcal{C}_k^*, \qquad \forall k \in [\![M]\!]\,. \tag{3.29}$$

For simplicity, we assume that the relaxation $\mathfrak{R}$ is bounded and attains an optimal solution, and hence we can discard the possibility of a certificate of dual infeasibility. It is also possible for there to be a positive duality gap between optimal solutions of the primal and dual; we assume this is not the case as our methods can fail otherwise (see Section 2.5.1). We proceed to the derivation of the cut.

The following lemma shows that the dual solutions to $\mathfrak{R}$ serve as $\mathcal{K}^*$ cuts which imply optimality guarantees on $\boldsymbol{x}$. Indeed, these cuts imply that a mixed-integer outer approximation problem can obtain an objective value no worse than the continuous relaxation $\mathfrak{R}$.

Lemma 6. *Suppose strong duality holds for the primal-dual pair $\mathfrak{R}$ and $\mathfrak{R}^*$, and that $\mathfrak{R}^*$ attains an optimal solution $(\boldsymbol{z}^k)_{k\in[\![M]\!]}$. Let C be the optimal objective value (of both the primal and dual). Suppose $\boldsymbol{x} \in \mathbb{R}^N$ satisfies:*

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x}, \boldsymbol{z}^k \right\rangle \in \mathbb{R}_+, \qquad \forall k \in [\![M]\!]\,. \tag{3.30}$$

Then $\langle \boldsymbol{c}, \boldsymbol{x}\rangle \geq C$.

Proof.

$$\langle \boldsymbol{c}, \boldsymbol{x} \rangle - C \tag{3.31}$$

$$= \langle \boldsymbol{c}, \boldsymbol{x} \rangle + \sum_{k \in [\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle \tag{3.32}$$

$$= \sum_{k \in [\![M]\!]} \left\langle \boldsymbol{x}, -(\boldsymbol{A}^k)^T \boldsymbol{z}^k \right\rangle + \sum_{k \in [\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle \tag{3.33}$$

$$= \sum_{k \in [\![M]\!]} \left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle \tag{3.34}$$

$$\geq 0. \tag{3.35}$$

□

These cuts derived from the continuous relaxation $\mathfrak{R}$ are analogous to those used to initialize the outer approximation algorithm in BONMIN [14] for convex MINLP. We believe that Lemma 6 is the first statement of these cuts in the context of mixed-integer conic optimization. Recall that in our work in Chapter 2 we did not derive methods for initializing the outer approximation.

3.2.3 $\mathcal{K}^*$ cuts from continuous subproblems

In Chapter 2 we proposed cuts that can be computed by solving a continuous conic subproblem where all integer-restricted variables are fixed to a particular solution. By considering more general box restrictions, we can make some new observations. Both SCIP and Gurobi solve SOCP restrictions during their branch-and-bound implementations as heuristics to find feasible solutions; however, the dual information from these subproblems is completely discarded[1]. The $\mathcal{K}^*$ cuts from the dual solution to such conic restrictions are globally valid and, if added to the LP relaxation, contain enough information to properly process a branch-and-bound node. See also [87].

Consider relaxing the integer constraints of $\mathfrak{M}$ and restricting the integer-constrained

[1]Personal communication with Zonghao Gu of Gurobi and Felipe Serrano of ZIB

variables to instead be contained in a box defined by vectors $(\boldsymbol{l}, \boldsymbol{u})$:

$$\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u}) \equiv \min_{\boldsymbol{x}} \quad \langle \boldsymbol{c}, \boldsymbol{x} \rangle : \tag{3.36}$$

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!], \tag{3.37}$$

$$x_i \in [l_i, u_i], \qquad \forall i \in [\![I]\!], \tag{3.38}$$

$$\boldsymbol{x} \in \mathbb{R}^N. \tag{3.39}$$

After encoding the box constraints using the $\mathbb{R}_+$ cone, the conic dual of $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$ is $\mathfrak{R}^*(\boldsymbol{l}, \boldsymbol{u})$:

$$\mathfrak{R}^*(\boldsymbol{l}, \boldsymbol{u}) \equiv \max_{\boldsymbol{z}^1, \ldots, \boldsymbol{z}^K, \boldsymbol{\alpha}, \boldsymbol{\beta}} \quad \sum_{i \in [\![I]\!]} (l_i \alpha_i - u_i \beta_i) - \sum_{k \in [\![M]\!]} \langle \boldsymbol{b}_k, \boldsymbol{z}_k \rangle : \tag{3.40}$$

$$\boldsymbol{c} + \sum_{i \in [\![I]\!]} (\beta_i - \alpha_i) \boldsymbol{e}(i) + \sum_{k \in [\![M]\!]} (\boldsymbol{A}^k)^T \boldsymbol{z}^k \in \{0\}^N, \tag{3.41}$$

$$\boldsymbol{z}^k \in \mathcal{C}_k^*, \qquad \forall k \in [\![M]\!], \tag{3.42}$$

$$\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{R}_+^I. \tag{3.43}$$

Optimality Subproblem $\mathcal{K}^*$ Cuts

The following lemma shows that if there is an optimal solution and strong duality holds for this primal-dual pair, the dual solution corresponds to $\mathcal{K}^*$ cuts that enforce an objective bound over the box defined by the vectors $(\boldsymbol{l}, \boldsymbol{u})$.

Lemma 7. *Suppose strong duality holds and the dual $\mathfrak{R}^*(\boldsymbol{l}, \boldsymbol{u})$ attains an optimal solution $\left(\boldsymbol{z}^1, \ldots, \boldsymbol{z}^K, \boldsymbol{\alpha}, \boldsymbol{\beta}\right)$. Let C be the optimal objective value (of both the primal and dual). Suppose $\boldsymbol{x} \in \mathbb{R}^N$ satisfies:*

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!] \setminus [\![K]\!], \tag{3.44}$$

$$x_i \in [l_i, u_i], \qquad \forall i \in [\![I]\!], \tag{3.45}$$

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle \in \mathbb{R}_+, \qquad \forall k \in [\![K]\!]. \tag{3.46}$$

Then $\langle \boldsymbol{c}, \boldsymbol{x} \rangle \geq C$.

Proof. This follows from Lemma 6 applied to $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$ and observing that $\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k$ for $k \in [\![M]\!] \setminus [\![K]\!]$ implies $\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle \in \mathbb{R}_+$ for $k \in [\![M]\!] \setminus [\![K]\!]$ and $x_i \in [l_i, u_i]$ implies $(-l_i + x_i)\alpha_i \in \mathbb{R}_+$ and $(u_i - x_i)\beta_i \in \mathbb{R}_+$. □

Feasibility Subproblem $\mathcal{K}^*$ Cuts

Suppose instead we have a certificate of infeasibility for $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$. Then that certificate yields $\mathcal{K}^*$ cuts that exclude all solutions when the integer variables are constrained to be in the box defined by $(\boldsymbol{l}, \boldsymbol{u})$.

Lemma 8. *Suppose* $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$ *is infeasible and we have a ray* $\left((\boldsymbol{z}^k)_{k \in [\![M]\!]}, \boldsymbol{\alpha}, \boldsymbol{\beta}\right)$ *satisfying:*

$$\sum_{i \in [\![I]\!]} (\beta_i - \alpha_i) \boldsymbol{e}(i) + \sum_{k \in [\![K]\!]} (\boldsymbol{A}^k)^T \boldsymbol{z}^k \in \{0\}^N, \tag{3.47}$$

$$\sum_{i \in [\![I]\!]} (u_i \beta_i - l_i \alpha_i) + \sum_{k \in [\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle < 0. \tag{3.48}$$

Then for all $\boldsymbol{x} \in \mathbb{R}^N$ *satisfying:*

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!] \setminus [\![K]\!], \tag{3.49}$$

$$x_i \in [l_i, u_i], \qquad \forall i \in [\![I]\!], \tag{3.50}$$

there exists a $k \in [\![K]\!]$ *such that* $\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle < 0$.

Proof.

$$\sum_{k\in[\![K]\!]} \left\langle \boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x}, \boldsymbol{z}^k \right\rangle \tag{3.51}$$

$$\leq \sum_{k\in[\![M]\!]} \left\langle \boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x}, \boldsymbol{z}^k \right\rangle + \sum_{i\in[\![I]\!]} (-l_i + x_i)\alpha_i + \sum_{i\in[\![I]\!]} (u_i - x_i)\beta_i \tag{3.52}$$

$$= \left\langle \boldsymbol{x}, \sum_{i\in[\![I]\!]} (\alpha_i - \beta_i)\boldsymbol{e}(i) - \sum_{k\in[\![M]\!]} (\boldsymbol{A}^k)^T\boldsymbol{z}^k \right\rangle + \sum_{k\in[\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle + \sum_{i\in[\![I]\!]} (u_i\beta_i - l_i\alpha_i) \tag{3.53}$$

$$= \sum_{k\in[\![M]\!]} \left\langle \boldsymbol{b}^k, \boldsymbol{z}^k \right\rangle + \sum_{i\in[\![I]\!]} (u_i\beta_i - l_i\alpha_i) \tag{3.54}$$

$$< 0. \tag{3.55}$$

Hence, one of the terms in the sum (3.51) must be negative. The first inequality follows from (3.49)-(3.49). □

3.2.4 Separation $\mathcal{K}^*$ cuts

Given a solution $\boldsymbol{x}$ for which $\boldsymbol{b}^k - \boldsymbol{A}^k\boldsymbol{x} \notin \mathcal{C}_k$ for some $k \in [\![K]\!]$, it is natural to attempt to generate new $\mathcal{K}^*$ cuts that exclude this solution to refine an existing outer approximation. Iterating this procedure (until the violations become sufficiently small) in the context of convex optimization is known generally as Kelley's cutting-plane method [50], the essence of which remains in use by state-of-the-art methods, e.g., [10] for MISOCP. In this section we show how to derive cuts which separate an infeasible solution from the set of feasible solutions, hence we call them separation $\mathcal{K}^*$ cuts. Although these cuts are not required in the algorithms we propose, they remain important for the purpose of comparison.

We begin with a trivial equivalence between separating hyperplanes and $\mathcal{K}^*$ cuts, which shows that any procedure that generates separating hyperplanes also generates separation $\mathcal{K}^*$ cuts.

Lemma 9. *Let $\mathcal{K}$ be a nonempty closed convex cone and let $\boldsymbol{y} \notin \mathcal{K}$. Suppose $(\boldsymbol{z}, \theta)$ defines a hyperplane that separates $\boldsymbol{y}$ from $\mathcal{K}$, i.e., $\langle \boldsymbol{y}, \boldsymbol{z}\rangle < \theta$ and $\langle \hat{\boldsymbol{y}}, \boldsymbol{z}\rangle \geq \theta, \forall \hat{\boldsymbol{y}} \in \mathcal{K}$.*

Then $\boldsymbol{z} \in \mathcal{K}^*$ and $\boldsymbol{z}$ induces a $\mathcal{K}^*$ cut that separates $\boldsymbol{y}$ from $\mathcal{K}$.

Proof. The optimal value of $\min_{\hat{\boldsymbol{y}} \in \mathcal{K}} \langle \boldsymbol{z}, \hat{\boldsymbol{y}} \rangle$ is bounded below by θ and therefore must be equal to zero (hence $\theta \leq 0$) since it is a homogeneous problem. Therefore $\langle \hat{\boldsymbol{y}}, \boldsymbol{z} \rangle \geq 0, \forall \hat{\boldsymbol{y}} \in \mathcal{K}$ so $\boldsymbol{z} \in \mathcal{K}^*$ by definition of $\mathcal{K}^*$ and $\langle \boldsymbol{y}, \boldsymbol{z} \rangle < 0$ by $\theta \leq 0$. □

More concretely, the specific closed convex cones which we offer as useful examples are all defined by $\mathcal{K} = \{\boldsymbol{y} : f(\boldsymbol{y}) \leq 0\}$ for some *positively homogeneous* convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ whose values and subgradients we know how to compute. (A positively homogeneous function satisfies $f(\alpha \boldsymbol{y}) = \alpha f(\boldsymbol{y})$ for $\alpha \geq 0$.) The following lemma shows that subgradients correspond to separation $\mathcal{K}^*$ cuts whenever $0 < f(\boldsymbol{y}) < \infty$.

Lemma 10. *Let $\mathcal{K}$ be a nonempty closed convex cone defined by $\mathcal{K} = \{\boldsymbol{y} : f(\boldsymbol{y}) \leq 0\}$ where $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is a homogeneous convex function. Then subgradients of f correspond to $\mathcal{K}^*$ cuts. Furthermore, when $\boldsymbol{y} \notin \mathcal{K}$ and $f(\boldsymbol{y}) < \infty$, a subgradient of f at $\boldsymbol{y}$ corresponds to a $\mathcal{K}^*$ cut which separates $\boldsymbol{y}$ from $\mathcal{K}$.*

Proof. Let $\boldsymbol{z}$ be a subgradient of f at some point $\boldsymbol{y}$ with $f(\boldsymbol{y})$ (For simplicity we have assumed $f(\boldsymbol{y}) < \infty$.) By definition of subgradient [6]:

$$f(\hat{\boldsymbol{y}}) \geq f(\boldsymbol{y}) + \langle \hat{\boldsymbol{y}} - \boldsymbol{y}, \boldsymbol{z} \rangle, \qquad \forall \hat{\boldsymbol{y}} \in \mathbb{R}^n. \tag{3.56}$$

We first claim that $f(\boldsymbol{y}) - \langle \boldsymbol{y}, \boldsymbol{z} \rangle = 0$. Rearranging the subgradient inequality (3.56), we obtain:

$$\langle \hat{\boldsymbol{y}}, \boldsymbol{z} \rangle - f(\hat{\boldsymbol{y}}) \leq \langle \boldsymbol{y}, \boldsymbol{z} \rangle - f(\boldsymbol{y}), \qquad \forall \hat{\boldsymbol{y}} \in \mathbb{R}^n. \tag{3.57}$$

Since the left-hand side is homogeneous, we can substitute $\alpha \boldsymbol{y}$ for $\hat{\boldsymbol{y}}$ to see that any solution except $\langle \boldsymbol{y}, \boldsymbol{z} \rangle - f(\boldsymbol{y}) = 0$ would lead to a contradiction for some choice of $\alpha \geq 0$.

Hence, the subgradient inequality reduces to:

$$f(\hat{\boldsymbol{y}}) \geq \langle \hat{\boldsymbol{y}}, \boldsymbol{z} \rangle, \qquad \forall \hat{\boldsymbol{y}} \in \mathbb{R}^n. \tag{3.58}$$

Then $\hat{\boldsymbol{y}} \in \mathcal{K}$ implies $\langle \hat{\boldsymbol{y}}, \boldsymbol{z} \rangle \leq 0$, and so $-\boldsymbol{z} \in \mathcal{K}^*$. Specifically, $\langle \boldsymbol{y}, -\boldsymbol{z} \rangle \geq 0$ is a $\mathcal{K}^*$ cut that separates $\boldsymbol{y}$ from $\mathcal{K}$ when $f(\boldsymbol{y}) > 0$ and $\boldsymbol{z}$ is a subgradient of f at $\boldsymbol{y}$. □

We proceed to give explicit formulas for separation $\mathcal{K}^*$ cuts for the cones we consider, including a case where we need special treatment because the homogeneous function can take the value ∞.

Separation Cuts for $\mathcal{L}$

The second-order cone $\mathcal{L}$ is defined by the homogeneous convex function $f(r, \boldsymbol{t}) = \|\boldsymbol{t}\|_2 - r$. Assuming $r \geq 0$ is enforced by initial cuts, $f(r, \boldsymbol{t}) > 0$ implies $\|\boldsymbol{t}\|_2 > 0$, hence applying Lemma 10 we obtain the separation $\mathcal{K}^*$ cut:

$$\left(1, \frac{-\boldsymbol{t}}{\|\boldsymbol{t}\|_2}\right) \in \mathcal{L}^*. \tag{3.59}$$

These separation cuts are commonly used across MISOCP solvers [10].

Separation Cuts for $\mathcal{P}$

Suppose we have a solution $\boldsymbol{t} \notin \mathcal{P}$. It is convenient to switch to smat space, so let $\boldsymbol{T} = \text{smat}(\boldsymbol{t}) \in \mathbb{S}^n$. $\boldsymbol{T} \notin \mathbb{S}^n_+$ implies $\lambda_{\min}(\boldsymbol{T}) < 0$. Let $\boldsymbol{\tau}$ be an eigenvector corresponding to the smallest eigenvalue of $\boldsymbol{T}$. Then:

$$\left\langle \boldsymbol{\tau}\boldsymbol{\tau}^T, \boldsymbol{T} \right\rangle = \lambda_{\min}(\boldsymbol{T}) < 0, \tag{3.60}$$

so $\boldsymbol{\tau}\boldsymbol{\tau}^T \in (\mathbb{S}^n_+)^*$ corresponds to a separation cut that excludes $\boldsymbol{T}$ from $\mathcal{P}$. It can also be seen that $\boldsymbol{\tau}\boldsymbol{\tau}^T$ is a subgradient of the homogeneous convex function $-\lambda_{\min}$ which defines the $\mathcal{P}$ cone.

These separation cuts are used in SCIP-SDP [67], a general-purpose MISDP solver, and in a specialized MISDP application by Nagarajan et al. [72].

Separation Cuts for $\mathcal{E}$

The exponential cone $\mathcal{E}$ requires more care. Instead of deriving separation cuts through the homogeneous function definition, we present the following procedure.

We assume initial cuts enforce $s \geq 0$ and $r \geq 0$. Suppose we have a solution $(r, s, t) \notin \mathcal{E}$ with $s > 0$, i.e., $r < s \exp\left(\frac{t}{s}\right)$. The following $\mathcal{K}^*$ cut is valid and separates (r, s, t) from $\mathcal{E}$:

$$\left(1, \frac{t-s}{s} \exp\left(\frac{t}{s}\right), -\exp\left(\frac{t}{s}\right)\right) \in \mathcal{E}^*. \tag{3.61}$$

If $s = 0, (r, s, t) \in \mathcal{E}$ iff $r \geq 0$ and $t \leq 0$, so assume we have $(r, 0, t) \notin \mathcal{E}$ and $t > 0$ and $r > 0$. In this case the following cut is valid and separates the point $(r, 0, t)$:

$$\left(\frac{t}{r}, -2\log\left(\frac{e^1 t}{2r}\right), -2\right) \in \mathcal{E}^*. \tag{3.62}$$

If we have $(0, 0, t) \notin \mathcal{E}$ (so $t > 0$), then any initial cut of the form (3.20) separates the point $(0, 0, t)$.

3.3 A branch-and-bound outer approximation algorithm

In this section we state a branch-and-bound algorithm for solving $\mathfrak{M}$ to global optimality based on LP relaxations and subproblem cuts. Our algorithm is a conic analogue of the Quesada and Grossmann [76] algorithm for convex MINLP. It differs from the iterative OA algorithm presented in Chapter 2 in that we use a single search tree instead of solving a sequence of MILP instances each with their own search tree. Our intention is to present a simplified and correct algorithm; later we will discuss how our implementation differs in some significant ways.

We assume that $\mathfrak{M}$ is bounded, i.e., it has an optimal solution or it is infeasible, and also that the continuous relaxation $\mathfrak{R}$ is bounded. We assume that strong duality holds for the relaxation and for any continuous subproblems. For convenience we also

assume that we have explicit initial bounds $\boldsymbol{l}^0, \boldsymbol{u}^0$ on the integer variables

We define the template LP subproblem that will be solved at each node of the branch-and-bound tree (recalling that cones C_k for $k \in [\![M]\!] \setminus [\![K]\!]$ are polyhedral):

$$\mathfrak{P}\left(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u}\right) \equiv \min_{\boldsymbol{x}} \quad \langle \boldsymbol{c}, \boldsymbol{x} \rangle : \tag{3.63}$$

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle \in \mathbb{R}_+, \qquad \forall \boldsymbol{z}^k \in \mathcal{Z}_k, k \in [\![K]\!], \tag{3.64}$$

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!] \setminus [\![K]\!], \tag{3.65}$$

$$x_i \in [l_i, u_i], \qquad \forall i \in [\![I]\!]. \tag{3.66}$$

Algorithm 3 describes our LP-based branch-and-bound algorithm. We recursively partition the possible assignments of integer variables by lower and upper bound vectors. Note that $\mathfrak{P}\left(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u}\right)$ is always a lower bound on $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$, so it can be used to prune the search tree analogously to the standard branch-and-bound algorithm where $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$ would be solved at each node of the search tree. Specifically, we can discard a branch of the search tree if $\mathfrak{P}\left(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u}\right)$ is infeasible or returns a bound that is worse than a known upper bound given by a feasible solution. Care must be taken, however, when the optimal solution to the LP is integer feasible. In this case we solve a continuous subproblem with all integer variables fixed to the values returned from the LP (regardless of $\boldsymbol{l}$ and $\boldsymbol{u}$ at the current node), add the corresponding $\mathcal{K}^*$ cuts globally, and re-insert the node into the processing queue. We split the discussion of correctness into two cases:

- If $\boldsymbol{l} = \boldsymbol{u}$ and the LP is feasible, then the node will be discarded when it is next processed. The next time the LP is solved, either it will be infeasible or it will (by Lemma 7) match the value of a known feasible solution which we just obtained by solving $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$. (Of course, the extra LP solve in this case is unnecessary and could be avoided with additional logic.)

- If the optimal solution to the LP is integer feasible and $\boldsymbol{l} \neq \boldsymbol{u}$, for correctness it is sufficient to show that the node will be processed a finite number of times. Consider the case where the optimal solution $\boldsymbol{x}$ to the LP is integer feasible and

we have already added the cuts corresponding to $\mathfrak{R}(\boldsymbol{x}_{\llbracket I \rrbracket}, \boldsymbol{x}_{\llbracket I \rrbracket})$. Then again by Lemma 7 the objective value of the LP must be greater than or equal to the value of a known feasible solution, so the corresponding branch of the search tree will be discarded.

Algorithm 3 is minimal with respect to the $\mathcal{K}^*$ cuts added in order to guarantee finite-time convergence. It is valid, of course, to add additional $\mathcal{K}^*$ cuts at any point in the algorithm. In particular, solving the continuous subproblem $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$ and adding the corresponding cuts at occasional nodes in the tree is analogous to the "hybrid" algorithm for convex MINLP implemented in BONMIN [14].

Algorithm 3 LP-based B&B algorithm for MICP

Solve relaxation $\mathfrak{R}$ with objective value $C_{\mathfrak{R}}$ and optimal dual solution $(\boldsymbol{z}^k)_{k\in[\![M]\!]}$.
Initialize global upper bound: $U \leftarrow \infty$
Initialize $\mathcal{K}^*$ **cut sets:** $\mathcal{Z}_1 \leftarrow \{\boldsymbol{z}^1\}, \ldots, \mathcal{Z}_K \leftarrow \{\boldsymbol{z}^K\}$ (optionally include cuts from Sec. 3.2.1)
Initialize node list: $\mathcal{N} \leftarrow \{(\boldsymbol{l}^0, \boldsymbol{u}^0, C_{\mathfrak{R}})\}$
while $\mathcal{N} \neq \emptyset$ **do**
    Select and remove a node $(\boldsymbol{l}, \boldsymbol{u}, L) \in \mathcal{N}$.
    if $L \geq U$ **then**
        continue
    end if
    Solve local LP relaxation $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$
    if $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$ is feasible and has objective value $L_{\mathfrak{P}} < U$ **then**
        Let $\boldsymbol{x}$ be an optimal solution of $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$
        if $x_i \in \mathbb{Z} \quad \forall i \in [\![I]\!]$ **then**
            Solve continuous subproblem $\mathfrak{R}(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$
            Let $(\boldsymbol{z}^k)_{k\in[\![M]\!]}$ be an optimal solution or ray of $\mathfrak{R}^*(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$
            Update $\mathcal{Z}_k \leftarrow \mathcal{Z}_k \cup \{\boldsymbol{z}^k\}$ for $k \in [\![K]\!]$.
            if $\mathfrak{R}(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$ is feasible **then**
                Let $\hat{\boldsymbol{x}}$ be an optimal solution of $\mathfrak{R}(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$
                if $\langle \boldsymbol{c}, \hat{\boldsymbol{x}} \rangle < U$ **then**
                    Update $U \leftarrow \langle \boldsymbol{c}, \hat{\boldsymbol{x}} \rangle$
                end if
            end if
            $\mathcal{N} \leftarrow \mathcal{N} \cup \{(\boldsymbol{l}, \boldsymbol{u}, \langle \boldsymbol{c}, \boldsymbol{x} \rangle)\}$ ▷ Re-process this node
        else ▷ $\boldsymbol{l} \neq \boldsymbol{u}$ – Branch
            Pick $i \in [\![I]\!]$ with $x_i \notin \mathbb{Z}$
            $\hat{\boldsymbol{l}} \leftarrow \boldsymbol{l}, \hat{\boldsymbol{u}} \leftarrow \boldsymbol{u}$.
            $\hat{l}_i \leftarrow \lfloor x_i \rfloor + 1$
            $\hat{u}_i \leftarrow \lfloor x_i \rfloor$
            $\mathcal{N} \leftarrow \mathcal{N} \cup \{(\boldsymbol{l}, \hat{\boldsymbol{u}}, \langle \boldsymbol{c}, \boldsymbol{x} \rangle), (\hat{\boldsymbol{l}}, \boldsymbol{u}, \langle \boldsymbol{c}, \boldsymbol{x} \rangle)\}$
        end if
    end if
end while

3.4 Numerical scaling of subproblem $\mathcal{K}^*$ cuts

So far we have assumed that all conic or LP subproblems are solved to an exact optimal solution. In this section, we relax this assumption slightly and account for small violations of the feasibility of the $\mathcal{K}^*$ cuts in the solution returned by the LP solver. We will show that under this more realistic, but still idealized, model of the LP solver we can still guarantee convergence up to any positive relative gap tolerance

(as we will define). Specifically, we suppose that the solutions to the LP subproblem $\mathfrak{P}\left(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u}\right)$ satisfy

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle \geq -\epsilon_{feas}, \qquad \forall \boldsymbol{z}^k \in \mathcal{Z}_k, k \in [\![K]\!], \tag{3.67}$$

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!] \setminus [\![K]\!], \tag{3.68}$$

$$x_i \in [l_i, u_i], \qquad \forall i \in [\![I]\!], \tag{3.69}$$

for some absolute tolerance ϵ_{feas}. We assume that (3.68) and (3.69) are satisfied exactly. The following two lemmas derive the proper scaling for subproblem cuts in this setting.

