A coordinate ascent method for solving semidefinite relaxations of non-convex quadratic integer programs

Dissertation
zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften

Der Fakultät für Mathematik
der Technischen Universität Dortmund
vorgelegt von

Jessica Maribel Montenegro Chingal
aus Tulcán

im April 2017

Dissertation

A coordinate ascent method for solving semidefinite relaxations of non-convex quadratic integer programs

Fakultät für Mathematik
Technische Universität Dortmund

Erstgutachter: Prof. Dr. Christoph Buchheim

Zweitgutachterin: Assoc. Prof. Dr. Angelika Wiegele

Tag der mündlichen Prüfung: 17. Mai 2017

Abstract

In this thesis we propose a coordinate ascent method for a class of semidefinite programming problems arising in the reformulation of non-convex quadratic optimization problems where the variables are restricted to subsets of the integer numbers.

It is known that non-convex quadratic integer problems are NP-hard for two reasons: the non-convexity of the objective function and the restrictions of integrality on the variables. Therefore no polynomial time algorithm is known for solving this class of optimization problems. Standard techniques for addressing these problems are reformulations through linearization or semidefinite programming (SDP), aiming at producing tight convex relaxations of the problem that are then embedded into branch-and-bound schemes. Semidefinite programming has been proved to be a powerful tool for constructing strong convex relaxations for several combinatorial optimization problems, however at increased computational cost. Interior-point based algorithms are the classical solution approaches for semidefinite programming problems, although it turns out that large scale instances are beyond the scope of these algorithms.

Buchheim and Wiegele have devised an SDP-based branch-and-bound algorithm (Q-MIST) for a class of mixed-integer quadratic programming problems, that contains the quadratic problems we are considering here. The semidefinite relaxations are solved using the SDP solver CSDP, which based on interior point methods. It has been proved experimentally that this approach outperforms the state-of-the-art non convex mixed-integer programming software COUENNE. Recently, Dong has studied the same class of quadratic problems, and has proposed a semi-infinite convex relaxation. The resulting separation problem is solved by a primal-barrier coordinate minimization algorithm.

In this thesis, we have developed an algorithm that on the one hand exploits the structure of the semidefinite relaxations proposed by Buchheim and Wiegele, namely a small total number of active constraints and constraint matrices characterized by a low rank. On the other hand, our algorithm exploits this special structure by solving the dual problem of the semidefinite relaxation, using a barrier method in combination with a coordinate-wise exact line search, motivated by the algorithm presented by Dong. The main ingredient of our algorithm is the computationally cheap update at each iteration and an easy computation of the exact step size. Compared to interior point methods, our approach is much faster in obtaining strong dual bounds. Moreover, no explicit separation and re-optimization is necessary even if the set of primal constraints is large, since in our dual approach this is covered by implicitly considering all primal constraints when selecting the next coordinate. Even more, the structure of the problem allows us to perform a plane search instead of a single line search, this speeds up the convergence of the algorithm. Finally, linear constraints are easily integrated into the algorithmic framework.

We have performed experimental comparisons on randomly generated instances, showing that our approach significantly improves the performance of Q-MIST when compared with CSDP and outperforms other specialized global optimization software, such as BARON.

Partial Publications and Collaboration Partners

Part of the results of Chapter 5 have been published in [15], and were obtained together with Prof. Buchheim from the University of Dortmund and Prof. Wiegele from University of Klagenfurt.

Acknowledgements

This work would have never been possible without the help and support of many people to whom I am greatly thankful.
I am grateful to my supervisor Prof. Buchheim, who has leading my research through all these years. From him I have not only learned scientific knowledge but also important lessons of life. I am also grateful to Prof. Wiegele for her collaboration with this research and for taking care of me during the time I spent in Klagenfurt.
I appreciate having had the opportunity to work in the LS-V of the University of Dortmund. Thanks to all my colleagues I have had the pleasure of meeting, we have shared good moments not only at working hours but also in the free time. Epecially I would like to thank to the honorable members of the Princess's room, Marianna and Long, it was great to share the office with you. Thanks also for having read my thesis, and for your good comments and observations.
The first three years of my PhD, I was supported by the Marie Curie Initial Training Network MINO (Mixed-Integer Nonlinear Optimization) funded by the European Union. The last year was funded by the DFG under grant BU 2313/4-2. I thank the chance I had to participate in these projects.
During these four years I have got to know people which had enriched my stay in Germany and make it unique. Eunice, Gaby, Jasmin, Laura, Malwina and Paula, thanks girls for making my time here much better.
A very special gratitude goes my family: estoy muy agradecida con mi familia, primeramente por alentarme a tomar el reto de hacer un doctorado y segundo por darme su inestimable apoyo en todo el camino. A pesar de la distancia nunca los sentí lejos y estuvieron siempre para mi cuando los necesite. Finalmente agradezco a mi esposo, definitivamente esta es una meta que hemos logrado cumplir juntos, sin su apoyo esto no hubiese sido posible. Hacemos un gran equipo!

Dortmund, May 2017

Contents

Chapter 1

Introduction

Quadratic programming (QP) problems require the minimization (or maximization) of a quadratic objective function subject to a set of linear equality or inequality constraints on the variables. QP constitutes an important class of problems in mathematical programming, its importance is twofold. On the one hand, QP with linear constraints can be seen as the most natural generalization of linear programming problems, where the linear objective function is replaced by a quadratic function. Moreover, the algorithmic importance of quadratic programming lies on the fact that it forms a principal computational component of several non-linear programming algorithms such as sequential quadratic programming (see, e.g.,[32]). On the other hand, there are several classes of problems that are naturally expressed as quadratic problems, the most classical ones include portfolio optimization, support vector machines, facility allocation, quadratic assignment problems. It is thus clear that QP is relevant from both the theoretical and practical point of view.

A general quadratic program has the following form

$$\begin{aligned} \min \quad & x^\top Q x + l^\top x + c \\ \text{s.t.} \quad & x \in \mathcal{X}, \end{aligned} \tag{QP}$$

where $Q \in \mathbb{R}^{n \times n}$ is an $n \times n$ symmetric matrix, l is a vector in $\mathbb{R}^n$, and c is a real number. The feasible region $\mathcal{X}$ is specified by constraints on the variable domains and by equality or inequality constraints.

In terms of complexity, solving (QP) is hard in general, except for a few cases that are known to be solvable in polynomial time. If the matrix Q is positive definite and $\mathcal{X}$ is convex, then (QP) becomes a *convex programming* problem and thus it can be solved in polynomial time [56]. There exist many efficient algorithms that can be applied to solve these problems (see, e.g., [29] and the references therein). In particular, it has been shown that when the input data of the problem is rational the ellipsoid method can be applied to solve convex quadratic programming problems in polynomial time [24]. Other polynomial time algorithms such as interior point methods have been proposed to solve convex quadratic problems, for some references see, e.g., [67].

Problem (QP) becomes NP-hard by imposing integrality on the variables even in the unbounded convex case. It is well-known that *convex quadratic optimization with unbounded integer* variables is equivalent to the *closest vector problem*, which has been proved to be NP-hard (see, e.g.,[30]). One of the classical algorithms to solve these problems was proposed by Fincke and Pohst [28]. More recently, a branch-and-bound approach was presented by Buchheim et al. in [13], and later improved in [14]. Other

methods for solving this kind of problems are reformulations through linearization (see, e.g., [58]) or semidefinite programming [21, 17].

Now, when Q is not positive semidefinite, (QP) is a non-convex problem. *Non-convex quadratic programming* has been also proved to be NP-hard. The first result reported in this direction was presented by Sahni [81]. The author proved that for a negative definite matrix Q, Problem (QP) is NP-hard. A similar result was also proved by Vavasis [90, 91] and Pardalos [71]. *Non-convex quadratic programming with box constraints*, i.e., constraints given by lower and upper bounds on the variables, is a fundamental NP-hard problem in global optimization. It is common to assume that the box constraints have the form $x \in [0,1]^n$. This problem is also called *BoxQP*. If Q is negative definite, i.e., the problem is concave, thus BoxQP has a global optimum which is found in one of the extreme points of the box constraints. In the indefinite case, although BoxQP is a continuous optimization problem, it is well-known to be NP-hard. BoxQPs are typically solved by branch-and-bound methods based on convex relaxations, among the most notable relaxations are linear relaxations obtained by applying the reformulation-linearization technique [62, 84], and semidefinite relaxations, see, e.g., [18]. Stronger relaxations have been proposed by a combination of both techniques, see, e.g., [22].

Probably one of the most relevant cases in QP, in the discrete case, is the *unconstrained binary quadratic programming* problem (UBQP), here $\mathcal{X} = \{0,1\}^n$. When linear constraints are added, this simple model embraces a wide range of applications in combinatorial optimization, including optimization problems on graphs, facility locations problems, 0-1 quadratic knapsack problems, among others. For a detailed description of applications see, e.g., the survey paper by Kochenberger et al. [54]. In this case, convexity of the objective function of Problem (QP) can be assumed without loss of generality, since non-convex quadratic functions in binary variables can be easily turned into convex ones [41]. In fact, this is done by adding a penalty term $\gamma \sum_{i=1}^{n}(x_i^2 - x_i)$ to the objective function, and determining a value of $\gamma > 0$ that makes the matrix $Q + \gamma I$ positive semidefinite. It has been proved (see, e.g., [70, 5, 85]) that UBQP is equivalent to the *maximum cut* (max-cut) problem, which is known to be NP-hard [52, 31]. Many different approaches have been presented to solve UBQP, these include linearization techniques, branch-and-bound algorithms, cutting plane methods, use of polyhedral theory and reformulation through semidefinite programming. See, e.g., [19, 54] for a summary of solution approaches for UBQP.

As it can be seen, semidefinite programming (SDP) is a common technique to address different types of quadratic problems. In this thesis, we will focus on solving the semidefinite programming problems arising in the reformulation of non-convex quadratic integer programs, where the only constraints are on variable domains, namely:

$$\begin{aligned} \min \quad & x^\top Q x + l^\top x + c \\ \text{s.t.} \quad & x \in D_1 \times \cdots \times D_n, \end{aligned} \tag{1.1}$$

where $D_i \subseteq \mathbb{Z}$. This thesis follows the line of research of the work of Buchheim and Wiegele [17], the authors proposed the use of semidefinite relaxations and a specialized branching scheme for solving unconstrained non-convex quadratic minimization problems where the variable domains are arbitrary closed subsets of $\mathbb{R}$. Their work is a generalization of the semidefinite programming approach to the max-cut problem or, equivalently, to the UBQP problem [59, 35]. The main idea behind it is the reformula-

tion of Problem (1.1) as a semidefinite problem, and the solution of a relaxation of the transformed problem within a branch-and-bound framework called Q-MIST. At each node of the branch-and-bound tree, Q-MIST calls an interior point method to solve a semidefinite relaxation obtained from Problem (1.1). It is well-known that interior point algorithms are theoretically efficient to solve semidefinite programs, they are able to solve medium to small size problems with high accuracy, but they are memory and time consuming, becoming less useful for large scale instances. For a survey on interior point methods for SDP see, e.g., [93].

Several researchers have proposed other approaches for solving SDPs that all attempt to overcome the practical difficulties of interior point methods. The most common ones include bundle methods [44] and (low rank) reformulation of the SDP as an unconstrained non-convex optimization problem together with the use of non-linear methods to solve the resulting problem [47, 20, 37]. Recently, another algorithm has been proposed by Dong [26] for solving a class of semidefinite programs. The author reformulates Problem (1.1) as a convex quadratically constrained problem, then convex relaxations are produced via a cutting surface procedure based on diagonal perturbations. The separation problem turns out to be a semidefinite problem with convex non-smooth objective function, and it is solved by a primal barrier coordinate minimization algorithm with exact line search.

Our main research focuses on improving Q-MIST by using an alternative method for solving the SDP relaxation of Problem (1.1). Our approach tries to exploit the specific problem structure of Problem (1.1), namely a small total number of (active) constraints and low rank constraint matrices that appear in the semidefinite relaxation. We exploit this special structure by solving the dual problem of the semidefinite relaxation of Problem (1.1), by means of a coordinate ascent algorithm that adapts and generalizes the algorithm proposed in [26].

Outline

This thesis is organized as follows. In Part I, we review all the necessary background on semidefinite programming. The second part contains, what we have called, the main ingredients for the approach presented in Part III.

Part II starts with Chapter 3, where we recall the branch-and-bound algorithm Q-MIST, rewrite the semidefinite relaxation of Problem (1.1) in a matrix form, compute its dual and point out the properties of this problem that will be used later.

Chapter 4 contains a general introduction to coordinate-wise optimization, and a short section about the Woodbury formula, which will be used in the consecutive section and later. The last section of this chapter has a complete description of the barrier coordinate descent method introduced by Dong [26].

In Part III, Chapters 5 and 6 contain our main contribution. First of all we adapt and extend the coordinate descent algorithm presented in [26]. Then, we improved the first approach by exploiting the special structure of the constraint matrices. We will see that this approach can be easily adapted to more general quadratic problems that include linear constraints.

Finally, in Chapter 7 we evaluate this approach within the branch-and bound framework of Q-MIST. The experiments show that our approach produces lower bounds as strong as the ones provided by Q-MIST and that it runs much faster for instances of large size.

Part I

Preliminaries

Chapter 2

Basics of semidefinite programming

Semidefinite programming (SDP) can be seen as an extension of linear programming (LP), namely, it is the optimization of a linear function over the positive semidefinite cone. As it was pointed out by Helmberg in [42], the development of interior point methods for semidefinite programming made it possible to optimize over this set efficiently. However, solving large scale problems as in linear programming is still out of reach in practice. Semidefinite programs arise in a natural way from problems whose data is given by matrices, in particular quadratic problems. SDP has a wide range of applications in both continuous and combinatorial optimization, see [33] and [89] for some applications of semidefinite programming.

This chapter is organized as follows. We begin by introducing some basic notation and definitions about positive semidefinite matrices. Semidefinite programs and their duals are introduced in the consecutive two sections. Section 2.4 contains a brief description of the geometric properties of the semidefinite programs. An introduction to interior point methods for semidefinite programming is reviewed in Section 2.5. We conclude the chapter describing the reformulation of binary quadratic problems as semidefinite problems.

2.1 Symmetric and positive semidefinite matrices

In this section, we report the notation and some preliminary results on positive semidefinite matrices that will be used throughout the thesis.

Let $M_{m,n}$ denote the set of $m \times n$ real matrices, and $M_n := M_{n,n}$ the set of square matrices of *order* n. We will mainly work with the set of symmetric matrices of order n, which we denote by S_n. The dimension of this vector space is $\binom{n+1}{2}$. The standard *inner product* between two matrices A, B in $M_{m,n}$ is:

$$\langle A, B\rangle = \operatorname{trace}(B^\top A) = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}b_{ij}.$$

From the definition of the inner product the following property is derived: For matrices A, B, C with adequate dimensions, it holds that

$$\langle AB, C\rangle = \left\langle A, CB^\top\right\rangle.$$

Notice that if $A, B \in S_n$,

$$\langle A, B\rangle = \operatorname{trace}(B^\top A) = \operatorname{trace}(BA) = \operatorname{trace}(AB).$$

The spectral decomposition theorem is probably one of the the most important theorems about real symmetric matrices. We need the following definition.

Definition 2.1. A scalar $\lambda \in \mathbb{R}$ is called *eigenvalue* of $A \in S_n$ if

$$Av = \lambda v,$$

for some $v \in \mathbb{R}^n$ with $v \neq 0$. The vector v is called *eigenvector* of A associated with λ.

Theorem 2.1 (Spectral decomposition theorem [48])**.** *Any matrix $A \in S_n$ can be decomposed as*

$$A = \sum_{i=1}^{n} \lambda_i v_i v_i^\top,$$

where $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$ are the eigenvalues of A, and $v_1, \ldots, v_n$ are corresponding eigenvectors which form an orthonormal basis of $\mathbb{R}^n$. In other words, the matrix A admits a factorization of the form $A = P\Lambda P^\top$, where P is the orthonormal matrix whose columns are the vectors v_i, and Λ is the diagonal matrix with the eigenvalues of A in its main diagonal. This factorization of A is also known as eigenvalue decomposition of A.

We have the following definition of positive semidefiniteness.

Definition 2.2.
A matrix $A \in S_n$ is *positive semidefinite* ($A \succeq 0$) if $x^\top Ax \geq 0$ for all $x \in \mathbb{R}^n$. The set of positive semidefinite matrices of order n is denoted by S_n^+.
A matrix $A \in S_n$ is *positive definite* ($A \succ 0$) if $x^\top Ax > 0$ for all $x \in \mathbb{R}^n \setminus \{0\}$. The set of positive definite matrices of order n is denoted by S_n^{++}.

From the definition above some properties of positive semidefinite matrices can be formulated, we state them in the following proposition.

Proposition 2.2.

(i) Each principal sub-matrix of a positive (semi)definite matrix is again positive (semi)definite.

(ii) All diagonal elements of a positive definite matrix are positive, and all diagonal elements of a positive semidefinite matrix are non-negative.

(iii) Let $A_i \in S_{n_i}$ for $i = 1, \ldots, k$. The symmetric matrix

$$A = \begin{pmatrix} A_1 & 0 & \ldots & 0 \\ 0 & A_2 & \ldots & 0 \\ \vdots & \vdots & \ldots & \vdots \\ 0 & 0 & \ldots & A_k \end{pmatrix}$$

is positive (semi)definite if and only if all A_i are positive (semi)definite.

(iv) Let $B \in M_n$ be a non-singular matrix. Then

$$A \in S_n^+ \iff B^\top AB \in S_n^+,$$
$$A \in S_n^{++} \iff B^\top AB \in S_n^{++}.$$

Proof. We prove the proposition for positive semidefinite matrices, it extends easily to positive definite matrices.

(i) This follows by considering the quadratic form $x^\top Ax$ of a positive semidefinite matrix $A \in S_n$. Let $I \subset \{1, \ldots, n\}$ be the set of indices of the rows and columns of any principal sub-matrix of A. Let $x \in \mathbb{R}^n$ such that $x_k = 0$ for $k \notin I$. Then

$$0 \leq x^\top Ax = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j = \sum_{i \in I} \sum_{j \in I} a_{ij} x_i x_j.$$

Hence $A_{I,I} \succeq 0$.

(ii) Every diagonal entry of A is a principal sub-matrix itself and thus is positive semidefinite, i.e., $a_{ii}x^2 \geq 0$ must hold for all $x \in \mathbb{R}$. Hence a_{ii} is non-negative.

(iii) Let $A \succeq 0$, in particular, the principal sub-matrices A_i are so. For the other direction, let $x = (x_1, \ldots, x_k)^\top \in \mathbb{R}^n$ where $x_i \in \mathbb{R}^{n_i}$ and $n = \sum_{i=1}^{k} n_i$, then

$$x^\top Ax = x_1^\top A_1 x_1 + \cdots + x_k^\top A_k x_k \geq 0,$$

since each $x_i^\top A_i x_i \geq 0$.

(iv) Let $A \succeq 0$, $x^\top Ax \geq 0$ for all $x \in \mathbb{R}^n$. Take $x = By$, for any $y \in \mathbb{R}^n$, we have

$$0 \leq x^\top Ax = y^\top B^\top ABy.$$

Hence $B^\top AB \succeq 0$. To prove the other direction, take $y = B^{-1}x$ for any $x \in \mathbb{R}^n$, we have that

$$0 \leq y^\top B^\top ABy = x^\top B^{-\top} B^\top AB B^{-1} x = x^\top Ax.$$

Then $A \succeq 0$.

□

This proposition contains only some of the many properties of positive (semi)definite matrices. See, e.g., [48] for more properties of positive semidefinite matrices. We have the following characterization of positive (semi)definite matrices in terms of its eigenvalues.

Theorem 2.3. *For $A \in S_n$, A is positive semidefinite if and only if all its eigenvalues are non-negative. It is positive definite if and only if all its eigenvalues are positive.*

Proof. From Theorem 2.1, we have that $A = P\Lambda P^\top$, P contains the orthonormal eigenvectors of A and $\Lambda = \operatorname{diag}(\lambda)$ its eigenvalues. Using Proposition 2.2 (iv) with $B := P$, we have that the matrix A is positive (semi)definite if and only if $P^\top AP = \Lambda$ is positive (semi)definite. From the same proposition, (ii), we have that Λ is positive (semi)definite if and only if all its diagonal elements are non-negative (resp. positive).

□

An important property that can be derived immediately from this theorem is that the determinant of a positive semidefinite matrix is non-negative. In fact, the determinant of a matrix is known to be the product of its eigenvalues, being non-negative when all its eigenvalues are non-negative, i.e., when the matrix is positive semidefinite. By the same reasoning, the determinant of a positive definite matrix is positive. This property of the positive semidefinite matrices has been exploited to define *logarithmic barrier functions*, as we will be shown later.

The following proposition can be proved using the last theorem.

