Novel Relaxation Techniques for Global Optimization of NLPs and MINLPs

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Chemical Engineering

Carlos Jose Nohra Khouri

B.S. Chemical Engineering, Universidad Central de Venezuela
M.S. Simulation Sciences, RWTH Aachen University

Carnegie Mellon University
Pittsburgh, PA

April, 2020

Acknowledgments

First and foremost, I would like to thank my advisor, Professor Nick Sahinidis for his guidance and support throughout my Ph.D. Working with Nick over the last five years has been a great privilege. He has been a pioneer in the field of deterministic global optimization. He has not only made significant theoretical and algorithmic contributions to this area, but has also developed efficient software tools which are widely used both in industry and academia. I truly admire his work ethic, dedication and responsiveness.

I am also grateful to my dissertation committee members, Professor Ignacio Grossmann, Professor Chrysanthos Gounaris and Professor Javier Peña, for their helpful guidance and suggestions. I have benefited a lot from attending the optimization courses offered by Professor Grossmann and Professor Gounaris. I am specially thankful to Professor Peña for his courses, Convex Optimization and Convex Analysis, which provided me with a deep understanding of the background material needed for this work.

I would also like to thank Dr. Arvind Raghunathan for providing me with the opportunity of completing two summer internships in his group at Mitsubishi Electric Research Laboratories (MERL). A significant part of this work has been done in collaboration with Arvind. Working with him has been a great experience. It is very difficult to find researchers of his caliber. I look forward to further collaboration with him as I join MERL as a Visiting Research Scientist.

I would also like to acknowledge the Department of Chemical Engineering and the Center for Advanced Process Decision-making (CAPD) for providing funding for this research.

I am thankful to the following former and current members of the Sahinidis Optimization Group with whom I interacted at different points of my Ph.D.: Mustafa Kılınç, Nikos

Ploskas, Aida Khajavirad, Nick Austin, Sreekanth Rajagopalan, Yash Puranik, Zach Wilson, Ben Sauk, Marissa Engle, Tong Zhang, Chenglin Zheng, Yang Yang, Yi Zhang, Brad Johnson, Yijia Sun, Christian Hubbs, Kaiwen Ma, Owais Sarwar and Anatoliy Kuznetsov. I enjoyed the extremely friendly atmosphere of this group. Mustafa, Nikos and Aida helped me during the early stages of my Ph.D. I thank Mustafa for answering some of the questions that I had about BARON at the beginning of my Ph.D. I am grateful to Nikos for helping me with several computer issues and for the interesting conversations that we used to have in the third floor office. I thank Aida for her guidance when I was starting to review the vast literature on convexification. Nick, Sree, Yash and Zach were senior Ph.D. students when I joined the group. I appreciate the very useful feedback that they always gave me at my group meeting presentations. Ben, Marissa and Tong are part of my Ph.D. class. During our first year, we took several courses together, and as the years passed, we had plenty of interesting discussions, both on scientific and philosophical matters. Yi visited the group when I was in my third year. We had many fruitful discussions and I had the pleasure to collaborate with him on one of his papers. Brad, Yijia, Christian, Kaiwen, Owais and Anatoliy joined the group as Ph.D. students after I did. I want to thank them for making my time in the third floor office very enjoyable.

I am grateful for having been part of a very collaborative PSE group and a great Department of Chemical Engineering. In particular, I am thankful to the following students, visitors and post-docs with whom I have interacted over the last five years at CMU: Cristiana Lara, Braulio Brunaud, Christopher Hanselman, Zixi Zhao, Qi Chen, Yixin Ye, Joyce Yu, David Thierry, Akang Wang, Dana McGuffin, Charles Sharkey, Rajarshi Sengupta, Michael Davidson, David Bernal, Can Li, Saif Rahaman, Natalie Isenberg, Hector Perez, Vibhav Dabadghao, Aliakbar Izadkhah, John Villaraga, Christina Schenk, Cornelius Masuku, Jan Kronqvist, Maria Paz Ochoa, Albania Villaroel, Kai Liu, Michael Short, Victor Anselmo and Ana Somoza.

Last but not the least, I want to express my deepest gratitude to my parents, Sarkis and Rose, for their unconditional love and support, and my siblings, Daniel, Carolina and Gabriel, for being my best friends. I am specially thankful to my parents because they always placed a very strong emphasis on education, raising my siblings and me in a multilingual home and exposing us to an environment that sparked our intellectual curiosity from an early age. My parents are an excellent example of the great success that the Lebanese diaspora has had all over the world. This thesis is dedicated to them.

Abstract

Over the last three decades rapid advances in computer technology coupled with new theoretical developments have led to significant progress in deterministic global optimization. While this progress has been considerable, this remains a relatively underdeveloped area, as there exists many classes of problems that global optimization algorithms are unable to solve to global optimality.

In this thesis, we present computational methodologies aimed at improving the performance of branch-and-bound-based global optimization algorithms. To this end, we propose novel relaxation and domain partitioning strategies for various classes of nonconvex nonlinear programs (NLPs) and mixed-integer nonlinear programs (MINLPs).

In the first part of this thesis, we consider nonconvex optimization problems containing convex-transformable functions. We introduce a new class of cutting planes derived by exploiting convex-transformability of intermediate expressions of factorable programs.

In the second part of this thesis, we turn our attention to nonconvex mixed-integer quadratic programs (MIQPs). We present a family of convex quadratic relaxations derived by convexifying nonconvex quadratic functions through uniform diagonal perturbations of the quadratic matrix. We investigate the theoretical properties of these quadratic relaxations and show that they are equivalent to some particular semidefinite programs. We also introduce novel branching variable selection strategies which can be used in conjunction with the proposed quadratic relaxations.

In the third part of this thesis, we consider a related class of convex quadratic relaxations. In particular, we propose a new class of quadratically constrained programming (QCP) relaxations derived via convex quadratic cuts obtained from non-uniform diagonal pertur-

bations of the quadratic matrix. We show that these relaxations are an outer-approximation of a semi-infinite convex program which under certain conditions is equivalent to a well-known semidefinite program relaxation.

To demonstrate the computational benefits of the ideas investigated in this thesis, we implement the proposed relaxation and domain partitioning strategies into the state-of-the art global optimization solver BARON. We test our implementation by conducting extensive computational studies on a variety of nonconvex problems. Results demonstrate that, for many test problems, the proposed techniques lead to order-of-magnitude speedups, resulting in a new version of BARON which outperforms other widely used global optimization solvers.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

We consider optimization problems of the form:

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & g(x) \leq 0 \\ & x \in \mathcal{X} \subseteq \mathbb{R}^{n-n_d} \times \mathbb{Z}^{n_d} \end{array} \tag{1.1}$$

where $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}^m$ are factorable functions which may be nonconvex. The formulation in (1.1) subsumes many classes of nonconvex nonlinear programs (NLPs) and mixed-integer nonlinear programs (MINLPs) which arise in variety of applications such as synthesis of process networks [35], pooling and blending in refinery operations [65], product design in mechanical engineering [44], molecular conformation [69], protein folding [67], molecular design of refrigerants [78], optimization of metabolic networks [70], facility location and quadratic assignment [25, 49, 51, 55], and max-cut problems [33]. Due to the nonconvexities in f and g and the presence of discrete variables, problem (1.1) may exhibit multiple local optima and can be very challenging to solve to global optimality.

State-of-the-art deterministic global optimization solvers rely on branch-and-bound algorithms in order to solve problems of the form (1.1) to global optimality. These algorithms were initially devised to solve discrete optimization problems [27, 50], and were subsequently adapted for solving more general problems involving both discrete and continuous variables [30]. The branch-and-bound procedure generates lower and upper bounds on the global solution of (1.1) over successively refined partitions of the search space. The

lower bounds are obtained by solving relaxations of (1.1), whereas the upper bounds are determined via local search or heuristics. This procedure stops when the difference between the best upper and lower bounds is within a user-defined tolerance $\epsilon > 0$. If the lower bounding and partitioning schemes satisfy certain conditions, the branch-and-bound algorithm is guaranteed to converge to the global optimum within ϵ-accuracy [38]. We illustrate the key ideas behind this algorithm through the following example.

Example 1.1. Consider the following nonconvex optimization problem:

$$\begin{aligned} \mu = \min_{x \in \mathbb{R}} \quad & f(x) = 2.3x^4 - 4.5x^2 + 0.5e^{1.5x} \\ \text{s.t.} \quad & x \in [-1.4, 1.0] \end{aligned} \tag{1.2}$$

It is simple to show that a convex relaxation for (1.2) is given by:

$$\begin{aligned} \mu_i^{LBD} = \min_{x \in \mathbb{R}} \quad & r_i(x) = 2.3x^4 - 4.5((\underline{x}_i + \bar{x}_i)x - \underline{x}_i\bar{x}_i) + 0.5e^{1.5x} \\ \text{s.t.} \quad & x \in [\underline{x}_i, \bar{x}_i] \end{aligned} \tag{1.3}$$

where the index i represents the i-th node of the branch-and-bound tree, and $\underline{x}_i$ and $\bar{x}_i$ respectively denote the lower and upper bounds corresponding to this node.

We start the branch-and-bound algorithm at the root-node by setting $i = 1$, $\underline{x}_1 = -1.4$ and $\bar{x}_1 = 1.0$ in (1.3). The solution of the resulting relaxation is attained at $x^* = -0.6$, which leads to the root-node lower bound $\mu_1^{\text{LBD}} = -6.9$. We then apply a local search method to (1.2) by using the relaxation solution as a starting point. This results in the root-node upper bound $\mu_1^{\text{UBD}} = -2.1$. Clearly, $\mu \in [-6.9, -2.1]$. The root-node lower and upper bounding steps are illustrated in Figure 1.1a.

Next, we partition the problem domain by branching on the middle point of the interval $[-1.4, 1.0]$. We create one node where $i = 2$ and $x \in [\underline{x}_2, \bar{x}_2] = [-1.4, -0.2]$, and another node where $i = 3$ and $x \in [\underline{x}_3, \bar{x}_3] = [-0.2, 1.0]$. At each of these nodes we solve a relaxation of the form (1.3), obtaining the lower bounds $\mu_2^{\text{LBD}} = -3.6$ and $\mu_3^{\text{LBD}} = -1.5$ (see Figure 1.1b). Since $\mu_3^{\text{LBD}} > \mu_1^{\text{UBD}}$, we have that $f(x) > \mu, \ \forall x \in [-0.2, 1.0]$. As a result, we fathom node 3 and conclude that $\mu \in [-3.6, -2.1]$.

We can further partition the problem domain by branching on the middle point of the interval $[-1.4, -0.2]$. This leads to one node where $i = 4$ and $x \in [\underline{x}_4, \bar{x}_4] = [-1.4, -0.8]$, and another node where $i = 5$ and $x \in [\underline{x}_5, \bar{x}_5] = [-0.8, -0.2]$. By solving the relaxations of the form (1.3) corresponding to these nodes we obtain the lower bounds $\mu_4^{\text{LBD}} = -2.5$ and $\mu_5^{\text{LBD}} = -1.8$ (see Figure 1.1c). Since $\mu_5^{\text{LBD}} > \mu_1^{\text{UBD}}$, we have that $f(x) > \mu, \ \forall x \in [-0.8, -0.2]$. Hence, we proceed to fathom node 5. As this is an illustrative example, we stop this algorithm at node 4 and conclude that $\mu \in [-2.5, -2.1]$. The search tree corresponding to this example is shown in Figure 1.1d.

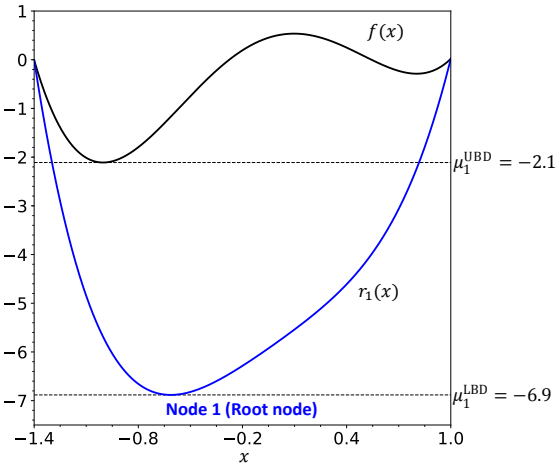
The performance of branch-and-bound algorithms is heavily influenced by several factors including: (i) the quality of the bounds obtained during the lower and upper bounding steps, (ii) the efficiency of the domain reduction methods used throughout the search, and (iii) the strategies employed to partition the problem domain [91].

The quality of the upper bounds is determined by the type of local search and heuristic methods used to find feasible solutions, whereas the quality of the lower bounds depends on the tightness of the relaxations constructed throughout the branch-and-bound tree. Tight relaxations lead to tight bounds, and often speed up the convergence of branch-and-bound algorithms.

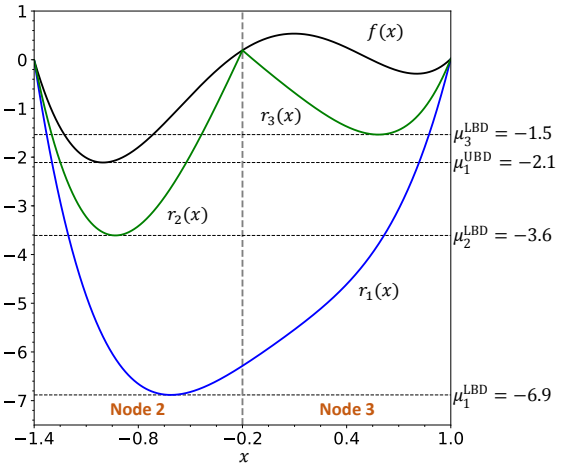
As their name suggests, domain reduction methods aim to reduce the size of the search space by excluding regions that do not contain optimal solutions. This reduction is achieved by tightening variable bounds through the application of feasibility-based and optimality-based techniques [75], and the exploitation of the first-order optimality conditions of the original problem [73, 97]. Although these strategies are not required to ensure convergence to the global optimum, they typically lead to significant improvements in the efficiency of branch-and-bound algorithms.

Domain partitioning strategies involve the splitting of the search space by selecting a branching variable and a branching point. These strategies have a very significant impact

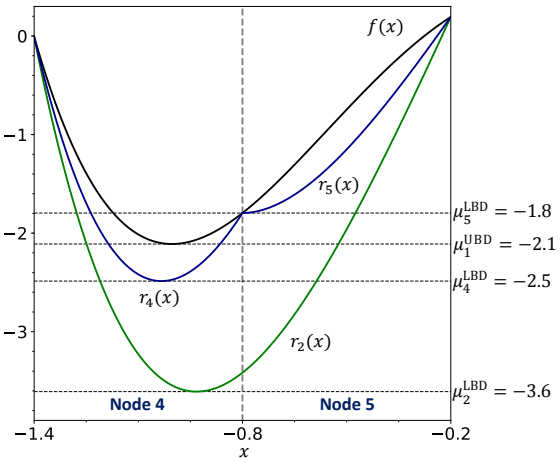
on the structure and size of the search tree and can considerably affect the performance of branch-and-bound algorithms [92].



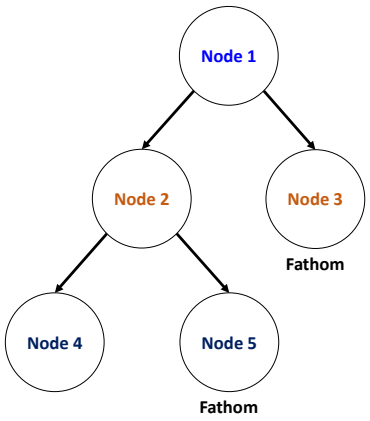
(a) Lower and upper bounding at node 1.



(b) Lower bounding at nodes 2 and 3.



(c) Lower bounding at nodes 4 and 5.



(d) Search tree.

Figure 1.1: Branch and bound procedure for Example 1.1.

Given the challenging nature of the problems that typically arise in global optimization, over the past three decades there has been very active research in the development of novel bounding schemes, domain reduction methods and partitioning strategies for

branch-and-bound algorithms. Many of these new developments have been incorporated into efficient branch-and-bound-based software packages which are capable of solving a variety of nonconvex problems to global optimality. Some examples of these solvers are ANTIGONE [64], BARON [77], COUENNE [12], CPLEX [41], GUROBI [36], LINDOGLOBAL [54] and SCIP [13].

Even though the aforementioned advances have had a very positive impact in numerous scientific and engineering applications, there currently exist many classes of problems which global optimization solvers are unable to solve to global optimality.

1.2 Outline of the thesis

In this thesis, we push the state-of-the-art in deterministic global optimization by introducing novel relaxation and partitioning strategies for various classes of nonconvex NLPs and MINLPs. Throughout the thesis, we make important theoretical and algorithmic contributions, and demonstrate the computational benefits of the proposed strategies by developing efficient implementations which are integrated into the global optimization solver BARON.

We start in Chapter 2 by considering nonconvex optimization problems which contain convex-transformable functions. We first present algorithms for identification of convex-transformable functions in general nonconvex problems. We then introduce a new class of cutting planes based on recently developed relaxations for convex-transformable functions. We integrate our recognition and cutting plane generation algorithms into BARON, and test our implementation by conducting numerical experiments on various classes of nonconvex problems. Results indicate that the proposed cutting planes considerably accelerate the convergence speed of the branch-and-bound algorithm.

In Chapter 3, we turn our attention to nonconvex quadratic programs (QPs) and mixed-

integer quadratic programs (MIQPs). In particular, we present a family of convex quadratic relaxations which are derived by convexifying nonconvex quadratic functions through perturbations of the quadratic matrix. We investigate the theoretical properties of these quadratic relaxations and show that they are equivalent to some particular semidefinite programs. We also introduce novel branching variable selection strategies which are motivated by the proposed quadratic relaxations. The new relaxation and branching techniques are implemented in BARON, and tested by conducting numerical experiments on a large collection of problems. Our numerical results show that the proposed implementation leads to a very significant improvement in the performance of BARON, resulting in order-of-magnitude speedups for many test problems.

Motivated by these results, in Chapter 4, we consider a related class of convex quadratic relaxations. In particular, we propose a new class of quadratically constrained programming (QCP) relaxations which are derived via convex quadratic cuts. To construct these quadratic cuts, we solve a separation problem involving a linear matrix inequality with a special structure that allows the use of specialized solution algorithms. We show that our relaxations are an outer-approximation of a semi-infinite convex program which under certain conditions is equivalent to a well-known semidefinite program relaxation. We implement these new relaxations in BARON and demonstrate their benefits by conducting an extensive computational study.

We conclude in Chapter 5 by summarizing the main contributions of this thesis and suggesting directions for future work.

Chapter 2

Global optimization of nonconvex problems with convex-transformable intermediates

2.1 Introduction

In this chapter, we consider the global optimization of nonconvex NLPs and MINLPs of the form:

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & g(x) \leq 0 \\ & x \in \mathcal{X} \subseteq \mathbb{Z}^{n_d} \times \mathbb{R}^{n-n_d} \end{array} \tag{2.1}$$

where $f : \mathcal{X} \to \mathbb{R}$, $g : \mathcal{X} \to \mathbb{R}^m$, and the objective function and/or constraints contain convex-transformable functions. A continuous real-valued function ϕ defined over a convex set $\mathcal{C} \subseteq \mathbb{R}^n$ is said to be convex-transformable (resp. concave-transformable) or G-convex (resp. G-concave) if there exists a continuous real-valued increasing function G defined on the range of ϕ, such that $G(\phi(x))$ is convex (resp. concave) over $\mathcal{C}$ [7]. Problems containing convex-transformable functions arise in a wide variety of scientific and engineering applications, including the synthesis of process networks [35], pooling and blending in refinery operations [65], molecular design of refrigerants [78], product design in mechanical engineering [44], and optimization of metabolic networks [70].

One of the most popular approaches for constructing convex relaxations of nonconvex optimization problems, including problems containing convex-transformable functions, is based on factorable programming techniques [60, 86, 92]. Given a factorable function,

these techniques proceed by introducing intermediate variables and constraints in an iterative manner, until each intermediate expression can be outer-approximated by its convex and/or concave envelopes. By applying this procedure to all factorable functions appearing in a given nonconvex optimization problem, it is possible to construct a convex relaxation, whose solution provides a valid bound on the optimal objective function value of the original problem. Due to their simplicity, factorable programming techniques have been successfully implemented in most global optimization packages. An important drawback of these techniques is the fact that they often result in large relaxation gaps.

With the aim of obtaining tighter relaxations of nonconvex optimization problems, considerable attention has been devoted in recent years to the problem of constructing the convex and concave envelopes of a nonconvex function. While significant advances have been made in this area, there are only a few cases in which it is possible to obtain closed-form expressions for the envelopes, or alternatively, develop a computationally efficient algorithm for generating facets of the envelopes. These instances involve functions with polyhedral envelopes, and several classes of low-dimensional functions with nonpolyhedral envelopes. For details on some of these convexification results, we refer the reader to [2, 9, 10, 42, 46, 47, 61, 62, 74, 80, 88, 89, 90]

An alternative method for strengthening factorable relaxations of nonconvex optimization problems is based on the use of functional transformations. A particular example of this approach is the use of power and exponential transformations to convexify signomial terms [52, 56, 57, 58]. A more general transformation scheme for constructing outer-approximations of nonconvex functions was recently proposed by Khajavirad et al. [45]. This technique exploits convex transformability of component functions of factorable programs, and it differs from other methods in that it is applicable to various classes of functional forms including signomials, products and ratios of convex and/or concave functions, and logarithmically-concave functions. As illustrated in [45], this transformation

method often leads to relaxations which are considerably tighter than those obtained by standard approaches.

Motivated by the potential of these new relaxations to enhance the performance of global solvers, in this chapter, we introduce a new class of cutting planes for convex-transformable functions, and describe its implementation into the branch-and-bound global solver BARON. The proposed implementation involves a recognition tool which can be used to identify convex-transformable functions in general nonconvex problems, including those present in intermediate expressions of factorable functions. Our cutting plane generation scheme is based on the construction of supporting hyperplanes to these new convex relaxations, which are then used at each node of the branch-and-bound tree to tighten BARON's polyhedral relaxations. By integrating the proposed cutting plane generation strategy at every node of the branch-and-bound tree, we are able to exploit BARON's bound tightening capabilities to obtain tight bounds for relaxation construction, as well as use the generated cutting planes for range reduction. We test our implementation by conducting extensive numerical experiments on a large collection of NLPs and MINLPs selected from publicly available test sets. Results demonstrate that the generated cutting planes accelerate the convergence speed of the branch-and-bound algorithm, by significantly reducing computational time, number of nodes in the search tree, and required memory.

The remainder of this chapter is organized as follows. In §2.2 we review relaxation construction techniques for convex-transformable functions. In §2.3, we describe our implementation and provide details on how the proposed cutting plane generation scheme is integrated into a branch-and-bound-global solver. In §2.4, we present the results of an extensive computational study analyzing the effect of the proposed cutting planes on the performance of the branch-and-bound algorithm. Finally, in §2.5 we provide conclusions from this work.

2.2 Relaxations of convex-transformable functions

In this section, we review relaxation construction techniques for various classes of G-convex functions. We start by illustrating how convex-transformability of nonconvex functions can be exploited for the construction of outer-estimators. Let ϕ be a G-convex function defined over the convex set $\mathcal{C} \subseteq \mathbb{R}^n$, where G is a transforming function defined over the range of ϕ, which we will denote by I_ϕ. Then, it can be shown that a convex underestimator for ϕ is given by (see Section 3 in [45] for details)

$$\tilde{\phi}^G(x) = \bar{G}^{-1}(G(\phi(x))) \tag{2.2}$$

where $\bar{G}$ is an overestimator for G over I_ϕ. Now, suppose that G is a convex function, and denote by $\underline{\phi}$ and $\bar{\phi}$, lower and upper bounds on ϕ over $\mathcal{C}$, respectively. Then, by using the concave envelope of G over I_ϕ, it is easy to verify that in this case (2.2) is equivalent to

$$\tilde{\phi}^G(x) = \big(G(\phi(x)) - G(\underline{\phi})\big)\left(\frac{\bar{\phi} - \underline{\phi}}{G(\bar{\phi}) - G(\underline{\phi})}\right) + \underline{\phi} \tag{2.3}$$

As an example, consider the univariate nonconvex function $\phi(x) = -(x^3 + x)$ defined over $\mathcal{C} = [-0.5, 0.5]$, and the transforming the function $G(t) = \exp(3t)$ defined on the range of ϕ over $\mathcal{C}$. It is simple to check that the composite function $G(\phi(x))$ is convex, which in turn implies that ϕ is convex-transformable. In addition, by employing (2.3), the following convex underestimator for ϕ over $\mathcal{C}$ can be obtained

$$\tilde{\phi}^G(x) = 0.196\ \exp(-3(x^3 + x)) - 0.655 \tag{2.4}$$

Note that for a given convex-transformable function the choice of the transforming function is not unique, i.e., there may exist many transforming functions for which $G(\phi(x))$ is convex, and which obviously can be used to derive underestimators of the form (2.3). Therefore, an important question that arises in this context is how we can choose G in order to obtain the tightest possible underestimator in (2.3). This question was addressed in [45]; in particular, the authors showed that the tightest relaxation of the form (2.3) is

obtained when G is equal to a *least convexifying transformation* for ϕ. Given a G^*-convex function ϕ, G^* is considered to be a least convexifying transformation for ϕ, if for every G for which ϕ is G-convex, GG^{*-1} is a convex function. For details on how to construct a least convexifying transformation for a given convex-transformable function, we refer the reader to [7, 45].

Now, we revisit the example considered above. In this case, it can be shown that a least convexifying transformation for ϕ is given by $G^*(t) = \exp(9t/8)$. By substituting G^* in (2.3), we obtain the following convex underestimator for ϕ

$$\tilde{\phi}^{G^*}(x) = 0.820\ \exp(-1.125(x^3 + x)) - 1.031 \tag{2.5}$$

The underestimators given in (2.4) and (2.5) are compared in Figure 2.1. As seen in the figure, $\tilde{\phi}^{G^*}$ is considerably tighter than $\tilde{\phi}^{G}$. Note that the univariate nonconvex function ϕ considered in this example has a nonpolyhedral convex envelope over $\mathcal{C} = [-0.5, 0.5]$ given by

$$\mathrm{conv}_{\mathcal{C}}\phi(x) = \begin{cases} \phi(x), & \text{if } -0.5 \leq x \leq w, \\ \phi(\bar{x}) + \phi'(w)(x - 0.5), & \text{if } w \leq x \leq 0.5. \end{cases} \tag{2.6}$$

where w is a point satisfying $\phi(w) = \phi(0.5) + \phi'(w)(w - 0.5)$ (see Figure 2.1). Since, by definition, the convex envelope of a function is its tightest possible convex underestimator, it is obvious that for this example $\mathrm{conv}_{\mathcal{C}}\phi(x)$ dominates $\tilde{\phi}^{G}$ and $\tilde{\phi}^{G^*}$. However, as mentioned earlier, for general nonconvex functions the characterization of the envelopes is a very difficult problem. Therefore, in cases in which the envelopes are not available the transformation outer-estimators discussed here can be used for tightening convex relaxations constructed through factorable programming techniques.

Analogous results for G-concave functions can be obtained in a similar manner. The transformation scheme outlined above was employed in [45] to derive underestimators (resp. overestimators) for various classes of G-convex (resp. G-concave) functions. In the following subsections, we summarize these results.

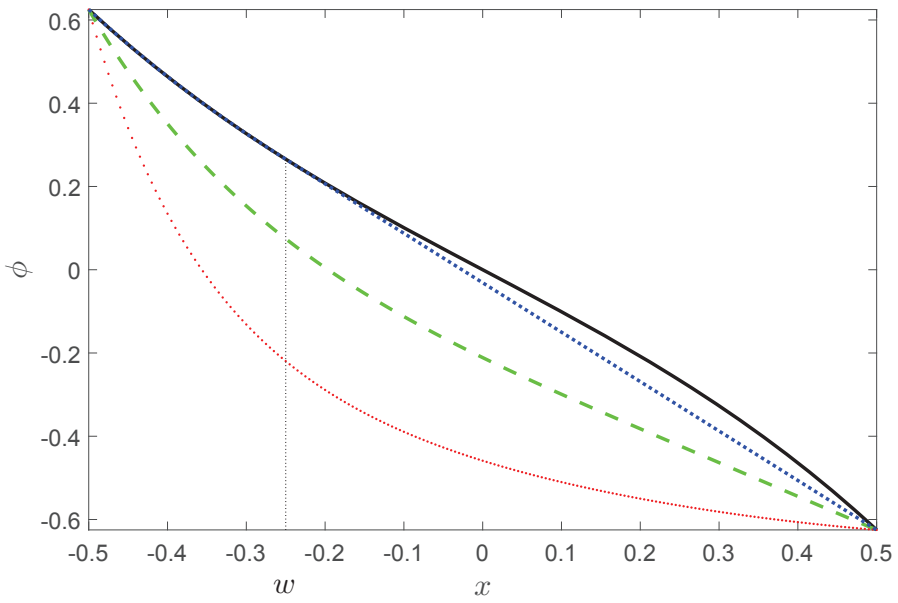


Figure 2.1: Comparison of underestimators for $\phi(x) = -(x^3 + x)$ over $\mathcal{C} = [-0.5, 0.5]$. The function ϕ is plotted in solid black, the transformation underestimators $\tilde{\phi}^G$ and $\tilde{\phi}^{G^*}$ in dotted red and dashed green, respectively, and the convex envelope of ϕ in dotted blue.

2.2.1 Signomials

In this subsection, we consider signomial functions defined over a subset of the nonnegative orthant. We start by reviewing conditions under which a signomial function is convex-transformable (resp. concave-transformable), and present its least convexifying (resp. concavifying) transformation, which is then used to construct a convex underestimator (resp. concave overestimator).

Proposition 2.1. (Proposition 10 in [45]) *Consider the function* $\phi = \prod_{i \in I} x_i^{\alpha_i}$, $\alpha_i \in \mathbb{R} \setminus \{0\}$, $\forall i \in I = \{1, \ldots, n\}$ *defined over a subset of the nonnegative orthant. The function* ϕ *is G-convex if and only if* $\alpha_i < 0$ *for all* $i \in I \setminus \{j\}$ *and* $\sum_{i \in I \setminus \{j\}} |\alpha_i| < \alpha_j < \sum_{i \in I \setminus \{j\}} |\alpha_i| + 1$. *Moreover, a least convexifying transformation for* ϕ *is given by*

$$G^*(t) = t^{\frac{1}{\sum_{i \in I} \alpha_i}} \tag{2.7}$$

Proposition 2.2. (Proposition 11 in [45]) *Consider the function* $\phi = \prod_{i\in I} x_i^{\alpha_i}$, $\alpha_i \in \mathbb{R}\setminus\{0\}$, $\forall i \in I = \{1,\dots,n\}$ *defined over a subset of the nonnegative orthant. The function* ϕ *is G-concave if and only if one of the following holds:*

(i) $\alpha_i > 0$ *for all* $i \in I$ *and* $\sum_{i\in I}\alpha_i > 1$,

(ii) $\alpha_j < 0$ *for some* $j \in I$ *such that* $\sum_{i\in I\setminus\{j\}}\alpha_i < |\alpha_j|$.

Moreover, a least concavifying transformation for ϕ *is given by* (2.7) *when condition (i) is met and by*

$$G^*(t) = -t^{\frac{1}{\sum_{i\in I}\alpha_i}} \tag{2.8}$$

when condition (ii) is met.

Now, we derive underestimators and overestimators for the signomial function ϕ over a subset $\mathcal{C}$ of the nonnegative orthant. Denote by $\underline{\phi}$ and $\bar{\phi}$ the lower and upper bounds on ϕ over $\mathcal{C}$, respectively, and let $\xi = \sum_{i\in I}\alpha_i$. By Proposition 2.1 and (2.3), the following is a convex underestimator for ϕ:

$$\tilde{\phi}^G(x) = \left(\phi^{\frac{1}{\xi}} - \underline{\phi}^{\frac{1}{\xi}}\right)\left(\frac{\bar{\phi}-\underline{\phi}}{\bar{\phi}^{\frac{1}{\xi}} - \underline{\phi}^{\frac{1}{\xi}}}\right) + \underline{\phi} \tag{2.9}$$

Using a similar argument, it follows from Proposition 2.2 that a concave overestimator for ϕ is also given by (2.9). Next, we illustrate the above relaxation construction technique through an example.

Consider the function $\phi(x) = x_1^{1.1}x_2^{0.3}$, $x_1 \in [1,5]$, $x_2 \in [0.5,10]$. We first construct a concave overestimator using factorable programming techniques. Let $x_3 = x_1^{1.1}$, and $x_4 = x_2^{0.3}$. Then, by overestimating x_3 with its concave envelope, and using the McCormick envelopes for the bilinear term x_3x_4, we obtain the following overestimator for $\phi(x)$

$$\tilde{\phi}^S(x) = \min\{5.87x_2^{0.3} + 0.99x_1 - 4.95,\ x_2^{0.3} + 2.43x_1 - 2.43\} \tag{2.10}$$

By proposition 2.2, the function ϕ is concave-transformable with $G^* = t^{1/1.4}$. Then, by using (2.9), the following overestimator for ϕ can be constructed

$$\tilde{\phi}^{G^*}(x) = 2.21\big(x_1^{1.1}x_2^{0.3}\big)^{1/1.4} - 1.09 \tag{2.11}$$

We compare both overestimators in Figure 2.2 at different cross-sections of the domain. As observed in the figure, the transformation overestimator is tighter in the center of the box, whereas the factorable overestimator dominates near the boundaries of the domain. This example illustrates that transformation outer-estimators may not globally dominate standard factorable outer-estimators. In fact, as shown in [45], the total relaxation gap of the transformation method may become larger than that of the standard factorable approach for signomials in higher dimensions and/or with larger exponents. With the aim of reducing this undesirable increase in the relaxation gaps, in [45], the authors proposed a recursive transformation and relaxation (RT) scheme for overestimating signomials containing three or more terms. This approach combines factorable and transformation relaxations in order to obtain tighter overestimators. We illustrate the benefits of this method through the following example.

Consider the function $\phi(x) = x_1^{0.5}x_2^{0.6}x_3^{0.8}$ defined over $\mathcal{C} = [0,6] \times [0.2,4] \times [1.1,3]$. A factorable decomposition of ϕ is given by

$$x_4 = x_1^{0.5} \qquad x_5 = x_2^{0.6} \qquad x_6 = x_1^{0.8} \qquad x_7 = x_4x_5 \qquad x_8 = x_7x_6 \tag{2.12}$$

By using the concave envelopes of bilinear terms x_4x_5 and x_7x_6, the following factorable overestimator for ϕ is obtained:

$$\tilde{\phi}^S = \min\left\{\begin{array}{c} 0.92x_1^{0.5} + 5.90x_2^{0.6} - 2.25, \\ 5.53x_1^{0.5}, \\ 0.41x_1^{0.5} + 2.64x_2^{0.6} + 5.63x_3^{0.8} - 7.08, \\ 2.48x_1^{0.5} + 5.63x_3^{0.8} - 6.07 \end{array}\right\} \tag{2.13}$$

According to Proposition 2.2, the function ϕ is concave-transformable, with $G^*(t) = t^{1/1.9}$. Then, from (2.9), we obtain the following overestimator for ϕ:

$$\tilde{\phi}^G(x) = 3.44\big(x_1^{0.5}x_2^{0.6}x_3^{0.8}\big)^{1/1.9} \tag{2.14}$$

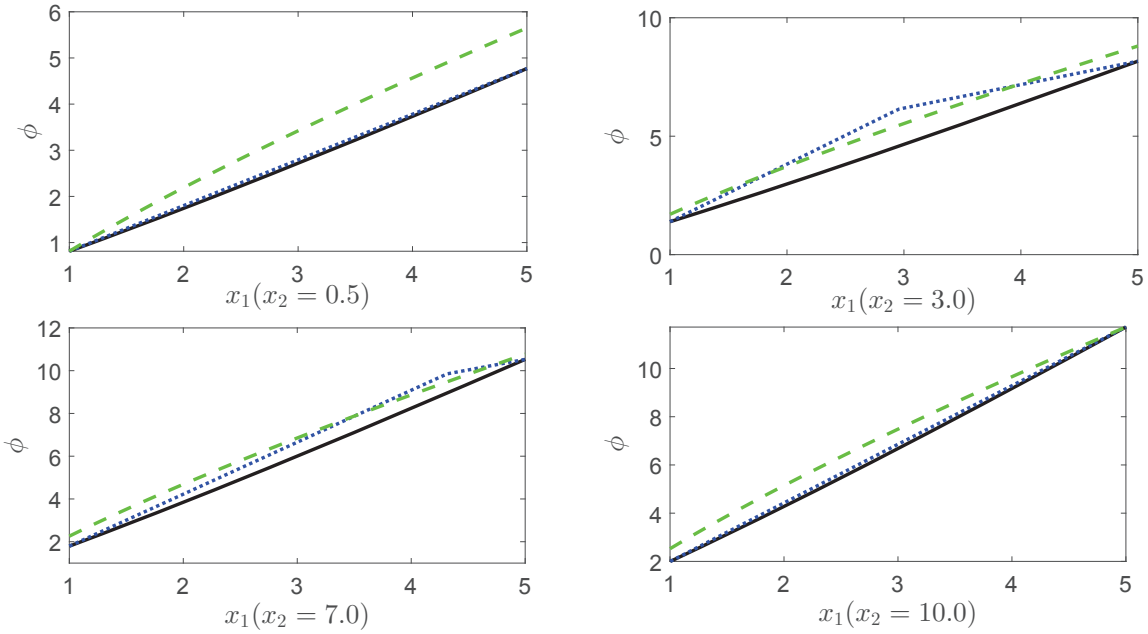


Figure 2.2: Comparison of overestimators for $\phi(x) = x_1^{1.1}x_2^{0.3}$ over $[1, 5] \times [0.5, 10]$. We plot the function ϕ in solid black, the factorable overestimator $\tilde{\phi}^S$ in dotted blue, and the transformation overestimator and $\tilde{\phi}^{G^*}$ in dashed green.

To assess the tightness of these two overestimators, we calculate the total relaxation gap associated with each approach as

$$\delta_{\text{tot}}^M = \int_{\mathcal{C}} \left(\tilde{\phi}^M(x) - \phi(x) \right) dx \tag{2.15}$$

where M indicates the method that is used to overestimate ϕ. By the previous relation we have that $\delta_{\text{tot}}^S = 63.9$, and $\delta_{\text{tot}}^G = 120.6$, indicating that for this example the transformation scheme given by (2.9) introduces a significantly larger relaxation gap than the factorable method.

Next, we overestimate ϕ via the RT scheme. We start by overestimating the relation $x_7 = x_1^{0.5}x_2^{0.6}$ using the transformation approach. Clearly, the signomial term $x_1^{0.5}x_2^{0.6}$ satisfies part (i) of Proposition 2.2, with $G^*(t) = t^{1/1.1}$. Thus, from (2.9) we obtain

$$x_7 \leq 1.17\left(x_1^{0.5}x_2^{0.6}\right)^{1/1.1} \tag{2.16}$$

Now, by combining the previous relation with the concave envelope of the bilinear term x_7x_6, we obtain the following RT overestimator for ϕ

$$\tilde{\phi}^{RT} = \min \left\{ \begin{array}{c} 2.82\big(x_1^{0.5}x_2^{0.6}\big)^{1/1.1}, \\ 1.26\big(x_1^{0.5}x_2^{0.6}\big)^{1/1.1} + 5.63x_3^{0.8} - 6.07 \end{array} \right\} \tag{2.17}$$

From (2.15) we have that $\delta_{\text{tot}}^{RT} = 44.9$, which corresponds to a 30% reduction in the total relaxation gap introduced by the factorable overestimator. For additional details on the RT approach, we direct the reader to Section 4.2 in [45].

2.2.2 Products and ratios of convex and/or concave functions

In the following, we present relaxations for products and ratios of convex and/or concave functions. In Propositions 2.3 and 2.4 we provide overestimators for concave-transformable products and ratios, whereas in Proposition 2.5 we consider convex-transformable products. These relaxations were derived in [45] by combining the results of Propositions 2.1 and 2.2 with composition rules for convex-transformable functions.

Proposition 2.3. (Proposition 16 in [45]) *Consider $\phi = \prod_{i\in I} \phi_i^{\alpha_i}$ over a box, where $\alpha_i > 0$ for all $i \in I$ and $\sum_{i\in I} \alpha_i > 1$. Let ϕ_i be concave and nonnegative for all $i \in I$. Then, ϕ is G-concave with $G(t) = t^{1/\xi}$, where $\xi = \sum_{i\in I} \alpha_i$. Furthermore, a concave overestimator for ϕ is given by:*

$$\tilde{\phi}^G(x) = \left(\phi^{\frac{1}{\xi}} - \underline{\phi}^{\frac{1}{\xi}}\right)\left(\frac{\bar{\phi} - \underline{\phi}}{\bar{\phi}^{\frac{1}{\xi}} - \underline{\phi}^{\frac{1}{\xi}}}\right) + \underline{\phi} \tag{2.18}$$

where $\underline{\phi}$ and $\bar{\phi}$ denote a lower and an upper bound on ϕ, respectively.

Proposition 2.4. (Proposition 17 in [45]) *Consider $\phi = \prod_{i\in I} \phi_i^{\alpha_i}$ over a box, where $\alpha_j < 0$ for some $j \in I$ and $\sum_{i\in I\setminus\{j\}} \alpha_i < |\alpha_j|$. Let ϕ_i be positive and concave for $i \in I \setminus \{j\}$, and let ϕ_j be positive and convex. Then, ϕ is G-concave with $G(t) = -t^{1/\xi}$, $\xi = \sum_{i\in I} \alpha_i$, and its associated overestimator $\tilde{\phi}^G$ is given by* (2.18).

Proposition 2.5. (Proposition 18 in [45]) *Consider* $\phi = \prod_{i\in I} \phi_i^{\alpha_i}$ *over a box, where* $\alpha_i < 0$ *for all* $i \in I \setminus \{j\}$ *and* $\sum_{i\in I\setminus\{j\}} |\alpha_i| < \alpha_j < \sum_{i\in I\setminus\{j\}} |\alpha_i| + 1$. *Let* ϕ_i *be positive and concave for* $i \in I \setminus \{j\}$, *and let* ϕ_j *be nonnegative and convex. Then,* ϕ *is* G*-convex with* $G(t) = t^{1/\xi}$, $\xi = \sum_{i\in I} \alpha_i$, *and its associated underestimator* $\tilde{\phi}^G$ *is given by* (2.18).