Lemma 11. *Suppose strong duality holds and the dual $\mathfrak{R}^*(\boldsymbol{l}, \boldsymbol{u})$ attains an optimal solution $(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Let C be the optimal objective value (of both the primal and dual). Let $\epsilon_{feas} > 0$, and $\epsilon_{relopt} > 0$. Suppose $\boldsymbol{x} \in \mathbb{R}^N$ satisfies:*

$$\boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x} \in \mathcal{C}_k, \qquad \forall k \in [\![M]\!] \setminus [\![K]\!], \tag{3.70}$$

$$x_i \in [l_i, u_i], \qquad \forall i \in [\![I]\!], \tag{3.71}$$

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \left(\frac{\epsilon_{feas} K}{\epsilon_{relopt}(|C| + 10^{-5})} \right) \boldsymbol{z}^k \right\rangle \geq -\epsilon_{feas}, \qquad \forall k \in [\![K]\!], \tag{3.72}$$

then

$$\frac{C - \langle \boldsymbol{c}, \boldsymbol{x} \rangle}{|C| + 10^{-5}} \leq \epsilon_{relopt}. \tag{3.73}$$

Proof. Suppose that

$$\left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \alpha \boldsymbol{z}^k \right\rangle \geq -\epsilon_{feas}, \qquad \forall k \in [\![K]\!], \tag{3.74}$$

for some $\alpha > 0$. Then:

$$C - \langle \boldsymbol{c}, \boldsymbol{x} \rangle \leq -\sum_{k \in [\![K]\!]} \left\langle \boldsymbol{b}^k - \boldsymbol{A}^k \boldsymbol{x}, \boldsymbol{z}^k \right\rangle \leq \frac{K \epsilon_{feas}}{\alpha}. \tag{3.75}$$

If we wish to imply (3.73) it suffices to enforce:

$$\frac{K\epsilon_{feas}}{\alpha(|C|+10^{-5})} \leq \epsilon_{relopt}, \tag{3.76}$$

which holds whenever:

$$\alpha \geq \frac{\epsilon_{feas}K}{\epsilon_{relopt}(|C|+10^{-5})}. \tag{3.77}$$

□

Lemma 12. *Suppose $\mathfrak{R}(\boldsymbol{l},\boldsymbol{u})$ is infeasible and we have a ray $\left((\boldsymbol{z}^k)_{k\in[\![M]\!]},\boldsymbol{\alpha},\boldsymbol{\beta}\right)$ satisfying:*

$$\sum_{i\in[\![I]\!]}(\beta_i-\alpha_i)\boldsymbol{e}(i)+\sum_{k\in[\![K]\!]}(\boldsymbol{A}^k)^T\boldsymbol{z}^k\in\{0\}^N, \tag{3.78}$$

$$\sum_{i\in[\![I]\!]}(u_i\beta_i-l_i\alpha_i)+\sum_{k\in[\![M]\!]}\left\langle\boldsymbol{b}^k,\boldsymbol{z}^k\right\rangle=-\gamma, \tag{3.79}$$

for some $\gamma>0$. Then for all $\boldsymbol{x}\in\mathbb{R}^N$ satisfying:

$$\boldsymbol{b}^k-\boldsymbol{A}^k\boldsymbol{x}\in\mathcal{C}_k, \qquad \forall k\in[\![M]\!]\setminus[\![K]\!], \tag{3.80}$$

$$x_i\in[l_i,u_i], \qquad \forall i\in[\![I]\!], \tag{3.81}$$

there exists a $k\in[\![K]\!]$ such that $\left\langle\boldsymbol{b}^k-\boldsymbol{A}^k\boldsymbol{x},\frac{K}{\gamma}\boldsymbol{z}^k\right\rangle\leq-1$.

Proof. This can be seen from:

$$\sum_{k\in[\![K]\!]}\left\langle\boldsymbol{b}^k-\boldsymbol{A}^k\boldsymbol{x},\boldsymbol{z}^k\right\rangle\leq\sum_{k\in[\![M]\!]}\left\langle\boldsymbol{b}^k-\boldsymbol{A}^k\boldsymbol{x},\boldsymbol{z}^k\right\rangle+\sum_{i\in[\![I]\!]}(-l_i+x_i)\alpha_i+\sum_{i\in[\![I]\!]}(u_i-x_i)\beta_i \tag{3.82}$$

$$=-\gamma, \tag{3.83}$$

where the final equality follows from the proof of Lemma 8. There must be a $k\in[\![K]\!]$ such that $\left\langle\boldsymbol{b}^k-\boldsymbol{A}^k\boldsymbol{x},\boldsymbol{z}^k\right\rangle$ has less than or equal to the average value which is bounded

by $\frac{-\gamma}{K}$. $\square$

In Algorithm 4 we adapt Algorithm 3 to handle cut scaling. To do so we need to introduce a relative optimality tolerance, which we define as follows.

Definition 4. *Let $\epsilon_{relopt} > 0$. Let $\boldsymbol{x}$ be a feasible solution to $\mathfrak{M}$ and let T be the optimal objective value of $\mathfrak{M}$. We say that $\boldsymbol{x}$ is optimal within relative tolerance ϵ_{relopt} if*

$$\frac{\langle \boldsymbol{c}, \boldsymbol{x} \rangle - T}{|\langle \boldsymbol{c}, \boldsymbol{x} \rangle| + 10^{-5}} \leq \epsilon_{relopt}. \tag{3.84}$$

In the following lemma we state precisely the guarantees on Algorithm 4.

Lemma 13. *Let $0 < \epsilon_{relopt} < 1$ and $0 < \epsilon_{feas} < 1$. When the LP subproblems enforce feasibility of the $\mathcal{K}^*$ cuts to the absolute tolerance ϵ_{feas}, i.e., (3.67), Algorithm 4 terminates in finite time either concluding that $\mathfrak{M}$ is infeasible or returning a feasible solution which is optimal within relative tolerance ϵ_{relopt}.*

Proof. Lemma 12 implies that the LP subproblem cannot return an integer solution corresponding to an integer assignment for which the conic subproblem is infeasible and for which we have already added corresponding scaled $\mathcal{K}^*$ cuts. Condition (3.73) implies that the condition $L_{\mathfrak{P}} < U - \epsilon_{relopt}(|U| + 10^{-5})$ on Line 11 cannot hold if the optimal solution of $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$ is integral and we have already added scaled $\mathcal{K}^*$ cuts corresponding to this feasible assignment for the conic subproblem. These two observations imply that each node in the search tree is processed a finite number of times.

It remains to show that we do not improperly discard branches of the search tree because of the condition on Line 11. Let $\boldsymbol{x}$ be the best feasible solution which the algorithm terminates with. Clearly the upper bound U always satisfies $\langle \boldsymbol{c}, \boldsymbol{x} \rangle \leq U$, so it always holds that $\langle \boldsymbol{c}, \boldsymbol{x} \rangle - \epsilon_{relopt}(|\langle \boldsymbol{c}, \boldsymbol{x} \rangle| + 10^{-5}) \leq U - \epsilon_{relopt}(|U| + 10^{-5})$ because $\epsilon_{relopt} < 1$. If we discard a branch of the search tree because $L_{\mathfrak{P}} \geq U - \epsilon_{relopt}(|U| + 10^{-5})$ then $L_{\mathfrak{P}} \geq \langle \boldsymbol{c}, \boldsymbol{x} \rangle - \epsilon_{relopt}(|\langle \boldsymbol{c}, \boldsymbol{x} \rangle| + 10^{-5})$, and since $L_{\mathfrak{P}}$ remains a valid lower bound, it follows that by discarding this branch, in the worst case we have discarded a

Algorithm 4 LP-based B&B algorithm for MICP with relative gap and cut scaling

Solve relaxation $\mathfrak{R}$ with objective value $C_{\mathfrak{R}}$ and optimal dual solution $(\boldsymbol{z}^k)_{k\in[\![M]\!]}$.
Initialize global upper bound: $U \leftarrow \infty$
Initialize $\mathcal{K}^*$ **cut sets:** $\mathcal{Z}_1 \leftarrow \{\boldsymbol{z}^1\}, \ldots, \mathcal{Z}_K \leftarrow \{\boldsymbol{z}^K\}$ (optionally include cuts from Sec. 3.2.1)
Initialize node list: $\mathcal{N} \leftarrow \{(\boldsymbol{l}^0, \boldsymbol{u}^0, C_{\mathfrak{R}})\}$
while $\mathcal{N} \neq \emptyset$ **do**
    Select and remove a node $(\boldsymbol{l}, \boldsymbol{u}, L) \in \mathcal{N}$.
    if $L \geq U$ **then**
        continue
    end if
    Solve local LP relaxation $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$ with relaxed feasibility (3.67)
    if $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$ is feasible and has objective value $L_{\mathfrak{P}} < U - \epsilon_{relopt}(|U| + 10^{-5})$ **then**
        Let $\boldsymbol{x}$ be an optimal solution of $\mathfrak{P}(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u})$
        if $x_i \in \mathbb{Z} \quad \forall i \in [\![I]\!]$ **then**
            Solve continuous subproblem $\mathfrak{R}(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$
            Let $((\boldsymbol{z}^k)_{k\in[\![M]\!]}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ be an optimal solution or ray of $\mathfrak{R}^*(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$
            if $\mathfrak{R}(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$ is feasible **then**
                Let $\hat{\boldsymbol{x}}$ be an optimal solution of $\mathfrak{R}(\boldsymbol{x}_{[\![I]\!]}, \boldsymbol{x}_{[\![I]\!]})$
                Rescale $\boldsymbol{z}^k \leftarrow \frac{\epsilon_{feas} K}{\epsilon_{relopt}(|\langle \boldsymbol{c}, \hat{\boldsymbol{x}} \rangle| + 10^{-5})} \boldsymbol{z}^k$ for $k \in [\![K]\!]$
                if $\langle \boldsymbol{c}, \hat{\boldsymbol{x}} \rangle < U$ **then**
                    Update $U \leftarrow \langle \boldsymbol{c}, \hat{\boldsymbol{x}} \rangle$
                end if
            else
                Let $\gamma \leftarrow \sum_{k\in[\![M]\!]} \langle \boldsymbol{b}^k, \boldsymbol{z}^k \rangle - \sum_{i\in[\![I]\!]} (l_i \alpha_i - u_i \beta_i)$
                Rescale $\boldsymbol{z}^k \leftarrow \frac{K}{\gamma} \boldsymbol{z}^k$ for $k \in [\![K]\!]$
            end if
            Update $\mathcal{Z}_k \leftarrow \mathcal{Z}_k \cup \{\boldsymbol{z}^k\}$ for $k \in [\![K]\!]$.
            $\mathcal{N} \leftarrow \mathcal{N} \cup \{(\boldsymbol{l}, \boldsymbol{u}, \langle \boldsymbol{c}, \boldsymbol{x} \rangle)\}$ ▷ Re-process this node
        else ▷ $\boldsymbol{l} \neq \boldsymbol{u}$ – Branch
            Proceed as in Algorithm 3
        end if
    end if
end while

better feasible solution with objective value $\langle \boldsymbol{c}, \boldsymbol{x} \rangle - \epsilon_{relopt}(|\langle \boldsymbol{c}, \boldsymbol{x} \rangle| + 10^{-5})$. Hence, by definition $\boldsymbol{x}$ is optimal within relative tolerance ϵ_{relopt}. □

As far as we are aware, such an analysis of OA approaches based on a LP solver with a feasibility tolerance is novel in both the conic MICP and convex MINLP settings. We believe that it is possible, but less straightforward, to develop a convex MINLP analogue of the results in this section.

3.5 Extensions of $\mathcal{K}^*$ cuts

3.5.1 Extreme $\mathcal{K}^*$ cuts

In this section we present an idea for decomposing a $\mathcal{K}^*$ cut into sum of multiple $\mathcal{K}^*$ cuts, if possible, thereby obtaining a stronger outer approximation for $\mathcal{K}$. The basic idea is that if a cut is not an extreme ray of the cone $\mathcal{K}$, then (by definition) it can be written as a nonnegative combination of extreme rays of $\mathcal{K}$. We can then add the $\mathcal{K}^*$ cut for each of these extreme rays, and together, these 'extreme' $\mathcal{K}^*$ cuts imply (via the nonnegative combination) the original cut.

It is not hard to see that any points on the boundaries of the $\mathcal{L}$ and $\mathcal{E}$ cones except $\mathbf{0}$ are extreme rays. This is not the case for the $\mathcal{P}$ cone, where the set of extreme rays is the set of rank-one matrices [47]. We describe here how to obtain extreme $\mathcal{K}^*$ cuts for this important case.

Consider a positive semidefinite conic constraint in smat space: $\boldsymbol{T} \in \mathbb{S}^n_+$. Suppose we have a dual point $\boldsymbol{W} \in \left(\mathbb{S}^n_+\right)^* = \mathbb{S}^n_+$, which yields the standard $\mathcal{K}^*$ cut $\langle \boldsymbol{W}, \boldsymbol{T} \rangle \in \mathbb{R}_+$. Rather than add this single $\mathcal{K}^*$ cut, we use the idea of extreme $\mathcal{K}^*$ cuts to try to get multiple stronger $\mathcal{K}^*$ cuts.

Consider forming an eigendecomposition of $\boldsymbol{W} \in \mathbb{S}^n_+$:

$$\boldsymbol{W} = \sum_{i \in [\![n]\!]} \lambda_i \boldsymbol{\omega}^i (\boldsymbol{\omega}^i)^T, \tag{3.85}$$

where for all $i \in [\![n]\!]$, $\lambda_i \geq 0$ is the ith eigenvalue and $\boldsymbol{\omega}^i$ is the ith eigenvector of $\boldsymbol{W}$. Note that:

$$\lambda_i \boldsymbol{\omega}^i (\boldsymbol{\omega}^i)^T \in \left(\mathbb{S}^n_+\right)^*, \qquad \forall i \in [\![n]\!], \tag{3.86}$$

and furthermore for each $i : \lambda_i > 0$, $\lambda_i \boldsymbol{\omega}^i (\boldsymbol{\omega}^i)^T$ is a nonzero rank-1 PSD matrix, i.e., an extreme ray of the cone. Thus we replace $\boldsymbol{W}$ with the following (up to n) extreme

$\mathcal{K}^*$ cuts:

$$\left\langle \boldsymbol{T}, \lambda_i \boldsymbol{\omega}^i (\boldsymbol{\omega}^i)^T \right\rangle \in \mathbb{R}_+ \qquad \forall i \in [\![n]\!] : \lambda_i > 0. \tag{3.87}$$

Aggregating these cuts by summing, we see that they imply the single $\mathcal{K}^*$ cut $\langle \boldsymbol{W}, \boldsymbol{T} \rangle \in \mathbb{R}_+$. Their validity is guaranteed by the fact that they themselves are $\mathcal{K}^*$ cuts, i.e. points of the dual cone.

3.5.2 Rotated SOC cuts for the PSD cone

Here we extend the idea of $\mathcal{K}^*$ cuts beyond polyhedral outer approximation; in particular, we develop outer approximations of the PSD cone $\mathcal{P}/\mathbb{S}_+$ based on the rotated second-order cone $\mathcal{V}$. We then show that a single '$\mathcal{V}$ cut' can be interpreted as an infinite family of linear $\mathcal{K}^*$ cuts. The following two lemmas which draw heavily from Kim et al. [55].

Lemma 14. *Let:*

$$\boldsymbol{T} = \begin{bmatrix} \phi_0 & \boldsymbol{\phi} \\ \boldsymbol{\phi}^T & \boldsymbol{\Phi} \end{bmatrix} \in \mathbb{S}^n, \qquad \boldsymbol{W} = \begin{bmatrix} \psi_0 & \boldsymbol{\psi} \\ \boldsymbol{\psi}^T & \boldsymbol{\Psi} \end{bmatrix} \in \left(\mathbb{S}^n_+\right)^*. \tag{3.88}$$

Suppose we can decompose $\boldsymbol{\Psi} \in \mathbb{S}^{n-1}_+$ into a sum of L rank-one matrices, i.e.,

$$\boldsymbol{\Psi} = \sum_{l \in [\![L]\!]} \boldsymbol{\nu}^l (\boldsymbol{\nu}^l)^T. \tag{3.89}$$

Note that a decomposition of this form can be obtained from an eigenvalue decomposition. Then $\boldsymbol{T} \in \mathbb{S}^n_+$ implies:

$$\left(\frac{\phi_0}{\sqrt{2}}, \frac{\langle \boldsymbol{\Psi}, \boldsymbol{\Phi} \rangle}{\sqrt{2}}, \left\langle \boldsymbol{\phi}, \boldsymbol{\nu}^1 \right\rangle, \ldots, \left\langle \boldsymbol{\phi}, \boldsymbol{\nu}^L \right\rangle \right) \in \mathcal{V}^{L+2}, \tag{3.90}$$

hence (3.90) *defines a valid outer approximation of $\mathcal{P}$.*

Proof. Kim et al. [55] prove a variant of the standard Schur-complement result that $\boldsymbol{T} \in \mathbb{S}^n_+$ iff $\phi_0 \geq 0, \boldsymbol{\Phi} \in \mathbb{S}^{n-1}_+$, and $\phi_0 \boldsymbol{\Phi} - \boldsymbol{\phi}\boldsymbol{\phi}^T \in \mathbb{S}^{n-1}_+$. Given $\boldsymbol{T}, \boldsymbol{W} \in \mathbb{S}^n_+$, it must

hold that $\left\langle \boldsymbol{\Psi}, \phi_0 \boldsymbol{\Phi} - \boldsymbol{\phi}\boldsymbol{\phi}^T \right\rangle \geq 0$, i.e.,

$$\phi_0 \left\langle \boldsymbol{\Psi}, \boldsymbol{\Phi} \right\rangle - \boldsymbol{\phi}^T \boldsymbol{\Psi} \boldsymbol{\phi} \geq 0. \tag{3.91}$$

Using $\boldsymbol{\phi}^T \boldsymbol{\Psi} \boldsymbol{\phi} = \sum_{l \in [L]} \left\langle \boldsymbol{\phi}, \boldsymbol{\nu}^l \right\rangle^2$, we obtain

$$\phi_0 \left\langle \boldsymbol{\Psi}, \boldsymbol{\Phi} \right\rangle \geq \sum_{l \in [\![L]\!]} \left\langle \boldsymbol{\phi}, \boldsymbol{\nu}^l \right\rangle^2, \tag{3.92}$$

which is precisely (3.90). □

Lemma 15. *Fix $\boldsymbol{\Psi} \in \mathbb{S}_+^{n-1}$ and obtain a decomposition of the form* (3.89)*. Then* (3.90) *holds if and only if $\langle \boldsymbol{W}, \boldsymbol{T} \rangle \geq 0$ for all $\psi_0 \in \mathbb{R}, \boldsymbol{\psi} \in \mathbb{R}^{n-1}$ such that:*

$$\boldsymbol{W} = \begin{bmatrix} \psi_0 & \boldsymbol{\psi} \\ \boldsymbol{\psi}^T & \boldsymbol{\Psi} \end{bmatrix} \in \left(\mathbb{S}_+^n\right)^*. \tag{3.93}$$

Proof. Theorem 3.3 of [55]. □

Note that the choice the upper-left diagonal element to decompose the matrices $\boldsymbol{T}$ and $\boldsymbol{W}$ in Lemma 14 is arbitrary. In particular, a single dual matrix $\boldsymbol{W} \in \left(\mathbb{S}_+^n\right)^*$ immediately yields n $\mathcal{V}$ outer approximation cuts by different choices of diagonal element.

We can use the $\mathcal{V}$ cuts for $\mathcal{P}$ to improve our initial fixed outer approximation for the PSD cone discussed in Section 3.2.1. Recall that any positive semidefinite matrix satisfies the condition that every 2×2 principal sub-matrix is PSD. Thus $\boldsymbol{T} \in \mathbb{S}_+^n$ implies:

$$\begin{bmatrix} T_{i,i} & T_{i,j} \\ T_{j,i} & T_{j,j} \end{bmatrix} \in \mathbb{S}_+^2, \qquad \forall i, j \in [\![n]\!] : i > j. \tag{3.94}$$

Note $T_{j,i} = T_{i,j}, \forall i, j \in [\![n]\!]$ because symmetry is already enforced (see Section 3.1.2). Furthermore when $i = j$ the condition is trivial because we impose $T_{i,i} \in \mathbb{R}_+, \forall i \in [\![n]\!]$ when we add the initial fixed $\mathcal{K}^*$ cuts described in Section 3.2.1.

We use the PSD cone $\mathcal{V}$ cuts described in (3.90) with a particular set of $\mathcal{P}^*$ points in order to impose the conditions (3.94). In contrast to the initial fixed polyhedral outer approximation of the $\mathcal{P}$ cone we proposed in Section 3.2.1, these $\mathcal{V}$ cuts give an initial SOCP outer approximation. As we will discuss, the $\mathcal{V}$ initial fixed cuts in fact imply the linear initial fixed cuts.

Consider the n (rank-1) dual cone points $\boldsymbol{e}(i)\boldsymbol{e}(i)^T \in \left(\mathbb{S}_+^n\right)^*, \forall i \in [\![n]\!]$. For each points we add a subset of the n possible SOC cuts we could add (one for each diagonal element as ϕ_0). Specifically, for dual cone point j, we add the $n-j$ SOC cuts choosing diagonal indices i for which $i > j$ for ϕ_0:

$$\left(\frac{T_{i,i}}{\sqrt{2}}, \frac{\left\langle \boldsymbol{e}(i)\boldsymbol{e}(i)^T, \boldsymbol{T} \right\rangle}{\sqrt{2}}, \langle \boldsymbol{e}(i), \boldsymbol{T}_{:,j} \rangle\right) = \left(\frac{T_{i,i}}{\sqrt{2}}, \frac{T_{j,j}}{\sqrt{2}}, T_{i,j}\right) \in \mathcal{V}^3, \tag{3.95}$$

where $\boldsymbol{T}_{:,j}$ denotes the jth column vector of $\boldsymbol{T}$. Thus we add $\frac{n(n-1)}{2}$ $\mathcal{V}$ cuts for a fixed initial outer approximation of the $\mathbb{S}_+^n$ cone. These cuts imply the condition (3.94), which by the Schur complement is equivalent to:

$$T_{i,i}T_{j,j} \geq T_{i,j}^2, \qquad \forall i, j \in [\![n]\!] : i > j, \tag{3.96}$$

which is equivalent to:

$$\left(\frac{T_{i,i}}{\sqrt{2}}, \frac{T_{j,j}}{\sqrt{2}}, T_{i,j}\right) \in \mathcal{V}^3, \qquad \forall i, j \in [\![n]\!] : i > j. \tag{3.97}$$

Ahmadi and Hall [3] discuss LP and SOCP outer approximations of the PSD cone. The cuts (3.95) correspond to the extreme rays of the dual cone of 'scaled diagonally dominant' matrices. As the authors argue in [3], the cone of scaled diagonally dominant matrices is an important subset of the PSD cone, so its dual cone provides a potentially useful initial fixed outer approximation of the PSD cone. Both cones are representable as an intersection of rotated second-order cone constraints. Furthermore, since the cone of diagonally dominant matrices is a strict subset of the cone of scaled diagonally dominant matrices, the PSD cone outer approximation we get from

these $\mathcal{V}$ initial fixed cuts is strictly stronger than the polyhedral outer approximation we described in Section 3.2.1.

3.5.3 Lifted $\mathcal{K}^*$ cuts for SOC cone disaggregation

In Chapter 2 we argue that a major motivation for using conic representations of convex constraints in the context of mixed-integer convex optimization is that conic formulations are already lifted or extended formulations of the original problem. Lifted formulations are known to accelerate outer-approximation approaches [83, 45].

However, once a problem is encoded in conic form, there may be further opportunities to exploit lifted formulations. For example, the second-order cone $\mathcal{L}$ is known to have multiple lifted formulations in terms of lower-dimensional cones [88, 9]. It may be computationally advantageous to provide the original formulation to the conic solver for subproblems, e.g., $\mathfrak{R}(\boldsymbol{l}, \boldsymbol{u})$, and use a lifted formulation in the outer-approximation problems, e.g., $\mathfrak{P}\left(\mathcal{Z}_1, \ldots, \mathcal{Z}_K, \boldsymbol{l}, \boldsymbol{u}\right)$. In this section, we exploit this idea principally for the second-order cone but with an eye towards more general applicability.

Consider a cone $\mathcal{C}$ which we know is a projection of a product of cones $\tilde{\mathcal{C}}_1 \times \cdots \times \tilde{\mathcal{C}}_L$ i.e. $\boldsymbol{y} \in \mathcal{C}$ if and only if there exists a $\boldsymbol{\pi}$ such that:

$$\boldsymbol{F}^l \boldsymbol{y} + \boldsymbol{G}^l \boldsymbol{\pi} \in \tilde{\mathcal{C}}_l, \qquad \forall l \in [\![L]\!], \tag{3.98}$$

for some $\boldsymbol{F}^1, \ldots, \boldsymbol{F}^L$ and $\boldsymbol{G}^1, \ldots, \boldsymbol{G}^L$ fixed.