Proposition 2.4. *If $A \in S_n^+$ and $a_{ii} = 0$ for some $i \in \{1, \ldots, n\}$, then $a_{ij} = 0$ for all $j \in \{1, \ldots, n\}$.*

Proof. Assume that $a_{ik} \neq 0$ for some $k \in \{1, \ldots, n\}$, without loss of generality we can also assume that $i < k$. Consider the principal sub-matrix of A defined by the rows and columns k and i, i.e.,

$$\begin{pmatrix} a_{ii} & a_{ik} \\ a_{ik} & a_{kk} \end{pmatrix}.$$

The determinant of the matrix above is $-a_{ik}^2$. From Proposition 2.2 (i) we know that every principal sub-matrix of a positive semidefinite matrix is positive semidefinite and from Theorem 2.3 that its determinant should be non-negative. We conclude that $a_{ij} = 0$ for all $j \in \{1, \ldots, n\}$. □

The following theorem states that the inner product of two positive semidefinite matrices is non-negative. In particular, this property is important to prove an essential property of the cone of semidefinite matrices, namely, its self-duality, see Section 2.4.

Theorem 2.5. *Let $A, B \in S_n^+$. Then $\langle A, B \rangle \geq 0$ and $\langle A, B \rangle = 0$ if and only if $AB = 0$.*

Proof. Let $B = P \Lambda P^\top$ be the eigenvalue decomposition of B, Λ is the diagonal matrix with the eigenvalues of B, $\lambda_i \geq 0$, in its main diagonal.

Define $C := P^\top A P$. We have that $C \succeq 0$ and therefore $c_{ii} \geq 0$ by Proposition 2.2 (ii). Therefore

$$\langle A, B \rangle = \left\langle A, P^\top \Lambda P \right\rangle = \left\langle P^\top A P, \Lambda \right\rangle = \langle C, \Lambda \rangle = \sum_{i=1}^{n} c_{ii} \lambda_i.$$

The last sum is non-negative since each term $c_{ii}\lambda_i \geq 0$.

Notice that $[\langle A, B \rangle = 0 \iff AB = 0]$ is equivalent to $[\langle C, \Lambda \rangle = 0 \iff C\Lambda = 0]$. Now, if $\langle A, B \rangle = 0$, then $\langle C, \Lambda \rangle = 0$ and therefore $\sum_{i=1}^{n} c_{ii}\lambda_i = 0$. Since all the terms in this sum are non-negative, it follows that $c_{ii}\lambda_i = 0$ for all i. Thus, if $\lambda_i > 0$, $c_{ii} = 0$ and since $C \succeq 0$, the i-th row/column of C must be zero. If instead, $c_{ii} > 0$, then $\lambda_i = 0$. It remains to prove that $(C\Lambda)_{ij} = 0$ for all j. Suppose that there exist i, j such that $(C\Lambda)_{ij} \neq 0$. Then $c_{ij}\lambda_j \neq 0$, but if $\lambda_j > 0$ then c_{jj} must be zero and therefore the complete row j, which is a contradiction. □

We conclude this section with a theorem that gives a characterization of positive semidefinite matrices using the so-called Schur complement. If A is a non-singular principal sub-matrix of the 2×2-block matrix

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix},$$

then $D - CA^{-1}B$ is known as Schur complement of A in M, named after a seminal lemma by the mathematician Issai Schur [77].

Matrices of the form $D - CA^{-1}B$ are very common in linear algebra, probably one of the most common uses is Gaussian elimination. See, e.g., [25] for more applications of the Schur complement. We are interested in the use of the Schur complement for testing positive (semi)definiteness of a matrix. The following theorem gives a criterion to decide whether a 2×2-block symmetric matrix is positive (semi)definite.

Theorem 2.6 (Schur complement)**.** *Let $A \in S_m^{++}$, $C \in S_n$, and $B \in M_{m,n}$. Then*

$$\begin{pmatrix} A & B \\ B^\top & C \end{pmatrix} \succ 0 \iff C - B^\top A^{-1} B \succ 0$$

and

$$\begin{pmatrix} A & B \\ B^\top & C \end{pmatrix} \succeq 0 \iff C - B^\top A^{-1} B \succeq 0.$$

Proof. We have that

$$M := \begin{pmatrix} I_m & -A^{-1}B \\ 0 & I_n \end{pmatrix}$$

is a non-singular matrix. From the Proposition 2.2 (iv) it follows that

$$\begin{pmatrix} A & B \\ B^\top & C \end{pmatrix}$$

is positive (semi)definite if and only if the matrix

$$M^\top \begin{pmatrix} A & B \\ B^\top & C \end{pmatrix} M = \begin{pmatrix} A & 0 \\ 0 & C - B^\top A^{-1} B \end{pmatrix}$$

is positive (semi)definite. From same proposition, (iii), the last matrix is positive (semi)definite if and only if $C - B^\top A^{-1} B$ is positive (semi)definite. □

In the next chapter we will see that the Schur complement appears also in the so-called Woodbury formula, which plays an important role in the algorithm proposed in Part III.

After having described some basic properties of positive semidefinite matrices, we can introduce the concept of semidefinite programming.

2.2 Semidefinite programming

Consider the following optimization problem, which is known as the *semidefinite program* in standard form:

$$\begin{aligned} &\min \ \langle Q, X \rangle \\ &\text{s.t. } \langle A_i, X \rangle = b_i \quad i = 1, \ldots, m \\ &\qquad X \succeq 0, \end{aligned} \tag{2.1}$$

where the matrices Q, A_i, $i = 1, \ldots, m$ are assumed to be symmetric matrices in M_n and $b \in \mathbb{R}^m$. There is no loss of generality in the assumption of symmetry of the

matrices Q and A_i. Since $\langle Q, X\rangle = \left\langle Q^\top, X\right\rangle$, if Q is not symmetric, one can replace it by $\frac{1}{2}(Q + Q^\top)$. The same is true for the constraint matrices A_i.

To simplify notation, define the linear operator $\mathcal{A}\colon S_n \longrightarrow \mathbb{R}^m$ as

$$\mathcal{A}(X) := \begin{pmatrix} \langle A_1, X\rangle \\ \vdots \\ \langle A_m, X\rangle \end{pmatrix},$$

the above problem is then rewritten in the following form:

$$\begin{aligned} \min\ & \langle Q, X\rangle \\ \text{s.t.}\quad & \mathcal{A}(X) = b \\ & X \succeq 0. \end{aligned} \tag{2.2}$$

The linear operator $\mathcal{A}$ has associated an adjoint operator, denoted by $\mathcal{A}^\top$, which by definition is the linear operator $\mathcal{A}^\top : \mathbb{R}^m \longrightarrow S_n$ satisfying the relation

$$\langle \mathcal{A}(X), y\rangle = \left\langle X, \mathcal{A}^\top y\right\rangle, \quad \forall X \in S_n, y \in \mathbb{R}^m.$$

It follows that

$$\langle \mathcal{A}(X), y\rangle = \sum_{i=1}^n y_i \operatorname{trace}(A_i X) = \operatorname{trace}(X \sum_{i=1}^n y_i A_i) = \left\langle X, \sum_{i=1}^m y_i A_i \right\rangle,$$

from which we obtain the explicit description

$$\mathcal{A}^\top y = \sum_{i=1}^m y_i A_i.$$

The dual problem of (2.2) is computed as follows: Let $y \in \mathbb{R}^m$ be the dual multipliers associated with the constraints $\langle A_i, X\rangle = b_i$. The primal constraints are lifted into the objective function:

$$\min_{X\succeq 0} \max_{y\in\mathbb{R}^m} \langle Q, X\rangle + y^\top(b - \mathcal{A}(X)),$$

then, exchanging the min with max yields

$$\max_{y\in\mathbb{R}^m} \min_{X\succeq 0} \langle b, y\rangle + \left\langle Q - \mathcal{A}^\top y, X\right\rangle.$$

The inner minimization over $X \succeq 0$ is bounded from below only if $Q - \mathcal{A}^\top y \succeq 0$. Finally, the *dual semidefinite problem* in standard form associated to Problem (2.1) can be rewritten as

$$\begin{aligned} \max\ & \langle b, y\rangle \\ \text{s.t.}\quad & \mathcal{A}^\top y + Z = Q \\ & Z \succeq 0 \\ & y \in \mathbb{R}^m. \end{aligned} \tag{2.3}$$

Later we will deal with semidefinite programs that are usually not in the standard form. Therefore it is important to notice what happens when Problem (2.1) contains equality and inequality constraints, i.e., if we have a primal SDP problem of the form

$$\begin{aligned} \min \ & \langle Q, X \rangle \\ \text{s.t.} \ & \mathcal{A}_1(X) = b_1 \\ & \mathcal{A}_2(X) \leq b_2 \\ & X \succeq 0, \end{aligned}$$

where $\mathcal{A}_1 \colon S_n \longrightarrow \mathbb{R}^{m_1}$, $\mathcal{A}_2 \colon S_n \longrightarrow \mathbb{R}^{m_2}$ are two linear operators, $b_1 \in \mathbb{R}^{m_1}$ and $b_2 \in \mathbb{R}^{m_2}$. In this case, the dual problem will have additional dual variables $t \in \mathbb{R}^{m_1}_-$:

$$\begin{aligned} \max \ & \langle b_1, y \rangle + \langle b_2, t \rangle \\ \text{s.t.} \ & \mathcal{A}_1^\top(y) + \mathcal{A}_2^\top(t) + Z = Q \\ & Z \succeq 0 \\ & y \in \mathbb{R}^{m_1} \\ & t \in \mathbb{R}^{m_2}_-. \end{aligned}$$

Observe that similar to linear programming, the set of constraints $\mathcal{A}_2(X) \leq b_2$ could be transformed into equality constraints by introducing non-negative slack variables $x_1, \ldots, x_{m_2}$, and then replacing X by a new matrix X' of the form

$$X' = \begin{pmatrix} X & 0 & 0 & \ldots & 0 \\ 0 & x_1 & 0 & \ldots & 0 \\ 0 & 0 & x_2 & \ldots & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & \ldots & x_{m_2} \end{pmatrix},$$

which is positive semidefinite if and only if $X \succeq 0$ and $x_1, \ldots, x_{m_2} \geq 0$. This will increase however the dimension of the problem.

In the special case when Q and A_i are diagonal matrices, Problem (2.1) reduces to a linear program. Indeed, let q and a_i denote the diagonal vectors of Q and A_i respectively, and x the diagonal entries of the matrix X. It holds that $x \geq 0$ if and only if $X \succeq 0$. We obtain that Problem (2.1) is in fact the linear problem:

$$\begin{aligned} \min \ & q^\top x \\ \text{s.t.} \ & a_i^\top x = b_i \quad i = 1, \ldots, m \\ & x \geq 0. \end{aligned}$$

Note that $\langle q, x \rangle = q^\top x$ and $\langle a_i, x \rangle = a_i^\top x$.

2.3 Duality theory

The property that the objective value of any primal feasible solution is greater or equal to the objective value of any dual feasible solution is called *weak duality*. We have the following

Lemma 2.7. *Let X be a primal feasible solution of Problem (2.2) and (y, Z) any dual feasible solution of Problem (2.3). Then $\langle Q, X\rangle \geq \langle b, y\rangle$.*

Proof. We have that:

$$\begin{aligned}\langle Q, X\rangle - \langle b, y\rangle &= \left\langle \mathcal{A}^\top y + Z, X\right\rangle - \langle \mathcal{A}(X), y\rangle \\ &= \left\langle \mathcal{A}^\top y, X\right\rangle + \langle Z, X\rangle - \langle \mathcal{A}(X), y\rangle \\ &= \langle \mathcal{A}(X), y\rangle + \langle Z, X\rangle - \langle \mathcal{A}(X), y\rangle \\ &= \langle Z, X\rangle \geq 0.\end{aligned}$$

The last inequality is true, because the inner product of two positive semidefinite matrices is non-negative, see Theorem 2.5. □

The quantity $\langle Z, X\rangle$ is known as *duality gap*. If the duality gap is zero, it is said that *strong duality* holds, then X and (y, Z) are primal dual optimal. In semidefinite programming, different from linear programming, it can happen that the duality gap is not zero for a primal-dual optimal pair or that the optimal value is not attained. We illustrate these facts by the following examples taken from [93].

Example 2.1. Consider

$$\begin{aligned}&\min\ ax_{11} \\ &\text{s.t.}\ \ x_{11} + 2x_{23} = 1 \\ &\qquad\quad x_{22} = 0 \\ &\qquad\quad X \in S_3^+,\end{aligned} \tag{P}$$

or, equivalently,

$$\begin{aligned}\min\ &\left\langle \begin{pmatrix} a & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, X \right\rangle \\ \text{s.t.}\ &\left\langle \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, X \right\rangle = 1 \\ &\left\langle \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, X \right\rangle = 0 \\ &\qquad\qquad X \succeq 0.\end{aligned}$$

Any feasible solution of (P) must satisfy $x_{23} = 0$, since $x_{22} = 0$, see Proposition 2.4. Therefore, (P) has optimal value a. Now, let us compute the dual problem:

$$\begin{aligned}\max\ &\left\langle \begin{pmatrix} 1 \\ 0 \end{pmatrix}, y \right\rangle \\ \text{s.t.}\ \ &y_1 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} + y_2 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} + Z = \begin{pmatrix} a & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ &Z \succeq 0,\end{aligned} \tag{D}$$

i.e.,

$$\begin{aligned} &\max\ y_1 \\ &\text{s.t.} \quad \begin{pmatrix} a - y_1 & 0 & 0 \\ 0 & -y_2 & -y_1 \\ 0 & -y_1 & 0 \end{pmatrix} \succeq 0. \end{aligned}$$

For each feasible solution of (D) it holds that $y_1 = 0$. Hence, its optimal objective value is 0. The optimal objective values of (P) and (D) are different, their gap is a.

Now, consider the following example, where the duality gap is zero, but the optimal value is not attained.

Example 2.2. Consider the following primal-dual pair of semidefinite problems

$$\begin{aligned} p^* = \min\ & x_{11} \\ \text{s.t.}\ & \begin{pmatrix} x_{11} & -1 \\ -1 & x_{22} \end{pmatrix} \succeq 0 \end{aligned}$$

and

$$\begin{aligned} d^* = \max\ & -2y_1 \\ \text{s.t.}\ & \begin{pmatrix} 1 & -y_1 \\ -y_1 & 0 \end{pmatrix} \succeq 0. \end{aligned}$$

Observe that both optimal objective values are zero. However, the infimum is not attained, since it is reached only in the limit when $x_{11} = \frac{1}{x_{22}}$ and $x_{22} \to \infty$.

The gap between primal and dual optimal objective values is guaranteed to be zero if at least one of both primal or dual problems has a strictly feasible point.

Definition 2.3. A matrix X is said to be *strictly feasible* for Problem (2.2) if it satisfies $\mathcal{A}(X) = b$ and $X \succ 0$. A pair (y, Z) is said to be *strictly feasible* for Problem (2.3) if it satisfies $\mathcal{A}^\top y + Z = Q$ and $Z \succ 0$.

It is said that the *Slater condition* holds for the primal problem, if there exists a strictly feasible matrix X for the primal problem (2.2), i.e., the intersection of $\mathcal{A}(X) = b$ and $\text{int}(S_n^+)$ is non-empty, the same holds for the dual problem (2.3). Later we will see that $\text{int}(S_n^+) = S_n^{++}$.

Theorem 2.8. *If both problems, namely Problem* (2.2) *and Problem* (2.3)*, are strictly feasible, then the duality gap is zero and both problems admit an optimal solution.*

For a proof of this theorem, see e.g., [92], as a special case of duality of linear programs on cones, or [83], for duality of general SDP.

It is possible to derive a version of the complementary slackness for SDP similar to linear programming. The proof follows immediately from Theorem 2.5.

Theorem 2.9. *Let X^* be a feasible primal solution of Problem* (2.2) *and (y^*, Z^*) a feasible dual solution of Problem* (2.3)*. Then X^* and (y^*, Z^*) are primal dual optimal, respectively, if and only if*

$$X^* Z^* = 0.$$

From the last theorem, it follows that if the Slater condition holds for both sides, a pair X and (y, Z) is primal-dual optimal if and only if

$$\begin{aligned} \mathcal{A}X = b, \quad X \succeq 0, \\ \mathcal{A}^\top y + Z = Q, \quad Z \succeq 0, \\ XZ = 0. \end{aligned} \tag{2.4}$$

These conditions are referred to as *KKT conditions for semidefinite programs.*

2.4 The positive semidefinite cone

In this section we discuss some important geometric properties of the set of positive semidefinite matrices. It follows immediately from Definition 2.2 that for $A, B \in S_n^+$, $\lambda A + (1-\lambda)B$ is positive semidefinite, for all $0 \leq \lambda \leq 1$. The same is true for S_n^{++}, thus both sets are convex.

A subset $K \subset S_n$, is called a *cone* if and only if $A \in K$ and $\alpha \geq 0$ implies $\alpha A \in K$. A cone is *pointed* if $K \cup (-K) = \{0\}$, and *full-dimensional* if its interior is nonempty. The *interior* of K, denoted by $\text{int}(K)$ is the largest open subset contained in K.

Proposition 2.10. *The set of positive semidefinite matrices, S_n^+, is a proper cone, i.e., it is a convex, pointed, closed and full-dimensional cone.*

Proof. For $A \in S_n^+$, and $\alpha A \in S_n^+$, it follows directly from the positive semidefiniteness of A that $x^\top(\alpha A)x = \alpha x^\top Ax \geq 0$ for all $x \in \mathbb{R}^n$. Therefore S_n^+ is a cone.

To prove that it is pointed, let $A \in S_n^+ \cup (-S_n^+)$, then $A \in S_n^+$ and $-A \in S_n^+$. Now, for any $x \in \mathbb{R}^n$, both $x^\top Ax \geq 0$ and $x^\top(-A)x \geq 0$ holds. Therefore A must be the zero matrix.

Define the following function

$$\begin{aligned} \lambda \colon S_n &\longrightarrow \mathbb{R}^n \\ A &\longmapsto \begin{pmatrix} \lambda_1(A) \\ \vdots \\ \lambda_n(A) \end{pmatrix}, \end{aligned}$$

where $\lambda_1(A) \geq \cdots \geq \lambda_n(A)$ are the eigenvalues of A. Notice that λ is a continuous function. Therefore, the inverse image of the closed set $[0,\infty)^n$, $\lambda^{-1}([0,\infty)^n)$, is closed in S_n. From Theorem 2.3 we derive $S_n^+ = \lambda^{-1}([0,\infty)^n)$.

We can see that S_n^+ is full-dimensional since $\lambda^{-1}((0,1)^n)$ is open in S_n^+ and contains the matrix $\frac{1}{2}I$, thus the interior of S_n^+ is not empty. □

Notice that S_n^{++} is not a cone, since the null matrix does not belong to it. One can prove that S_n^{++} is in fact the interior of S_n^+.

Proposition 2.11. $\text{int}(S_n^+) = S_n^{++}$.

Proof. Let λ be defined as in the last proof. Let $A \in \text{int}(S_n^+)$, then there exists $\epsilon > 0$ such that $A - \epsilon I \in S_n^+$. We have that $\lambda(A - \epsilon I) = \lambda(A) - \epsilon\lambda(I) = \lambda(A) - \epsilon \geq 0$ (see Theorem 2.3). Then $\lambda(A) > 0$ and therefore $A \in S_n^{++}$.

Conversely, let $A \in S_n^{++}$, it follows that $\lambda(A) > 0$, from Theorem 2.3. Then $\lambda(A) \in (0, 2\lambda(A))$, which implies that $A \in \lambda^{-1}((0, 2\lambda(A))) \subset S_n^+$, i.e., A is in the interior of S_n^+. □

This property of S_n^+ is particularly important in barrier methods as the interior point methods described in the following section and in Part III.

For a proper cone K, its *dual cone* K^* is defined as

$$K^* := \{A \in S_n \mid \langle A, B\rangle \geq 0 \quad \forall B \in K\},$$

and it is said that K is self-dual if $K = K^*$.

The semidefinite cone is self-dual, i.e.,

$$S_n^+ = (S_n^+)^* := \{Y : \langle X, Y\rangle \geq 0 \quad \forall X \in S_n^+\}.$$

The proof follows from the following

Proposition 2.12. *A matrix $A \in S_n$ is positive semidefinite if and only if $\langle A, B\rangle \geq 0$ for all $B \in S_n^{++}$.*

Proof. Suppose that $A, B \succeq 0$. From Theorem 2.5 it follows that $\langle A, B\rangle \geq 0$. Conversely, if $\langle A, B\rangle \geq 0$ for all $B \in S_n^+$, let $x \in \mathbb{R}^n$ and take $B = xx^\top$. Then $\langle A, B\rangle = \left\langle A, xx^\top \right\rangle = x^\top Ax \geq 0$. □

We review now some properties of the facial structure of the cone of positive semidefinite matrices, first we recall the definition of a face of a cone. If K is a closed cone, $F \subset K$ is said to be a face of K if it is a sub-cone, and if for any pair of elements $X, Y \in K$ such that $X + Y \in F$, it holds that $X, Y \in F$.

The next theorem gives a characterization of the faces of the cone of positive semidefinite matrices.

Theorem 2.13. *Any face F of the semidefinite cone falls in one of the following cases*

(i) $F = \emptyset$,

(ii) $F = \{0\}$,

(iii) $F = \{X \in S_n : X = PSP^\top, S \in S_k^+\}$ *with* $P \in M_{n,k}$ *and* $\operatorname{rank}(P) = k$, *for* $k \in \{1, \ldots, n\}$.