2.2.3 Log-concave functions

Now, we turn out attention to a particular class of logarithmically concave functions. Recall, that a real-valued function ϕ is logarithmically concave or log-concave, if it is positive over its domain, and $\log \phi$ is concave [20]. For example, consider the function $\phi(x) = (x_1 + x_2)^{0.5}(2x_1 + 5x_2)^{0.7}$, defined over $\mathcal{C} = [1, 5] \times [0.5, 2]$. Obviously, $\phi(x) > 0$, and $\log \phi(x)$ is concave for all $x \in \mathcal{C}$, which implies that ϕ is log-concave. Moreover, by Proposition 2.3, $\phi(x)$ is also G-concave and can be overestimated using (2.18). As another example, consider the function $\phi(x) = x_1^2 \exp(x_2)$, defined over $\mathcal{C} = [0.5, 10] \times [-5, 5]$. In this case, it is clear that $\phi(x)$ is also log-concave, since $\phi(x)$ is positive and $\log \phi(x)$ is concave for all $x \in \mathcal{C}$. However, this function does not satisfy any of the propositions discussed in the previous subsections, and cannot be overestimated by the transformation techniques outlined above. The following proposition, which is a minor modification of a result derived in [45], extends the transformation method discussed in the previous subsections to the class of log-concave functions considered in this example.

Proposition 2.6. (Extension of Proposition 19 in [45]) *Consider the function*

$$\phi(x) = \frac{f(x)^a \exp(g_0(x))}{k_0 + k_1 \sum_{i\in I} \exp(g_i(x))}, \alpha, k_0, k_1 > 0 \tag{2.19}$$

over a convex set $\mathcal{C} \subset \mathbb{R}^n$. *Let* $f(x)$ *be concave and positive,* $g_0(x)$ *be concave, and* $g_i(x)$, $i \in I$ *be convex over* $\mathcal{C}$. *Then,* ϕ *is log-concave. Further, a concave overestimator of* ϕ *over* $\mathcal{C}$ *is given by:*

$$\tilde{\phi}^G(x) = \frac{\left(\log \phi - \log \underline{\phi}\right)\left(\bar{\phi} - \underline{\phi}\right)}{\log\left(\bar{\phi}/\underline{\phi}\right)} + \underline{\phi} \tag{2.20}$$

where $\underline{\phi}$ and $\bar{\phi}$ are a lower and an upper bound on ϕ, respectively.

For examples illustrating the benefits of the relaxations presented in Propositions 2.3–2.6, we refer the reader to [45].

2.3 Implementation in a branch-and-bound algorithm

For some functional forms, the transformation outer-estimators discussed in the previous section have a more complex structure than widely used factorable outer-estimators. Direct incorporation of these complex relaxations may reduce performance of global solvers that solve nonlinear convex relaxations to obtain lower bounds. However, as we detail in this section, this is not an issue for global solvers that construct polyhedral relaxations in the lower bounding step. In this case, it suffices to generate supporting hyperplanes for the transformation outer-estimators, which can then be used as valid inequalities to tighten an existing polyhedral relaxation.

In order to examine the impact of the G-convexity relaxations presented in §2.2 on the performance of branch-and-bound algorithms, we have incorporated these relaxations into the global solver BARON. Our implementation consists of recognition algorithms for identifying several classes of convex-transformable functions, and a cutting plane generation strategy, which constructs cuts for convex-transformable functions at every node of the branch-and-bound tree. The following subsections provide a detailed description of our implementation.

2.3.1 Identification of convex-transformable functions in general nonconvex problems

In this section, we introduce a set of recognition routines for the identification of convex-transformable functions in general nonconvex problems. We start by listing the various

classes of convex transformable functions which are recognized by our implementation. This list has been compiled after carrying out an extensive survey of nonconvex problems appearing in a wide variety of scientific and engineering applications.

1. Signomial functions of the form

$$\phi(x) = \prod_{i \in I} x_i^{\alpha_i}, \alpha_i \in \mathbb{R} \setminus \{0\}, \ \ \forall i \in I = \{1, \ldots, n\} \tag{2.21}$$

over a subset of the nonnegative orthant. In particular, we consider signomials satisfying Proposition 2.1, and part (i) of Proposition 2.2. Concave-transformable signomials satisfying part (ii) of Proposition 2.2 are ignored here, since the factorable overestimator constructed by BARON in this case globally dominates the transformation overestimator given by (2.9) (see Proposition 12 in [45] for details).

2. Products and ratios of the form

$$\phi(x) = (f(x))^{\alpha}(g(x))^{\beta}, x \in \mathcal{H}, \alpha, \beta \in \mathbb{R} \tag{2.22}$$

where $\mathcal{H} \subseteq \mathbb{R}^n$ denotes a box and f and g are convex and/or concave functions defined over $\mathcal{H}$. In particular, we consider the following cases:

   (a) $f(x) = 1, \alpha = 0, g(x) = a_0 + \sum_{i=1}^n a_i x_i^2, \beta = -p$, where $a_i > 0, \forall i \in \{0, 1, \ldots, n\}$, and $p > 0$.

   (b) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g(x) = b_0 + \sum_{i=1}^n b_i x_i$.

   (c) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g(x) = b_0 + b_1 \log\left(\sum_{i=1}^n c_i x_i\right)$, where $b_1 > 0$.

(d) $f(x) = a_0 + a_1 \log\left(\sum_{i=1}^n c_i x_i\right)$, $g(x) = b_0 + b_1 \log\left(\sum_{i=1}^n d_i x_i\right)$, where $a_1, b_1 > 0$.

(e) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g(x) = b_0 + \sum_{i=1}^n \left(b_i/x_i + c_i x_i + d_i x_i^2 + e_i x_i^3 + f_i x_i^4\right)$, where g is a concave function.

(f) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g(x) = b_0 + \sum_{i=1}^n b_i x_i^p$, where $b_i > 0, \forall i \in \{1, \ldots, n\}$, and $0 < p < 1$.

(g) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g(x) = b_0 + b_1 \exp\left(\sum_{i=1}^n c_i x_i\right)$, where $b_1 < 0$.

Note that G-convexity or G-concavity of the functions listed above is determined by the values of the exponents α and β and the convexity and/or concavity properties of the functions f and g.

3. Log-concave functions of the form

$$\phi(x) = \frac{f(x)^\alpha \exp(g_0(x))}{k_0 + k_1 \sum_{i \in I} \exp g_i(x)}, x \in \mathcal{H}, \alpha, k_0, k_1 > 0, \tag{2.23}$$

where $\mathcal{H} \subseteq \mathbb{R}^n$ denotes a box, $f(x)$ is concave and positive, $g_0(x)$ is concave, and $g_i(x), i \in I$ is convex over $\mathcal{H}$. In particular, we consider the following cases:

(a) $f(x) = 1$, $g_0(x) = 0$, $k_0, k_1 > 0$, $I = \{1\}$, $g_1(x) = a_0 + \sum_{i=1}^n a_i x_i$.

(b) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g_0(x) = b_0 + \sum_{i=1}^n b_i x_i$, $k_0 = k_1 = 1$, $I = \emptyset$.

(c) $f(x) = \log\left(a_0 + \sum_{i=1}^n a_i x_i\right)$, $g_0(x) = b_0 + \sum_{i=1}^n b_i x_i$, $k_0 = k_1 = 1$, $I = \emptyset$.

(d) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g_0(x) = b_0 + b_1/(c_0 + \sum_{i=1}^n c_i x_i)^p$, $k_0 = k_1 = 1$, $I = \emptyset$, where $p > 0$, and $b_1 < 0$.

(e) $f(x) = a_0 + \sum_{i=1}^n a_i x_i$, $g_0(x) = b_0 + \sum_{i=1}^n b_i x_i$, $k_0 = k_1 = 1$, $I = \{1, 2\}$, $g_1(x) = c_0 + \sum_{i=1}^n c_i x_i$, $g_2(x) = d_0 + \sum_{i=1}^n d_i x_i$.

Remark 1. In our implementation, we ignore G-convex and G-concave functions that are convex or concave, since BARON is equipped with a powerful module that exploits convexity or concavity properties of such functions for relaxation construction [48]. For example, consider the function $\phi(x_1) = x_1 \log(x_1)$, $x_1 > 0$. Clearly, this function belongs to Class 2(c) above, with $n = 1$, $a_0 = b_0 = 0$, $a_1 = b_1 = c_1 = 1$, and $\alpha = \beta = 1$. It is simple to check that ϕ satisfies the conditions given by Proposition 2.3, and thus is G-concave. However, ϕ is also convex, and it is, as a result, ignored by our implementation. As another example, consider the function $\phi(x_1) = (a_0 + a_1 x_1)(b_0 + b_1 x_1)$, where $(a_0 + a_1 x_1) \geq 0$ and $(b_0 + b_1 x_1) \geq 0$. This function belongs to Class 2(b) above, with $n = 1$, and $\alpha = \beta = 1$. It is easy to verify that ϕ also satisfies the conditions given by Proposition 2.3, and thus is G-concave. However, ϕ is convex when $a_1 b_1 \geq 0$, and concave when $a_1 b_1 \leq 0$. Therefore, we also ignore this function in our implementation.

Remark 2. We exclude from the above list G-convex (resp. G-concave) functions for which the convex (resp. concave) envelopes are available and implemented in BARON. For example, consider the function $\phi(x) = x_1^{1.5} x_2^{2.4} x_3^{1.1}$, $x_1, x_2, x_3 \geq 0$, which belongs to Class 1 above. It is simple to check that ϕ satisfies part (i) of Proposition 2.2, and thus is G-concave. However, ϕ is also component-wise convex, and as shown in [88], the factorable

relaxation method provides the concave envelope for component-wise convex signomials defined over a box in the nonnegative orthant. We ignore this function in our implementation since its envelope is already implemented in BARON. As another example, consider the function $\phi(x) = x_1^\alpha \exp x_2$, $x_1 > 0$, $x_2 \in \mathbb{R}$, $\alpha \in (0,1)$. Clearly, this function belongs to Class 3(b) above, with $n = 2$, $a_0 = b_0 = a_2 = b_1 = 0$, and $a_1 = b_2 = 1$. Obviously, ϕ satisfies Proposition 2.6, and thus is G-concave. However, ϕ has a nonpolyhedral concave envelope (see Theorem 2 in [47] for details), which is implemented BARON. As a result, this function is also ignored by our implementation.

Next, we briefly review the factorable reformulation algorithm implemented in BARON, which relies on the introduction of intermediate variables and constraints in order to decompose each factorable expression appearing in the original problem (see Algorithm 1 for details). As an example of this reformulation, consider the following factorable function over the positive orthant:

$$f(x) = (1 + x_1/x_2)^{-0.7}(x_1 + x_2x_3)^{0.8} + x_2 + x_3 \tag{2.24}$$

This function does not contain any convex-transformable subexpressions. However, by applying Algorithm 1, the intermediate variables $x_4 = x_1/x_2$ and $x_5 = x_2x_3$ are introduced to obtain a factorable reformulation of f. Now, if we consider the function f in the augmented relaxation space $\mathbb{R}^5$, it is clear that the subexpression $(1 + x_4)^{-0.7}(x_1 + x_5)^{0.8}$ corresponds to a function of Class 2(b), which is G-convex by Proposition 2.5.

Now, we turn our attention to our recognition routines, which are presented in Algorithms 2–4. Given a factorable reformulation of an optimization problem, in Algorithm 2 we scan each bilinear term of the form $x_k = x_ix_j$, and subsequently proceed by reconstructing intermediate expressions in order to identify different types of convex-transformable functions. By employing Algorithm 2, we are able to recognize all classes of convex transformable functions listed at the beginning of this section, with the exception

of signomial functions involving three or more variables, and convex-transformable functions that are reciprocals of other nonlinear functions. To identify such functions, we use Algorithms 3 and 4.

Algorithm 3 relies on a set of subroutines implemented in BARON for the identification of multilinear functions (see Section 3.1 in [9] for details). Given a list of functions $\mathcal{ML} = \{L_k(x) = \sum_{i \in I_k} c_i \prod_{j \in T_{ki}} x_j, k \in K\}$ identified by BARON's multilinear module, Algorithm 3 starts by decomposing each function $L_k(x)$ into multilinear terms of the form $\prod_{j \in T_{ki}} x_j$. Subsequently, each multilinear term containing three or more variables is analyzed in order to determine if it corresponds to a signomial function of Class 1. In Algorithm 4, we start by scanning all the monomial expressions of the form $x_i = x_j^p$. For each monomial relation with $p = -1$, we proceed by reconstructing intermediate expressions in order to determine if they correspond to functions of Classes 2(a) or 3(a). For example, consider the following function over the positive orthant:

$$f(x) = x_1^{0.6} x_2^{0.7} x_3^{0.8} + (1 + x_1)^{-1.1} \left(x_2 + \frac{x_3}{x_1} \right)^{1.5} + \frac{1}{\sqrt{1 + x_1^2 + 2x_2^2}} \tag{2.25}$$

During the execution of Algorithm 1, the following intermediate variables are introduced to obtain a factorable reformulation of f:

$$\begin{array}{llll} x_4 = x_1^{0.6} & x_8 = x_6 x_7 & x_{12} = x_2 + x_{11} & x_{16} = x_2^2 \\ x_5 = x_2^{0.7} & x_9 = 1 + x_1 & x_{13} = x_{12}^{1.5} & x_{17} = 1 + x_{15} + 2x_{16} \\ x_6 = x_4 x_5 & x_{10} = x_9^{-1.1} & x_{14} = x_{10} x_{13} & x_{18} = x_{17}^{0.5} \\ x_7 = x_3^{0.8} & x_{11} = x_3 / x_1 & x_{15} = x_1^2 & x_{19} = x_{18}^{-1} \end{array} \tag{2.26}$$

Our recognition routines identify the following functions:

$$\begin{array}{ll} \phi_1 = x_1^{0.6} x_2^{0.7} & \phi_4 = (1 + x_1)^{-1.1} (x_2 + x_{11})^{1.5} \\ \phi_2 = x_6 x_3^{0.8} & \phi_5 = 1 / \sqrt{1 + x_1^2 + 2x_2^2} \\ \phi_3 = x_1^{0.6} x_2^{0.7} x_3^{0.8} & \end{array} \tag{2.27}$$

where ϕ_1, ϕ_2, ϕ_3 are G-concave by Proposition 2.2, ϕ_4 is G-convex by Proposition 2.5, and ϕ_5 is G-concave by Proposition 2.4. The functions ϕ_1, ϕ_2, and ϕ_4 are recognized by Algorithm 2 by checking the bilinear terms $x_4 x_5$, $x_6 x_7$, and $x_{10} x_{13}$. The signomial function ϕ_3

Algorithm 1 Standard factorable reformulation in BARON

Given a collection of nonlinear factorable functions $\mathcal{F} = \{f_i(x), i \in Q\}$.
Initialize the number of intermediate variables $j = 0$, the list of intermediate relations $\mathcal{I} = \emptyset$, and the lists of indices of intermediate monomial, power, logarithmic, linear and bilinear variables, $\mathcal{M} = \emptyset$, $\mathcal{P} = \emptyset$, $\mathcal{G} = \emptyset$, $\mathcal{L} = \emptyset$, $\mathcal{B} = \emptyset$, respectively.
For each function $f_i(x) \in \mathcal{F}$:
    If $f_i(x)$ is a univariate function (i.e., monomial, power, or logarithm) **then**
        update $j \leftarrow j + 1$ and add the univariate relation $y_j = f_i(x)$ to $\mathcal{I}$
        If $f_i(x)$ is a univariate monomial function **then**
            add j to $\mathcal{M}$
        Else If $f_i(x)$ is a univariate power function **then**
            add j to $\mathcal{P}$
        Else If $f_i(x)$ is a univariate logarithmic function **then**
            add j to $\mathcal{G}$
        End If
    Else If $f_i(x) = g(x)/h(x)$ **then**
        update $j \leftarrow j + 3$ and introduce the variables y_{j-2}, y_{j-1}, and y_j
        let $y_{j-2} = g(x)$ and $y_{j-1} = h(x)$; add $h(x)$ and $g(x)$ to $\mathcal{F}$
        add the bilinear relation $y_{j-2} = y_{j-1}y_j$ to $\mathcal{I}$
        add j to $\mathcal{B}$
    Else If $f_i(x) = \prod_{k=1}^{l} g_k(x)$ **then**
        For $k = 1$ to l
            update $j \leftarrow j + 1$, let $y_j = g_k(x)$, and add $g_k(x)$ to $\mathcal{F}$
        End For
        update $j \leftarrow j + 1$ and add the bilinear relation $y_j = y_{j-l}y_{j-l+1}$ to $\mathcal{I}$
        add j to $\mathcal{B}$
        For $k = 3$ to l
            update $j \leftarrow j + 1$ and add the bilinear relation $y_j = y_{j-1}y_{j-l+1}$ to $\mathcal{I}$
            add j to $\mathcal{B}$
        End For
    Else If $f_i(x) = \sum_{k=1}^{l} a_k g_k(x)$ **then**
        For $k = 1$ to l
            update $j \leftarrow j + 1$, let $y_j = g_k(x)$, and add $g_k(x)$ to $\mathcal{F}$
        End For
        update $j \leftarrow j + 1$ and add the linear relation $y_j = \sum_{k=1}^{l} a_k y_{j-k}$ to $\mathcal{I}$
        add j to $\mathcal{L}$
    Else If $f_i(x) = h(g(x))$ **then**
        update $j \leftarrow j + 2$ and introduce the variables y_{j-1}, and y_j
        let $y_{j-1} = g(x)$ and $y_j = h(y_{j-1})$; add $g(x)$ and $h(y_{j-1})$ to $\mathcal{F}$
    End If
End For

Algorithm 2 Identification of convex-transformable products and ratios in general nonconvex problems

Given a factorable reformulation of an optimization problem obtained by applying Algorithm 1, a list of intermediate relations $\mathcal{I}$, and the lists of indices of original $\mathcal{O}$, monomial $\mathcal{M}$, power $\mathcal{P}$, logarithmic $\mathcal{G}$, linear $\mathcal{L}$ and bilinear $\mathcal{B}$ variables.
Initialize the list of convex-transformable functions $\mathcal{J} = \emptyset$.
For each bilinear expression $x_k = x_i x_j \in \mathcal{I}$:
    If $k > \max\{i, j\}$ **then**
        If x_i corresponds to a monomial relation $x_i = x_{l_1}^{p_1} \in \mathcal{I}$ **then**
            set $\alpha_1 \leftarrow p_1$ and $i_1 \leftarrow l_1$
        Else
            set $\alpha_1 \leftarrow 1$ and $i_1 \leftarrow i$
        End If
        If x_j corresponds to a monomial relation $x_j = x_{l_2}^{p_2} \in \mathcal{I}$ **then**
            set $\alpha_2 \leftarrow p_2$ and $i_2 \leftarrow l_2$
        Else
            set $\alpha_2 \leftarrow 1$ and $i_2 \leftarrow j$
        End If
        Let $\phi_k(x) = f(x)^{\alpha_1} g(x)^{\alpha_2}$, where $\{x_{i_1} = f(x), x_{i_2} = g(x)\} \in \mathcal{I}$
        If ($i_1 \in \mathcal{L}$ **and** $i_2 \in \mathcal{O}$) **or** ($i_1 \in \mathcal{O}$ **and** $i_2 \in \mathcal{L}$) **or** ($i_1 \in \mathcal{L}$ **and** $i_2 \in \mathcal{L}$) **then**
            If $\phi_k(x)$ corresponds to any of the functions of Classes 2(b)–(g) **then**
                add $\phi_k(x)$ to $\mathcal{J}$; **cycle**
            End If
        Else If ($i_1 \in \mathcal{G}$ **and** $i_2 \in \mathcal{O}$) **or** ($i_1 \in \mathcal{O}$ **and** $i_2 \in \mathcal{G}$) **or** ($i_1 \in \mathcal{G}$ **and** $i_2 \in \mathcal{L}$) **or** ($i_1 \in \mathcal{L}$ **and** $i_2 \in \mathcal{G}$) **or** ($i_1 \in \mathcal{G}$ **and** $i_2 \in \mathcal{G}$) **then**
            If $\phi_k(x)$ corresponds to any of the functions of Classes 2(c)–(d) **then**
                add $\phi_k(x)$ to $\mathcal{J}$; **cycle**
            End If
        Else If ($i_1 \in \mathcal{P}$ **and** $i_2 \in \mathcal{O}$) **or** ($i_1 \in \mathcal{O}$ **and** $i_2 \in \mathcal{P}$) **or** ($i_1 \in \mathcal{P}$ **and** $i_2 \in \mathcal{L}$) **or** ($i_1 \in \mathcal{L}$ **and** $i_2 \in \mathcal{P}$) **then**
            If $\phi_k(x)$ corresponds to any of the functions of Classes 3(b), (d)–(e) **then**
                add $\phi_k(x)$ to $\mathcal{J}$; **cycle**
            End If
        Else If ($i_1 \in \mathcal{G}$ **and** $i_2 \in \mathcal{P}$) **or** ($i_1 \in \mathcal{P}$ **and** $i_2 \in \mathcal{G}$) **then**
            If $\phi_k(x)$ corresponds to a function of Class 3(c) **then**
                add $\phi_k(x)$ to $\mathcal{J}$; **cycle**
            End If
        End If
        Let $\phi_k(x_{i_1}, x_{i_2}) = x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2}$
        If $\phi_k(x_{i_1}, x_{i_2})$, corresponds to a signomial function of Class 1 **then**
            add $\phi_k(x_{i_1}, x_{i_2})$ to $\mathcal{J}$
        End If

Algorithm 2 Identification of convex-transformable products and ratios in general nonconvex problems (cont.)

Else If $i > \max\{k, j\}$ **then**
  If x_k corresponds to a monomial relation $x_k = x_{l_1}^{p_1} \in \mathcal{I}$ **then**
    set $\alpha_1 \leftarrow p_1$ and $i_1 \leftarrow l_1$
  Else
    set $\alpha_1 \leftarrow 1$ and $i_1 \leftarrow k$
  End If
  If x_j corresponds to a monomial relation $x_j = x_{l_2}^{p_2} \in \mathcal{I}$ **then**
    set $\alpha_2 \leftarrow -p_2$ and $i_1 \leftarrow l_2$
  Else
    set $\alpha_2 \leftarrow -1$ and $i_1 \leftarrow j$
  End If
  Let $\phi_k(x) = f(x)^{\alpha_1} g(x)^{\alpha_2}$, where $\{x_{i_1} = f(x), x_{i_2} = g(x)\} \in \mathcal{I}$
  If ($i_1 \in \mathcal{L}$ **and** $i_2 \in \mathcal{O}$) **or** ($i_1 \in \mathcal{O}$ **and** $i_2 \in \mathcal{L}$) **or** ($i_1 \in \mathcal{L}$ **and** $i_2 \in \mathcal{L}$) **then**
    If $\phi_k(x)$ corresponds to a function of Class 2(b) **then**
      add $\phi_k(x)$ to $\mathcal{J}$; **cycle**
    End If
  End If
  Let $\phi_k(x_{i_1}, x_{i_2}) = x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2}$
  If $\phi_k(x_{i_1}, x_{i_2}) = x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2}$, corresponds to a signomial function of Class 1 **then**
    add $\phi_k(x_{i_1}, x_{i_2})$ to $\mathcal{J}$
  End If
End If
End For

is identified by Algorithm 3 from the analysis of the multilinear term $x_4x_5x_7$, while the function ϕ_5 is recognized by Algorithm 4, as it appears in a monomial relation.

2.3.2 Cut generation

The recognition routines described in the previous section are executed before the start of the branch-and-bound search. Once all convex-transformable expressions of interest have been identified, we employ a cut generation algorithm in order to construct supporting hyperplanes to the transformation outer-estimators of all G-convex and G-concave intermediates. These cuts are generated at each node of the branch-and-bound tree, and utilized to

Algorithm 3 Identification of convex-transformable signomials involving three or more variables

Given a factorable reformulation of an optimization problem obtained by applying Algorithm 1, a set of multilinear functions $\mathcal{ML} = \{L_k(x) = \sum_{i \in I_k} c_i \prod_{j \in T_{ki}} x_j, k \in K\}$ identified by BARON's multilinear module, a list of intermediate expressions $\mathcal{I}$, and a list for storing convex-transformable functions $\mathcal{J}$.
Initialize auxiliary arrays α and d
For each function $L_k(x) \in \mathcal{ML}$:
    For each $i \in I_k$:
        If $|T_{ki}| \geq 3$ **then**
            Initialize auxiliary variable $m = 0$
            For each $j \in T_{ki}$:
                update $m \leftarrow m + 1$
                If x_j corresponds to a monomial relation $x_j = x_l^p \in \mathcal{I}$ **then**
                    set $\alpha(m) \leftarrow p$ and $d(m) \leftarrow l$
                Else
                    set $\alpha(m) \leftarrow 1$ and $d(m) \leftarrow j$
                End If
            End For
            Construct the function $\phi_k(x) = \prod_{p=1}^{m} x_{d(p)}^{\alpha(p)}$
            If $\phi_k(x)$ corresponds to a signomial function of Class 1 **then**
                add $\phi_k(x)$ to $\mathcal{J}$
            End If
        End If
    End For
End For

tighten the polyhedral relaxations constructed by BARON. Note that by executing our cutting plane generation algorithm at each node of the branch-and-bound tree, we can fully exploit BARON's bound tightening capabilities, and use tight bounds for relaxation construction, which clearly has a significant impact on the quality of the resulting relaxations. Another advantage of integrating our cutting plane generation scheme at each node of the branch-and-bound tree, is the fact that our cutting planes can be used for feasibility- and optimality-based bound range reduction (see [76, 93] for details).

Before providing a detailed description of our cutting plane generation strategy, we briefly review BARON's polyhedral relaxation constructor. At a given node in the branch-

Algorithm 4 Identification of convex-transformable functions that are reciprocals of other nonlinear functions

Given a factorable reformulation of an optimization problem obtained by applying Algorithm 1, a list of intermediate relations $\mathcal{I}$, the lists of indices of monomial $\mathcal{M}$ and linear $\mathcal{L}$ variables, and a list for storing convex-transformable functions $\mathcal{J}$.
For each monomial expression of the form $x_i = x_j^{p_1} \in \mathcal{I}$
    If $p_1 = -1$ **then**
        If $j \in \mathcal{L}$ **then**
            Construct the function $\phi_i(x) = f(x)^{p_1}$, where $\{x_j = f(x) \in \mathcal{I}\}$
            If $\phi_i(x)$ corresponds to a function of Classes 2(a) or 3(a) **then**
                add $\phi_i(x)$ to $\mathcal{J}$
            End If
        Else If $j \in \mathcal{M}$ **then**
            If x_j corresponds to a monomial relation $x_j = x_k^{p_2} \in \mathcal{I}$ **then**
                Construct the function $\phi_i(x) = g(x)^{p_1 p_2}$, where $\{x_k = g(x) \in \mathcal{I}\}$
                If $\phi_i(x)$ corresponds to a function of Class 2(a) **then**
                    add $\phi_i(x)$ to $\mathcal{J}$
                End If
            End If
        End If
    End If
End For

and-bound tree, BARON first constructs an initial linear-programming based relaxation by outer-approximating all convex functions appearing in the factorable decomposition of the original problem with subgradient inequalities (see [93] for details). This relaxation is then solved, and subsequently refined, by adding various classes of cutting planes in an iterative fashion. These cutting planes are added to the current relaxation only if they are violated by the previous relaxation solution.

Our cutting plane generation scheme is integrated within BARON's polyhedral relaxation constructor. Note that the cutting planes generated by our algorithm are not included in the initial polyhedral outer-approximation constructed at a given node, and are only used in the subsequent rounds of cut generation. At a given round of cut generation, we scan all convex-transformable expressions identified during recognition. Suppose that each of these expressions is stored in list J, and has the form $y_j = \phi_j(x)$, $j \in J$, where y_j

is an intermediate variable introduced during the execution of Algorithm 1, and $\phi_j(x)$ is a G-convex or G-concave function appearing in the original problem, or in its factorable reformulation. Moreover, let (x^*, y_j^*) be the projection of the current relaxation solution onto the (x, y_j) space. If ϕ_j is G-convex (resp. G-concave), then, we construct a convex underestimator (resp. concave overestimator) $\tilde{\phi}_j^G$ employing the transformation scheme outlined in §2.2. Next, we generate a cutting plane corresponding to the supporting hyperplane of $\tilde{\phi}_j^G$ at the relaxation solution. If this cutting plane violates the relaxation solution, then, we compare the transformation relaxation $\tilde{\phi}_j^G$ with a standard factorable relaxation $\tilde{\phi}_j^S$ of ϕ_j. If $\tilde{\phi}_j^G$ is tighter than $\tilde{\phi}_j^S$ at $x = x^*$, then, we calculate the Euclidean distance d between the generated cutting plane and the relaxation solution. If d is greater than a predefined threshold, then, the generated cutting plane is added to the current relaxation. The entire cutting plane generation strategy is described in Algorithm 5.

Remark 3. During cut generation, we employ the RT scheme discussed in §2.2.1 to construct overestimators for signomials involving three or more variables and satisfying part (i) of Proposition 2.2. For all other signomial functions of Class 1, we use relation (2.9) to construct the corresponding transformation outer-estimators.

Remark 4. For functions $\phi(x) = (f(x))^{\alpha}(g(x))^{\beta}$ of Class 2(e), we verify concavity of $g(x) = b_0 + \sum_{i=1}^{n} \left(b_i/x_i + c_i x_i + d_i x_i^2 + e_i x_i^3 + f_i x_i^4\right)$, $x \in \mathcal{H}$, by bounding the eigenvalues of its Hessian $\nabla^2 g(x)$ over $\mathcal{H}$. Note that $\nabla^2 g(x)$ is a diagonal matrix with diagonal elements $\lambda_i(x_i) = 2b_i/x_i^3 + 2d_i + 6e_i x_i + 12 f_i x_i^2$, $i \in I = \{1, \ldots, n\}$. We perform an initial concavity assessment during the execution of the recognition subroutines. Assume that $\mathcal{H} = [\underline{x}_1, \bar{x}_1] \times \cdots \times [\underline{x}_n, \bar{x}_n]$. We calculate $\lambda_i^{\max} = \max_{x_i \in [\underline{x}_i, \bar{x}_i]}\{\lambda_i(x_i)\}$ and $\lambda_i^{\min} = \min_{x_i \in [\underline{x}_i, \bar{x}_i]}\{\lambda_i(x_i)\}$, for all $i \in I$. If $\lambda_i^{\min} > 0$, for some $i \in I$, then, g is nonconcave over $\mathcal{H}$, and we do not include ϕ in the list of convex-transformable functions. On the other hand, if $\lambda_i^{\max} \leq 0$, for all $i \in I$, then, we mark g as concave, and do not check its concavity in subsequent

nodes of the branch-and-bound tree. If we cannot prove or disprove concavity of g during recognition, then, g is marked for later check in the branch-and-bound tree, as its concavity properties may change due to range reduction or branching operations. If g becomes concave at a given node of the branch-and-bound tree, then, we generate cutting planes for ϕ according to Algorithm 5.

Remark 5. For univariate functions of Class 3(a), and Class 3(b) with $\alpha = 1$, we use the corresponding convex and concave envelopes for cut generation. Recall that a univariate function $\phi(x)$ defined over an interval $[\underline{x}, \bar{x}]$ is said to be convexoconcave (concavoconvex), if for some $\hat{x} \in [\underline{x}, \bar{x}]$, $\phi(x)$ is convex (concave) over $[\underline{x}, \hat{x}]$, and concave (convex) over $[\hat{x}, \bar{x}]$ [91]. First, consider the univariate function of Class 3(a) $\phi(x_1) = 1/\left(k_0 + k_1 \exp(a_0 + a_1 x_1)\right)$, $k_0, k_1 > 0$, $x_1 \in [\underline{x}_1, \bar{x}_1]$. It is easy to verify that ϕ is convexoconcave when $a_1 < 0$, and concavoconvex when $a_1 > 0$. Now, consider the univariate function of Class 3(b) $\phi(x_1) = (a_0 + a_1 x_1) \exp(b_0 + b_1 x_1)$, $x_1 \in [\underline{x}_1, \bar{x}_1]$. In this case, it is also simple to check that ϕ is convexoconcave when $a_1 < 0$, and concavoconvex when $a_1 > 0$. Denote by $\mathrm{conv}_{\mathcal{C}}\phi$ the convex envelope of ϕ over $\mathcal{C} = [\underline{x}_1, \bar{x}_1]$. It is simple to show that the convex envelope of ϕ is given by

$$\mathrm{conv}_{\mathcal{C}}\phi(x_1) = \begin{cases} \phi(x_1), & \text{if } \underline{x}_1 \leq x_1 \leq w_1, \\ \phi(\bar{x}_1) + \phi'(w_1)(x - \bar{x}_1), & \text{if } w_1 \leq x_1 \leq \bar{x}_1. \end{cases} \tag{2.28}$$

when ϕ is convexoconcave, and by

$$\mathrm{conv}_{\mathcal{C}}\phi(x_1) = \begin{cases} \phi(\underline{x}_1) + \phi'(w_2)(x - \underline{x}_1), & \text{if } \underline{x}_1 \leq x_1 \leq w_2, \\ \phi(x_1), & \text{if } w_2 \leq x_1 \leq \bar{x}_1. \end{cases} \tag{2.29}$$

when ϕ is concavoconvex, where w_1 and w_2 are points satisfying $\phi(w_1) = \phi(\bar{x}_1) + \phi'(w_1)(w_1 - \bar{x}_1)$ and $\phi(w_2) = \phi(\underline{x}_1) + \phi'(w_2)(w_2 - \underline{x}_1)$, respectively (see Figure 2.3). Analogous expressions can be obtained for the concave envelopes of ϕ in a similar manner.

Algorithm 5 Cutting plane generation strategy for convex-transformable functions at each node of the branch-and-bound tree

Given the relaxation solution, n_J convex-transformable functions stored in list $\mathcal{J}$, and a parameter $\theta > 0$.
For each function $y_j = \phi_j(x)$ in $\mathcal{J}$
    Let (x^*, y_j^*) be the projection of the current relaxation solution onto the (x, y_j) space
    If ϕ_j is a G-convex **then**
        Construct a convex underestimator $\tilde{\phi}_j^G$ using the techniques of §2.2
        Generate a cutting plane of the form $y_j \geq \tilde{\phi}_j^G(x^*) + \nabla\tilde{\phi}_j^G(x^*)^T(x - x^*)$
        If $y_j^* < \tilde{\phi}_j^G(x^*)$ **then**
            Construct a convex underestimator $\tilde{\phi}_j^S$ using the factorable approach
            If $\tilde{\phi}_j^G(x^*) > \tilde{\phi}_j^S(x^*)$ **then**
                Let d be the Euclidean distance between the hyperplane $\tilde{\phi}_j^G(x^*) + \nabla\tilde{\phi}_j^G(x^*)^T(x - x^*) - y_j = 0$ and the point (x^*, y_j^*).
                If $d > \theta$ **then**
                    add the cut to the relaxation
                End If
            End If
        End If
    Else If ϕ_j is a G-concave **then**
        Construct a concave overestimator $\tilde{\phi}_j^G$ using the techniques of §2.2
        Generate a cutting plane of the form $y_j \leq \tilde{\phi}_j^G(x^*) + \nabla\tilde{\phi}_j^G(x^*)^T(x - x^*)$
        If $y_j^* > \tilde{\phi}_j^G(x^*)$ **then**
            Construct a concave overestimator $\tilde{\phi}_j^S$ using the factorable approach
            If $\tilde{\phi}_j^G(x^*) < \tilde{\phi}_j^S(x^*)$ **then**
                Let d be the Euclidean distance between the hyperplane $\tilde{\phi}_j^G(x^*) + \nabla\tilde{\phi}_j^G(x^*)^T(x - x^*) - y_j = 0$ and the point (x^*, y_j^*).
                If $d > \theta$ **then**
                    add the cut to the relaxation
                End If
            End If
        End If
    End If
End For

Remark 6. With the aim of avoiding poorly scaled relaxations, we check the cutting planes generated during the execution of Algorithm 5 to ensure that they are properly scaled. Poorly scaled cuts are not added to an existing relaxation. To perform this check, we follow

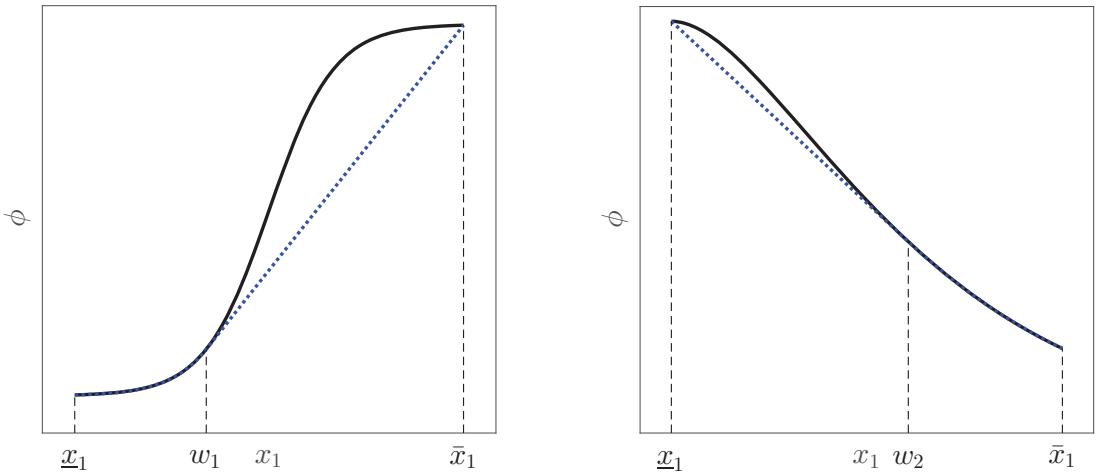


Figure 2.3: Convex envelopes for (a) $\phi(x_1) = 1/\left(k_0 + k_1 \exp(a_0 + a_1 x_1)\right)$, $k_0, k_1 > 0$, $a_1 < 0$, and (b) $\phi(x_1) = (a_0 + a_1 x_1)\exp(b_0 + b_1 x_1)$, $a_1 > 0$. The function ϕ is shown in solid black, and its convex envelope in dotted blue.

a strategy similar to that described in [48]. Namely, we examine each of the cut coefficients and check if their absolute values lie between sufficiently small and large constants. We also perform this check for the absolute values of the ratios between the different cut coefficients. For details on the techniques used within BARON to check the safety of a given cut, we refer the reader to Section 2.4 in [48].

2.4 Computational results

In this section, we present the results of an extensive computational study that we have conducted in order to investigate the impact of the proposed implementation on the performance of the branch-and-bound global solver BARON. For our numerical experiments, we consider a large number of nonconvex problems compiled from the `GlobalLib` [32], `PrinceLib` [72], `MINLPLib` [26], `AIMMSLib` [40], and `NRCLib` [8] collections.

In our experiments, all problems are solved in minimization form, with relative/abso-

lute tolerances of 10^{-6} and a time limit of 500 seconds. We allow up to 4 rounds of cutting plane generation at a given node in the branch-and-bound tree, and set the deep cut measure θ to 10^{-3} (see Algorithm 5). For other algorithmic parameters, we employ the default settings of the GAMS/BARON distribution [77]. All experiments are performed under GAMS 24.6.1 on a 64-bit Intel Xeon X5650 2.66Ghz processor running CentOS release 7.

2.4.1 The test set

We consider a test set of 262 nonconvex optimization problems containing a variety of convex transformable functions that are recognized by our implementation. The main characteristics of the selected models are provided in Table 2.1, which includes the number of problems selected from each test library, along with information on the the minimum, maximum, and average number of constraints (m), variables (n), nonzero elements in the constraints and objective (nz), and nonlinear elements in the constraints and objective (nnz).

In Table 2.2, we provide statistics on the different classes of convex-transformable functions appearing in the test problems. For each collection, we indicate the number and in parentheses the percentage, of problems containing each of the three classes of functions described in §2.3.1. As observed from Table 2.2, functions of Class 2 appear with the highest frequency in the test problems.

2.4.2 Impact of the proposed cutting planes on the performance of BARON

We solve the test problems described in the previous subsection using BARON 17.2, with and without the cutting planes for convex-transformable functions (CTF cutting planes). We denote the former algorithm by BARONctf, and the latter by BARONdef. To examine both strategies, and since the proposed recognition and cut generation routines are extremely fast, we first exclude from the test set all problems for which BARONctf did not

Table 2.1: Size statistics for the test set.

Test library	GlobalLib	PrinceLib	MINLPLib	AIMMSLib	NRCLib
Number of problems	46	56	113	15	32
m					
Min	1	1	1	1	63
Max	10399	11162	4981	1449	1242
Average	734	561	731	240	646
n					
Min	3	3	3	3	65
Max	15637	10805	2721	2549	1517
Average	1152	732	486	289	765
nz					
Min	3	3	3	3	170
Max	54591	34729	11685	8697	3866
Average	4967	2463	1904	1084	1965
nnz					
Min	2	2	2	2	84
Max	31256	30002	2262	7260	1608
Average	2386	1506	153	714	829

add any CTF cutting planes during the branch-and-bound search (157 instances). In addition, we remove all trivial problems from the test set (19 instances). In the context of this comparison, a problem is regarded as trivial if it can be globally solved by both algorithms in less than half second. After eliminating all of these problems from the original test set, we obtain a new test set consisting of 86 problems which are used for the computational analysis of this section.

We first assess the performance of the two algorithms in terms of the computational time taken to solve the test problems to global optimality. For this comparison, which is presented in Figure 2.4, we use the performance profiles described in [28], and employ as the performance metric the ratio of the time that an algorithm takes to solve a problem versus the best time of all algorithms. As can be seen from the figure, BARONctf clearly

Table 2.2: Classes of convex-transformable functions appearing in the test problems.

Test library	GlobalLib	PrinceLib	MINLPLib	AIMMSLib	NRCLib
Number of problems	46	56	113	15	32
Class 1: Signomial functions	6 (13%)	10 (18%)	14 (12%)	3 (20%)	0 (0%)
Class 2: Products and ratios of convex and/or concave functions	31 (67%)	43 (77%)	99 (88%)	13 (87%)	32 (100%)
Class 3: Log-concave functions	13 (28%)	9 (16%)	7 (6%)	1 (7%)	0 (0%)

outperforms BARONdef, demonstrating that the generated cutting planes significantly enhance the performance of the global solver.