We can observe that any set of $\mathcal{K}^*$ cuts for the product of cones $\tilde{\mathcal{C}}_1 \times \cdots \times \tilde{\mathcal{C}}_L$ would project to at least one $\mathcal{K}^*$ cut for $\mathcal{C}$. Let $\tilde{\boldsymbol{z}}^l \in \tilde{\mathcal{C}}_l$ for each $l \in [\![L]\!]$. Then given any set of multipliers $\boldsymbol{\alpha} \geq \boldsymbol{0}$ such that $\sum_{l \in [\![L]\!]} \alpha_l (\boldsymbol{G}^l)^T \tilde{\boldsymbol{z}}_{l \in [\![L]\!]} = \boldsymbol{0}$ (which project out $\boldsymbol{\pi}$ when aggregating the lifted $\mathcal{K}^*$ cuts), the cuts imply $\left\langle \sum_{l \in [\![L]\!]} \alpha_l (\boldsymbol{F}^l)^T \tilde{\boldsymbol{z}}^l, \boldsymbol{y} \right\rangle \geq 0, \forall \boldsymbol{y} \in \mathcal{C}$, so $\sum_{l \in [\![L]\!]} \alpha_l (\boldsymbol{F}^l)^T \tilde{\boldsymbol{z}}^l \in \mathcal{C}^*$.

The strength of lifted cuts of this form is that a small collection of cuts in the lifted space can project down to a large number of $\mathcal{K}^*$ cuts in the original space [88].

For the case of the second-order cone $\mathcal{L}$, Vielma et al. [88] were the first to present

a lifted formulation as $(r, \boldsymbol{t}) \in \mathcal{L}^{n+1}$ iff $\exists \boldsymbol{\pi} \in \mathbb{R}^n$ such that:

$$(r, \pi_i, t_i) \in \mathcal{V}, \qquad \forall i \in [\![n]\!], \tag{3.99}$$

$$r - \sum_{i \in [\![n]\!]} 2\pi_i \in \mathbb{R}_+. \tag{3.100}$$

Vielma et al. [88] showed that adding inequalities in the extended space (r, π_i, t_i) yields tighter polyhedral approximations and accelerates outer-approximation methods for MISOCP.

One may prefer to keep the $\boldsymbol{\pi}$ variables explicitly in the outer approximation problem, but not in the conic subproblem, because the conic solvers may handle the original SOC problem more efficiently than they handle the lifted problem. If the lifting were instead performed as a preprocessing step to $\mathfrak{M}$, this would not be possible, and furthermore some information would be lost, e.g., with initial cuts based on ℓ_1 that we present soon. The discussion below shows how to transform a $\mathcal{L}^*$ cut into a lifted cut compatible with the formulation (3.99)-(3.100).

Given a dual point $(u, \boldsymbol{w}) \in (\mathcal{L}^n)^*$, consider the n 3-dimensional vectors:

$$\left(\frac{w_i^2}{2\|\boldsymbol{w}\|_2^2}, 1, \frac{w_i}{\|\boldsymbol{w}\|_2} \right) \in \mathcal{V}^*, \qquad \forall i \in [\![n]\!]. \tag{3.101}$$

We show that the lifted $\mathcal{K}^*$ cuts on (r, π_i, t_i) from these points imply the original cut $\langle (u, \boldsymbol{w}), (r, \boldsymbol{t}) \rangle \in \mathbb{R}_+$. Suppose $r, \boldsymbol{\pi}, \boldsymbol{t}$ satisfy:

$$\frac{w_i^2}{2\|\boldsymbol{w}\|_2^2} r + \pi_i + \frac{w_i}{\|\boldsymbol{w}\|_2} t_i \geq 0, \qquad \forall i \in [\![n]\!], \tag{3.102}$$

$$r - \sum_{i \in [\![n]\!]} 2\pi_i \geq 0. \tag{3.103}$$

Aggregating constraints (3.102) over i we obtain:

$$\sum_{i\in[\![n]\!]}\left(\frac{w_i^2}{2\|\boldsymbol{w}\|_2^2}r+\pi_i+\frac{w_i}{\|\boldsymbol{w}\|_2}t_i\right) \tag{3.104}$$

$$=\frac{\|\boldsymbol{w}\|_2^2}{2\|\boldsymbol{w}\|_2^2}r+\sum_{i\in[\![n]\!]}\pi_i+\sum_{i\in[\![n]\!]}\frac{w_it_i}{\|\boldsymbol{w}\|_2} \tag{3.105}$$

$$=\frac{r}{2}+\sum_{i\in[\![n]\!]}\pi_i+\sum_{i\in[\![n]\!]}\frac{w_it_i}{\|\boldsymbol{w}\|_2} \tag{3.106}$$

$$\geq 0. \tag{3.107}$$

Using the constraint (3.103) or $\frac{r}{2}\geq\sum_{i\in[\![n]\!]}\pi_i$:

$$\left\langle\left(1,\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\right),(r,\boldsymbol{t})\right\rangle \tag{3.108}$$

$$=r+\sum_{i\in[\![n]\!]}\frac{w_it_i}{\|\boldsymbol{w}\|_2} \tag{3.109}$$

$$\geq\frac{r}{2}+\sum_{i\in[\![n]\!]}\pi_i+\sum_{i\in[\![n]\!]}\frac{w_it_i}{\|\boldsymbol{w}\|_2} \tag{3.110}$$

$$\geq 0. \tag{3.111}$$

We see that the $\mathcal{K}^*$ cuts in lifted space imply the original $\mathcal{K}^*$ cut $\left\langle\left(1,\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\right),(r,\boldsymbol{t})\right\rangle\in\mathbb{R}_+$.

We can use the lifting for $\mathcal{L}$ to improve our initial fixed outer approximation discussed in Section 3.2.1. Note that $(r,\boldsymbol{t})\in\mathcal{L}^{n+1}$ implies $\frac{\|\boldsymbol{t}\|_1}{\sqrt{n}}\leq r$, which could be used to generate an initial polyhedral outer approximation for $\mathcal{L}$. The epigraph of the ℓ_1 norm cannot be compactly formulated in the space of $(r,\boldsymbol{t})$ directly; the most straightforward approach would be to introduce auxiliary variables modeling $|t_i|$ for each i. Instead of adding these auxiliary variables, we propose a more economical method for enforcing the ℓ_1 constraint by using the auxiliary variables $\boldsymbol{\pi}$ which we assume are already present in the model for adding lifted cuts.

Take the two dual points $\left(1,\frac{\pm\mathbf{1}}{\sqrt{n}}\right)\in(\mathcal{L}^{n+1})^*$ and turn them into lifted $\mathcal{K}^*$ cuts by following the discussion in the previous section. Note $\|\frac{\pm\mathbf{1}}{\sqrt{n}}\|_2=1$, so $\frac{w_i}{\|\boldsymbol{w}\|_2}=\frac{\pm\mathbf{1}}{\sqrt{n}},\forall i\in$

$[\![n]\!]$. The $2n$ lifted $\mathcal{K}^*$ cuts are:

$$\left\langle \left(\frac{1}{2n}, 1, \pm\frac{1}{\sqrt{n}} \right), (r, \pi_i, t_i) \right\rangle \in \mathbb{R}_+, \qquad \forall i \in [\![n]\!], \tag{3.112}$$

that is,

$$\frac{1}{2n} r + \pi_i + \frac{\pm 1}{\sqrt{n}} t_i \geq 0, \qquad \forall i \in [\![n]\!]. \tag{3.113}$$

By construction these define a valid set of lifted $\mathcal{K}^*$ cuts because $\left(\frac{1}{2n}, 1, \pm\frac{1}{\sqrt{n}}\right) \in \mathcal{V}^*$. If we pick a set of signs $s_i \in \{-1, +1\}$ for $i \in [\![n]\!]$ and aggregate the cuts defined by $\left(\frac{1}{2n}, 1, s_i\frac{1}{\sqrt{n}}\right)$, we obtain

$$\sum_{i \in [\![n]\!]} \left(\frac{1}{2n} r + \pi_i + \frac{s_i}{\sqrt{n}} t_i \right) \tag{3.114}$$

$$= \frac{r}{2} + \sum_{i \in [\![n]\!]} \pi_i + \sum_{i \in [\![n]\!]} \frac{s_i t_i}{\sqrt{n}}, \tag{3.115}$$

which together with $\frac{r}{2} \geq \sum_{i \in [\![n]\!]} \pi_i$ we obtain

$$r + \sum_{i \in [\![n]\!]} \frac{s_i t_i}{\sqrt{n}} \geq 0 \tag{3.116}$$

The 2^n choices of signs s_i together imply:

$$r \geq \sum_{i \in [\![n]\!]} \frac{|t_i|}{\sqrt{n}} = \frac{\|\boldsymbol{t}\|_1}{\sqrt{n}}, \tag{3.117}$$

which is the ℓ_1 condition we wanted to enforce.

3.6 `Pajarito` solver and related software

In this section we describe our solver, Pajarito, and the surrounding software architecture. This section may be of particular interest to authors of mathematical optimization software or advanced users.

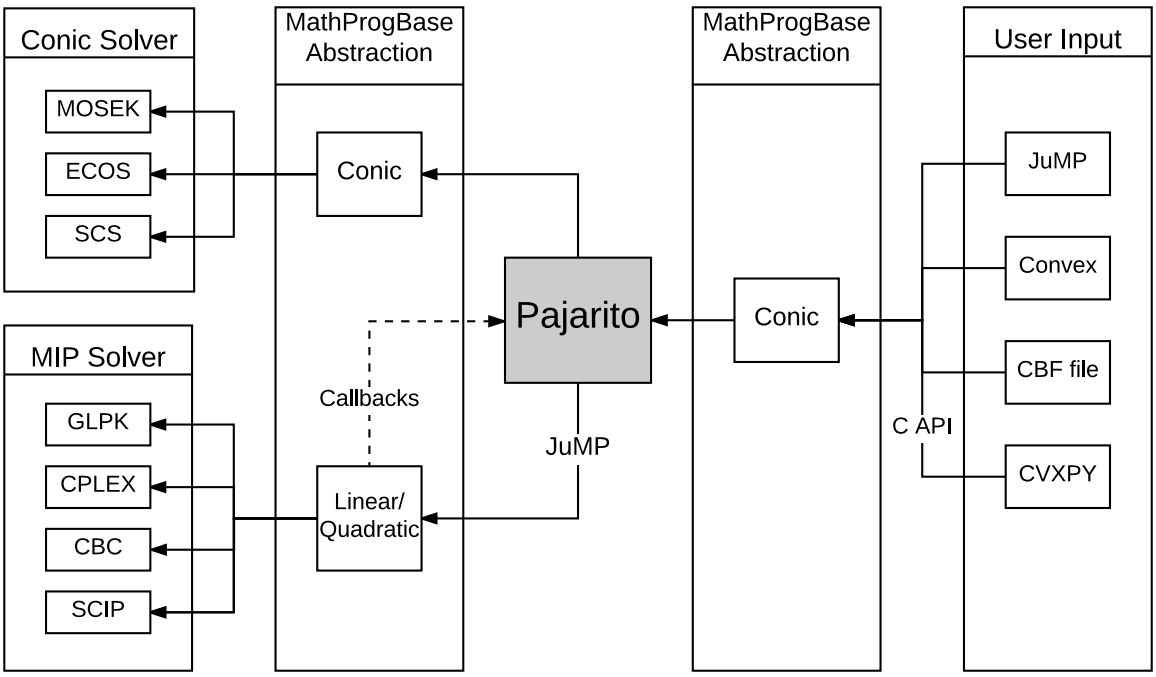


Figure 3-1: Pajarito fully exploits the MathProgBase abstraction layer to simultaneously accept input from many forms and call out to MIP and continuous conic solvers in a solver-independent way.

3.6.1 JuliaOpt

Pajarito is the first MICP solver written in the Julia language [12], while existing convex MINLP solvers like α-ECP, Artelys Knitro, Bonmin, DICOPT, FilMINT, MINLP_BB, and SBB reviewed by Bonami et al. [15] are written in C, C++, or Fortran to our knowledge. Julia is a high-level programming language which can match the performance of these lower-level languages for writing solvers [62] with much less boilerplate code. Pajarito's code is compact, and we intend for it to be reusable and extensible by other researchers. Pajarito is fully integrated with the JuliaOpt ecosystem, including the powerful MathProgBase abstraction layer.

3.6.2 MathProgBase

The MathProgBase abstraction layer, initially developed as a backend for JuMP [30], is a standardized API in Julia for interacting with solvers which currently includes specifications for `Linear/Quadratic` solvers (e.g., from linear optimization to mixed-

integer quadratic optimization), `Conic` solvers (both continuous and mixed-intger), and derivative-based `Nonlinear` solvers. The breadth of problem classes covered by MathProgBase distinguishes it from similar abstraction layers like OSI [79], a COIN-OR library in C++. Pajarito interacts with the MathProgBase interface in two ways. First, it implements the `Conic` solver interface so that it may act as a mixed-integer conic solver and take input from the JuMP [30] and Convex.jl [85] modeling interfaces and other external interfaces which will be discussed later. Second, Pajarito uses JuMP directly to solve the mixed-integer relaxation, and for the branch-and-cut algorithm Pajarito uses JuMP's solver-independent callback functionality. Under the hood, JuMP interacts with these solvers via the `Linear/Quadratic` interface, which allows us to easily exchange the MILP solver. To interact with conic solvers, Pajarito directly manipulates the conic matrix data and uses the `Conic` solver interface. See Figure 3-1 for an illustration.

Beyond the choice of programming language and abstractions, the most significant architectural difference between Pajarito and existing MICP solvers is that Pajarito takes the conic-form problem $\mathfrak{M}$ as input while previously developed solvers interact with the instance almost exclusively through oracles to query values and derivatives of the constraints and objective function.[2] Unlike derivative-based input, conic-form input can be described compactly with the matrices $\boldsymbol{A}^k$, the vectors $\boldsymbol{c}$ and $\boldsymbol{b}^k$, and a small data structure describing the cones $\mathcal{C}_k$ (assuming these are taken from a small number of known cones). This compact representation makes the interface to the solver particularly straightforward; it is analogous in complexity to that of LP and MILP solvers.

We have developed a proof-of-concept C API (`cmpb`) which enables access to MathProgBase `Conic` solvers by embedding Julia as a shared library. In collaboration with Steven Diamond and Baris Ungun, we were able to demonstrate accessing Pajarito from CVXPY [26], a Python-based disciplined convex modeling package, through the `cmpb` interface. At this time we would recommend this interface for expert users only

[2] As part of our preliminary work [63], we developed a derivative-based algorithm which is included in the release of Pajarito.

because it has not been well tested. In Pajarito's README file we provide more guidance on the recommended ways of using the solver.

The use of MathProgBase does come with a couple of drawbacks that we would like to highlight. First, MathProgBase does not attempt to provide an abstraction for solver parameters like convergence tolerances. In cases where we need certain tolerances on the subproblem solvers in order for Pajarito to converge to a requested tolerance, it is the user's responsibility to set the correct tolerances on the subproblem solver. For example, we ask users to manually adjust the MILP solver's linear feasibility tolerance and integer feasibility tolerance for improved convergence behavior. These cases are documented in Pajarito's README file.

The second drawback is that the abstraction for solver callbacks in MathProgBase was designed primarily around shared behavior between CPLEX and Gurobi. For example, MathProgBase defines a lazy constraint callback where we can add cuts to exclude a given incumbent. The MILP solver, however, may choose to ignore these cuts for numerical reasons and accept the incumbent anyway. Solvers like CPLEX and SCIP provide facilities to force rejection of an incumbent, but these are not currently accessible through the abstraction layer. Additionally, through the lazy callback we do not have the ability to provide new incumbents to the solver. A heuristic callback is available for this purpose, but no guarantees are available as to when we have an opportunity to provide a new solution. For nominal correctness of Algorithm 3, we must be able to update the incumbent (Line 20) before the next node is processed. Lacking this functionality, we cannot claim correctness of an implementation of Algorithm 3 based on the available callbacks in MathProgBase. In practice, however, we have not observed issues attributable to not being able to provide a new incumbent at the right time. We have observed issues related to not being able to force rejection of an incumbent.

3.6.3 `Pajarito`

Pajarito implements a number of algorithmic variants of the methods described in this section. At the highest level, users may choose between running an iterative OA

algorithm (as described in Chapter 2) and an LP-based branch-and-bound algorithm similar to Algorithms 3 and 4. The branch-and-bound algorithm is implemented using MathProgBase callbacks and is subject to the limitations described in Section 3.6.2. We call the branch-and-bound algorithm *MIP-solver-driven* (MSD) because the MILP solver is responsible for managing convergence and stopping criteria. We use a lazy constraint callback that is invoked whenever the MILP solver finds an integer-feasible solution. Based on this solution, we add corresponding cuts, which may be subproblem cuts as in Section 3.2.3 and/or separation cuts as in Section 3.2.4 according to the options set by the user. If we solve a conic subproblem and find a feasible solution, we provide this solution to the solver through the heuristic callback. By default, we implement the cut scaling described in Section 3.4 which may be disabled by a user option. We have options to enable or disable all of the extensions described in Section 3.5. The LP-based branch-and-bound method has been tested with CPLEX and Gurobi but in principle could work with any solver that implements the MathProgBase callback abstraction.

3.6.4 CBLIB and ConicBenchmarkUtilities

We use the CBF format proposed by H. Friberg [34] to encode benchmark problems. The CBF format was originally designed to support the second-order cone and the PSD cone. We collaborated with H. Friberg to extend the format to support exponential cones, now available as CBF version 2. We developed `ConicBenchmarkUtilties.jl`, a Julia interface to the CBF format which includes utilities to translate between MathProgBase `Conic` format and CBF format.

CBLIB is a benchmark library with problems in CBF format also developed and maintained by H. Friberg. We contributed a number of new problem instances to CBLIB. In Chapter 2, we described translating instances from the MINLPLIB2 library into Convex.jl format. For this experiments in this chapter, we took a representative subset of these instances[3] and translated them to CBF format and contributed them to CBLIB. Per family, the instances with corresponding counts are `gams01` (1), `rsyn` (48),

[3]Available at `https://github.com/mlubin/MICPExperiments`.

`syn` (48), `tls` (6), and `clay` (12), for a total of total 115 instances, including the first exponential cone instances in CBLIB. For some of these instances, we cleaned them of tiny values artificially introduced for derivative-based solvers [41]. For example, the instance `rsyn0805h` has a constraint:

$$\left(\frac{x_{289}}{10^{-6}+b_{306}} - 1.2 * \log\left(1+\frac{x_{285}}{10^{-6}+b_{306}}\right)\right) * (10^{-6}+b_{306}) \leq 0 \tag{3.118}$$

which we re-encode as

$$x_{289} - 1.2\widehat{\log}\,(b_{306}+x_{285}, b_{306}) \leq 0, \tag{3.119}$$

where $\widehat{\log}(x,y) = y\log(x/y)$ is the perspective function of log which has a natural representation using $\mathcal{E}$.

3.7 Computational experiments

We present here computational experiments investigating the algorithmic developments within the cuts framework and then proceed to compare the performance of Pajarito with alternative open-source and commercial solvers for MISOCP.

3.7.1 Methods and presentation of results

The set of instances we use in our computational testing and benchmarks are a selection of 120 MISOCP instances from the CBLIB [34] library. Instances were chosen as representatives of each family of instances in the library, cutting out problems that were either too small to be useful in measuring relative performance or too large for any solver to solve within our one-hour time limit.

In order to be able to run a large number experiments quickly, we chose to use the Amazon EC2 cloud computing platform. We use `m4.xlarge` EC2 computing instances with 16GB RAM for all runs. All MIP and conic solvers are run in single-threaded mode. We note that timing results on EC2 may be subject to random

variability across runs because the computing nodes are virtual machines. We have not yet attempted to quantify this variability.

In addition to the callback-based MIP-solver-driven (MSD) algorithm discussed in this paper, we also compare in some cases with the iterative OA algorithm proposed in Chapter 2. The iterative algorithm is simpler to implement but the MSD algorithm should be expected to perform relatively better.

The MIP solvers we experiment with are Cbc 2.9.8, GLPK 4.61, and CPLEX 12.7.0. Cbc and GLPK are open source, and CPLEX is commercial. We have only implemented the callback-based algorithm for CPLEX; we use Cbc and GLPK within the iterative algorithm only. For continuous second-order cone solvers we use the commercial MOSEK solver version 8 and the open-source ECOS solver [28] version 2.0.5. When paired with an open-source MIP solver and an open-source SOCP solver, Pajarito can be considered a purely open-source MISOCP solver.

In conducting our computational tests, we encountered a surprising number of cases where a solver reported an incorrect answer. We used a filter to search for constraint violations above 10^{-3} and objective value disagreements by more than 0.01% and manually excluded inconsistent instances. Most of these cases we were able to verify that the MIP solver was responsible for giving a 'wrong' answer.

In the discussions that follow, we compare different methods and solvers by counting termination status. The 'conv' status indicates that an instance was claimed to be solved to global optimality within tolerances and we do not have evidence indicating otherwise. The 'wrong' status indicates that the instance was claimed to be solved to global optimality within tolerances, but we manually excluded the result according to the criteria discussed above. The 'not conv' status indicates that the solver stopped because it could not proceed, e.g., because no numerically violated cuts could be added. The 'limit' status indicates that the solver stopped because it hit the time limit or ran out of memory.

As a performance summary, we compute the shifted geometric mean defined as $(\prod_{i \in [\![n]\!]}(t_i + s))^{\frac{1}{n}} - s$ for values $t_1, \ldots, t_n$ and shift s [2]. The shift is designed to decrease the relative influence of "easy" instances. For execution times we shift use

a shift of 10 seconds. For iteration counts we shift by 1 iteration. Geometric means are computed over the set of instances with 'conv' status, which biases the measure against solvers that solve more hard instances.

In addition to the summary statistics, we use performance profiles [27, 37] to compare relative performance by instance. They should be interpreted as follows: For a fixed factor F, the level of each solver on the P axis represents the proportion of instances solved within a factor of F of the fastest solution time. Therefore, higher is better. At $F = 1$, the proportion represents exactly the instances on which the solver was the fastest. As F increases, we can observe if the solver reliably solves problems nearly as fast as the best recorded time. To decrease the influence of very easy instances which solver in a few seconds, we add 10 seconds to all solution times before computing the performance profiles.

The scripts and data used to run our experiments are available at `github.com/mlubin/PajaritoSupplement`.

3.7.2 Testing `Pajarito`

In this section, we present computational experiments addressing two key questions regarding $\mathcal{K}^*$ cuts. First, we ask whether subproblem cuts are sufficient in practice to prove global optimality and evaluate the effect of our proposed numerical scaling of $\mathcal{K}^*$ subproblem cuts. Second, we compare the performance of the separation and subproblem cuts. Separation cuts are already implemented in commercial MISOCP solvers, while subproblem cuts are not. For these algorithmic questions, we fix the MILP solver as CPLEX and use Mosek for the conic subproblems.

How does subproblem cut scaling affect convergence?

In principle, one should be able to prove global optimality using only the subproblem cuts. However, our model does not account for all possible sources of numerical imprecision in a practical implementation. Here, we investigate if the subproblem cuts are indeed sufficient. These experiments are motivated by our initial computational

options		termination status counts				statistics (conv only)	
presolve	scale	conv	wrong	not conv	limit	time(s)	iterations
✓	✓	95	1	3	21	39.59	4.07
✓		87	2	9	22	43.85	4.27
	✓	92	0	0	28	34.70	4.08
		92	0	1	27	37.98	4.08

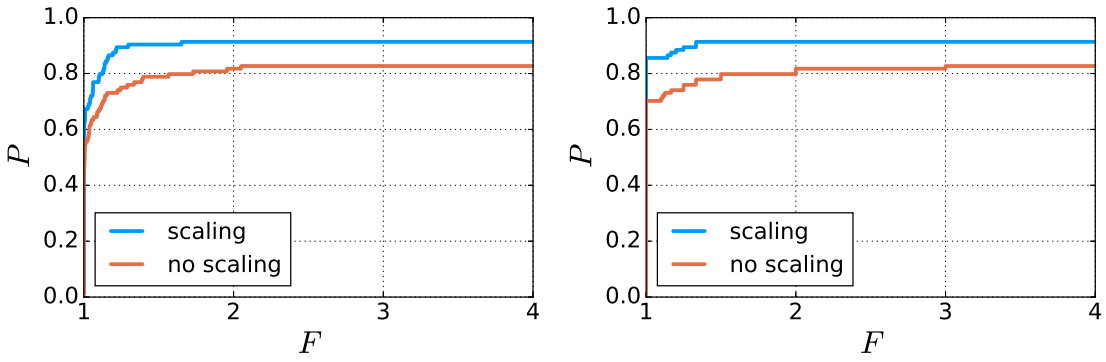
Table 3.1: Termination statuses and shifted geometric mean of solve time and iteration count on the MISOCP library, for the iterative subproblem cuts algorithm with and without MIP presolve and subproblem cuts scaling

experiences where we saw a number of convergence failures on the iterative algorithm proposed in Chapter 2. Hence, we compare here only on the simpler iterative algorithm.