For a proof of this theorem see [43] and the references therein. The feasible set of any semidefinite program is the intersection of an affine sub-space defined by the linear constraints with the semidefinite cone. In case of the primal problem (2.2), we denote the feasible set by

$$\mathcal{P} := \{X \in S_n : \mathcal{A}(X) = b, X \succeq 0\}.$$

For the dual problem (2.3), the feasible set is

$$\mathcal{D} := \{(y, Z) \in \mathbb{R}^m \times S_n : Z \succeq 0\}.$$

The semidefinite cone is not a polyhedral cone, therefore in general the feasible sets $\mathcal{P}$ and $\mathcal{D}$ are not polyhedral.

It is well known that the faces of the intersection of convex sets are the intersections of the faces of the sets. The faces of $\mathcal{P}$ and $\mathcal{D}$ can be directly derived from Theorem 2.13.

Corollary 2.14. *Any face of $\mathcal{P}$ is a set of the following form*

$$\{X \in \mathcal{P} : X = PSP^\top, S \in S_k^+\}$$

for a certain matrix $P \in M_{n,k}$ with $k = \operatorname{rank}(P)$. Any face of $\mathcal{D}$ is a set of the following form

$$\{(y, Z) \in \mathcal{D} : Z = QDQ^\top, D \in S_r^+\},$$

for a certain matrix $Q \in M_{n,r}$, with $r = \operatorname{rank}(Q)$.

Optimal solutions of problems (2.2) and (2.3) are expected to have small rank [73]. The next property is related with upper bounds for the rank of a matrix contained in a face of $\mathcal{P}$ or $\mathcal{D}$.

Theorem 2.15. *[73] Let F be a face of dimension d of the feasible set of (2.2). For $X \in F$ the rank $r = \operatorname{rank}(X)$ is bounded by*

$$\binom{r+1}{2} \leq m + d.$$

Let F be a face of dimension d of $\{Z \succeq 0 : \exists y \in \mathbb{R}^n : Q - \mathcal{A}^\top y = Z\}$. For $(y, Z) \in F$, the rank $r = \operatorname{rank}(Z)$ is bounded by

$$\binom{r+1}{2} \leq \binom{n+1}{2} - m + k.$$

2.5 Interior point methods for SDP

Semidefinite programs are in particular convex optimization problems. It is proved in the work of Grötschel, Lovóvasz and Schrijver [38] that they can be solved in polynomial time to any desired precision using for instance the Ellipsoid method (see e.g. [39]). From a practical point of view, Ellipsoid methods are not efficient for most applications, including SDP. The main approaches for solving SDP problems are interior point methods and first order non-linear methods. Interior point methods provide the possibility of obtaining polynomial time algorithms for semidefinite programs, they are able to solve small to medium size problems with high accuracy, but they are memory and time consuming so that they become less useful when solving large scale instances.

Interior point methods are a very large class of methods, developed with several variants. Refer for instance to the book [53] for a complete treatment. We describe here a primal-dual interior point method that solves (2.2) and (2.3) simultaneously, with the intention of getting an idea of how they work and understand its limitations with large size instances.

A valid interior point algorithm requires to assume that Slater's condition is satisfied for both primal and dual problems. For $\mu > 0$, consider the following perturbed system of the KKT conditions (2.4):

$$\begin{aligned} \mathcal{A}X = b, \quad X \succ 0, \\ \mathcal{A}^\top y + Z = Q, \quad Z \succ 0, \\ XZ = \mu I. \end{aligned} \tag{2.5}$$

The latter system can also be obtained as the KKT conditions of the barrier problem for (2.3)

$$\begin{aligned} \min \quad & \langle b, y\rangle - \mu \log\det Z \\ \text{s.t.} \quad & \mathcal{A}^\top y + Z = Q \\ & y \in \mathbb{R}^m, \end{aligned}$$

where μ is the penalty parameter. The barrier term $-\mu \log\det Z$ will avoid that Z leaves the interior of the positive semidefinite cone. Indeed, $\det Z \to 0$ when Z approaches the boundary of the semidefinite cone, and thus the barrier term will grow to infinity. From Proposition 2.11 we know that Z will remain positive definite during the course of the iterations.

The set of solutions (X_μ, y_μ, Z_μ) of system (2.5) for $\mu > 0$, is the so-called *central path*. It was shown that the central path is a smooth curve [55]. Additionally, it can be proved that if $\mu \to 0$ the central path converges to a point (X^*, y^*, Z^*) such that X^* is an optimal solution of the primal problem and (y^*, Z^*) is an optimal solution of the dual problem (see, e.g, [43]). Therefore, the idea of interior point methods is to compute an approximate solution of (2.4) by solving (2.5). To simplify notation, define

$$F_\mu(X, y, Z) := \begin{pmatrix} \mathcal{A}X - b, \\ \mathcal{A}^\top y + Z - Q \\ XZ - \mu I \end{pmatrix},$$

and system (2.5) corresponds to

$$F_\mu(X, y, Z) = 0, \quad X, Z \succeq 0.$$

Newton's method is applied to find a direction $(\Delta X, \Delta y, \Delta Z)$ towards the optimal point (X^*, y^*, Z^*), which must satisfy

$$F_\mu + \nabla F_\mu \cdot (\Delta X, \Delta y, \Delta Z) = 0.$$

Therefore the direction $(\Delta X, \Delta y, \Delta Z)$ is the solution of the linearized system

$$\begin{aligned} \mathcal{A}\Delta X &= -(\mathcal{A}X - b), \\ \mathcal{A}^\top \Delta y + \Delta Z &= -(\mathcal{A}^\top y + Z - Q), \\ \Delta X Z + X \Delta Z &= \mu I - XZ. \end{aligned} \tag{2.6}$$

In general, XZ is not symmetric and the direct application of Newton's method will produce non-symmetric matrices ΔX and ΔZ. Several approaches have been proposed to deal with this problem. We do not go into details, but refer to the survey [87] for a complete description about search directions for interior point methods in semidefinite programming. In Algorithm 1, we summarize a general framework for interior point methods for SDP.

2.6 SDP for binary quadratic programming

SDP has wide applicability in combinatorial optimization. Semidefinite programs provide a powerful tool for constructing strong convex relaxations for several NP-hard combinatorial optimization problems. We illustrate the main idea on the unconstrained

Algorithm 1: Primal-dual interior point method

Input: $\mathcal{A}(\cdot), b \in \mathbb{R}^n, Q \in M_n$, starting point (X^0, y^0, Z^0) $X \succ 0$, $Z \succ 0$, μ

1 update $\mu > 0$ compute $(\Delta X, \Delta y, \Delta Z)$ by solving (2.6);

2 choose $\alpha \in (0, 1)$ such that $X + \alpha\Delta X$, $y + \alpha\Delta y$ and $Z + \alpha\Delta Z$ remain feasible;

3 if $\|\mathcal{A}X - b\|$ and $\|\mathcal{A}^\top y + Z - Q\|_F$ and $\langle X, Z\rangle$ are small enough then **stop**, else **goto** 1.

quadratic binary optimization problem. In the next chapter, we will see that this semidefinite relaxation can be extended to more general quadratic problems.

For many problems in combinatorial optimization the integer program and its linear relaxation optima may be quite different. This is particularly the case for the max-cut problem [34]. One approach to deal with this large discrepancy between the optima of the original problem and its linear relaxation is to consider relaxations tighter than linear programming relaxations. It has been shown that using semidefinite programming to approximate max-cut produces tighter relaxations than the linear relaxation, the first work in this direction was presented by Goemans and Williamson [35]. This approach and their result had an important impact on the area of combinatorial optimization, leading to a lot of research activity for getting tight approximations for various problems.

Consider the unconstrained binary quadratic optimization problem

$$\max_{x \in \{0,1\}^n} x^\top Q x. \tag{2.7}$$

In order to reformulate this problem, we introduce the following change of variable $X = xx^\top$, with $x \in \{0, 1\}^n$. Observe that, by making use of the inner product of matrices, the objective function of the above problem can be equivalently rewritten as

$$x^\top Q x = \langle Qx, x\rangle = \left\langle Q, xx^\top \right\rangle.$$

Also, observe that X is a rank-one positive semidefinite matrix, and its diagonal elements are all equal to 0 or 1. This leads to a non-convex formulation of Problem (2.7),

$$\begin{aligned} \max \quad & \langle Q, X\rangle \\ \text{s.t.} \quad & \operatorname{rank}(X) = 1 \\ & x_{ii} \in \{0, 1\} \\ & X \succeq 0. \end{aligned} \tag{2.8}$$

Removing the rank-one constraint and relaxing the constraint $x_{ii} \in \{0, 1\}$ by $0 \leq x_{ii} \leq 1$ give a semidefinite relaxation of Problem (2.7):

$$\begin{aligned} \max \quad & \langle Q, X\rangle \\ \text{s.t.} \quad & 0 \leq x_{ii} \leq 1 \\ & X \succeq 0. \end{aligned}$$

However, as mentioned in [46], this relaxation turns out to be of poor quality and one can get tighter relaxations. In fact, the non-convex constraint $X - xx^\top = 0$ can be relaxed to the convex constraint $X - xx^\top \succeq 0$. Also, exploiting the fact

that $\text{diag}(xx^\top) = x$, we obtain the following semidefinite relaxation for 0-1 quadratic programming

$$\begin{aligned} \max \quad & \langle Q, X \rangle \\ \text{s.t.} \quad & X - \text{diag}(X)\text{diag}(X)^\top \succeq 0. \end{aligned}$$

Using the Schur complement, Theorem 2.6, the semidefinite constraint

$$X - \text{diag}(X)\text{diag}(X)^\top \succeq 0$$

is equivalent to

$$\bar{X} := \begin{pmatrix} 1 & \text{diag}(X)^\top \\ \text{diag}(X) & X \end{pmatrix} \succeq 0.$$

We have the following

Lemma 2.16. *Problem* (2.7) *is equivalent to*

$$\begin{aligned} \max \quad & \langle Q, X \rangle \\ \textit{s.t.} \quad & \text{rank}(\bar{X}) = 1 \\ & \bar{X} \succeq 0. \end{aligned} \tag{2.9}$$

Proof. Let $x \in \mathbb{R}^n$ be a feasible solution of (2.7), and define $X = xx^\top$. We have that

$$\bar{X} := \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix}$$

is a feasible solution of (2.9): $\bar{x}_{ii} = x_i^2 \in \{0, 1\}$ for all $i \in \{1, \ldots, n\}$, i.e., $x = \text{diag}(X)$. It is clear that the rank of $\bar{X}$ is 1. Moreover, since $v^\top X v = v^\top x x^\top v = (v^\top x)^2 \geq 0$, for all $v \in \mathbb{R}^n$, X is positive semidefinite and by Theorem 2.6, $\bar{X}$ is positive semidefinite as well.

Now, choose a feasible solution $\bar{X}$ of (2.9). We have that

$$\bar{X} = \begin{pmatrix} 1 & \text{diag}(X)^\top \\ \text{diag}(X) & X \end{pmatrix} \succeq 0$$

which by Theorem 2.6 implies that $X \succeq 0$. Now, from $\text{rank}(\bar{X}) = 1$, it follows that $X = \text{diag}(X)\text{diag}(X)^\top$ and $x_{ii} = x_{ii}^2 \in \{0, 1\}$. Take $x := \text{diag}(X)$, thus $x_i \in \{0, 1\}$ for all $i \in \{1, \ldots, n\}$, i.e., x is a feasible solution of (2.7). □

Problem (2.7) is known to be equivalent to the well-known max-cut problem which is modeled on $\{-1, 1\}^n$ variables, see for instance [5]. The latter results can be extended to more general quadratic optimization problems with linear and/or quadratic constraints. Consider the following 0-1 quadratic program with one inequality constraint

$$\begin{aligned} \max \quad & x^\top Q x \\ \text{s.t.} \quad & a^\top x \leq b, \\ & x \in \{0, 1\}, \end{aligned}$$

with $Q \in S_n$, $a \in \mathbb{N}^n$ and $b \in \mathbb{N}$. This problem can be interpreted as the quadratic knapsack problem [74]. We are given a knapsack with capacity b and a set of items $\{1, \ldots, n\}$,

each one characterized by a positive weight a_i, and the profit q_{ii} obtained if the item i is selected. In addition, q_{ij} is the profit generated if both items i and j are selected. The quadratic knapsack problem asks for a selection of a subset of items that does not exceed the capacity of the knapsack and gives maximum profit.

Following the linearization described above, the knapsack constraint $a^\top x \leq b$ can be modeled by restricting the diagonal elements of X

$$\langle \mathrm{Diag}(a), X \rangle \leq b,$$

yielding a semidefinite relaxation of the 0-1 quadratic knapsack problem

$$\begin{array}{ll} \max & \langle Q, X \rangle \\ \text{s.t.} & \langle \mathrm{Diag}(a), X \rangle \leq b \\ & \bar{X} \succeq 0. \end{array}$$

The inequality $a^\top x \leq b$ can be represented in other ways, aiming at producing tighter relaxations, see, e.g., [46] for literature on how to model the knapsack constraint. On the other hand, it is clear that this approach can be easily generalized to model more than one linear constraint.

Part II

Main ingredients

Chapter 3

Q-MIST

In this chapter, we focus on quadratic problems with non-convex objective function in which the only constraints are the ones that enforce each variable to belong to a specific finite sub-set of $\mathbb{Z}$. A study of slightly more general version of these problems was done by Buchheim and Wiegele [17], they considered arbitrary closed sub-sets of $\mathbb{R}$ and proposed to solve a semidefinite relaxation of such problems within a branch-and-bound framework called Q-MIST: Quadratic Mixed-Integer Semidefinite Programming Technique. This chapter will be dedicated to recall some of the main results of the work of Buchheim and Wiegele [17], and present some preliminary results that are the basis for our contribution in Part III.

We thus consider non-convex quadratic mixed-integer optimization problems of the form

$$\begin{aligned} \min \quad & \hat{f}(x) = x^\top \hat{Q} x + \hat{l}^\top x + \hat{c} \\ \text{s.t.} \quad & x \in D_1 \times \cdots \times D_n, \end{aligned} \tag{3.1}$$

where $\hat{Q} \in S_n$ is not necessarily positive semidefinite, $\hat{l} \in \mathbb{R}^n$, $\hat{c} \in \mathbb{R}$, and $D_i \subseteq \mathbb{R}$ for all $i = 1, \ldots, n$.

This problem formulation encloses a wide range of quadratic problems, depending on the definition of D_i. All the domains D_i might be taken equal, for example $D_i = \mathbb{R}$ or $D_i = \mathbb{Z}$ in which case (3.1) reduces to optimizing a quadratic function over $\mathbb{R}^n$, or $\mathbb{Z}^n$. By setting $D_i = \{0, 1\}$ we have the unconstrained binary quadratic problem (see Section 2.6), and the mixed-integer formulation of (3.1) is obtained by taking different sets D_i. In this thesis we consider finite sets D_i of the form $\{l_i, \ldots, u_i\}$ with $l_i, u_i \in \mathbb{Z}$.

We begin by describing how to obtain a semidefinite relaxation of Problem (3.1), then we introduce Algorithm Q-MIST. In the successive two sections we formulate the semidefinite relaxation of Problem (3.1) in a matrix form and compute the dual problem, respectively. In the last section we prove strict feasibility of the semidefinite relaxation and its dual.

3.1 Semidefinite relaxation

Semidefinite relaxations for quadratic problems can already be found in an early paper of Lovász in 1979 [59], but it was not until the work of Goemans and Williamson in 1995 [35] that they started to catch a wider interest. For such problems, the basic idea

of a semidefinite relaxation is as follows: given any vector $x \in \mathbb{R}^n$, the matrix $xx^\top \in S_n$ is rank-one, symmetric and positive semidefinite. In particular, also the augmented matrix

$$\begin{pmatrix}1\\x\end{pmatrix}\begin{pmatrix}1\\x\end{pmatrix}^\top = \begin{pmatrix}1 & x^\top\\ x & xx^\top\end{pmatrix} \in S_{n+1}$$

is positive semidefinite. This simple and well-known fact has been used to produce semidefinite reformulations of various quadratic problems. Essentially, the objective function of Problem (3.1) is linearized by adding one new variable x_{ij} for each product of variables $x_i x_j$. With this aim, define the function

$$\begin{aligned}\ell\colon \mathbb{R}^n &\longrightarrow S_{n+1}\\ x &\longmapsto \ell(x) = \begin{pmatrix}1\\x\end{pmatrix}\begin{pmatrix}1\\x\end{pmatrix}^\top\end{aligned}$$

and the matrix $Q \in S_{n+1}$ as

$$Q = \begin{pmatrix}c & \frac{1}{2}\hat{l}^\top\\ \frac{1}{2}\hat{l} & \hat{Q}\end{pmatrix}.$$

The objective function of Problem (3.1) can be written as the inner matrix product

$$x^\top \hat{Q} x + \hat{l}^\top x + \hat{c} = \langle Q, \ell(x)\rangle\,,$$

obtaining a linear function with respect to $\ell(x)$. Now, in order to solve Problem (3.1), it is needed to investigate the image of the feasible set $D_1 \times \cdots \times D_n$ under ℓ

$$\ell(D_1 \times \cdots \times D_n) = \left\{ \begin{pmatrix}1 & x^\top\\ x & xx^\top\end{pmatrix} : x \in D_1 \times \cdots \times D_n \right\}.$$

In fact, Problem (3.1) can be written in an equivalent way as

$$\begin{aligned}\min\quad & \langle Q, X\rangle\\ \text{s.t.}\quad & X \in \operatorname{conv} \ell(D_1 \times \cdots \times D_n).\end{aligned}$$

With this transformation the complexity of the problem has moved from the objective function to the feasible region. The following theorem, taken from [17], yields a characterization of $\ell(D_1 \times \cdots \times D_n)$.

Theorem 3.1. *[17] Let $X \in S_{n+1}$. Then $X \in \ell(D_1 \times \cdots \times D_n)$ if and only if*

(a) $(x_{0i}, x_{ii}) \in P(D_i) := \operatorname{conv}\{(u, u^2) \mid u \in D_i\}$ *for all* $i = 1, \ldots, n$,

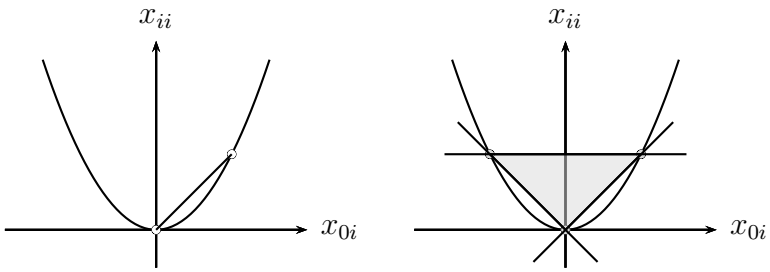
(b) $x_{00} = 1$,

(c) $\operatorname{rank}(X) = 1$, *and*

(d) $X \succeq 0$.

Proof. The first implication is easy to see, $\ell(x)$ satisfies (a)-(d) for any $x \in D_1 \times \cdots \times D_n$, since

$$\ell(x) = \begin{pmatrix}1 & x^\top\\ x & xx^\top\end{pmatrix},$$



(a) $D_i = \{0, 1\}$ (b) $D_i = \{-1, 0, 1\}$

Figure 3.1: The set $P(D_i)$, and its polyhedral description

which is a rank-one positive semidefinite matrix with the first entry equal to one and $(x_{0i}, x_{ii}) = (x_i, x_i^2) \in P(D_i)$ for all $i = 1, \ldots, n$.

For the other direction, let X be such that (a)-(d) are satisfied. We have to prove that there exists $x \in D_1 \times \cdots \times D_n$ for which $X = \ell(x)$. Since $X \succeq 0$, $\operatorname{rank}(X) = 1$ and $x_{00} = 1$, we know that there exists $x \in \mathbb{R}^n$ that satisfies $X = \ell(x)$ and $x_{ii} = x_{0i}^2$. Furthermore, from $(x_{0i}, x_{ii}) \in P(D_i)$ and the strict convexity of $x_{0i} \mapsto x_{0i}^2$, we get that $x_{0i} \in D_i$ and thus $x = (x_{01}, \ldots, x_{0n}) \in D_1 \times \cdots \times D_n$. □

To have a better understanding of $\ell(D_1 \times \cdots \times D_n)$, let us have a closer look at $P(D_i)$. By our assumption, the set D_i is a finite sub-set of $\mathbb{Z}$. In this case, $P(D_i)$ is a polytope in $\mathbb{R}^2$ with $|D_i|$ many extreme points. It has therefore a representation as the set of solutions of a system of $|D_i|$ linear inequalities. Figure 3.1 shows two examples where the set $P(D_i)$ is illustrated for $D_i = \{0, 1\}$ and $D_i = \{-1, 0, 1\}$. In the first case, the set $P(D_i)$ is nothing but the straight line joining the two points $(0, 0)$ and $(1, 1)$. In the second example, $P(D_i) = \operatorname{conv}\{(-1, 1), (0, 0), (1, 1)\}$ or equivalently the solution set of the following inequality system:

$$\begin{aligned} x_{ii} &\leq 1 \\ -x_{ii} + x_{0i} &\leq 0 \\ -x_{ii} - x_{0i} &\leq 0 \end{aligned}$$

This intuitive idea is generalized in the following

Lemma 3.2. *Let $D_i = \{l_i, \ldots, u_i\}$ with $l_i, u_i \in \mathbb{Z}$ and $n_i := |D_i| = u_i - l_i + 1$. Then $P(D_i)$ is completely described by $n_i - 1$ lower bounding facets*

$$-x_{ii} + (j + (j + 1))x_{0i} \leq j(j + 1), \quad j = l_i, l_i + 1, \ldots, u_i - 1,$$

and one upper bounding facet

$$x_{ii} - (l_i + u_i)x_{0i} \leq -l_i u_i.$$

Proof. The lower bounding facets are those linking points (j, j^2) and $(j + 1, (j + 1)^2)$ for $j = l_i, l_i + 1, \ldots, u_i - 1$, while the upper bounding facet links (l_i, l_i^2) and (u_i, u_i^2). □

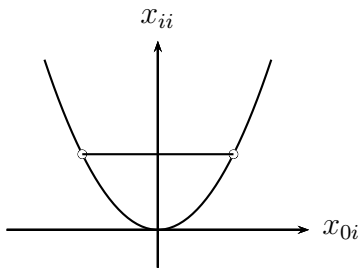


Figure 3.2: For $D_i = \{-1, 1\}$, the set $P(D_i)$ is the line linking the points $(-1, 1)$ and $(1, 1)$.