Next, we examine the impact of the CTF cutting planes by considering the total number of nodes in the branch-and-bound tree, and the maximum number of nodes stored in memory. This analysis, which is shown in Figures 2.5 and 2.6, only considers nontrivial problems for which at least one of the two algorithms proves global optimality within the time limit of 500 seconds (34 instances). Here, we also employ performance profiles, but use performance measures based on the total number of nodes and required memory. As observed in the figures, for most problems, the proposed CTF cutting planes also result in a significant reduction in the total number of iterations and memory required to prove optimality. In our experiments we observed that there are a few instances for which the new cuts lead to a increase in the number of nodes in the tree. This behavior could be attributed to the fact that for a given instance the introduced cuts may affect branching decisions, resulting in a completely different branch-and-bound tree.

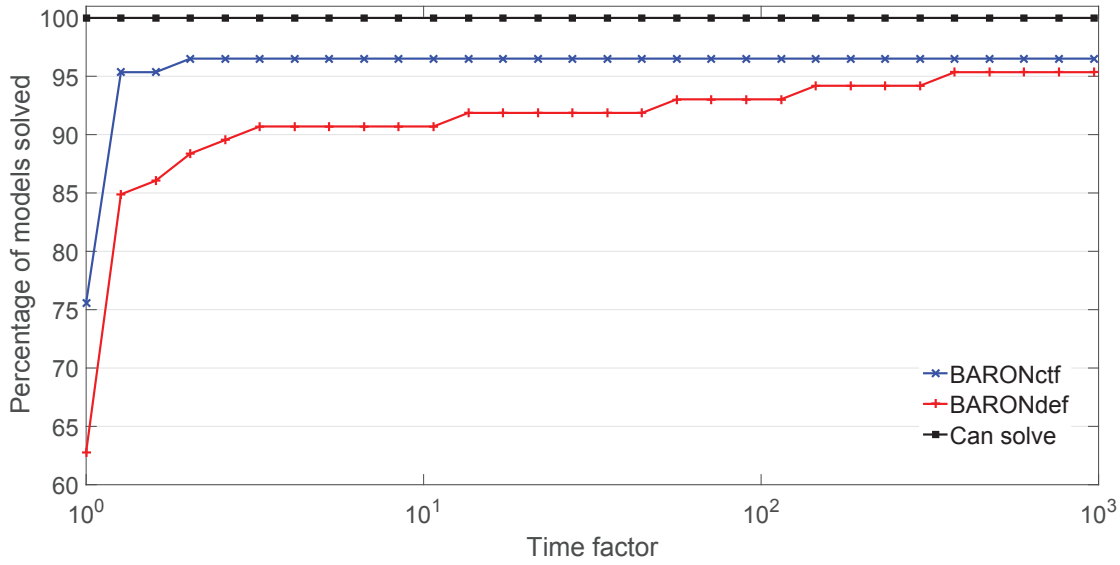


Figure 2.4: Impact of the proposed implementation on the computational time for 86 nontrivial test problems for which BARONctf adds cutting planes for convex-transformable functions during the branch-and-bound search.

For the 34 instances considered in the above comparison, the addition of our cutting planes leads to average reductions of 19% in the CPU time, 18% in the total number of nodes in the branch-and-bound tree, and 11% in the maximum number of nodes in memory. Moreover, the proposed implementation increases by 6% the number of problems that can be solved to global optimality within 500 seconds.

Finally, we analyze the best lower bounds obtained during the branch-and-bound search for nontrivial problems that neither of the two algorithms are able to solve to global optimality within the time limit (52 instances). The results of this analysis are presented in Figure 2.7. As seen in the figure, for most of the problems considered in this comparison, the CTF cutting planes have little effect on the best lower bounds.

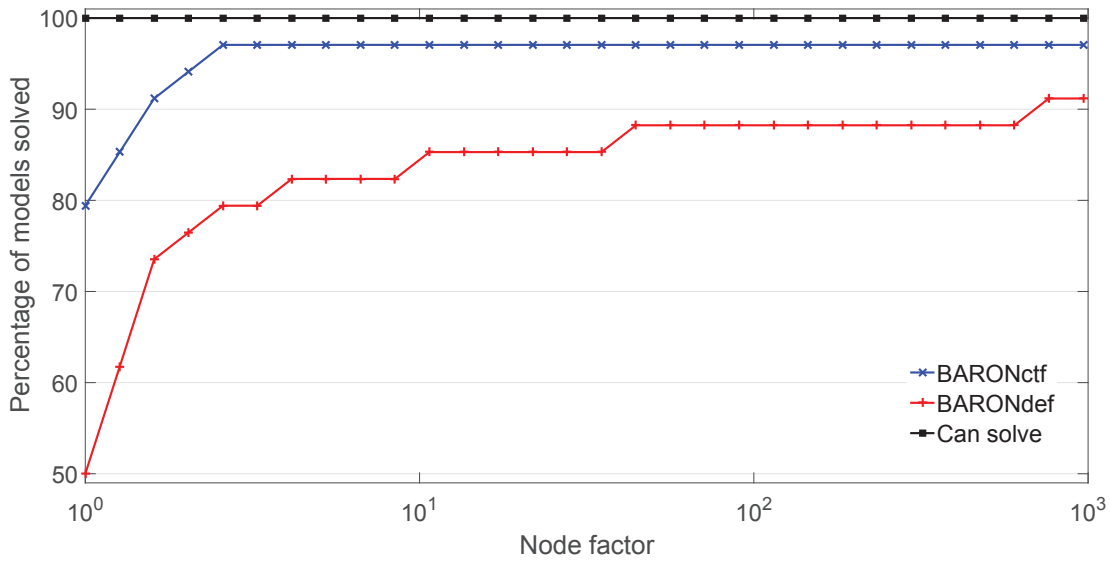


Figure 2.5: Impact of the proposed cutting planes on the total number of nodes for 34 nontrivial problems that are solved to global optimality by at least one of the two algorithms.

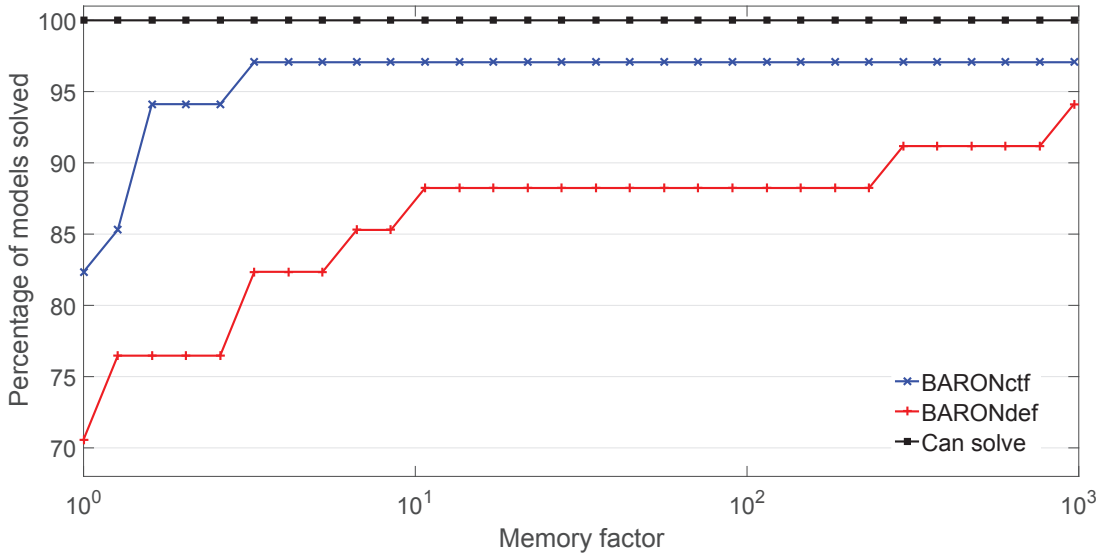


Figure 2.6: Impact of the proposed cutting planes on the memory requirements for 34 nontrivial problems that are solved to global optimality by at least one of the two algorithms.

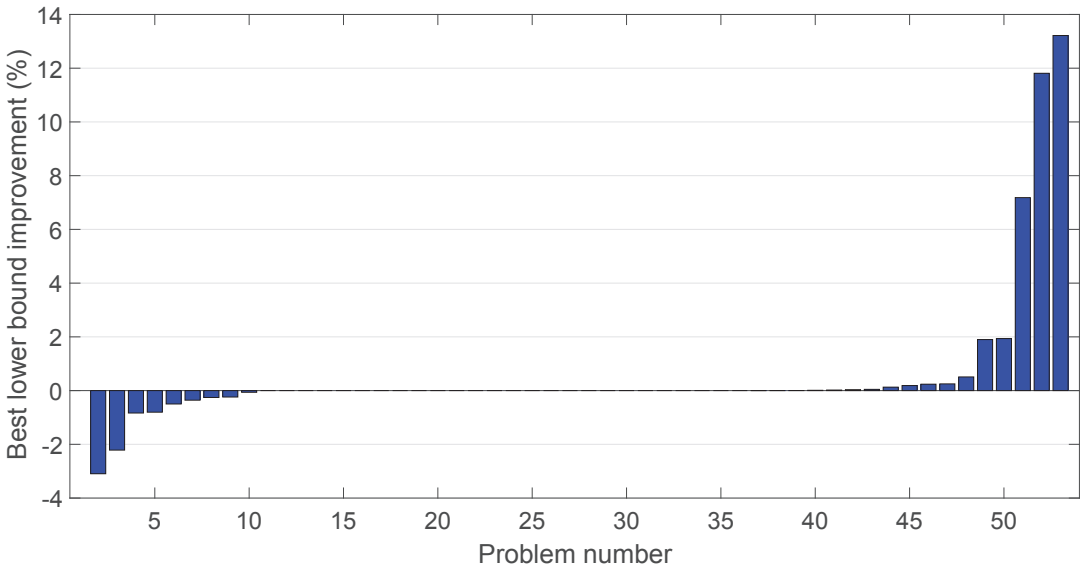


Figure 2.7: Impact of the proposed implementation on the best lower bounds obtained during the branch-and-bound search for 52 nontrivial problems that neither of the two algorithms are able to solve to global optimality within the time limit.

2.5 Conclusions

In this chapter, we examined the effect of integrating G-convexity relaxations into a branch-and-bound global optimization solver. We presented algorithms for the recognition of convex-transformable functions in general nonconvex problems, and introduced a cutting plane generation scheme based on the construction of supporting hyperplanes to G-convexity relaxations. The proposed implementation was integrated within the branch-and-reduce global solver BARON. To assess the benefits of our approach, we tested our implementation on a large number of nonconvex problems selected from a variety of test libraries. Our computational analysis shows that, for our test problems, the generated cutting planes accelerate the convergence speed of the branch-and-bound algorithm, leading to a nearly 20% reduction in the average computational time and total number of nodes in the search tree, and enabling BARON to solve more problems to global optimality.

Chapter 3

Spectral relaxations and branching strategies for global optimization of mixed-integer quadratic programs

3.1 Introduction

We address the global optimization of nonconvex QPs and MIQPs of the form:

$$\begin{array}{ll} \min\limits_{x\in\mathbb{R}^n} & x^TQx + q^Tx \\ \text{s.t.} & Ax = b \\ & Cx \leq d \\ & l \leq x \leq u \\ & x_i \in \mathbb{Z},\ \ \forall i \in J \subseteq \{1,\ldots,n\} \end{array} \tag{3.1}$$

where $Q \in \mathbb{R}^{n\times n}$ is a symmetric matrix which may be indefinite, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $C \in \mathbb{R}^{p\times n}$, and $d \in \mathbb{R}^p$. We assume that lower and upper bounds are finite, i.e., $-\infty < l_i < u_i < \infty,\ \forall i \in \{1,\ldots,n\}$. For the sake of brevity, we use the notation $\mathcal{X} = \{x \in \mathbb{R}^n \mid Ax = b,\ Cx \leq d,\ l \leq x \leq u\}$ in the rest of this chapter. Note also that even though we allow (3.1) to include constraints of the form $Cx \leq d$, we do not use information from these inequalities in order to convexify this problem.

QPs and MIQPs of the form (3.1) arise in a wide variety of scientific and engineering applications including facility location and quadratic assignment [25, 49, 51, 55], molecular conformation [69] and max-cut problems [33]. Given their practical importance, these classes of problems have been studied extensively in the literature and are known to be

very challenging to solve to global optimality.

State-of-the-art global optimization solvers rely on spatial branch-and-bound algorithms to solve problems of the form (3.1) to global optimality. The efficiency of these algorithms primarily depends on the quality of the relaxations constructed during the bounding step. Commonly used relaxations for bounding nonconvex QPs and MIQPs can be broadly classified into three groups. The first group consists of polyhedral relaxations. These relaxations are typically derived via factorable programming methods [60, 86, 92] and reformulation-linearization techniques (RLT) [81, 82, 83, 84, 85]. Both of these approaches involve the introduction of auxiliary variables and additional constraints leading to relaxations which are formulated in a higher dimensional space. The second group is given by semidefinite programming (SDP) relaxations. These relaxations are also constructed in a lifted space by introducing a symmetric matrix of new variables of the form $X = xx^T$. This nonconvex expression is subsequently relaxed by requiring the matrix $X - xx^T$ to be positive semidefinite. This approach has received significant attention in recent years [5, 11, 21, 23, 71, 87]. The third group involves convex quadratic relaxations. These relaxations can be derived through different approaches including separable programming procedures [68], d.c. programming techniques [95], and quadratic convex reformulation methods [14, 15, 16, 17].

In this chapter, we investigate a family of relaxations which falls under the third group. In particular, we consider convex quadratic relaxations which are derived by convexifying the objective function of (3.1) through diagonal and nondiagonal perturbations of the quadratic matrix Q. We revisit a very well-known technique which uses the smallest eigenvalue of the matrix Q to convexify nonconvex quadratic functions of the form x^TQx. Through numerical experiments, we show that, despite its simplicity, this technique leads to convex quadratic relaxations which in many cases are significantly tighter than the polyhedral relaxations that are typically used by state-of-the-art global optimization solvers.

Motivated by these promising results, we refine this approach in several directions and make several theoretical and algorithmic contributions.

Our first contribution is a novel convex quadratic relaxation for problems of the form (3.1) which is derived by using information from both the matrix Q and the equality constraints $Ax = b$. Under this approach, the quadratic function $x^T Qx$ is convexified by constructing a perturbation of the matrix Q obtained by solving a generalized eigenvalue problem involving both the Q and the A matrices. We show that the resulting relaxation is at least as tight as the relaxation constructed by using the smallest eigenvalue of the matrix Q.

In our second contribution, we consider another convex quadratic relaxation in which the quadratic function $x^T Qx$ is convexified by using the smallest eigenvalue of the matrix $Z^T QZ$, where Z is a basis for the nullspace of the matrix A. We devise a relatively simple procedure which allows us to approximate the bound given by this relaxation without having to compute the basis Z. Moreover, we show that the relaxations obtained through this technique are at least as tight as the other two quadratic relaxations mentioned above. Unlike the factorable, RLT, and SDP relaxations, which are typically used for bounding problems of the form (3.1), the quadratic relaxations considered in this chapter are constructed in the space of the original problem variables. Additionally, they are very inexpensive to solve.

In our third contribution, we prove that the aforementioned quadratic relaxations are equivalent to some particular SDP relaxations. These results facilitate the theoretical comparisons with other relaxations that have been proposed in the literature. In particular, we show that the convexification using the smallest eigenvenvalue of $Z^T QZ$ leads to the best relaxation in the class of relaxations considered in this chapter.

Our fourth contribution addresses the question of how to improve the proposed quadratic relaxations with branching. We introduce a novel eigenvalue-based branching variable selection strategy for nonconvex binary quadratic programs. This strategy involves an

effective approximation of the impact of branching decisions on the quality of the corresponding relaxations.

In order to investigate the impact of the proposed techniques on the performance of branch-and-bound algorithms, we develop an implementation which integrates the relaxations and branching strategies considered in this chapter into the state-of-the-art global optimization solver BARON [76]. The new quadratic relaxations are incorporated into BARON's portfolio of relaxations and are invoked according to a new dynamic relaxation selection rule which switches between different classes of relaxations based on their relative strength. We test our implementation by conducting extensive numerical experiments on a large collection of problems. Results demonstrate that the proposed implementation leads to a very significant improvement in the performance of BARON. Moreover, for many of the test problems, our implementation results in a new version of BARON which outperforms other state-of-the-art optimization solvers including CPLEX and GUROBI.

The remainder of this chapter is organized as follows. In §3.2 we review various relaxation approaches which have been considered in the literature for bounding nonconvex QPs and MIQPs. Then, in §3.3 we present the convex quadratic relaxations considered in this chapter and investigate their theoretical properties. In §3.4 we introduce novel eigenvalue-based branching strategies. This is followed by a description of our computational implementation in §3.5. In §3.6, we present the results of an extensive computational study which includes a comparison between different classes of relaxations, an analysis of the impact of the proposed implementation on the performance of BARON, and a comparison between several state-of-the-art global optimization solvers. Finally, §3.7 presents conclusions from this work.

Notation

We denote by $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$ the set of natural, integer, and real numbers, respectively. The set of nonnegative real numbers is denoted by $\mathbb{R}_{\geq 0}$. We use $\mathbb{1} \in \mathbb{R}^n$ to denote a vector of ones. The i-th element of a vector $x \in \mathbb{R}^n$ is denoted by x_i. Given a vector $d \in \mathbb{R}^n$, the notation $\text{diag}(d)$ is used for the diagonal matrix whose diagonal entries are given by the elements of d. The unit vector in the i-th direction is denoted by e_i. We denote by I_n the $n \times n$ identity matrix. For a matrix $A \in \mathbb{R}^{m \times n}$, we use $A_{i\cdot}$, $A_{\cdot j}$ and A_{ij}, to denote its i-th row, j-th column and (i, j)-th entry, respectively. Let $\mathbb{S}^n$ denote the set of $n \times n$ real, symmetric matrices. Given a matrix $M \in \mathbb{S}^n$, we use λ_i to represent its i-th eigenvalue and v^i for the corresponding eigenvector. For $M \in \mathbb{S}^n$, the notation $M \succcurlyeq 0$ and $M \succ 0$, indicates that M is positive semidefinite and positive definite, respectively. Let $M, N \in \mathbb{S}^n$ with $N \succ 0$. We use $\lambda_{\min}(M)$ to represent the smallest eigenvalue of M. Similarly, we denote by $\lambda_{\min}(M, N)$ the smallest generalized eigenvalue of the problem $Mv = \lambda Nv$, where $v \in \mathbb{R}^n$. The inner product between matrices $M, P \in \mathbb{S}^n$ is denoted by $\langle M, P \rangle = \sum_{i=1}^n \sum_{j=1}^n M_{ij} P_{ij}$.

3.2 Current relaxations for nonconvex QPs and MIQPs

In this section, we review various types of relaxations that have been proposed in the literature for bounding problems of the form (3.1).

3.2.1 Polyhedral relaxations

We start this section by reviewing one of the simplest polyhedral relaxations which can be constructed for problems of the form (3.1). The procedure used to derive this relaxation consists of two steps. In the first step, we introduce the new variables $X_{ij} = x_i x_j$ in order to obtain a reformulation of (3.1) in a higher-dimensional space. In the second step, we relax the integrality conditions and convexify the bilinear terms $x_i x_j$ by using their McCormick

envelopes [2, 60]. This results in the following linear relaxation of (3.1):

$$\min_{x\in\mathcal{X},X} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}X_{ij} + \sum_{i=1}^{n} q_i x_i \tag{3.2a}$$

$$\text{s.t.} \quad X_{ij} \geq l_i x_j + l_j x_i - l_i l_j,\ i = 1,\dots,n, j = i,\dots,n \tag{3.2b}$$

$$X_{ij} \geq u_i x_j + u_j x_i - u_i u_j,\ i = 1,\dots,n, j = i,\dots,n \tag{3.2c}$$

$$X_{ij} \leq l_i x_j + u_j x_i - l_i u_j,\ i = 1,\dots,n, j = i,\dots,n \tag{3.2d}$$

$$X_{ij} \leq u_i x_j + l_j x_i - u_i l_j,\ i = 1,\dots,n, j = i,\dots,n \tag{3.2e}$$

$$X_{ij} = X_{ji},\ i = 1,\dots,n, j = (i+1),\dots,n \tag{3.2f}$$

where (3.2b)–(3.2e) are the so-called *McCormick inequalities*. This relaxation is often referred to as the *McCormick relaxation* of (3.1). Due to their simplicity, McCormick relaxations have been implemented in most global optimization packages. However, an important drawback of these relaxations is the fact that they often lead to relatively weak bounds. As a result, McCormick relaxations are typically tightened by adding various classes of valid inequalities such as RLT-based cuts [98, 99], facets of the envelopes of edge-concave and multilinear subexpressions [9, 10, 63, 66], SDP-based cuts [29, 79] and mixed-integer cuts [19].

Another popular approach for obtaining a polyhedral relaxation for (3.1) relies on the reformulation linearization techniques (RLT) [83]. To apply these techniques to (3.1), we start by defining the following *bound factors*:

$$(x_i - l_i) \geq 0, \quad (u_i - x_i) \geq 0, \quad i = 1,\dots,n \tag{3.3}$$

and the *constraint factors*:

$$\left(d_k - \sum_{i=1}^{n} C_{ki}x_i\right) \geq 0,\ k = 1,\dots,p \tag{3.4}$$

The RLT procedure also involves two steps. The first step, also known as the *reformulation phase*, consists in constructing a problem equivalent to (3.1) by adding redundant nonlinear constraints. These additional constraints are obtained by multiplying each equation

in (3.1) by each variable x_j, and by taking all the possible pairwise products involving the bound and constraint factors. The second step, also known as the *linearization phase*, involves the relaxation of the integrality conditions and linearization of all the nonlinear terms $x_i x_j$ by introducing the new variables X_{ij}. The application of this procedure to (3.1) leads to the following linear relaxation:

$$
\begin{aligned}
\min_{x\in\mathcal{X},X}\ & \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}X_{ij} + \sum_{i=1}^{n} q_i x_i \\
\text{s.t. } & \text{Eqs. (3.2b)} - \text{(3.2f)} \\
& \sum_{i=1}^{n} A_{ki}X_{ij} = b_k x_j,\ k = 1,\dots,m,\ j = 1,\dots n \\
& \sum_{i=1}^{n} C_{ki}X_{ij} - l_j \sum_{i=1}^{n} C_{ki}x_i - d_k x_j \le -l_j d_k,\ k = 1,\dots,p,\ j = 1,\dots n \\
& -\sum_{i=1}^{n} C_{ki}X_{ij} + u_j \sum_{i=1}^{n} C_{ki}x_i + d_k x_j \le u_j d_k,\ k = 1,\dots,p,\ j = 1,\dots n, \\
& -\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ki}C_{lj}X_{ij} + \sum_{i=1}^{n}(d_l C_{ki} + d_k C_{li})x_i \le d_k d_l,\ k,l = 1,\dots,p
\end{aligned}
\tag{3.5}
$$

This relaxation is often referred to as the *first-level RLT relaxation* of (3.1). For the case in which (3.1) is a box-constrained problem, the first-level RLT relaxation (3.5) is equivalent to the McCormick relaxation (3.2).

The RLT procedure can be used to construct an n-level hierarchy of relaxations for (3.1), where at each level of the hierarchy the resulting relaxation is at least as tight as the relaxation corresponding to the previous level. However, this relaxation strengthening comes at a heavy computational price, since the number of variables and constraints increases quickly, which makes the resulting relaxations very expensive to solve. In this chapter, we limit our attention to the first-level RLT relaxations. For additional details on the RLT approach, we refer the reader to [81, 82, 83, 84, 85].

3.2.2 SDP relaxations

To derive one of the simplest SDP relaxations for (3.1), we first reformulate this problem by introducing a symmetric matrix of new variables $X = xx^T$. Then, we relax this nonconvex equation to a semidefinite constraint and drop the integrality conditions to obtain the following SDP relaxation:

$$\min_{x\in\mathcal{X},X} \ \langle Q, X\rangle + q^T x \tag{3.6a}$$
$$\text{s.t.} \ \ X - xx^T \succcurlyeq 0 \tag{3.6b}$$

This relaxation is often referred to as the *Shor relaxation* of (3.1) [87]. The Shor relaxation can be strengthened by including additional valid constraints. This can be achieved, for instance, by adding the McCormick inequalities corresponding to the diagonal elements of the matrix X, which results in the following SDP relaxation:

$$\min_{x\in\mathcal{X},X} \ \langle Q, X\rangle + q^T x \tag{3.7a}$$
$$\text{s.t.} \ \ X - xx^T \succcurlyeq 0 \tag{3.7b}$$
$$X_{ii} \geq 2l_i x_i - l_i^2, \ i = 1,\dots,n \tag{3.7c}$$
$$X_{ii} \geq 2u_i x_i - u_i^2, \ i = 1,\dots,n \tag{3.7d}$$
$$X_{ii} \leq u_i x_i + l_i x_i - u_i l_i, \ i = 1,\dots,n \tag{3.7e}$$

It is easy to verify that (3.7c) and (3.7d) are implied by $X - xx^T \succcurlyeq 0$ and hence redundant in this formulation.

The relaxation (3.7) can be further tightened by including constraints derived from $Ax = b$. For example, we can construct the following SDP relaxation by considering a lifting of the valid equalities $\sum_{j=1}^n A_{kj}x_i x_j = b_k x_i, \ k = 1,\dots,m, \ i = 1,\dots,n$, into the space of (x, X):

$$\min_{x\in\mathcal{X},X} \ \langle Q, X\rangle + q^T x \tag{3.8a}$$

$$\text{s.t. Eqs. (3.7b)} - \text{(3.7e)} \tag{3.8b}$$

$$\sum_{j=1}^{n} A_{kj} X_{ij} = b_k x_i,\ k = 1, \ldots, m,\ i = 1, \ldots, n \tag{3.8c}$$

Another alternative involves the addition of a single constraint derived by lifting the valid equality $(Ax - b)^T(Ax - b) = 0$ into the space of (x, X). This leads to the following SDP relaxation:

$$\min_{x \in \mathcal{X}, X} \ \langle Q, X \rangle + q^T x \tag{3.9a}$$

$$\text{s.t. Eqs. (3.7b)} - \text{(3.7e)} \tag{3.9b}$$

$$\langle A^T A, X \rangle - 2(A^T b)^T x + b^T b = 0 \tag{3.9c}$$

Note that the SDPs (3.8) and (3.9) are equivalent (see Proposition 5 in [31] for details). An SDP relaxation even tighter than (3.9) can be constructed by including all of the McCormick inequalities instead of only considering those corresponding to the diagonal elements of the matrix X. The resulting relaxation is given by:

$$\min_{x \in \mathcal{X}, X} \ \langle Q, X \rangle + q^T x \tag{3.10a}$$

$$\text{s.t. Eqs. (3.7b), (3.2b)} - \text{(3.2e)} \tag{3.10b}$$

$$\langle A^T A, X \rangle - 2(A^T b)^T x + b^T b = 0 \tag{3.10c}$$

As shown in [5], when all of the constraints (3.2b)-(3.2e) are considered, this relaxation can become very expensive to solve. For a detailed discussion on SDP relaxations, we refer the reader to [5, 11, 21, 23, 71, 87].

3.2.3 Convex quadratic relaxations

In the following, we briefly discuss some convex quadratic relaxations which have been proposed in the literature for problems of the form (3.1).

3.2.3.1 Separable programming relaxation

A well-known procedure for deriving a convex quadratic relaxation of (3.1) was proposed by Pardalos et al. [68]. This method consists of two steps. In the first step, we use the eigendecomposition of Q to express this matrix as $Q = \sum_{i=1}^{n} \lambda_i v^i v^{i^T}$, and introduce the new variables $y_i = v^{i^T} x,\ i = 1, \ldots, n$. This results in the following reformulation of (3.1):

$$\begin{array}{ll} \min\limits_{x \in \mathcal{X}, y} & \sum\limits_{i:\lambda_i > 0} \lambda_i y_i^2 + \sum\limits_{i:\lambda_i < 0} \lambda_i y_i^2 + q^T x \\ \text{s.t.} & y_i = v^{i^T} x,\ i = 1, \ldots, n \\ & L_i \leq y_i \leq U_i,\ i = 1, \ldots, n \\ & x_i \in \mathbb{Z},\ \forall i \in J \end{array} \tag{3.11}$$

where L_i and U_i denote lower and upper bounds on y_i, respectively. Note that this transformation leads to a reformulated problem with a separable objective function. In the second step, we relax the integrality conditions in (3.11) and use the concave envelope of y_i^2 over $[L_i, U_i]$ to derive the following relaxation:

$$\begin{array}{ll} \min\limits_{x \in \mathcal{X}, y} & \sum\limits_{i:\lambda_i > 0} \lambda_i y_i^2 + \sum\limits_{i:\lambda_i < 0} \lambda_i \left((L_i + U_i) y_i - L_i U_i \right) + q^T x \\ \text{s.t.} & y_i = v^{i^T} x,\ i = 1, \ldots, n \\ & L_i \leq y_i \leq U_i,\ i = 1, \ldots, n \end{array} \tag{3.12}$$

Under this approach the bounds on the y_i variables are obtained through the solution of the following linear programs:

$$L_i = \min_{x \in \mathcal{X}} \ v^{i^T} x, \qquad U_i = \max_{x \in \mathcal{X}} \ v^{i^T} x, \qquad i = 1, \ldots, n \tag{3.13}$$

3.2.3.2 D.C. programming relaxations

Let $\mathcal{C} \subseteq \mathbb{R}^n$ be a convex set and $f : \mathcal{C} \to \mathbb{R}$ a nonconvex function. Then, we say that f is a d.c. function if it can be expressed as the difference of two convex functions [95]. It is simple to show that the objective function of (3.1) is a d.c. function (see chapter 3 in [39] for details). In order to construct a generic d.c. programming relaxation for (3.1), we can proceed as follows. First, we decompose the objective function as $f(x) = x^T Q x + q^T x =$

$g(x) - h(x)$, where $g(x)$ and $h(x)$ are both convex quadratic functions. Second, we drop the integrality conditions and substitute $h(x)$ with a concave overestimator over $\mathcal{X}$, which we denote by $\bar{h}_{\mathcal{X}}(x)$. This leads to the following relaxation:

$$\min_{x \in \mathcal{X}} \quad g(x) - \bar{h}_{\mathcal{X}}(x) \tag{3.14}$$

Clearly, given a nonconvex quadratic function, a d.c. decomposition is not unique [95]. Therefore, the quality of the relaxation (3.14) depends to a very large extent on the choice of the functions g and h, as well as the tightness of the concave overestimator of h. A particular type of relaxation which can be derived through d.c. programming techniques is the classical αBB relaxation [4], which for (3.1) takes the form:

$$\min_{x \in \mathcal{X}} \quad x^T Q x + q^T x - \sum_{i=1}^{n} \alpha_i (x_i - l_i)(u_i - x_i) \tag{3.15}$$

where $a_i,\ i = 1, \ldots, n$, are nonnegative parameters chosen such that the objective function of (3.15) is convex over $\mathcal{X}$. Another example of a d.c. programming relaxation for (3.1) is the one proposed by Bomze and Locatelli [18]:

$$\min_{x \in \mathcal{X}} \quad g_{x_0,B}(x) - \text{conc}_{\mathcal{X}} h_{x_0,B}(x) \tag{3.16}$$

where $g_{x_0,B}(x) = (x - x_0)^T (Q + B)(x - x_0)$, $h_{x_0,B}(x) = x^T B x - q^T x + x_0^T (Q + B) x_0 - 2 x_0^T (Q + B) x$, $x_0 \in \mathbb{R}^n$, $B \in \mathbb{S}^n$ such that $Q + B \succcurlyeq 0$ and $B \succcurlyeq 0$, and $\text{conc}_{\mathcal{X}} h_{x_0,B}(x)$ denotes the concave envelope of $h_{x_0,B}(x)$ over $\mathcal{X}$. The matrix B, which can be fully dense, is referred to as a difference of convex decomposition (d.c.d.) of Q. Let B_1 and B_2 be two d.c.d.s. of Q. Then, B_1 is said to *dominate* B_2 if $B_2 - B_1 \succcurlyeq 0$. If no other d.c.d. of Q dominates B_1, then B_1 is said to be an *undominated* d.c.d. of Q. Bomze and Locatelli [18] showed that: (i) the bound given by (3.16) is independent of x_0, and (ii) the tightest relaxation of the form (3.16) is obtained when B is an undominated d.c.d. of Q. Unfortunately, the procedure for constructing an undominated d.c.d. of Q involves the solution of a sequence of semidefinite programs. Moreover, even though $\text{conc}_{\mathcal{X}} h_{x_0,B}(x)$ is polyhedral and its facets can be obtained from the solution of a particular a linear program, this linear program

can be very expensive to solve because its size grows exponentially with the number of variables n.

3.2.3.3 Relaxations based on quadratic convex reformulations

These relaxations are constructed in the context of the Quadratic Convex Reformulation (QCR) methods. To illustrate these techniques, we consider the case in which all the variables of (3.1) are binary:

$$\begin{array}{ll} \min\limits_{x\in\{0,1\}^n} & x^TQx + q^Tx \\ \text{s.t.} & Ax = b,\ Cx \leq d \end{array} \tag{3.17}$$

In the interest of brevity, we use the notation $\mathcal{X}_B = \{x \in \{0,1\}^n \mid Ax = b,\ Cx \leq d\}$ in the remainder of this section.

The QCR approaches involve two steps. The first step consists in reformulating (3.17) into an equivalent binary quadratic program whose continuous relaxation is convex. This is achieved by perturbing the quadratic matrix Q. In the second step, the reformulated problem is solved using a branch-and-bound algorithm. At each node of the branch-and-bound tree, the lower bound is obtained by solving the continuous relaxation of the reformulated problem, which is a convex quadratic program. Note that the quadratic relaxations solved throughout the branch-and-bound tree only differ from one another on the binary variables that are fixed.

One of the earliest references to these methods is found in a paper by Hammer and Rubin [37], in which the following reformulation for (3.17) is proposed:

$$\min_{x\in\mathcal{X}_B} \quad x^TQ_\lambda x + q_\lambda^T x \tag{3.18}$$

where $Q_\lambda = Q - \min(0, \lambda_{\min}(Q))I_n$ and $q_\lambda = q + \min(0, \lambda_{\min}(Q))\mathbb{1}$. It is simple to check that $Q_\lambda \succcurlyeq 0$, and that the objective functions of (3.17) and (3.18) are equivalent $\forall x \in \{0,1\}^n$.

In recent years, Hammer and Rubin's approach was refined by Billionnet and Elloumi [14] who considered the following reformulation of (3.17):

$$\min_{x \in \mathcal{X}_B} \quad x^T Q_{d_u} x + q_{d_u}^T x \tag{3.19}$$

where $Q_{d_u} = Q + \text{diag}(d_u)$, $q_{d_u} = q - d_u$, and $d_u \in \mathbb{R}^n$. The perturbation parameter d_u is determined in a way such that $Q_{d_u} \succcurlyeq 0$ and the bound given by the continuous relaxation of (3.19) is as tight as possible. This is achieved by solving the SDP (3.7) and setting the entries of d_u to the optimal values of the dual variables associated with the constraints (3.7e).

In a subsequent paper, Billionnet et al. [17] used information from the equality constraints to improve the bound given by the continuous relaxation of (3.19). In particular, they considered the following reformulation of (3.17):

$$\min_{x \in \mathcal{X}_B} \quad x^T Q_{d_q, \Theta_q} x + q_{d_q, \Theta_q}^T x \tag{3.20}$$

where $Q_{d_q,\Theta_q} = Q + \text{diag}(d_q) + \frac{1}{2}(\Theta_q^T A + A^T \Theta_q^T)$, $q_{d_q,\Theta_q} = q - d_q - \Theta_q^T b$, $d_q \in \mathbb{R}^n$ and $\Theta_q \in \mathbb{R}^{m \times n}$. Similarly to (3.19), the perturbation parameters d_q and Θ_q are chosen such that $Q_{d_q,\Theta_q} \succcurlyeq 0$ and the bound of the continuous relaxation of (3.20) is maximized. This is done by solving the SDP (3.8), and setting the entries of d_q and Θ_q to the optimal values of the dual variables associated with the constraints (3.7e) and (3.8c), respectively. Note that the continuous relaxations of (3.19) and (3.20) provide the same bounds as the SDP relaxations (3.7) and (3.8), respectively.

In more recent papers, the QCR approach has been extended beyond the binary case to some particular classes of general integer and mixed-integer quadratic programs [15, 16]. In these extensions, the general integer variables are replaced with their binary expansions and the perturbation parameters are determined by solving the semidefinite programs (3.9) and (3.10). For a detailed discussion on QCR methods, we refer the reader to [14, 15, 16, 17].

3.3 Spectral relaxations for nonconvex QPs and MIQPs

In the following, we present a family of convex quadratic relaxations for problems of the form (3.1), and investigate their theoretical properties. Before providing a detailed derivation of these relaxations, we state two results which we will repeatedly use throughout this section. First, we recall that the minimum eigenvalue of a matrix M and the minimum generalized eigenvalue of a pair of matrices (M, N), with $M, N \in \mathbb{S}^n, N \succ 0$, can be expressed in terms of the Rayleigh quotient as [34]:

$$\lambda_{\min}(M) = \min_{x \neq 0} \ \frac{x^T M x}{x^T x} \ \text{ and } \ \lambda_{\min}(M, N) = \min_{x \neq 0} \ \frac{x^T M x}{x^T N x}. \tag{3.21}$$

Second, we provide a particularly useful formulation for the dual of a certain SDP.

Proposition 3.1. *Consider the following SDP*

$$\min_{x \in \mathcal{X}, X} \ \langle \hat{Q}, X \rangle + \hat{q}^T x \tag{3.22a}$$
$$\text{s.t.} \ \ X - xx^T \succcurlyeq 0 \tag{3.22b}$$
$$\langle \hat{C}_i, X \rangle + \hat{c}_i x + \hat{d}_i = 0, \ i = 1, \dots, q_1 \tag{3.22c}$$
$$\langle \bar{C}_i, X \rangle + \bar{c}_i x + \bar{d}_i \leq 0, \ i = 1, \dots, q_2 \tag{3.22d}$$

for some $\hat{Q}, \hat{C}_i, \bar{C}_i \in \mathbb{S}^n, \hat{q}, \hat{c}_i, \bar{c}_i \in \mathbb{R}^n$ and $\hat{d}_i, \bar{d}_i \in \mathbb{R}$. The dual of (3.22) *is given by*

$$\max_{\alpha \in \mathbb{R}^{q_1}, \beta \in \mathbb{R}^{q_2}_{\geq 0} : \hat{Q}_{\alpha,\beta} \succcurlyeq 0} \left\{ \min_{x \in \mathcal{X}} \ \ x^T \hat{Q}_{\alpha,\beta} x + \hat{q}_{\alpha,\beta}^T x + \hat{d}_{\alpha,\beta} \ \ \right\}.$$

where $\hat{Q}_{\alpha,\beta} = \hat{Q} + \sum_{i=1}^{q_1} \alpha_i \hat{C}_i + \sum_{i=1}^{q_2} \beta_i \bar{C}_i$, $\hat{q}_{\alpha,\beta} = \hat{q} + \sum_{i=1}^{q_1} \alpha_i \hat{c}_i + \sum_{i=1}^{q_2} \beta_i \bar{c}_i$, and $\hat{d}_{\alpha,\beta} = \sum_{i=1}^{q_1} \alpha_i \hat{d}_i + \sum_{i=1}^{q_2} \beta_i \bar{d}_i$.

Proof. By dualizing the constraints (3.22c) using the multipliers $\alpha_i \in \mathbb{R}$, for $i = 1, \dots, q_1$, and the constraints (3.22d) using the multipliers $\beta_i \in \mathbb{R}_{\geq 0}$, for $i = 1, \dots, q_2$, we have that the Lagrangian dual of the SDP (3.22) is given by:

$$\max_{\alpha\in\mathbb{R}^{q_1},\beta\in\mathbb{R}^{q_2}_{\geq 0}} \left\{ \begin{array}{ll} \min\limits_{x\in\mathcal{X}} & \langle \hat{Q}_{\alpha,\beta}, X\rangle + \hat{q}_{\alpha,\beta}^T x + \hat{d}_{\alpha,\beta} \\ \text{s.t.} & X - xx^T \succcurlyeq 0 \end{array} \right\} \tag{3.23}$$

Since the variables x have finite lower and upper bounds, the set $\mathcal{X}$ is bounded. Hence, in order for the inner minimization to be bounded below, we need to choose α and β such that $\hat{Q}_{\alpha,\beta} \succcurlyeq 0$. This restriction on α and β implies that the optimal solution of the inner minimization problem satisfies $X = xx^T$. The claim follows after substituting $X = xx^T$.

□

3.3.1 Eigenvalue relaxation

A very well-known technique for deriving convex quadratic relaxations for (3.1) starts by considering the following reformulation of (3.1):

$$\begin{array}{ll} \min\limits_{x\in\mathcal{X}} & x^TQx + q^Tx + \alpha_e \sum\limits_{i=1}^n x_i^2 - \alpha_e \sum\limits_{i=1}^n x_i^2 \\ \text{s.t.} & x_i \in \mathbb{Z}, \ \ \forall i \in J \end{array} \tag{3.24}$$

where α_e is a nonnegative scalar. By dropping the integrality conditions from (3.24) and using the concave envelope of x_i^2 over $[l_i, u_i]$, we obtain the following relaxation:

$$\min_{x\in\mathcal{X}} \quad x^TQx + q^Tx + \alpha_e \sum_{i=1}^n x_i^2 - \alpha_e \sum_{i=1}^n ((l_i + u_i)x_i - l_iu_i) \tag{3.25}$$

which can be equivalently written as:

$$\min_{x\in\mathcal{X}} \quad x^TQ_{\alpha_e}x + q_{\alpha_e}^T x + k_{\alpha_e} \tag{3.26}$$

where $Q_{\alpha_e} = Q + \alpha_e I_n$, $q_{\alpha_e} = q - \alpha_e(l + u)$, and $k_{\alpha_e} = \alpha_e l^T u$.

In order to ensure that (3.26) is a convex relaxation of (3.1), it is sufficient to choose $\alpha_e \geq -\min(0, \lambda_{\min}(Q))$, since this renders the matrix Q_{α_e} positive semidefinite. Moreover, it is simple to check that $\alpha_e = -\min(0, \lambda_{\min}(Q))$ provides the tightest convex relaxation of the form (3.26) for which $Q_{\alpha_e} \succcurlyeq 0$. We refer to this convex relaxation as the *eigenvalue relaxation* of (3.1).