In the first two rows of Table 3.1, we illustrate the effect of the cut scaling option. With cut scaling disabled, Pajarito returns 2 incorrect answers and is unable to converge on 9 instances. With cut scaling enabled, the convergence behavior improves improves substantially to 1 incorrect answer and failure to converge on 3 instances. Also, the shifted geometric mean time and number of iterations decrease. In Figure 3-2 we show performance profiles comparing solution time and iteration count with and without cut scaling.

The incorrect answers on termination can be attributed to the MILP solver. In order to verify this statement, we experimented with disabling CPLEX's 'presolve' functionality which is responsible for performing a number of simplifying transformations on the MILP before solving it. Solutions in the transformed space may not satisfy the tolerances which the users specifies for the original problem. In the last two rows of Table 3.1, we show the results of running with presolve disabled, both with and without cut scaling. Disabling presolve, although it results in decreased performance, eliminates all instances of incorrect answers and leaves only 1 instance of nonconvergence when scaling is disabled.

We conclude from these experiments that cut scaling is important and effective in improving convergence, and that remaining failures to converge, on these instances, can be attributed to numerical inaccuracies of the MILP solver.



(a) Solution time

(b) Iteration count

Figure 3-2: Performance profiles for solve time and iterations on the MISOCP library, for the iterative subproblem cuts algorithm (MIP presolve enabled) with and without subproblem cuts scaling

Do subproblem cuts perform better than separation cuts?

While a convex MINLP analogue of the subproblem cuts algorithm has been implemented, e.g., in BONMIN, our implementation is the first such one for mixed-integer conic optimization using conic solvers. Here, we compare with the more common separation-based cuts which are already implemented in MISOCP solvers like Gurobi, CPLEX, and SCIP. Recall, we refer to the callback-based branch-and-bound algorithm as MSD. Our MSD algorithm with separation cuts is analogous to, but less efficient than, the methods implemented these standalone solvers.

In Table 3.2 and Figure 3-3 we compare our implementation of separation and subproblem cuts under the iterative and MSD algorithms. The MSD algorithm with subproblem cuts solves more instances and is faster than the MSD algorithm with separation cuts. Subproblem cuts also perform better than separation cuts for the iterative algorithm, which is relatively slower than MSD as we had expected.

3.7.3 Comparing MICP solvers

Having investigated some important algorithmic questions, we now make comparisons with other existing solvers.

options		termination status counts				statistics (conv only)	
alg	cuts	conv	wrong	not conv	limit	time(s)	iterations
iter	sep	96	1	0	23	55.23	6.76
iter	subp	95	1	3	21	39.59	4.07
MSD	sep	95	1	0	24	20.86	–
MSD	subp	100	0	1	19	17.56	–

Table 3.2: Termination statuses and shifted geometric mean of solve time and iteration count on the MISOCP library, for the iterative and MSD versions of the separation cuts and subproblem cuts algorithms

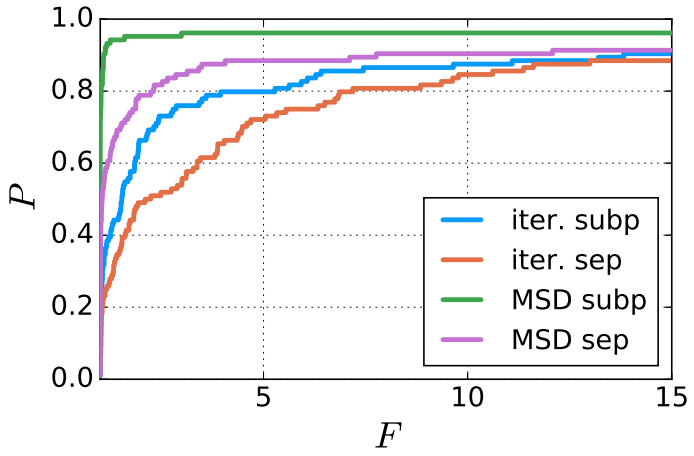


Figure 3-3: Performance profile for solution time on the MISOCP library, for the iterative and MSD versions of the separation cuts and subproblem cuts algorithms

	termination status counts				
solver	conv	wrong	not conv	limit	time(s)
BONMIN-BB	37	27	10	46	82.95
BONMIN-OA	30	8	29	53	72.12
BONMIN-OA-D	35	8	29	48	64.25
Paj-CBC-ECOS	81	8	0	31	51.48
Paj-GLPK-ECOS	68	0	2	50	42.75

Table 3.3: Termination statuses and shifted geometric mean of solve time on the MISOCP library, for BONMIN and default iterative Pajarito solvers using CBC/GLPK and ECOS

Open-source comparison with BONMIN

We are unaware of any mainstream open-source solvers designed for solving MISOCP problems. Instead, we compare with BONMIN, a convex MINLP solver. The NLP representation is not well suited for MISOCPs since the the functional representation of $\mathcal{L}$ cone has points of nondifferentiability, but nevertheless BONMIN is the closest competitor to our knowledge. We note that ECOS includes a very naive branch-and-bound implementation for MISOCP; however in our preliminary testing it was not competitive enough to merit a full comparison.

We compare three different approaches using BONMIN. BONMIN-BB uses BONMIN's continuous branch-and-bound algorithm, BONMIN-OA uses BONMIN's outer approximation algorithm, and BONMIN-OA-D uses BONMIN's outer approximation algorithm to which we manually provide the disaggregated form of the $\mathcal{L}$ cone. These algorithms are described in more detail at [14].

In Pajarito, we use ECOS as our open-source conic solver and experiment with both Cbc and GLPK as MILP solvers. Table 3.3 summarizes the results on the benchmark library and Figure 3-4 illustrates the relative performance across instances with a performance profile plot. BONMIN fails a large number of times and solves significantly fewer instances than Pajarito. Cbc generally performs faster than GLPK but is less reliable (on two instances, Cbc returned solutions that violated integrality restrictions).

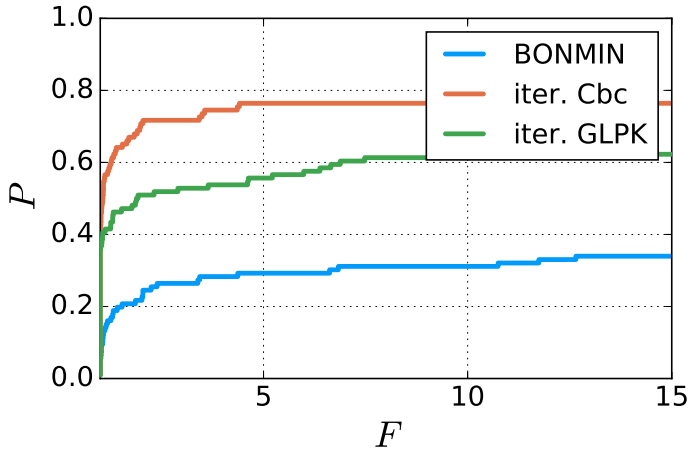


Figure 3-4: Performance profile for solution time on the MISOCP library, for BONMIN (best of 3 algorithms) and open-source default iterative Pajarito-based solvers

	termination status counts				
solver	conv	wrong	not conv	limit	time(s)
SCIP	78	1	0	41	43.36
CPLEX	96	3	5	16	14.30
Paj-iter-CPLEX-MOSEK	96	1	0	23	38.70
Paj-MSD-CPLEX-MOSEK	101	0	0	19	18.12

Table 3.4: Termination statuses and shifted geometric mean of solve time on the MISOCP library, for SCIP and CPLEX MISOCP solvers and default MSD and iterative Pajarito-based solvers using CPLEX and MOSEK

Comparisons with CPLEX and SCIP

Here we compare our iterative and MSD algorithms using CPLEX as a MILP solver and Mosek as a conic solver with SCIP and CPLEX as MISOCP solvers. We note that SCIP, although an academic solver, is not available under an open source license.

Table 3.4 summarizes the results on the benchmark library and Figure 3-5 illustrates the relative performance across instances with a performance profile plot. CPLEX's MISOCP solver is generally the fastest, although Pajarito's MSD algorithm solves more instances in the time limit and performs competitively on instances which are solved by both CPLEX and Pajarito.

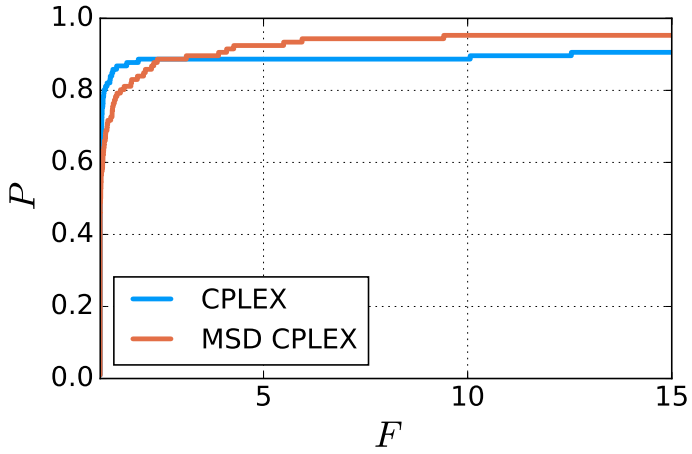


Figure 3-5: Performance profile for solution time on the MISOCP library, for CPLEX MISOCP solver and default MSD Pajarito solver using CPLEX and MOSEK

3.8 Future work

In near future we intend to test Pajarito on a broader collection of problems which include exponential cone and PSD cone constraints. We note that Pajarito has already been independently benchmarked on a suite of MISDP problems by Tristan Gally, reported at the recent SIAM Conference on Optimization in Vancouver. In these tests, the MSD algorithm of Pajarito with subproblem cuts performed competitively with two algorithms implemented in SCIP-SDP, a continuous branch-and-bound algorithm and an LP-based branch-and-bound algorithm using separation cuts. On one family of instances, Pajarito solved the most instances out of all solvers tested. On another family, Pajarito performed poorly because the MILP solver rarely found feasible solutions and hence subproblem cuts were rarely added. It is clear that subproblem cuts will be useful in general-purpose mixed-integer conic solvers, but there remains more work to do on deciding when and how frequently to add cuts, questions which we have not considered here.

An interesting direction of future work would be to experiment with first-order solvers like SCS [74] which would return less accurate solutions but may be able to efficiently warm-start when solving the sequences of subproblems that arise within LP-based branch-and-bound. Our analysis of the algorithm has assumed so far that the solution from the conic solver is exact, which seems to be sufficient for solvers

like Mosek and ECOS which are based on interior-point methods. In order to achieve finite-time convergence of our method when using first order solvers, it may be necessary to consider error bounds on the conic solver, e.g., by using Renegar's condition measure [77, 23, 33].

Chapter 4

Mixed-integer convex representability

4.1 Preliminaries

By $\mathbb{N}$ we will refer to the nonnegative integers $\{0, 1, 2, \ldots\}$. We will often work with projections of a set $M \subseteq \mathbb{R}^{n+p+d}$ for some $n, p, d \in \mathbb{N}$. We identify the variables in $\mathbb{R}^n$, $\mathbb{R}^p$ and $\mathbb{R}^d$ of this set as $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ and we let

$$\operatorname{proj}_x(M) = \left\{\boldsymbol{x} \in \mathbb{R}^n : \exists (\boldsymbol{y}, \boldsymbol{z}) \in \mathbb{R}^{p+d} \text{ s.t. } (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M\right\}.$$

We similarly define $\operatorname{proj}_y(M)$ and $\operatorname{proj}_z(M)$.

Definition 5. *Let $M \subseteq \mathbb{R}^{n+p+d}$ be a closed, convex set and $S \subseteq \mathbb{R}^n$. We say M induces an MICP formulation of S if and only if*

$$S = \operatorname{proj}_x\left(M \cap \left(\mathbb{R}^{n+p} \times \mathbb{Z}^d\right)\right). \tag{4.1}$$

*A set $S \subseteq \mathbb{R}^n$ is **MICP representable** if and only if there exists an MICP formulation of S. If such formulation exists for a closed polyhedron M then we say S is (additionally) MILP representable.*

Definition 6. *For a set of integral vectors $\boldsymbol{z}^1, \boldsymbol{z}^2, \ldots, \boldsymbol{z}^k \in \mathbb{Z}^d$ we define the integral cone* $\operatorname{intcone}(\boldsymbol{z}^1, \boldsymbol{z}^2, \ldots, \boldsymbol{z}^k) = \{\sum_{i=1}^k \lambda_i \boldsymbol{z}^i : \lambda_i \in \mathbb{N}, i \in [\![k]\!]\}$.

4.2 Bounded and other restricted MICP representability results

Definition 7. *A set S is **bounded MICP (MILP)** representable if there exists an MICP (MILP) formulation which satisfies $\left|\text{proj}_z\left(M \cap \left(\mathbb{R}^{n+p} \times \mathbb{Z}^d\right)\right)\right| < \infty$. That is, there are only finitely many feasible assignments of the integer variables $\boldsymbol{z}$.*

It is easy to see that bounded MICP formulations can represent *at most* a finite union of projections of closed, convex sets. To date, however, there are no precise necessary conditions over these sets for the existence of a bounded MICP formulation. For instance, Ceria and Soares [19] provide an MICP formulation for the finite union of closed, convex sets under the condition that the sets have the same *recession cone* (set of unbounded directions). In the following lemma we close this gap and give a simple, explicit formulation for any finite union of projections of closed, convex sets without assumptions on recession directions.

Lemma 16. *$S \subseteq \mathbb{R}^n$ is bounded MICP representable if and only if there exist nonempty, closed, convex sets $T_1, T_2, \ldots, T_k \subset \mathbb{R}^{n+p}$ for some $p, k \in \mathbb{N}$ such that $S = \bigcup_{i\in[\![k]\!]} \text{proj}_x T_i$. In particular $\boldsymbol{x} \in S$ iff there exist $\boldsymbol{x}^i \in \mathbb{R}^n, \boldsymbol{y}^i \in \mathbb{R}^p$ for $i \in [\![k]\!]$ and $\boldsymbol{t} \in \mathbb{R}^k, \boldsymbol{z} \in \mathbb{Z}^k$ such that*

$$\boldsymbol{x} = \sum_{i\in[\![k]\!]} \boldsymbol{x}^i, \quad (\boldsymbol{x}^i, \boldsymbol{y}^i, z_i) \in \hat{T}_i \; \forall i \in [\![k]\!], \quad \sum_{i\in[\![k]\!]} z_i = 1, \; \boldsymbol{0} \leq \boldsymbol{z} \leq \boldsymbol{1}, \tag{4.2a}$$

$$||\boldsymbol{x}^i||_2^2 \leq z_i t_i, \quad \forall i \in [\![k]\!], \boldsymbol{t} \geq 0 \tag{4.2b}$$

where $\hat{T}_i$ is the closed conic hull of T_i, i.e., $\text{cl}(\{(\boldsymbol{x}, \boldsymbol{y}, z) : (\boldsymbol{x}, \boldsymbol{y})/z \in T_i, z > 0\})$. This defines a bounded MICP representation of S.

Proof. Note that the constraints define a convex set because the conic hull of a convex set is convex, and $||\boldsymbol{x}^i||_2^2 \leq z_i t_i$ is a form of the rotated second-order cone, which is also convex. Any feasible assignment of the integer vector $\boldsymbol{z}$ has at most one nonzero component. Without loss of generality we may take this to be the first component, so $z_1 = 1$. Since t_i is unrestricted in the positive direction, the constraint $||\boldsymbol{x}^1||_2^2 \leq t_i$

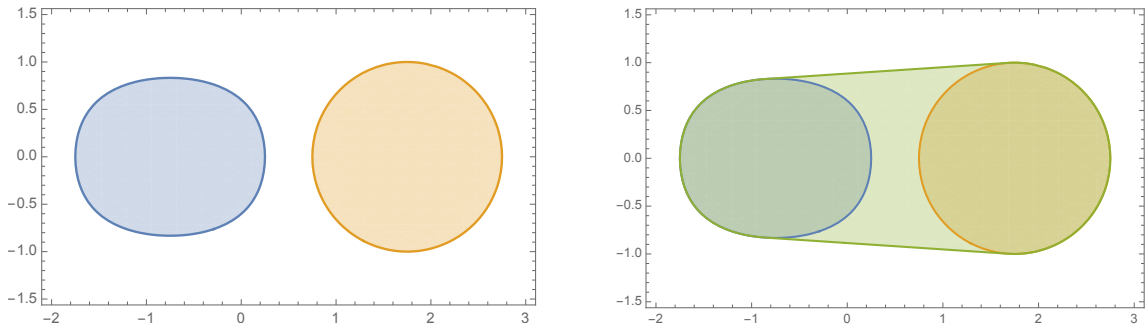


Figure 4-1: On the left, two convex sets $\{(x, y) : (x + 0.75)^2 + e^{y^2} \leq 2\}$ and $\{(x, y) : (x - 1.75)^2 + y^2 \leq 1\}$, resp. By using a set of mixed-integer convex constraints, one can represent the nonconvex constraint that (x, y) belongs to the union of the two sets. When the integer restrictions in this formulation are relaxed, one obtains the convex hull of the two sets, shaded in green on the right.

imposes no restrictions on the vector $\boldsymbol{x}^1$ and $\boldsymbol{x}^1 \in \text{proj}_x T_1$ iff there exists $\boldsymbol{y}^1 \in \mathbb{R}^p$ such that $(\boldsymbol{x}^1, \boldsymbol{y}^1, 1) \in \hat{T}_1$. For $i > 1$, the constraint $||\boldsymbol{x}^i||_2^2 \leq 0$ implies $\boldsymbol{x}^i = 0$, and this is feasible because $(\mathbf{0}, \mathbf{0}, 0) \in \hat{T}_i$ given $\hat{T}_i$ is nonempty by assumption. □

A desirable property of *MILP* formulations is *local idealness*, when every extreme point of the convex set defined by relaxing the integrality constraints in fact satisfies that all integer variables take integer values [86]. (Hence, optimizing a linear function over the relaxation would yield an optimal solution which satisfies integrality.) Local idealness implies that the convex relaxation corresponds precisely to the convex hull of the union; see Figure 4-1 for an illustration. The formulation by Ceria and Soares for unions of convex sets with a common recession cone is locally ideal, and the following lemma shows that our formulation is as well. We would advise readers to consider [46] and Section 2.5.1 before applying this formulation in practice.

Lemma 17. *The formulation in Lemma 16 is locally ideal. We will show that any nonintegral feasible point is a convex combination of integral points.*

Proof. For simplicity of exposition suppose $k = 2$. Suppose we have a feasible, non-integral point $\boldsymbol{\beta} = (\boldsymbol{x}, \boldsymbol{x}^1, \boldsymbol{y}^1, z_1, t_1, \boldsymbol{x}^2, \boldsymbol{y}^2, z_2, t_2)$, so $0 < z_1, z_2 < 1$ and $z_1 + z_2 = 1$. Define

$$\boldsymbol{\beta}^1 = (\boldsymbol{x}^1/z_1, \boldsymbol{x}^1/z_1, \boldsymbol{y}^1/z_1, 1, t_1/z_1, \mathbf{0}, \mathbf{0}, 0, 0) \tag{4.3}$$

and

$$\boldsymbol{\beta}^2 = (\boldsymbol{x}^2/z_2, \mathbf{0}, \mathbf{0}, 0, 0, \boldsymbol{x}^2/z_2, \boldsymbol{y}^2/z_2, 1, t_2/z_2). \tag{4.4}$$

Then by construction $\boldsymbol{\beta} = z_1\boldsymbol{\beta}^1 + z_2\boldsymbol{\beta}^2$. One may verify as well that $\boldsymbol{\beta}^1$ and $\boldsymbol{\beta}^2$ satisfy conditions (4.2). □

Known MICP representability results for unbounded integers are more limited. For the case in which M is a rational polyhedron Jerowslow and Lowe [49] showed that a set $S \subseteq \mathbb{R}^n$ is rational MILP representable if and only if there exist integer vectors $\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t \subseteq \mathbb{Z}^n$ and rational polytopes $S_1, S_2, \ldots, S_k$ such that

$$S = \bigcup_{i \in [\![k]\!]} S_i + \text{intcone}(\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t). \tag{4.5}$$

Characterization (4.5) does not hold in general for non-polyhedral M. However, using results from [25] it is possible to show that it holds for some pure integer cases as well.

Example 1. *Theorem 6 in [25] can be used to show that for any $\alpha > 0$, $P_\alpha := \{\boldsymbol{x} \in \mathbb{Z}^2 : x_1 x_2 \geq \alpha\}$ satisfies a representation of the form* (4.5) *with each polyhedron S_i containing a single integer vector for each $i \in [\![k]\!]$.*

The only mixed-integer and non-polyhedral result we are aware of is a characterization of the form (4.5) when M is the intersection of a rational polyhedron with an ellipsoidal cylinder having a rational recession cone [24]. An identical proof also holds when the recession cone of M is a rational subspace and M is contained in a rational polyhedron with the same recession cone as M. We can further extend this result to the following simple proposition.

Proposition 1. *If M induces an MICP-formulation of S and $M = C + K$ where C is a compact convex set and K is a rational polyhedral cone, then for some $k, t \in \mathbb{N}$ there exist compact convex sets $S_1, S_2, \ldots, S_k$ and integer vectors $\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t \subseteq \mathbb{Z}^n$ such that*

$$S = \bigcup_{i \in [\![k]\!]} S_i + \text{intcone}(\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t). \tag{4.6}$$

Proof. This argument is an extension of Theorem 11.6 of [86]. Suppose $M = C + K$ where C is a compact convex set and K is a polyhedral cone generated by rational rays $(\boldsymbol{r}_x^1, \boldsymbol{r}_y^1, \boldsymbol{r}_z^1), (\boldsymbol{r}_x^2, \boldsymbol{r}_y^2, \boldsymbol{r}_z^2), \ldots, (\boldsymbol{r}_x^t, \boldsymbol{r}_y^t, \boldsymbol{r}_z^t)$. (We may assume without loss of generality that these rays furthermore have integer components). We will prove that there exist sets $S_1, \ldots, S_k$ such that

$$S = \text{proj}_x \left(M \cap \left(\mathbb{R}^{n+p} \times \mathbb{Z}^d \right) \right) = \bigcup_{i \in [\![k]\!]} S_i + \text{intcone}(\boldsymbol{r}_x^1, \boldsymbol{r}_x^2, \ldots, \boldsymbol{r}_x^t), \tag{4.7}$$

where each S_i is a projection of a closed convex set.

For any $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*) \in M$ there exist a finite (via Carathéodory) set of extreme points $(\boldsymbol{x}^1, \boldsymbol{y}^1, \boldsymbol{z}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2, \boldsymbol{z}^2), \ldots, (\boldsymbol{x}^w, \boldsymbol{y}^w, \boldsymbol{z}^w)$ of C and nonnegative multipliers $\boldsymbol{\lambda}$, $\boldsymbol{\gamma}$ with $\sum_{i \in [\![w]\!]} \lambda_i = 1$ such that

$$(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*) = \sum_{i \in [\![w]\!]} \lambda_i (\boldsymbol{x}^i, \boldsymbol{y}^i, \boldsymbol{z}^i) + \sum_{j \in [\![t]\!]} \gamma_j (\boldsymbol{r}_x^j, \boldsymbol{r}_y^j, \boldsymbol{r}_z^j). \tag{4.8}$$

Define

$$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}) = \sum_{i \in [\![w]\!]} \lambda_i (\boldsymbol{x}^i, \boldsymbol{y}^i, \boldsymbol{z}^i) + \sum_{j \in [\![t]\!]} (\gamma_j - \lfloor \gamma_j \rfloor)(\boldsymbol{r}_x^j, \boldsymbol{r}_y^j, \boldsymbol{r}_z^j) \tag{4.9}$$

and

$$(\boldsymbol{x}^\infty, \boldsymbol{y}^\infty, \boldsymbol{z}^\infty) = \sum_{j \in [\![t]\!]} \lfloor \gamma_j \rfloor (\boldsymbol{r}_x^j, \boldsymbol{r}_y^j, \boldsymbol{r}_z^j) \tag{4.10}$$

so that $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*) = (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}) + (\boldsymbol{x}^\infty, \boldsymbol{y}^\infty, \boldsymbol{z}^\infty)$.

Note that $(\boldsymbol{x}^\infty, \boldsymbol{y}^\infty, \boldsymbol{z}^\infty) \in \text{intcone}((\boldsymbol{r}_x^1, \boldsymbol{r}_y^1, \boldsymbol{r}_z^1), \ldots, (\boldsymbol{r}_x^t, \boldsymbol{r}_y^t, \boldsymbol{r}_z^t)) =: M^\infty$ and $\hat{\boldsymbol{z}} = \boldsymbol{z}^* - \boldsymbol{z}^\infty \in \mathbb{Z}^d$, so $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$ belongs to a bounded set

$$\hat{M} = (C + B) \cap (\mathbb{R}^{n+p} \times \mathbb{Z}^d), \tag{4.11}$$

where $B = \{\sum_{j \in [\![t]\!]} \gamma_j (\boldsymbol{r}_{\boldsymbol{x}}^j, \boldsymbol{r}_{\boldsymbol{y}}^j, \boldsymbol{r}_{\boldsymbol{z}}^j) : \boldsymbol{0} \leq \boldsymbol{\gamma} \leq \boldsymbol{1}\}$. Since this decomposition holds for any points $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*) \in M$, it follows that $M \subseteq \hat{M} + M^\infty$. Since $\hat{M}$ is bounded, $\hat{M}$ is a finite union of bounded convex sets. Also $\hat{M} + M^\infty \subseteq M$ is easy to show, so we've demonstrated $M = \hat{M} + M^\infty$. The statement (4.7) follows from projection of

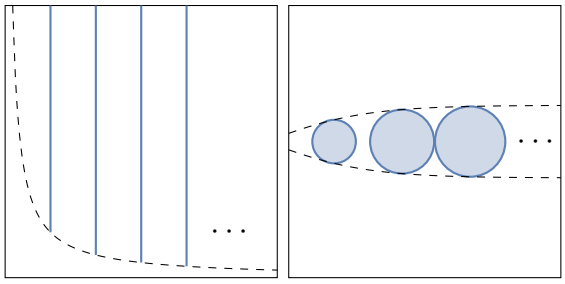


Figure 4-2: The mixed-integer hyperbola and the collection of balls with increasing and concave radius are mixed-integer convex representable but do not satisfy the conditions of Proposition 1.