Notice that in case $|D_i| = 2$, meaning that the variable is binary, there is only one lower bounding facet that together with the upper bounding facet results in a single equation. However, we will not make a case distinction.

In the general case, when $D_i \subset \mathbb{R}$, an infinite number of inequalities may be needed to model the constraint $(x_{0i}, x_{ii}) \in P(D_i)$. However, in [17] an exact separation algorithm has been devised for $P(D_i)$. Since this thesis is focused on integer sets, this algorithm is not described here.

Notice that the characterization of $\ell(D_1 \times \cdots \times D_n)$ contains a non-convex constraint: $\text{rank}(X) = 1$, whereas the remainder of the problem is convex. This means that a convex relaxation of (3.1) is obtained by dropping the rank-one constraint:

$$\begin{aligned} \min \quad & \langle Q, X \rangle \\ \text{s.t.} \quad & (x_{0i}, x_{ii}) \in P(D_i) \quad \forall i = 1, \ldots n \\ & x_{00} = 1 \\ & X \succeq 0. \end{aligned} \tag{3.2}$$

The latter problem is a semidefinite program, since the constraints $(x_{0i}, x_{ii}) \in P(D_i)$ can be replaced by the set of linear constraints of Lemma 3.2, it is in fact a generalization of the well-known semidefinite relaxation for the max-cut problem [34], where $D_i = \{-1, 1\}$, and $P(D_i)$ is simply given by the equation $x_{ii} = 1$, see Figure 3.2. Notice that the set $\{-1, 1\}$ does not fit formally into the type of sets D_i we are considering, however, it is well known that the max-cut problem can be reformulated equivalently with $\{0, 1\}$-variables, see e.g., [5].

3.2 Branch-and-bound algorithm

In this section we describe the Algorithm Q-MIST proposed by Buchheim and Wiegele in [17], it is an SDP-based branch-and-bound algorithm to solve Problem (3.1). It was designed to deal with general closed sub-sets D_i of $\mathbb{R}$, and was implemented to use the SDP solver CSDP [11]. At each node of the branch-and-bound-tree, Q-MIST solves a relaxation of type (3.2), then calls an exact separation algorithm to produce additional cutting planes and solves a new SDP problem whenever new cutting plans have been added.

The choice of the branching variable is motivated by the following

Algorithm Q-MIST: Branch-and-bound algorithm for Problem (3.1)

Input: $\hat{Q} \in S_n, \hat{l} \in \mathbb{R}^n, \hat{c} \in \mathbb{R}, D_i, i = 1, \ldots, n, \epsilon > 0$
Output: $x^* \in \mathbb{R}^n$ s.t. $\hat{f}(x^*)$ differs at most ϵ from optimal value of (3.1)

set $U = \infty$;
let $\mathcal{P}$ be (3.2);
set $\mathcal{S} = \{\mathcal{P}\}$;
while $\mathcal{S} \neq \emptyset$ **do**
    choose $\mathcal{P} \in \mathcal{S}$;
    let $\mathcal{S} = \mathcal{S} \setminus \mathcal{P}$;
    repeat
        solve $\mathcal{P}$ to obtain X;
        for $i = 1$ ***to*** n **do**
            if possible, separate (x_{0i}, x_{ii}) from $P(D_i)$;
            add generated cutting planes to $\mathcal{P}$;
    until *no cutting plane has been generated*;
    round the fractional point $\hat{x} := (x_{01}, \ldots, x_{0n})$ to obtain a feasible solution $\ell(\hat{x})$ of $\mathcal{P}$;
    if $\hat{f}(\hat{x}) < U$ **then**
        let $U = f(\hat{x})$;
        let $x^* = \hat{x}$;
    if $\langle Q, X \rangle < U - \epsilon$ **then**
        find i maximizing $x_{ii} - x_{0i}^2$;
        obtain $\mathcal{P}_1$ from $\mathcal{P}$ by replacing D_i by $D_i \cap (-\infty, x_{0i}]$;
        obtain $\mathcal{P}_2$ from $\mathcal{P}$ by replacing D_i by $D_i \cap [x_{0i}, \infty)$;
        let $\mathcal{S} \cup \{\mathcal{P}_1, \mathcal{P}_2\}$;

Corollary 3.3. *[17] Let $X \in S^+_{n+1}$ be such that $x_{00} = 1$ and $x_{ii} = x^2_{0i}$. Then, $(x_{01}, \ldots, x_{0n})$ is a feasible solution of Problem (3.1), for the appropriate sets D_i.*

Proof. It is known that since $X \in S^+_{n+1}$, the determinant of any principal sub-matrix of X is non-negative (see Proposition 2.2 and Theorem 2.3). Now, consider the determinant of the sub-matrix of X given by the rows and columns 0, i and j: we have that $-(x_{ij} - x_{0i}x_{0j})^2 \geq 0$, it follows that $x_{ij} = x_{0i}x_{0j}$. Since, in addition, $x_{ii} = x^2_{0i}$ for all $i = 1, \ldots, n$, then $X = \ell(x)$, where $x = (x_{01}, \ldots, x_{0n})$. □

From the last corollary it follows that feasibility is guaranteed once $x_{ii} = x^2_{0i}$ for all $i = 1, \ldots, n$. Based on this fact, in Step 18 of Algorithm Q-MIST (see page 29), the variable i where $x_{ii} = x^2_{0i}$ is most violated is chosen as branching variable. In practice, when we are dealing with continuous variables, this condition will never be reached. The following lemma has been proved.

Lemma 3.4. *[17] Let $D_i \subseteq [0, 1]$ for all $i = 1, \ldots, n$ and x^* be an optimal solution of Problem (3.1). Then there exists $\delta > 0$ such that for every optimal solution X of Problem (3.2) with $x_{ii} - x^2_{0i} \leq \delta$ it follows that $\hat{f}(x^*) - \langle Q, X\rangle \leq \epsilon$.*

Moreover, it has been proved that Q-MIST terminates after a finite number of iterations if all D_i are bounded.

Theorem 3.5. *[17] Let D_i be bounded. Then for any $\epsilon > 0$, Algorithm Q-MIST terminates in finite time with a feasible solution x that satisfies $\hat{f}(x) \leq \hat{f}^* + \epsilon$, where $\hat{f}^*$ is the optimal value of Problem (3.1).*

3.3 Matrix formulation

In this section we introduce some notation to express the constraints of Problem (3.2) using matrices and present some properties of Problem (3.1) that will be needed later in Part III.

The relaxation (3.2) contains the constraint $x_{00} = 1$, this fact is exploited to rewrite the polyhedral description of $P(D_i)$ presented in Lemma 3.2 as

$$\begin{aligned}(\beta_{ij} - j(j+1))x_{00} - x_{ii} + (j + (j+1))x_{0i} \leq \beta_{ij}, \quad j = l_i, l_i + 1, \ldots, u_i - 1\\ (\beta_{iu_i} + l_i u_i)x_{00} + x_{ii} - (l_i + u_i)x_{0i} \leq \beta_{iu_i}\end{aligned}$$

for an arbitrary vector $\beta \in \mathbb{R}^m$, with $m = \sum_{i=1}^n n_i$. The introduction of β does not change the primal problem (3.2), but it has a strong impact on the dual problem: the dual feasible set and objective function are both affected by β, as shown below.

The resulting inequalities are written in matrix form in the following way:

$$\langle A_{ij}, X\rangle \leq \beta_{ij} \ .$$

To keep analogy with the facets, for each variable $i \in \{1, \ldots, n\}$, the index ij represents the inequalities corresponding to lower bounding facets $j = l_i, l_i+1, \ldots, u_i-1$ and $j = u_i$ corresponds to the upper bounding facet; see Figure 3.3 for an illustration.

It is clear that, since each constraint links only the variables x_{00}, x_{0i} and x_{ii}, the constraint matrices $A_{ij} \in S_{n+1}$ are very sparse. Indeed, they are zero everywhere

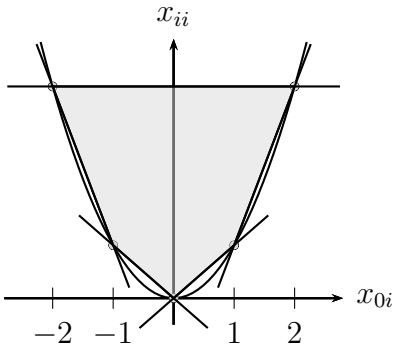


Figure 3.3: The polytope $P(\{-2,-1,0,1,2\})$. Lower facets are indexed, from left to right, by $j = -2,-1,0,1$, the upper facet by 2.

except in the entries 00, $i0$, $0i$ and ii. More precisely, in the case of an upper bound constraint we have

$$\begin{aligned}(A_{ij})_{00} &= \beta_{iu_i} + l_i u_i,\\ (A_{ij})_{0i} &= (A_{ij})_{i0} = -\tfrac{1}{2}(l_i + u_i),\\ (A_{ij})_{ii} &= 1,\end{aligned}$$

while in the case of a lower bound constraint we have

$$\begin{aligned}(A_{ij})_{00} &= \beta_{ij} - j(j+1),\\ (A_{ij})_{0i} &= (A_{ij})_{i0} = j + \tfrac{1}{2},\\ (A_{ij})_{ii} &= -1.\end{aligned}$$

To be consistent, the constraint $x_{00} = 1$ is also written in matrix form as $\langle A_0, X\rangle = 1$, where $A_0 := e_0 e_0^\top \in \mathcal{S}_{n+1}$. In summary, Problem (3.2) can now be written as

$$\begin{aligned}\min \quad & \langle Q, X\rangle \\ \text{s.t.} \quad & \langle A_0, X\rangle = 1 \\ & \langle A_{ij}, X\rangle \le \beta_{ij} \quad \forall j = l_i, \dots, u_i, \quad \forall i = 1, \dots, n \\ & X \succeq 0.\end{aligned} \tag{3.3}$$

The following simple observation is crucial for the algorithm presented in Part III of this thesis.

Lemma 3.6. *All constraint matrices A_{ij} have rank one or two. The rank of A_{ij} is one if and only if*

(a) the facet is upper bounding, i.e., $j = u_i$, and $\beta_{iu_i} = \frac{1}{4}(l_i - u_i)^2$, or

(b) the facet is lower bounding, i.e., $j < u_i$, and $\beta_{ij} = -\frac{1}{4}$.

This property of the constraint matrices will be exploited later when solving the dual problem of (3.3) using a coordinate-wise approach, leading to a computationally cheap update at each iteration and an easy computation of the exact step size.

3.4 Dual problem

In order to derive the dual of Problem (3.3), the linear operator $\mathcal{A}\colon S_{n+1} \longrightarrow \mathbb{R}^{m+1}$ is introduced:

$$\mathcal{A}(X) := \begin{pmatrix} \langle A_0, X\rangle \\ \langle A_{ij}, X\rangle_{j\in\{l_i,\dots,u_i\},i\in\{1,\dots,n\}} \end{pmatrix}.$$

Moreover, a dual variable $y_0 \in \mathbb{R}$ is associated with the constraint $\langle A_0, X\rangle = 1$ and a dual variable $y_{ij} \leq 0$ with the constraint $\langle A_{ij}, X\rangle \leq \beta_{ij}$, for all $j \in \{l_i, \dots, u_i\}$ and $i \in \{1, \dots, n\}$, and $y \in \mathbb{R}^{m+1}$ is defined as

$$y := \begin{pmatrix} y_0 \\ (y_{ij})_{j\in\{l_i,\dots,u_i\},i\in\{1,\dots,n\}} \end{pmatrix}.$$

The adjoint operator to $\mathcal{A}$ is obtained as

$$\mathcal{A}^\top y = y_0 A_0 + \sum_{i=1}^{n}\sum_{j=l_i}^{u_i} y_{ij} A_{ij},$$

and the dual semidefinite program of Problem (3.3) is

$$\begin{aligned} \max \quad & \langle b, y\rangle \\ \text{s.t.} \quad & Q - \mathcal{A}^\top y \succeq 0 \\ & y_0 \in \mathbb{R} \\ & y_{ij} \leq 0 \quad \forall j = l_i, \dots, u_i, \forall i = 1, \dots, n, \end{aligned} \tag{3.4}$$

the vector $b \in \mathbb{R}^{m+1}$ being defined as $b_0 = 1$ and $b_{ij} = \beta_{ij}$ for $i = 1, \dots, n$, $j = l_i, \dots, u_i$.

We conclude this section by emphasizing some characteristics of any feasible solution of Problem (3.3).

Lemma 3.7. *Let X^* be a feasible solution of Problem (3.3). For $i \in \{1, \dots, n\}$, consider the active set*

$$\mathscr{A}_i = \{j \in \{l_i, \dots, u_i\} \mid \langle A_{ij}, X^*\rangle = \beta_{ij}\}$$

corresponding to variable i. Then

(i) for all $i \in \{1, \dots, n\}$, $|\mathscr{A}_i| \leq 2$, and

(ii) if $|\mathscr{A}_i| = 2$, then $x^*_{ii} = (x^*_{0i})^2$ and $x^*_{0i} \in D_i$.

Proof. The polytope $P(D_i)$ is two-dimensional with non-degenerate vertices. Due to the way the inequalities $\langle A_{ij}, X\rangle \leq \beta_{ij}$ are defined it is impossible to have more than two inequalities intersecting at one point; see for example Figures 3.1 and 3.3. Therefore, a given point $(x^*_{ii}, x^*_{0i}) \in P(D_i)$ satisfies zero, one, or two inequalities with equality. In the last case, we have $x^*_{ii} = (x^*_{0i})^2$ by construction, which implies $x^*_{0i} \in D_i$. □

For the dual problem (3.4), Lemma 3.7 (i) means that at most $2n + 1$ out of the $m + 1$ variables can be non-zero in an optimal solution. Such a small number of non-zero variables motivates to consider a coordinate-wise optimization method to solve the dual problem (3.4). Moreover, by Lemma 3.7 (ii), if two dual variables corresponding to the same primal variable are non-zero in an optimal dual solution, then this primal variable will obtain an integer feasible value in the optimal primal solution.

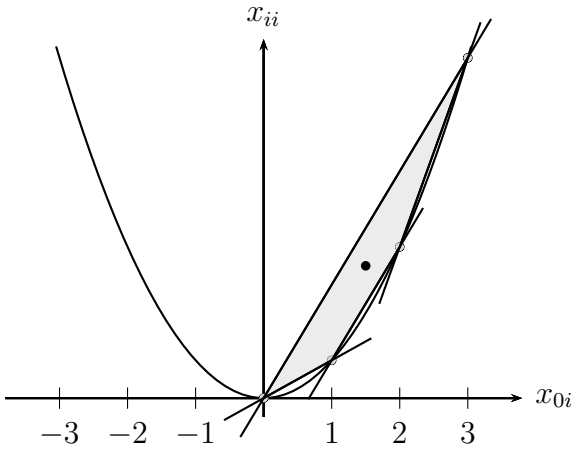


Figure 3.4: The polytope $P(\{0,1,2,3\})$. By definition we have $x_i = \frac{1}{2}(l_i + u_i) = 1.5$. Thus $l_i(x_i) = 3x_i - 2$, defined over the domain $[1,2]$, and $u_i(x_i) = 3x_i$, over $[0,3]$. From the definition of X^0, we have that $x^0_{0i} = 1.5$ and $x^0_{ii} = \frac{1}{2}(l_i(1.5) + u_i(1.5)) = 3.5$. The strictly feasible point (1.5,3.5) is represented by a black dot.

3.5 Primal and dual strict feasibility

We prove here that Problem (3.2) and its dual, Problem (3.4), are strictly feasible. From Theorem 2.8 we can thus conclude that strong duality holds and thus both problems attain their optimal solutions.

We need the definitions of the functions bounding x_{ii} in terms of x_{0i}, which are given by the upper and the lower bounding facets described in Lemma 3.2:

$$x_i \in [j, j+1] \longmapsto l_i(x_i) := (2j+1)x_i - j(j+1) \quad j = l_i, \ldots, u_i - 1$$

and

$$x_i \in [l_i, u_i] \longmapsto u_i(x_i) := (l_i + u_i)x_i - l_i u_i.$$

Notice that in case of binary variables, $l_i(x_i) = u_i(x_i)$.

Theorem 3.8. *Assume in the following that binary variables are modeled by one equation (instead of two inequalities). Problem* (3.2) *is strictly feasible.*

Proof. Define $x_0 := 1$ and $x_i := \frac{1}{2}(l_i + u_i)$ and let

$$x^0_{ij} := \begin{cases} x_i x_j & \text{if } i \neq j \\ \frac{1}{2}(l_i(x_i) + u_i(x_i)) & \text{otherwise.} \end{cases}$$

By the Schur complement, Theorem 2.6, $X^0 \succ 0$ if and only if

$$X^0_{\{1,\ldots,n\},\{1,\ldots,n\}} - X^0_{\{1,\ldots,n\},0} X^0_{0,\{1,\ldots,n\}} \succ 0\ .$$

The latter matrix is a diagonal matrix with entries $\frac{1}{2}(l_i(x_i) + u_i(x_i)) - x_i^2 > 0$. By construction, it is clear that X^0 satisfies all equations (concerning x_{00} and resulting from binary variables) and that it strictly satisfies all inequalities (see Figure 3.4). □

Theorem 3.9. *Problem* (3.4) *is strictly feasible.*

Proof. If $Q \succ 0$, we have that $y^0 = 0$ is a feasible solution of Problem (3.4). Otherwise, define $a \in \mathbb{R}^n$ by $a_i = (A_{iu_i})_{0i}$ for $i = 1, \dots, n$. Moreover, define

$$\tilde{y} := \min\{\lambda_{min}(\hat{Q}) - 1, 0\},$$
$$y_0 := \hat{c} - \tilde{y} \sum_{i=1}^{n} (A_{iu_i})_{00} - 1 - (\tfrac{1}{2}\hat{l} - \tilde{y}a)^\top (\tfrac{1}{2}\hat{l} - \tilde{y}a),$$

and $y^0 \in \mathbb{R}^{m+1}$ as

$$y^0 := \begin{pmatrix} y_0 \\ (y_{ij})_{j \in \{l_i, \dots, u_i\}, i \in \{1, \dots, n\}} \end{pmatrix}, \quad y_{ij} = \begin{cases} \tilde{y}, & j = u_i, i = 1, \dots, n \\ 0, & \text{otherwise.} \end{cases}$$

We have $y^0_{ij} \leq 0$ by construction, so it remains to show that $Q - \mathcal{A}^\top y^0 \succ 0$. To this end, first note that

$$\tilde{c} := \hat{c} - y_0 - \tilde{y} \sum_{i=1}^{n} (A_{iu_i})_{00} = 1 + (\tfrac{1}{2}\hat{l} - \tilde{y}a)^\top (\tfrac{1}{2}\hat{l} - \tilde{y}a) > 0 \,. \tag{3.5}$$

By definition,

$$\begin{aligned} Q - \mathcal{A}^\top y^0 &= Q - y_0 A_0 - \tilde{y} \sum_{i=1}^{n} A_{iu_i} \\ &= Q - y_0 A_0 - \tilde{y} \begin{pmatrix} \sum_{i=1}^{n} (A_{iu_i})_{00} & a^\top \\ a & I_n \end{pmatrix} \\ &= \begin{pmatrix} \tilde{c} & (\frac{1}{2}\hat{l} - \tilde{y}a)^\top \\ \frac{1}{2}\hat{l} - \tilde{y}a & \hat{Q} - \tilde{y} I_n \end{pmatrix}, \end{aligned}$$

which by Schur complement and (3.5) is positive definite if

$$(\hat{Q} - \tilde{y} I_n) - \tfrac{1}{\tilde{c}} (\tfrac{1}{2}\hat{l} - \tilde{y}a)(\tfrac{1}{2}\hat{l} - \tilde{y}a)^\top \succ 0.$$

Denoting $B := (\frac{1}{2}\hat{l} - \tilde{y}a)(\frac{1}{2}\hat{l} - \tilde{y}a)^\top$, we have

$$\lambda_{max}(B) = (\tfrac{1}{2}\hat{l} - \tilde{y}a)^\top (\tfrac{1}{2}\hat{l} - \tilde{y}a) \geq 0$$

and thus

$$\begin{aligned} \lambda_{min}\left((\hat{Q} - \tilde{y} I_n) - \tfrac{1}{\tilde{c}} B \right) &\geq \lambda_{min}(\hat{Q} - \tilde{y} I_n) + \tfrac{1}{\tilde{c}} \lambda_{min}(-B) \\ &= \lambda_{min}(\hat{Q}) - \tilde{y} - \frac{\lambda_{max}(B)}{1 + \lambda_{max}(B)} > 0 \end{aligned}$$

by definition of $\tilde{y}$. We have found y^0 such that $y^0 \leq 0$ and $Q - \mathcal{A}^\top y^0 \succ 0$, we know that there exists $\epsilon > 0$ small enough such that $y^0 - \epsilon \mathbb{1}$ is strictly feasible, i.e., such that $y^0 - \epsilon \mathbb{1} < 0$ and $Q - \mathcal{A}^\top (y^0 - \epsilon \mathbb{1}) \succ 0$

□

We can thus formulate the following corollary, which is a direct consequence of Theorem 2.8.