Two interesting observations can be made on (3.26) when $\alpha_e \geq -\min(0, \lambda_{\min}(Q))$. First, the derivation of (3.26) can be seen as an application of the d.c. programming technique reviewed in §3.2.3.2, whereby the objective function of (3.1) is expressed as the difference of the convex quadratic functions $g(x) = x^T Q x + q^T x + \alpha_e \sum_{i=1}^n x_i^2$ and $h(x) = \alpha_e \sum_{i=1}^n x_i^2$. Second, (3.26) is equivalent to the αBB relaxation discussed in §3.2.3.2 if we set $\alpha_i = \alpha_e, \ \forall i = 1, \dots n$, in (3.15).

Note also that if all the variables in (3.1) are binary and $\alpha_e = -\min(0, \lambda_{\min}(Q))$, then (3.26) is equivalent to the continuous relaxation of the convex binary quadratic program (3.18), which was considered by Hammer and Rubin [37].

Even though the eigenvalue relaxation is relatively simple to construct, in many cases it can be significantly tighter than the polyhedral relaxations commonly used in state-of-the-art global optimization solvers (see §3.6.2). Motivated by this observation, we further investigate the theoretical properties of this relaxation. In particular, we show that the eigenvalue relaxation is equivalent to the following SDP:

$$\min_{x \in \mathcal{X}, X} \ \langle Q, X \rangle + q^T x \tag{3.27a}$$

$$\text{s.t.} \ \ X - x x^T \succcurlyeq 0 \tag{3.27b}$$

$$\langle I_n, X \rangle - (l + u)^T x + l^T u \leq 0 \tag{3.27c}$$

The SDP in (3.27) can be obtained from (3.7) by aggregating the constraints in (3.7e) and dropping the redundant inequalities (3.7c) and (3.7d). However, unlike the SDP (3.7), the optimal objective of SDP (3.27) can be obtained by solving the QP (3.26).

Proposition 3.2. *Suppose that the matrix Q is indefinite. Let $\alpha_e = -\lambda_{\min}(Q)$ in* (3.26)*. Denote by μ_{EIG} and $\mu_{\text{SDP\_EIG}}$ the optimal objective function values in* (3.26) *and* (3.27)*, respectively. Then, $\mu_{\text{EIG}} = \mu_{\text{SDP\_EIG}}$.*

Proof. The proof of this proposition relies on strong duality holding for (3.27). We start by showing that (3.27) admits a strictly feasible solution. Let $\bar{x} \in \mathbb{R}^n$ be a vector such that

$A\bar{x} = b$, $C\bar{x} < d$, and $l < \bar{x} < u$. Recall that the concave envelope of x_i^2 over $[l_i, u_i]$ is given by $(l_i + u_i)x_i - l_i u_i$. Since $l_i < \bar{x}_i < u_i, \forall i = 1, \ldots, n$, it follows that $(l_i + u_i)x_i - l_i u_i - \bar{x}_i^2 > 0, \forall i = 1, \ldots, n$. Define $\epsilon := (l_1 + u_1)x_1 - l_1 u_1 - \bar{x}_1^2$. Clearly, there exists $\delta \in \mathbb{R}$ such that $0 < \delta < \epsilon$. Let $\bar{X} \in \mathbb{S}^n$ be the matrix satisfying:

$$\begin{array}{l} \bar{X}_{11} = (l_1 + u_1)\bar{x}_1 - l_1 u_1 - \delta \\ \bar{X}_{ii} = (l_i + u_i)\bar{x}_i - l_i u_i, \ i = 2, \ldots, n \\ \bar{X}_{ij} = \bar{X}_{ji} = \bar{x}_i \bar{x}_j, \ i = 1, \ldots, n, \ j = (i+1), \ldots, n \end{array} \tag{3.28}$$

Then, it is simple to check that (3.27c) is strictly satisfied by $(\bar{x}, \bar{X})$. Define $\hat{X} := \bar{X} - \bar{x}\bar{x}^T$. From this definition, it follows that $\hat{X} \in \mathbb{S}^n$ is a diagonal matrix with entries given by:

$$\begin{array}{l} \hat{X}_{11} = (l_1 + u_1)\bar{x}_1 - l_1 u_1 - \bar{x}_1^2 - \delta \\ \hat{X}_{ii} = (l_i + u_i)\bar{x}_i - l_i u_i - \bar{x}_i^2, \ i = 2, \ldots, n \end{array} \tag{3.29}$$

It is clear that $\hat{X}_{ii} > 0$, $\forall i = 1, \ldots, n$. It follows that the matrix $\hat{X}$ is positive definite and $(\bar{x}, \bar{X})$ is a strictly feasible solution to (3.27). Therefore, Slater's condition is satisfied by (3.27), which implies that strong duality holds and the optimal value of the dual problem is attained.

Now, we consider the dual of (3.27). By Proposition 3.1, this dual is given by:

$$\max_{\alpha_e \in \mathbb{R}_{\geq 0} : Q_{\alpha_e} \succcurlyeq 0} \left\{ \min_{x \in \mathcal{X}} \quad x^T Q_{\alpha_e} x + q_{\alpha_e}^T x + k_{\alpha_e} \quad \right\} \tag{3.30}$$

where α_e is the multiplier for the constraint (3.27c), $Q_{\alpha_e} = Q + \alpha_e I_n$, $q_{\alpha_e} = q - \alpha_e(l + u)$, and $k_{\alpha_e} = \alpha_e l^T u$. Since the matrix Q is indefinite, it is clear that the matrix $Q_{\alpha_e} \succcurlyeq 0$ for $\alpha_e \geq -\lambda_{\min}(Q)$. Then, it is simple to check that the maximum of (3.30) is attained for $\alpha_e = -\lambda_{\min}(Q)$, which implies that $\mu_{\text{SDP\_EIG}} = \mu_{\text{EIG}}$. □

3.3.2 Generalized eigenvalue relaxation

In this section, we propose a new type of quadratic relaxation for (3.1) which improves the bounds given by the eigenvalue relaxation by incorporating information from the equality constraints. We start by considering the following reformulation of (3.1):

$$\begin{array}{ll}\min\limits_{x\in\mathcal{X}} & x^TQx + q^Tx + \alpha_g \sum\limits_{i=1}^{n} x_i^2 - \alpha_g \sum\limits_{i=1}^{n} x_i^2 + \alpha_g\|Ax-b\|^2 \\ \text{s.t.} & x_i \in \mathbb{Z},\ \ \forall i \in J\end{array} \tag{3.31}$$

where α_g is a nonnegative scalar. As done in §3.3.1, we can obtain a quadratic relaxation of (3.1) by dropping the integrality conditions from (3.31) and using the concave envelope of x_i^2 over $[l_i, u_i]$:

$$\min_{x\in\mathcal{X}} \quad x^TQx + q^Tx + \alpha_g \sum_{i=1}^{n} x_i^2 - \alpha_g \sum_{i=1}^{n} ((l_i+u_i)x_i - l_iu_i) + \alpha_g\|Ax-b\|^2 \tag{3.32}$$

This relaxation can be equivalently written as:

$$\min_{x\in\mathcal{X}} \quad x^TQ_{\alpha_g}x + q_{\alpha_g}^Tx + k_{\alpha_g} \tag{3.33}$$

where $Q_{\alpha_g} = Q + \alpha_g(I_n + A^TA)$, $q_{\alpha_g} = q - \alpha_g(l + u + 2A^Tb)$, and $k_{\alpha_g} = \alpha_g(l^Tu + b^Tb)$.

In the following proposition, we provide a condition for choosing α_g which ensures that the above problem is a convex relaxation of (3.1).

Proposition 3.3. *Let $\alpha_g \geq -\min(0, \lambda_{\min}(Q, I_n + A^TA))$ in* (3.33)*. Then,* (3.33) *is a convex quadratic program.*

Proof. To establish the convexity of (3.33), it suffices to verify that $Q_{\alpha_g} = Q + \alpha_g(I_n + A^TA)$ is positive semidefinite. From the definition of the Rayleigh quotient for the generalized eigenvalue pair $(Q, I_n + A^TA)$ in (3.21) we obtain:

$$\lambda_{\min}(Q, I_n + A^TA) \leq \frac{x^TQx}{x^T\left(I_n + A^TA\right)x},\ \ \forall x \neq 0 \tag{3.34}$$

which using the positive definiteness of $(I_n + A^TA)$ can be equivalently written as:

$$x^TQ_{\alpha_g}x \geq \left(\alpha_g + \lambda_{\min}(Q, I_n + A^TA)\right)x^T\left(I_n + A^TA\right)x,\ \ \forall x \neq 0. \tag{3.35}$$

It is readily verified that $Q_{\alpha_g} \succcurlyeq 0$ for $\alpha_g \geq -\min(0, \lambda_{\min}(Q, I_n + A^TA))$. □

From Proposition 3.3, it follows that $\alpha_g = -\min(0, \lambda_{\min}(Q, I_n + A^TA))$ provides the tightest convex relaxation of the form (3.33) for which $Q_{\alpha_g} \succcurlyeq 0$. We refer to this convex

relaxation as the *generalized eigenvalue relaxation* of (3.1). Next, we show that this relaxation is at least as tight as the eigenvalue relaxation.

Proposition 3.4. *Suppose that* $\alpha_e = -\min(0, \lambda_{\min}(Q))$ *in* (3.26) *and* $\alpha_g = -\min(0, \lambda_{\min}(Q, I_n + A^T A))$ *in* (3.33). *Denote by* μ_{EIG} *and* μ_{GEIG} *the optimal objective function values in (3.26) and (3.33), respectively. Then,* $\mu_{\text{GEIG}} \geq \mu_{\text{EIG}}$.

Proof. To prove that $\mu_{\text{GEIG}} \geq \mu_{\text{EIG}}$, it suffices to show that $\alpha_g \leq \alpha_e$. We will use the definition of the Rayleigh quotient in (3.21). We consider the following cases:

(i) $\lambda_{\min}(Q) \geq 0$. This implies that $x^T Q x \geq 0,\ \forall x \in \mathbb{R}^n \setminus \{0\}$. Moreover, it is clear that $x^T(I_n + A^T A)x > 0,\ \forall x \in \mathbb{R}^n \setminus \{0\}$. Then, from (3.21) it follows that $\lambda_{\min}(Q, I_n + A^T A) \geq 0$. Hence, $\alpha_e = \alpha_g = 0$, which implies that $\mu_{\text{GEIG}} = \mu_{\text{EIG}}$.

(ii) $\lambda_{\min}(Q) < 0$. This implies that $\exists x \in \mathbb{R}^n$ such that $x^T Q x < 0$. From (3.21), it follows that $\lambda_{\min}(Q, I_n + A^T A) < 0$. Then, it is clear that $\alpha_e = -\lambda_{\min}(Q)$ and $\alpha_g = -\lambda_{\min}(Q, I_n + A^T A)$. Define the set $D = \{x \in \mathbb{R}^n : x \neq 0, x^T Q x < 0\}$. Clearly, D is nonempty. It is easy to verify that the minimum in (3.21) occurs for $x \in D$. Combining this observation with $x^T(I + A^T A)x \geq x^T x$, we obtain

$$\frac{x^T Q x}{x^T (I_n + A^T A)\, x} \geq \frac{x^T Q x}{x^T x}, \quad \forall x \in D. \tag{3.36}$$

This proves that $\lambda_{\min}(Q, I_n + A^T A) \geq \lambda_{\min}(Q)$, which implies that $\alpha_g \leq \alpha_e$, and $\mu_{\text{GEIG}} \geq \mu_{\text{EIG}}$.

□

Note the idea of using information from the equality constraints to convexify objective functions containing nonconvex quadratic terms has been considered before in the literature. In particular, this idea has been exploited in the context of the QCR methods discussed in §3.2.3.3.

Even though our approach also relies on the use of information from the equality constraints to convexify the objective function of (3.1), it differs from the QCR techniques considered in [14, 15, 16, 17] in three important ways. First, our technique does not seek the development of a reformulation of the original problem but instead the construction of cheap quadratic relaxations which can be incorporated into a branch-and-bound framework. Second, under our approach, at a given node of the branch-and-bound tree, we update the perturbation parameters used to construct these quadratic relaxations. This is done by solving the eigenvalue or generalized eigenvalue problems involving the submatrices of Q and $I_n + A^T A$ obtained after eliminating the rows and columns corresponding to the variables that have been fixed. This update results in tighter bounds, and as shown in §3.6.3–3.6.5, it can have a very significant impact on the performance of branch-and-bound algorithms, especially in the binary case, in which our relaxations can be used in conjunction with the branching strategy introduced in §3.4. By contrast, in the QCR methods, the perturbation parameters used to convexify the problem are calculated only once, prior to the initialization of the branch-and-bound tree, and are not updated during the execution of the branch-and-bound algorithm. Third, in our method, the perturbation parameters can be obtained by solving an eigenvalue or generalized eigenvalue problem, which is often inexpensive. Under the QCR approaches, calculating the perturbation parameters involves the solution of an SDP, which is more computationally expensive.

Observe also that, unlike our approach, the separable programming and d.c. programming techniques described in §3.2.3.1 and §3.2.3.2 do not incorporate information from the equality constraints to improve the bound of the resulting relaxations.

We next show that the generalized eigenvalue relaxation is equivalent to the following SDP:

$$\min_{x \in \mathcal{X}, X} \ \langle Q, X \rangle + q^T x \tag{3.37a}$$

$$\text{s.t. } X - xx^T \succcurlyeq 0 \tag{3.37b}$$

$$\langle I_n, X\rangle - (l+u)^T x + l^T u + \langle A^T A, X\rangle - \left(2A^T b\right)^T x + b^T b \leq 0 \tag{3.37c}$$

The SDP in (3.37) can be obtained from (3.9) by aggregating the constraints in (3.7e) and (3.9c), and dropping the redundant inequalities (3.7c) and (3.7d). However, unlike the SDP (3.9), the optimal objective of SDP (3.37) can be obtained by solving the QP (3.33).

Proposition 3.5. *Suppose that the matrix Q is indefinite. Assume that $\alpha_g = -\lambda_{\min}(Q, I_n + A^T A)$ in* (3.33)*. Denote by μ_{GEIG} and $\mu_{\text{SDP\_GEIG}}$ the optimal objective function values in (3.33) and (3.37), respectively. Then, $\mu_{\text{GEIG}} = \mu_{\text{SDP\_GEIG}}$.*

Proof. We will rely on strong duality holding for (3.37) and follow the same line of arguments used in the proof of Proposition of 3.2. We start by showing that (3.37) admits a strictly feasible solution. Let $\bar{x} \in \mathbb{R}^n$ be a vector such that $A\bar{x} = b$, $C\bar{x} < d$, and $l < \bar{x} < u$. Recall that the concave envelope of x_i^2 over $[l_i, u_i]$ is given by $(l_i + u_i)x_i - l_i u_i$. Since $l_i < \bar{x}_i < u_i, \forall i = 1, \ldots, n$, it follows that $(l_i + u_i)x_i - l_i u_i - \bar{x}_i^2 > 0, \forall i = 1, \ldots, n$. Define $\epsilon := (l_1 + u_1)x_1 - l_1 u_1 - \bar{x}_1^2$. Clearly, there exists $\delta \in \mathbb{R}$ such that $0 < \delta < \epsilon$. Let $\bar{X} \in \mathbb{S}^n$ be the matrix satisfying:

$$\begin{aligned}
\bar{X}_{11} &= \frac{(l_1 + u_1)\bar{x}_1 - l_1 u_1 + \Phi_{11}\bar{x}_1^2 - \delta}{1 + \Phi_{11}} \\
\bar{X}_{ii} &= \frac{(l_i + u_i)\bar{x}_i - l_i u_i + \Phi_{ii}\bar{x}_i^2}{1 + \Phi_{ii}},\ i = 2, \ldots, n, \\
\bar{X}_{ij} &= \bar{X}_{ji} = \bar{x}_i \bar{x}_j,\ i = 1, \ldots, n,\ j = i+1, \ldots, n
\end{aligned} \tag{3.38}$$

where Φ_{ii} denotes the i-th diagonal element of $A^T A$. Then, it is simple to check that (3.37c) is strictly satisfied by $(\bar{x}, \bar{X})$. Define $\hat{X} := \bar{X} - \bar{x}\bar{x}^T$. It is clear that $\hat{X}$ is diagonal with entries:

$$\begin{aligned}
\hat{X}_{11} &= \frac{(l_1 + u_1)\bar{x}_1 - l_1 u_1 - \bar{x}_1^2 - \delta}{1 + \Phi_{11}} \\
\hat{X}_{ii} &= \frac{(l_i + u_i)\bar{x}_i - l_i u_i - \bar{x}_i^2}{1 + \Phi_{ii}},\ i = 2, \ldots, n,
\end{aligned} \tag{3.39}$$

It is easy to verify that $\hat{X}_{ii} > 0$, $\forall i = 1, \ldots, n$. It follows that $\hat{X}$ is positive definite and $(\bar{x}, \bar{X})$ is a strictly feasible solution to (3.37). Therefore, Slater's condition is satisfied

by (3.37), which implies that strong duality holds and the optimal value of the dual problem is attained.

Now, we consider the dual of (3.37). By Proposition 3.1, this dual is given by:

$$\max_{\alpha_g \in \mathbb{R}_{\geq 0}: Q_{\alpha_g} \succcurlyeq 0} \left\{ \min_{x \in \mathcal{X}} \quad x^T Q_{\alpha_g} x + q_{\alpha_g}^T x + k_{\alpha_g} \quad \right\} \tag{3.40}$$

where α_g is the multiplier for the constraint (3.37c), $Q_{\alpha_g} = Q + \alpha_g(I_n + A^T A)$, $q_{\alpha_g} = q - \alpha_g(l + u + 2A^T b)$, and $k_{\alpha_g} = \alpha_g(l^T u + b^T b)$.

Since Q is indefinite, Proposition 3.3 implies that $Q_{\alpha_g} \succcurlyeq 0$ for $\alpha_g \geq -\lambda_{\min}(Q, I_n + A^T A)$. Then, it is easy to verify that the maximum of (3.40) is attained when $\alpha_g = -\lambda_{\min}(Q, I + A^T A)$, which implies that $\mu_{\text{SDP\_GEIG}} = \mu_{\text{GEIG}}$.

□

3.3.3 Eigenvalue relaxation in the nullspace of the equality constraints

In this section, we consider another convex quadratic relaxation of (3.1) which also incorporates information from the equality constraints in order to convexify the objective function. This relaxation can be formulated as:

$$\min_{x \in \mathcal{X}} \quad x^T Q_{\alpha_z} x + q_{\alpha_z}^T x + k_{\alpha_z} \tag{3.41}$$

where $Q_{\alpha_z} = Q + \alpha_z I_n$, $q_{\alpha_z} = q - \alpha_z(l + u)$, $k_{\alpha_z} = \alpha_z l^T u$, and α_z is a nonnegative scalar. As discussed in §3.3.1, we must select a suitable α_z in order to ensure that (3.41) is a convex relaxation of (3.1). As indicated previously, one such α_z can be determined by using the smallest eigenvalue of the matrix Q. However, as we show in the next proposition, there exists another method for constructing such α_z which makes use of the nullspace of the equality constraints of (3.1).

Proposition 3.6. *Denote by Z an orthonormal basis for the nullspace of the matrix A. Let $\alpha_z \geq -\min(0, \lambda_{\min}(Z^T Q Z))$ in* (3.41)*. Then,* (3.41) *is a convex quadratic program when restricted to the nullspace of the matrix A.*

Proof. Let $\mathcal{H} = \{x \in \mathbb{R}^n \mid Ax = b\}$, and denote by r the rank of A. It is clear than any point satisfying $Ax = b$ can be expressed as $x = x_h + Zx_z$, where $x_h \in \mathcal{H}$, $x_z \in \mathbb{R}^{n-r}$, and $Z \in \mathbb{R}^{n \times n-r}$. By using this transformation, we can write (3.41) as:

$$\begin{array}{ll} \min\limits_{x_z} & (x_h + Zx_z)^T Q_{\alpha_z} (x_h + Zx_z) + q_{\alpha_z}^T (x_h + Zx_z) + k_{\alpha_z} \\ \text{s.t.} & C (x_h + Zx_z) \leq d \\ & l \leq (x_h + Zx_z) \leq u. \end{array} \tag{3.42}$$

It is easily verified that (3.42) is convex for all $\alpha_z \geq -\min(0, \lambda_{\min}(Z^T Q Z))$. □

From Proposition 3.6, it follows that the tightest relaxation of the form (3.41) is obtained by setting $\alpha_z = -\min(0, \lambda_{\min}(Z^T Q Z))$. We refer to this convex relaxation of (3.1) as the *eigenvalue relaxation in the nullspace of* A. In the following proposition, we show that this relaxation is at least as tight as the generalized eigenvalue relaxation.

Proposition 3.7. *Assume that* $\alpha_g = -\min(0, \lambda_{\min}(Q, I_n + A^T A))$ *in* (3.33) *and let* $\alpha_z = -\min(0, \lambda_{\min}(Z^T Q Z))$ *in* (3.41). *Let* μ_{GEIG} *and* μ_{EIGZ} *denote the optimal objective function values in* (3.33) *and* (3.41), *respectively. Then,* $\mu_{\text{EIGZ}} \geq \mu_{\text{GEIG}}$.

Proof. To prove that $\mu_{\text{EIGZ}} \geq \mu_{\text{GEIG}}$, it suffices to show that $\alpha_z \leq \alpha_g$. Similar to (3.21), the smallest eigenvalue of $Z^T Q Z$ can be expressed as:

$$\lambda_{\min}(Z^T Q Z) = \min_{x \neq 0, Ax = 0} \frac{x^T Q x}{x^T x} = \min_{x \neq 0, Ax = 0} \frac{x^T Q x}{x^T (I_n + A^T A) x} \tag{3.43}$$

where for the second equality we used the fact that the minimization is over vectors x that lie in the null space of A. The restriction of vectors x to the null space of A also implies that $\lambda_{\min}(Z^T Q Z) \geq \lambda_{\min}(Q, I_n + A^T A)$. This is easily seen by noting that the Rayleigh quotient expression for the generalized eigenvalue of the pair $(Q, I_n + A^T A)$ in (3.21) is over a larger domain. Hence, $\alpha_z \leq \alpha_g$, and $\mu_{\text{EIGZ}} \geq \mu_{\text{GEIG}}$.

□

From Proposition 3.7, it follows that the eigenvalue relaxation in the nullspace of A can be potentially tighter than the generalized eigenvalue relaxation. However, an important

drawback of this relaxation is the fact that it requires the computation of a basis Z for the nullspace of A, which can be computationally expensive. Therefore, an important question that arises in this context is whether we can obtain a good approximation of $\lambda_{\min}(Z^TQZ)$ without having to explicitly compute the basis Z. This question is addressed by the following proposition.

Proposition 3.8. *Let δ be a real scalar. Then, the following hold:*

(a) If the matrix Q is indefinite, $\lambda_{\min}(Q, I_n + \delta A^TA)$ is a strictly increasing function of δ for $\delta \geq 1$.

(b) $\lim_{\delta\to\infty} \lambda_{\min}(Q, I_n + \delta A^TA) = \min(0, \lambda_{\min}(Z^TQZ))$.

Proof. We start with the proof of (a). Let $\delta_1, \delta_2 \in \mathbb{R}$ be two scalars such that $\delta_2 > \delta_1 \geq 1$. Define the set $D = \{x \in \mathbb{R}^n : x \neq 0, x^TQx < 0\}$. Since the matrix Q is indefinite by assumption, it is clear that $D \neq \emptyset$. From the definition of the set D, it is easy to check that the following inequality holds:

$$\frac{x^TQx}{x^T\left(I_n + \delta_2 A^TA\right)x} > \frac{x^TQx}{x^T\left(I_n + \delta_1 A^TA\right)x}, \quad \forall x \in D \tag{3.44}$$

Using the definition of the Rayleigh quotient in (3.21), $D \neq \emptyset$ and (3.44), it is simple to verify that $\lambda_{\min}(Q, I_n + \delta_2 A^TA) > \lambda_{\min}(Q, I_n + \delta_1 A^TA)$ which proves (a).

To prove (b), consider the Rayleigh quotient definition in (3.21) for the pair $(Q, I_n + \delta A^TA)$. Let $x = y+z$, where $y, z \in \mathbb{R}^n$ are orthogonal vectors which belong to the row space and nullspace of the matrix A, respectively. Then, by using this transformation in (3.21), we have:

$$\lim_{\delta\to\infty} \lambda_{\min}(Q, I_n + \delta A^TA) = \lim_{\delta\to\infty} \min_{(y+z)\neq 0} \frac{(y+z)^TQ(y+z)}{(y+z)^T(y+z) + \delta y^TA^TAy}. \tag{3.45}$$

To determine the limit in (3.45), we consider the following cases:

(i) $y \neq 0$. In this case, we obtain:

$$\min_{(y+z)\neq 0} \lim_{\delta\to\infty} \frac{(y+z)^T Q(y+z)}{(y+z)^T(y+z) + \delta y^T A^T A y} = 0. \tag{3.46}$$

(ii) $y = 0$. In this case, (3.45) reduces to:

$$\lim_{\delta\to\infty} \min_{z\neq 0} \frac{z^T Q z}{z^T z} = \lim_{\delta\to\infty} \min_{z\neq 0, Az=0} \frac{z^T Q z}{z^T z} = \lambda_{\min}(Z^T Q Z). \tag{3.47}$$

Then, it follows that $\lim_{\delta\to\infty} \lambda_{\min}(Q, I_n + \delta A^T A) = \min(0, \lambda_{\min}(Z^T Q Z))$. □

Proposition 3.8 has very important consequences since it suggests we can approximate the bound given by the eigenvalue relaxation in the nullspace of A by solving the following quadratic program for a sufficiently large value of δ:

$$\min_{x\in\mathcal{X}} \quad x^T Q x + q^T x + \alpha(\delta)(x^T x - (l+u)^T x + l^T u) + \alpha(\delta)\cdot\delta\cdot\|Ax - b\|^2 \tag{3.48}$$

where $\alpha(\delta) = -\lambda_{\min}(Q, I_n + \delta A^T A)$. Note that, for $\delta = 1$, (3.48) corresponds to the generalized eigenvalue relaxation introduced in §3.3.2.

Since $\lambda_{\min}(Q, I_n + \delta A^T A)$ is a strictly increasing function of δ for $\delta \geq 1$, Proposition 3.8 implies that as δ is increased, $\alpha(\delta)$ will converge to either 0 or $-\lambda_{\min}(Z^T Q Z)$. The case in which $\alpha(\delta)$ converges to 0 is particularly interesting since it indicates that $\lambda_{\min}(Z^T Q Z) \geq 0$, and the continuous relaxation of (3.1) is convex when restricted to the nullspace of A. Note that $\lambda_{\min}(Q) < 0$ does not necessarily imply that $\lambda_{\min}(Z^T Q Z) < 0$, and as a result, the continuous relaxation of (3.1) may be convex when restricted to the nullspace of A, even if it is nonconvex in the space of the original problem variables.

Observe that the quadratic term $\alpha(\delta)\cdot\delta\cdot\|Ax - b\|^2$ vanishes for any solution x feasible in (3.48). This term is included in the objective function of (3.48) to ensure that the matrix $Q + \alpha(\delta)(I_n + \delta A^T A)$ is positive semidefinite. However, this term need not be included for (3.48) to be convex. In fact, Proposition 3.6 implies that (3.48) is a convex quadratic program for $\alpha(\delta) \geq -\min(0, \lambda_{\min}(Z^T Q Z))$. From the definition of $\alpha(\delta)$ and Proposition 3.8, it

follows that $\alpha(\delta) \geq -\min(0, \lambda_{\min}(Z^T Q Z))$ holds for any $\delta \geq 1$. As a result, the quadratic term $\alpha(\delta) \cdot \delta \cdot \|Ax - b\|^2$ can be dropped from the objective function of (3.48), which simplifies this relaxation to:

$$\min_{x \in \mathcal{X}} \quad x^T Q x + q^T x + \alpha(\delta)(x^T x - (l+u)^T x + l^T u) \tag{3.49}$$

This simplification has two significant practical advantages. First, it allows us to preserve the sparsity pattern defined by the quadratic matrix Q of the original problem (3.1). Second, it prevents the relaxation from becoming ill-conditioned since δ does not figure in the objective function of (3.49) and is only used to determine $\alpha(\delta)$. Note that we can use a relatively simple iterative procedure in order to determine a value of δ which leads to a good approximation of the bound provided by the eigenvalue relaxation in the nullspace of A. We detail such procedure in §3.5.

By considering a quadratic relaxation of the form (3.49), there is no need to project onto the nullspace of A. This is particularly advantageous in the context of the branching variable selection rules that we introduce in §3.4, since the branching decisions are easier to interpret in the space of the original problem variables.

We finish this section by showing that the eigenvalue relaxation in the null space of A is equivalent to the following SDP:

$$\begin{aligned}
\min_{x \in \mathcal{X}, X} \quad & \langle Q, X \rangle + q^T x && \text{(3.50a)} \\
\text{s.t.} \quad & X - x x^T \succcurlyeq 0 && \text{(3.50b)} \\
& \langle I_n, X \rangle - (l+u)^T x + l^T u \leq 0 && \text{(3.50c)} \\
& \langle A^T A, X \rangle - \left(2A^T b\right)^T x + b^T b = 0 && \text{(3.50d)}
\end{aligned}$$

The SDP in (3.50) can be obtained from (3.9) by aggregating the constraints in (3.7e), and dropping the redundant inequalities (3.7c) and (3.7d). However, unlike the SDP (3.9), the optimal objective of SDP (3.50) can be obtained by solving the QP (3.41).

Proposition 3.9. *Suppose that the matrix Z^TQZ is indefinite. Assume that $\alpha_z = -\lambda_{\min}(Z^TQZ)$ in* (3.41)*. Denote by μ_{EIGZ} and $\mu_{\text{SDP\_EIGZ}}$ the optimal objective function values in* (3.41) *and* (3.50)*, respectively. Then, $\mu_{\text{EIGZ}} = \mu_{\text{SDP\_EIGZ}}$.*

Proof. Note that unlike the SDPs (3.27) and (3.37), (3.50) does not admit a strictly feasible solution. To illustrate this, we note that, for any point x satisfying $Ax = b$, the constraint (3.50d) can be equivalently written as follows:

$$\langle A^TA, X\rangle - \left(2A^Tb\right)^T x + b^Tb + \langle A^TA, xx^T\rangle - \langle A^TA, xx^T\rangle = 0 \tag{3.51a}$$
$$\implies \langle A^TA, X - xx^T\rangle + (Ax-b)^T(Ax-b) = 0 \tag{3.51b}$$
$$\implies \langle A^TA, X - xx^T\rangle = 0 \tag{3.51c}$$

which implies that $X - xx^T$ cannot be positive definite for the pairs (x, X) that are feasible in (3.50). It follows that we cannot apply the strong duality theorem to (3.50). As a result, the proof of this proposition relies on different arguments from those used in the proofs of Propositions 3.2 and 3.5. We proceed in two steps:

(i) We show that the dual problem of (3.50) is equivalent to (3.41). By weak duality of (3.50), this implies that $\mu_{\text{SDP\_EIGZ}} \geq \mu_{\text{EIGZ}}$.

(ii) We construct a feasible solution for (3.50) which attains the same objective function value as an optimal solution of (3.41). This completes the proof by showing that $\mu_{\text{SDP\_EIGZ}} \leq \mu_{\text{EIGZ}}$.

To prove (i), we use Proposition 3.1 to write the dual of (3.50) as:

$$\max_{\alpha_z\in\mathbb{R}_{\geq 0},\beta_z\in\mathbb{R}: Q_{\alpha_z,\beta_z}\succcurlyeq 0} \left\{ \min_{x\in\mathcal{X}} \quad x^TQ_{\alpha_z,\beta_z}x + q_{\alpha_z,\beta_z}^Tx + k_{\alpha_z,\beta_z} \quad \right\} \tag{3.52}$$

where α_z and β_z are multipliers for (3.50c) and (3.50d), respectively, $Q_{\alpha_z,\beta_z} = Q + \alpha_z I_n + \beta_z A^TA$, $q_{\alpha_z,\beta_z} = q - \alpha_z(l+u) - 2\beta_z A^Tb$, and $k_{\alpha_z,\beta_z} = \alpha_z l^Tu + \beta_z b^Tb$. Let $\delta_z = \beta_z/\alpha_z$. By substituting $\beta_z = \delta_z\alpha_z$ in (3.52), the dual becomes:

$$\max_{\alpha_z \in \mathbb{R}_{\geq 0}, \delta_z \in \mathbb{R}: Q_{\alpha_z,\delta_z} \succcurlyeq 0} \left\{ \min_{x \in \mathcal{X}} \quad x^T Q_{\alpha_z,\delta_z} x + q_{\alpha_z,\delta_z}^T x + k_{\alpha_z,\delta_z} \quad \right\} \tag{3.53}$$

where $Q_{\alpha_z,\delta_z} = Q + \alpha_z(I_n + \delta_z A^T A)$, $q_{\alpha_z,\delta_z} = q - \alpha_z(l + u + 2\delta_z A^T b)$, and $k_{\alpha_z,\delta_z} = \alpha_z(l^T u + \delta_z b^T b)$. Note that the quadratic term $\alpha_z \delta_z \|Ax - b\|^2$ vanishes for any x feasible in the inner minimization problem. As a result, (3.53) can be posed as:

$$\max_{\alpha_z \in \mathbb{R}_{\geq 0}, \delta_z \in \mathbb{R}: Q_{\alpha_z,\delta_z} \succcurlyeq 0} \left\{ \min_{x \in \mathcal{X}} \quad x^T (Q + \alpha_z I_n) x + (q - \alpha_z(l + u))^T x + \alpha_z l^T u \quad \right\} \tag{3.54}$$

Since $Z^T Q Z$ is indefinite, Q is indefinite as well. From Proposition 3.3, it follows that, for a given value of δ_z, $Q_{\alpha_z,\delta_z} \succcurlyeq 0$ when $\alpha_z \geq -\lambda_{\min}(Q, I_n + \delta_z A^T A)$. Then, by using the fact that $\lambda_{\min}(Z^T Q Z) < 0$ and Proposition 3.8, it is easy to verify that the maximum of (3.54) is attained when $\alpha_z = -\lim_{\delta_z \to \infty} \lambda_{\min}(Q, I_n + \delta_z A^T A) = -\lambda_{\min}(Z^T Q Z)$. This implies that the dual of (3.50) is equivalent to (3.41), and by weak duality of (3.50), it follows that $\mu_{\text{SDP\_EIGZ}} \geq \mu_{\text{EIGZ}}$.

Next, we prove (ii). Let $\hat{x}$ denote the optimal solution of (3.41). Define $\hat{X} = \hat{x}\hat{x}^T + \gamma Z v (Z v)^T$, where $\gamma = (l+u)^T \hat{x} - l^T u - \hat{x}^T \hat{x}$, and v denotes the eigenvector corresponding to the smallest eigenvalue of the matrix $Z^T Q Z$. We first show that $(\hat{x}, \hat{X})$ is feasible in (3.50). By definition, $\hat{x} \in \mathcal{X}$. Consider (3.50b). Recall that the concave envelope of x_i^2 over $[l_i, u_i]$ is given by $(l_i + u_i)x_i - l_i u_i$. As a result, it is clear that each term $(l_i + u_i)\hat{x}_i - l_i u_i - \hat{x}_i^2$ is nonnegative, which in turn implies that $\gamma \geq 0$. Moreover, since the matrix $Zv(Zv)^T \succcurlyeq 0$, it follows that $\hat{X} - \hat{x}\hat{x}^T \succcurlyeq 0$.

Consider (3.50c) and (3.50d). Substituting $(\hat{x}, \hat{X})$ into (3.50c), we obtain:

$$\begin{aligned} &\langle I_n, \hat{x}\hat{x}^T + \gamma Z v (Z v)^T \rangle - (l+u)^T \hat{x} + l^T u \\ &= \hat{x}^T \hat{x} + \gamma v^T Z^T Z v - (l+u)^T \hat{x} + l^T u = \hat{x}^T \hat{x} + \gamma - (l+u)^T \hat{x} + l^T u = 0. \end{aligned}$$

Similarly, substituting $(\hat{x}, \hat{X})$ into (3.50d) yields:

$$\begin{aligned} &\langle A^T A, \hat{x}\hat{x}^T + \gamma Z v (Z v)^T \rangle - \left(2A^T b\right)^T \hat{x} + b^T b \\ &= \hat{x}^T A^T A \hat{x} - \left(2A^T b\right)^T \hat{x} + b^T b + \gamma v^T Z^T A^T A Z v = (A\hat{x} - b)^T (A\hat{x} - b) = 0. \end{aligned}$$

Let $f(x, X)$ be the objective function of (3.50). The value of f at $(\hat{x}, \hat{X})$ is:

$$\begin{aligned} f(\hat{x}, \hat{X}) &= \langle Q, \hat{x}\hat{x}^T + \gamma Zv(Zv)^T \rangle + q^T\hat{x} \\ &= \hat{x}^T Q\hat{x} + \gamma v^T Z^T QZv + q^T\hat{x} \\ &= \hat{x}^T Q\hat{x} - \gamma\alpha_z + q^T\hat{x} \\ &= \hat{x}^T(Q + \alpha_z I_n)\hat{x} + (q - \alpha_z(l+u))^T\hat{x} + \alpha_z l^T u = \mu_{\text{EIGZ}} \end{aligned} \tag{3.55}$$

where we have relied on the fact that $v^T Z^T QZv = -\alpha_z = \lambda_{\min}(Z^T QZ)$. Since $(\hat{x}, \hat{X})$ is feasible in (3.50), from (3.55) it follows that $\mu_{\text{SDP\_EIGZ}} \leq \mu_{\text{EIGZ}}$. Hence, $\mu_{\text{SDP\_EIGZ}} = \mu_{\text{EIGZ}}$. □

3.3.4 Further insights into the proposed quadratic relaxations

The quadratic relaxations introduced in this chapter can be derived through the following four-step recipe:

(R1) identify a (possibly empty) set $\mathcal{J}$ of quadratic functions of the form $f_j(x) = x^T S_j x + s_j^T x + \eta_j$, where $S_j \in \mathbb{S}^n, s_j \in \mathbb{R}^n, \eta_j \in \mathbb{R}$, such that $f_j(x) = 0$ for $x \in \Omega := \{x \in \mathbb{R}^n \mid Ax = b\}$;

(R2) construct an initial relaxation for (3.1) as

$$\min_{x\in\mathcal{X}} \quad x^T Qx + q^T x + \alpha(x^T x - (l+u)^T x + l^T u) + \textstyle\sum_{j\in\mathcal{J}} \beta_j f_j(x) \tag{3.56}$$

where $\alpha \in \mathbb{R}_{\geq 0}, \beta_j \in \mathbb{R}$, such that $Q + \alpha I_n + \sum_{j\in\mathcal{J}} \beta_j S_j \succcurlyeq 0$;

(R3) find α^*, β^* such that the bound given by the relaxation (3.56) is maximized

$$(\alpha^*, \beta^*) = \arg \max_{\alpha\in\mathbb{R}_{\geq 0}, \beta\in\mathbb{R}^{|\mathcal{J}|}: Q_{\alpha,\beta}\succcurlyeq 0} \left\{ \min_{x\in\mathcal{X}} \quad x^T Q_\alpha x + q_\alpha^T x + k_\alpha \quad \right\} \tag{3.57}$$

where $Q_\alpha = Q + \alpha I_n$, $Q_{\alpha,\beta} = Q_\alpha + \sum_{j\in\mathcal{J}} \beta_j S_j$, $q_\alpha = q - \alpha(l+u)$, $k_\alpha = \alpha l^T u$, and β is the $|\mathcal{J}|$-dimensional vector whose entries are the parameters β_j;

(R4) obtain the relaxation

$$\min_{x \in \mathcal{X}} \quad x^T Q_{\alpha^*} x + q_{\alpha^*}^T x + k_{\alpha^*}. \tag{3.58}$$

Observe that the parameters β_j are not present in the objective function of the inner minimization problem in (3.57) and the objective function in (3.58) since $f_j(x) = 0$ for $x \in \mathcal{X} \subset \Omega$ (due to (R1)). The three spectral relaxations presented in §3.3.1–3.3.3 can be identified with (3.57) by noting that:

- $\mathcal{J} = \emptyset$, $\alpha^* = -\min(0, \lambda_{\min}(Q))$ for the eigenvalue relaxation (3.26);
- $\mathcal{J} = \{1\}$, $f_1(x) = \sum_{i=1}^m (A_{i\cdot}x - b_i)^2$, $\alpha^* = -\min(0, \lambda_{\min}(Q, I_n + A^T A))$, $\beta_1^* = \alpha^*$ for the generalized eigenvalue relaxation (3.33). Note that in this case a further restriction that $\beta_1 = \alpha$ is imposed in (3.57); and
- $\mathcal{J} = \{1\}$, $f_1(x) = \sum_{i=1}^m (A_{i\cdot}x - b_i)^2$, $\alpha^* = -\min(0, \lambda_{\min}(Z^T Q Z))$ and $\beta_1^* = +\infty$ for the eigenvalue relaxation on the nullspace of A (3.41).

From Propositions 3.4 and 3.7 we know that the lower bound obtained from the eigenvalue relaxation in the nullspace of A (3.41) is at least as large as those provided by the other spectral relaxations. Further, the computation of α^* can be done efficiently.

The recipe (R1)-(R4) is preferable from a computational standpoint since the resulting relaxation is a quadratic program inheriting the sparsity of the problem. However, the step (R1) allows for other choice for the functions $f_j(x)$ that have been considered in the literature (see Faye and Roupin [31]). Some examples for the functions satisfying (R1) are [31]: $(x_j(A_{i\cdot}x - b_i))$, $((A_{j\cdot}x - b_j)(A_{i\cdot}x - b_i))$, $\left(x^T A_{j\cdot}^T A_{i\cdot} x - b_j b_i\right)$. This naturally raises the question: *Can we improve on the bound provided by* (3.41) *when restricted to the class of relaxations in* (3.57)? In the rest of the section, we show that we cannot improve on the bound provided by the eigenvalue relaxation on the nullspace of A (3.41). Thus, establishing that (3.41) is the best among the class of relaxations in (3.57).

We begin by recalling the properties of functions satisfying (R1).

Proposition 3.10. *Let $f(x) = x^T Sx + s^T x + \eta$ be a quadratic function. Then, $f(x) = 0$ for all $x \in \Omega := \{x \in \mathbb{R}^n \mid Ax = b\}$ if and only if $S = A^T W^T + WA$, $s = A^T \nu - 2Wb$, $\eta = -b^T \nu$ for some $W \in \mathbb{R}^{n \times m}$ and $\nu \in \mathbb{R}^m$.*

Proof. This follows from Theorem 1 in [31]. □

Following Proposition 3.10, we assume without loss of generality that $S_j = A^T W_j^T + W_j A$ for some $W_j \in \mathbb{R}^{n \times m}$ in the rest of this section.