M onto the x variables. □

Unfortunately, MICP-representable sets in general may not have a representation of the form (4.6), even when S_i is allowed to be any convex set (not necessarily bounded). We illustrate this with a simple variation on the pure-integer example above.

Example 2. *Let* $S := \{\boldsymbol{x} \in \mathbb{N} \times \mathbb{R} : x_1 x_2 \geq 1\}$ *be the set depicted in Figure 4-2. For each* $z \in \mathbb{N}, z \neq 0$ *let* $A_z := \{x \in \mathbb{R}^2 : x_1 = z,\ x_2 \geq 1/z\}$ *so that* $S = \bigcup_{z=1}^{\infty} A_z$. *Suppose for contradiction that* S *satisfies* (4.5) *for convex sets* S_i. *By convexity of* S_i *and finiteness of* k *there exists* $z_0 \in \mathbb{Z}$ *such that* $\bigcup_{i \in [\![k]\!]} S_i \subset \bigcup_{z \in [\![z_0 - 1]\!]} A_z$. *Because* $\min_{\boldsymbol{x} \in A_{z_0}} x_2 < \min_{\boldsymbol{x} \in A_z} x_2$ *for all* $z \in [\![z_0 - 1]\!]$ *we have that there exists* $j \in [\![t]\!]$ *such that the second component of* $\boldsymbol{r}^j$ *is strictly negative. However, this implies that there exists* $\boldsymbol{x} \in S$ *such that* $x_2 < 0$ *which is a contradiction with the definition of* S.

4.3 Rational MICP

In this section, we address the case of infinitely many possible integer assignments by considering a restricted subset of these formulations which we call *rational MICP*. Rational MICP provides the additional useful structure that the set of feasible integer points has an integral recession direction. This definition includes as a subset formulations defined by rational polyhedra, i.e., rational MILP formulations.

4.3.1 Preliminaries

Definition 8. *Given an MICP formulation induced by M, we say $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$ is a **$\boldsymbol{z}$-projected set** of the formulation when $A_{\boldsymbol{z}} \neq \emptyset$.*

Definition 9. *We say that an unbounded convex set $C \subseteq \mathbb{R}^d$ is **rationally unbounded** if the image C' of any rational affine mapping of C, is either bounded or there exists $\boldsymbol{r} \in \mathbb{Z}^d \setminus \{\boldsymbol{0}\}$ such that $\boldsymbol{x} + \lambda \boldsymbol{r} \in C'$ from any $\boldsymbol{x} \in C'$ and $\lambda \geq 0$. (i.e., $\boldsymbol{r}$ is a recession direction.)*

Definition 10. *We say that a set S is **rational MICP representable** if it has a MICP representation induced by the set M and the convex set $\text{proj}_z(M)$ is either bounded or rationally unbounded.*

We now establish a number of properties of rational MICP that we use in our main results.

Lemma 18. *If $C_1 \subseteq \mathbb{R}^{n_1}$ and $C_2 \subseteq \mathbb{R}^{n_2}$ are bounded or rationally unbounded sets, then $C_1 \times C_2$ is bounded or rationally unbounded.*

Proof. Follows from the definition. □

Lemma 19. *Let $S_1, S_2 \subseteq \mathbb{R}^n$ be nonempty rational MICP representable sets. Then their union $S = S_1 \cup S_2$ is rational MICP representable.*

Proof. For some $p_1, p_2, d_1, d_2 \in \mathbb{N}$ and closed convex sets M_1 and M_2 we have $\boldsymbol{x} \in S_1$ iff $\exists \boldsymbol{y}^1 \in \mathbb{R}^{p_1}, \boldsymbol{z}^1 \in \mathbb{Z}^{d_1}$ such that $(\boldsymbol{x}, \boldsymbol{y}^1, \boldsymbol{z}^1) \in M_1$ and $\boldsymbol{x} \in S_2$ iff $\exists \boldsymbol{y}^2 \in \mathbb{R}^{p_2}, \boldsymbol{z}^2 \in \mathbb{Z}^{d_2}$ such that $(\boldsymbol{x}, \boldsymbol{y}^2, \boldsymbol{z}^2) \in M_2$. We claim that $\boldsymbol{x} \in S$ iff there exist $\boldsymbol{x}^1, \boldsymbol{x}^2 \in \mathbb{R}^n, \boldsymbol{y}^1 \in$

$\mathbb{R}^{p_1}, \boldsymbol{y}^2 \in \mathbb{R}^{p_2}, t \in \mathbb{R}, \boldsymbol{z}^1 \in \mathbb{Z}^{d_1}, \boldsymbol{z}^2 \in \mathbb{Z}^{d_2}, z' \in \mathbb{Z}$ such that

$$(\boldsymbol{x}^1, \boldsymbol{y}^1, \boldsymbol{z}^1) \in M_1, \tag{4.12a}$$

$$(\boldsymbol{x}^2, \boldsymbol{y}^2, \boldsymbol{z}^2) \in M_2, \tag{4.12b}$$

$$||\boldsymbol{x} - \boldsymbol{x}^1||_2^2 \leq tz', \tag{4.12c}$$

$$||\boldsymbol{x} - \boldsymbol{x}^2||_2^2 \leq t(1 - z'), \tag{4.12d}$$

$$t \geq 0, \tag{4.12e}$$

$$0 \leq z' \leq 1. \tag{4.12f}$$

Equivalence follows from noting that $z' = 0$ implies $\boldsymbol{x} \in S_1$ and $z' = 1$ implies $\boldsymbol{x} \in S_2$ and that, e.g., for any point $\boldsymbol{x}^1 \in S_1$ there exist solutions satisfying conditions (4.12) and integrality on $\boldsymbol{z}^1, \boldsymbol{z}^2$ with $z' = 0$.

The conditions (4.12) define a closed convex set $M \subseteq \mathbb{R}^{3n+p_1+p_2+d_1+d_2+2}$. Let $C_1 = \text{proj}_z(M_1), C_2 = \text{proj}_z(M_2)$ and $C = \text{proj}_z(M)$ be the projections onto the last $d_1, d_2,\ d_1 + d_2 + 1$ variables of respectively. We may see that $C = C_1 \times C_2 \times [0, 1]$, so by Lemma 18, if C_1 and C_2 are bounded or rationally unbounded, then so is C. Therefore S is rational MICP representable.

□

Lemma 20. *If S is rational MILP representable then it is rational MICP representable.*

Proof. If S has a rational MILP representation induced by a closed rational polyhedron M then $C = \text{proj}_z(M)$ is a closed rational polyhedron. Closed rational polyhedra are either bounded or have a rational recession direction. □

Corollary 1. *If S_1 and S_2 are rational MILP representable then $S_1 \cup S_2$ is rational MICP representable.*

Lemma 21. *Suppose $S \subseteq \mathbb{R}^n$ has a rational MICP representation induced by M with MICP dimension d, and let $\mathcal{R} : \mathbb{R}^d \to \mathbb{R}^d$ be an invertible affine transformation. Then*

the set $S' \subseteq \mathbb{R}^n$ *defined by*

$$\boldsymbol{x} \in S' \text{ iff } \exists \boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, \mathcal{R}(\boldsymbol{z})) \in M, \tag{4.13}$$

is rational MICP representable with MICP dimension at most d*. Furthermore, if* $\mathcal{R}$ *maps integers to integers, i.e.,* $\mathcal{R}(\mathbb{Z}^d) \subseteq \mathbb{Z}^d$*, then* $S' \subseteq S$*, and if the inverse transformation maps integers to integers, i.e.,* $\mathcal{R}^{-1}(\mathbb{Z}^d) \subseteq \mathbb{Z}^d$ *then* $S \subseteq S'$*.*

Proof. Define the extended affine transformation $\mathcal{R}_{\text{ext}} : \mathbb{R}^{n+p+d} \to \mathbb{R}^{n+p+d}$ by $\mathcal{R}_{\text{ext}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = (\boldsymbol{x}, \boldsymbol{y}, \mathcal{R}(\boldsymbol{z}))$. Let $M' = \mathcal{R}_{\text{ext}}^{-1}(M)$ be the image of M under $\mathcal{R}_{\text{ext}}^{-1}$. We claim that M' induces a rational MICP formulation of S'. Note $(\boldsymbol{x}, \boldsymbol{y}, \mathcal{R}(\boldsymbol{z})) \in M$ iff $\mathcal{R}_{\text{ext}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M$ iff $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M'$, so

$$\boldsymbol{x} \in S' \text{ iff } \exists \boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M'. \tag{4.14}$$

Also, M' is convex since it is the image of a convex set under an affine transformation. Finally, $\text{proj}_z(M')$ is the image under the rational affine mapping $\mathcal{R}^{-1}$ of $\text{proj}_z(M)$, so $\text{proj}_z(M')$ is either bounded or rationally unbounded. The set M' is closed because M is closed and $\mathcal{R}$ is invertible.

For the inclusion statements, suppose $\boldsymbol{x} \in S'$ with corresponding $\boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d$ such that $(\boldsymbol{x}, \boldsymbol{y}, \mathcal{R}(\boldsymbol{z})) \in M$. If $\mathcal{R}(\boldsymbol{z}) \in \mathbb{Z}^d$ then $\boldsymbol{x} \in S$. Suppose now $\boldsymbol{x} \in S$ with corresponding $\boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d$ such that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M$. If $\mathcal{R}^{-1}(\boldsymbol{z}) \in \mathbb{Z}^d$ then $\boldsymbol{x} \in S'$.

□

Lemma 22. *Suppose* $S \subseteq \mathbb{R}^n$ *has a rational MICP representation induced by* M *with MICP dimension* d*. Then the set* $S' \subseteq \mathbb{R}^n$ *defined by*

$$\boldsymbol{x} \in S' \text{ iff } \exists \boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1} \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M, \tag{4.15}$$

is rational MICP representable with MICP dimension at most $d - 1$*.*

Proof. Note that S' can be called a relaxation of S since it is derived by relaxing the integrality restriction on the variable z_0, so clearly $S \subseteq S'$. Let C be the projection of

M onto the last d variables and C' the projection of M onto the last $d-1$ variables. We need only show that C' is either bounded or rationally unbounded. This follows since C' is the image of C under the rational linear mapping that discards the first dimension and acts as an identity on the remaining dimensions. Note that the set M does not change in the definition of S'. □

In the following lemma, following Rockafellar [78] we refer to $\text{aff}(\cdot)$ as the affine hull of a set, $\text{int}(\cdot)$ and $\text{relint}(\cdot)$ as the interior and relative interior, respectively, $\text{dom}(\cdot)$ the domain of a function and $\partial\cdot$ the subdifferential of a function.

Lemma 23. *Let $C \subseteq \mathbb{R}^d$ be a convex set, $h : C \to \mathbb{R}$ a nonpositive convex function and $(\boldsymbol{x}^i)_{i\in[\![k]\!]} \subset C$ such that $h(\boldsymbol{x}^1) = 0$ and $x^1 \in \text{relint}\left(\text{aff}\left((\boldsymbol{x}^i)_{i\in[\![k]\!]}\right) \cap C\right)$. Then $h(\boldsymbol{x}) = 0$ for all $x \in \text{aff}\left((\boldsymbol{x}^i)_{i\in[\![k]\!]}\right) \cap C$.*

Proof. After an affine transformation we may assume without loss of generality that $\text{aff}\left((\boldsymbol{x}^i)_{i\in[\![k]\!]}\right) = \mathbb{R}^d$. Let $\bar{h} : \mathbb{R}^d \to \mathbb{R} \cap \{\infty\}$ so that $\bar{h}(\boldsymbol{x}) = h(\boldsymbol{x})$ for all $\boldsymbol{x} \in C$ and $\bar{h}(\boldsymbol{x}) = \infty$ otherwise. We have that $\boldsymbol{x}^1 \in \text{int}\left(\text{dom}\left(\bar{h}\right)\right) = \text{int}\,(C)$ and hence $\partial\bar{h}\,(\boldsymbol{x}^1)$ is nonempty and bounded [78]. If there exist $\boldsymbol{u} \in \partial\bar{h}\,(x^1) \setminus \{0\}$ then for sufficiently small $\varepsilon > 0$ we have $\boldsymbol{x}^1 + \varepsilon\boldsymbol{u} \in \text{int}\,(C)$ and $0 \geq \bar{h}\,(\boldsymbol{x}^1 + \varepsilon\boldsymbol{u}) \geq \bar{h}\,(\boldsymbol{x}^1) + \varepsilon||\boldsymbol{u}||_2 > 0$, which is a contradiction. Hence $\partial\bar{h}\,(\boldsymbol{x}^1) = \{\mathbf{0}\}$ so $h\,(\boldsymbol{x}) = \bar{h}\,(\boldsymbol{x}) \geq \bar{h}\,(\boldsymbol{x}^1) = 0$ for all $\boldsymbol{x} \in C$. □

Lemma 24. *Let $\boldsymbol{r} \in \mathbb{Z}^d$ nonzero with $\gcd(r_1, \ldots, r_d) = 1$. Then there exists a $d \times d$ unimodular matrix $\boldsymbol{U}$ with $\boldsymbol{r}$ as the last column.*

Proof. Recall [73, p. 189]: A square invertible, integer matrix $\boldsymbol{H} \in \mathbb{Z}^{d\times d}$ is said to be in Hermite normal form if it is 1) lower triangular, 2) has positive entries on the diagonal, and 3) has nonpositive entries off the diagonal with magnitude smaller than the element on the diagonal for the same row. Then if $\boldsymbol{A}$ is a square invertible matrix, there exists a unimodular matrix $\boldsymbol{U}$ such that $\boldsymbol{AU} = \boldsymbol{H}$. Since $\boldsymbol{A}$ is invertible, we have $\boldsymbol{U} = \boldsymbol{A}^{-1}\boldsymbol{H}$. Since $\boldsymbol{H}$ is lower triangular, the last column of $\boldsymbol{U}$ is a positive integer multiple (H_{dd}) of the last column of $\boldsymbol{A}^{-1}$. We'll use this property to prove the claim.

Now, let $\boldsymbol{B}$ be a rational invertible matrix with $\boldsymbol{r}$ on the last column (which is always possible because we can complete $\boldsymbol{r}$ into a rational basis of $\mathbb{R}^d$). The matrix $\boldsymbol{B}$ is rational so $\boldsymbol{B}^{-1}$ is as well. Let $q \in \mathbb{N}$ be a positive number such that $q\boldsymbol{B}^{-1}$ has all integer entries and consider the decomposition such that $(q\boldsymbol{B}^{-1})\boldsymbol{U} = \boldsymbol{H}$, i.e., $\boldsymbol{U} = \frac{1}{q}\boldsymbol{B}\boldsymbol{H}$.

We see via this decomposition that there exists a unimodular matrix $\boldsymbol{U}$ with last column equal to the vector $(H_{dd}/q)r$. The unique solution to the system $\boldsymbol{U}\boldsymbol{x} = \boldsymbol{r}$ is the vector $(q/H_{dd})\boldsymbol{e}(d)$, which must be integral since $\boldsymbol{U}$ is unimodular. Therefore (q/H_{dd}) is a positive integer. Entries of unimodular matrices must be integral, so we require $(H_{dd}/q)\boldsymbol{r} \in \mathbb{Z}^d$. Since $\gcd(r_1, \ldots, r_d) = 1$ by assumption, we must have that H_{dd}/q is also a positive integer. It follows that $H_{dd}/q = 1$. □

Lemma 25. *Let M be an MICP formulation for $S \subseteq \mathbb{R}^n$. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$. Then for any $\boldsymbol{c} \in \mathbb{R}^n$, the function $g_{\boldsymbol{c}}(\boldsymbol{z}) : C \to \mathbb{R} \cup \{\infty\}$ defined by $g_{\boldsymbol{c}}(\boldsymbol{z}) = \sup\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in A_{\boldsymbol{z}}\}$ is concave.*

Proof. Note $A_{\boldsymbol{z}}$ is nonempty for $\boldsymbol{z} \in C$, so $g_{\boldsymbol{c}}(\boldsymbol{z})$ is well defined. Choose any two $\boldsymbol{z}^1, \boldsymbol{z}^2 \in C$ and $\lambda \in [0, 1]$. We will show $g_{\boldsymbol{c}}(\lambda\boldsymbol{z}^1 + (1-\lambda)\boldsymbol{z}^2) \geq \lambda g_{\boldsymbol{c}}(\boldsymbol{z}^1) + (1-\lambda)g_{\boldsymbol{c}}(\boldsymbol{z}^2)$. Let $\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3, \cdots$ and $\boldsymbol{y}^1, \boldsymbol{y}^2, \boldsymbol{y}^3, \cdots$ be sequences contained in $A_{\boldsymbol{z}^1}$ and $A_{\boldsymbol{z}^2}$ respectively such that $\lim_{i\to\infty} \boldsymbol{c}^T\boldsymbol{x}^i = g_{\boldsymbol{c}}(\boldsymbol{z}^1)$ and $\lim_{i\to\infty} \boldsymbol{c}^T\boldsymbol{y}^i = g_{\boldsymbol{c}}(\boldsymbol{z}^2)$. Since M is convex, it follows that for each i, $\lambda\boldsymbol{x}^i + (1-\lambda)\boldsymbol{x}^i \in A_{\lambda\boldsymbol{z}^1+(1-\lambda)\boldsymbol{z}^2}$, so

$$g_{\boldsymbol{c}}(\lambda\boldsymbol{z}^1 + (1-\lambda)\boldsymbol{z}^2) \geq \lim_{i\to\infty} \boldsymbol{c}^T(\lambda\boldsymbol{x}^i + (1-\lambda)\boldsymbol{y}^i) = \lambda g_{\boldsymbol{c}}(\boldsymbol{z}^1) + (1-\lambda)g_{\boldsymbol{c}}(\boldsymbol{z}^2). \quad (4.16)$$

□

Lemma 26. *Let M be an MICP formulation for $S \subseteq \mathbb{R}^n$. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$. Then for any $\boldsymbol{c} \in \mathbb{R}^n$, the function $f_{\boldsymbol{c}}(\boldsymbol{z}) : C \to \mathbb{R} \cup \{\infty\}$ defined by $f_{\boldsymbol{c}}(\boldsymbol{z}) = \inf\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in A_{\boldsymbol{z}}\}$ is convex.*

Proof. Analogous to proof of Lemma 25. □

Lemma 27. *Let $l \in \mathbb{R}$ and let $f : [l, \infty) \to \mathbb{R} \cup \{-\infty\}$ be an extended-value concave function. If $\exists x > x' \in [l, \infty)$ such that $f(x) < f(x')$ then $\lim_{x\to\infty} f(x) = -\infty$.*

Proof. Look at the set of supergradients at x, which is in the relative interior of the domain. A supergradient with zero or positive slope would contradict $f(x) < f(x')$, so there has to be a supergradient with negative slope which forces $f(x)$ to $-\infty$ as $x \to \infty$. □

Lemma 28. *Suppose $S \subseteq \mathbb{R}^n$ is a closed set with MICP representation induced by M. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$ and $C = \text{proj}_z(M)$. Then $S = \bigcup_{\boldsymbol{z} \in (C \cap \mathbb{Z}^d)} \text{cl}(A_{\boldsymbol{z}})$.*

Proof. By definition, $S = \bigcup_{\boldsymbol{z} \in (C \cap \mathbb{Z}^d)} A_{\boldsymbol{z}}$, so $S \subseteq \bigcup_{\boldsymbol{z} \in (C \cap \mathbb{Z}^d)} \text{cl}(A_z)$. Fix $\boldsymbol{z} \in C \cap \mathbb{Z}^d$. Since $A_{\boldsymbol{z}} \subseteq S$ and S is closed, it follows that $\text{cl}(A_{\boldsymbol{z}}) \subseteq S$, hence the desired statement holds. □

Lemma 29. *Let $S \subseteq \mathbb{R}^n$ be a set which has an MICP representation induced by M. Let $C = \text{proj}_z(M)$ and $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$. Let $\boldsymbol{r} \in \mathbb{Z}^d$ and $\boldsymbol{z} \in C \cap \mathbb{Z}^d$. If the set $\ell_{\boldsymbol{z}} = \{\boldsymbol{z} + \lambda \boldsymbol{r} : \lambda \in \mathbb{R}\} \cap C$ is unbounded and $\exists$ a bounded set $T \subset \mathbb{R}^n$ and λ' such that $\forall \lambda \geq \lambda'$ with $\lambda \in \mathbb{Z}$, we have that $A_{\boldsymbol{z}+\lambda \boldsymbol{r}} \subseteq T$, i.e., the $\boldsymbol{z}$-projected sets along $\ell_{\boldsymbol{z}}$ at integer $\boldsymbol{z}$ points are eventually contained in a bounded region, then $\bigcup_{\boldsymbol{z}' \in \ell_{\boldsymbol{z}}} \text{cl}(A_{\boldsymbol{z}'}) = \bigcup_{\boldsymbol{z}' \in \ell_{\boldsymbol{z}} \cap \mathbb{Z}^d} \text{cl}(A_{\boldsymbol{z}'})$, i.e., integrality can be relaxed along this ray modulo closure.*

Proof. Assume that $\boldsymbol{z} + \lambda \boldsymbol{r} \in C$ for all $\lambda \geq 0$. Fix $\boldsymbol{c} \in \mathbb{R}^n$ and let $g_{\boldsymbol{c}}(\boldsymbol{z}) = \sup\{\boldsymbol{c}^T \boldsymbol{x} : \boldsymbol{x} \in A_{\boldsymbol{z}}\}$. Recall from Lemma 25 that $g_{\boldsymbol{c}}$ is concave over its domain C. Note that since T is bounded, there exist finite bounds α and β (depending on $\boldsymbol{c}$ but not λ, $\boldsymbol{z}$, or $\boldsymbol{r}$) such that $\alpha \leq g_{\boldsymbol{c}}(\boldsymbol{z} + \lambda \boldsymbol{r}) \leq \beta$ whenever $A_{\boldsymbol{z}} \subseteq T$, i.e., whenever $\lambda \geq \lambda'$ and $\lambda \in \mathbb{Z}$. We now claim that $g_{\boldsymbol{c}}(\boldsymbol{z} + \lambda \boldsymbol{r})$ is nondecreasing as a function of λ. If it strictly decreases anywhere, then $\lim_{\lambda \to \infty} g_{\boldsymbol{c}}(\boldsymbol{z} + \lambda \boldsymbol{r}) = -\infty$ by Lemma 27, which leads to a contradiction with the lower bound.

Since the choice of $\boldsymbol{c}$ was arbitrary, it follows that for any $\lambda_1 \leq \lambda_2$ such that $\boldsymbol{z} + \lambda_1 \boldsymbol{r}, \boldsymbol{z} + \lambda_2 \boldsymbol{r} \in \ell_{\boldsymbol{z}}, g_{\boldsymbol{c}}(\boldsymbol{z} + \lambda_1 \boldsymbol{r}) \leq g_{\boldsymbol{c}}(\boldsymbol{z} + \lambda_2 \boldsymbol{r}) \forall \boldsymbol{c} \in \mathbb{R}^n$. Seen as a function of $\boldsymbol{c}$, $g_{\boldsymbol{c}}(\boldsymbol{z})$ is the support function of $A_{\boldsymbol{z}}$, so it follows that $\text{cl}(A_{\boldsymbol{z}+\lambda_1 \boldsymbol{r}}) \subseteq \text{cl}(A_{\boldsymbol{z}+\lambda_2 \boldsymbol{r}})$ [48, p. 225]. The desired claim (in the nontrivial $\subseteq$ direction) follows by noting that for any $\boldsymbol{z}' \in \ell_{\boldsymbol{z}}$, there exists $\boldsymbol{z}'' \in \ell_{\boldsymbol{z}} \cap \mathbb{Z}^d$ such that $\text{cl}(A_{\boldsymbol{z}'}) \subseteq \text{cl}(A_{\boldsymbol{z}''})$.

□

Corollary 2. *Suppose $S \subseteq \mathbb{R}^n$ is a bounded set which has an MICP representation induced by M. Let $C = \text{proj}_z(M)$ and $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$. Let $\boldsymbol{r} \in \mathbb{Z}^d$ and $\boldsymbol{z} \in C \cap \mathbb{Z}^d$. If the set $\ell_{\boldsymbol{z}} = \{\boldsymbol{z} + \lambda \boldsymbol{r} : \lambda \in \mathbb{R}\} \cap C$ is unbounded then $\bigcup_{\boldsymbol{z}' \in \ell_{\boldsymbol{z}}} \text{cl}(A_{\boldsymbol{z}'}) = \bigcup_{\boldsymbol{z}' \in \ell_{\boldsymbol{z}} \cap \mathbb{Z}^d} \text{cl}(A_{\boldsymbol{z}'})$, i.e., integrality can be relaxed along this ray modulo closure.*

Proof. Take $T = S$ in Lemma 29, since $A_{\boldsymbol{z}} \subseteq S \, \forall \boldsymbol{z} \in C \cap \mathbb{Z}^d$. □

4.3.2 Representability of compact sets

Theorem 1. *Suppose $S \subseteq \mathbb{R}^n$ is a compact set which has a rational MICP representation. Then S is a finite union of compact convex sets.*

Proof. Let $M \subseteq \mathbb{R}^{n+p+d}$ be a convex set that induces a rational MICP formulation of S. Let $C = \text{proj}_z(M)$ be the convex set which is bounded or rationally unbounded by assumption. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$, so that, by Lemma 28, $S = \bigcup_{\boldsymbol{z} \in (C \cap \mathbb{Z}^d)} \text{cl}(A_{\boldsymbol{z}})$. If C is bounded then S is precisely a finite union of compact convex sets. Suppose then that C is unbounded. Since C is rationally unbounded, let $\boldsymbol{r} \in \mathbb{Z}^d$ such that the ray $\boldsymbol{z} + \lambda \boldsymbol{r} \in C \, \forall \boldsymbol{z} \in C, \lambda \geq 0$.