Corollary 3.10. *Problem* (3.2) *and its dual, Problem* (3.4)*, both admit optimal solutions and there is no duality gap.*

Chapter 4

Coordinate-wise optimization

This chapter is divided into three sections, it begins with a review of the general idea of coordinate descent methods, then we recall the Woodbury formula that is needed in the consecutive section and also later in Part III. The last section describes a specialized primal-barrier coordinate descent algorithm, proposed by Dong [26], designed to solve a semidefinite problem that has a similar structure to Problem (3.4).

4.1 Introduction to coordinate descent methods

Coordinate descent methods are iterative algorithms in which at each iteration one of the coordinates is adjusted in order to optimize the objective function, while the other components are fixed at their values from the current iteration. These simple steps lead to lower-dimensional sub-problems that are thus easier to solve than the full dimensional problem. This method has been used for years in many applications such as image reconstruction [12, 27, 95], machine learning [23, 88, 49, 9], data analysis [4], among others. In general, it is applied to a variety of problems where large or high-dimensional data sets are involved, due to its cheap cost per iteration. In this section we give an overview of the main idea of the algorithm.

We consider the following unconstrained minimization problem

$$\min_{x} \quad f(x) = f(x_1, \ldots, x_n), \tag{4.1}$$

where the function $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$ is continuous. Further assumptions on the structure of f may be made, leading to different variants of the method and convergence properties. The basic coordinate descent framework for the problem above is shown in Algorithm 2. At each iteration a coordinate of x is adjusted by a certain updated scheme while the other coordinates are held fixed.

The coordinates can be selected in a *cyclical* manner, namely, $i_0 = 1$, $i_k = [i_{k-1} \mod n] + 1$, $k \in \mathbb{N}$. One could also select the next iterate *randomly*. If the gradient of f is known, then it is possible to choose the coordinates looking at the gradient, choosing the coordinate corresponding to the component of the gradient with the largest absolute value, this is known as *Gauss-Southwell* rule.

In relation to the update scheme, one possible approach is to compute $x_{i_k}^{(k)}$ by minimizing the objective function with respect to x_{i_k} while the remaining components

Algorithm 2: Coordinate descent algorithm for Problem (4.1)

1 Set $k = 0$ and initialize $x^{(0)} \in \mathbb{R}^n$;
2 **repeat**
3 choose $i_k \in \{1, \ldots, n\}$;
4 update $x_{i_k}^{(k-1)}$ to $x_{i_k}^{(k)}$ by certain scheme;
5 set $x_i^{(k)} = x_i^{(k-1)}$ for $i \in \{1, \ldots, n\} \setminus i_k$;
6 $k \leftarrow k + 1$;
7 **until** *termination condition is satisfied*;

are held fixed:

$$x_{i_k}^{(k)} = \arg\min_{x_{i_k}} f(x_1^{(k-1)}, \ldots, x_{i_k-1}^{(k-1)}, x_{i_k}, x_{i_k+1}^{(k-1)}, \ldots, x_n^{(k-1)}).$$

In general, the update schemes are determined according to the structure of the problem to be solved. If, in particular, the objective function f is differentiable, a coordinate-gradient descent scheme can be applied, namely, by adjusting the coordinate i_k as follows

$$x_{i_k}^{(k)} = x_{i_k}^{(k-1)} - \alpha_k \nabla_{i_k} f(x^{(k-1)}),$$

where α_k is the step size which can be computed by *line search*. This is motivated by the fact that $-\nabla f(x^{(k)})$ is a descent direction.

If α_k is such that the objective function is minimized on the direction e_{i_k}, i.e,

$$f(x^{(k)} + \alpha_k e_{i_k}) = \min_{\alpha > 0} f(x^{(k)} + \alpha e_{i_k}),$$

then such a line search is known as *exact line search*, and α_k is called optimal step size. Notice that once an exact coordinate optimization is performed on the direction i_k, we are guaranteed that in the next iteration $\nabla_{i_{k-1}} f(x^{(k)}) = 0$. If in addition, the next coordinate is chosen using the Gauss-Southwell rule, we cannot have $i_k = i_{k-1}$.

The different coordinate selection rules and step size computation affect the convergence behavior of the coordinate descent algorithm, depending on the type of problem, and they may exploit the specific problem structure. Standard references for literature about convergence properties of these methods are, e.g., [75, 82, 60, 8]. In the approach presented in Part III, we will apply a coordinate-wise optimization method using the Gauss-Southwell rule and exact line search, where the objective function is continuously differentiable, strictly convex and has bounded level sets. Unfortunately we have not found in the literature convergence results of the iterates generated by this rule.

We present here a theorem taken from [69], for cyclical rule, that ensures convergence of Algorithm 2 assuming that the objective function f is continuously differentiable, strictly quasi-convex and has bounded level sets. Recall that if $f \colon \mathbb{R}^n \longrightarrow \mathbb{R}$, any non-empty set of the form

$$\mathcal{L}_z(f) := \{x \in \mathbb{R}^n \mid f(x) \leq z\},$$

for $z \in \mathbb{R}$, is a *(lower) level set* of f associated with z. If f is a convex function, then all its level sets are convex (see, e.g., [8]).

Theorem 4.1. *[69] Assume that $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$ is continuously differentiable, strictly quasi-convex and has bounded level sets. Then for any $x^0 \in \mathbb{R}^n$ the sequence $x^{(k)}$ generated by Algorithm 2 using the cyclical rule and exact line search is well defined and converges to the unique minimizer of f.*

4.2 The Woodbury formula

This section is dedicated to review the Woodbury formula which will be needed later in this chapter and in Part III.

Consider the following linear system of equations

$$(A + UV)x = b,$$

where $A \in M_n$ is non-singular, $U \in M_{n,p}$ and $V \in M_{p,n}$. In order to solve this system, we introduce the variable $y = Vx$ and obtain the following system

$$\begin{aligned} Ax + Uy &= b \\ Vx - Iy &= 0, \end{aligned}$$

which written in matrix form is

$$\begin{pmatrix} A & U \\ V & -I \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}.$$

Notice that $A + UV$ becomes the Schur complement of $-I$ in the matrix

$$\begin{pmatrix} A & U \\ V & -I \end{pmatrix}$$

(see Section 2.1). We have that $x = A^{-1}(b - Uy)$, and substituting this into the equation $y = Vx$, we obtain

$$y = (I + VA^{-1}U)VA^{-1}b.$$

Using this in $x = A^{-1}(b - Uy)$, we get

$$x = (A^{-1} - A^{-1}U(I + VA^{-1}U)VA^{-1})b.$$

Since b was chosen arbitrarily, we can conclude that

$$(A + UV)^{-1} = A^{-1} - A^{-1}U(I + VA^{-1}U)^{-1}VA^{-1},$$

we thus have proved the following theorem.

Theorem 4.2 (Woodbury formula)**.** *Let $A \in M_{n,n}$ be a non-singular matrix, $U \in M_{n,p}$ and $V \in M_{p,n}$. Then the inverse of the matrix $A + UV$, if it exists, is*

$$(A + UV)^{-1} = A^{-1} - A^{-1}U(I + VA^{-1}U)^{-1}VA^{-1}. \tag{4.2}$$

In the special case where $p = 1$, i.e., U is a column vector and V is a row vector, we have the following corollary.

Corollary 4.3 (Sherman-Morrison-Woodbury formula)**.** *Let $A \in M_{n,n}$ and $u, v \in \mathbb{R}^n$. Then*

$$(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1 + v^\top A^{-1}u}. \tag{4.3}$$

Equation (4.2) is known as *Woodbury formula* [40], while (4.3) as *Sherman-Morrison-Woodbury* formula. The Woodbury formula shows how to update the inverse of a matrix altered by the addition of a matrix of small rank. Given a non-singular matrix A, let $B := A + uv^\top$, where $u, v \in \mathbb{R}^n$. The product $uv^\top$ is a matrix of rank one. If the inverse of A is already known, the Woodbury formula is a shortcut to compute the inverse of B. Then, a rank-one change in the matrix A results in a rank-one change in its inverse. In terms of the number of operations needed to compute the inverse of B, it means that it can be computed in $O(n^2)$ operations instead of $O(n^3)$, when it is computed from scratch.

The Woodbury formula can be seen as the generalization to a rank p perturbation of A. In the following we will refer to both formulae as Woodbury formula, and always mention the rank of the perturbation. The complexity of computing the inverse using the general Woodbury formula is $O(pn^2)$ and $O(p^3)$ operations are needed to compute the inverse of $(I + VA^{-1}U)$. The latter is constant for fixed value of p.

4.3 A barrier coordinate minimization approach

In this section we review the main results of a recent paper of Dong [26], which is focused on the generation of convex quadratic relaxations for the same class of problems discussed in Chapter 3:

$$\begin{aligned} \min \quad & x^\top Qx + q^\top x \\ \text{s.t.} \quad & x \in D_1 \times \cdots \times D_n, \end{aligned} \tag{4.4}$$

also here each set D_i is assumed to be a closed and bounded subset of $\mathbb{R}$. Problem (4.4) can be reformulated as the linear program

$$\begin{aligned} \min_{x,v} \quad & v + q^\top x \\ \text{s.t.} \quad & (v, x) \in \mathcal{H}, \end{aligned} \tag{4.5}$$

over the set $\mathcal{H} := \text{conv}\{(v, x) \mid v \geq x^\top Qx, x \in D_1 \times \cdots \times D_n\}$. We begin by studying the convex quadratic valid constraints for $\mathcal{H}$ introduced by Dong in [26], they are basically produced by perturbing the quadratic form $x^\top Qx$ with separable terms. We will see that by adding all possible cutting surfaces of this type, we get a convex relaxation of Problem (4.4) that is equivalent to the semidefinite relaxation of Problem (3.2), described in the previous chapter. Finally, at the end of this section, we concentrate on the method designed to solve the resulting separation problem.

4.3.1 Cutting surfaces from diagonal perturbations

Let us begin by introducing some required notation. We changed a little bit the notation in [26] to keep consistence with the notation introduced of the previous chapter. For each variable i, consider the following set

$$S_i := \{(x, x^2) \mid x \in D_i\}.$$

Since D_i is assumed to be closed and bounded, we can define $L_i := \min\{x \mid x \in D_i\}$ and $R_i := \max\{x \mid x \in D_i\}$. Additionally, for every set S_i, denote by $\ell_i(\cdot)$ its lower convex envelop, i.e., ℓ_i is the largest convex function defined on the interval $[L_i, R_i]$, such that $\ell_i(x) \leq x^2$ for every $x \in D_i$, and by $u_i(\cdot)$ its upper convex envelop, i.e., the smallest concave function on $[L_i, R_i]$, such that $u_i(x) \geq x^2$ for every $x \in D_i$. Observe that in the case of finite sets D_i considered in the last chapter, $l_i(\cdot)$ and $u_i(\cdot)$ are defined exactly as in Section 3.5. In the general case, $D_i \subseteq \mathbb{R}$, as it was mentioned in [17], under some mild conditions, it is possible to fully characterize $l_i(\cdot)$ and $u_i(\cdot)$. See also Algorithm SepP in [17]. In the notation of the last chapter, we have that $P(D_i) = \text{conv}(S_i)$. Even more, the following proposition holds, its proof is clear for finite integer sets, see Figure 3.4.

Proposition 4.4. *[26] Let D_i, S_i, L_i, R_i, $\ell_i(\cdot)$ and $u_i(\cdot)$ be defined as before, then*

(i) $P(D_i) = \text{conv}(S_i) = \{(x_i, y_i) \mid \ell_i(x_i) \leq y_i \leq u_i(x_i)\}$,

(ii) $x_i^2 \leq \ell_i(x_i) \leq u_i(x_i)$ *for every* $x_i \in [L_i, R_i]$.

We can now formulate the following lemma, that defines a set of convex valid constraints for $\mathcal{H}$.

Lemma 4.5. *Let $d \in \mathbb{R}^n$ be such that $Q + \text{diag}(d) \succeq 0$, and $x_i \mapsto y_i(x_i)$ some uni-variate function, such that if $d_i > 0$, $y_i(x_i)$ is concave and $y_i(x_i) \geq x_i^2$; and if $d_i < 0$, $y_i(x_i)$ is convex and $y_i(x_i) \leq x_i^2$, for all $i = \{1, \ldots, n\}$. Then*

$$v \geq x^\top Q x + \sum_{i=1}^{n} d_i(x_i^2 - y_i(x_i)) \tag{4.6}$$

is a valid convex inequality for $\mathcal{H}$.

Proof. From the way d and y_i are defined, we have that $d_i(x_i^2 - y_i(x_i)) \leq 0$ for all i, and therefore that (4.6) is a valid inequality. To prove the convexity, observe that

$$\begin{aligned} x^\top Q x + \sum_{i=1}^{n} d_i(x_i^2 - y_i(x_i)) &= x^\top Q x + \sum_{i=1}^{n} d_i x_i^2 - \sum_{i=1}^{n} d_i y_i(x_i) \\ &= x^\top Q x + x^\top \text{diag}(d) x - \sum_{i=1}^{n} d_i y_i(x_i) \\ &= x^\top (Q + \text{diag}(d)) x - \sum_{i=1}^{n} d_i y_i(x_i). \end{aligned}$$

Since $Q + \text{diag}(d) \succeq 0$, the quadratic form $x^\top (Q + \text{diag}(d))x$ is convex (see, e.g., [8]), the convexity of $-d_i y_i(x_i)$ follows by construction. □

Notice that since it is desirable to have a perturbation as large as possible, the best choices for $y_i(x_i)$ are

$$y_i(x_i) = \begin{cases} \ell_i(x_i), & d_i < 0, \\ u_i(x_i), & d_i > 0, \end{cases}$$

obtaining in this way $d_i(x_i^2 - y_i(x_i))$ as close to zero as possible (without violating convexity). The following inequality is thus a convex valid constraint for $\mathcal{H}$

$$v \geq x^\top Q x + \sum_{i:d_i<0}^{n} d_i(x_i^2 - \ell_i(x_i)) + \sum_{i:d_i>0}^{n} d_i(x_i^2 - u_i(x_i)), \tag{4.7}$$

for any vector d that satisfies $Q+\text{diag}(d) \succeq 0$. Denote by $\mathcal{D}$ the set of all vectors $d \in \mathbb{R}^n$ such that $Q + \text{diag}(d) \succeq 0$. A vector d will be called *admissible* if $d \in \mathcal{D}$. With all the admissible vectors, we obtain a convex relaxation of Problem (4.4)

$$\begin{array}{ll} \min\limits_{x,v} & v + q^\top x \\ \text{s.t.} & v \geq x^\top Q x + \sum\limits_{i:d_i<0}^{n} d_i(x_i^2 - \ell_i(x_i)) + \sum\limits_{i:d_i>0}^{n} d_i(x_i^2 - u_i(x_i)) \quad \forall d \in \mathcal{D} \\ & L_i \leq x_i \leq R_i \quad \forall i = 1, \ldots, n. \end{array} \tag{4.8}$$

Notice that since there is an infinite number of admissible vectors d, the relaxation above is a semi-infinite convex program.

Dong has presented an algorithm to solve Problem (4.8) in an iterative manner, starting with some initial set $\mathcal{D}$ and updating it by adding new violated constraints of type (4.7). We will not explain here the algorithm, instead we will concentrate on the algorithm devised to generate such constraints. But before, we show that Problem (4.8) is in fact equivalent to the semidefinite relaxation of Problem (3.1) described in Chapter 3.

Recall that in [26] the constant part of Problem (3.1) is assumed to be 0. We have that Problem (3.2) can be written as

$$\begin{array}{ll} \min & x^\top Q x + q^\top x \\ \text{s.t.} & (x_i, x_{ii}) \in P(D_i) \quad \forall i = 1, \ldots, n \\ & \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0. \end{array} \tag{4.9}$$

Proposition 4.4 (i), implies that the set of constraints $(x_i, x_{ii}) \in P(D_i)$ is equivalent to

$$\ell_i(x_i) \leq x_{ii} \leq u_i(x_i) \quad \forall i = 1, \ldots, n,$$

therefore, Problem (4.9) is equivalent to

$$\begin{array}{ll} \min & x^\top Q x + q^\top x \\ \text{s.t.} & \ell_i(x_i) \leq x_{ii} \leq u_i(x_i) \quad \forall i = 1, \ldots, n, \\ & \begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0. \end{array} \tag{4.10}$$

Theorem 4.6. *[26] Let μ_{sdp} and μ_{sicp} denote the optimal values for the semidefinite relaxation* (4.9)*, and the semi-infinite convex problem* (4.8)*, respectively. We have $\mu_{sdp} = \mu_{sicp}$.*

Proof. From Proposition 4.4 (ii), we know that $\ell_i(x_i) \leq u_i(x_i)$ implies $L_i \leq x_i \leq R_i$, and from the Schur complement, Theorem 2.6,

$$\begin{pmatrix} 1 & x^\top \\ x & X \end{pmatrix} \succeq 0 \iff X \succeq x x^\top,$$

we have thus that Problem (4.9) is equivalent to

$$\begin{aligned} \min_{x,v} \quad & v + q^\top x \\ \text{s.t.} \quad & L_i \leq x_i \leq R_i \quad \forall i = 1, \ldots, n \\ & v = \min_X \{\langle Q, X\rangle \mid X \succeq xx^\top, \ell_i(x_i) \leq x_{ii} \leq u_i(x_i)\}. \end{aligned}$$

To prove $\mu_{sdp} \geq \mu_{sicp}$, we only need to prove that for any (x, X) in the feasible region of Problem (4.9), and any $d \in \mathcal{D}$, we have

$$\langle Q, X\rangle \geq x^\top Q x + \sum_{i:d_i<0} d_i(x_i^2 - \ell_i(x_i)) + \sum_{i:d_i>0} d_i(x_i^2 - u_i(x_i)).$$

The last inequality is equivalent to

$$\left\langle Q + \operatorname{diag}(d), X - xx^\top \right\rangle \geq x^\top Q x + \sum_{i:d_i<0} d_i(x_{ii} - \ell_i(x_i)) + \sum_{i:d_i>0} d_i(x_{ii} - u_i(x_i)) \geq 0,$$

which is valid since $x_{ii} = x_i^2$, $x_{ii} - \ell_i(x_i) \geq 0$, and $x_{ii} - u_i(x_i) \geq 0$ holds for every i, and $Q + \operatorname{diag}(d) \succeq 0$ implies $\left\langle Q + \operatorname{diag}(d), X - xx^\top \right\rangle \geq 0$.