We will compare the relaxations in the class (3.57) with the eigenvalue relaxation in the nullspace of A (3.41) through the respective SDP formulations. To this end, consider the SDP:

$$\begin{aligned} \min_{x \in \mathcal{X}, X} \quad & \langle Q, X \rangle + q^T x && \text{(3.59a)} \\ \text{s.t.} \quad & X - xx^T \succcurlyeq 0 && \text{(3.59b)} \\ & \langle I_n, X \rangle - (l+u)^T x + l^T u \leq 0 && \text{(3.59c)} \\ & \langle S_j, X \rangle + s_j^T x + \eta_j = 0, \ j \in \mathcal{J}. && \text{(3.59d)} \end{aligned}$$

The next proposition shows that SDP (3.59) is the dual of (3.57).

Proposition 3.11. *Let $\mathcal{J} \neq \emptyset$ be a set of quadratic functions satisfying (R1). The dual of the SDP* (3.59) *is given by* (3.57)*.*

Proof. By dualizing the constraints (3.59c) and (3.59d) with the multipliers $\alpha \in \mathbb{R}_{\geq 0}$ and $\beta_j \in \mathbb{R}, j \in \mathcal{J}$, respectively, we can use Proposition 3.1 to obtain the claim.

□

The next result shows that the feasible set of the SDP (3.50) is in general a subset of the feasible set of the SDP (3.59). Further, we provide conditions on the choice of quadratic functions in $\mathcal{J}$ so that equality holds.

Proposition 3.12. *Let $\mathcal{F}_{\text{SDP\_EIGZ}}$ and $\mathcal{F}_{\text{SDP\_EIGJ}}$ denote the feasible regions of the SDPs in* (3.50) *and* (3.59)*, respectively. Then, the following holds:*

(i) $\mathcal{F}_{\text{SDP\_EIGZ}} \subseteq \mathcal{F}_{\text{SDP\_EIGJ}}$.

(ii) If $\exists\, \omega_j, j \in \mathcal{J}$ *such that* $\sum_{j\in\mathcal{J}} \omega_j W_j = A^T$ *then* $\mathcal{F}_{\text{SDP\_EIGZ}} = \mathcal{F}_{\text{SDP\_EIGJ}}$.

Proof. We start by proving (i). Recall from (3.51) that any $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDP\_EIGZ}}$ satisfies $\langle A^T A, \bar{X} - \bar{x}\bar{x}^T\rangle = 0$. Hence, $\bar{X}$ takes the form $\bar{X} = \bar{x}\bar{x}^T + ZVZ^T$ for all $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDP\_EIGZ}}$, where $Z \in \mathbb{R}^{n\times n-r}$ is a basis for the null space of A and $V \in \mathbb{S}^{n-r}$. For any $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDP\_EIGZ}}$ it follows that for all $j \in \mathcal{J}$:

$$\begin{aligned} & \langle S_j, \bar{X}\rangle + s_j^T\bar{x} + \eta_j && \text{(3.60a)}\\ = \;& \langle S_j, \bar{X} - \bar{x}\bar{x}^T\rangle + \bar{x}^T S_j\bar{x} + s_j^T\bar{x} + \eta_j && \text{(3.60b)}\\ = \;& \langle S_j, \bar{X} - \bar{x}\bar{x}^T\rangle = \langle A^T W_j^T + W_j A, ZVZ^T\rangle = 0 && \text{(3.60c)}\end{aligned}$$

where (3.60b) follows from adding and subtracting $\bar{x}^T S_j\bar{x}$, the first equality in (3.60c) follows from (R1), the second equality in (3.60c) from Proposition 3.10 and the final equality due to Z being a basis for the nullspace of A. Thus $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDP\_EIGJ}}$ proving the claim in (i).

Consider the claim in (ii). Suppose that there exist $\omega_j, j \in \mathcal{J}$ such that the condition in (ii) holds. We perform a linear combination of the inequalities in (3.59d) using ω_j to obtain for any $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDP\_EIGJ}}$:

$$\begin{aligned} 0 = \;& \sum_{j\in\mathcal{J}} \omega_j \left(\langle S_j, \bar{X}\rangle + s_j^T\bar{x} + \eta_j\right) && \text{(3.61a)}\\ = \;& \sum_{j\in\mathcal{J}} \omega_j \left(\langle S_j, \bar{X} - \bar{x}\bar{x}^T\rangle + \bar{x}^T S_j\bar{x} + s_j^T\bar{x} + \eta_j\right) && \text{(3.61b)}\\ = \;& \sum_{j\in\mathcal{J}} \omega_j\langle S_j, \bar{X} - \bar{x}\bar{x}^T\rangle = 2\langle A^T A, \bar{X} - \bar{x}\bar{x}^T\rangle && \text{(3.61c)}\end{aligned}$$

where (3.61b) follows from adding and subtracting $\bar{x}^T S_j\bar{x}$, the first equality in (3.61c) follows from (R1), the second equality in (3.61c) from Proposition 3.10 and the condition in (ii). Thus $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDP\_EIGZ}}$ proving the claim in (ii).

□

Note that Faye and Roupin [31] proved the equivalence between the SDP (3.50) and a similar SDP where (3.50d) is replaced by the constraints derived by lifting quadratic functions of the form $x_j(A_{i\cdot}x - b_i) = 0$, $i = 1, \ldots, m$, $j = 1, \ldots, n$ into the space of (x, X). Proposition 3.12 considerably expands the set of quadratic functions for which the feasible set of the resulting SDP is equal to $\mathcal{F}_{\text{SDP\_EIGZ}}$ (claim in (ii)). It is easy to verify that all of the examples of quadratic functions satisfying (R1) described in Faye and Roupin [31] do satisfy the condition in (ii). Further, the claim in (i) shows that there exist no quadratic functions satisfying (R1) for which the resulting SDP can have a smaller feasible region than the SDP (3.50). This brings us to the main result on the claim that the relaxation (3.41) is indeed the best among the class of relaxations in (3.57).

Theorem 3.1. *Suppose that Z^TQZ is indefinite and that the set $\mathcal{J}$ is chosen such that (R1) holds. Assume that $\alpha_z = -\lambda_{\min}(Z^TQZ)$ in* (3.41)*. Denote by μ_{EIGZ} and μ_{EIGJ} the optimal objective function values in* (3.41) *and* (3.57)*, respectively. Then, $\mu_{\text{EIGJ}} \leq \mu_{\text{EIGZ}}$.*

Proof. Let $\mu_{\text{SDP\_EIGZ}}$ and $\mu_{\text{SDP\_EIGJ}}$ denote the optimal objective values of the SDPs in (3.50) and (3.59), respectively. By Proposition 3.9 we have that $\mu_{\text{EIGZ}} = \mu_{\text{SDP\_EIGZ}}$. By Proposition (3.12)(i) we have that $\mu_{\text{SDP\_EIGJ}} \leq \mu_{\text{SDP\_EIGZ}}$. By Proposition 3.11 and weak duality we have that $\mu_{\text{EIGJ}} \leq \mu_{\text{SDP\_EIGJ}}$. Hence, $\mu_{\text{EIGJ}} \leq \mu_{\text{EIGZ}}$, proving the claim.

□

We finish this section by providing a theoretical comparison between the spectral relaxations studied in §3.3.1–3.3.3 and some SDP relaxations described in §3.2.2.

Theorem 3.2. *Assume that the matrix Z^TQZ is indefinite. Suppose that $\alpha_e = -\lambda_{\min}(Q)$ in* (3.26)*, $\alpha_g = -\lambda_{\min}(Q, I_n + A^TA)$ in* (3.33)*, and $\alpha_z = -\lambda_{\min}(Z^TQZ)$ in* (3.41)*. Denote by μ_{EIG}, μ_{GEIG}, μ_{EIGZ}, $\mu_{\text{SDP\_d}}$, $\mu_{\text{SDP\_dax}}$, and $\mu_{\text{SDP\_da}}$ the optimal objective function values of* (3.26)*,* (3.33)*,* (3.41)*,* (3.7)*,* (3.8)*, and* (3.9)*, respectively. Then, the following holds:*

(i) $\mu_{\text{SDP\_d}} \geq \mu_{\text{EIG}}$.

(ii) $\mu_{\text{SDP\_dax}} = \mu_{\text{SDP\_da}} \geq \mu_{\text{EIGZ}} \geq \mu_{\text{GEIG}} \geq \mu_{\text{EIG}}$.

Proof. We start by proving (i). Denote by $\mu_{\text{SDP\_EIG}}$ the optimal objective function value in (3.27). By Proposition 3.2, we have that $\mu_{\text{EIG}} = \mu_{\text{SDP\_EIG}}$. Hence, we can prove (ii) by comparing the SDPs (3.7) and (3.27). The constraints (3.7c) and (3.7d) are implied by (3.7b), and as a result, can be droped from (3.7). Therefore, (3.7) and (3.27) only differ in the constraints (3.7e) and (3.27c). It is simple to verify that the inequality (3.27c) can be obtained by aggregating the McCormick inequalities (3.7e), which implies that $\mu_{\text{SDP\_d}} \geq \mu_{\text{SDP\_EIG}} = \mu_{EIG}$.

Now, we prove (ii). As stated in §3.2.3.1, the relationship $\mu_{\text{SDP\_dax}} = \mu_{\text{SDP\_da}}$ follows from a result given in [31]. To show that $\mu_{\text{SDP\_da}} \geq \mu_{\text{EIGZ}}$, we follow the same line of arguments used for proving (i). Let $\mu_{\text{SDP\_EIGZ}}$ be the optimal objective function value in (3.50). Proposition 3.9 implies that $\mu_{\text{EIGZ}} = \mu_{\text{SDP\_EIGZ}}$. Therefore, to prove (ii), we can simply compare the SDPs (3.9) and (3.50). The constraints (3.7c) and (3.7d) are also redundant in (3.9), and can be dropped from this formulation as well. Similar to the previous case, (3.9) and (3.50) only differ in the constraints (3.7e) and (3.50c). As stated above, the inequality (3.50c) is implied by the inequalities (3.7e). Hence, $\mu_{\text{SDP\_da}} \geq \mu_{\text{SDP\_EIGZ}} = \mu_{EIGZ}$. The inequalities $\mu_{GEIG} \geq \mu_{EIG}$ and $\mu_{EIGZ} \geq \mu_{GEIG}$, follow from Propositions 3.4 and 3.7, respectively. This completes the proof of the claim in (ii). □

3.4 Spectral branching for nonconvex binary QPs

In this section, we introduce new eigenvalue-based branching variable selection strategies for nonconvex binary QPs. These strategies are inspired by the strong branching rule which was initially proposed for mixed-integer linear programs [6], and can be used along with the quadratic relaxations discussed in §3.3.1–3.3.3. For simplicity, we only describe

our branching strategies for the eigenvalue relaxation, which rely on the smallest eigenvalue of Q and its associated eigenvector. The branching rules for the quadratic relaxations described in §3.3.2 and 3.3.3 are similar, but they make use of the smallest generalized eigenvalue of the pair $(Q, I + \delta A^T A)$ and its corresponding eigenvector.

We first introduce some notation. Let $\mathcal{F}$ be the set of indices of the variables that are fixed at the current node. Denote by $\mathcal{B} = \{1, \ldots, n\} \setminus \mathcal{F}$ the set of branching candidates. Let $\bar{Q}$ be the $\mathbb{R}^{|\mathcal{B}|\times|\mathcal{B}|}$ sub-matrix of Q obtained by eliminating the rows and columns corresponding to the variables in $\mathcal{F}$. Define the bijection $\sigma : \mathcal{B} \to \{1, \ldots, |\mathcal{B}|\}$, which maps $i \in \mathcal{B}$ to the $\sigma(i)$-th row and $\sigma(i)$-th column of $\bar{Q}$.

Assume that we branch on variable x_i, $i \in \mathcal{B}$ by creating two nodes, one where $x_i = 0$ and another where $x_i = 1$. At these descendant nodes, the eigenvalue relaxation is constructed by considering the smallest eigenvalue of the submatrix obtained by eliminating the $\sigma(i)$-th row and $\sigma(i)$-th column of $\bar{Q}$. We denote this submatrix by $\hat{Q}$. In this context, a potentially good branching rule may consist in branching on the variable which leads to the largest increase in the smallest eigenvalue of $\hat{Q}$. Note that, at a given node of the branch-and-bound tree, this rule requires the solution of $|\mathcal{B}|$ eigenvalue problems, each one involving a submatrix of $\bar{Q}$ obtained by eliminating the row and column corresponding to a particular index $i \in \mathcal{B}$. We call this rule *spectral branching with complete enumeration*. The index corresponding to this branching rule, denoted as $i_{\text{exact}} \in C$, can be mathematically expressed as:

$$i_{\text{exact}} = \arg\max_{i\in\mathcal{B}} \lambda_{\min}\left(P_{\sigma(i)}\bar{Q}P_{\sigma(i)}^T\right) \tag{3.62}$$

where $P_{\sigma(i)}$ is a $(|\mathcal{B}|-1)\times|\mathcal{B}|$ matrix obtained by removing the $\sigma(i)$-th row from the $|\mathcal{B}|\times|\mathcal{B}|$ identity matrix. Note that $\hat{Q} = P_{\sigma(i)}\bar{Q}P_{\sigma(i)}^T$ results in a matrix where the $\sigma(i)$-th row and $\sigma(i)$-th column of $\bar{Q}$ are removed.

The computational complexity of complete enumeration is $\Omega(|\mathcal{B}|^3)$. We are not aware of

any efficient approach for obtaining i_{exact} that avoids complete enumeration. We instead rely on a lower bound for $\lambda_{\min}(\cdot)$ that will be obtained without computing an eigenvalue and is computationally inexpensive. Gershgorin's Circle Theorem (GCT) [34] provides such a lower bound estimate. The GCT states that: *every eigenvalue of a $t \times t$ matrix T lies in one of the circles $\mathcal{C}_k(T) = \{\lambda \ : \ |\lambda - T_{kk}| \leq \sum_{l \neq k} |T_{kl}|\}$ for $k = 1, \ldots, t$.* A lower bound estimate for the smallest eigenvalue of the matrix T based on the GCT, denoted as $\underline{\lambda}^{\text{GCT}}_{\min}(T)$ is:

$$\underline{\lambda}^{\text{GCT}}_{\min}(T) = \min_{k \in \{1,\ldots,t\}} \left(T_{kk} - \sum_{l \neq k} |T_{kl}| \right) \tag{3.63}$$

Using the GCT-based lower bound estimate we can then define a branching variable index as:

$$i_{\text{GCT}} = \arg\max_{i \in \mathcal{B}} \underline{\lambda}^{\text{GCT}}_{\min} \left(P_{\sigma(i)} \bar{Q} P^T_{\sigma(i)} \right). \tag{3.64}$$

Note that the index i_{GCT} can be determined without having to compute the matrix $P_{\sigma(i)} \bar{Q} P^T_{\sigma(i)}$. This approach has a computational complexity of $O(|\mathcal{B}|^2)$ and is computationally inexpensive compared to complete enumeration.

The choice of i_{GCT} can be viewed as a pessimistic estimate since it is obtained by maximizing the worst-case bound for the smallest eigenvalue. Instead, we employ a different approach to determine the branching variable. Let v be the eigenvector corresponding to the smallest eigenvalue of $\bar{Q}$. Then, we select as a branching variable, denoted by i_{approx}, the one which corresponds to the entry of v with the largest absolute value, i.e.

$$i_{\text{approx}} = \arg\max_{i \in \mathcal{B}} |v_{\sigma(i)}| \tag{3.65}$$

where $v_{\sigma(i)}$ denotes the $\sigma(i)$-th component of v. We call this rule *approximate spectral branching*. The computational complexity of this rule is $O(|\mathcal{B}|)$.

To appreciate the intuition behind this choice, we recall the proof for the GCT. From the definition of the eigenvalue, we have

$$
\begin{aligned}
&\bar{Q}v = \lambda_{\min}(\bar{Q})v \\
\implies &\sum_{j\in\mathcal{B}} \bar{Q}_{\sigma(i_{\text{approx}})\,\sigma(j)} v_{\sigma(j)} = \lambda_{\min}(\bar{Q}) v_{\sigma(i_{\text{approx}})} \\
\implies &\lambda_{\min}(\bar{Q}) - Q_{\sigma(i_{\text{approx}})\,\sigma(i_{\text{approx}})} = \sum_{\substack{j\in\mathcal{B},\\ j\neq i_{\text{approx}}}} \bar{Q}_{\sigma(i_{\text{approx}})\,\sigma(j)} \frac{v_{\sigma(j)}}{v_{\sigma(i_{\text{approx}})}} \\
\implies &|\lambda_{\min}(\bar{Q}) - Q_{\sigma(i_{\text{approx}})\,\sigma(i_{\text{approx}})}| \leq \sum_{\substack{j\in\mathcal{B},\\ j\neq i_{\text{approx}}}} |\bar{Q}_{\sigma(i_{\text{approx}})\,\sigma(j)}| \\
\implies &\lambda_{\min}(\bar{Q}) \in \mathcal{C}_{\sigma(i_{\text{approx}})}(\bar{Q})
\end{aligned}
$$

where the first implication follows from the $\sigma(i_{\text{approx}})$-th row of the equality, the second implication is obtained by rearranging and dividing by $v_{\sigma(i_{\text{approx}})}$ and the inequality follows from $|v_{\sigma(j)}/v_{\sigma(i_{\text{approx}})}| \leq 1$ by definition of i_{approx}. In essence, i_{approx} identifies the particular Gershgorin circle that bounds the smallest eigenvalue $\lambda_{\min}(\bar{Q})$. Thus, the choice of i_{approx} as the branching variable can be interpreted as eliminating the particular Gershgorin circle to which $\lambda_{\min}(\bar{Q})$ belongs. In that sense, this can be viewed as an optimistic estimate.

To illustrate the effectiveness of i_{GCT} and i_{approx} in mimicking i_{exact}, we performed some numerical experiments. We generated matrices Q of sizes $n \in \{50, 100\}$ and densities $\rho \in \{0.25, 0.50, 1.00\}$, and computed i_{exact} by complete enumeration. Denote by i_{worst} the index corresponding to the worst choice of branching variable, i.e.:

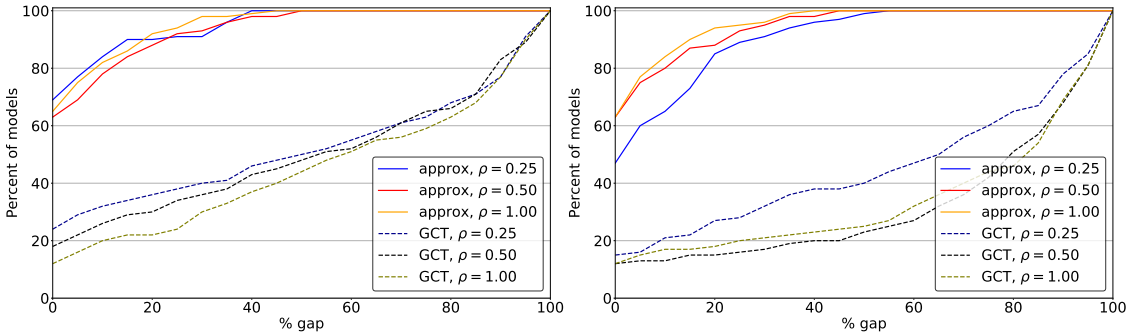
$$
i_{\text{worst}} = \arg\min_{i\in\mathcal{B}} \lambda_{\min}\left(P_{\sigma(i)} Q P^T_{\sigma(i)}\right) \tag{3.66}
$$

Then, the effectiveness of i_{x} is measured using the metric:

$$
\%\text{ gap} = \frac{\lambda_{\min}\left(P_{\sigma(i_{\text{x}})} Q P^T_{\sigma(i_{\text{x}})}\right) - \lambda_{\min}\left(P_{\sigma(i_{\text{exact}})} Q P^T_{\sigma(i_{\text{exact}})}\right)}{\lambda_{\min}\left(P_{\sigma(i_{\text{worst}})} Q P^T_{\sigma(i_{\text{worst}})}\right) - \lambda_{\min}\left(P_{\sigma(i_{\text{exact}})} Q P^T_{\sigma(i_{\text{exact}})}\right)} \times 100 \tag{3.67}
$$

where $\text{x} \in \{\text{approx}, \text{GCT}\}$. A smaller value of $\%$ gap for i_{x} represents a better approximation of i_{exact}. To obtain a statistic of the effectiveness of these approaches, we generated 100 different instances of Q for each matrix size and density. Figure 3.1 shows cumulative plots of the percentage of instances for which the $\%$ gap is below a certain value. It is evident

from the plots that the approximate spectral branching strategy is a better choice than the GCT-based branching rule.



(a) Instances of Q with $n = 50$

(b) Instances of Q with $n = 100$

Figure 3.1: Cumulative plots comparing the effectiveness of the approximate spectral branching and the GCT-based branching strategies.

3.5 Implementation of the proposed relaxation and branching strategies into BARON

By default, BARON's portfolio of relaxations consists of linear programming (LP), nonlinear programming (NLP) and mixed-integer linear programming (MILP) relaxations [48, 59, 92]. In our implementation, we have expanded this portfolio by adding a new class of convex QP relaxations. These relaxations are constructed whenever the original model supplied to BARON is of the form (3.1). We take advantage of BARON's convexity detector (see [48] for details) in order to determine the type of QP relaxation that will be constructed at a given node in the branch-and-bound tree. If the current node is convex, our QP relaxation is the continuous relaxation of (3.1) subject to the variable bounds of the current node. On the other hand, if the current node is nonconvex, we construct one of the QP relaxations introduced in §3.3.1–3.3.3. The relaxation (3.49) is selected by default if the

original problem contains equality constraints. Otherwise, our QP relaxation constructor automatically switches to the eigenvalue relaxation (3.26).

To solve the eigenvalue and generalized eigenvalue problems that arise during the construction of the relaxations discussed in §3.3.1–3.3.3, we use the subroutines included in the linear algebra library LAPACK [3]. When constructing these quadratic relaxations, we only consider the variables that have not been fixed at the current node. We use CPLEX as a subsolver for the new QP relaxations. The relaxation solution returned by the QP subsolver is used at the current node only if it satisfies the KKT conditions. This KKT test is similar to the optimality checks that BARON performs on the solutions returned by the LP and NLP subsolvers (see [48] for details).

Another important component of our implementation is the approximate spectral branching rule described in §3.4. This strategy is activated whenever the original problem supplied to BARON is a nonconvex binary QP. When this strategy is disabled, BARON uses reliability branching [1] to select among binary branching variables.

Finding δ

As stated in §3.3.3, when constructing the quadratic relaxation (3.49), we use a sufficiently large value of δ in order to obtain a good approximation of the bound provided by the eigenvalue relaxation in the nullspace of A. We use an iterative procedure to determine such value of δ. We start by setting $\delta = 1$ and computing $\lambda_{\min}(Q, I_n + \delta A^T A)$. Then, in each iteration of this procedure, we increase δ by a factor of σ and we use the resulting δ to compute a new value of $\lambda_{\min}(Q, I_n + \delta A^T A)$. The procedure terminates when either the relative change in $\lambda_{\min}(Q, I_n + \delta A^T A)$ is within a tolerance $relTol$ or the number of iterations reaches $maxIter$. In our numerical experiments, we set $\sigma = 10$, $maxIter = 5$, and $relTol = 10^{-3}$. This iterative procedure is executed at the root node only, and the value of δ determined during its execution is used throughout the entire branch-and-bound tree.

Dynamic relaxation selection strategy

We have implemented a dynamic relaxation selection strategy which is used for problems of the form (3.1) and switches between polyhedral and quadratic relaxations based on their relative strength. This dynamic strategy is motivated by two key observations. First, the strength of a given relaxation may depend on particular characteristics of the problem under consideration. Second, a particular type of relaxation may become stronger than other classes of relaxations as we move down the branch-and-bound tree.

In the context of this strategy, we dynamically adjust the frequencies at which we solve the different types of relaxations during the branch-and-bound search. Denote by $\omega_{lp} \in [1, \bar{\omega}_{lp}]$ and $\omega_{qp} \in [1, \bar{\omega}_{qp}]$ the frequencies with which we solve the LP and QP relaxations, respectively. Let f_{lp} and f_{qp} be the optimal objective function values of the LP and QP relaxations, respectively. At the beginning of the global search, we set $\omega_{lp} = 1$ and $\omega_{qp} = 1$, which indicates that both the LP and QP relaxations will be solved at every node of the branch-and-bound tree. At nodes where both LP and QP relaxations are solved, we compare their corresponding objective function values. If $f_{qp} - f_{lp} \geq absTol$, we increase ω_{qp} by setting $\omega_{qp} = \max(1, \omega_{qp}/\sigma_{qp})$, and decrease ω_{lp} by setting $\omega_{lp} = \min(\bar{\omega}_{lp}, \omega_{lp} \cdot \sigma_{lp})$. Conversely, if $f_{qp} - f_{lp} < absTol$, we increase ω_{lp} by setting $\omega_{lp} = \max(1, \omega_{lp}/\sigma_{lp})$, and decrease ω_{qp} by setting $\omega_{qp} = \min(\bar{\omega}_{qp}, \omega_{qp} \cdot \sigma_{qp})$. In our numerical experiments, we set $\sigma_{lp} = 10$, $\sigma_{qp} = 2$, $\bar{\omega}_{lp} = 1000$, $\bar{\omega}_{qp} = 10$, and $absTol = 10^{-3}$.

Even though BARON's portfolio of relaxations also includes MILP relaxations, in our dynamic relaxation selection strategy, we only compare the bounds given by the LP and QP relaxations. Since MILP relaxations can be computationally expensive, BARON uses a heuristic to decide if an MILP relaxation will be solved at the current node [59]. In our implementation, this heuristic is invoked only if at the current node the QP relaxation is weaker than the LP relaxation. Otherwise, the MILP relaxation is skipped altogether.

3.6 Computational results

In this section, we present the results of an extensive computational study conducted to investigate the impact of the techniques proposed in this chapter on the performance of branch-and-bound algorithms. We start by describing the test set used for the numerical experiments in §3.6.1. Then, in §3.6.2, we provide a numerical comparison between the spectral relaxations introduced in §3.3.1–3.3.3 and some of the relaxations reviewed in §3.2. In §3.6.3, we analyze the impact of the implementation described in §3.5 on the performance of the global optimization solver BARON. This is followed by a comparison between several state-of-the-art global optimization solvers in §3.6.4. Finally in §3.6.5, we compare BARON and the QCR approach discussed in §3.2.3.3.

Throughout this section, all experiments are conducted under GAMS 30.1.0 on a 64-bit Intel Xeon X5650 2.66GHz processor with a single-thread. We solve all problems in minimization form. For the experiments described in §3.6.2, the linear and convex quadratic programs are solved using CPLEX 12.10, whereas the SDPs are solved using MOSEK 9.1.9. For the experiments considered in §3.6.3–3.6.5, we consider the following global optimization solvers: ANTIGONE 1.1, BARON 19.12, COUENNE 0.5, CPLEX 12.10, GUROBI 9.0, LINDOGLOBAL 12.0 and SCIP 6.0. When dealing with nonconvex problems, we: (i) run all solvers with relative/absolute tolerances of 10^{-6} and a time limit of 500 seconds, and (ii) set the CPLEX option `optimalitytarget` to 3 and the GUROBI option `nonconvex` to 2 in order to ensure that these two solvers search for a globally optimal solution. For other algorithmic parameters, we use default settings. The computational times reported in our experiments do not include the time required by GAMS to generate problems and interface with solvers; only times taken by the solvers are reported.

3.6.1 The test set

We consider a test set consisting of 960 Cardinality Binary Quadratic Programs (CBQPs), 30 Quadratic Semi-Assignment Problems (QSAPs), 246 Box-Constrained Quadratic Programs (BoxQPs), and 315 Equality Integer Quadratic Programs (EIQPs). In the following, we describe each of these four collections in detail.

3.6.1.1 Cardinality Binary Quadratic Programs

The CBQP instances are of the form:

$$\begin{array}{ll} \min\limits_{x} & x^T Q x + q^T x \\ \text{s.t.} & \sum\limits_{i=1}^{n} x_i = \kappa \\ & x_i \in \{0,1\}, \;\; i = 1, \ldots, n \end{array} \tag{3.68}$$

where $Q \in \mathbb{S}^n$ is an indefinite matrix, $q \in \mathbb{R}^n$ and $\kappa \in \{1, \ldots, n\}$. For our experiments, we use the 960 CBQP instances generated by Lima and Grossmann [53]. These problems were constructed for $\kappa \in \{n/5, n/1.25\}$, and matrices Q with sizes $n \in \{50, 75, 100, 200, 300, 400\}$ and densities $\rho \in \{0.10, 0.50, 0.75, 1.00\}$. The nonzero entries of Q and q are randomly generated from uniform distributions defined over the intervals $[-100, 100]$, $[-1, -1]$, $[0, 1]$, and $[0, 100]$.

3.6.1.2 Quadratic Semi-Assignment Problems

The QSAP instances are of the form:

$$\begin{array}{ll} \min\limits_{x} & \sum\limits_{i=1}^{n_1-1} \sum\limits_{j+1=1}^{n_1} \sum\limits_{k=1}^{n_2} \sum\limits_{l=1}^{n_2} Q_{ikjl} x_{ik} x_{jl} + \sum\limits_{i=1}^{n_1} \sum\limits_{k=1}^{n_2} q_{ik} x_{ik} \\ \text{s.t.} & \sum\limits_{k=1}^{n_2} x_{ik} = 1, \;\; i = 1, \ldots, n_1 \\ & x_{ik} \in \{0,1\}, \;\; i = 1, \ldots, n_1, k = 1, \ldots, n_2 \end{array} \tag{3.69}$$

where $n_1 > n_2$. For our experiments, we constructed 30 QSAP instances for which the number of total variables ranges from 15 to 280, and the coefficients Q_{ikjl} and q_{ik} are ran-

domly generated in $[-50, 50]$ according to a uniform distribution.

3.6.1.3 Box-Constrained Quadratic Programs

The BoxQP instances are of the form:

$$\begin{array}{ll} \min\limits_{x} & x^T Q x + q^T x \\ \text{s.t.} & 0 \leq x_i \leq 1, \;\; i = 1, \ldots, n \end{array} \tag{3.70}$$

where $Q \in \mathbb{S}^n$ is an indefinite matrix and $q \in \mathbb{R}^n$. For our experiments, we consider an expanded version of the set of the BoxQP instances generated in [22, 24, 96]. The original collection consists of: 54 instances with $20 \leq n \leq 60$ and $0.2 \leq \rho \leq 1.0$ generated in [96], 36 instances with $70 \leq n \leq 100$ and $0.25 \leq \rho \leq 0.75$ generated in [24], and 9 instances with $n = 125$ and $0.25 \leq \rho \leq 0.75$ generated in [22].

For our experiments, we constructed 15 additional instances with $70 \leq n \leq 125$ and $\rho = 1.00$, and 132 additional instances with $150 \leq n \leq 400$ and $0.25 \leq \rho \leq 1.0$, obtaining an expaded collection with 246 instances. For the additional 147 instances, as well as for the 99 instances considered in [22, 24, 96], the nonzero entries of Q and q are integers randomly generated in $[-50, 50]$ according to a uniform distribution.

3.6.1.4 Equality Integer Quadratic Programs

The EIQP instances are of the form:

$$\begin{array}{ll} \min\limits_{x} & x^T Q x + q^T x \\ \text{s.t.} & A_{1\cdot} x = b_1 \\ & 0 \leq x_i \leq u_i, \;\; i = 1, \ldots, n \\ & x_i \in \mathbb{N}, i = 1, \ldots, n \end{array} \tag{3.71}$$

where $Q \in \mathbb{S}^n$ is an indefinite matrix, $q \in \mathbb{R}^n$, $A_1 \in \mathbb{R}^{1 \times n}$ and $b_1 \in \mathbb{R}$. For our experiments, we consider an expanded version of the set of randomly generated EIQP instances used in [16]. The original collection consists of three classes of instances generated as follows:

- EIQP1: The entries of Q and q are uniformly distributed integers in the interval $[-100, 100]$, the entries of A_1 are uniformly distributed integers in $[1, 50]$, $b_1 = 15 \cdot \sum_{i=1}^{n} A_{1i}$, and $u_i = 30,\ i = 1, \ldots, n$.
- EIQP2: the entries of Q and q are generated as described for EIQP1, the entries of A_1 are uniformly distributed integers in $[1, 100]$, $b_1 = 20 \cdot \sum_{i=1}^{n} A_{1i}$, and $u_i = 50,\ i = 1, \ldots, n$.
- EIQP3: the entries of Q and q are generated as described for EIQP1, the entries of A_1 and b_1 as described for EIQP2, and $u_i = 70,\ i = 1, \ldots, n$.

In [16], 5 different instances were generated for each of these three classes and for each value of $n \in \{20, 30, 40\}$, obtaining a total of 45 instances. For our experiments, we constructed 90 additional instances of each class by considering values of n ranging from 60 to 400. This leads to an expanded collection consisting of 315 instances.

3.6.2 Comparison between relaxations

In this section, we present a comparison between the spectral relaxations introduced in §3.3.1–3.3.3, the convex quadratic relaxation (3.12), the first-level RLT relaxation (3.5), and the SDP relaxations (3.7) and (3.9). We construct performance profiles based on the root-node relaxation gap defined as:

$$\text{GAP} = \left(\frac{f_{UBD} - f_{LBD}}{\max(|f_{LBD}|, 10^{-3})} \right) \times 100 \tag{3.72}$$

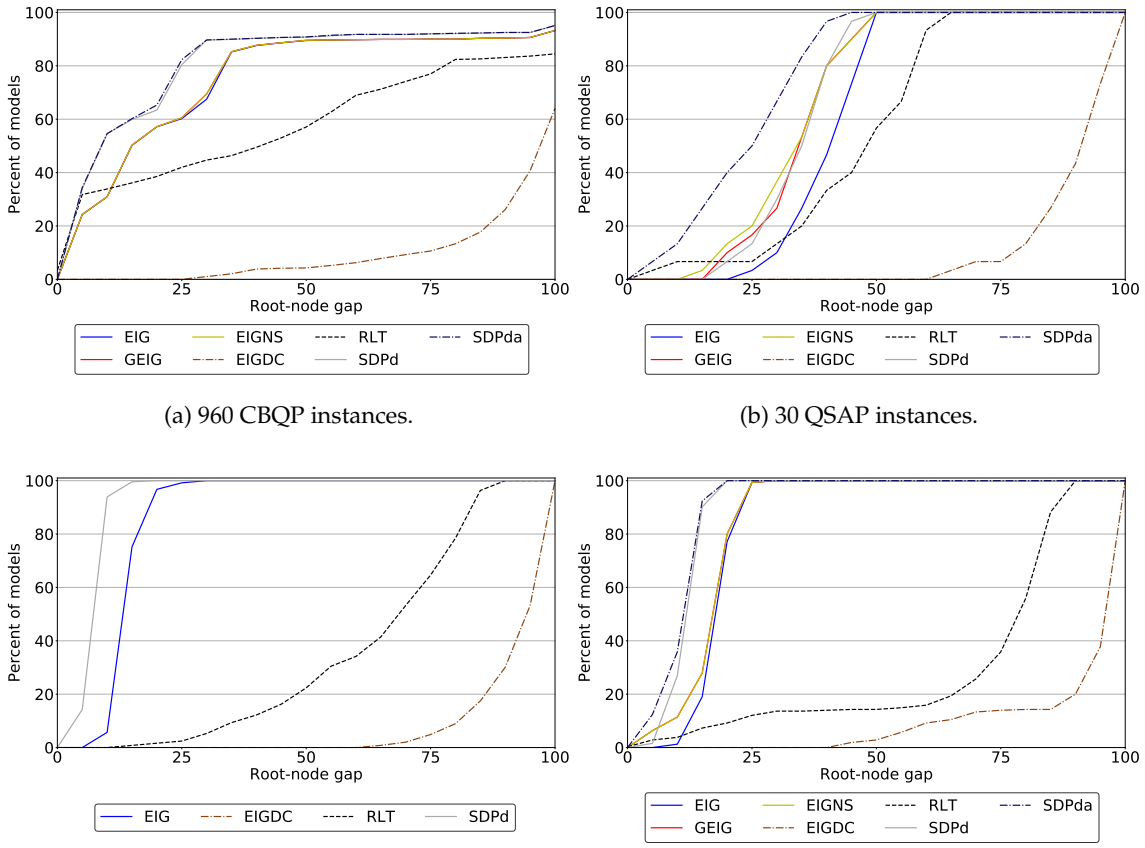
where f_{LBD} is the root-node relaxation lower bound, and f_{UBD} is the best upper bound available for a given instance. The following notation is used to refer to the different relaxations:

- EIG: Eigenvalue relaxation (3.26).
- GEIG: Generalized eigenvalue relaxation (3.33).

- EIGNS: Eigenvalue relaxation in the nullspace of A (3.41).
- EIGDC: Quadratic relaxation (3.12) based on the eigdecomposition of Q.
- RLT: First-level RLT relaxation (3.5).
- SDPd: SDP relaxation (3.7).
- SDPda: SDP relaxation (3.9).

The performance profiles are presented in Figures 3.2a–3.2d. These profiles show the percentage of models for which the gap defined in (3.72) is below a certain threshold. As seen in the figures, the SDP relaxations give the tightest bounds, followed by the spectral relaxations. For these instances, both the RLT relaxation (3.5) and the quadratic relaxation (3.12) provide relatively weak bounds. Note that for the CBQP instances, the spectral relaxations provide very similar bounds. In the case of the QSAP and EIQP problems, the difference between the bounds given by spectral relaxations is more significant.

We also compare these root-node relaxations in terms of their solution times. To that end, in Figures 3.3a–3.3d, we present the geometric means of the CPU times required to solve the different classes of relaxations. For the quadratic relaxation based on the eigdecomposition of Q, the CPU time includes the time required to solve the convex QP (3.12) and the time taken by the bounding LPs (3.13). We group the instances based on their size. As the figures indicate, the spectral relaxations are relatively inexpensive regardless of the characteristics of the problem. As the size of the problem increases, the RLT relaxations become more expensive to solve, and in some cases, these RLT relaxations are orders of magnitude more expensive than the other relaxations. Note that the separable programming procedure described in §3.2.3.1 does not only lead to relatively weak bounds, but it is also computationally expensive since it requires the solution of $2n$ linear programs. Even though for most of the problems considered in the experiments the SDP relaxations can be solved within 10 seconds, they are between one and two orders of magnitude more expensive than the spectral relaxations. These results indicate that the quadratic relaxations



(a) 960 CBQP instances.

(b) 30 QSAP instances.

(c) 246 BoxQP instances.

(d) 315 EIQP instances.

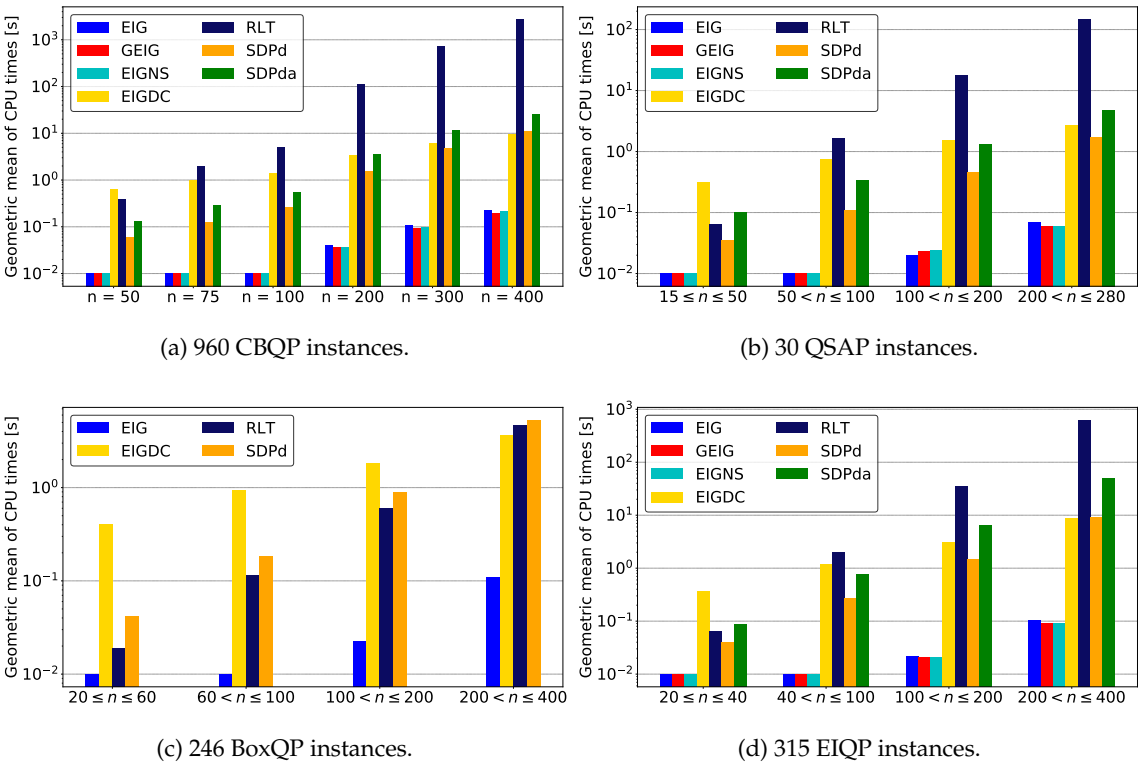
Figure 3.2: Comparison between the root-node relaxations gaps.

introduced in this chapter do not only provide relatively strong bounds, but they are also very cheap to solve.

3.6.3 Impact of the implementation on BARON's performance

In this section, we demonstrate the benefits the proposed relaxation and branching techniques on the performance of the global optimization solver BARON. In our experiments, we consider the following versions of BARON 19.12:

- BARONnoqp: BARON without the spectral relaxations and without the spectral



(a) 960 CBQP instances.

(b) 30 QSAP instances.

(c) 246 BoxQP instances.

(d) 315 EIQP instances.

Figure 3.3: Geometric means of the CPU times required to solve the root-node relaxations.

branching rule.

- BARONnosb: BARON with the spectral relaxations but without the spectral branching rule.
- BARONqp1: BARON with the spectral relaxations and the approximate spectral branching rule.