We prove now that without loss of generality, we may assume $\boldsymbol{r} = \boldsymbol{e}(1)$. First, rescale $\boldsymbol{r}$ if necessary so that $\gcd(\boldsymbol{r}) = 1$. Then there exists a $d \times d$ unimodular matrix $\boldsymbol{U}$ with $\boldsymbol{r}$ as the first column via Lemma 24 (we can freely permute columns of a unimodular matrix). The columns of the unimodular matrix $\boldsymbol{U}$ form a lattice basis of $\mathbb{Z}^d$, so the matrix can be thought of as mapping from integers expressed in the nonstandard basis (with $\boldsymbol{r}$) to the standard basis. We have that $\boldsymbol{x} \in S$ iff $\exists \boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d$ such that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{U}\boldsymbol{z}) \in M$. Following Lemma 21, we can redefine M to be M' such that $\boldsymbol{x} \in S$ iff $\exists \boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d$ such that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M'$ and $\boldsymbol{e}(1)$ is a ray of $C' := \text{proj}_z(M')$.

Consider the set Q defined by

$$\boldsymbol{x} \in Q \text{ iff } \exists \, \boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1} \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M. \tag{4.17}$$

Note that formulation (4.17) is obtained solely by relaxing the integrality constraint

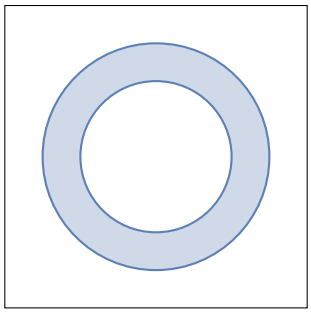


Figure 4-3: The annulus is not rational MICP representable because it is compact but not a finite union of compact convex sets (Theorem 1).

on the first component of the d integer-constrained variables, so clearly $S \subseteq Q$ and Q is rational MICP representable with one fewer integer dimension via Lemma 22.

Corollary 2 combined with Lemma 28 imply that no new points are created in the relaxation, i.e., $Q = S$. We now have a formulation for S with one fewer dimension and can repeat our argument until C is bounded (it is trivially bounded when $d = 0$). $\square$

This theorem implies that many sets are *not* rational MICP representable. This includes the annulus (Figure 4-3), the set of rank 1 contained in some compact domain (a more general result was proven in [65]), and the set $\{1/n : n = 1, 2, \ldots\} \cup \{0\}$. Furthermore, it follows that if the graph of a function over a compact domain is compact and rational MICP representable, then the function is piecewise linear with finitely many pieces, because the graph of a function is a convex set if and only if it is affine in the interior of the set, and closedness of the graph (which is assumed) implies that the function is continuous. In the following section we consider as well epigraphs of functions.

4.3.3 Representability of epigraphs on a compact domain

Theorem 2. *Let R be a compact set and $f : R \to \mathbb{R}$ such that the epigraph $S = \{(x', \boldsymbol{x}) \in \mathbb{R} \times R : x' \geq f(\boldsymbol{x})\}$ is closed and rational MICP representable where there exits an upper bound u on f such that whenever a $\boldsymbol{z}$-projected set contains a point $(x', \boldsymbol{x})$ it also contains $(u, \boldsymbol{x})$. Then S is a finite union of closed convex sets.*

Proof. Let $M \subseteq \mathbb{R}^{n+p+d}$ be a convex set that induces a rational MICP formulation of

S. Let $C = \text{proj}_z(M)$ be the convex set which is bounded or rationally unbounded by assumption. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{n+p} \times \{\boldsymbol{z}\}))$. So that, by Lemma 28, $S = \bigcup_{\boldsymbol{z} \in (C \cap \mathbb{Z}^d)} \text{cl}(A_{\boldsymbol{z}})$. If C is bounded then S is precisely a finite union of closed convex sets. Suppose then that C is unbounded. Since C is rationally unbounded, let $\boldsymbol{r} \in \mathbb{Z}^d$ such that the ray $\boldsymbol{z} + \lambda \boldsymbol{r} \in C \, \forall \boldsymbol{z} \in C, \lambda \geq 0$. By the same argument as in Theorem 1, we may assume $\boldsymbol{r} = \boldsymbol{e}(1)$ without loss of generality, so the ray $\ell_{\boldsymbol{z}} := \{\boldsymbol{z} + \lambda \boldsymbol{e}(1) : \lambda \geq 0\}$ is contained in C for all $\boldsymbol{z} \in C$.

Consider the set Q defined by

$$(x', \boldsymbol{x}) \in Q \text{ iff } \exists \, \boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1} \text{ such that } (x', \boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M. \tag{4.18}$$

Note that formulation (4.18) is obtained solely by relaxing the integrality constraint on the first component of the d integer-constrained variables, so clearly $S \subseteq Q$ and Q is rational MICP representable with one fewer integer dimension via Lemma 22.

Let $A'_{\boldsymbol{z}}$ be the projection of $A_{\boldsymbol{z}}$ onto the last $n-1$ variables, i.e., onto the domain R of f. Let $(x', \boldsymbol{x}) \in Q$ with corresponding $\boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1}$ such that $(x', \boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M$. As in Lemma 29, we can argue because R is compact that along a ray we have the containment $\text{cl}(A'_{(z_0, \bar{\boldsymbol{z}})}) \subseteq \text{cl}(A'_{(z_0 + \lambda, \bar{\boldsymbol{z}})})$ for any $\lambda \geq 0$. So in particular there exists $x'' \in \mathbb{R}$ such that $(x'', \boldsymbol{x}) \in S$, i.e., $\boldsymbol{x} \in R$, because $\text{cl}(A_{(\lceil z_0 \rceil, \bar{\boldsymbol{z}})}) \in S$. In terms of having points in Q that are not in S, we only need to worry about the case $x' < x''$ because since S is an epigraph, $x' > x''$ implies $(x', \boldsymbol{x}) \in S$. If $x' < x''$ then the function $h_{\boldsymbol{x}}(\boldsymbol{z}) = \inf\{\beta' : (\beta', \boldsymbol{\beta}) \in A_{\boldsymbol{z}}, \boldsymbol{\beta} = \boldsymbol{x}\}$ increases at some point along the ray $\ell_{(z_0, \bar{\boldsymbol{z}})}$. By an extension of Lemma 26 we see that for any $\boldsymbol{x}$ fixed, $h_{\boldsymbol{x}}(\boldsymbol{z})$ is convex in $\boldsymbol{z} \in C$. Convexity combined with increasing at some point in $\ell_{(z_0, \bar{\boldsymbol{z}})}$ implies that $\lim_{\lambda \to \infty} h_{\boldsymbol{x}}(z_0 + \lambda, \bar{\boldsymbol{z}}) = \infty$ by Lemma 27. However, this contradicts our assumption on the $\boldsymbol{z}$-projected set of the formulation which implies $h_{\boldsymbol{x}}(z_0 + \lambda, \bar{\boldsymbol{z}}) \leq u \, \forall \lambda \geq 0$ with $z_0 + \lambda \in \mathbb{Z}$. From this discussion we conclude that no new points are created in the relaxation, i.e., $Q = S$. We now have a formulation for S with one fewer dimension and can repeat our argument until C is bounded (it is trivially bounded when $d = 0$). $\square$

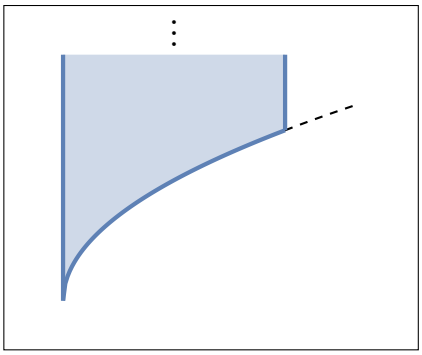


Figure 4-4: The set $\{(x, y) : y \geq \sqrt{x}, 0 \leq x \leq 3\}$ is not rational MICP representable (with an additional technical condition) because it is not a finite union of convex sets; see Theorem 2.

This result can be applied to show that epigraphs of nonconvex functions over a compact domain may not be MICP representable, see, for example, Figure 4-4. In fact, it follows that a closed epigraph for a function defined over a compact domain is rational MICP representable if and only if the function is piecewise convex with finitely many pieces.

4.3.4 Representability of subsets of the natural numbers

In [65] we looked at representability of the natural numbers. We proved an equivalence between rational MICP and rational MILP union (finite set) using a broader definition of rational MICP (allowing exclusion of finitely many points). Here we revisit this question using the new, simplified definition. A key new observation is the closure of rational MICP under finite unions (Lemma 19), a property we discovered after [65]. We also develop new machinery so the proof of the main result is simplified (c.f. Lemma 5 from [65]).

Definition 11. *We say that an infinite subset of the natural numbers $S \subseteq \mathbb{N}$ is periodic if $\exists t \in \mathbb{N}, t > 0$ such that $x \in S$ then $x + rt \in S \, \forall r \in \mathbb{N}$. We say t is a period of S (periods are not unique).*

Note that finite unions of infinite periodic sets are periodic.

Lemma 30. *An infinite subset S of the natural numbers is periodic iff it is rational MILP representable.*

Proof. ($\Rightarrow$): Suppose S is periodic with period t. We will show that there is a finite of rational numbers S_0 such that $S = S_0 + \text{intcone}(t)$, because the right-hand side is rational MICP representable. For every $i \in [\![t-1]\!]$ either there exists a unique minimal $r_i \in \mathbb{N}$ such that $i + r_i t \in S$ or $i + rt \notin S \, \forall r \in \mathbb{N}$. Supposing r_i exists, then $i + (r_i + r')t \in S \, \forall r' \in \mathbb{N}$ because S is periodic, and all integers in S with remainder i modulo t are generated in this manner. Define S_0 be the collection of all such r_i for $i \in [\![t-1]\!]$.

($\Leftarrow$): Suppose S is rational MICP representable. Then $S = S_0 + \text{intcone}(r)$ for some finite set $S_0 \subset \mathbb{N}$ and $r \in \mathbb{N}$. Then by definition S is periodic with period r. $\square$

Lemma 31. *Suppose $S \subseteq \mathbb{N}$ is rational MICP representable with MICP dimension d. Then either S is a finite set or there exists $k \in \mathbb{N}$ such that $S = S_0 \cup \bigcup_{i \in [\![k]\!]} S_i$ where S_0 an infinite periodic subset of $\mathbb{N}$, and for each $i \in [\![k]\!]$, S_i is rational MICP representable with MICP-dimension at most $d-1$.*

Proof. Let $M \subseteq \mathbb{R}^{1+p+d}$ be a convex set that induces an MICP formulation of S. If S is bounded then we're done. Suppose S is unbounded. Let $C := \text{proj}_z(M)$ be the convex set which is rationally unbounded by assumption. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{1+p} \times \{\boldsymbol{z}\}))$ which for any $\boldsymbol{z} \in C \cap \mathbb{Z}^d$ by assumption is equal to some element of $\mathbb{N}$. Since C is rationally unbounded, let $\boldsymbol{r} \in \mathbb{Z}^d$ such that the ray $\boldsymbol{z} + \lambda \boldsymbol{r} \in C \, \forall \boldsymbol{z} \in C, \lambda \geq 0$. By the same argument as in Theorem 1, we may assume $\boldsymbol{r} = \boldsymbol{e}(1)$ without loss of generality, so the ray $\ell_{\boldsymbol{z}} := \{\boldsymbol{z} + \lambda \boldsymbol{e}(1) : \lambda \geq 0\}$ is contained in C for all $\boldsymbol{z} \in C$.

Let $\{T_i\}_{i \in [\![2^d]\!]}$ be such that $C \cap \mathbb{Z}^d = \bigcup_{i \in [\![2^d]\!]} T_i$ and $z_j \equiv z'_j \mod 2$ for all $j \in [\![d]\!]$, $i \in [\![2^d]\!]$ and $\boldsymbol{z}, \boldsymbol{z}' \in T_i$. Let $S_i = \bigcup_{\boldsymbol{z} \in T_i} A_{\boldsymbol{z}}$ be the subset of $\mathbb{N}$ generated by T_i. We claim that **$\mathbf{S_i}$ is either an infinite periodic subset of $\mathbb{N}$ or is rational MICP representable with MICP dimension at most $d-1$.** This would prove the desired result because there are finitely many S_i sets and finite unions of periodic sets are periodic, so we can take S_0 to be the union of the S_i sets that are periodic. Then we may renumber the indices i to match the statement in the lemma.

Define $f(\boldsymbol{z}) := \inf\{x : x \in A_{\boldsymbol{z}}\}$ and $g(\boldsymbol{z}) := \sup\{x : x \in A_{\boldsymbol{z}}\}$ Define $h : C \to \mathbb{R}$ as $h(\boldsymbol{z}) := f(\boldsymbol{z}) - g(\boldsymbol{z})$. The function h is concave, nonnegative, and takes values 0 for

$\boldsymbol{z} \in C \cap \mathbb{Z}^d$. For fixed $i \in [\![2^d]\!]$ we have $\frac{\boldsymbol{z}+\boldsymbol{z}'}{2} \in C \cap \mathbb{Z}^d$ and $\ell_{\frac{\boldsymbol{z}+\boldsymbol{z}'}{2}} \subset C$ for any $\boldsymbol{z}, \boldsymbol{z}' \in T_i$. Then $L := \text{conv}\left(\left\{\ell_{\boldsymbol{z}}, \ell_{\frac{\boldsymbol{z}+\boldsymbol{z}'}{2}}, \ell_{\boldsymbol{z}'}\right\}\right)$ is a subset of C which importantly contains integer points in its relative interior. Let $\tilde{\boldsymbol{z}}$ be one such integer point. Then $h(\tilde{\boldsymbol{z}}) = 0$ which implies h must be entirely zero over all L by Lemma 23. This implies $f(\boldsymbol{z}) = g(\boldsymbol{z})$ is both convex and concave on L and so is affine. Then, since the choice of $\boldsymbol{z}, \boldsymbol{z}' \in T_i$ was arbitrary there exist $\boldsymbol{\alpha}^i \in \mathbb{R}^d, \beta_i \in \mathbb{R}$ such that $f(\tilde{\boldsymbol{z}}) = \tilde{\boldsymbol{z}}^T \boldsymbol{\alpha}^i + \beta_i \, \forall \tilde{\boldsymbol{z}} \in \ell_{\boldsymbol{z}}$ for any $\boldsymbol{z} \in T_i$. Let $s_i = \boldsymbol{e}(1)^T \boldsymbol{\alpha}^i$. We know that f takes nonnegative integer values at integer points along a ray in direction $\boldsymbol{e}(1)$ so s_i in particular must be a nonnegative integer.

For the cases where $\mathbf{s_i} \neq \mathbf{0}$, we claim that the set of numbers S_i is periodic with period s_i. For any $x \in S_i$, $\exists \boldsymbol{z} \in C \cap \mathbb{Z}^d$ such that $A_{\boldsymbol{z}} = \{x\} = \{f(\boldsymbol{z})\}$. Then $\boldsymbol{z} + \boldsymbol{e}(1) \in C \cap \mathbb{Z}^d$ and $A_{\boldsymbol{z}} = \{f(\boldsymbol{z} + \boldsymbol{e}(1))\} = \{x + s_i\}$ so $x + s_i \in S_i$.

For the cases where $\mathbf{s_i} = \mathbf{0}$, we will show that S_i is rational MICP representable with MICP dimension at most $d - 1$. Note that S_i is rational MICP representable because it can be obtained by performing an invertible affine transformation on C. That is, let $\tilde{\boldsymbol{z}} \in T_i$, then

$$\boldsymbol{x} \in S_i \text{ iff } \exists \boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{z}} + 2\boldsymbol{z}) \in M. \tag{4.19}$$

Let M_i be the convex set that induces the rational MICP representation of S_i via Lemma 21. Consider the set Q_i defined by

$$\boldsymbol{x} \in Q_i \text{ iff } \exists \, \boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1} \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M_i. \tag{4.20}$$

Note that formulation (4.20) is obtained solely by relaxing the integrality constraint on the first component of the d integer-constrained variables, so clearly $S_i \subseteq Q_i$ and Q_i is rational MICP representable with MICP dimension at most $d-1$ via Lemma 22. We now claim that $S_i = Q_i$. Let $x \in Q_i$ with corresponding $\boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1}$ such that $(x, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M_i$. Let $C_i = \text{proj}_z(M_i)$ be the projection of M_i onto the last d variables. Note that $\boldsymbol{e}(1)$ remains a recession direction of C_i. Therefore $(z_0 + \lambda, \bar{\boldsymbol{z}}) \in C_i \, \forall \lambda \geq 0$, and in particular $\exists z_0' \in \mathbb{Z}$ such that $z_0' > z_0$ and $(z_0', \bar{\boldsymbol{z}}) \in C_i$.

We know that $f(\boldsymbol{z}) = \boldsymbol{z}^T\boldsymbol{\alpha}^i + \beta_i$ on $\ell_{(z_0',\bar{\boldsymbol{z}})}$ and since $s_i = 0$, the ray starting at $(z_0', \bar{\boldsymbol{z}})$ projects to a single natural number, i.e., $A_{\boldsymbol{z}} = \{\sum_{j\in[\![d-1]\!]} \alpha^i_{j+1}\bar{z}_j + \beta_i\}\, \forall \boldsymbol{z} \in \ell_{(z_0',\bar{\boldsymbol{z}})}$. Since the $\boldsymbol{z}$-projected sets are eventually bounded on the ray, we may apply Lemma 29 to conclude that $x \in S_i$. □

Theorem 3. *Let $S \subseteq \mathbb{N}$ with $|S| = \infty$. Then the following are equivalent:*

(a) S is rational MICP representable.

(b) There exists a finite set S_0 and an infinite periodic set S_1 such that $S = S_0 \cup S_1$.

(c) There exists a finite set S_0 and a rational-MILP-representable set S_1 such that $S = S_0 \cup S_1$.

Proof. (a) ⇒ (b): Repeatedly apply Lemma 31.

(b) ⇒ (c): Use Lemma 33 to obtain a rational MILP formulation of S_1.

(c) ⇒ (a): Corollary 1. □

With this result we can gain insight into rational MICP. For Theorems 1 and 2, we do not know if the rationality assumption is essential for the result or if simply provides helpful structure for the proof. This is not the case for the natural numbers.

For example, general (non-rational) MILP and also MICP can represent subsets of $\mathbb{N}$ which are not periodic. This can even be achieved when the MICP formulation is described with rational data. For instance we could require M to have a representation of the form $\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \in K$ where $\boldsymbol{A}$ and $\boldsymbol{b}$ are an appropriately sized rational matrix and rational vector, and K is a specially structured convex cone (e.g. the semidefinite cone or a product of second-order cones defined as $\mathcal{L}_n := \{(t, \boldsymbol{x}) \in \mathbb{R}^n : ||\boldsymbol{x}||_2 \leq t\}$) or to have polynomial constraints with rational coefficients. Unfortunately these restrictions can still result in representable sets which are not periodic and hence not rational MILP representable. We present such an example below.

Example 3. *For $x \in \mathbb{R}$ let $f(x) = x - \lfloor x \rfloor$. For $\varepsilon > 0$ consider the set*

$$K_\varepsilon = \{\boldsymbol{x} \in \mathbb{R}^2 : (x_2 + \varepsilon, x_1, x_1) \in \mathcal{L}_3, (2x_1 + 2\varepsilon, x_2, x_2) \in \mathcal{L}_3, x_1, x_2 \geq 0\} \tag{4.21}$$

$$= \{\boldsymbol{x} \in \mathbb{R}^2 : \sqrt{2}x_1 - \varepsilon \leq x_2 \leq \sqrt{2}x_1 + \sqrt{2}\varepsilon, \quad x_1, x_2 \geq 0\} \tag{4.22}$$

and $S_\varepsilon = \{x_1 \in \mathbb{R} : \exists x_2 \text{ s.t. } (x_1, x_2) \in K_\varepsilon \cap \mathbb{Z}^2\} = \{x \in \mathbb{N} : f(\sqrt{2}x) \notin (\varepsilon, 1 - \sqrt{2}\varepsilon)\}$. *Let* $\varepsilon_0 < 1/(1+\sqrt{2})$ *be rational (e.g.* $\varepsilon = 0.4$*). Suppose that for some* $a, b \in \mathbb{N}$,$a \geq 1$ *it holds* $ak + b \in S_{\varepsilon_0}$ *for all* $k \in \mathbb{N}$. $\emptyset \neq (\varepsilon_0, 1 - \sqrt{2}\varepsilon_0) \subseteq (0, 1)$*, so by Kronecker's Approximation Theorem we have that there exist* $k_0 \in \mathbb{N}$ *such that* $f(\sqrt{2}(ak_0 + b)) \in (\varepsilon_0, 1 - \sqrt{2}\varepsilon_0)$ *which is a contradiction. Therefore the set* S_{ε_0} *is not periodic and in particular it is not rational MILP representable.*

Corollary 3. *The intersection of two rational MICP representable sets is not in general rational MICP representable.*

Proof. The set S_ε of Example 3 is the intersection of two rational-MICP representable sets: $\{x_1 \in \mathbb{Z} : \exists x_2 \in \mathbb{Z} \text{ s.t.} \sqrt{2}x_1 - \varepsilon \leq x_2\}$ and $\{x_1 \in \mathbb{Z} : \exists x_2 \in \mathbb{Z} \text{ s.t.} x_2 \leq \sqrt{2}x_1 + \sqrt{2}\varepsilon\}$. The statement now follows from Theorem 3. □

4.3.5 Representability of piecewise linear functions

Definition 12. *The continuous function* $\mathcal{P} : \mathbb{R}_+ \to \mathbb{R}$ *is a* $\mathcal{PWL}$*-function if it is piecewise linear on with breakpoints only at integer values. If, furthermore,* $\mathcal{P}(i) \in \mathbb{Q}\, \forall i \in \mathbb{N}$ *then we say* $\mathcal{P}(i)$ *is a rational* $\mathcal{PWL}$*-function.*

$\mathcal{PWL}$-functions are uniquely defined by their values at the integers $\{\mathcal{P}(i)\}_{i\in\mathbb{N}}$. We parameterize unit-step segments by their starting point and slope as $P_{(i,x,c)} := \text{conv}(\{(i, x), (i+1, x+c)\})$. The graph of $\mathcal{P}$ is $\bigcup_{i\in\mathbb{N}} P_{(i,\mathcal{P}(i),\mathcal{P}(i+1)-\mathcal{P}(i))}$.

Definition 13. *We say that a* $\mathcal{PWL}$*-function* $\mathcal{P}$ *is periodic if* $\exists t \in \mathbb{N}, t > 0$ *such that* $\forall i \in [\![t-1]\!]$ *and* $\forall r \in \mathbb{N}$*, and* $\mathcal{P}(rt + i + 1) - \mathcal{P}(rt + i) = \mathcal{P}(i + 1) - \mathcal{P}(i)$*. We say* t *is a period of* S *(periods are not unique).*

Definition 14. *We say that an infinite subset* S *of segments is periodic if* $\exists t \in \mathbb{N}, t > 0$ *such that if for some* i, x, c *we have* $P_{(i,x,c)} \subset S$ *then* $\forall r \in \mathbb{N}$ $P_{(rt+i,x',c)} \subset S$ *for some* x'*. We say* t *is a period of* S *(periods are not unique).*

A finite union of non-overlapping infinite periodic subsets of segments is also periodic: take the product of the periods.

Lemma 32. *If a periodic infinite subset S of segments is the graph of some continuous $\mathcal{PWL}$-function $\mathcal{P}$ then $\mathcal{P}$ is periodic.*

Lemma 33. *Let S be the graph of a rational $\mathcal{PWL}$-function $\mathcal{P}$. Then $\mathcal{P}$ is periodic iff S is rational MILP representable. Furthermore, we can write an explicit rational MILP formulation of S where each $\boldsymbol{z}$-projected set is a full segment.*

Proof. ($\Rightarrow$): We will show that the segments in S are finitely generated. Let t be a period of $\mathcal{P}$. Let $T_0 = \bigcup_{i \in [\![t-1]\!]} P_{(i,\mathcal{P}(i),\mathcal{P}(i+1)-\mathcal{P}(i))}$ be the graph of $\mathcal{P}$ over the interval $[0,t]$. T_0 is a finite union of bounded rational polyhedra, because the endpoints of the segments are rational numbers. Let $\boldsymbol{r} = (t, \mathcal{P}(t) - \mathcal{P}(0))$. Then we claim $S = T_0 + \text{intcone}(\boldsymbol{r})$, the right-hand side being rational MILP representable where each $A_{\boldsymbol{z}}$ is a full segment. To prove the $\supseteq$ direction consider $\tilde{P} := P_{(i,\mathcal{P}(i),s)} + \lambda \boldsymbol{r}$ for some $i \in [\![t-1]\!]$ and $\lambda \in \mathbb{N}$ where $s = \mathcal{P}(i+1) - \mathcal{P}(i)$. $\tilde{P}$ is the segment $P_{(i+\lambda t, \mathcal{P}(i)+\lambda(\mathcal{P}(t)-\mathcal{P}(0)),s)}$. Since $\mathcal{P}$ is periodic, the slope of $\tilde{P}$ matches the slope of $\mathcal{P}$ between $i+\lambda t$ and $i+\lambda t+1$ by definition, so it remains to show that $\mathcal{P}(i) + \lambda(\mathcal{P}(t) - \mathcal{P}(0)) = \mathcal{P}(i + \lambda t)$. We can write

$$\mathcal{P}(i+\lambda t) = \sum_{k=0}^{i+\lambda t} (\mathcal{P}(k+1) - \mathcal{P}(k)) + \mathcal{P}(0) \tag{4.23}$$

$$= \sum_{k=0}^{\lambda t - 1} (\mathcal{P}(k+1) - \mathcal{P}(k)) + \sum_{k=\lambda t}^{i+\lambda t} (\mathcal{P}(k+1) - \mathcal{P}(k)) + \mathcal{P}(0) \tag{4.24}$$

$$= \lambda(\mathcal{P}(t) - \mathcal{P}(0)) + \sum_{k=0}^{i} (\mathcal{P}(k+1) - \mathcal{P}(k)) + \mathcal{P}(0) \tag{4.25}$$

$$= \lambda(\mathcal{P}(t) - \mathcal{P}(0)) + \mathcal{P}(i). \tag{4.26}$$

The $\subseteq$ direction follows by reversing the same argument.