To prove the other direction we need to compute the dual of Problem (4.10). Let $d_i^+ \geq 0$ be the Lagrange multipliers associated with the constraints $x_{ii} - \ell_i(x_i) \geq 0$ and $d_i^- \geq 0$ be the ones corresponding to $u_i(x_i) - x_{ii} \geq 0$. The dual problem is

$$\max_{d^+, d^- \in \mathbb{R}^n_+} \left\{ \begin{array}{ll} \min\limits_{x,X} & \langle Q, X\rangle + q^\top x - \sum_i d_i^-(x_{ii}^2 - \ell_i(x_i)) - \sum_i d_i^+(u_i(x_i) - x_{ii}) \\ & \text{s.t. } X \succeq xx^\top, \quad L_i \leq x_i \leq R_i \quad \forall x_i \end{array} \right\}.$$

Notice that, unless $\ell_i(\cdot)$ and $u_i(\cdot)$ are linear functions (which would imply $P(D_i)$ has empty interior), one can always find a strictly primal feasible point $(\bar{x}, \bar{X})$ of the last problem. As pointed out in the previous chapter, in the case of binary variables for having strict feasibility, some of the inequality constraints have to be modeled as one equation. Therefore, strong duality holds and the dual optimal value is attained. From Proposition 4.4 (ii), we have that $u_i(x_i) \geq \ell_i(x_i)$ for all $x_i \in [L_i, R_i]$. Without loss of generality, assume $d_i^+ = 0$ or $d_i^- = 0$ but not both, define $d := d_i^+ - d_i^-$. Moreover, assume that $Q + \operatorname{diag}(d) \succeq 0$, otherwise the inner minimization over (x, X) is unbounded from below. Furthermore, assume $X - xx^\top = 0$. With all these assumptions, the dual problem is simplified as

$$\mu_{sdp} = \max_{d: Q + \operatorname{diag}(d) \succeq 0} \left\{ \begin{array}{ll} \min\limits_{x} & x^\top (Q + \operatorname{diag}(d)) x + q^\top x - \sum\limits_{i:d_i<0} d_i \ell_i(x_i) - \sum\limits_{i:d_i>0} d_i u_i(x_i) \\ & \text{s.t.} \quad L_i \leq x_i \leq R_i \quad \forall x_i \end{array} \right\}.$$

Since the dual optimal value is attained, there exists d^* such that $Q + \operatorname{diag}(d^*) \succeq 0$ and

$$\begin{aligned} \mu_{sdp} = \min_x \quad & x^\top (Q + \operatorname{diag}(d^*)) x + q^\top x - \sum_{i:d_i^*<0} d_i^* \ell_i(x_i) - \sum_{i:d_i^*>0} d_i^* u_i(x_i) \\ \text{s.t.} \quad & L_i \leq x_i \leq R_i \quad \forall x_i. \end{aligned}$$

The optimal value of the last problem bounds the value of Problem (4.8) from below, therefore $\mu_{sdp} = \mu_{sicp}$. □

Let $(\bar{x}, \bar{v})$ be a point in the feasible region of Problem (4.5). Then the separation problem can be formulated as the following convex program

$$\begin{array}{ll} \inf & \sum_i g_i(d_i) \\ \text{s.t.} & Q + \operatorname{diag}(d) \succeq 0 \\ & d \in \mathbb{R}^n, \end{array} \tag{4.11}$$

where the function g_i is defined as follows

$$g_i(d_i) := \begin{cases} \alpha_i d_i, & d_i < 0, \\ \gamma_i d_i, & d_i \geq 0, \end{cases}$$

with $\alpha_i := \ell_i(\bar{x}_i) - \bar{x}_i^2$ and $\gamma_i = u_i(\bar{x}_i) - \bar{x}_i^2$.

It is easy to see that $g_i(d_i)$ is a convex function, from Proposition 4.4 we have that $\bar{x}^2 \leq \ell_i(\bar{x}_i) \leq u_i(\bar{x}_i)$, therefore $0 \leq \ell_i(\bar{x}_i) - \bar{x}^2 \leq u_i(\bar{x}_i) - \bar{x}^2$, i.e., $0 \leq \alpha_i \leq \gamma_i$. Observe that Problem (4.11) has a similar structure to Problem (3.4). In the latter problem we are optimizing a linear function subject to a semidefinite constraint $Q - \mathcal{A}^\top y \succeq 0$ that contains matrices of rank two and the dual variables are restricted to be non-positive. We are interested in the algorithm proposed in [26] to solve Problem (4.11), which will be described in the following section. We will see that the main ideas of this algorithm can be extended to solve Problem (3.4).

4.3.2 A barrier coordinate descent algorithm to solve the separation problem

In this section we describe the algorithm presented by Dong [26] to solve Problem (4.11). The author proposed a primal-barrier coordinate minimization algorithm with exact line search. It solves the log-det form of Problem (4.11):

$$\begin{array}{ll} \min\limits_d & f(d;\sigma) := \sum_{i=1}^n g_i(d_i) - \sigma \log\det(Q + \operatorname{diag}(d)) \\ \text{s.t.} & Q + \operatorname{diag}(d) \succ 0, \end{array}$$

where the penalty parameter $\sigma > 0$ is updated iteratively. The optimality condition for this non-differentiable convex problem is

$$0 \in \partial f(d), Q + \operatorname{diag}(d) \succeq 0.$$

The constraint $Q + \operatorname{diag}(d) \succ 0$ cannot be active, since $\{d \in \mathbb{R}^n \mid Q + \operatorname{diag}(d) \succ 0\}$ is an open set. The sub-differential of $f(d;\sigma)$ is

$$\partial f(d;\sigma) = -\operatorname{Diag}((Q + \operatorname{diag}(d))^{-1}) + \oplus \partial g_i(d_i),$$

where

$$\partial g_i(d_i) = \begin{cases} \alpha_i, & d_i < 0, \\ [\alpha_i, \gamma_i], & d_i = 0, \\ \gamma_i, & d_i \geq 0. \end{cases}$$

Let $\bar{d}$ be a feasible vector. At each iteration of the algorithm, one coordinate of the vector $\bar{d}$ will be updated and the inverse matrix $V := (Q + \text{diag}(d))^{-1}$ has to be computed. An important ingredient of the algorithm is the quick update of V, this is done using the Woodbury formula rank one update, see Section 4.2.

The coordinate direction is chosen using the following criterion:

$$i \in \arg\max_j \{|s(\bar{d})|_j\} \tag{4.12}$$

where

$$s(\bar{d}) := \min\{\|u\|_2 \mid u \in \partial f(\bar{d}; \sigma)\}.$$

Then an exact line search is performed along the chosen coordinate, i.e., the step size Δd_i^* along coordinate i satisfies

$$\Delta d_i^* \in \arg\min_{\Delta d_i^*} \{f(\bar{d} + \Delta d_i e_i; \sigma) \mid Q + \text{diag}(\bar{d} + \Delta d_i e_i) \succ 0\}. \tag{4.13}$$

One can derive an explicit formula for Δd_i^*, by solving this problem. The following lemma gives a lower bound on the step size Δd_i. The proof is rather technical, see [26] for complete details.

Lemma 4.7. *[26] Let $\bar{d}$ be a vector such that $Q + \text{diag}(\bar{d}) \succ 0$ and define the matrix $V = (Q + \text{diag}(\bar{d}))^{-1}$. Then for each i, $Q + \text{diag}(\bar{d} + \Delta d_i e_i) \succ 0$ if and only if $\Delta d_i > \frac{1}{v_{ii}}$.*

In order to solve (4.13), we need to compute the sub-differential of the function $f(\bar{d} + \Delta d_i e_i; \sigma)$ at each coordinate i. We obtain

$$\partial f(\bar{d} + \Delta d_i e_i; \sigma) = \partial_i g_i(\bar{d}_i + \Delta d_i) - ((Q + \text{diag}(d) + \Delta d_i e_i e_i^\top)^{-1})_{ii},$$

where

$$\partial_i g_i(\bar{d}_i + \Delta d_i) = \begin{cases} \alpha_i, & \Delta d_i < -\bar{d}_i, \\ [\alpha_i, \gamma_i], & \Delta d_i = -\bar{d}_i, \\ \gamma_i, & \Delta d_i > -\bar{d}_i. \end{cases} \tag{4.14}$$

The inverse of the matrix $(Q + \text{diag}(d) + \Delta d_i e_i e_i^\top)$ is computed with the Woodbury formula (4.3):

$$(Q + \text{diag}(d) + \Delta d_i e_i e_i^\top)^{-1} = (V^{-1} + \Delta d_i e_i e_i^\top)^{-1} = V - \frac{\Delta d_i (Ve_i)(Ve_i)^\top}{1 + \Delta d_i e_i^\top V e_i} = V - \frac{\Delta d_i v_{ii}^2}{1 + \Delta d_i v_{ii}}.$$

Therefore, we have that

$$\sigma\left((Q + \text{diag}(d) + \Delta d_i e_i e_i^\top)^{-1}\right)_{ii} = \sigma\left(V - \frac{\Delta d_i v_{ii}^2}{1 + \Delta d_i v_{ii}}\right)_{ii}. \tag{4.15}$$

Finally, the solution of (4.13) is obtained by intersecting the non-linear curve (4.15) and the piecewise linear curve (4.14) with the constraint $\Delta d_i > -\frac{1}{v_{ii}}$ from Lemma 4.7.

Once the step size is computed, at each iteration the following updates are performed:

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

Algorithm 3: Barrier coordinate descent algorithm for Problem (4.11)

Input: $Q \in S_n$, $\sigma > 0$, $\bar{d} \in \mathbb{R}^n$ s.t. $Q + \text{diag}(d) \succeq 0$
Output: A feasible $\bar{d}$ solving approximately Problem (4.11)

```
Compute V^(0) ← (Q + diag(d̄))^{-1};
for k ← 0 until max-iterations do
    Choose a coordinate direction e_i according to (4.12);
    Update d̄_i ← d̄_i + Δd_i^*, where Δd_i solves (4.13);
    Update V using the Woodbury formula;
    Update σ using some update rule;
    Terminate if some stopping criterion is met;
```

$$V \leftarrow V - \frac{\Delta d_i^* v_i v_i^\top}{1 + \Delta d_i^* v_{ii}},$$

where v_i is the ith column of the previous matrix V. The algorithm proposed by Dong to solve Problem (4.11) is summarized in Algorithm 3. At each step of Algorithm 3, as explained in Section 4.2, the most expensive task is the update of matrix V, which is done in $O(n^2)$.

Part III

The new approach

Chapter 5

A coordinate ascent method

The following chapters contain the main contribution of our research. We will use the elements described in Part II as follows: we propose to solve Problem (3.1) based on the branch-and-bound framework Q-MIST described in Chapter 3 and an extension of the coordinate-wise algorithm of Section 4.3.2.

Our approach tries to exploit the specific structure of Problem (3.1), namely a small total number of (active) constraints and low rank constraint matrices that appear in the semidefinite relaxation. We exploit this special structure by solving the dual problem (3.4) by coordinate-wise optimization methods, in order to obtain fast lower bounds to be used inside the branch-and-bound framework Q-MIST. Our approach is motivated by Algorithm 3 proposed by Dong [26], which was recalled in Section 4.3. As it was already mentioned, Problem (3.1) is reformulated as a convex quadratically constrained problem, then convex relaxations are produced via a cutting surface procedure based on diagonal perturbations. The separation problem turns out to be a semidefinite problem with convex non-smooth objective function, and it is solved by a primal barrier coordinate minimization algorithm with exact line search.

As can be seen, the dual problem (3.4) has a similar structure to the semidefinite problem (4.11), therefore similar ideas can be applied to solve it. Observe that however Problem (3.4) is more general, it contains more general constraints with matrices of rank two (instead of one) and most of our variables are constrained to be non-positive. Another difference is that we deal with an exponentially large number of constraints, out of which only a few are non-zero however. On the other hand, our objective function is linear, instead of scalar which is the case for Problem (4.11).

As a first step, we introduce a penalty term in the objective function of Problem (3.4) to model the semidefinite constraint $Q - \mathcal{A}^\top y \succeq 0$. We obtain

$$\begin{aligned} \max \quad & f(y;\sigma) := \langle b, y\rangle + \sigma \log\det(Q - \mathcal{A}^\top y) \\ \text{s.t.} \quad & Q - \mathcal{A}^\top y \succ 0 \\ & y_0 \in \mathbb{R} \\ & y_{ij} \leq 0 \quad \forall j = l_i, \ldots, u_i, \forall i = 1, \ldots, n \end{aligned} \tag{5.1}$$

for $\sigma > 0$. The penalty term will tend to minus infinity if an eigenvalue of $(Q - \mathcal{A}^\top y)$ tends to zero, in other words, if $(Q - \mathcal{A}^\top y)$ approaches the boundary of the semidefinite cone (see Section 2.4). Therefore the role of the penalty term is to prevent that dual variables will leave the set $\{y \in \mathbb{R}^{m+1} \mid Q - \mathcal{A}^\top y \succ 0\}$. We do not introduce a

penalization to model the non-negativity constraints $y_{ij} \leq 0$. We will see later that we can handle these constraints separately in an efficient way.

Observe that f is strictly concave, indeed it is a sum of a linear function and the log det function, which is a strictly concave function on the positive definite cone (see e.g., [43]). In the next theorem we will prove that the level sets of f are bounded. We recall the definition of upper level sets, in this case for the maximization problem (5.1):

$$\mathcal{L}_f(z) := \{y_0 \in \mathbb{R}, y_{ij} \leq 0 \mid Q - \mathcal{A}^\top y \succ 0, f(y;\sigma) \geq z\}.$$

Theorem 5.1. *For all $\sigma > 0$, all level sets of the objective function of Problem* (5.1) *are bounded.*

Proof. We will need to prove some intermediate steps. We define the following set

$$\mathcal{N} := \{y \in \mathbb{R}^m \mid y_{ij} \leq 0\}.$$

We first prove that for all $y \in \mathcal{N} \setminus \{0\}$ such that $\mathcal{A}^\top y = 0$, it holds that $\langle b, y\rangle \neq 0$. For this, assume that there exists $y \in \mathcal{N}$ such that $\mathcal{A}^\top y = 0$ and $\langle b, y\rangle = 0$. We have thus that there exist $i' \in \{1, \ldots, n\}$ and $j' \in \{l_{i'}, \ldots, u_{i'}\}$ such that

$$A_{i'j'} = \delta_0 A_0 + \sum_{ij \neq i'j'} \delta_{ij} A_{ij} \quad \text{and} \quad b_{i'j'} = \delta_0 b_0 + \sum_{ij \neq i'j'} \delta_{ij} b_{ij}$$

with $\delta_0 \in \mathbb{R}$ and $\delta_{ij} \leq 0$.

By Theorem 3.8, we know that there exists a strictly feasible solution $X^0 \succ 0$ of Problem (3.3), for which

$$\left\langle A_0, X^0 \right\rangle = b_0 \quad \text{and} \quad \left\langle A_{ij}, X^0 \right\rangle < b_{ij} \ \forall ij.$$

Thus

$$b_{i'j'} > \left\langle A_{i'j'}, X^0 \right\rangle = \delta_0 \left\langle A_0, X^0 \right\rangle + \sum_{ij \neq i'j'} \delta_{ij} \left\langle A_{ij}, X^0 \right\rangle \geq \delta_0 b_0 + \sum_{ij \neq i'j'} \delta_{ij} b_{ij} = b_{i'j'},$$

but this is a contradiction.

Secondly, observe that for all $y \in \mathcal{N}$, it holds that

$$\begin{aligned} \left\langle Q, X^0 \right\rangle - \langle y, b\rangle &\geq \left\langle Q, X^0 \right\rangle - \left\langle y, \mathcal{A}(X^0) \right\rangle \\ &= \left\langle Q, X^0 \right\rangle - \left\langle \mathcal{A}^\top y, X^0 \right\rangle \\ &= \left\langle Q - \mathcal{A}^\top y, X^0 \right\rangle \\ &\geq \lambda_{\max}(Q - \mathcal{A}^\top y)\lambda_{\min}(X^0). \end{aligned}$$

The last inequality follows by Lemma 1.2.4 in [43]. We have that $\lambda_{\min}(X^0) > 0$ since $X^0 \succ 0$. Thus

$$\lambda_{\max}(Q - \mathcal{A}^\top y) \leq \frac{1}{\lambda_{\min}(X^0)}(\left\langle Q, X^0 \right\rangle - \langle b, y\rangle). \tag{5.2}$$

Since the level sets $\mathcal{L}_f(z)$ are convex and closed, in order to prove that they are bounded, it is enough to prove that they do not contain an unbounded ray. We will

prove thus that for all feasible solutions $\bar{y}$ of Problem (5.1), and all $y \in \mathcal{N} \setminus \{0\}$ there exists s such that $f(\bar{y} + sy; \sigma) < z$ for all $z \in \mathbb{R}$.

First, consider the case when $\mathcal{A}^\top y = 0$, then

$$f(\bar{y} + sy; \sigma) = \langle b, \bar{y} \rangle + s \langle b, y \rangle + \sigma \log \det(Q)$$

and $\langle b, y \rangle \neq 0$ as argued above. Now, take the limit when $s \to \infty$ of $f(\bar{y} + sy; \sigma)$: if $\langle b, y \rangle > 0$, then $f(\bar{y} + sy; \sigma) \to \infty$, but this contradicts primal feasibility. If, instead $\langle b, y \rangle < 0$, then $f(\bar{y} + sy; \sigma) \to -\infty$.

On the other hand, if $\mathcal{A}^\top y \neq 0$, we have that either $\lambda_{\min}(Q - \mathcal{A}^\top \bar{y} - s^* \mathcal{A}^\top y) = 0$ for some $s^* > 0$, and hence

$$\lim_{s \to s^*} \log \det(Q - \mathcal{A}^\top \bar{y} - s\mathcal{A}^\top y) = -\infty,$$

or

$$\lim_{s \to \infty} \lambda_{\max}(Q - \mathcal{A}^\top \bar{y} - s\mathcal{A}^\top y) = \infty,$$

and from (5.2) it follows that $\langle b, \bar{y} + sy \rangle$ must tend to $-\infty$ when $s \to \infty$. In the second case, observe that $p(s) := \det(Q - \mathcal{A}^\top \bar{y} - s\mathcal{A}^\top y)$ is a polynomial in s, and denote $h(s) := \langle b, \bar{y} + sy \rangle = \langle b, \bar{y} \rangle + \langle b, y \rangle s$. We have that

$$\lim_{s \to \infty} \frac{\log p(s)}{h(s)} = \lim_{s \to \infty} \frac{\frac{p'(s)}{p(s)}}{\langle b, y \rangle} = \lim_{s \to \infty} \frac{p'(s)}{\langle b, y \rangle \, p(s)} = 0.$$

This means that $h(s)$ dominates $\log p(s)$ when $s \to \infty$. Thus $f(\bar{y} + sy) \to -\infty$. □

Recall that from Theorem 4.1 the boundedness of the lower level sets and the strict convexity of the function guarantee the convergence of a coordinate descent method, when using the cyclical rule to select the coordinate direction and exact line search to compute the step length. Therefore in the case of maximizing a strictly concave function with bounded upper level sets a coordinate ascent algorithm with cyclical rule and exact line search will converge to its unique maximizer. Due to the structure of our problem, we consider that applying the Gauss-Southwell rule to choose the coordinate direction, will most likely converge as well. Below we describe a general algorithm to solve Problem (5.1) in a coordinate-wise maximization manner.

Outline of a barrier coordinate ascent algorithm for Problem (3.4)

1: **Starting point:** choose $\sigma > 0$ and any feasible solution y of (3.4).
2: **Direction:** choose a coordinate direction e_{ij}.
3: **Step size:** using exact line search, determine the step length s.
4: **Move along chosen coordinate:** $y \leftarrow y + se_{ij}$.
5: **Decrease** the penalty parameter σ.
6: **Go to (2)**, unless some stopping criterion is satisfied.

In the following sections, we will explain each step of this algorithm in detail. We propose to choose the ascent direction based on a coordinate-gradient scheme, similar to [26]. We thus need to compute the gradient of the objective function of Problem (5.1). See, e.g., [43] for more details on how to compute the gradient. We have that

$$\nabla_y f(y; \sigma) = b - \sigma \mathcal{A}((Q - \mathcal{A}^\top y)^{-1}).$$

For the following, we denote

$$W := (Q - \mathcal{A}^\top y)^{-1},$$

so that

$$\nabla_y f(y;\sigma) = b - \sigma\mathcal{A}(W)\ . \tag{5.3}$$

We will see that, due to the particular structure of the gradient of the objective function, the search of the ascent direction reduces to considering only a few possible candidates among the exponentially many directions. In the chosen direction, we solve a one-dimensional minimization problem to determine the step size. It turns out that this problem has a closed form solution. Each iteration of the algorithm involves the update of the vector of dual variables and the computation of the inverse of an $(n+1)\times(n+1)$-matrix, which only changes by a factor of one constraint matrix when changing the value of the dual variable. We will see later that, thanks to the Woodbury formula and to the fact that our constraint matrices are rank-two matrices, the inverse of the matrix W can be easily computed, the updates at each iteration of the algorithm can be performed in $O(n^2)$ time, which is crucial for the performance of the algorithm proposed. We will see that the special structure of Problem (3.1) can be exploited even more, considering the fact that the constraint matrix associated with the dual variable y_0 has rank-one, and that every linear combination with another linear constraint matrix still has rank at most two. This suggests that we can perform a plane-search rather than a line search, and simultaneously update two dual variables and still recompute the inverse matrix in $O(n^2)$ time. Thus, the main ingredient of our algorithm is the computationally cheap update at each iteration and an easy computation of the optimal step size.

This chapter is organized as follows. We conclude this section, describing the choice of a feasible starting point. In the next two sections, we describe in detail how to choose the coordinate ascent direction and the computation of the step size. A general overview of the coordinate ascent algorithm is given in Section 5.3. Later, in Section 5.4, the general approach is extended by exploiting the fact that y_0 does not have a non-positivity constraint and the properties of A_0. In the last section, we propose an algorithm to compute the primal solution using the dual variables information.

In case of [26], the choice of the starting point was more intuitive. The simple form of the semidefinite constraint allowed to easily decide how far from the boundary of the positive semidefinite cone to start the iterative procedure. In our case, the situation is more complex, so we propose the following choice for a feasible starting point, that seems to perform well in practice.

If $Q \succ 0$, we can safely choose $y^{(0)} = 0$ as starting point. Otherwise, $y^{(0)}$ can be taken as y^0 defined as in the proof of Theorem 3.9. Recall that the computation of y^0 is based on the Schur complement and some properties of positive semidefinite matrices (see Section 2.1). The idea is quite intuitive, it involves however the computation of the smallest eigenvalue of $\hat{Q}$. We will see later that this, together with the computation of the inverse of $Q - \mathcal{A}^\top y^{(0)}$, are the most expensive tasks in our algorithm, in fact it requires $O(n^3)$ time.