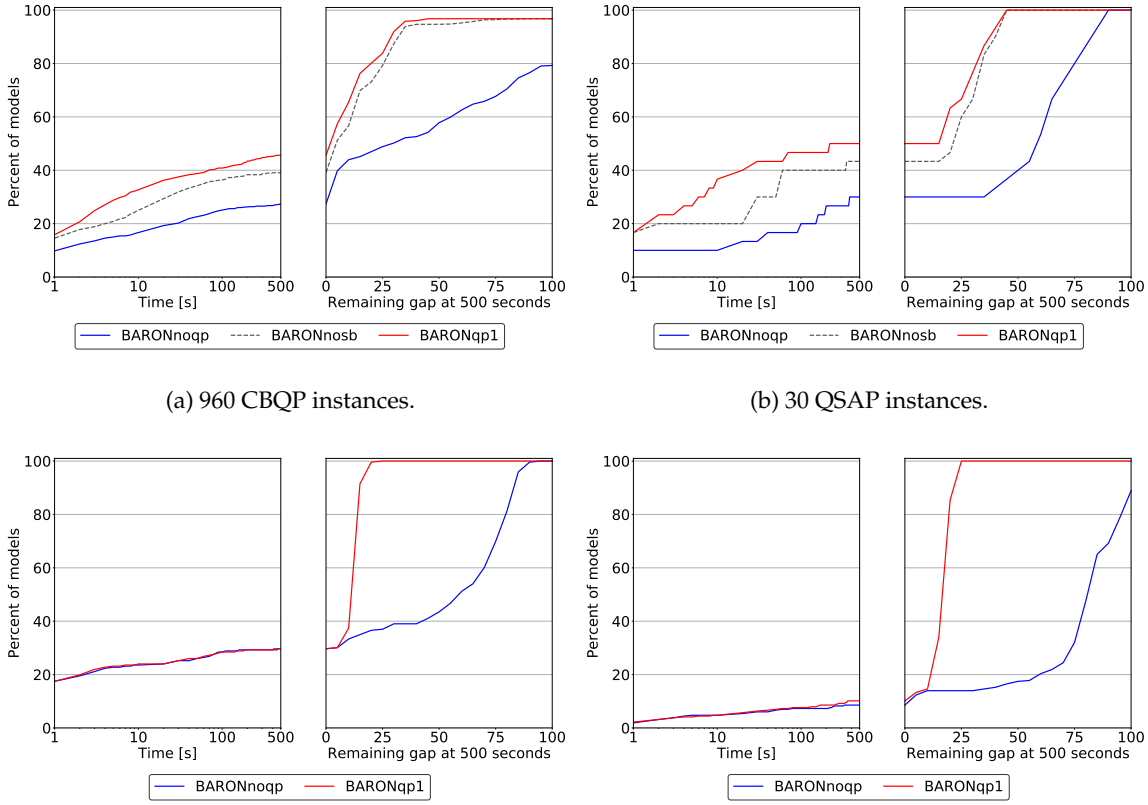
As mentioned previously, the spectral branching rule introduced in § 3.4 is only used for the binary instances. In order to analyze the impact of our implementation, we start by comparing the different versions of BARON through performance profiles. For instances which can be solved to global optimality within the time limit of 500 seconds, we use performance profiles based on CPU times. In this case, for a given solver, we plot the per-

centage of models that can be solved within a certain amount of time. For problems for which global optimality cannot be proven within the time limit, we employ performance profiles based on the optimality gaps at termination. These gaps are determined according to (3.72) by using the best lower and upper bounds reported by the solver under consideration. In this case, for a given solver, we plot the percentage of models for which the remaining gap is below a given threshold.

The performance profiles are presented in Figures 3.4a–3.4d. As seen in the figures, our implementation leads to very significant improvements in the performance of BARON. Clearly, for the CBQP and QSAP instances, both the spectral relaxations and the spectral branching strategy result in a version of BARON which is able to solve many more problems to global optimality. In addition, in cases in which global optimality cannot be proven within the time limit, BARONqp1 terminates with much smaller relaxation gaps than BARONnoqp.

Next, we provide a more detailed comparison between BARONqp1 and BARONnoqp. To this end, we eliminate from the test set all the problems that can be solved trivially by both solvers (146 instances). A problem is regarded as trivial if it can be solved by both solvers in less than one second. After eliminating all of these problems from the original test set, we obtain a new test set consisting of 1405 instances.

We first consider the nontrivial problems that are solved to global optimality by at least one of the two the versions of the solver (412 instances). For this analysis, we compare the performance of the two solvers by considering the ratios between their computational times. In this comparison, we say that the two solvers perform similarly if their CPU times are within 10% of each other. The results are presented in Figure 3.5a. As the figure indicates, BARONqp1 is significantly faster than BARONnoqp. For nearly 50% of the problems considered in this comparison, BARONqp1 is at least one of magnitude faster than BARONnoqp



(a) 960 CBQP instances.

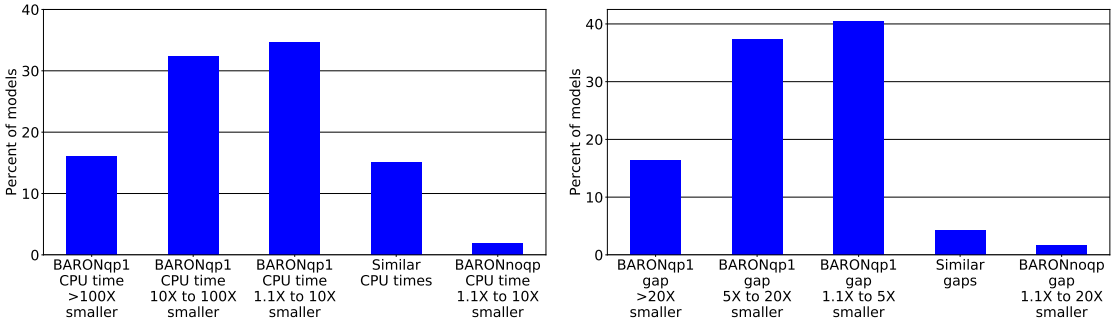
(b) 30 QSAP instances.

(c) 246 BoxQP instances.

(d) 315 EIQP instances.

Figure 3.4: Comparison between the different versions of BARON.

Now, we consider the nontrivial problems that neither of the two solvers are able to solve to global optimality within the time limit (993 instances). In this case, we analyze the performance of these solvers by comparing the gaps reported at termination. For the purposes of this comparison, we say that two solvers obtain similar gaps if their remaining gaps are within 10% of each other. The results are presented in Figure 3.5b. As seen in the figure, for more than 90% considered in this comparison, BARONqp1 reports significantly termination gaps than BARONnoqp.



(a) CPU times (412 nontrivial instances).

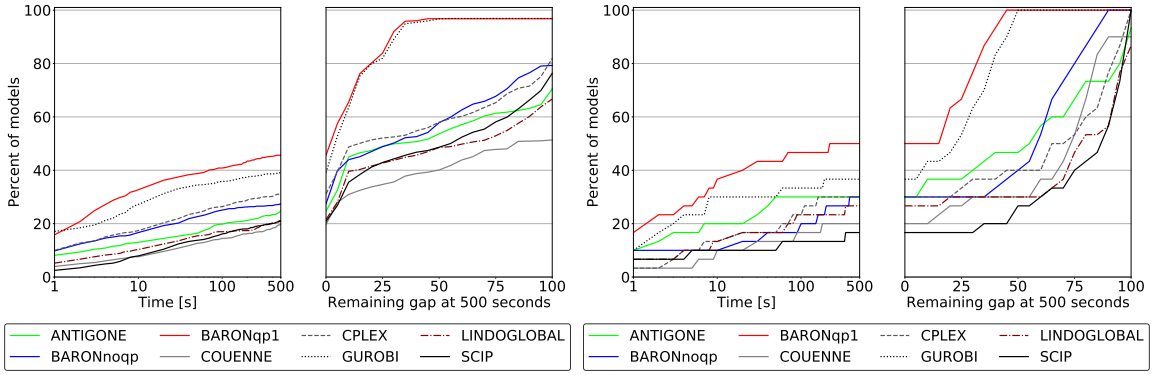
(b) Relative gaps (993 nontrivial instances).

Figure 3.5: One-to-one comparison between BARONqp1 and BARONnoqp.

3.6.4 Comparison between global optimization solvers

In this section, we provide a comparison between different state-of-the-art global optimization solvers using the same type of performance profiles considered in the previous section. These profiles are shown in Figures 3.6a–3.6d. As seen in these figures, BARONqp1 performs well in comparison to other solvers. For both the CBQP and QSAP instances, BARONqp1 is faster than the other solvers and is able to solve many more problems to global optimality. For the QSAP and BoxQP instances, BARONqp1 also terminates with significantly smaller gaps than the other solvers in cases in which global optimality cannot be proven within the time limit. Note that many of the BoxQP and EIQP instances are very challenging and cannot be globally solved within the time limit by solvers considered in this analysis.

Next, we provide a detailed analysis involving BARONqp1, CPLEX and GUROBI. For this analysis, we use the same type of bar plots employed in Figure 3.5. We start by presenting a one-to-one comparison between BARONqp1 and CPLEX. To this end, we eliminate from the test set all the problems that can be solved trivially by both solvers (124 instances), obtaining a new test set with 1427 instances. In Figure 3.7a, we consider the



(a) 960 CBQP instances.

(b) 30 QSAP instances.

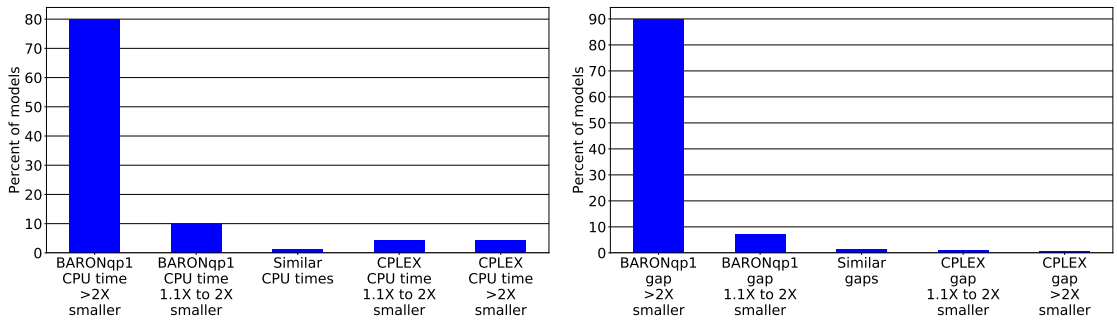
(c) 246 BoxQP instances.

(d) 315 EIQP instances.

Figure 3.6: Comparison between global optimization solvers.

nontrivial problems that are solved to global optimality by at least one of the two solvers (445 instances), whereas in Figure 3.7b, we consider nontrivial problems that neither of the two solvers are able to solve to global optimality within the time limit (982 instances). As both figures show, BARONqp1 performs significantly better than CPLEX. For 80% of the instances considered in Figure 3.7a, BARONqp1 is at least twice as fast as CPLEX. Similarly, for 90% of the instances considered in Figure 3.7b, the termination gaps reported by BARONqp1 are at least twice as small as those obtained by CPLEX.

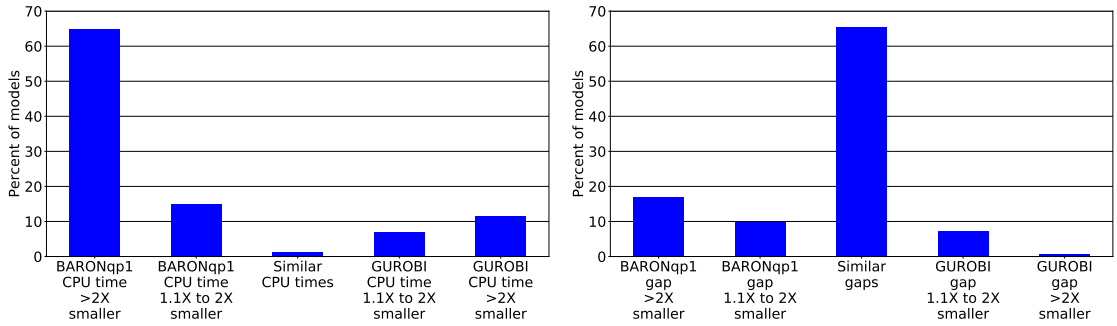
Now, we present a similar one-to-one comparison between BARONqp1 and GUROBI.



(a) CPU times (445 nontrivial instances). (b) Relative gaps (982 nontrivial instances).

Figure 3.7: One-to-one comparison between BARONqp1 and CPLEX.

Once again, we eliminate from the test set all the problems that can be solved trivially by both solvers (185 instances), resulting in a new test set with 1366 instances. In Figure 3.8a, we consider the nontrivial problems that are solved to global optimality by at least one of the two solvers (380 instances), whereas in Figure 3.8b, we consider nontrivial problems that neither of the two solvers are able to solve to global optimality within the time limit (986 instances). For more than 60% of the instances considered in Figure 3.8a, BARONqp1 is at least twice as fast as GUROBI, whereas for most of the problems considered in Figure 3.8b, the two solvers report similar termination gaps.



(a) CPU times (380 nontrivial instances). (b) Relative gaps (986 nontrivial instances).

Figure 3.8: One-to-one comparison between BARONqp1 and GUROBI.

3.6.5 Comparison with the QCR method

In this section, we provide a numerical comparison between the relaxation and branching strategies proposed in this chapter, and the QCR approach reviewed in §3.2.3.3. To this end, we consider the 990 binary CBQP and QSAP instances described in §3.6.1.

To apply the QCR method, we proceed in two steps. In the first step, for each test problem, we solve the SDP relaxation (3.8) with MOSEK and use its dual solution to construct a reformulated convex binary quadratic program of the form (3.20). As mentioned in §3.2.3.3, the reformulated problems are equivalent to the original problems when all variables are binary.

In the second step, we solve the reformulated problems using a customized version of BARON, which denote by BARONqcr. This version of BARON only differs from the default one in two aspects. First, BARONqcr is devised in a way such that, at a given node of the branch-and-bound tree, the lower bound is obtained by solving the continuous relaxation of the reformulated problem, which is a convex QP. To this end, in the lower bounding routines of BARONqcr, we have disabled the LP, NLP, MILP and recently introduced spectral relaxations. Second, in BARONqcr we have also turned off the spectral branching rule and replaced it with the reliability branching strategy described in [59]. Recall that, under the QCR approach, the perturbation parameters used to derive the reformulated problem are not updated during the execution of the branch-and-bound algorithm. As a result, the QP relaxations that are constructed at different nodes of the branch-and-bound tree of BARONqcr only differ from one another in the variables that are fixed. We solve all of these convex QP relaxations by using CPLEX.

In our experiments, we run BARONqcr with the same relative/absolute tolerances and time limit used for BARONqp1. For all of the considered instances, the amount of time required to solve the SDP relaxation involved in the first step of the QCR method was much

smaller than the CPU time corresponding to BARONqcr. As a result, when comparing BARONqp1 and BARONqcr, we ignore the time required to solve these SDPs.

We first compare BARONqp1 and BARONqcr in terms of the lower bounds reported at the root-node. In this case, we say that a given solver obtains a better lower bound if the relative lower bound difference is greater than 10^{-3}. For cases in which the magnitude of the lower bound is below one, we use absolute differences. The results are presented in Table 3.1. In this table, for each test library, we provide the number, and in parentheses the percentage, of problems for which a given solver reports better root-node lower bounds. As the results in this table indicate, BARONqcr obtains better root-node lower bounds for most of the instances considered in this comparison.

Note that at the root node of the branch-and-bound tree, the lower bound obtained by BARONqcr is given by the continuous relaxation of (3.20), and as a result, it is equal to the bound provided by the SDP relaxation (3.8). On the other hand, for many of the problems considered in this comparison, the eigenvalue relaxation in the nullspace of A is tighter than the polyhedral relaxations implemented in BARONqp1. Hence, in these cases, BARONqp1 relies on this quadratic relaxation to obtain lower bounds. As shown in Theorem 3.2, the SDP relaxation (3.8) is at least as tight as the eigenvalue relaxation in the nullspace of A. Therefore, it is not surprising that, for many of the problems considered in Table 3.1, BARONqcr provides tighter root-node bounds than BARONqp1.

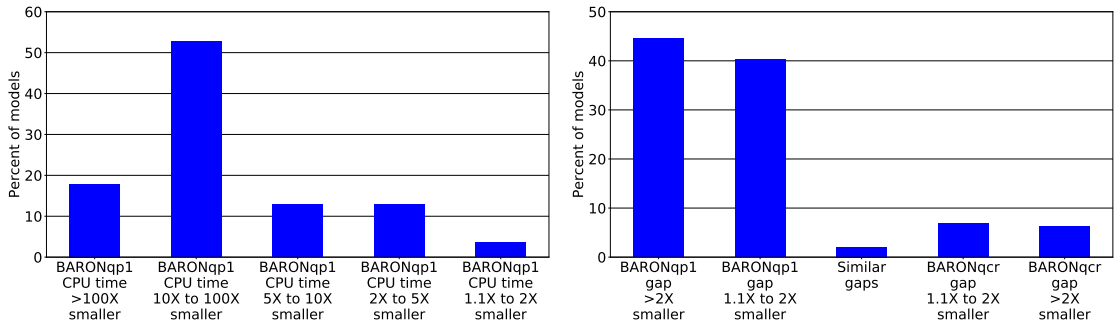
Table 3.1: Root-node lower bounds given by BARONqp1 and BARONqcr.

Test library	BARONqp1 better	BARONqcr better
CBQP	141 (15%)	819 (85%)
QSAP	2 (7%)	28 (93%)

Now, we analyze how the branch-and-bound algorithms of BARONqp1 and BARONqcr perform relative to each other. For this analysis, we use the same type of bar plots

employed in previous sections. We start by eliminating from the test set all the problems that can be solved trivially by both solvers (11 instances), obtaining a new test set with 979 instances. In Figure 3.9a, we consider the nontrivial problems that are solved to global optimality by at least one of the two solvers (442 instances), whereas in Figure 3.9b, we consider nontrivial problems that neither of the two solvers are able to solve to global optimality within the time limit (537 instances). As both figures show, BARONqp1 performs significantly better than BARONqcr. For nearly 70% of the instances considered in Figure 3.9a, BARONqp1 is at least an order of magnitude faster than BARONqcr. Similarly, for more than 80% of the instances considered in Figure 3.9b, the termination gaps reported by BARONqp1 are smaller than those obtained by BARONqcr.

Even though BARONqcr reports tighter root-node lower bounds than BARONqp1 for most of the instances considered in this comparison, during the branch-and-bound search, the lower bounds obtained by BARONqp1 improve much more quickly than those provided by BARONqcr. This is due to the fact that, in BARONqp1, we update the perturbation parameters used to construct the quadratic relaxations as we branch. In addition, BARONqp1 makes use of the approximate spectral rule introduced in §3.4, which as shown in §3.6.3, also has a significant impact on the performance of this solver.



(a) CPU times (442 nontrivial instances).

(b) Relative gaps (537 nontrivial instances).

Figure 3.9: One-to-one comparison between BARONqp1 and BARONqcr.

3.7 Conclusions

In this chapter, we introduced a family of convex quadratic relaxations for nonconvex QPs and MIQPs. We studied the theoretical properties of these relaxations and showed that they are equivalent to some particular SDPs. We also devised a novel branching variable selection strategy which involves an approximation of the impact of the branching decisions on the quality of these relaxations. To assess the benefits of our approach, we incorporated the proposed relaxation and branching techniques into the global optimization solver BARON, and tested our implementation on a large collection of problems. Results demonstrated that, for our test problems, our implementation leads to a very significant improvement in the performance of BARON, enabling it to solve many more problems to global optimality.

Chapter 4

SDP-quality bounds via convex quadratic relaxations for global optimization of mixed-integer quadratic programs

We address the global optimization of problems of the form:

$$\begin{array}{ll} \min\limits_{x\in\mathbb{R}^n} & x^TQx + q^Tx \\ \text{s.t.} & Ax = b \\ & x_i \in S_i,\ \forall i \in [n] := \{1,\dots,n\} \end{array} \tag{4.1}$$

where $Q \in \mathbb{R}^{n\times n}$ is a symmetric matrix which may be indefinite, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. For each $i \in [n]$, we assume that S_i is a bounded set given by the union of finitely many closed intervals in $\mathbb{R}$. Throughout this chapter, we also assume that A has rank m and use $Z \in \mathbb{R}^{n\times m}$ to denote an orthonormal basis for the nullspace of A.

The formulation in (4.1) subsumes many classes of problems including nonconvex QPs and MIQPs which typically arise in applications including facility location and quadratic assignment [49], molecular conformation [69] and max-cut problems [33]. These problems have received considerable attention in recent years and can be very challenging to solve to global optimality.

State-of-the-art global optimization solvers rely on branch-and-bound algorithms in order to solve nonconvex problems of the form (4.1). The efficiency of these algorithms

depends to a large extent on the tightness and the computational cost of the relaxations solved during the lower bounding step. Commonly used relaxations for (4.1) include the polyhedral, semi-definite programming (SDP) and convex quadratic relaxations reviewed in §3.2.

In Chapter 3, we derived convex quadratic relaxations of (4.1) by convexifing the objective function through uniform diagonal perturbations of Q. These perturbations were constructed by solving eigenvalue and generalized eigenvalue problems involving Q and A. Through numerical experiments, we demonstrated that these relaxations are not only inexpensive to solve, but can also provide very tight bounds, significantly improving the performance of branch-and-bound algorithms.

Motivated by these results, in this chapter, we consider a related class of convex quadratic relaxations. In particular, we investigate quadratically constrained programming (QCP) relaxations for (4.1). These relaxations are derived via convex quadratic cuts obtained from nonuniform diagonal perturbations of Q. We show that these relaxations: (i) are at least as tight as the spectral relaxations introduced in 3.3.1–3.3.3, and (ii) provide a very good approximation of the bounds given by certain SDP relaxations of (4.1).

The idea of using convex quadratic inequalities to approximate the bounds of certain SDP relaxations of (4.1) has been investigated before in the literature. Saxena et al. [79] considered the SDP relaxation of 4.1 obtained after adding the RLT inequalities and proposed a procedure to project the feasible region of this relaxation onto the space of original variables. This projection relies on convex quadratic cuts derived from an SDP separation program which is solved by applying a sub-gradient-based algorithm. Even though the relaxations generated through this approach are nearly as tight as the original SDP formulation, the separation problem used to derive the quadratic cuts can be very expensive to solve.

Dong [29] used a different SDP relaxation for (4.1) which only includes the diagonal

RLT inequalities and showed this SDP is equivalent to a particular semi-infinite program. This semi-infinite formulation served as a motivation to construct convex QCP relaxations for (4.1) via convex quadratic cuts derived from a semidefinite separation problem with a special structure. To ensure that the solution of the separation problem is finitely attained, Dong [29] proposed a regularized version of this semidefinite program, and demonstrated that it can solved very efficiently through a specialized coordinate descent algorithm. Our work is partially inspired by the ideas proposed by Dong [29]. We refine his approach in several directions and make various theoretical and algorithmic contributions.

Our first contribution is a new class of convex QCP relaxations for (4.1) constructed by using information from both Q and the constraints $Ax = b$. These relaxations are derived from a semi-infinite program that generalizes the semi-infinite formulation proposed in [29]. Under our approach, we use the matrix $A^T A$ in order to modify the semidefinite constraint of the separation problem solved in [29]. This modification allows us to construct convex QCP relaxations which are at least as tight as those considered in [29].

In our second contribution, we provide conditions under which our semi-infinite formulation is equivalent to a well-known SDP relaxation of (4.1). Moreover, we show that this SDP is the best relaxation in the class of SDP relaxations considered in this chapter.

In our third contribution, we present a new analysis of the separation problem used to derive our quadratic cuts. In particular, we provide results on the finite attainment of this SDP by using its dual formulation. These results also apply to the separation problem in [29], which is a special case of ours.

Motivated by this analysis, in our fourth contribution, we propose a new regularization approach for the semidefinite separation problem and modify the coordinate descent algorithm introduced in [29] accordingly. Through numerical experiments, we show that the quadratic cuts derived from our regularized separation problem provide a much better approximation of certain SDP bounds than the quadratic cuts obtained from the regularized

separation problem proposed in [29].

In order to assess the computational benefits of the proposed techniques, we implement the new quadratic relaxations in the global optimization solver BARON. These relaxations are incorporated into BARON's portfolio of relaxations and invoked according to a dynamic relaxation selection strategy introduced in 3.5. For binary quadratic programs, our implementation also relies on a spectral branching strategy also developed in 3.4. We test our implementation on a large set of problems. Numerical results show that the new quadratic relaxations lead to a significant improvement in the performance of BARON, resulting in a new version of this solver which outperforms other state-of-the-art solvers such as CPLEX and GUROBI for many of our test problems.

The remainder of this chapter is organized as follows. In §4.1 we introduce the relaxations considered in this chapter and investigate their theoretical properties. In §4.2, we provide a new analysis on the finite attainment of the semidefinite separation problem and present our regularization approach. In §4.3 we introduce the version of coordinate minimization algorithm used to solve our regularized separation problem. This is followed by a description of our implementation in §4.4. In §4.5, we present an extensive computational study which investigates the effectiveness of our regularization approach, the impact of the proposed relaxations on the performance of BARON, and the performance of several state-of-the-art global optimization solvers on our test problems. Finally, in §4.6, we present conclusions from this work.

Throughout this chapter, we use the same notation presented at the beginning of Chapter 3.

4.1 Construction and theoretical analysis of convex quadratic relaxations

4.1.1 A family of semi-infinite programming relaxations

We start by considering the following reformulation of (4.1):

$$\left.\begin{array}{ll} \min\limits_{x,y,v} & v + q^T x \\ \text{s.t.} & v \geq x^T \left(Q + \text{diag}(d)\right) x - d^T y + \alpha \|Ax - b\|^2 \\ & Ax = b \\ & (x_i, y_i) \in \mathcal{C}_i, \ \forall i \in [n] \end{array}\right\} \tag{4.2}$$

where $y \in \mathbb{R}^n$, $v \in \mathbb{R}$, $d \in \mathbb{R}^n$ is a vector used to perturb the diagonal entries of Q, $\alpha \in \mathbb{R}_{\geq 0}$, and $\mathcal{C}_i := \{(x_i, y_i) \in \mathbb{R}^2 : x_i \in S_i, \ y_i = x_i^2\}$. Define $L_i := \min\{s \in \mathbb{R} : s \in S_i\}$ and $U_i := \max\{s \in \mathbb{R} : s \in S_i\}$. It is simple to show that the convex hull of $\mathcal{C}_i$ is given by (see Proposition 1 in [29]):

$$\text{conv}(\mathcal{C}_i) = \{(x_i, y_i) \in \mathbb{R}^2 : L_i \leq x_i \leq U_i, \ l_i(x_i) \leq y_i \leq u_i(x_i)\} \tag{4.3}$$

where $l_i(\cdot)$ is the tightest convex extension of x_i^2 when x_i is restricted to S_i (see [90] for convex extensions) and $u_i(\cdot)$ is the concave envelope of x_i^2 over $[L_i, U_i]$. By replacing $\mathcal{C}_i$ with $\text{conv}(\mathcal{C}_i)$ in (4.2), we obtain the following relaxation of (4.1):

$$\left.\begin{array}{ll} \min\limits_{(x,y)\in\mathcal{F},v} & v + q^T x \\ \text{s.t.} & v \geq x^T \left(Q + \text{diag}(d)\right) x - d^T y + \alpha \|Ax - b\|^2 \end{array}\right\} \tag{4.4}$$

where $\mathcal{F} = \{x, y \in \mathbb{R}^n : Ax = b, \ (x_i, y_i) \in \text{conv}(\mathcal{C}_i), \ \forall i \in [n]\}$. Let $\mathcal{D}_\alpha := \{d \in \mathbb{R}^n : Q + \text{diag}(d) + \alpha A^T A \succcurlyeq 0\}$. Clearly, (4.4) is a convex problem for any vector $d \in \mathcal{D}_\alpha$. By considering all such vectors, we obtain the following semi-infinite convex program (SICP):

$$\min_{(x,y)\in\mathcal{F},v} \quad v + q^T x \tag{4.5a}$$

$$\text{s.t.} \ \ v \geq x^T \left(Q + \text{diag}(d)\right) x - d^T y + \alpha \|Ax - b\|^2, \ \forall d \in \mathcal{D}_\alpha \tag{4.5b}$$

Since any solution feasible in (4.2), is feasible in (4.5) as well, this SICP is also a relaxation of (4.1). To illustrate this, let $(\bar{x}, \bar{y}, \bar{v})$ be a solution feasible to (4.2). For each $i \in [n]$, we

have $(\bar{x}_i, \bar{y}_i) \in \mathcal{C}_i \subseteq \text{conv}(\mathcal{C}_i)$. Moreover, since $\bar{y}_i = \bar{x}_i^2,\ \forall i \in [n]$, and $A\bar{x} = b$, we have $\bar{v} = \bar{x}^T \left(Q + \text{diag}(d)\right) \bar{x} - d^T \bar{y} + \alpha \|A\bar{x} - b\|^2 = \bar{x}^T Q \bar{x}$.

Observe that the quadratic term $\alpha\|Ax - b\|^2$ in (4.5b) vanishes for any x feasible in (4.5), and is included in (4.5b) to ensure that $Q + \text{diag}(d) + \alpha A^T A$ is positive semidefinite. The next proposition shows that this term need not be included for (4.5) to be convex.

Proposition 4.1. *The following simplified version of the SICP* (4.5)*:*

$$\min_{(x,y)\in\mathcal{F},v} \quad v + q^T x \tag{4.6a}$$

$$\text{s.t.} \quad v \geq x^T \left(Q + \text{diag}(d)\right) x - d^T y,\ \forall d \in \mathcal{D}_\alpha \tag{4.6b}$$

is a convex optimization problem.

Proof. This proof relies on the projection of the feasible set of (4.6) onto the nullspace of A. Let $\mathcal{H} = \{x \in \mathbb{R}^n : Ax = b\}$. Clearly, any point satisfying $Ax = b$ can be expressed as $x = \hat{x} + Zx_z$, where $\hat{x} \in \mathcal{H}$ and $x_z \in \mathbb{R}^{n-m}$. By using this transformation, (4.6) can be equivalently written as:

$$\min_{x_z,y,v} \quad v + q^T \left(\hat{x} + Zx_z\right) \tag{4.7a}$$

$$\text{s.t.} \quad v \geq \left(\hat{x} + Zx_z\right)^T \left(Q + \text{diag}(d)\right) \left(\hat{x} + Zx_z\right) - d^T y,\ \forall d \in \mathcal{D}_\alpha \tag{4.7b}$$

$$L_i \leq \hat{x}_i + e_i^T Z x_z \leq U_i,\ \forall i \in [n] \tag{4.7c}$$

$$l_i \left(\hat{x}_i + e_i^T Z x_z\right) \leq y_i \leq u_i \left(\hat{x}_i + e_i^T Z x_z\right),\ \forall i \in [n] \tag{4.7d}$$

To prove that (4.7) is convex, it suffices to show that $Z^T \left(Q + \text{diag}(d)\right) Z$ is positive semidefinite. By definition, any vector $d \in \mathcal{D}_\alpha$ satisfies:

$$w^T \left(Q + \text{diag}(d) + \alpha A^T A\right) w \geq 0,\ \forall w \in \mathbb{R}^n \tag{4.8}$$

Let $w = Zw_z$, where $w_z \in \mathbb{R}^{n-m}$. For this choice of w, (4.8) becomes

$$w_z^T Z^T \left(Q + \text{diag}(d)\right) Z w_z \geq 0,\ w_z \in \mathbb{R}^{n-m} \tag{4.9}$$

Clearly, (4.9) holds for all vectors $w_z \in \mathbb{R}^{n-m}$. Hence, $Z^T\left(Q + \text{diag}(d)\right) Z$ is positive semidefinite for any $d \in \mathcal{D}_\alpha$. This completes the proof. □

By setting $\alpha = 0$ in (4.6), we obtain the following SICP relaxation of (4.1) which was considered in [29]:

$$\min_{(x,y)\in\mathcal{F},v} \quad v + q^T x \tag{4.10a}$$

$$\text{s.t.} \;\; v \geq x^T\left(Q + \text{diag}(d)\right) x - d^T y, \;\; \forall d \in \mathcal{D} \tag{4.10b}$$

where $\mathcal{D} := \mathcal{D}_0 = \{d \in \mathbb{R}^n : Q + \text{diag}(d) \succcurlyeq 0\}$. The formulation in (4.10) served as a motivation in [29] to develop an algorithm to construct convex relaxations for (4.1) by using a finite number of quadratic cuts of the form (4.10b). As we demonstrate in §4.1.4, a similar algorithm can be devised based on the SICP (4.5).

Note that, in (4.6), the set $\mathcal{D}_\alpha$ is parameterized by the scalar α. An interesting question that arises in this context is how we can choose α to obtain the tightest relaxation in (4.6). This question is addressed by the following proposition.

Proposition 4.2. *Let α_1 and α_2 be real scalars such that $0 \leq \alpha_1 \leq \alpha_2$. Denote by μ_{SICPda1} and μ_{SICPda2} the optimal objective function values in the SICP* (4.6) *for α_1 and α_2, respectively. Define $\mathcal{D}_\infty := \{d \in \mathbb{R}^n : Z^T(Q + \text{diag}(d))Z \succcurlyeq 0\}$. Then, the following holds:*

(i) $\mu_{\text{SICPda2}} \geq \mu_{\text{SICPda1}}$.

(ii) The tightest relaxation of form (4.6) *is obtained when $\alpha \to \infty$.*

(iii) $\lim_{\alpha\to\infty} \mathcal{D}_\alpha = \mathcal{D}_\infty$.

Proof. We start with the proof of (i). Denote $\mathcal{D}_{\alpha_1}$ and $\mathcal{D}_{\alpha_2}$ the sets of diagonal perturbations parametrized by α_1 and α_2, respectively. To prove the claim in (i), it suffices to show that $\mathcal{D}_{\alpha_1} \subseteq \mathcal{D}_{\alpha_2}$. Let $\bar{d} \in \mathcal{D}_{\alpha_1}$. By definition, $\bar{d}$ satisfies:

$$w^T \left(Q + \text{diag}(\bar{d}) + \alpha_1 A^T A\right) w \geq 0, \ \forall w \in \mathbb{R}^n \tag{4.11}$$

As $A^T A \succcurlyeq 0$ and $\alpha_2 - \alpha_1 \geq 0$, we have that $(\alpha_2 - \alpha_1) w^T A^T A w \geq 0, \ \forall w \in \mathbb{R}^n$. This condition combined with (4.11) implies

$$w^T \left(Q + \text{diag}(\bar{d}) + \alpha_2 A^T A\right) w \geq 0, \ \forall w \in \mathbb{R}^n \tag{4.12}$$

It follows that $\bar{d} \in \mathcal{D}_{\alpha_2}$. Hence, $\mathcal{D}_{\alpha_1} \subseteq \mathcal{D}_{\alpha_2}$, which completes the proof of (i). The claim in (ii) follows directly from (i). To prove (iii), we need to show that for any $\bar{d} \in \mathcal{D}_\infty$ the following condition holds:

$$\lim_{\alpha \to \infty} w^T \left(Q + \text{diag}(\bar{d}) + \alpha A^T A\right) w \geq 0, \ \forall w \in \mathbb{R}^n \tag{4.13}$$

Clearly, any $w \in \mathbb{R}^n$ can be written as $w = w_A + Z w_z$, where $w_A \in \text{range}(A^T)$ and $w_z \in \mathbb{R}^{n-m}$. Suppose that $w_A \neq 0$. Then, $w^T A^T A w = w_A^T A^T A w_A > 0$, and it is easy to show that (4.13) holds in the limit as $\alpha \to \infty$. Now assume that $w_A = 0$. Since $AZ = 0$, the left-hand side of (4.13) reduces to $w_z^T Z^T \left(Q + \text{diag}(\bar{d})\right) Z w_z$, which is nonnegative because $\bar{d} \in D_\infty$. This proves the claim in (iii). □

Observe that a direct consequence of Proposition 4.2(i) is that, for any $\alpha > 0$, the bound provided by (4.6) is at least as large as that given by (4.10).

4.1.2 Relationship between the semi-infinite and semidefinite formulations

In [29], it was shown that the semi-infinite program (4.10) is equivalent to the following SDP relaxation of (4.1) (see Section 2 in [29] for details):

$$\min_{(x,y) \in \mathcal{F}, X} \ \langle Q, X \rangle + q^T x \tag{4.14a}$$

$$\text{s.t.} \ \ X - x x^T \succcurlyeq 0 \tag{4.14b}$$

$$X_{ii} = y_i, \ \forall i \in [n] \tag{4.14c}$$

Motivated by this result, in this section we investigate the relationship between the SICP (4.6) and the following SDP relaxation of (4.1):

$$\min_{(x,y)\in\mathcal{F},X} \ \langle Q, X\rangle + q^T x \tag{4.15a}$$
$$\text{s.t.} \ \ X - xx^T \succcurlyeq 0 \tag{4.15b}$$
$$X_{ii} = y_i, \ \forall i \in [n] \tag{4.15c}$$
$$\langle A^T A, X\rangle - 2(A^T b)^T x + b^T b = 0 \tag{4.15d}$$

We start by showing that, for any $\alpha > 0$, the optimal solution of the SICP (4.6) is bounded by the optimal solutions of the SDPs (4.14) and (4.15).

Proposition 4.3. *Assume that $\alpha > 0$ in* (4.6)*. Denote by μ_{SICPda}, μ_{SDPd} and μ_{SDPda} the optimal objective function values in* (4.6)*,* (4.14) *and* (4.15)*, respectively. Then, $\mu_{\text{SDPd}} \leq \mu_{\text{SICPda}} \leq \mu_{\text{SDPda}}$.*

Proof. We start by proving that $\mu_{\text{SDPd}} \leq \mu_{\text{SICPda}}$. Since $\alpha > 0$, Proposition 4.2(i) implies that $\mu_{\text{SICPd}} \leq \mu_{\text{SICPda}}$. By Theorem 1 in [29] we have that $\mu_{\text{SICPd}} = \mu_{\text{SDPd}}$. Hence, $\mu_{\text{SDPd}} \leq \mu_{\text{SICPda}}$. To prove that $\mu_{\text{SICPda}} \leq \mu_{\text{SDPda}}$, it suffices to show that for any $(\bar{x}, \bar{y}, \bar{X})$ feasible in (4.15) and $\bar{d} \in \mathcal{D}_\alpha$, the following condition holds:

$$\langle Q, \bar{X}\rangle \geq \bar{x}^T(Q + \text{diag}(\bar{d}))\bar{x} - \bar{d}^T\bar{y}. \tag{4.16}$$

To this end, consider the following inequality:

$$\langle Q + \text{diag}(\bar{d}) + \alpha A^T A, \bar{X} - \bar{x}\bar{x}^T\rangle \geq 0 \tag{4.17}$$

which is valid by the feasibility of $\bar{x}$ and $\bar{X}$ in (4.15) and the self-duality of the positive semi-definite cone. This inequality can be equivalently written as:

$$\langle Q, \bar{X}\rangle \geq \bar{x}^T(Q + \text{diag}(\bar{d}))\bar{x} - \sum_{i=1}^{n} d_i\bar{X}_{ii} - \alpha\langle A^T A, \bar{X}\rangle + \alpha\bar{x}^T A^T A\bar{x}. \tag{4.18}$$

From the feasibility of $(\bar{x}, \bar{y}, \bar{X})$ in (4.15), it follows that $\bar{X}_{ii} = \bar{y}_i, \forall i \in [n]$, and $\langle A^T A, \bar{X}\rangle = 2b^T A\bar{x} - b^T b$. Then, the inequality in (4.18) becomes:

$$\langle Q, \bar{X} \rangle \geq \bar{x}^T (Q + \text{diag}(\bar{d}))\bar{x} - \bar{d}^T \bar{y} + \alpha (A\bar{x} - b)^T (A\bar{x} - b) \tag{4.19}$$

Since $A\bar{x} - b = 0$, (4.19) is equivalent to (4.16), which completes the proof.

□

Now, we consider the case in which $\alpha \to \infty$ in the SICP (4.6). By Proposition 4.2(iii), the resulting SICP can be written as:

$$\min_{(x,y)\in\mathcal{F},v} \quad v + q^T x \tag{4.20a}$$

$$\text{s.t.} \quad v \geq x^T (Q + \text{diag}(d))\, x - d^T y, \ \forall d \in \mathcal{D}_\infty \tag{4.20b}$$

Proposition 4.2(ii) implies that (4.20) is the tightest relaxation of the form (4.6) that can be constructed for (4.1). Moreover, from Proposition 4.3, we know that (4.15) provides an upper bound on the optimal solution of (4.20). Therefore, an important question is the existence of conditions under which these two relaxations are equivalent. This question is addressed by the following theorem.

Theorem 4.1. *Let* μ_{SDPda} *and* $\mu_{\text{SICPda}\infty}$ *denote the optimal objective function values in* (4.15) *and* (4.20)*, respectively. Define* $\mathcal{X} := \{x \in \mathbb{R}^n : Ax = b, \ L_i < x_i < U_i, \ \forall i \in [n]\}$*. Assume that the following conditions hold:*

(i) $\exists\, x^{(1)}, x^{(2)} \in \mathbb{R}^n$ *such that* $Ax^{(1)} = Ax^{(2)} = b$ *and* $x_i^{(1)} \neq x_i^{(2)}, \ \forall\, i \in [n]$.

(ii) $\exists\, \widetilde{x} \in \mathcal{X}$ *such that* $l_i(\widetilde{x}_i) = \widetilde{x}_i^2$ *and* $l_i(\widetilde{x}_i) < u_i(\widetilde{x}_i), \ \forall i \in [n]$.

Then, $\mu_{\text{SDPda}} = \mu_{\text{SICPda}\infty}$.