($\Leftarrow$): Suppose S is rational MILP representable. Then there exist $\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t \subseteq \mathbb{Z}^n$ and rational polytopes $S_1, S_2, \ldots, S_k$ such that

$$S = \bigcup_{i \in [\![k]\!]} S_i + \text{intcone}(\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t). \tag{4.27}$$

In particular, for any subset $T \subset S$, we have $T + \text{intcone}(\boldsymbol{r}^1, \boldsymbol{r}^2, \ldots, \boldsymbol{r}^t) \subset S$. Since S is the graph of some piecewise linear function $\mathcal{P}$, one of the integer rays must have a positive first component, say $r_1^1 > 0$. Then r_1^1 must be a period of $\mathcal{P}$ because any segment in S translated by $\boldsymbol{r}^1$ retains the same slope and belongs to the graph of $\mathcal{P}$. □

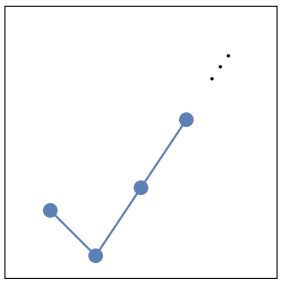


Figure 4-5: The graph of the piecewise linear function depicted above taking values $1, 0, 1.5, 3, 4.5, ...$ at $i = 0, 1, 2, 3, 4, ...$ respectively is rational MICP representable but not rational MILP representable. The graph is representable if the first segment on the left is excluded.

Lemma 33 is sufficient to separate rational MILP from rational MICP for the graphs of rational $\mathcal{PWL}$-functions. The graph depicted in Figure 4-5 is not MILP representable because the corresponding function is not periodic. It is rational MICP representable because it is the union of the segment on the left with the remaining segments, which are rational MILP representable. We now investigate this question further and see that, essentially, finite unions are the only representability power gained.

We will consider, perhaps with some loss of generality, MICP formulations for the graph of $\mathcal{P}$ where each $\boldsymbol{z}$-projected set is a complete unit-step line segment.

Lemma 34. *Suppose S is a subset of segments of the graph of a rational $\mathcal{PWL}$-function $\mathcal{P}$ which is rational MICP or MICP representable with MICP dimension d where each $\boldsymbol{z}$-projected set is a full unit-step segment. Then either S is bounded or there exists a $k \in \mathbb{N}$ such that $S = S_0 \cup \bigcup_{i \in [\![k]\!]} S_k$ where S_0 a periodic infinite subset of segments and for each i, S_i is a rational MICP representable subset of segments included in S with MICP dimension at most $d - 1$.*

Proof. Let $M \subseteq \mathbb{R}^{2+p+d}$ be a convex set that induces an MICP formulation of S. If S is bounded then we're done. Suppose S is unbounded. Let $C := \text{proj}_z(M)$ be the convex set which is rationally unbounded by assumption. Let $A_{\boldsymbol{z}} = \text{proj}_x(M \cap (\mathbb{R}^{1+p} \times \{\boldsymbol{z}\}))$ which for any $\boldsymbol{z} \in C \cap \mathbb{Z}^d$ by assumption is equal to the segment P_i for some i. Since C is rationally unbounded, let $\boldsymbol{r} \in \mathbb{Z}^d$ such that the ray $\boldsymbol{z} + \lambda \boldsymbol{r} \in C \, \forall \boldsymbol{z} \in C, \lambda \geq 0$. By the same argument as in Theorem 1, we may assume $\boldsymbol{r} = \boldsymbol{e}(1)$ without loss of generality, so the ray $\ell_{\boldsymbol{z}} := \{\boldsymbol{z} + \lambda \boldsymbol{e}(1) : \lambda \geq 0\}$ is contained in C for all $\boldsymbol{z} \in C$.

Let $\{T_i\}_{i \in [\![2^d]\!]}$ be such that $C \cap \mathbb{Z}^d = \bigcup_{i \in [\![2^d]\!]} T_i$ and $z_j \equiv z'_j \mod 2$ for all $j \in [\![d]\!]$, $i \in [\![2^d]\!]$ and $\boldsymbol{z}, \boldsymbol{z}' \in T_i$. Let $S_i = \bigcup_{\boldsymbol{z} \in T_i} A_{\boldsymbol{z}}$ be the subset of segments generated by T_i.

We claim that **all segments in S_i have the same slope**. For fixed $i \in [\![2^d]\!]$ we have $\frac{\boldsymbol{z}+\boldsymbol{z}'}{2} \in C \cap \mathbb{Z}^d$ and $\ell_{\frac{\boldsymbol{z}+\boldsymbol{z}'}{2}} \subset C$ for any $\boldsymbol{z}, \boldsymbol{z}' \in T_i$. Then $L := \text{conv}\left(\left\{\ell_{\boldsymbol{z}}, \ell_{\frac{\boldsymbol{z}+\boldsymbol{z}'}{2}}, \ell_{\boldsymbol{z}'}\right\}\right)$ is a subset of C which importantly contains integer points in its relative interior. Let $\tilde{\boldsymbol{z}}$ be one such integer point. Then $A_{\tilde{\boldsymbol{z}}} = P_j$ for some j with slope $c = \mathcal{P}(j+1) - \mathcal{P}(j)$. Define $h_c(\boldsymbol{z}) = \sup\{x_1 + (-1/c)x_2 : \boldsymbol{x} \in A_{\boldsymbol{z}}\} - \inf\{x_1 + (-1/c)x_2 : \boldsymbol{x} \in A_{\boldsymbol{z}}\}$. We can see that h_c is concave (as a sum of two functions which are concave by Lemmas 25 and 26), nonnegative, and takes value zero at $\tilde{\boldsymbol{z}}$ and so must be entirely zero over all L by Lemma 23. But $h_c(\boldsymbol{z}) = 0$ implies that $A_{\boldsymbol{z}}$ falls in an affine subspace of the form $\{\boldsymbol{x} \in \mathbb{R}^2 : x_2 = cx_1 + \gamma_{\boldsymbol{z}}\}$ for some $\gamma_{\boldsymbol{z}} \in \mathbb{R}$ [48, p. 209], so all segments in S_i must have the same slope since the choice of $\boldsymbol{z}, \boldsymbol{z}'$ was arbitrary.

We will now characterize the starting points for the segments contained in S_i. Define $f_1(\boldsymbol{z}) = \inf\{x_1 : \boldsymbol{x} \in A_{\boldsymbol{z}}\}$ and $g_1(\boldsymbol{z}) = \sup\{x_1 : \boldsymbol{x} \in A_{\boldsymbol{z}}\}$. By Lemmas 26 and 25, f is convex and g is concave. Define $h' : C \to \mathbb{R}$ as $h'(\boldsymbol{z}) = g_1(\boldsymbol{z}) - f_1(\boldsymbol{z})$. Analogously to h_c, h' is concave and nonnegative. It also satisfies $h'(\boldsymbol{z}) = 1 \, \forall \boldsymbol{z} \in C \cap \mathbb{Z}^d$ because by assumption $A_{\boldsymbol{z}}$ is a segment that spans a unit step in the first coordinate. Consider L as before for some choice of $\boldsymbol{z}, \boldsymbol{z}' \in T_i$. Then, by Lemma 23, $h'(\tilde{\boldsymbol{z}}) = 1$ for all $\tilde{\boldsymbol{z}} \in L$ because L has an integer point in its relative interior. It follows that $f_1(\tilde{\boldsymbol{z}}) = g_1(\tilde{\boldsymbol{z}}) - 1 \, \forall \tilde{\boldsymbol{z}} \in L$, so in particular f is both concave and convex and is therefore affine. Then, since the choice of $\boldsymbol{z}, \boldsymbol{z}' \in T_i$ was arbitrary there exist $\boldsymbol{\alpha}^i \in \mathbb{R}^d, \beta_i \in \mathbb{R}$ such that $f_1(\tilde{\boldsymbol{z}}) = \tilde{\boldsymbol{z}}^T \boldsymbol{\alpha}^i + \beta_i \, \forall \tilde{\boldsymbol{z}} \in \ell_{\boldsymbol{z}}$ for any $\boldsymbol{z} \in T_i$. Let

$s_i = \boldsymbol{e}(1)^T \boldsymbol{\alpha}^i$. We know that f_1 takes nonnegative integer values at integer points along a ray in direction $\boldsymbol{e}(1)$ so s_i in particular must be a nonnegative integer.

For the cases where $\mathbf{s_i} \neq \mathbf{0}$, we claim that the set of segments S_i is periodic. Note that the set of segments is unbounded because $f(\tilde{\boldsymbol{z}})$ generates an infinite arithmetic progression along any ray $\ell_{\boldsymbol{z}}$ for $\boldsymbol{z} \in T_i$. The latter combined with the observation that all segments in S_i have the same slope implies that if $P_{(j,\mathcal{P}(j),c)}$ is a segment in S_i then $P_{(j+rs_i,\mathcal{P}(j+rs_i),c)} \subset S_i \, \forall r \in \mathbb{N}$. Now, take the union of all S_i for i with $s_i \neq 0$ to obtain S_0 in the statement of the lemma. There are at most 2^d such i so this is a finite union.

For the cases where $\mathbf{s_i} = \mathbf{0}$, we claim, to complete the proof, that the set of segments S_i is rational MICP representable with MICP dimension at most $d-1$. Then we may renumber the indices i to match the statement in the lemma. Note that S_i is rational MICP representable because it can be obtained by performing an invertible affine transformation on C. That is, let $\tilde{\boldsymbol{z}} \in T_i$, then

$$\boldsymbol{x} \in S_i \text{ iff } \exists \boldsymbol{y} \in \mathbb{R}^p, \boldsymbol{z} \in \mathbb{Z}^d \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{z}} + 2\boldsymbol{z}) \in M. \tag{4.28}$$

Let M_i be the convex set that induces the rational MICP representation of S_i via Lemma 21. Consider the set Q_i defined by

$$\boldsymbol{x} \in Q_i \text{ iff } \exists \, \boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1} \text{ such that } (\boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M_i. \tag{4.29}$$

Note that formulation (4.29) is obtained solely by relaxing the integrality constraint on the first component of the d integer-constrained variables, so clearly $S_i \subseteq Q_i$ and Q_i is rational MICP representable with MICP dimension at most $d-1$ via Lemma 22. We now claim that $S_i = Q_i$.

Let $\boldsymbol{x} \in Q_i$ with corresponding $\boldsymbol{y} \in \mathbb{R}^p, z_0 \in \mathbb{R}, \bar{\boldsymbol{z}} \in \mathbb{Z}^{d-1}$ such that $(\boldsymbol{x}, \boldsymbol{y}, z_0, \bar{\boldsymbol{z}}) \in M_i$. Let $C_i = \text{proj}_z(M_i)$ be the projection of M_i onto the last d variables. Note that $\boldsymbol{e}(1)$ remains a recession direction of C_i, and the ray $\ell_{(z_0,\bar{\boldsymbol{z}})}$ contains the ray $\ell_{(\lceil z_0 \rceil,\bar{\boldsymbol{z}})}$, and both are contained in C_i. We claim that $\text{cl}(A_{\boldsymbol{z}})$ is a constant segment $\forall \boldsymbol{z} \in \ell_{(z_0,\bar{\boldsymbol{z}})} \cap \mathbb{Z}^d$. Since $s_i = 0$, we have that $f_1(\boldsymbol{z})$ (properly redefined on C_i) is

constant $\forall \boldsymbol{z} \in \ell_{(\lceil z_0 \rceil, \bar{\boldsymbol{z}})}$. The value $f_1(\boldsymbol{z})$ is the first component of the starting point of the segment $\text{cl}(A_{\boldsymbol{z}}) \subseteq S_i$ when $\boldsymbol{z} \in \mathbb{Z}^d \cap C_i$, and since segments in S_i form part of the graph of a $\mathcal{PWL}$-function, there can be at most one segment in the range $[f_1(\boldsymbol{z}), f_1(\boldsymbol{z})]$, hence $\text{cl}(A_{\boldsymbol{z}})$ must be a unique constant segment $\forall \boldsymbol{z} \in \ell_{(z_0, \bar{\boldsymbol{z}})} \cap \mathbb{Z}^d$. Now we may apply Lemma 29 to conclude that $\boldsymbol{x} \in S_i$.

□

Theorem 4. *Let $\mathcal{P}$ be a rational $\mathcal{PWL}$-function. Let S be the graph of $\mathcal{P}$. Then the following are equivalent:*

(a) S is rational MICP representable where each $\boldsymbol{z}$-projected set is a full unit-step segment.

(b) There exists a finite subset of segments S_0 and an infinite periodic subset of segments S_1 such that $S = S_0 \cup S_1$.

(c) There exists a finite subset of segments T_0 and a rational-MILP representable subset of segments T_1 such that $S = T_0 \cup T_1$, where each $\boldsymbol{z}$-projected set in the MILP formulation is a full unit-step segment.

Proof. (a) ⇒ (b): Repeatedly apply Lemma 34.

(b) ⇒ (c): It is possible that the infinite periodic subset of segments S_1 contains gaps, i.e., does not define the graph of a piecewise linear function. However, since S_0 is finite we can define T_1 to be the infinite set of segments that begins after all segments in S_0. Define $T_0 = S \setminus T_1$. T_1 remains periodic and defines the graph of a periodic piecewise linear function, so we can apply Lemma 33 to obtain a rational MILP formulation of T_1 (it does not matter that the function does not start at 0).

(c) ⇒ (a): Corollary 1. The $\boldsymbol{z}$-projected sets are preserved from the MILP to the MICP formulation. □

4.4 How many integer variables are needed: a necessary condition

In [65], we developed the *midpoint lemma* as a necessary condition that we used to show that a number of classes of sets are not MICP representable, regardless of any rationality condition. Here we develop a variant of the midpoint lemma that is a necessary condition for MICP representability using a fixed number of integer variables.

Definition 15. *We say that a set $S \subseteq \mathbb{R}^n$ is w-**strongly nonconvex**, if there exists a subset $R \subseteq S$ with $|R| = w$ such that for all pairs $\boldsymbol{x}, \boldsymbol{y} \in R$,*

$$\frac{\boldsymbol{x} + \boldsymbol{y}}{2} \notin S, \tag{4.30}$$

that is, an subset of points in S of cardinality w such that the midpoint between any pair is not in S.

Lemma 35. ***(The finite-dimensional midpoint lemma)*** *Let $S \subseteq \mathbb{R}^n$. If S is w-strongly nonconvex, then S cannot be MICP representable with MICP-dimension less than $\lceil \log_2(w) \rceil$.*

Proof. Suppose we have R as in the statement above and there exists an MICP formulation of S, that is, a closed convex set $M \subset \mathbb{R}^{n+p+d}$ such that $\boldsymbol{x} \in S$ iff $\exists \boldsymbol{z} \in \mathbb{Z}^d, y \in \mathbb{R}^p$ such that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \in M$. Then for each point $\boldsymbol{x} \in R$ we associate at least one integer point $\boldsymbol{z}^{\boldsymbol{x}} \in \mathbb{Z}^d$ and a $\boldsymbol{y}^{\boldsymbol{x}} \in \mathbb{R}^p$ such that $(\boldsymbol{x}, \boldsymbol{y}^{\boldsymbol{x}}, \boldsymbol{z}^{\boldsymbol{x}}) \in M$. If there are multiple such pairs of points $\boldsymbol{z}^{\boldsymbol{x}}, \boldsymbol{y}^{\boldsymbol{x}}$ then for the purposes of the argument we may choose one arbitrarily.

Suppose $d < \lceil \log_2(w) \rceil$. We will derive a contradiction by proving that there exist two points $\boldsymbol{x}, \boldsymbol{x}' \in R$ such that the associated integer points $\boldsymbol{z}^{\boldsymbol{x}}, \boldsymbol{z}^{\boldsymbol{x}'}$ satisfy

$$\frac{\boldsymbol{z}^{\boldsymbol{x}} + \boldsymbol{z}^{\boldsymbol{x}'}}{2} \in \mathbb{Z}^d. \tag{4.31}$$

Indeed, this property combined with convexity of M, i.e., $\left(\frac{\boldsymbol{x}+\boldsymbol{x}'}{2}, \frac{\boldsymbol{y}^{x}+\boldsymbol{y}^{\boldsymbol{x}'}}{2}, \frac{\boldsymbol{z}^{\boldsymbol{x}}+\boldsymbol{z}^{\boldsymbol{x}'}}{2} \right) \in M$

would imply that $\frac{\boldsymbol{x}+\boldsymbol{x}'}{2} \in S$, which contradicts the definition of R.

Recall a basic property of integers that if $i, j \in \mathbb{Z}$ and $i \equiv j \pmod 2$, i.e., i and j are both even or odd, then $\frac{i+j}{2} \in \mathbb{Z}$. We say that two integer vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{Z}^d$ have the same *parity* if α_i and β_i are both even or odd for each component $i = 1, \ldots, d$. Trivially, if $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ have the same parity, then $\frac{\boldsymbol{\alpha}+\boldsymbol{\beta}}{2} \in \mathbb{Z}^d$. Given that we can categorize any integer vector according to the 2^d possible choices for whether its components are even or odd, and we notice that from any collection of integer vectors of size greater than $2^d + 1$ we must have at least one pair that has the same parity. Therefore since $|R| = w \geq 2^d + 1$ we can find a pair $\boldsymbol{x}, \boldsymbol{x}' \in R$ such that their associated integer points $\boldsymbol{z}^{\boldsymbol{x}}, \boldsymbol{z}^{\boldsymbol{x}'}$ have the same parity and thus satisfy (4.31). □

It is interesting to recognize the corner cases $w = 2$ and $w = \infty$. The former implies that S is not convex, and hence we need at least one integer variable. The latter implies that S is not MICP representable, corresponding to the original midpoint lemma.

For finite w, an illustrative exaple is the binary hypercube $S = \{0, 1\}^n$, for which we may take $R = S$ since no midpoints are in S. Note $|R| = 2^n$, so the lemma implies that we need at least n integer variables to represent S. We can indeed write an MICP (MILP) formulation for S using n binary variables . A simple MILP formulation is

$$\boldsymbol{x} \in \{0, 1\}^n \text{ iff } \exists\, \boldsymbol{z} \in \mathbb{Z}^n \text{ s.t. } \boldsymbol{x} = \boldsymbol{z}, \boldsymbol{0} \leq z \leq \boldsymbol{1}. \tag{4.32}$$

Appendix A

Tables

Table A.1: MINLPLIB2 instances. "Conic rep" column indicates which cones are used in the conic representation of the instance (second-order cone and/or exponential). CPLEX is capable of solving only second-order cone instances. Times in seconds.

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
batch	Exp	1	0.26	2	0.60	0.66	0.56	–
batchdes	Exp	1	0.11	1	0.07	0.16	0.02	–
batchs101006m	Exp	3	3.26	10	1.88	5.10	76.96	–
batchs121208m	Exp	3	6.74	4	3.14	13.09	316.14	–
batchs151208m	Exp	3	10.72	6	7.97	16.90	516.04	–
batchs201210m	Exp	2	25.14	8	14.92	29.12	970.51	–
clay0203h	SOC	5	1.42	9	0.90	0.70	1.28	0.35
clay0203m	SOC	6	1.61	10	0.40	0.86	0.34	0.37
clay0204h	SOC	1	1.85	3	3.60	7.14	5.72	1.61
clay0204m	SOC	1	0.55	3	0.33	2.55	3.30	1.02
clay0205h	SOC	3	24.40	4	20.89	78.19	168.28	8.93
clay0205m	SOC	3	8.11	6	5.50	9.63	61.91	1.77
clay0303h	SOC	5	2.41	9	0.97	1.53	1.96	0.54
clay0303m	SOC	7	2.60	10	0.58	1.73	0.76	0.68
clay0304h	SOC	9	13.87	11	5.27	2.50	26.33	1.42
clay0304m	SOC	13	18.97	16	2.84	7.09	7.20	2.13
clay0305h	SOC	3	52.97	4	23.81	1.97	139.27	23.32

Table A.1: MINLPLIB2 instances continued

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
clay0305m	SOC	3	11.83	7	6.16	12.90	52.53	2.51
du-opt	SOC	7	3.19	61	0.76	>36000	0.11	1.54
du-opt5	SOC	4	1.55	22	0.22	0.75	0.11	1.97
enpro48pb	Exp	1	0.51	2	0.22	1.73	0.85	–
enpro56pb	Exp	1	0.60	1	0.22	1.52	4.47	–
ex1223	ExpSOC	1	0.06	3	0.07	0.14	0.03	–
ex1223a	SOC	0	0.02	1	0.03	0.11	0.02	0.01
ex1223b	ExpSOC	1	0.08	3	0.07	0.15	0.02	–
ex4	SOC	2	1.06	2	0.13	1.15	0.25	0.86
fac3	SOC	2	0.19	6	0.15	0.24	0.16	0.07
netmod_dol2	SOC	7	49.97	33	167.49	33.93	293.76	12.58
netmod_kar1	SOC	12	8.05	102	56.45	3.32	122.98	7.68
netmod_kar2	SOC	12	8.14	102	56.35	3.30	122.28	7.66
no7_ar25_1	SOC	3	67.97	2	25.19	82.09	17601.54	54.34
no7_ar2_1	SOC	1	8.87	1	7.06	31.81	14957.66	21.83
no7_ar3_1	SOC	3	91.36	4	71.04	392.98	16495.95	126.09
no7_ar4_1	SOC	4	107.58	5	85.87	274.72	17865.83	48.97
no7_ar5_1	SOC	5	115.25	7	69.23	68.90	17452.47	32.60
nvs03	SOC	1	0.03	1	0.06	0.13	0.18	0.00
slay04h	SOC	2	0.32	5	0.19	0.68	0.53	0.14
slay04m	SOC	2	0.17	5	0.11	0.57	0.32	0.18
slay05h	SOC	3	0.65	9	0.60	3.29	1.57	0.37
slay05m	SOC	3	0.28	7	0.18	0.84	1.02	0.16
slay06h	SOC	2	0.76	12	1.94	5.26	4.65	0.69
slay06m	SOC	2	0.32	9	0.29	1.57	2.94	0.42
slay07h	SOC	3	1.75	15	5.04	18.35	9.96	0.98
slay07m	SOC	3	0.56	12	0.66	2.51	5.75	0.67
slay08h	SOC	3	2.65	22	27.27	180.20	26.47	1.50
slay08m	SOC	2	0.58	21	2.89	3.69	13.17	0.96
slay09h	SOC	3	4.36	36	163.31	92.70	79.79	1.93
slay09m	SOC	3	1.11	28	17.22	11.01	33.36	1.54
slay10h	SOC	4	21.94	80	8155.02	11745.37	442.46	7.55

Table A.1: MINLPLIB2 instances continued

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
slay10m	SOC	4	4.36	77	1410.08	516.81	167.81	1.80
syn05h	Exp	1	0.07	2	0.09	0.31	0.17	–
syn05m	Exp	1	0.04	2	0.07	0.28	0.14	–
syn05m02h	Exp	1	0.15	1	0.06	0.33	0.11	–
syn05m02m	Exp	1	0.08	1	0.07	0.33	0.29	–
syn05m03h	Exp	1	0.23	1	0.07	0.33	0.13	–
syn05m03m	Exp	1	0.12	1	0.07	0.32	0.30	–
syn05m04h	Exp	1	0.29	1	0.07	0.38	0.19	–
syn05m04m	Exp	1	0.17	1	0.08	0.32	0.61	–
syn10h	Exp	0	0.10	1	0.04	0.20	0.09	–
syn10m	Exp	1	0.08	2	0.04	0.25	0.23	–
syn10m02h	Exp	1	0.27	1	0.09	0.46	0.21	–
syn10m02m	Exp	1	0.16	2	0.09	0.42	3.05	–
syn10m03h	Exp	1	0.38	1	0.08	0.59	0.24	–
syn10m03m	Exp	1	0.23	1	0.08	0.54	10.47	–
syn10m04h	Exp	1	0.53	1	0.11	0.52	0.19	–
syn10m04m	Exp	1	0.34	1	0.11	0.72	40.41	–
syn15h	Exp	1	0.22	1	0.06	0.29	0.14	–
syn15m	Exp	1	0.10	2	0.07	0.30	0.32	–
syn15m02h	Exp	1	0.51	1	0.09	0.47	0.18	–
syn15m02m	Exp	1	0.24	1	0.09	0.44	5.51	–
syn15m03h	Exp	1	44.15	1	0.13	0.99	0.23	–
syn15m03m	Exp	1	0.38	2	0.11	0.66	25.67	–
syn15m04h	Exp	1	1.47	1	0.14	1.61	0.32	–
syn15m04m	Exp	1	0.50	2	0.14	1.43	186.20	–
syn20h	Exp	2	0.33	2	0.10	0.34	0.20	–
syn20m	Exp	1	0.13	2	0.06	0.39	1.31	–
syn20m02h	Exp	2	1.07	2	0.15	0.57	0.41	–
syn20m02m	Exp	2	0.44	2	0.10	0.73	381.88	–
syn20m03h	Exp	1	1.21	1	0.13	1.52	0.78	–
syn20m03m	Exp	2	0.64	2	0.15	2.00	993.73	–
syn20m04h	Exp	1	1.81	1	0.19	2.41	1.11	–