5.1 Choice of an ascent direction

We improve the objective function coordinate-wise: at each iteration k of the algorithm, we choose an ascent direction $e_{ij^{(k)}} \in \mathbb{R}^{m+1}$ where $ij^{(k)}$ is a coordinate of the gradient with maximum absolute value

$$ij^{(k)} \in \arg\max_{ij} |\nabla_y f(y;\sigma)_{ij}| \,. \tag{5.4}$$

However, moving a coordinate ij to a positive direction is allowed only if $y_{ij} < 0$, so that the coordinate $ij^{(k)}$ in (5.4) has to be chosen among those satisfying either

$$\nabla_y f(y;\sigma)_{ij} > 0 \quad \text{and} \quad y_{ij} < 0$$

or

$$\nabla_y f(y;\sigma)_{ij} < 0.$$

The entries of the gradient depend on the type of inequality. By (5.3), we have

$$\begin{aligned}\nabla_y f(y;\sigma)_{ij} &= \beta_{ij} - \sigma \langle W, A_{ij} \rangle \\ &= \begin{cases} \beta_{ij} - \sigma((\beta_{ij} - j(j+1))w_{00} + (2j+1)w_{0i} - w_{ii}) & j = l_i, \dots, u_i - 1, \\ \beta_{iu_i} - \sigma((\beta_{iu_i} + l_i u_i)w_{00} - (l_i + u_i)w_{0i} + w_{ii}) & j = u_i. \end{cases}\end{aligned}$$

The number of lower bounding facets for a single primal variable i is $u_i - l_i$, which is not polynomial in the input size from a theoretical point of view. From a practical point of view, a large domain D_i may slow down the coordinate selection if all potential coordinates have to be evaluated explicitly.

However, the regular structure of the gradient entries corresponding to lower bounding facets for variable i allows to limit the search to at most three candidates per variable. To this end, we define the function

$$\begin{aligned}\varphi_i \colon [l_i, u_i - 1] &\longrightarrow \mathbb{R} \\ j &\longmapsto \beta_{ij} - \sigma((\beta_{ij} - j(j+1))w_{00} + (2j+1)w_{0i} - w_{ii}) \,.\end{aligned}$$

Our task is then to find a minimizer of $|\varphi_i|$ over $\{l_i, \dots, u_i - 1\}$. As φ_i is a uni-variate quadratic function, we can restrict our search to at most three candidates, namely the bounds l_i and $u_i - 1$ and the rounded global minimizer of φ_i, if it belongs to $l_i, \dots, u_i - 1$; the latter is

$$\left\lceil \tfrac{w_{0i}}{w_{00}} - \tfrac{1}{2} \right\rfloor \,.$$

In summary, taking into account also the upper bounding facets and the coordinate zero, we need to test at most $1 + 4n$ candidates in order to solve (5.4), independent of the sets D_i.

5.2 Computation of the step size

We compute the step size $s^{(k)}$ by exact line search in the chosen direction. For this we need to solve the following one-dimensional maximization problem

$$s^{(k)} = \arg\max_{s} \{ f(y^{(k)} + s e_{ij^{(k)}}; \sigma) \mid Q - \mathcal{A}^\top (y^{(k)} + s e_{ij^{(k)}}) \succ 0, s \leq -y_{ij^{(k)}} \} \,, \tag{5.5}$$

unless the chosen coordinate is zero, in which case the upper bound on s is dropped. Note that the function

$$s \mapsto f(y^{(k)} + se_{ij^{(k)}}; \sigma)$$

is strictly concave on $\{s \in \mathbb{R} \mid Q - \mathcal{A}^\top(y^{(k)} + se_{ij^{(k)}}) \succ 0\}$. By the first order optimality conditions, we thus need to find an $s^{(k)} \in \mathbb{R}$ satisfying the semidefinite constraint $Q - \mathcal{A}^\top(y^{(k)} + s^{(k)}e_{ij^{(k)}}) \succ 0$ such that either

$$\nabla_s f(y^{(k)} + s^{(k)}e_{ij^{(k)}}; \sigma) = 0 \quad \text{and} \quad y_{ij^{(k)}} + s^{(k)} \leq 0$$

or

$$\nabla_s f(y^{(k)} + s^{(k)}e_{ij^{(k)}}; \sigma) > 0 \quad \text{and} \quad s^{(k)} = -y^{(k)}_{ij^{(k)}}.$$

In order to simplify the notation, we omit the index (k) in the following. From the definition, we have

$$\begin{aligned} f(y + se_{ij}; \sigma) &= \langle b, y\rangle + s\,\langle b, e_{ij}\rangle + \sigma \log\det(Q - \mathcal{A}^\top y - s\mathcal{A}^\top e_{ij}) \\ &= \langle b, y\rangle + \beta_{ij}s + \sigma \log\det(W^{-1} - sA_{ij}). \end{aligned}$$

Then, the gradient is

$$\nabla_s f(y + se_{ij}; \sigma) = \beta_{ij} - \sigma \left\langle A_{ij}, (W^{-1} - sA_{ij})^{-1} \right\rangle. \tag{5.6}$$

The next lemma states that if the coordinate direction is chosen as explained in the previous section, and the gradient (5.6) has at least one root in the right direction of the line search, then there exists always a feasible step length.

Lemma 5.2.

(i) Let the coordinate ij be such that $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$. If there exists $s \geq 0$ for which $\nabla_s f(y + se_{ij};\sigma) = 0$, then for the smallest positive s^+ with $\nabla_s f(y + s^+e_{ij};\sigma) = 0$, one of the following holds:

(a) $y + s^+e_{ij}$ is dual feasible

(b) $s^+ > -y_{ij}$, $y - y_{ij}e_{ij}$ is dual feasible, and $\nabla_s f(y - y_{ij}e_{ij};\sigma) > 0$.

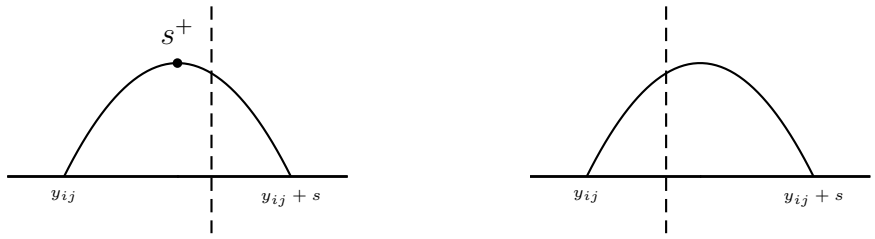
(ii) Let the coordinate ij be such that $\nabla_y f(y;\sigma)_{ij} < 0$. If there exists $s \leq 0$ for which $\nabla_s f(y + se_{ij};\sigma) = 0$, then for the biggest negative s^- with $\nabla_s f(y + s^-e_{ij};\sigma) = 0$ it holds that $y + s^-e_{ij}$ is dual feasible.

Proof. We prove (i), the proof of (ii) follows analogous ideas. Let y be a feasible point of Problem (5.1) and ij such that $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$. Choose the smallest positive s^+ with $\nabla_s f(y + s^+e_{ij};\sigma) = 0$ and assume that (a) is false, we then have to show that (b) holds.

If (a) is false, then $y + s^+e_{ij}$ is not dual feasible, this means that either $Q - \mathcal{A}^\top(y + s^+e_{ij})$ is not positive definite or $s^+ \geq -y_{ij}$.

In the first case, there must exist some $0 < s' \leq s^+$ with $f(s,\sigma) \to -\infty$ for $s \to s'$. From the continuous differentiability of $f(s,\sigma)$ on the feasible region and since $\nabla_y f(y;\sigma)_{ij} > 0$, there exists $0 \leq s'' \leq s'$ with $\nabla_s f(y + s''e_{ij};\sigma) = 0$, in contradiction to the minimality of s^+.

In the second case, by the same reasoning, we may assume that $y + se_{ij}$ is dual feasible for all $0 \leq s \leq s^+$. If there is no $s' \in [0, s^+]$ with $\nabla_s f(y + s'e_{ij};\sigma) = 0$, we must have $\nabla_s f(y + s'e_{ij};\sigma) > 0$ for all $s' \in [0, s^+]$, again by continuous differentiability and $\nabla_y f(y;\sigma)_{ij} > 0$. □



(a) $s^+ \leq s$ (b) $s^+ = -y_{ij}$

Figure 5.1: Illustration of the existence of an optimal step size s^+, Theorem 5.3 (i)

If in addition we exploit that the level sets of the function are bounded, as shown by Theorem 5.1, then we can derive the following theorem:

Theorem 5.3.

(i) Let the coordinate ij be such that $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$. Then the gradient (5.6) *has at least one positive root, and for the smallest positive root s^+, either $y + s^+ e_{ij}$ is dual feasible and $\nabla_s f(y + s^+ e_{ij};\sigma) = 0$, or $y_{ij} + s^+ > 0$ and $\nabla_s f(y - y_{ij} e_{ij};\sigma) > 0$.*

(ii) Let the coordinate ij be such that $\nabla_y f(y;\sigma)_{ij} < 0$. Then the gradient (5.6) *has at least one negative root, and for the biggest negative root s^-, $y + s^- e_{ij}$ is dual feasible and $\nabla_s f(y + s^- e_{ij};\sigma) = 0$.*

Proof. We prove (i), the proof of (ii) follows analogous ideas. Let y a feasible point of Problem (5.1) and ij such that $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$.

From Theorem 5.1, we know that $\mathcal{L}_f(z)$, the level set of f at $z := f(y)$, is bounded. Thus, when moving in the positive direction of the gradient from y to $y + se_{ij}$, at some point either the value of the function f at $y + se_{ij}$ will be equal to $f(y)$, or $y_{ij} + s > 0$.

In the first case, from continuous differentiability of the function $s \mapsto f(y + se_{ij};\sigma)$, and using $\nabla_y f(y;\sigma)_{ij} > 0$, we have that there exists $s^+ \leq -y_{ij}$ such that $\nabla_s f(y + s^+ e_{ij};\sigma) = 0$. By Lemma 5.2, the smallest s^+ is feasible (which also directly follows from $y + s^+ e_{ij} \in \mathcal{L}_f(z)$).

Otherwise, if $y_{ij} + s > 0$, choose $s^+ = -y_{ij}$ and assume that $\nabla_s f(y - y_{ij} e_{ij};\sigma) \leq 0$. This means that there was a point where the gradient changed its direction and thus, from the same arguments as before, there must be $s^* \leq s^+$ such that $\nabla_s f(y + s^* e_{ij};\sigma) = 0$, but this is a contradiction. Therefore the gradient of f at $y + s^+ e_{ij}$ remains non-negative. See Figure 5.1 for an illustration. □

Observe that the computation of the gradient requires to compute the inverse of $W^{-1} - sA_{ij}$, it is worth mentioning that this is the crucial task since it is a matrix of order $n+1$. Notice however that W^{-1} is changed by a rank-one or rank-two matrix sA_{ij}; see Lemma 3.6. Therefore, we will compute the inverse matrix $(W^{-1} - sA_{ij})^{-1}$ using the Woodbury formula for the rank-one or rank-two update, see Section 4.2.

More precisely, each constraint matrix A_{ij} can be factored as follows:

$$A_{ij} = E_{ij} I C_{ij},$$

where $E_{ij} \in M_{n+1,2}$ is defined by $E_{ij} := (e_0\ e_i)$, $e_0, e_i \in \mathbb{R}^{n+1}$, $C_{ij} \in M_{2,n+1}$ is defined by $C := (A_{ij})_{\{0,i\},\{0,\dots,n\}}$, and I is the 2×2-identity matrix. By the Woodbury formula (4.2),

$$(W^{-1} - sA_{ij})^{-1} = (W^{-1} - sE_{ij}IC_{ij})^{-1} = W + WE_{ij}(\tfrac{1}{s}I - C_{ij}WE_{ij})^{-1}C_{ij}W\ . \quad (5.7)$$

Notice that the matrix $\frac{1}{s}I - C_{ij}WE_{ij}$ is a 2×2-matrix, so its inverse can be easily computed even as a closed formula.

On the other hand, from Lemma 3.6, we know under which conditions a constraint matrix A_{ij} has rank-one. In that case, we obtain the following factorization:

$$A_{ij} = (A_{ij})_{ii} v v^\top, \quad (5.8)$$

where $v := (A_{ij})_{0i} e_0 + (A_{ij})_{ii} e_i$. The inverse of $(W^{-1} - sA_{ij})$ is then computed using the Woodbury formula for rank-one update (4.3),

$$(W^{-1} - sA_{ij})^{-1} = (W^{-1} - s(A_{ij})_{ii} v v^\top)^{-1} = W + \frac{(A_{ij})_{ii} s}{1 - (A_{ij})_{ii} s v^\top W v} W v v^\top W. \quad (5.9)$$

Now, we need to find the value of s that makes the gradient in (5.6) zero, this requires to solve the following equation

$$\beta_{ij} - \sigma \left\langle A_{ij}, (W^{-1} - sA_{ij})^{-1} \right\rangle = 0.$$

In order to solve this equation, we distinguish two possible cases, depending on the rank of the constraint matrix of the chosen coordinate. We use the factorizations of the matrix A_{ij} explained above.

Rank-two. By replacing the inverse matrix (5.7) in the gradient (5.6) and setting it to zero, we obtain

$$\beta_{ij} - \sigma \langle A_{ij}, W \rangle - \sigma \left\langle A_{ij}, WE_{ij}(\tfrac{1}{s}I + C_{ij}WE_{ij})^{-1}C_{ij}W \right\rangle = 0.$$

Due to the sparsity of the constraint matrices A_{ij}, the inner matrix product is simplified a lot, in fact we have to compute only the entries 00, $0i$, $0i$ and ii of the matrix product $WE_{ij}(\frac{1}{s}I + C_{ij}WE_{ij})^{-1}C_{ij}W$. We obtain a rational equation on s of degree two, namely

$$\frac{\beta_{ij}\alpha_1 w s^2 + (2\sigma\alpha_1 w - \alpha_2\beta_{ij})s + \beta_{ij} - \sigma\alpha_2}{\alpha_1 w s^2 - \alpha_2 s + 1} = 0,$$

where

$$\begin{aligned} \alpha_1 &:= (A_{ij})_{00}(A_{ij})_{ii} - (A_{ij})_{0i}^2, \\ \alpha_2 &:= (A_{ij})_{00} w_{00} + 2(A_{ij})_{0i} w_{0i} + (A_{ij})_{ii} w_{ii}, \\ w &:= w_{00} w_{ii} - w_{0i}^2. \end{aligned}$$

Theorem 5.3 shows that, since $s \mapsto f(y + se_{ij}; \sigma)$ is continuously differentiable on the level sets, the denominator of the latter equation can not become zero before finding a point where the gradient is zero. Therefore, the step size s is obtained setting the numerator to zero, and using the quadratic formula for the roots of the general quadratic equation:

$$s = \frac{-2\sigma\alpha_1 w + \alpha_2\beta_{ij} \pm \sqrt{((2\sigma\alpha_1 w - \alpha_2\beta_{ij})^2 - 4\beta_{ij}\alpha_1 w(\beta_{ij} - \sigma\alpha_2)}}{2\beta_{ij}\alpha_1 w}.$$

Then, according to Theorem 5.3 we will need to take the smallest/biggest s on the right direction of the chosen coordinate.

Rank-one. In case the rank of A_{ij} is one, the computations can be simplified. We proceed as before, replacing (5.9) in the gradient (5.6) and setting it to zero:

$$\beta_{ij} - \sigma \left\langle (A_{ij})_{ii} vv^\top, W + \frac{(A_{ij})_{ii} s}{1 - (A_{ij})_{ii} sv^\top Wv} Wvv^\top W \right\rangle = 0.$$

Denote $t := \left\langle vv^\top, W \right\rangle = v^\top Wv = v_0^2 w_{00} + 2v_0 v_i w_{0i} + v_i^2 w_{ii}$, then $\left\langle vv^\top, Wvv^\top W \right\rangle = (v^\top Wv)^2 = t^2$. Replacing this in the last equation yields

$$\beta_{ij} - \sigma(A_{ij})_{ii} t - \sigma t^2 \frac{(A_{ij})_{ii}^2 s}{1 - (A_{ij})_{ii} ts} = 0. \tag{5.10}$$

The last expression turns out to be a rational equation linear in s, and the step size is

$$s = \frac{1}{(A_{ij})_{ii} t} - \frac{\sigma}{\beta_{ij}}.$$

Notice that $s \neq \frac{1}{(A_{ij})_{ii} t}$ and hence the denominator in (5.10) is different from zero. We have to point out that the zero coordinate can also be chosen as ascent direction, in that case the gradient is

$$\nabla_s f(y + se_0; \sigma) = 1 - \sigma \left\langle A_0, (W^{-1} - sA_0)^{-1} \right\rangle.$$

As before, the inverse of $W^{-1} - sA_0$ is computed using the Woodbury formula for rank-one update

$$(W^{-1} - sA_0)^{-1} = (W^{-1} - se_0 e_0^\top)^{-1} = W + \frac{s}{1 - sw_{00}} (We_0)(We_0)^\top.$$

The computation of the step size becomes simpler, we just need to find a solution of the linear equation

$$1 - \sigma \left\langle A_0, (W^{-1} - sA_0)^{-1} \right\rangle = 0.$$

Solving the last equation, the step size is

$$s = \frac{1}{w_{00}} - \sigma.$$

A similar formula for the step size is obtained for other cases when the constraint matrix A_{ij} has rank-one and corresponds to an upper facet such that $l_i = -u_i$. Since

in this case $(A_{ij})_{00} = (A_{ij})_{0i} = 0$ and $(A_{ij})_{ii} = 1$, the factorization of A_{ij} in (5.8) reduces to

$$A_{ij} = e_i e_i^\top,$$

and $t = w_{ii}$. Thus, the step is:

$$s = \frac{1}{w_{ii}} - \frac{\sigma}{\beta_{ij}}.$$

With the step size $s^{(k)}$ determined, we use the following formulae for a fast update, again making use of the Woodbury formula:

$$\begin{aligned} y^{(k+1)} &:= y^{(k)} + s^{(k)} e_{ij^{(k)}} \\ W^{(k+1)} &:= W^{(k)} + W^{(k)} E_{ij^{(k)}} \left(\tfrac{1}{s^{(k)}} I - C_{ij^{(k)}} W^{(k)} E_{ij^{(k)}}\right)^{-1} C_{ij^{(k)}} W^{(k)}, \end{aligned}$$

or

$$W^{(k+1)} := W^{(k)} + \frac{(A_{ij})_{ii} s^{(k)}}{1 - (A_{ij})_{ii} s^{(k)}} (W^{(k)} v^{(k)})(W^{(k)} v^{(k)})^\top.$$

5.3 Algorithm overview

Our approach to solve Problem (3.4) is summarized in Algorithm CD.

Algorithm CD: Barrier coordinate ascent algorithm for Problem (3.4)

Input: $Q \in S_{n+1}$
Output: A lower bound on the optimal value of Problem (3.3)

1 Use Lemma 3.9 to compute $y^{(0)}$ such that $Q - \mathcal{A}^\top y^{(0)} \succ 0$
2 Compute $W^{(0)} \leftarrow (Q - \mathcal{A}^\top y^{(0)})^{-1}$
3 **for** $k = 0, 1, 2, \ldots$ **do**
4 Choose a coordinate direction $e_{ij^{(k)}}$ as described in Section 5.1
5 Compute the step size $s^{(k)}$ as described in Section 5.2
6 Update $y^{(k+1)} \leftarrow y^{(k)} + s^{(k)} e_{ij^{(k)}}$
7 Update $W^{(k)}$ using the Woodbury formula
8 Update σ
9 Terminate if some stopping criterion is met
10 **return** $\langle b, y^{(k)} \rangle$

Before entering the main loop, the running time of Algorithm CD is dominated by the computation of the minimum eigenvalue of $\hat{Q}$ needed to compute $y^{(0)}$ and by the computation of the inverse of the matrix $Q - \mathcal{A}^\top y^{(0)}$. Both can be done in $O(n^3)$ time. Each iteration of the algorithm can be performed in $O(n^2)$. Indeed, as discussed in Section 5.1, we need to consider $O(n)$ candidates for the coordinate selection, so that this task can be performed in $O(n)$ time. For calculating the step size and updating the matrix $W^{(k)}$, we need $O(n^2)$ time using the Woodbury formula.

Notice that Algorithm CD produces a feasible solution $y^{(k)}$ of Problem (3.4) at every iteration and hence a valid lower bound $\langle b, y^{(k)} \rangle$ for Problem (3.3). In particular, when used within a branch-and-bound algorithm, this means that Algorithm CD can be stopped as soon as $\langle b, y^{(k)} \rangle$ exceeds a known upper bound for Problem (3.3). Otherwise,

the algorithm can be stopped after a fixed number of iterations or when other criteria show that only a small further improvement of the bound can be expected.

The choice of an appropriate termination rule however is closely related to the update of σ performed in Step 8. The aim is to find a good balance between the convergence for fixed σ and the decrease of σ. This is further discussed in Chapter 7.