Proof. In [29], the equivalence between (4.10) and (4.14) was established by applying the strong duality theorem to this SDP. Unlike (4.14), the SDP (4.15) does not admit a strictly feasible solution. To illustrate this, note that for any x satisfying $Ax = b$, (4.15d) can be equivalently written as:

$$\langle A^TA, X\rangle - \left(2A^Tb\right)^T x + b^Tb + \langle A^TA, xx^T\rangle - \langle A^TA, xx^T\rangle = 0 \tag{4.21a}$$
$$\implies \langle A^TA, X - xx^T\rangle = 0. \tag{4.21b}$$

From the satisfaction of the positive semidefinite constraint in (4.15b), self-duality of the positive semidefinite cone and (4.21b), it follows that for any feasible point $(\bar{x}, \bar{y}, \bar{X})$ to the SDP in (4.15) we have

$$\bar{X} = \bar{x}\bar{x}^T + Z\bar{X}_zZ^T, \text{ where } \bar{X}_z \in \mathbb{S}^{n-m}. \tag{4.22}$$

This implies that the maximum rank of $\bar{X}$ cannot exceed $n-m+1$, and as a result, there exists no feasible point for which (4.15b) is strictly satisfied. It follows that the strong duality theorem does not apply to (4.15). Therefore, in this proof, we will rely on an auxiliary SDP given by:

$$\min_{x_z, y, X_z} \quad \left\langle Q, ZX_zZ^T\right\rangle + (2Q\hat{x} + q)^T (Zx_z) + \hat{x}^TQ\hat{x} + q^T\hat{x} \tag{4.23a}$$
$$\text{s.t.} \quad X_z - x_zx_z^T \succcurlyeq 0 \tag{4.23b}$$
$$\hat{x}_i^2 + 2\hat{x}_i\left(e_i^TZx_z\right) + e_i^TZX_zZ^Te_i = y_i, \ \forall i \in [n] \tag{4.23c}$$
$$L_i \leq \hat{x}_i + e_i^TZx_z \leq U_i, \ \forall i \in [n] \tag{4.23d}$$
$$l_i\left(\hat{x}_i + e_i^TZx_z\right) \leq y_i \leq u_i\left(\hat{x}_i + e_i^TZx_z\right), \ \forall i \in [n] \tag{4.23e}$$

where $x_z \in \mathbb{R}^{n-m}$ and $X_z \in \mathbb{S}^{n-m}$, and $\hat{x}$ satisfies $A\hat{x} = b$. Denote by μ_{SDPdaz} and μ_{DSDPdaz}, the optimal objective function values of the SDP (4.23) and its dual, respectively. To prove $\mu_{\text{SDPda}} = \mu_{\text{SICPda}_\infty}$, we proceed in three steps:

(a) We show that the SDPs (4.15) and (4.23) are equivalent. This implies that $\mu_{\text{SDPda}} = \mu_{\text{SDPdaz}}$.

(b) We demonstrate that, under assumptions (i) and (ii), there exists a strictly feasible solution for the SDP (4.23). This allows us to apply the strong duality theorem to (4.23) establishing that $\mu_{\text{SDPdaz}} = \mu_{\text{DSDPdaz}}$.

(c) We prove that the dual of the SDP (4.23) provides a lower bound on the optimal solution of the SICP (4.20), i.e., $\mu_{\text{SICPda}\infty} \geq \mu_{\text{DSDPdaz}} = \mu_{\text{SDPdaz}} = \mu_{\text{SDPda}}$. This result combined with Proposition 4.3 completes the proof by showing that $\mu_{\text{SDPda}} = \mu_{\text{SICPda}\infty}$.

We start with the proof of (a). We show that given a feasible point for (4.15) we can construct a feasible point for (4.23) with equal objective and the vice-versa. Suppose $(\bar{x}, \bar{y}, \bar{X})$ is feasible to (4.15). From (4.22), we have that $\exists\, \bar{X}_z \in \mathbb{S}^{n-m}$ such that $\bar{X} = \bar{x}\bar{x}^T + Z\bar{X}_z Z^T$ and $\bar{X}_z \succcurlyeq 0$. Then it is readily verified that $\hat{x} = \bar{x}$, $x_z = 0$, $y = \bar{y}$, and $X_z = \bar{X}_z$ is feasible to (4.23). Further, the objective values of the two SDPs are also identical. Now, assume that $(\breve{x}_z, \breve{y}, \breve{X}_z)$ is feasible to (4.23). Then it is easy to check that $x = \hat{x} + Z\breve{x}_z$, $y = \breve{y}$, and $X = \hat{x}\hat{x}^T + \hat{x}(Z\breve{x}_z)^T + (Z\breve{x}_z)\hat{x}^T + Z\breve{X}_z Z^T$ is feasible to (4.15) with equal objective. This proves the claim in (a).

Next, we prove (b). Define the scalar δ as:

$$\delta := \min_{i\in[n]} \frac{u_i(\tilde{x}_i) - l_i(\tilde{x}_i)}{e_i^T Z Z^T e_i}. \tag{4.24}$$

From assumption (i), it follows that $e_i^T Z \neq 0,\ \forall i \in [n]$. Moreover, assumption (ii) implies that $u_i(\tilde{x}_i) - l_i(\tilde{x}_i) > 0,\ \forall i \in [n]$. Hence, $0 < \delta < \infty$. Let $\hat{x} = \tilde{x}$, $\bar{x}_z = 0$, $\bar{y}_i = \tilde{x}_i^2 + \epsilon e_i^T Z Z^T e_i,\ i \in [n]$, and $\bar{X}_z = \epsilon I_{n-m}$, where $0 < \epsilon < \delta$. It is simple to check that this choice is feasible in the SDP (4.23), and further, the inequalities (4.23b), (4.23d) and (4.23e) are satisfied strictly. Thus, Slater's constraint qualification holds for (4.23) and we have that $\mu_{\text{SDPdaz}} = \mu_{\text{DSDPdaz}}$.

Now we prove (c). Let $d_i \in \mathbb{R},\ i \in [n]$, be the multipliers associated with the constraints (4.23c). Then, the dual of (4.23) can be written as:

$$\max_{d\in\mathbb{R}^n} \left\{ \begin{array}{ll} \min\limits_{x_z, y, X_z} & \langle Q_{d,Z}, X_z\rangle + q_{d,\hat{x}}^T (Z x_z) + k_{d,\hat{x}} - d^T y \\ \text{s.t.} & X_z - x_z x_z^T \succcurlyeq 0 \\ & L_i \leq \hat{x}_i + e_i^T Z x_z \leq U_i,\ \forall i \in [n] \\ & l_i\left(\hat{x}_i + e_i^T Z x_z\right) \leq y_i \leq u_i\left(\hat{x}_i + e_i^T Z x_z\right),\ \forall i \in [n] \end{array} \right\} \tag{4.25}$$

where $Q_{d,Z} = Z^T Q_d Z$, $Q_d = Q + \text{diag}(d)$, $q_{d,\hat{x}} = 2Q_d\hat{x} + q$, and $k_{d,\hat{x}} = \hat{x}^T Q_d \hat{x} + q^T\hat{x}$. For the minimization problem in (4.25) to be bounded below, we need to choose d such that $Q_{d,Z} \succcurlyeq 0$. This restriction on d implies that $X_z = x_z x_z^T$ holds at any optimal solution to the inner minimization problem. As a result, the dual in (4.25) can be simplified as:

$$\max_{d\in\mathcal{D}_\infty} \left\{ \begin{array}{ll} \min\limits_{x_z,y} & (\hat{x}+Zx_z)^T Q_d (\hat{x}+Zz_x) + q^T(\hat{x}+Zx_z) - d^T y \\ \text{s.t.} & L_i \le \hat{x}_i + e_i^T Z x_z \le U_i,\ \forall i \in [n] \\ & l_i\left(\hat{x}_i + e_i^T Z x_z\right) \le y_i \le u_i\left(\hat{x}_i + e_i^T Z x_z\right),\ \forall i \in [n]. \end{array} \right\} \tag{4.26}$$

Since strong duality holds for (4.23) and its dual (4.26), both problems attain their optimal objective functions values. This implies that $\exists\, d^* \in \mathcal{D}_\infty$ such that:

$$\mu_{\text{DSDPdaz}} = \left. \begin{array}{ll} \min\limits_{x_z,y,v} & v + q^T(\hat{x}+Zx_z) \\ \text{s.t.} & v \ge (\hat{x}+Zx_z)^T (Q + \text{diag}(d^*))(\hat{x}+Zx_z) - {d^*}^T y \\ & L_i \le \hat{x}_i + e_i^T Z x_z \le U_i,\ \forall i \in [n] \\ & l_i\left(\hat{x}_i + e_i^T Z x_z\right) \le y_i \le u_i\left(\hat{x}_i + e_i^T Z x_z\right),\ \forall i \in [n] \end{array} \right\} \tag{4.27}$$

It is easy to show that (4.27) is a relaxation of (4.20). To illustrate this, note that by replacing x and $\mathcal{D}_\infty$ in (4.20) with $\hat{x} + Zx_z$ and $\{d^*\}$, respectively, we obtain (4.27). Hence, $\mu_{\text{SICPda}\infty} \ge \mu_{\text{DSDPdaz}}$. Combining this condition with the results of (a), (b) and Proposition 4.3, we obtain $\mu_{\text{SICPda}\infty} = \mu_{\text{SDPda}}$, which completes the proof.

□

Observe that if assumption (i) in Theorem 4.1 does not hold, then there must exist a set $J \subseteq [n]$ such that x_i is fixed for $i \in J$. In this case, the corresponding variables can be eliminated to obtain a reduced problem. Note also that the satisfaction of assumption (ii) in Theorem 4.1 depends on the form of the functions $l_i(x_i)$ and $u_i(x_i)$. It is easy to show that $u_i(x) = (L_i + U_i)x_i - L_i U_i$. On the other hand, the choice of $l_i(x_i)$ depends on the form of the set S_i. If, for a given $i \in [n]$, $l_i(x_i) = u_i(x_i)$, then assumption (ii) fails to hold. This occurs, for example, when x_i is binary, i.e., $S_i = \{0, 1\}$, because in this case $l_i(x_i) = u_i(x_i) = x_i$.

4.1.3 Further insights into the semidefinite relaxation

Observe that the SDP relaxation (4.15) can be derived from (4.14) by adding the valid equality $(Ax - b)^T(Ax - b) = 0$ and lifting it into the space of (x, X). Clearly, we can construct other SDP relaxations for (4.1) by including other classes of constraints derived from $Ax = b$. In general, we can apply the following procedure:

(R1) identify a (possibly empty) set $\mathcal{J}$ of quadratic functions of the form $f_j(x) = x^T C_j x + c_j^T x + \gamma_j$, where $C_j \in \mathbb{S}^n, c_j \in \mathbb{R}^n, \gamma_j \in \mathbb{R}$, such that $f_j(x) = 0$ for $x \in \Omega := \{x \in \mathbb{R}^n \mid Ax = b\}$;

(R2) construct an SDP relaxation for (4.1) as

$$\begin{aligned} \min_{(x,y)\in\mathcal{F},X} \quad & \langle Q, X\rangle + q^T x && \text{(4.28a)}\\ \text{s.t.} \quad & X - xx^T \succcurlyeq 0 && \text{(4.28b)}\\ & X_{ii} = y_i, \ \forall i \in [n] && \text{(4.28c)}\\ & \langle C_j, X\rangle + c_j^T x + \gamma_j = 0, \ \forall j \in \mathcal{J} && \text{(4.28d)} \end{aligned}$$

where the constraints (4.28d) are obtained by lifting the valid equalities $x^T C_j x + c_j^T x + \gamma_j = 0, \ \forall j \in \mathcal{J}$ into the space of (x, X).

Note that this procedure is similar to the recipe introduced in §3.3.4. As stated in §3.3.4, there are different types of functions $f_j(x)$ that satisfy the condition in (R1). As a result, a natural question in this context is whether we can improve on the bound given by (4.15) when restricted to the class of relaxations in (4.28). We address this question in the remainder of this section by demonstrating that (4.15) is the best relaxation among the class of relaxations in (4.28).

We start by showing that the feasible set of (4.28) is contained within that of (4.15), and provide conditions on the choice of quadratic functions in $\mathcal{J}$ for the two sets to be identical.

Proposition 4.4. *Let $\mathcal{F}_{\text{SDPda}}$ and $\mathcal{F}_{\text{SDPdaJ}}$ denote the feasible regions of the SDPs in* (4.15) *and* (4.28)*, respectively. Then, the following holds:*

(i) $\mathcal{F}_{\text{SDPda}} \subseteq \mathcal{F}_{\text{SDPdaJ}}$.

(ii) If $\exists\, \omega_j, j \in \mathcal{J}$ *such that* $\sum_{j\in\mathcal{J}} \omega_j W_j = A^T$ *then* $\mathcal{F}_{\text{SDPda}} = \mathcal{F}_{\text{SDPdaJ}}$.

Proof. In this proof we will follow the same line of arguments used in the proof of Proposition 3.12. We first prove (i). From (4.22), any $(\bar{x}, \bar{y}, \bar{X}) \in \mathcal{F}_{\text{SDPda}}$ satisfies $\bar{X} = \bar{x}\bar{x}^T + Z\bar{X}_z Z^T$, where $\bar{X}_z \in \mathbb{S}^{n-m}$. For any $(\bar{x}, \bar{y}, \bar{X}) \in \mathcal{F}_{\text{SDPda}}$ and $\forall j \in \mathcal{J}$:

$$\begin{aligned} & \langle C_j, \bar{X}\rangle + c_j^T\bar{x} + \gamma_j && \text{(4.29a)}\\ = \ & \langle C_j, \bar{X} - \bar{x}\bar{x}^T\rangle + \bar{x}^T C_j\bar{x} + c_j^T\bar{x} + \gamma_j && \text{(4.29b)}\\ = \ & \langle C_j, \bar{X} - \bar{x}\bar{x}^T\rangle = \langle A^T W_j^T + W_j A, Z\bar{X}_z Z^T\rangle = 0 && \text{(4.29c)} \end{aligned}$$

where (4.29b) follows from adding and subtracting $\bar{x}^T C_j\bar{x}$, the first equality in (4.29c) follows from (R1), the second equality in (4.29c) from Proposition 3.10 and the final equality from the fact that Z is a basis for the nullspace of A. Thus $(\bar{x}, \bar{X}) \in \mathcal{F}_{\text{SDPdaJ}}$ proving the claim in (i). Now, we prove the claim in (ii). Assume that there exist $\omega_j, j \in \mathcal{J}$ such that the condition in (ii) holds. By performing a linear combination of the inequalities in (4.28d) using ω_j, we obtain that for any $(\bar{x}, \bar{y}, \bar{X}) \in \mathcal{F}_{\text{SDPdaJ}}$:

$$\begin{aligned} 0 = \ & \sum_{j\in\mathcal{J}} \omega_j \left(\langle C_j, \bar{X}\rangle + c_j^T\bar{x} + \gamma_j\right) && \text{(4.30a)}\\ = \ & \sum_{j\in\mathcal{J}} \omega_j \left(\langle C_j, \bar{X} - \bar{x}\bar{x}^T\rangle + \bar{x}^T C_j\bar{x} + c_j^T\bar{x} + \gamma_j\right) && \text{(4.30b)}\\ = \ & \sum_{j\in\mathcal{J}} \omega_j \langle C_j, \bar{X} - \bar{x}\bar{x}^T\rangle = 2\langle A^T A, \bar{X} - \bar{x}\bar{x}^T\rangle && \text{(4.30c)} \end{aligned}$$

where (4.30b) follows from adding and subtracting $\bar{x}^T C_j\bar{x}$, the first equality in (4.30c) follows from (R1), the second equality in (4.30c) from Proposition 3.10 and the condition in (ii). Thus $(\bar{x}, \bar{X}, \bar{y}) \in \mathcal{F}_{\text{SDPda}}$ proving the claim in (ii).

□

Now, we are ready to prove the main result of this section.

Theorem 4.2. *Suppose that $\mathcal{J}$ is chosen such that (R1) holds. Denote by $\mu_{\text{SICPda}\infty}$, μ_{SDPda} and μ_{SDPdaJ} the optimal objective function values in* (4.20), (4.15) *and* (4.28), *respectively. Then,*

(i) $\mu_{\text{SDPdaJ}} \leq \mu_{\text{SDPda}}$

(ii) If the assumptions in Theorem 4.1 hold, then $\mu_{\text{SDPdaJ}} \leq \mu_{\text{SICPda}\infty}$.

Proof. The claim in (i) follows from Proposition 4.4. If the assumptions in Theorem 4.1 hold, then $\mu_{\text{SDPda}} = \mu_{\text{SICPda}\infty}$ and the claim in (ii) follows from (i).

□

4.1.4 Cutting Surface Algorithm

By replacing $\mathcal{D}_\alpha$ in (4.6b) with a set $\mathcal{D}_\alpha^{(k)}$ of finite dimension, we devise an iterative cutting surface algorithm which allows us to derive convex QCP relaxations for (4.1). At the k-th iteration of this algorithm, the following relaxation is solved:

$$\min_{(x,y)\in\mathcal{F},v} \quad v + q^T x \tag{4.31a}$$

$$\text{s.t.} \quad v \geq x^T \left(Q + \text{diag}(d)\right) x - d^T y, \ \forall d \in \mathcal{D}_\alpha^{(k)} \tag{4.31b}$$

This iterative approach is described in Algorithm 6. Note that at each iteration of this algorithm, a separation problem is solved in order to construct a new quadratic cut of the form (4.31b). Observe also that the parameter α is fixed during this algorithm. Proposition 4.2 suggests that we should select a large value of α in order to improve the bound given by (4.31). We describe a procedure to determine such a value of α in §4.4.

Another interesting observation about Algorithm 6 is that the parameter α only appears in the separation problem (see §4.2 for details), since the term $\alpha\|Ax - b\|^2$ is not included

Algorithm 6 A cutting surface procedure to derive QCP relaxations for (4.1)

Input: Q, q, A, b, and algebraic expressions for $l(\cdot)$ and $u(\cdot)$.
Output: A lower bound μ_{QCPda} on the optimal solution of (4.1).
If $m = 0$ **then**
    Set $\alpha = 0$
Else
    Choose a positive value of α according to the procedure described in §4.4.
End If
Set $\mathcal{D}_\alpha^{(1)} = \{d^{(1)}\}$, where $d^{(1)} \in \mathbb{R}^n$ is a perturbation for which (4.31) is convex.
Solve (4.31). Let $(\bar{x}, \bar{y}, \bar{v})$ be an optimal solution to this relaxation.
Set $\mu_{\text{QCPda}} = \bar{v} + q^T \bar{x}$.
For $k = 2$ to MaxNC
    Solve a separation problem to find a new perturbation $d^{(k)}$.
    If the convex quadratic cut (4.31b) with $d = d^{(k)}$ violates $(\bar{x}, \bar{y}, \bar{v})$ **then**
        $\mathcal{D}_\alpha^{(k+1)} \leftarrow \mathcal{D}_\alpha^{(k)} \cup \{d^{(k)}\}$
        Solve (4.31). Let $(\bar{x}, \bar{y}, \bar{v})$ be an optimal solution to this relaxation.
        Set $\mu_{\text{QCPda}} = \bar{v} + q^T \bar{x}$.
    Else
        Terminate
    End If
End For

in (4.31b). This is particularly advantageous because it allows us to preserve the sparsity pattern of Q in the quadratic constraints (4.31b). In addition, by dropping this term from (4.31b), we prevent the relaxation from becoming ill-conditioned for large values of α.

To construct the first relaxation of Algorithm 6, we need to specify an initial perturbation $d^{(1)}$. For simplicity, we set $d^{(1)} = \mu \mathbb{1}$, where $\mu \in \mathbb{R}_{\geq 0}$. For this choice of $d^{(1)}$, (4.31) becomes:

$$\min_{(x,y)\in\mathcal{F},v} \quad v + q^T x \tag{4.32a}$$

$$\text{s.t.} \quad v \geq x^T (Q + \mu I_n) x - \mu \mathbb{1}^T y \tag{4.32b}$$

In order to select μ, we consider two cases depending on the value of m:

(i) $m = 0$. In this case (4.1) is an unconstrained optimization problem. We run Algorithm 6 only if Q is indefinite and set $\mu = -\lambda_{\min}(Q)$. It is easy to verify that this choice

of μ renders $Q + \mu I_n$ positive semidefinite, thus ensuring the convexity of (4.32).

(ii) $m > 0$. In this case (4.1) contains at least one equality constraint. We run Algorithm 6 only if $Z^T Q Z$ is indefinite and set $\mu = -\lambda_{\min}(Z^T Q Z)$. It is simple to check that, for this choice of μ, (4.32) is a convex problem. To this end, note that the projection of (4.32) onto the nullspace of A can be obtained from (4.7) by considering a single quadratic constraint in (4.7b) and setting $d = \mu \mathbb{1}$. The resulting problem is convex when $Z^T (Q + \mu I_n) Z \succcurlyeq 0$. It is easy to verify that our choice of μ satisfies this condition.

Since (4.32) contains a single quadratic constraint and $\mu \geq 0$, we can eliminate the variables y and v, and rewrite this QCP as the following quadratic program:

$$\min_{x \in \mathcal{X}} \quad x^T (Q + \mu I_n) x + q^T x - \mu \sum_{i=1}^{n} u_i(x_i) \tag{4.33}$$

By setting $\mu = -\lambda_{\min}(Q)$ and $\mu = -\lambda_{\min}(Z^T Q Z)$ in (4.33), we obtain the eigenvalue relaxation and the eigenvalue relaxation in the nullspace of A, respectively. These relaxations were introduced in §3.3.1 and 3.3.3, respectively. Note that in (ii), we could use the same initial perturbation as in (i). However, as shown in Proposition 3.7, the perturbation given in (ii) can lead to a tighter initial bound.

4.2 Analysis and regularization of the separation problem

We start this section by presenting the separation problem solved in Algorithm 6. Let $(\bar{x}, \bar{y}, \bar{v})$ be an optimal solution to the relaxation (4.31). Then, in order to construct a quadratic inequality of the form (4.31b) that is maximally violated by $(\bar{x}, \bar{y}, \bar{v})$, we can solve the following optimization problem:

$$\sup_{d \in \mathcal{D}_\alpha} \quad \sum_{i=1}^{n} \left(\bar{x}_i^2 - \bar{y}_i \right) d_i \tag{4.34}$$

Note that this SDP is parametrized by the value of α determined at the beginning of Algorithm 6. Observe also that the separation problem considered in [29] is a particular instance of (4.34), obtained by setting $\alpha = 0$ in (4.34). For the remainder of this section, we will cast (4.34) as:

$$\inf_{d \in \mathcal{D}_\alpha} \quad \eta^T d \tag{4.35}$$

where $\eta_i := \bar{y}_i - \bar{x}_i^2, \ \forall i \in [n]$. As shown in [29], the attainment of the infimum in (4.35) is not guaranteed, and in fact, may depend on the problem data. We illustrate this behavior through the following example.

Example 4.1. *Let* $Q = \left[\begin{smallmatrix} 0 & 2 \\ 2 & -1 \end{smallmatrix}\right]$, $A = \left[\begin{smallmatrix} 0 & 1 \end{smallmatrix}\right]$, *and* $\alpha = 1$ *in* (4.35). *Consider the following cases:*

(i) $\bar{x} = \bar{y} = \left[\begin{smallmatrix} 0.5 & 0.5 \end{smallmatrix}\right]^T$. *In this case, the infimum in* (4.35) *is attained for* $d_1^* = d_2^* = 2$.

(ii) $\bar{x} = \bar{y} = \left[\begin{smallmatrix} 0.4 & 0 \end{smallmatrix}\right]^T$. *In this case, the infimum in* (4.35) *cannot be attained since it occurs as* $d_1 \to 0$ *and* $d_2 \to \infty$.

To further analyze the attainment of (4.35), we construct the dual of this SDP. Let $Y \in \mathbb{S}^n$ be the matrix of dual variables associated with the semidefinite constraint $Q + \text{diag}(d) + \alpha A^T A \succcurlyeq 0$. Then, the dual of (4.35) is:

$$\sup_{Y \succcurlyeq 0} \quad -\langle Q + \alpha A^T A, Y \rangle \tag{4.36a}$$

$$\text{s.t.} \ \ Y_{ii} = \eta_i, \ \forall i \in [n] \tag{4.36b}$$

Let $P := \{i \in [n] \ : \ \bar{y}_i = \bar{x}_i^2\}$. From (4.36b), it follows that $Y_{ii} = 0, \ \forall i \in P$. Hence, if $P \neq \emptyset$, (4.36) does not admit a strictly feasible solution and, as a result, strong duality may not hold for the primal-dual pair (4.35),(4.36). Note that in Example (4.1), we have $P = \emptyset$ for (i), and $P = \{2\}$ for (ii).

Observe that in the separation step of Algorithm 6, we do not need to solve (4.35) to optimality, but rather derive quadratic cuts that can be used to tighten a relaxation of

the form (4.31). As a result, we can replace (4.35) with a regularized separation problem constructed in a way such that its optimum is always attained. One option is to regularize (4.35) as discussed in [29]. To this end, we can add to the objective function of (4.35) the term $\lambda \sum_{i=1}^{n} [d_i]_+$, where $\lambda = \sum_{i=1}^{n} (\bar{y}_i - \bar{x}_i^2)$, and $[d_i]_+$ is equal to d_i if $d_i > 0$, and 0 otherwise. This leads to the following regularized separation problem:

$$\inf_{d \in \mathcal{D}_\alpha} \quad \eta^T d + \lambda \sum_{i=1}^{n} [d_i]_+ \tag{4.37}$$

It is simple to show that the infimum in (4.37) is always attained (see Proposition 3 in [29] for details). Note that the parameter λ is always positive unless the current relaxation is exact.

In this chapter, we propose an alternative regularization for (4.35). We modify this problem by adding to the objective function the quadratic term $\rho d^T d$, where ρ is a positive scalar. The resulting regularized separation problem is given by:

$$\inf_{d \in \mathcal{D}_\alpha} \quad \eta^T d + \rho d^T d \tag{4.38}$$

As we show in the following proposition, this regularization also gives rise to a separation problem for which the optimum is always attained.

Proposition 4.5. *Let $\rho > 0$ in* (4.38)*. Then, the optimal solution to the semidefinite program* (4.38) *is always attained at some finite point.*

Proof. This proof relies on strong duality holding for (4.38) and its dual. Denote by $Y \in \mathbb{S}^n$ the matrix of dual variables associated with the semidefinite constraint $Q + \text{diag}(d) + \alpha A^T A \succcurlyeq 0$. Then, the dual of (4.38) is:

$$\sup_{Y \succcurlyeq 0} \quad -\langle Q + \alpha A^T A, Y \rangle - \frac{1}{4\rho} \sum_{i=1}^{n} (Y_{ii} - \eta_i)^2 \tag{4.39}$$

Now, let $\bar{d} = \mu \mathbb{1}$ and $\bar{Y} = I_n$, where $\mu > -\min(0, \lambda_{\min}(Q + \alpha A^T A))$. Clearly, $\bar{d}$ and $\bar{Y}$ are strictly feasible in (4.38) and (4.39), respectively. Hence, strong duality holds, and both SDPs attain their optimal solutions. □

4.3 Solution of the regularized separation problem

In this section, we described the algorithm that we use to solve the regularized separation problem proposed in §4.2. This algorithm is a modification of the coordinate descent method introduced in [29]. To solve (4.38), our algorithm operates on the following penalized log-det problem:

$$\begin{array}{ll} \inf\limits_{d} & f(d;\sigma) := G(d) - \sigma \text{log-det}\left(Q + \text{diag}(d) + \alpha A^T A\right) \\ \text{s.t.} & Q + \text{diag}(d) + \alpha A^T A \succ 0 \end{array} \tag{4.40}$$

where $G(d) = \sum_{i=1}^{n} g_i(d_i)$, $g_i(d_i) = \eta_i d_i + \rho d_i^2,\ \forall i \in [n]$, and σ is a positive penalty parameter. The optimality condition for (4.40) can be expressed as:

$$\nabla f(d;\sigma) = 0, \quad Q + \text{diag}(d) + \alpha A^T A \succ 0 \tag{4.41}$$

where the gradient of $f(d;\sigma)$ has the form:

$$\nabla f(d;\sigma) = \nabla G(d) - \sigma \text{diag}\left(\left[Q + \text{diag}(d) + \alpha A^T A\right]^{-1}\right) \tag{4.42}$$

with $\nabla G(d)_i = \eta_i + 2\rho d_i,\ \forall i \in [n]$. At each iteration of this algorithm, we update a feasible vector $\bar{d}$ and an inverse matrix $V := \left[Q + \text{diag}(\bar{d}) + \alpha A^T A\right]^{-1}$. Based on the optimality condition (4.41), we perform coordinate minimization by choosing an index i which corresponds to the entry of $\nabla f(\bar{d};\sigma)$ with the largest magnitude:

$$i = \arg\max_{j=1,\ldots,n} \left\{ \left| \nabla f(\bar{d};\sigma)_j \right| \right\} \tag{4.43}$$

This choice of i leads to the following one-dimensional minimization problem:

$$\Delta d_i^* \in \operatorname*{argmin}_{\Delta d_i} \left\{ f(\bar{d} + \Delta d_i e_i;\sigma) \ :\ Q + \text{diag}(\bar{d} + \Delta d_i e_i) + \alpha A^T A \succ 0 \right\} \tag{4.44}$$

As we show in the next proposition, it is possible to find a closed-form expression for the optimal solution of (4.44).

Proposition 4.6. *The optimal solution to the one-dimensional problem* (4.44) *is:*

$$\Delta d_i^* = -(\phi_i + \tau_i) + \sqrt{(\phi_i - \tau_i)^2 + \kappa} \tag{4.45}$$

where $\phi_i = 1/(2V_{ii})$, $\tau_i = (\eta_i + 2\rho\bar{d}_i)/(4\rho)$ *and* $\kappa = \sigma/(2\rho)$.

Proof. At the optimal solution of (4.44) the following holds:

$$\frac{\partial f(\bar{d} + \Delta d_i e_i; \sigma)}{\partial \Delta d_i} = \eta_i + 2\rho(\bar{d}_i + \Delta d_i) - \frac{\sigma V_{ii}}{1 + \Delta d_i V_{ii}} = 0 \tag{4.46}$$

By solving for Δd_i in (4.46), we obtain the roots $\Delta d_i^{*(\pm)} = -(\phi_i + \tau_i) \pm \sqrt{(\phi_i - \tau_i)^2 + \kappa}$. It is easy to show that $\Delta d_i^{*(+)}$ is the only one of these two solutions that is feasible in (4.44). By applying Lemma 1 from [29], we have:

$$Q + \text{diag}(\bar{d} + \Delta d_i e_i) + \alpha A^T A \succ 0 \iff \Delta d_i > -1/V_{ii} \tag{4.47}$$

Therefore, for $\Delta d_i^{*(+)}$ to be feasible in (4.44) we must have $z + \sqrt{z^2 + \kappa} > 0$, where $z = (\phi_i - \tau_i)$. It is simple to check that this condition is always satisfied. To this end, note that $\kappa > 0$. This implies $z + \sqrt{z^2 + \kappa} > z + |z| \geq 0, \ \forall z \in \mathbb{R}$. Using a similar analysis, we can show that $\Delta d_i^{*(-)}$ is infeasible in (4.44).

□

After solving the one-dimensional problem (4.44), we update $\bar{d}$ as:

$$\bar{d} \leftarrow \bar{d} + \Delta d_i^* e_i \tag{4.48}$$

and update V using the Sherman-Morrison formula:

$$V \leftarrow V - \frac{\Delta d_i^* V_{\cdot i} V_{\cdot i}^T}{1 + \Delta d_i^* V_{ii}} \tag{4.49}$$

In our numerical experiments, we noticed that, for very small values of ρ, some of the entries of $\bar{d}$ become very large after performing the update in (4.48). This is not surprising because: (i) for very small values of ρ, the regularized separation problem (4.38) exhibits a similar behavior to the original separation problem (4.35), and (ii) as discussed in 4.2, the finite attainment of (4.35) is not guaranteed. To address this issue, we propose an adaptive strategy in order to adjust the value of ρ used in (4.38). At a given iteration of

our algorithm, after performing the update in (4.48), we determine the entry of $\bar{d}$ with the largest magnitude, i.e.:

$$\bar{d}_{\max} = \max_{j=1,\dots,n} \{|\bar{d}_j|\} \tag{4.50}$$

If $\bar{d}_{\max}$ is at least an order of magnitude larger than the smallest eigenvalue of $Q+\alpha A^T A$, we increase ρ by multiplying it by a factor ρ_{upd}, and restart the coordinate descend algorithm with this new value of ρ. In practice, this adaptive strategy only requires a few restarts before finding a suitable value of ρ. For the first run of our algorithm, we set $\rho = \rho_{\text{init}}$.

Once (4.40) has been solved within a given precision, we update the penalty parameter σ according to the following condition:

$$\sigma \leftarrow \max\{\sigma_{\min}, \sigma_{\text{upd}}\sigma\} \quad \text{if} \quad \frac{\|\nabla f(\bar{d};\sigma)\|_2}{\|\eta\|_2} \leq \epsilon_{\text{upd}} \tag{4.51}$$

where $\nabla f(\bar{d};\sigma)$ is used as a measure of optimality. We check the relative improvement in the objective function of (4.38) every $\omega_{\text{check}} n$ iterations, and terminate our algorithm if this relative improvement is smaller than ϵ_{check}.

Our coordinate minimization strategy is summarized in Algorithm 7. Note that if $Q + \alpha A^T A$ is positive semidefinite, then $Z^T Q Z$ is also positive semidefinite and (4.1) is convex when restricted to the nullspace of A. In this case, it suffices to solve the continuous relaxation of (4.1) in order to obtain a lower bound. As a result, the separation procedure outlined in Algorithm 7 is only used if $Q + \alpha A^T A$ is indefinite. We start Algorithm 7 with an initial perturbation $\hat{d} = -1.5\lambda_{\min}(Q + \alpha A^T A)\mathbb{1}$. We set MaxIter $= 500n$, $\sigma_{\min} = 10^{-5}$, $\sigma_{\text{upd}} = 0.8$, $\epsilon_{\text{upd}} = 0.03$, $\omega_{\text{check}} = 10$ and $\epsilon_{\text{check}} = 10^{-4}$. We initialize ρ_{init} as:

$$\rho_{\text{init}} = 10^{-4} \frac{10^{4\lfloor \log_{10}(\delta_{\max}) \rfloor}}{\max\{1, \lfloor Q_{\max}/100 \rfloor Q_{\max}\}} \tag{4.52}$$

where $Q_{\max}$ and $\delta_{\max}$ are given by:

$$Q_{\max} = \max_{i=1,\dots,n,\ j=i,\dots,n} \{|Q_{ij}|\}, \quad \delta_{\max} = \max_{i=1,\dots,n} \{U_i - L_i\} \tag{4.53}$$

and set $\rho_{\text{upd}} = 10$. The initial value of σ_{init} is determined as:

$$\sigma_{\text{init}} = \text{median} \left\{ \left| \frac{\eta_i + 2\rho\hat{d}_i}{V_{ii}} \right| \right\}_{i=1}^{n} \tag{4.54}$$

Algorithm 7 Barrier coordinate minimization algorithm used to solve the smooth regularized separation problem (4.38)

Input: Q, A, α, and an optimal solution $(\bar{x}, \bar{y}, \bar{v})$ to (4.31).
Output: A vector $\bar{d}$ that solves (4.38).
Set $\rho = \rho_{\text{init}}$, where ρ_{init} is determined using (4.52).
Set $\bar{d} = \hat{d}$, where $\hat{d} = 1.5\mu\mathbb{1}$ and $\mu = -\lambda_{\min}(Q + \alpha A^T A)$.
Set $\sigma = \sigma_{\text{init}}$, where σ_{init} is calculated according to (4.54).
Calculate $V = \left[Q + \text{diag}(\bar{d}) + \alpha A^T A\right]^{-1}$ and set $k = 0$.
while $(k < \text{MaxIter})$
    Update $k \leftarrow k + 1$.
    Determine an index i according to (4.43) and calculate Δd_i^* using (4.45).
    Update $\bar{d}$ according to (4.48) and determine $\bar{d}_{\max}$ using (4.50).
    If $(\bar{d}_{\max} > 10\mu)$ **then**
        Update $\rho \leftarrow \rho_{\text{upd}}\rho$ and **goto** 4.
    End If
    Update V according to (4.49).
    Adjust σ according to (4.51).
    If $(k \bmod (\omega_{\text{check}} n) = 0)$ **then**
        Terminate if the improvement in the objective of (4.38) is smaller than ϵ_{check}.
    End If
End while

Even though Algorithm 7 is a variant of the barrier coordinate minimization algorithm introduced in [29], there are two key differences between these two algorithms. First, unlike the algorithm presented in [29], Algorithm 7 does not rely on nonsmooth optimization techniques because the objective function of our regularized separation problem (4.38) is smooth. Second, the regularization parameter λ used in (4.37) is fixed throughout the execution of the algorithm proposed in [29]. By contrast, in Algorithm 7, we adaptively adjust the regularization parameter ρ used in (4.38). As we demonstrate in §4.5.2, because of this adaptive strategy, the quadratic cuts derived by solving (4.38) with our algorithm lead to significantly tighter relaxations than the quadratic cuts obtained through the solution

of (4.37) with the algorithm proposed in [29].

4.4 Implementation in a branch-and-bound algorithm

As discussed in §3.5, BARON's portfolio of relaxations consists of LP, QP, NLP and MILP relaxations. As part of our implementation, we have expanded this portfolio by adding the convex relaxations described in §4.1.4. These new relaxations are only used if the original model supplied to BARON is of the form (4.1).

At the root node of the branch-and-bound tree, we solve convex QCP relaxations of the form (4.31) by running Algorithm 6. In each iteration of this algorithm, we generate quadratic cuts of the form (4.31b) by solving the regularized separation problem (4.38) with Algorithm 7. In our implementation, we set MaxNC $= 21$. As indicated in §4.1.4, for $m > 0$, the initial perturbation used in Algorithm 6 is set as $d^{(1)} = \mu \mathbb{1}$, where $\mu = -\lambda_{\min}(Z^T Q Z)$. Recall that in §3.3.3, we showed that it is possible to approximate $\lambda_{\min}(Z^T Q Z)$ without having to explicit compute the basis Z. In particular, we proved that:

$$\lim_{\alpha \to \infty} \lambda_{\min}(Q, I_n + \alpha A^T A) = \min(0, \lambda_{\min}(Z^T Q Z)) \tag{4.55}$$

Using this result, we set $\mu = \mu(\alpha) := -\lambda_{\min}(Q, I_n + \alpha A^T A)$. From (4.55), it follows that, for a sufficiently large value of α, $\mu(\alpha)$ will converge to 0 if (4.1) is convex when restricted to the nullspace of A, or $-\lambda_{\min}(Z^T Q Z)$ otherwise. To find such value of α, we follow the iterative procedure presented in §3.5. This is the same value of α that we use in Algorithm 6.

At nodes other than the root-node, we solve QP relaxations of the form:

$$\min_{(x,y) \in \mathcal{F}} \quad x^T \left(Q + \operatorname{diag}(d)\right) x + q^T x - d^T y \tag{4.56}$$

where $d \in \mathcal{D}_\alpha$. We proceed as follows:

(i) We solve an initial relaxation of the form (4.56) by setting $d = d^{\text{parent}}$, where d^{parent} is

a diagonal perturbation originating from the parent node. Let $(\bar{x}, \bar{y})$ be the optimal solution of this initial QP relaxation and denote by $\bar{\mu}_{\text{QP}}$ its optimal objective function value.

(ii) We use this relaxation solution to construct a new perturbation d^{new} by solving the regularized separation problem (4.38) with Algorithm 7.

(iii) If $\bar{\mu}_{\text{QP}} < \bar{x}^T \left(Q + \text{diag}(d^{\text{new}})\right) \bar{x} - (d^{\text{new}})^T \bar{y}$, we solve a second quadratic relaxation of the form (4.56) by setting $d = d^{\text{new}}$. Let $(\hat{x}, \hat{y})$ be an optimal solution of this relaxation and denote by $\hat{\mu}_{\text{QP}}$ its optimal objective function value. If $\hat{\mu}_{\text{QP}} \geq \bar{\mu}_{\text{QP}}$ (resp. $\hat{\mu}_{\text{QP}} < \bar{\mu}_{\text{QP}}$), we use the bound $\hat{\mu}_{\text{QP}}$ (resp. $\bar{\mu}_{\text{QP}}$) and pass d^{new} (resp. d^{parent}) to the descendant nodes of the current node.

For the descendant nodes of the root-node, we set $d^{\text{parent}} = d^{\text{root}}$, with d^{root} being a surrogate perturbation vector determined as:

$$d^{\text{root}} = \frac{1}{\sum_{i=1}^{\text{NC}} \nu_i} \sum_{i=1}^{\text{NC}} \nu_i d^{(i)}, \tag{4.57}$$

where NC is the number of quadratic cuts generated during the execution of Algorithm 6 at the root-node, $d^{(i)}$ are the diagonal perturbations, and ν_i are the optimal Lagrange multipliers associated with the quadratic constraints of the last root node relaxation of the form (4.31).

The decision to solve QP relaxations instead of QCP relaxations at nodes other than the root node is motivated by two key observations. First, the convex QCP relaxations of the form (4.31) are at least an order of magnitude more expensive than the QP relaxations of the form (4.56). Second, often a single quadratic cut of the form (4.31b) leads to a significant bound improvement. As a result, there is little gain in running Algorithm 7 more than once. Since the first QP relaxation constructed at the descendant nodes always uses a diagonal perturbation originating from the parent node, the monotonicity of the bounds

generated during the branch-and-bound search is guaranteed.

To solve the eigenvalue and generalized eigenvalue problems that arise during the construction of the relaxations discussed above, we use the subroutines included in the linear algebra library LAPACK [3]. At a given node of the branch-and-bound tree, we only consider the variables that have not been fixed in order to construct our relaxations. We solve the convex QCP relaxations with IPOPT and the convex QP relaxations with CPLEX. The relaxation solutions returned by these solvers are used at a given node only if they satisfy the KKT conditions. At nodes at which (4.1) is convex, we do not use the relaxations described in this section, and solve instead a continuous relaxation of (4.1) subject to the variable bounds of the current node.

When all the variables in (4.1) are binary, we have that $l_i(x_i) = u_i(x_i) = x_i,\ \forall i \in [n]$, and we can eliminate the y variables from (4.31) and (4.56). For continuous and general integer variables, we use $l_i(x_i) = x_i^2,\ i \in [n]$. For general integer variables, this choice of $l_i(x_i)$ does not lead to the convex hull of $\mathcal{C}_i$, but it allows us to construct a convex outer-approximation for this set.

Our implementation relies on the dynamic relaxation selection strategy proposed in 3.5 to adjust the frequencies at which we solve polyhedral and quadratic relaxations during the branch-and-bound search. Moreover, if (4.1) is a binary quadratic program, we use the spectral braching variable selection rule introduced in 3.4. The QP relaxations (4.56) are only used during the branch-and-bound search if, at the root-node, Algorithm 6 gives a tighter bound than BARON's LP relaxation. Otherwise, we disable these QP relaxations and utilize the spectral relaxations proposed in §3.3.1–3.3.3.