Table A.1: MINLPLIB2 instances continued

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
syn20m04m	Exp	2	0.91	2	0.27	9.77	1806.83	–
syn30h	Exp	3	0.73	3	0.12	0.59	0.28	–
syn30m	Exp	3	0.28	3	0.09	0.49	90.26	–
syn30m02h	Exp	3	1.77	3	0.21	12.98	0.44	–
syn30m02m	Exp	3	0.82	4	0.19	1.67	1041.13	–
syn30m03h	Exp	3	2.24	3	0.40	11444.39	1.23	–
syn30m03m	Exp	3	1.28	3	0.27	7.78	1878.32	–
syn30m04h	Exp	3	3.51	3	0.49	>36000	2.74	–
syn30m04m	Exp	3	1.81	4	0.42	37.94	3113.33	–
syn40h	Exp	3	0.92	4	0.19	0.55	0.33	–
syn40m	Exp	2	0.35	4	0.97	0.52	484.94	–
syn40m02h	Exp	3	2.15	3	0.31	2073.62	1.03	–
syn40m02m	Exp	3	1.18	3	0.24	5.74	1550.39	–
syn40m03h	Exp	4	4.20	4	0.59	2.88	5.27	–
syn40m03m	Exp	4	2.33	5	0.52	204.94	2921.63	–
syn40m04h	Exp	3	8.56	4	1.02	>36000	20.31	–
syn40m04m	Exp	5	4.61	5	0.87	974.05	8048.34	–
synthes1	Exp	2	0.06	3	0.04	0.24	0.11	–
synthes2	Exp	2	0.07	3	0.05	0.42	0.13	–
synthes3	Exp	2	0.09	6	0.10	0.34	0.13	–
rsyn0805h	Exp	1	0.38	1	0.14	0.40	1.10	–
rsyn0805m	Exp	2	0.49	2	0.25	0.87	53.62	–
rsyn0805m02h	Exp	5	2.38	5	0.71	0.73	3.71	–
rsyn0805m02m	Exp	4	2.41	4	2.16	11.21	1617.65	–
rsyn0805m03m	Exp	3	3.26	3	4.08	10.71	2930.70	–
rsyn0805m04m	Exp	2	2.32	2	2.31	19.17	5202.46	–
rsyn0810m	Exp	1	0.37	2	0.24	1.17	211.18	–
rsyn0810m02h	Exp	3	1.87	3	0.58	1.61	9.79	–
rsyn0810m02m	Exp	3	2.20	4	5.78	49.36	3098.62	–
rsyn0810m03h	Exp	3	3.19	3	1.36	1.99	26.42	–
rsyn0810m03m	Exp	3	4.29	3	6.04	41.61	3582.39	–
rsyn0810m04h	Exp	2	3.54	3	1.31	2.87	8.61	–

Table A.1: MINLPLIB2 instances continued

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
rsyn0810m04m	Exp	3	3.74	4	3.77	52.17	5943.63	–
rsyn0815h	Exp	1	19.15	1	0.27	1.27	1.77	–
rsyn0815m	Exp	2	0.49	2	0.23	1.21	171.89	–
rsyn0815m02m	Exp	4	2.39	5	1.94	58.70	2565.52	–
rsyn0815m03h	Exp	5	11.58	5	5.21	38.80	31.62	–
rsyn0815m03m	Exp	5	5.66	4	4.59	217.30	3914.97	–
rsyn0815m04h	Exp	3	6.16	3	2.03	4.73	20.55	–
rsyn0815m04m	Exp	3	6.40	4	7.78	1609.07	7313.05	–
rsyn0820h	Exp	2	1.02	3	0.42	2.04	1.55	–
rsyn0820m	Exp	2	0.61	2	0.24	3.74	772.36	–
rsyn0820m02h	Exp	2	2.28	3	0.59	2.83	90.89	–
rsyn0820m02m	Exp	3	2.27	3	1.90	712.08	3138.98	–
rsyn0820m03h	Exp	2	3.55	2	1.37	4.72	135.69	–
rsyn0820m03m	Exp	3	4.08	3	5.14	6372.80	5220.60	–
rsyn0820m04h	Exp	4	7.75	4	2.66	6.25	50.72	–
rsyn0820m04m	Exp	3	7.22	3	8.65	13412.29	8314.96	–
rsyn0830h	Exp	3	1.27	3	0.41	2.53	2.84	–
rsyn0830m	Exp	4	0.96	4	0.37	3.37	1012.27	–
rsyn0830m02m	Exp	5	10.95	5	1.83	131.12	9151.72	–
rsyn0830m03h	Exp	2	4.77	2	1.45	6.70	59.98	–
rsyn0830m03m	Exp	4	5.79	4	3.45	4044.25	10519.40	–
rsyn0830m04h	Exp	3	8.44	3	2.35	14.23	209.80	–
rsyn0830m04m	Exp	4	11.62	4	11.47	>36000	12709.29	–
rsyn0840h	Exp	2	1.15	2	0.30	3.22	0.94	–
rsyn0840m	Exp	3	0.86	2	0.26	2.96	1117.90	–
rsyn0840m02h	Exp	2	2.97	3	0.72	5.10	8.43	–
rsyn0840m02m	Exp	3	3.05	4	1.53	675.24	4443.70	–
rsyn0840m03h	Exp	3	7.24	3	1.85	>36000	41.84	–
rsyn0840m03m	Exp	5	7.92	5	2.47	4662.04	10511.67	–
rsyn0840m04h	Exp	2	40.03	2	2.40	18.71	453.32	–
rsyn0840m04m	Exp	4	18.14	4	7.62	>36000	15336.01	–
gbd	SOC	0	0.01	1	0.04	0.19	0.12	0.00

Table A.1: MINLPLIB2 instances continued

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
ravempb	Exp	1	0.79	4	0.33	0.80	0.42	–
port[...]cal050_1	SOC	12	32.66	989	>36000	133.43	452.49	3.31
m3	SOC	0	0.04	1	0.68	0.33	0.38	0.07
m6	SOC	1	0.39	1	0.16	2.07	658.83	0.17
m7	SOC	0	0.42	1	0.59	4.99	10431.03	0.69
m7_ar25_1	SOC	1	0.55	1	0.37	1.90	2763.66	0.16
m7_ar2_1	SOC	1	2.47	1	2.19	5.59	14002.89	1.58
m7_ar3_1	SOC	1	2.33	1	1.88	5.53	25222.75	0.82
m7_ar4_1	SOC	0	0.31	1	0.35	2.08	20537.24	0.84
m7_ar5_1	SOC	0	1.30	1	0.34	11.88	38924.33	0.98
fo7	SOC	4	38.44	3	27.68	89.18	3584.70	23.67
fo7_2	SOC	2	12.52	2	12.52	43.35	6298.85	4.88
fo7_ar25_1	SOC	4	22.95	4	9.87	21.94	16685.13	9.92
fo7_ar2_1	SOC	3	15.19	2	8.68	25.56	16123.12	11.04
fo7_ar3_1	SOC	3	27.00	3	11.61	28.79	16539.34	22.16
fo7_ar4_1	SOC	2	11.31	2	9.61	47.19	14674.12	10.27
fo7_ar5_1	SOC	1	4.44	1	5.66	19.63	16634.28	12.67
fo8	SOC	3	79.22	2	79.50	145.26	6383.13	52.92
fo8_ar25_1	SOC	4	141.68	3	45.80	121.69	23823.27	63.09
fo8_ar2_1	SOC	4	159.12	3	59.24	319.27	19979.89	60.09
fo8_ar3_1	SOC	1	10.34	1	14.65	70.68	20336.26	37.85
fo8_ar4_1	SOC	1	12.03	1	10.53	62.21	21961.80	62.60
fo8_ar5_1	SOC	1	29.66	2	23.26	94.63	24442.99	59.75
fo9	SOC	4	210.11	3	534.56	2079.40	4200.36	227.52
fo9_ar25_1	SOC	6	6390.32	6	1430.17	2819.53	25608.54	1240.89
fo9_ar2_1	SOC	2	490.08	2	205.19	896.42	19595.03	631.46
fo9_ar3_1	SOC	1	18.55	1	16.77	730.51	24190.96	103.84
fo9_ar4_1	SOC	1	56.32	2	40.77	1440.47	32284.58	785.75
fo9_ar5_1	SOC	3	131.24	2	39.47	724.35	30368.10	725.60
flay02h	SOC	2	0.10	2	0.09	0.26	1.37	0.02
flay02m	SOC	2	0.06	2	0.05	0.15	0.10	0.04
flay03h	SOC	8	0.98	8	0.40	0.62	0.30	0.20

Table A.1: MINLPLIB2 instances continued

		Pajarito		Bonmin		SCIP	Knitro	CPLEX
Instance	Conic rep.	Iter	Time	Iter	Time	Time	Time	Time
flay03m	SOC	8	0.44	8	0.17	0.26	0.14	0.24
flay04h	SOC	24	23.43	24	19.92	3.75	6.60	1.14
flay04m	SOC	22	8.24	22	4.43	1.98	2.54	1.00
flay05h	SOC	164	6709.06	181	6583.08	221.67	357.72	96.62
flay05m	SOC	171	5030.20	180	3258.45	51.94	118.96	68.91
flay06h	SOC	31	>36000	30	>36000	13327.17	883.97	6958.36
flay06m	SOC	56	>36000	68	>36000	2803.53	279.87	4752.04
o7	SOC	8	2778.14	9	1623.33	2074.22	3060.64	526.94
o7_2	SOC	5	803.25	5	435.47	899.41	6423.68	128.95
o7_ar25_1	SOC	3	421.01	4	259.10	433.72	16789.95	455.29
o7_ar2_1	SOC	1	72.03	1	41.51	209.30	15504.16	68.66
o7_ar3_1	SOC	3	1041.48	4	338.68	874.36	17193.08	875.63
o7_ar4_1	SOC	7	2665.40	7	1486.87	1080.95	17803.19	535.17
o7_ar5_1	SOC	4	662.44	4	309.86	545.20	21972.83	216.84
o8_ar4_1	SOC	3	7192.54	4	2736.05	6939.85	26448.75	8447.35
o9_ar4_1	SOC	6	14143.93	5	7248.84	34990.47	31569.13	21722.78
gams01	**ExpSOC**	**6**	**23414.37**	**>19**	**>36000**	**>36000**	**>36000**	–

Bibliography

[1] Kumar Abhishek, Sven Leyffer, and Jeff Linderoth. FilMINT: An outer approximation-based solver for convex mixed-integer nonlinear programs. *INFORMS Journal on Computing*, 22(4):555–567, 2010.

[2] Tobias Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[3] Amir Ali Ahmadi and Georgina Hall. *Sum of Squares Basis Pursuit with Linear and Second Order Cone Programming.* Contemporary Mathematics, 2017.

[4] Amir Ali Ahmadi, Alex Olshevsky, Pablo A. Parrilo, and John N. Tsitsiklis. NP-hardness of deciding convexity of quartic polynomials and related problems. *Mathematical Programming*, 137(1-2):453–476, 2013.

[5] P. Belotti, T. Berthold, and K. Neves. Algorithms for discrete nonlinear optimization in FICO Xpress. In *2016 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pages 1–5, July 2016.

[6] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization.* Society for Industrial and Applied Mathematics, 2001.

[7] A. Ben-Tal and A. Nemirovski. Lectures on modern convex optimization. `www2.isye.gatech.edu/~nemirovs/Lect_ModConvOpt.pdf`, 2013.

[8] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization.* Princeton University Press, 2009.

[9] Aharon Ben-Tal and Arkadi Nemirovski. On polyhedral approximations of the second-order cone. *Mathematics of Operations Research*, 26(2):193–205, 2001.

[10] Timo Berthold, Stefan Heinz, and Stefan Vigerske. Extending a CIP framework to solve MIQCPs. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, pages 427–444, New York, NY, 2012. Springer New York.

[11] Dimitri P. Bertsekas. *Nonlinear Programming.* Athena Scientific, 2nd edition, 1999.

[12] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[13] Robert Bixby, Chris Maes, and Renan Garcia. Recent developments in the Gurobi optimizer. Gurobi Optimization Workshop, INFORMS Annual Meeting, Philadelphia, PA, November, 2015.

[14] Pierre Bonami, Lorenz T. Biegler, Andrew R. Conn, Gérard Cornuéjols, Ignacio E. Grossmann, Carl D. Laird, Jon Lee, Andrea Lodi, François Margot, Nicolas Sawaya, and Andreas Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

[15] Pierre Bonami, Mustafa Kılınç, and Jeff Linderoth. Algorithms and software for convex mixed integer nonlinear programs. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 1–39. Springer New York, 2012.

[16] Pierre Bonami, Jon Lee, Sven Leyffer, and Andreas Wächter. On branching rules for convex mixed-integer nonlinear optimization. *Journal of Experimental Algorithmics*, 18, 2013.

[17] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[18] R. H. Byrd, J. Nocedal, and R.A. Waltz. Knitro: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer, 2006.

[19] Sebastián Ceria and João Soares. Convex programming for disjunctive convex optimization. *Mathematical Programming*, 86(3):595–614, 1999.

[20] Venkat Chandrasekaran and Parikshit Shah. Relative entropy optimization and its applications. *Mathematical Programming*, 161(1):1–32, 2017.

[21] Chun-Chi Chen, Noushin Ghaffari, Xiaoning Qian, and Byung-Jun Yoon. Optimal hybrid sequencing and assembly: Feasibility conditions for accurate genome reconstruction and cost minimization strategy. *Computational Biology and Chemistry*, pages –, 2017.

[22] C. Coffrin, H. Hijazi, and P. Van Hentenryck. Strengthening the SDP relaxation of AC power flows with convex envelopes, bound tightening, and valid inequalities. *IEEE Transactions on Power Systems*, PP(99):1–1, 2016.

[23] Felipe Cucker, Javier Peña, and Vera Roshchina. Solving second-order conic systems with variable precision. *Mathematical Programming*, 150(2):217–250, 2015.

[24] Alberto Del Pia and Jeff Poskin. Ellipsoidal mixed-integer representability, 2016. Optimization Online, `http://www.optimization-online.org/DB_HTML/2016/05/5465.html`.

[25] Santanu S. Dey and Diego A. Morán R. Some properties of convex hulls of integer points contained in general convex sets. *Mathematical Programming*, 141:507–526, 2013.

[26] Steven Diamond and Stephen Boyd. CVXPY: A python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[27] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[28] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076, 2013.

[29] Sarah Drewes and Stefan Ulbrich. Subgradient based outer approximation for mixed integer second order cone programming. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 41–59. Springer New York, 2012.

[30] Iain Dunning, Joey Huchette, and Miles Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[31] Marco A. Duran and Ignacio E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36(3):307–339, 1986.

[32] R. Fourer and Dominique Orban. DrAmpl: a meta solver for optimization problem analysis. *Computational Management Science*, 7(4):437–463, 2009.

[33] Robert M. Freund, Fernando Ordóñez, and Kim-Chuan Toh. Behavioral measures and their correlation with ipm iteration counts on semi-definite programming problems. *Mathematical Programming*, 109(2):445–475, 2007.

[34] Henrik A. Friberg. CBLIB 2014: a benchmark library for conic mixed-integer and continuous optimization. *Mathematical Programming Computation*, 8(2):191–214, 2016.

[35] Tristan Gally, Marc E. Pfetsch, and Stefan Ulbrich. A framework for solving mixed-integer semidefinite programs. *Optimization Methods and Software*, 0(0):1–39, 2017.

[36] Ralph E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bull. Amer. Math. Soc.*, 64(5):275–278, 09 1958.

[37] Nicholas Gould and Jennifer Scott. A note on performance profiles for benchmarking software. *ACM Transactions on Mathematical Software*, 43(2), 2016.

[38] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. cvxr.com/cvx.

[39] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008.

[40] Michael Grant, Stephen Boyd, and Yinyu Ye. Disciplined convex programming. In Leo Liberti and Nelson Maculan, editors, *Global Optimization*, volume 84 of *Nonconvex Optimization and Its Applications*, pages 155–210. Springer US, 2006.

[41] Oktay Günlük and Jeff Linderoth. Perspective reformulation and applications. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 61–89. Springer New York, 2012.

[42] Omprakash K. Gupta and A. Ravindran. Branch and bound experiments in convex nonlinear integer programming. *Management Science*, 31(12):1533–1546, 1985.

[43] Iiro Harjunkoski, Tapio Westerlund, Ray Pörn, and Hans Skrifvars. Different transformations for solving non-convex trim-loss problems by MINLP. *European Journal of Operational Research*, 105(3):594–603, 1998.

[44] Le Thi Khanh Hien. Differential properties of euclidean projection onto power cone. *Mathematical Methods of Operations Research*, 82(3):265–284, 2015.

[45] Hassan Hijazi, Pierre Bonami, and Adam Ouorou. An outer-inner approximation for separable mixed-integer nonlinear programs. *INFORMS Journal on Computing*, 26(1):31–44, 2014.

[46] Hassan Hijazi and Leo Liberti. Constraint qualification failure in action. *Operations Research Letters*, 44(4):503 – 506, 2016.

[47] Richard D. Hill and Steven R. Waters. On the cone of positive semidefinite matrices. *Linear Algebra and its Applications*, 90:81 – 88, 1987.

[48] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms*. Springer Verlag, Heidelberg, 1996. Two volumes - 2$^{\text{nd}}$ printing.

[49] R. G. Jeroslow and J. K. Lowe. Modeling with integer variables. *Mathematical Programming Studies*, 22:167–184, 1984.

[50] J. E. Kelley Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[51] Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey, editors. *50 Years of Integer Programming 1958-2008 - From the Early Years to the State-of-the-Art*. Springer, 2010.

[52] Mustafa R. Kılınç, Jeff Linderoth, and James Luedtke. Lift-and-project cuts for convex mixed integer nonlinear programs. *Mathematical Programming Computation*, pages 1–28, 2017.

[53] Mustafa R. Kılınç. *Disjunctive Cutting Planes and Algorithms for Convex Mixed Integer Nonlinear Programming.* PhD thesis, University of Wisconsin-Madison, 2011.

[54] Fatma Kılınç-Karzan. On minimal valid inequalities for mixed integer conic programs. *Mathematics of Operations Research*, 41(2):477–510, 2016.

[55] Sunyoung Kim, Masakazu Kojima, and Makoto Yamashita. Second order cone programming relaxation of a positive semidefinite constraint. *Optimization Methods and Software*, 18(5):535–541, 2003.

[56] B. Kocuk, S. S. Dey, and X. Sun. New formulation and strong MISOCP relaxations for AC optimal transmission switching problem. *IEEE Transactions on Power Systems*, PP(99):1–1, 2017.

[57] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, 2016.

[58] Sven Leyffer. *Deterministic Methods for Mixed Integer Nonlinear Programming.* PhD thesis, University of Dundee, 1993.

[59] Sven Leyffer, Jeff Linderoth, Jim Luedtke, Ashutosh Mahajan, Todd Munson, and Meenarli Sharma. Minotaur: Toolkit for mixed integer nonlinear optimization problems. `https://wiki.mcs.anl.gov/minotaur/index.php/Main_Page` accessed 2017-04-16.

[60] Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284(1–3):193–228, 1998. International Linear Algebra Society (ILAS) Symposium on Fast Algorithms for Control, Signals and Image Processing.

[61] M. Lubin, D. Bienstock, and J. P. Vielma. Two-sided linear chance constraints and extensions. *ArXiv e-prints*, July 2015.

[62] M. Lubin and I. Dunning. Computing in operations research using Julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

[63] M. Lubin, E. Yamangil, R. Bent, and J. P. Vielma. Extended formulations in mixed-integer convex programming. In Quentin Louveaux and Martin Skutella, editors, *Integer Programming and Combinatorial Optimization: 18th International Conference, IPCO 2016, Liège, Belgium, June 1-3, 2016, Proceedings*, pages 102–113. Springer International Publishing, 2016.

[64] M. Lubin, E. Yamangil, R. Bent, and J. P. Vielma. Polyhedral approximation in mixed-integer convex optimization. *ArXiv e-prints*, 2016.

[65] M. Lubin, I. Zadik, and J. P. Vielma. Mixed-integer convex representability. *ArXiv e-prints*, November 2016.

[66] M. Lubin, I. Zadik, and J. P. Vielma. Regularity in mixed-integer convex representability. *ArXiv e-prints*, 2017.

[67] Stephen J. Maher, Tobias Fischer, Tristan Gally, Gerald Gamrath, Ambros Gleixner, Robert Lion Gottwald, Gregor Hendel, Thorsten Koch, Marco E. Lübbecke, Matthias Miltenberger, Benjamin Müller, Marc E. Pfetsch, Christian Puchert, Daniel Rehfeldt, Sebastian Schenker, Robert Schwarz, Felipe Serrano, Yuji Shinano, Dieter Weninger, Jonas T. Witt, and Jakob Witzig. The SCIP Optimization Suite 4.0. Technical Report 17-12, ZIB, Takustr.7, 14195 Berlin, 2017.

[68] R. Kipp Martin. *Large Scale Linear and Integer Optimization: A Unified Approach: A Unified Approach*. Springer US, 1999.

[69] S. Misra, M. Vuffray, A. Y. Lokhov, and M. Chertkov. Towards Optimal Sparse Inverse Covariance Selection through Non-Convex Optimization. *ArXiv e-prints*, March 2017.

[70] Hans Mittelmann. MINLP benchmark. `http://plato.asu.edu/ftp/minlp_old.html` accessed 2016-05-13.

[71] Mosek ApS. Modeling Cookbook revision 2.0.1. `http://docs.mosek.com/MOSEKModelingCookbook-letter.pdf` accessed 2017-04-12.

[72] Harsha Nagarajan, Sivakumar Rathinam, Swaroop Darbha, and Kumbakonam Rajagopal. Algorithms for synthesizing mechanical systems with maximal natural frequencies. *Nonlinear Analysis: Real World Applications*, 13(5):2154 – 2162, 2012.

[73] George L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1999.

[74] Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 2016.

[75] Mert Pilanci, Martin J. Wainwright, and Laurent Ghaoui. Sparse learning via boolean relaxations. *Math. Program.*, 151(1):63–87, June 2015.

[76] I. Quesada and I.E. Grossmann. An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Computers & Chemical Engineering*, 16(10):937–947, 1992.

[77] James Renegar. Some perturbation theory for linear programming. *Mathematical Programming*, 65(1):73–91, 1994.

[78] R.T. Rockafellar. *Convex Analysis*. Princeton landmarks in mathematics and physics. Princeton University Press, 1997.

[79] Matthew Saltzman, László Ladányi, and Ted Ralphs. The COIN-OR Open Solver Interface: Technology overview. Presented at CORS/INFORMS Banff May 2004. `https://www.coin-or.org/Presentations/CORS2004-OSI.pdf`.

[80] Santiago Akle Serrano. *Algorithms for unsymmetric cone optimization and an implementation for problems with the exponential cone*. PhD thesis, Stanford University, 2015.

[81] X. Shen, S. Diamond, Y. Gu, and S. Boyd. Disciplined convex-concave programming. *ArXiv e-prints*, 2016.

[82] K. Sundar, H. Nagarajan, M. Lubin, L. Roald, S. Misra, R. Bent, and D. Bienstock. Unit commitment with N-1 security and wind uncertainty. In *2016 Power Systems Computation Conference (PSCC)*, pages 1–7, June 2016.

[83] Mohit Tawarmalani and Nikolaos V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2):225–249, 2005.

[84] Andrea Tramontani. Solving MISOCP in CPLEX by cone disaggregation and lift-and-project cuts. 2015 Mixed Integer Programming Workshop, June 1st – 4th, 2015. The Gleacher Center, Chicago, IL.

[85] Madeleine Udell, Karanveer Mohan, David Zeng, Jenny Hong, Steven Diamond, and Stephen Boyd. Convex optimization in Julia. In *Proceedings of HPTCDL '14*, pages 18–28, Piscataway, NJ, USA, 2014. IEEE Press.

[86] Juan Pablo Vielma. Mixed integer linear programming formulation techniques. *SIAM Review*, 57(1):3–57, 2015.

[87] Juan Pablo Vielma, S. Ahmed, and G. Nemhauser. A lifted linear programming branch-and-bound algorithm for mixed integer conic quadratic programs. *INFORMS Journal on Computing*, 20:438–450, 2008.

[88] Juan Pablo Vielma, Iain Dunning, Joey Huchette, and Miles Lubin. Extended formulations in mixed integer conic quadratic programming. *Mathematical Programming Computation*, pages 1–50, 2016.

[89] Stefan Vigerske. MINLPLIB2 library. `http://www.gamsworld.org/minlp/minlplib2/html/` accessed 2016-05-13.