4.5 Computational results

In this section, we investigate the impact of the relaxations proposed in §4.1.4 on the performance of branch-and-bound algorithms. We start by introducing the test set used for the numerical experiments in §4.5.1. Then, in §4.5.2, we show the effectiveness of the regularization approach discussed in §4.2. In §4.5.3, we demonstrate the benefits of the implementation described in §4.4 on the performance of BARON. This is followed by a comparison between several state-of-the-art global optimization solvers in §4.5.4.

Our experiments are conducted under GAMS 30.1.0 on a 64-bit Intel Xeon X5650 2.66GHz processor with a single-thread. For the experiments described in §4.5.2, we solve the QP relaxations with CPLEX 12.10, the QCP relaxations with IPOPT 3.12 and the SDP relaxations with MOSEK 9.1.9. For the experiments considered in §4.5.3–4.5.4, we consider the following global optimization solvers: ANTIGONE 1.1, BARON 19.12, COUENNE 0.5, CPLEX 12.10, GUROBI 9.0, LINDOGLOBAL 12.0 and SCIP 6.0. In this case we: (i) run all solvers with relative/absolute tolerances of 10^{-6} and a time limit of 500 seconds, and (ii) set the CPLEX option `optimalitytarget` to 3 and the GUROBI option `nonconvex` to 2 to ensure that these two solvers search for a globally optimal solution. We use default settings for other algorithmic parameters.

4.5.1 The test set

For our experiments, we consider a large collection of nonconvex problems of the form (4.1) consisting of the 1551 CBQP, QSAP, BoxQP and EIQP instances described in §3.6.1.

4.5.2 Experiments with root-node relaxations

In this section, we provide a numerical comparison between two versions of Algorithm 6 which differ in the separation procedure used to derive the convex quadratic cuts of the

form (4.31b). We use the following notation for the relaxations considered in this comparison:

(i) EIG: Eigenvalue relaxation (3.26).

(ii) EIGNS: Eigenvalue relaxation in the nullspace of A (3.41).

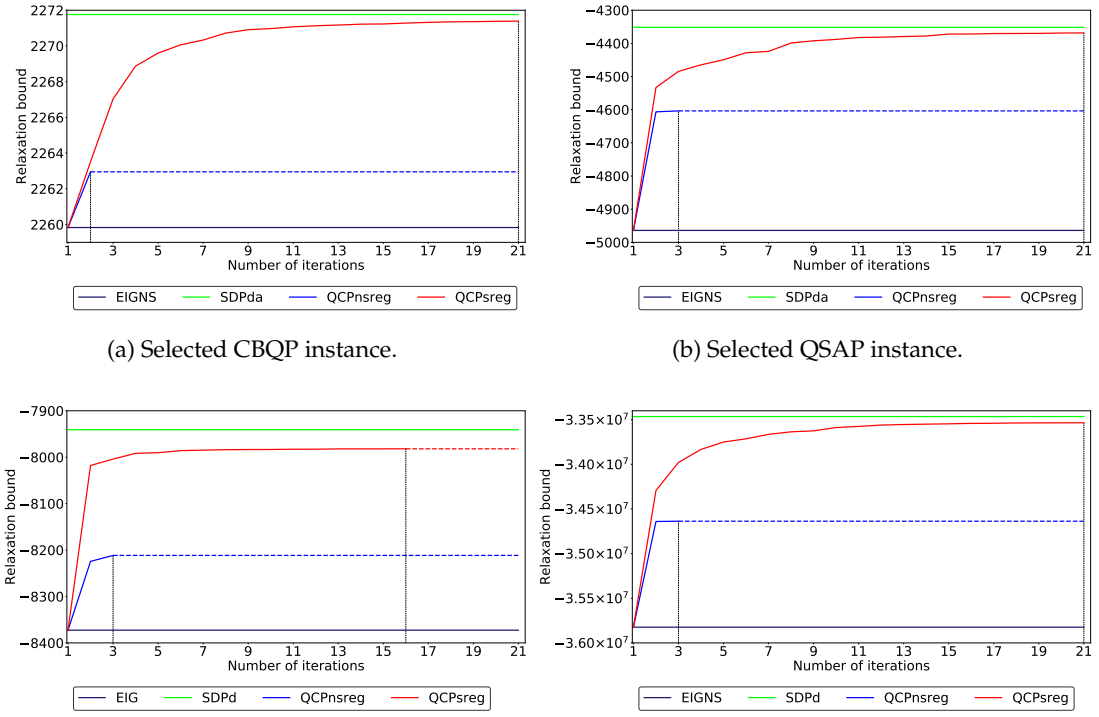
(iii) SDPd: SDP relaxation (4.14).

(iv) SDPda: SDP relaxation (4.15).

(v) QCPnsreg: QCP relaxation (4.31), where the quadratic cuts (4.31b) are obtained by solving (4.37) with algorithm proposed in [29].

(vi) QCPsreg: QCP relaxation (4.31), where the quadratic cuts (4.31b) are obtained by solving the our separation problem (4.38) with Algorithm 7.

In our experiments, we run the two versions of Algorithm 6 by setting the maximum number of iterations MaxNC to 21. We first compare these relaxations by selecting one instance from each of the four test libraries mentioned in §4.5.1. The results of this comparison are presented in Figures 4.1a–4.1d. In these figures, we plot the lower bounds of the QCP relaxations against the number of iterations, and use a dashed vertical line to indicate the iteration number at which each version of Algorithm 6 terminates. We use horizontal lines to represent the lower bounds provided by the spectral and SDP relaxations. As seen in the figures, the quadratic cuts derived by solving (4.38) with Algorithm 7 lead to significantly tighter QCP relaxations than the quadratic cuts obtained through the solution of (4.37) with the algorithm proposed in [29]. Note that under our approach, a few quadratic cuts are sufficient in order to obtain a good approximation of the lower bounds given by the SDP relaxations.



(a) Selected CBQP instance.

(b) Selected QSAP instance.

(c) Selected BoxQP instance.

(d) Selected EIQP instance.

Figure 4.1: Comparison between the two versions of the cutting surface algorithm for selected test problems.

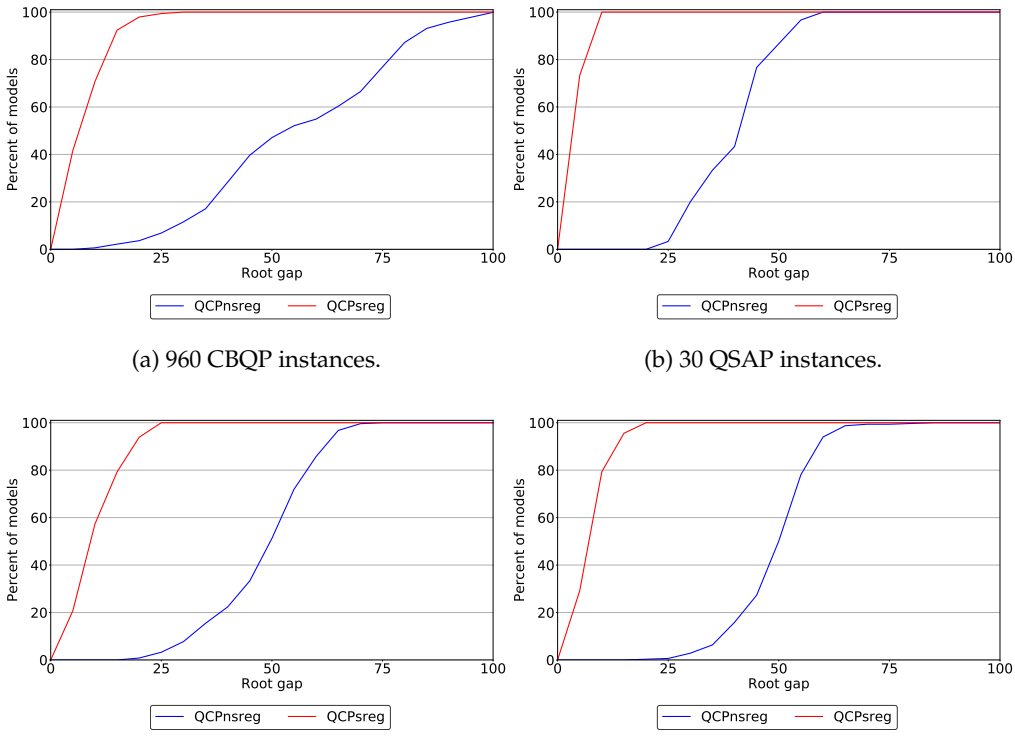
Now, we compare the two versions of Algorithm 6 by considering all the instances contained in each of the test libraries. To this end, we construct performance profiles based on the following root-node relaxation gap:

$$\text{Root gap} = \left(\frac{\mu_{\text{SDP}} - \mu_{\text{QCP}}}{\mu_{\text{SDP}} - \mu_{\text{QP}}} \right) \times 100 \tag{4.58}$$

where μ_{QCP} is the lower bound given by the last QCP relaxation solved in a given version of Algorithm 6, and μ_{QP} and μ_{SDP} denote the lower bounds provided by the corresponding spectral and SDP relaxations. A smaller gap represents a better approximation of the corresponding SDP bound.

The performance profiles are presented in Figures 4.2a–4.2d. These profiles show the

percentage of models for which the gap defined in (4.58) is below a certain threshold. Clearly, the QCP relaxations constructed via our separation procedure provide significantly smaller gaps than the QCP relaxations derived with the separation algorithm proposed in [29].



(a) 960 CBQP instances.

(b) 30 QSAP instances.

(c) 246 BoxQP instances.

(d) 315 EIQP instances.

Figure 4.2: Comparison between the two versions of the cutting surface algorithm for all test problems.

4.5.3 Impact of the implementation on BARON's performance

In this section, we demonstrate the benefits the relaxations introduced in this chapter on the performance of the global optimization solver BARON. In our experiments, we compare the following versions of BARON 19.12:

(i) BARONqp1: Version of BARON for which we disable the quadratic relaxations proposed in this chapter. This is the version of BARON which makes use of the spectral relaxations introduced in Chapter 3.

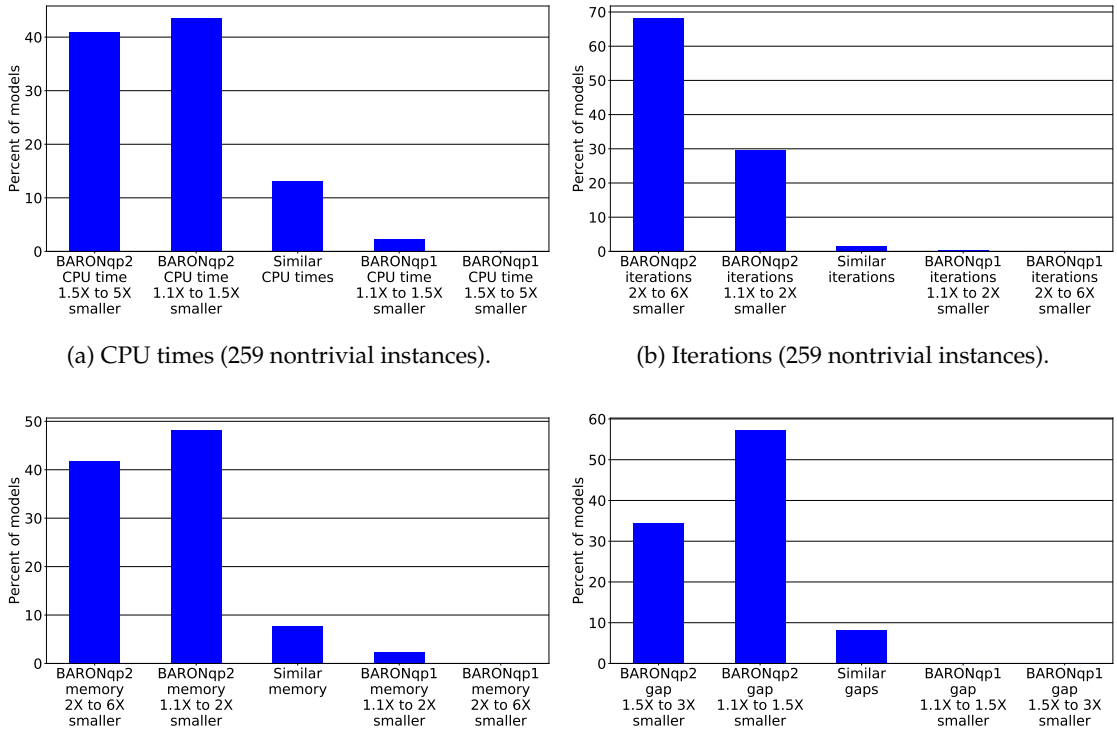
(ii) BARONqp2: Version of BARON which uses the quadratic relaxations proposed in this chapter as described in §4.4.

In this comparison, we exclude from the test set all problems for which the new quadratic relaxations are not activated by BARONqp2 during the branch-and-bound search (367 instances). We also eliminate problems that can be solved trivially by both solvers (62 instances). A problem is regarded as trivial if it can be solved by both solvers in less than one second. After eliminating all of these problems from the original test set, we obtain a new test set consisting of 1122 instances.

We first consider the nontrivial problems that are solved to global optimality by at least one of the two the versions of the solver (259 instances). For this analysis, we compare the performance of the two solvers by considering the following metrics: (i) CPU time, (ii) total number of nodes in the branch-and-bound tree (iterations), and (iii) maximum number of nodes stored in memory (memory). In this comparison, we say that the two solvers perform similarly if any of these metrics are within 10% of each other. The results are presented in Figures 4.3a–4.3c. As the figures indicate, for nearly 90% of the problems considered in this comparison, our implementation leads to a significant reduction in CPU time, number of iterations, and memory requirements.

Now, we consider the nontrivial problems that neither of the two solvers are able to solve to global optimality within the time limit (863 instances). In this case, we analyze the performance of these solvers by comparing the relative gaps reported at termination. These gaps are determined according to (3.72) by using the best lower and upper bounds reported by the solver under consideration. In this comparison, we say that two solvers

obtain similar gaps if their relative gaps are within 10% of each other. The results are presented in Figure 4.3d. As seen in the figure, for more than 90% of the problems considered in this comparison, BARONqp2 reports significantly smaller gaps than BARONqp1.



(a) CPU times (259 nontrivial instances).

(b) Iterations (259 nontrivial instances).

(c) Memory (259 nontrivial instances).

(d) Relative gaps (863 nontrivial instances).

Figure 4.3: One-to-one comparison between BARONqp1 and BARONqp2.

We finish this section by providing a more detailed analysis of the results presented in Figures 4.3a–4.3d. To this end, we calculate the shifted geometric means for each of the metrics considered in these figures. We use a shift factor of 1 for the CPU times and relative gaps, and a shift factor of 10 for the total number of nodes and maximum number of nodes stored in memory. The results are presented in Table 4.1. As seen in the table, BARONqp2 significantly outperforms BARONqp1 for each of the considered metrics.

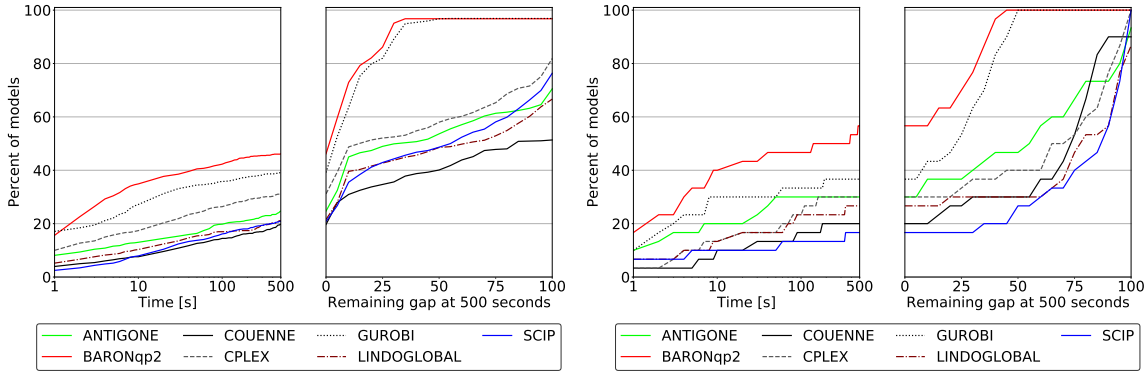
Table 4.1: Shifted geometric means for BARONqp1 and BARONqp2.

Solver	CPU Time (259 instances)	Iterations (259 instances)	Memory (259 instances)	Relative gaps (863 instances)
BARONqp1	14.0	926.1	25.9	11.8
BARONqp2	9.9	391.7	13.5	8.7
Improvement (%)	29.6	57.7	47.7	25.7

4.5.4 Comparison between global optimization solvers

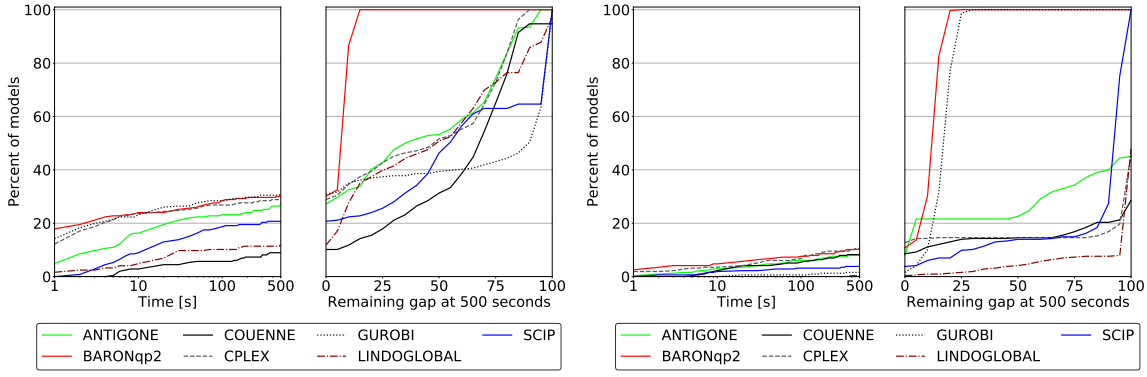
We start this section by comparing several state-of-the-art global optimization solvers using the same type of performance profiles employed in §3.6.4. These profiles are shown in Figures 4.4a–4.4d. As seen in these figures, BARONqp2 performs well relative to the other solvers. For the CBQP and QSAP instances, BARONqp2 is faster than the other solvers and solves many more problems to global optimality. For problems for which global optimality cannot be proven within the time limit, BARONqp2 terminates with smaller gaps than the other solvers.

Next, we provide a more detailed analysis involving BARONqp2, CPLEX and GUROBI. We use the same type of bar plots employed in §4.5.3. We start by presenting a one-to-one comparison between BARONqp2 and CPLEX. To this end, we eliminate from the test set all the problems that can be solved trivially by both solvers (124 instances), obtaining a new test set with 1427 instances. In Figure 4.5a, we consider the nontrivial problems that are solved to global optimality by at least one of the two solvers (453 instances), whereas in Figure 4.5b, we consider nontrivial problems that neither of the two solvers can solve to global optimality within the time limit (974 instances). As both figures indicate, BARONqp2 performs significantly better than CPLEX. For nearly 90% of the instances considered in Figure 4.5a, BARONqp2 is at least 1.1 times faster than CPLEX, whereas for more than 98% of the instances considered in Figure 4.5b, BARONqp2 reports significantly



(a) 960 CBQP instances.

(b) 30 QSAP instances.



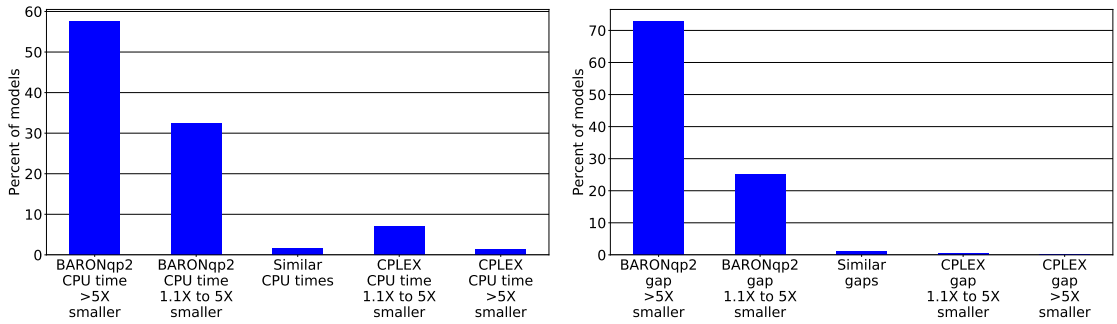
(c) 246 BoxQP instances.

(d) 315 EIQP instances.

Figure 4.4: Comparison between global optimization solvers.

smaller gaps than CPLEX.

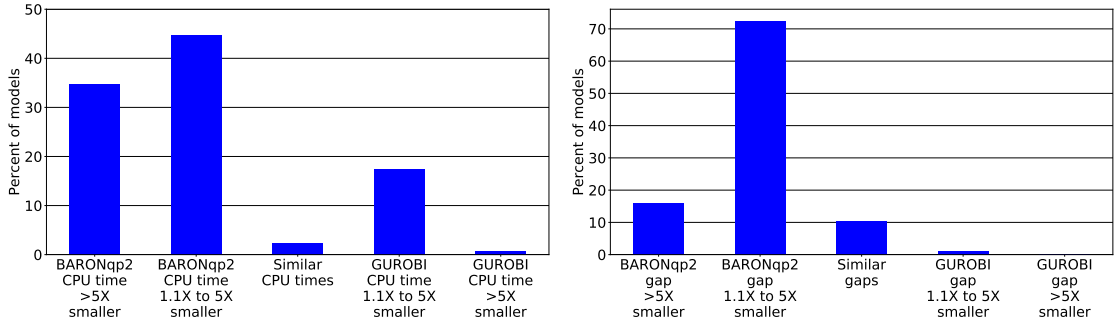
Finally, we present a one-to-one comparison between BARONqp2 and GUROBI. Once again, we eliminate from the test set all the problems that can be solved trivially by both solvers (183 instances), which leads to a new test set with 1368 instances. In Figure 4.6a, we consider the nontrivial problems that are solved to global optimality by at least one of the two solvers (391 instances), whereas in Figure 4.6b, we consider nontrivial problems that neither of the two solvers are able to solve to global optimality within the time limit (977 instances). For more than 80% of the instances considered in Figure 4.6a, BARONqp2



(a) CPU times (453 nontrivial instances). (b) Relative gaps (974 nontrivial instances).

Figure 4.5: One-to-one comparison between BARONqp2 and CPLEX.

is at least 1.1 faster than GUROBI, whereas for nearly 90% of the instances considered in Figure 4.6b, BARONqp2 terminates with considerably smaller gaps than GUROBI.



(a) CPU times (391 nontrivial instances). (b) Relative gaps (977 nontrivial instances).

Figure 4.6: One-to-one comparison between BARONqp2 and GUROBI.

4.6 Conclusions

In this chapter, we introduced a family of convex quadratic relaxations for nonconvex MIQPs which are constructed via convex quadratic cuts. In order to derive these quadratic cuts, we proposed a smooth regularized separation problem which is solved by using a

variant of a coordinate minimization algorithm recently introduced in [29]. Moreover, we studied the theoretical properties of the resulting relaxations and provided conditions under which they are equivalent to certain SDPs. To assess the benefits of our approach, we incorporated the proposed relaxation techniques into the global optimization solver BARON, and tested our implementation on a large collection of problems. Results demonstrated that, for our test problems, our implementation leads to a very significant improvement in the performance of BARON.

Chapter 5

Conclusions and future work

In this chapter, we summarize the main contributions of this thesis and provide directions for future work.

5.1 Key contributions

In the following, we highlight the major contributions of this thesis:

Global optimization of problems with convex-transformable intermediates

- We proposed algorithms for the recognition of convex-transformable functions in general nonconvex problems.
- We introduced a new class of cutting planes based on recently developed relaxations for convex-transformable functions.
- We integrated the proposed recognition and cutting plane generation algorithms into the global solver BARON. We demonstrated the computational benefits of this implementation by conducting numerical experiments on a large collection of nonconvex problems involving convex-transformable functions.

Spectral relaxations and branching strategies for global optimization of MIQPs

- We introduced a family of convex quadratic relaxations for nonconvex QPs and MIQPs. We demonstrated that these relaxations can be constructed through pertur-

bations of the quadratic matrix and used information from the equality constraints in order to improve the resulting bounds.

- We investigated the theoretical properties of the proposed relaxations and proved that they are equivalent to certain SDPs.
- We devised novel eigenvalue-based variable selection branching strategies which can be used with nonconvex binary quadratic programs. These strategies are inspired by strong branching and involve an effective approximation of the impact of branching decisions on the quality of the corresponding relaxations.
- We integrated the proposed relaxation and branching strategies into the global solver BARON. We demonstrated the computational benefits of this implementation by conducting an extensive computational study on a variety of nonconvex QPs and MIQPs.

SDP-approximating convex quadratic relaxations for global optimization of MIQPs

- We proposed a new class of convex QCP relaxations for nonconvex QPs and MIQPs. These relaxations are constructed via convex quadratic cuts which can be obtained from a semidefinite separation problem with a special structure that allows the use of specialized solution algorithms.
- We showed that the proposed relaxations are an outer-approximation of a semi-infinite convex program which generalizes a semi-infinite formulation that had been previously considered in the literature. Moreover, we proved that under certain conditions our semi-infinite formulation is equivalent to a well-known semidefinite program relaxation.
- We devised a novel regularization approach for the above-mentioned separation problem and showed its benefits through numerical experiments.
- We implemented the proposed relaxations in the global solver BARON. We investi-

gated the impact of this implementation by performing an extensive computational study on a variety of nonconvex QPs and MIQPs.

5.2 Future work

In the following, we outline directions for future work:

Global optimization of problems with convex-transformable intermediates

- Future research might investigate the impact of the implementation described in Chapter 2 in the context of solving applications concerning convex-transformable functions. An important application in economics involves the construction of mathematical models which describe the relationship between product attributes and consumer choices. These models rely on parameters that are typically determined from data thorugh maximum likelihood estimation methods [94]. In some cases, the resulting parameter estimation problems are nonconvex and involve concave-transformable expressions which fall into the class of functions discussed in §2.2.3. An interesting question is whether the cutting planes proposed in Chapter 2 can lead to tighter relaxations for these classes of nonconvex problems and enable global optimization solvers to solve practically relevant instances to global optimality.

Spectral relaxations and branching strategies for global optimization of MIQPs

- Future work might extend the convexification techniques proposed in Chapter 3 to more general classes of problems such as nonconvex quadratically-constrained quadratic programs (QCQPs). Given a nonconvex QCQP, each nonconvex quadratic constraint can be relaxed in the same way in which the objective function was relaxed in Chapter 3, leading to a convex QCQP relaxation. A key question in this context is

how tight the resulting QCPQP relaxations would be in comparison with the polyhedral and SDP relaxations which are typically used for bounding nonconvex QCQPs.

- Another important issue is related to the scalability of the methods used for solving the eigenvalue and generalized eigenvalue problems that arise during the construction of the relaxations proposed in Chapter 3. The implementation described in §3.5 makes use of LAPACK subroutines in order to solve these problems. In the numerical experiments described in §3.6, we considered problems containing up to 400 variables. For these instances, the LAPACK subroutines are very efficient. However, these subroutines rely on direct eigenvalue methods which may not scale well as the corresponding matrices increase in size. As a result, an implementation capable of handling larger problems might have to make use of iterative procedures such as Krylov Subspace eigenvalue methods.

SDP-approximating convex quadratic relaxations for global optimization of MIQPs

- The convex QCP relaxations proposed in Chapter 4 can be cast as linearly-constrained problems where the objective is given by the maximum of a set of convex quadratic functions. An interesting question in this context is whether it is possible to solve this formulation through specialized first-order algorithms such as the ones used in [43] for solving the Constrained Lasso problem.
- Future research might also investigate whether it is possible to construct convex quadratic relaxations which approximate the bounds given by SDP relaxations different from those considered in Chapter 4. An approximation of this type might also rely on convex quadratic cuts which are derived from a semidefinite separation problem. In order for these relaxations to be useful in a branch-and-bound setting, it is crucial to design algorithms capable of solving this separation problem efficiently.

Bibliography

[1] T. Achterberg, T. Koch, and A. Martin. Branching rules revisited. *Operations Research Letters*, 33:42–54, 2005.

[2] F. A. Al-Khayyal and J. E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8:273–286, 1983.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' guide*, volume 9. Siam, 1999.

[4] I. P. Androulakis, C. D. Maranas, and C. A. Floudas. αBB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7:337–363, 1995.

[5] K. M. Anstreicher. Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. *Journal of Global Optimization*, 43:471–484, 2009.

[6] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. Finding cuts in the TSP, Technical Report 95-05, DIMACS, 1995.

[7] M. Avriel, W. E. Diewert, S. Schaible, and I. Zang. *Generalized Concavity*. Plenum Press, 1988.

[8] E. Ayotte-Sauv. NRC Library, 2016. Personal Communication.

[9] X. Bao, A. Khajavirad, N. V. Sahinidis, and M. Tawarmalani. Global optimization of nonconvex problems with multilinear intermediates. *Mathematical Programming Computation*, 7:1–37, 2015.

[10] X. Bao, N. V. Sahinidis, and M. Tawarmalani. Multiterm polyhedral relaxations for nonconvex, quadratically-constrained quadratic programs. *Optimization Methods and Software*, 24:485–504, 2009.

[11] X. Bao, N. V. Sahinidis, and M. Tawarmalani. Semidefinite relaxations for quadratically constrained quadratic programming: A review and comparisons. *Mathematical Programming*, 129:129–157, 2011.

[12] P. Belotti. *COUENNE:* A user's Manual, 2009. Technical report, Lehigh University.

[13] T. Berthold, G. Gamrath, G. Hendel, S. Heinz, T. Koch, M. Pfetsch, S. Vigerske, R. Waniek, M. Winkler, and K. Wolter. *SCIP 3.2,* User's Manual, 2016.

[14] A. Billionnet and S. Elloumi. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. *Mathematical Programming*, 109:55–68, 2007.

[15] A. Billionnet, S. Elloumi, and A. Lambert. Extending the QCR method to general mixed-integer programs. *Mathematical programming*, 131:381–401, 2012.

[16] A. Billionnet, S. Elloumi, and A. Lambert. An efficient compact quadratic convex reformulation for general integer quadratic programs. *Computational Optimization and Applications*, 54:141–162, 2013.

[17] A. Billionnet, S. Elloumi, and M. C. Plateau. Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method. *Discrete Applied Mathematics*, 157:1185–1197, 2009.

[18] I. M. Bomze and M. Locatelli. Undominated d.c. decompositions of quadratic functions and applications to branch-and-bound approaches. *Computational Optimization and Applications*, 28:227–245, 2004.

[19] P. Bonami, O. Günlük, and J. Linderoth. Globally solving nonconvex quadratic programming problems with box constraints via integer programming methods. *Mathematical Programming Computation*, pages 1–50, 2018.

[20] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[21] C. Buchheim and A. Wiegele. Semidefinite relaxations for non-convex quadratic mixed-integer programming. *Mathematical Programming*, 141:435–452, 2013.

[22] S. Burer. Optimizing a polyhedral-semidefinite relaxation of completely positive programs. *Mathematical Programming Computation*, 2:1–19, 2010.

[23] S. Burer and D. Vandenbussche. A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. *Mathematical Programming*, 113:259–282, 2008.

[24] S. Burer and D. Vandenbussche. Globally solving box-constrained nonconvex quadratic programs with semidefinite-based finite branch-and-bound. *Computational Optimization and Applications*, 43:181–195, 2009.

[25] R. E. Burkard, E. Cela, P. M. Pardalos, and L. S. Pitsoulis. The quadratic assignment problem. In R. Horst and P. M. Pardalos (eds.), *Handbook of Combinatorial Optimization*, Kluwer Academic Publishers, Boston, MA, pages 1713–1809, 1998.

[26] M. R. Bussieck, A. S. Drud, and A. Meeraus. MINLPLib–A collection of test models

for mixed-integer nonlinear programming. *INFORMS Journal on Computing*, 15:114–119, 2003.

[27] R. J. Dakin. A tree search algorithm for mixed integer programming problems. *Computer Journal*, 8:250–255, 1965.

[28] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[29] H. Dong. Relaxing nonconvex quadratic functions by multiple adaptive diagonal perturbations. *SIAM Journal on Optimization*, 26:1962–1985, 2016.

[30] J. E. Falk and R. M. Soland. An algorithm for separable nonconvex programming problem. *Management science*, 15:550–569, 1969.

[31] A. Faye and F. Roupin. Partial lagrangian relaxation for general quadratic programming. *4OR*, 5:75–88, 2007.

[32] GLOBAL Library. `http://www.gamsworld.org/global/globallib.htm`.

[33] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of ACM*, 42:1115–1145, 1995.

[34] G. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, third edition, 1996.

[35] I. E. Grossmann. Mixed-integer programming approach for the synthesis of integrated process flowsheets. *Computers & Chemical Engineering*, 9:463–482, 1985.

[36] Gurobi Optimization. *GUROBI Optimizer Reference Manual Version 9.0*, 2020. Available at `https://www.gurobi.com/wp-content/plugins/hd_documentations/documentation/9.0/refman.pdf`.

[37] P. L. Hammer and A. A. Rubin. Some remarks on quadratic programming with 0-1 variables. *RAIRO-Operations Research*, 4:67–79, 1970.

[38] R. Horst and H. Tuy. *Global optimization: Deterministic approaches*. Springer Verlag, Berlin, 1996.

[39] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer Verlag, Berlin, Third edition, 1996.

[40] M. Hunting. AIMMS Library, 2016. Personal Communication.

[41] IBM ILOG. *CPLEX 12.7 User's Manual*, 2016. Available at `http://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.0/ilog.odms.studio.help/pdf/usrcplex.pdf`.

[42] M. Jach, D. Michaels, and R. Weismantel. The convex envelope of $(n-1)$-convex functions. *SIAM Journal on Optimization*, 19:1451–1466, 2008.

[43] G. M. James, C. Paulson, and P. Rusmevichientong. The Constrained Lasso. Technical report, University of Southern California, 2012.

[44] A. Khajavirad and J. J. Michalek. A deterministic lagrangian-based global optimization approach for quasiseparable nonconvex mixed-integer nonlinear programs. *Journal of Mechanical Design*, 131:051009, 2009.

[45] A. Khajavirad, J. J. Michalek, and N. V. Sahinidis. Relaxations of factorable functions with convex-transformable intermediates. *Mathematical Programming*, 144:107–140, 2014.

[46] A. Khajavirad and N. V. Sahinidis. Convex envelopes of products of convex and component-wise concave functions. *Journal of Global Optimization*, 52:391–409, 2011.

[47] A. Khajavirad and N. V. Sahinidis. Convex envelopes generated from finitely many compact convex sets. *Mathematical Programming*, 137:371–408, 2013.

[48] A. Khajavirad and N. V. Sahinidis. A hybrid LP/NLP paradigm for global optimization relaxations. *Mathematical Programming Computation*, 10:383–421, 2018.

[49] T. C. Koopmans and M. Beckmann. Assignment problems and the location of economic activities. *Econometrica: Journal of the Econometric Society*, 25:53–76, 1957.

[50] A. H. Land and A. G. Doig. An automatic method for solving discrete programming problems. *Econometrica*, 28:497–520, 1960.

[51] E. L. Lawler. The quadratic assignment problem. *Management science*, 9:586–599, 1963.

[52] H. Li, J. Tsai, and C. A. Floudas. Convex underestimation for posynomial functions of positive variables. *Optimization Letters*, 2:333–340, 2008.

[53] R. M. Lima and I. E. Grossmann. On the solution of nonconvex cardinality boolean quadratic programming problems: a computational study. *Computational Optimization and Applications*, 66:1–37, 2017.

[54] Y. Lin and L. Schrage. The global solver in the LINDO API. *Optimization Methods and Software*, 24:657–668, 2009.

[55] E. M. Loiola, N. M. M. de Abreu, P.O. Boaventura-Netto, P. Hahn, and T. Querido. A survey for the quadratic assignment problem. *European Journal of Operational Research*, 176:657–690, 2007.

[56] H. Lu, H. Li, C. E. Gounaris, and C. A. Floudas. Convex relaxation for solving posynomial programs. *Journal of Global Optimization*, 46:147–154, 2010.

[57] A. Lundell, J. Westerlund, and T. Westerlund. Some transformation techniques with applications in global optimization. *Journal of Global Optimization*, 43:391–405, 2009.

[58] A. Lundell and T. Westerlund. Convex underestimation strategies for signomial functions. *Optimization Methods and Software*, 24:505–522, 2009.

[59] M. Kılınç and N. V. Sahinidis. Exploiting integrality in the global optimization of mixed-integer nonlinear programming problems in BARON. *Optimization Methods and Software*, 33:540–562, 2019.

[60] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[61] C. A. Meyer and C. A. Floudas. Trilinear monomials with mixed sign domains: Facets of the convex and concave envelopes. *Journal of Global Optimization*, 29:125–155, 2004.

[62] C. A. Meyer and C. A. Floudas. Convex envelopes for edge-concave functions. *Mathematical Programming*, 103:207–224, 2005.

[63] R. Misener and C. A. Floudas. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming*, 136:155–182, 2012.

[64] R. Misener and Ch. A. Floudas. ANTIGONE: Algorithms for coNTinuous/Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 59:503–526, 2014.

[65] R. Misener, C. E. Gounaris, and C. A. Floudas. Mathematical modeling and global optimization of large-scale extended pooling problems with the (EPA) complex emissions constraints. *Computers & Chemical Engineering*, 34:1432 – 1456, 2010.

[66] R. Misener, J. B. Smadbeck, and C. A. Floudas. Dynamically generated cutting planes

for mixed-integer quadratically constrained quadratic programs and their incorporation into glomiqo 2. *Optimization Methods and Software*, 30:215–249, 2015.

[67] A. Neumaier. Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Review*, 39:407460, 1997.

[68] P. M. Pardalos, J. H. Glick, and J. B. Rosen. Global minimization of indefinite quadratic problems. *Computing*, 539:281–291, 1987.

[69] A.T. Phillips and J.B. Rosen. A quadratic assignment formulation of the molecular conformation problem. *Journal of Global Optimization*, 4:229–241, 1994.

[70] P. K. Polisetty, E. P. Gatzke, and E. O. Voit. Yield optimization of regulated metabolic systems using deterministic branch-and-reduce methods. *Biotechnology and Bioengineering*, 99:1154–1169, 2008.

[71] S. Poljak, F. Rendl, and H. Wolkowicz. A recipe for semidefinite relaxation for (0, 1)-quadratic programming. *Journal of Global Optimization*, 7:51–73, 1995.

[72] Princeton Library. `http://www.gamsworld.org/performance/princetonlib/princetonlib.htm`.

[73] Y. Puranik and N. V. Sahinidis. Bounds tightening based on optimality conditions for nonconvex box-constrained optimization. *Journal of Global Optimization*, 67:59–77, 2017.

[74] A. D. Rikun. A convex envelope formula for multilinear functions. *Journal of Global Optimization*, 10:425–437, 1997.

[75] H. S. Ryoo and N. V. Sahinidis. A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8:107–139, 1996.

[76] N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8:201–205, 1996.

[77] N. V. Sahinidis. *BARON User Manual v. 19.2.7*, 2019. Available at `http://www.minlp.com/downloads/docs/baron%20manual.pdf`.

[78] N. V. Sahinidis, M. Tawarmalani, and M. Yu. Design of alternative refrigerants via global optimization. *AIChE Journal*, 49:1761–1775, 2003.

[79] A. Saxena, P. Bonami, and J. Lee. Convex relaxations of non-convex mixed integer quadratically constrained programs: projected formulations. *Mathematical programming*, 130:359–413, 2011.

[80] H. D. Sherali. Convex envelopes of multilinear functions over a unit hypercube and over special discrete sets. *Acta Mathematica Vietnamica*, 22:245–270, 1997.

[81] H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal of Discrete Mathematics*, 3:411–430, 1990.

[82] H. D. Sherali and W. P. Adams. A hierarchy of relaxations and convex hull characterizations for mixed- integer zero-one programming problems. *Discrete Applied Mathematics*, 52(1):83–106, 1994.

[83] H. D. Sherali and W. P. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*, volume 31 of *Nonconvex Optimization and its Applications*. Kluwer Academic Publishers, Dordrecht, 1999.

[84] H. D. Sherali and A. Alameddine. A new reformulation-linearization technique for bilinear programming problems. *Journal of Global Optimization*, 2:379–410, 1992.

[85] H. D. Sherali and C. H. Tuncbilek. A reformulation-convexification approach for solving nonconvex quadratic programming problems. *Journal of Global Optimization*, 7:1–31, 1995.

[86] H. D. Sherali and H. Wang. Global optimization of nonconvex factorable programming problems. *Mathematical Programming*, 89:459–478, 2001.

[87] N.Z. Shor. Quadratic optimization problems. *Soviet Journal of Computer and Systems Sciences*, 25:1–11, 1987.

[88] M. Tawarmalani, J-P. Richard, and C. Xiong. Explicit convex and concave envelopes through polyhedral subdivisions. *Mathematical Programming*, 138:531–577, 2013.

[89] M. Tawarmalani and N. V. Sahinidis. Semidefinite relaxations of fractional programs via novel techniques for constructing convex envelopes of nonlinear functions. *Journal of Global Optimization*, 20:137–158, 2001.

[90] M. Tawarmalani and N. V. Sahinidis. Convex extensions and convex envelopes of l.s.c. functions. *Mathematical Programming*, 93:247–263, 2002.

[91] M. Tawarmalani and N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer Academic Publishers, Dordrecht, 2002.

[92] M. Tawarmalani and N. V. Sahinidis. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99:563–591, 2004.

[93] M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103:225–249, 2005.

[94] K. Train. *Discrete Choice Methods With Simulation*. Cambridge University Press, second edition, 2009.

[95] H. Tuy. DC optimization: Theory, methods and algorithms. In *Handbook of global optimization*, pages 149–216. Springer Verlag, 1995.

[96] D. Vandenbussche and G. L. Nemhauser. A polyhedral study of nonconvex quadratic programs with box constraints. *Mathematical Programming*, 102:531–557, 2005.

[97] Y. Zhang, N. V. Sahinidis, C. Nohra, and G. Rong. Optimality-based domain reduction for inequality-constrained nlp and minlp problems. *Journal of Global Optimization*, DOI 10.1007/s10898-020-00886-z, 2020.

[98] K. Zorn and N. V. Sahinidis. Global optimization of general nonconvex problems with intermediate bilinear substructures. *Optimization Methods and Software*, 29:442–462, 2013.

[99] K. Zorn and N. V. Sahinidis. Global optimization of general nonconvex problems with intermediate polynomial substructures. *Journal of Global Optimization*, 59:673–693, 2014.