5.4 Two dimensional approach

In Algorithm CD, we change only one coordinate in each iteration, as this allows to update the matrix $W^{(k)}$ in $O(n^2)$ time using the Woodbury formula. This was due to the fact that all constraint matrices in the primal SDP (3.3) have rank at most two. However, taking into account the special structure of the constraint matrix A_0, one can observe that every linear combination of any constraint matrix A_{ij} with A_0 still has rank at most two. In other words, we can simultaneously update the dual variables y_0 and y_{ij} and still recompute $W^{(k)}$ in $O(n^2)$ time. Geometrically, we thus search along the plane spanned by the coordinates $(e_0, e_{ij^{(k)}})$ rather than the line spanned by a single coordinate $e_{ij^{(k)}}$. For sake of readability, we again omit the index (k) in the following.

Let ij be a given coordinate and denote by s the step size along coordinate e_{ij} and by s_0 the step size along e_0. At each iteration we then perform an update of the form

$$y \leftarrow y + s_0 e_0 + s e_{ij} \ .$$

The value of the objective function in the new point is

$$f(y + s_0 e_0 + s e_{ij}; \sigma) = \langle b, y \rangle + s_0 + s\beta_{ij} + \sigma \log\det(W^{-1} - s_0 A_0 - s A_{ij}) \ .$$

Our first aim is to obtain a closed formula for the optimal step length s_0 in terms of a fixed step length s. For this, we exploit the fact that the update of coordinate e_0 is rank-one, and that the zero coordinate does not have a sign restriction. Consider the gradient of $f(y + s_0 e_0 + s e_{ij}; \sigma)$ with respect to s_0:

$$\nabla_{s_0} f(y + s_0 e_0 + s e_{ij}; \sigma) = 1 - \sigma \left\langle A_0, (W^{-1} - s_0 A_0 - s A_{ij})^{-1} \right\rangle \ . \tag{5.11}$$

Defining $W(s) := (W^{-1} - s A_{ij})^{-1}$ and using the Woodbury formula for rank-one update, we obtain

$$(W^{-1} - s_0 A_0 - s A_{ij})^{-1} = (W(s)^{-1} - s_0 A_0)^{-1} = W(s) + \frac{s_0(s)}{1 - s_0(s) w_{00}} (W(s)e_0)(W(s)e_0)^\top.$$

Substituting the last expression in the gradient (5.11) and setting the latter to zero, we get

$$s_0(s) := s_0 = \frac{1}{w(s)_{00}} - \sigma.$$

It remains to compute $w(s)_{00}$, which can be done using the Woodbury formula for rank-two updates. In summary, we have shown

Lemma 5.4. *Let s be a given step size along coordinate direction e_{ij}, then*

$$s_0 = \frac{1}{w(s)_{00}} - \sigma \tag{5.12}$$

is the unique maximizer of $f(y + s_0 e_0 + s e_{ij}; \sigma)$, and hence the optimum step size along coordinate e_0.

The next task is to compute a step length s such that $(s_0(s), s)$ is an optimal two-dimensional step in the coordinate plane spanned by (e_0, e_{ij}). To this end, we consider the function

$$g_{ij}(s) := f(y + s_0(s)e_0 + se_{ij}; \sigma)$$

over the set $\{s \in \mathbb{R} \mid Q - \mathcal{A}^\top(y + s_0(s)e_0 + se_{ij}) \succ 0\}$ and solve the problem

$$\max_s \{g_{ij}(s) \mid Q - \mathcal{A}^\top(y^{(k)} + s_0(s)e_0 + se_{ij^{(k)}}) \succ 0, s \leq -y_{ij}^{(k)}\} . \tag{5.13}$$

Since the latter problem is uni-variate and differentiable, we need to find $s \in \mathbb{R}$ such that

$$(g'_{ij}(s) = 0 \text{ and } s \leq -y_{ij}) \quad \text{or} \quad (g'_{ij}(s) > 0 \text{ and } s = -y_{ij}).$$

The derivative of $g_{ij}(s)$ is

$$g'_{ij}(s) = s'_0(s) + \beta_{ij} - \sigma \left\langle s'_0(s)A_0 + A_{ij}, (W^{-1} - s_0(s)A_0 - sA_{ij})^{-1} \right\rangle , \tag{5.14}$$

which is a quadratic rational function. The next lemma shows that at least one of the two roots of $g'_{ij}(s)$ leads to a feasible update if the direction ij is an ascent direction. Similar to Theorem 5.3 in the one dimensional approach, the proof is based on Theorem 5.1.

Theorem 5.5.

(i) Let the coordinate ij be such that $g'_{ij}(0) > 0$ and $y_{ij} < 0$. The expression (5.14) has at least one positive root, and for the smallest positive root s^+, either the point $y + s_0(s^+)e_0 + s^+e_{ij}$ is dual feasible and $g'_{ij}(s^+) = 0$, or $y_{ij} + s^+ > 0$ and $g'_{ij}(-y_{ij}) > 0$.

(ii) Let the coordinate ij be such that $g'_{ij}(0) < 0$. The expression (5.14) has at least one negative root, and for the biggest negative s^-, the point $y + s_0(s^-)e_0 + s^-e_{ij}$ is dual feasible and $g'_{ij}(s^-) = 0$.

This theorem guarantees that if there exists s in the feasible region with $g'_{ij}(s) = 0$, then from continuous differentiability of the function on the level sets, the denominator of $g'_{ij}(s)$, which is $(A_{ij})_{ii}ws - w_{00}^2$ can not be zero. Therefore the roots of $g'_{ij}(s)$ are found by setting the numerator to zero,

$$\begin{aligned}(\alpha - (A_{ij})_{0i}^2(A_{ij})_{ii})w^2s^2 + (2(A_{ij})_{0i}^2 w_{00} - 2(A_{ij})_{ii}\alpha w_{00} - \sigma w)ws \\ + w_{00}\alpha + \sigma(A_{ij})_{ii}w_{00}w + 2(A_{ij})_{0i}w_{00}w_{0i} + (A_{ij})_{ii}w_{0i}^2 = 0,\end{aligned}$$

we obtain the following formula for the step size

$$s = -\frac{1}{\delta}(w\sigma + 2\alpha(A_{ij})_{ii}w_{00} - 2(A_{ij})_{0i}^2 w_{00} \pm \sqrt{\rho}),$$

where

$$\begin{aligned} w &= w_{00}w_{ii} - w_{0i}^2, \\ \alpha &= (A_{ij})_{00} - \beta_{ij}, \end{aligned}$$

$$\rho = \sigma^2 w^2 (A_{ij})_{ii}^4 - 4\alpha (A_{ij})_{ii}^3 w_{0i}^2 - 8(A_{ij})_{0i}(A_{ij})_{ii}^2 w_{0i}(\alpha w_{00} - \frac{1}{2}(A_{ij})_{0i} w_{0i})$$
$$- 4(A_{ij})_{0i}^2 (A_{ij})_{ii} w_{00}(a w_{00} - 2(A_{ij})_{0i} w_{0i}) + 4(A_{ij})_{0i}^4 w_{00}^2$$
$$\delta = 2(A_{ij})_{ii}(\alpha (A_{ij})_{ii} - (A_{ij})_{0i}^2) w.$$

It remains to discuss the choice of the coordinate ij, which is similar to the one-dimensional approach: we choose the coordinate direction e_{ij} such that

$$ij := \arg\max_{ij} |g'_{ij}(0)|, \tag{5.15}$$

where moving into the positive direction of a coordinate e_{ij} is allowed only if $y_{ij} < 0$, thus the candidates are those coordinates satisfying

$$(g'_{ij}(0) > 0 \text{ and } y_{ij} < 0) \quad \text{or} \quad g'_{ij}(0) < 0.$$

Note that

$$g'_{ij}(0) = \begin{cases} j(j+1) - 2\frac{w_{0i}}{w_{00}} j - \frac{w_{0i}}{w_{00}} - (\sigma w_{00} - 1)\frac{w_{0i}^2}{w_{00}^2} + \sigma w_{ii} & j = l_i, \ldots, u_i - 1, \\ l_i u_i + \frac{w_{0i}}{w_{00}}(l_i + u_i) + (\sigma w_{00} - 1)\frac{w_{0i}^2}{w_{00}^2} - \sigma w_{ii} & j = u_i, \end{cases}$$

therefore, as before, we do not need to search over all potential coordinates ij, since the regular structure of $g'_{ij}(0)$ for the lower bounding facets $j \in \{l_i, \ldots, u_i - 1\}$ for each variable i allows us to restrict the search to at most three candidates per variable. Thus only $4n$ potential coordinate directions must be considered.

Using these ideas, a slightly different version of Algorithm CD is obtained by changing Steps 4, 5 and 6 adequately, we call it Algorithm CD2D. In Chapter 7, we compare Algorithm CD and its improved version, Algorithm CD2D, experimentally.

Algorithm CD2D: Two-coordinate maximization algorithm for Problem (3.4)

Input: $Q \in S_{n+1}$
Output: A lower bound on the optimal value of Problem (3.3)
1 Use Lemma 3.9 to compute $y^{(0)}$ such that $Q - \mathcal{A}^\top y^{(0)} \succ 0$
2 Compute $W^{(0)} \leftarrow (Q - \mathcal{A}^\top y^{(0)})^{-1}$
3 **for** $k = 0, 1, 2, \ldots$ **do**
4 Choose a coordinate direction $e_{ij^{(k)}}$ as (5.15)
5 Compute the step sizes $s_0^{(k)}$ and $s^{(k)}$ according to (5.13) and (5.12)
6 Update $y^{(k+1)} \leftarrow y^{(k)} + s_0^{(k)} e_0 + s^{(k)} e_{ij^{(k)}}$
7 Update $W^{(k)}$ using the Woodbury formula
8 Update σ
9 Terminate if some stopping criterion is met
10 **return** $\langle b, y^{(k)} \rangle$

5.5 Primal solutions

This section contains an algorithm to compute an approximate solution of Problem (3.3) using the information given by the dual optimal solution of Problem (3.4).

We will prove that under some additional conditions the approximate primal solution produced is actually the optimal solution.

Let y^* be an optimal solution for the dual problem (3.4), the corresponding primal optimal solution $X^* \in S^+_{n+1}$ must satisfy the complementarity condition

$$(Q - \mathcal{A}^\top y^*)X^* = 0 \tag{5.16}$$

and the primal feasibility conditions $X^* \succeq 0$ and

$$\begin{cases} \langle A_0, X^* \rangle & = 1, \\ \langle A_{ij}, X^* \rangle & = \beta_{ij} \quad \forall i,j \in \mathscr{A}(y^*), \end{cases} \tag{5.17}$$

where $\mathscr{A}(y^*) := \{i,j \mid y_{ij} < 0\}$.

Notice that in order to find a primal optimal solution X^*, we need to solve a semidefinite program, and this is in general computationally too expensive. Since this has to be done at every node of the branch-and-bound tree, we need to devise an alternative method to compute an approximate matrix X that will be used mainly for taking a branching decision in Algorithm Q-MIST. The idea is to ignore the semidefinite constraint $X \succeq 0$. We thus proceed as follows. We consider the spectral decomposition $Q - \mathcal{A}^\top y^* = P\mathrm{diag}(\lambda)P^\top$. Since $Q - \mathcal{A}^\top y^* \succeq 0$, we have $\lambda \geq 0$. Define $Z := P^\top XP$, then $X = PZP^\top$ and (5.16) is equivalent to

$$0 = (P\mathrm{diag}(\lambda)P^\top)(PZP^\top) = P\mathrm{diag}(\lambda)ZP^\top.$$

Since P is a regular matrix, the last equation implies that

$$\mathrm{diag}(\lambda)Z = 0,$$

which is at the same time equivalent to say that $z_{ij} = 0$ whenever $\lambda_i > 0$ or $\lambda_j > 0$. Replacing also $X = PZP^\top$ in (5.17), we have

$$\begin{aligned} 1 = \langle A_0, X \rangle = \left\langle A_0, PZP^\top \right\rangle = \left\langle P^\top A_0 P, Z \right\rangle, \\ \beta_{ij} = \langle A_{ij}, X \rangle = \left\langle A_{ij}, PZP^\top \right\rangle = \left\langle P^\top A_{ij} P, Z \right\rangle. \end{aligned}$$

This suggests, instead of solving the system (5.17) and (5.16) in order to compute X, to solve the above system and then compute $X = PZP^\top$. The system above can be simplified, since Z has a zero row/column for each $\lambda_l > 0$. Thus it is possible to reduce the dimension of the problem as follows: let $\bar{A}$ be the sub-matrix of A where all rows and columns l with $\lambda_l > 0$ are removed; let r be the number of positive entries of λ. Let $Y \in S_{n+1-r}$, we have that the system above is equivalent to

$$\begin{cases} \left\langle \overline{P^\top A_0 P}, Y \right\rangle & = 1 \\ \left\langle \overline{P^\top A_{ij} P}, Y \right\rangle & = \beta_{ij} \quad \forall i,j \in \mathscr{A}(y^*). \end{cases}$$

Then we can extend Y by zeros to obtain $Z \in S_{n+1}$, and finally compute $X = PZP^\top$. We formulate this procedure in Algorithm 4. In the implementation of the algorithm, we will consider always the smallest eigenvalue of $Q - \mathcal{A}^\top y$ as zero, this means that r is at least 1, and there may be more zero eigenvalues considered as zero, depending on the allowed tolerance.

Notice that we are not enforcing explicitly that $Y \succeq 0$, but if Y turns out to be positive semidefinite, then Z is positive semidefinite and therefore X as well. We have the following theorem.

Algorithm 4: Compute approximate solution of (3.3) using dual information

Input: $y^* \in \mathbb{R}^{m+1}$ optimal solution of Problem (3.4)
Output: $X \in S_{n+1}$
1 Compute $P \in M_{n+1}$ orthogonal and $\lambda \geq 0$ such that $Q - \mathcal{A}^\top y^* = P\text{diag}(\lambda)P^\top$
2 Find a solution $Y \in S_{n+1-r}$ of the system of equations (5.18)
3 Set $Z \in S_{n+1}$ as $z_{ij} = 0, \forall ij$, except for $i, j = 1, \ldots, n+1-r$, where $z_{ij} = y_{ij}$
4 Compute $X = PZP^\top$
5 **return** X

Theorem 5.6. *Let y^* be a feasible solution of the dual problem* (3.4) *and $X^* \in S_{n+1}$ the corresponding matrix produced by Algorithm 4. If $X^* \succeq 0$, then (X^*, y^*) are primal-dual optimal solutions of Problems* (3.3) *and* (3.4).

Proof. Let X^* be produced by Algorithm 4 such that it is positive semidefinite. We have that X^* is a feasible solution of Problem (3.3), since it satisfies the set of active constraints for the optimal dual solution y^*:

$$\langle A_0, X\rangle = \left\langle A_0, PZP^\top\right\rangle = \left\langle P^\top A_0 P, Z\right\rangle = \left\langle \overline{P^\top A_0 P}, Y\right\rangle = 1$$

and

$$\langle A_{ij}, X\rangle = \left\langle A_{ij}, PZP^\top\right\rangle = \left\langle P^\top A_{ij} P, Z\right\rangle = \left\langle \overline{P^\top A_{ij} P}, Y\right\rangle = \beta_{ij}$$

for all $ij \in \mathscr{A}(y^*)$, this holds since $Y \in S_{n+1-r}$ is the solution of the system of equations (5.18). It also satisfies complementarity slackness:

$$(Q - \mathcal{A}^\top y^*)X^* = P\text{diag}(\lambda)P^\top PZP^\top = P\text{diag}(\lambda)ZP^\top = 0,$$

where the last equation holds since Z is computed as in Step 3 of Algorithm 4. Namely, if $\lambda_i = 0$, then the corresponding row i of $\text{diag}(\lambda)Z$ is equal to zero. The other rows of $\text{diag}(\lambda)Z$ are equal to zero from the definition of Z. □

Corollary 5.7. *Let y^* be a feasible solution of the dual problem* (3.4). *If the solution of*

$$\begin{aligned}(Q - \mathcal{A}^\top y^*)X &= 0\\ \langle A_0, X\rangle &= 1,\\ \langle A_{ij}, X\rangle &= \beta_{ij} \quad \forall i, j \in \mathscr{A}(y^*),\end{aligned}$$

is unique, then Algorithm 4 produces that solution.

Proof. Let X^* be computed by Algorithm 4. Since Y in step 3 of Algorithm 4 is the solution of (5.18), then X^* also solves (5.17), which will be unique if the system (5.17) has a unique solution. □

In summary, we have proposed a faster approach than solving a semidefinite problem, but without any guarantee that the solution obtained will satisfy the positive semidefiniteness constraint. However there are theoretical reasons to argue that this approach will work in practice. In [2], it was proved that dual non-degeneracy in semidefinite programming implies the existence of a unique optimal primal solution.

Additionally, it was also proved that dual non-degeneracy is a generic property. Putting these two facts together, it means that for random generated instances the probability of having a unique optimal primal solution is one. From the practical point of view, we have implemented Algorithm 4 and run experiments to check the positive semidefiniteness of the matrix X. We will see that for the random instances considered in Chapter 7 this approach seems to work well in practice.

Chapter 6

Adding linear constraints

Several optimization problems like the quadratic knapsack problem [74, 46], among others, can be modeled as a quadratic problem with linear constraints. Linear constraints can be easily included in the current setting of our problem, we will see that doing some minor changes our approach can be extended to solve quadratic problems with additional linear constraints.

Consider the following problem

$$\begin{aligned} \min \quad & x^\top \hat{Q} x + \hat{l}^\top x + \hat{c} \\ \text{s.t.} \quad & a_j^\top x \leq b_j \quad \forall j = 1, \ldots, p \\ & x \in D_1 \times \cdots \times D_n \, . \end{aligned} \tag{6.1}$$

Notice that the set of linear constraints $a_j^\top x \leq b_j$ can be equivalently written as

$$\left\langle A_j, \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^\top \right\rangle \leq \beta_j,$$

where

$$A_j = \begin{pmatrix} \beta_j - b_j & \frac{a_{j0}}{2} & \ldots & \frac{a_{jn-1}}{2} \\ \frac{a_{j0}}{2} & 0 & \ldots & 0 \\ \vdots & & \ddots & \\ \frac{a_{jn-1}}{2} & 0 & \ldots & 0 \end{pmatrix}.$$

Following a similar procedure as the described in Section 3.1, we can formulate a semidefinite relaxation of Problem (6.1) as follows

$$\begin{aligned} \min \quad & \langle Q, X \rangle \\ \text{s.t.} \quad & \langle A_0, X \rangle = 1 \\ & \langle A_{ij}, X \rangle \leq \beta_{ij} \quad \forall j = l_i, \ldots, u_i, i = 1, \ldots, n \\ & \langle A_j, X \rangle \leq \beta_j \quad \forall j = 1, \ldots, p \\ & X \succeq 0. \end{aligned} \tag{6.2}$$

The matrices Q, A_0 and A_{ij} are defined as in Section 3.3. Observe that the new constraint matrices A_j have rank two. We will see that Algorithms CD and CD2D can be extended to solve this more general class of quadratic problem.

We are interested in the dual of Problem (6.2), which can be calculated as

$$\begin{aligned} \max \quad & \langle b, y \rangle \\ \text{s.t.} \quad & Q - \mathcal{A}^\top y \succeq 0 \\ & y_0 \in \mathbb{R} \\ & y_{ij} \leq 0 \quad \forall j = l_i, \ldots, u_i, \forall i = 1, \ldots, n \\ & y_j \leq 0 \quad \forall j = 1, \ldots, p. \end{aligned} \tag{6.3}$$

Here, the adjoint operator $\mathcal{A}^\top$ is defined as

$$\mathcal{A}^\top y = y_0 A_0 + \sum_{i=1}^{n} \sum_{j=l_i}^{u_i} y_{ij} A_{ij} + \sum_{j=1}^{p} y_j A_j,$$

where $y \in \mathbb{R}^{m+p+1}$ is defined as

$$y = \begin{pmatrix} y_0 \\ (y_{ij})_{j \in \{l_i, \ldots, u_i\}, i \in \{1, \ldots, n\}} \\ (y_j)_{j \in \{1, \ldots, p\}} \end{pmatrix},$$

the p additional dual variables are associated with the constraints $\langle A_j, X \rangle \leq \beta_j$, for $j \in \{1, \ldots, p\}$. The vector $b \in \mathbb{R}^{m+p+1}$ is defined as before with p additional entries $(\beta_j)_{j \in \{1, \ldots, p\}}$ corresponding to the right hand sides of the new linear constraints, namely

$$b = \begin{pmatrix} 1 \\ (\beta_{ij})_{j \in \{l_i, \ldots, u_i\}, i \in \{1, \ldots, n\}} \\ (\beta_j)_{j \in \{1, \ldots, p\}} \end{pmatrix}.$$

Again, we want to solve the log-det form of Problem (6.3)

$$\begin{aligned} \max \quad & f(y; \sigma) := \langle b, y \rangle + \sigma \log \det(Q - \mathcal{A}^\top y) \\ \text{s.t.} \quad & Q - \mathcal{A}^\top y \succ 0 \\ & y_0 \in \mathbb{R} \\ & y_{ij} \leq 0 \quad \forall j = l_i, \ldots, u_i, \forall i = 1, \ldots, n \\ & y_j \leq 0 \quad \forall j = 1, \ldots, p. \end{aligned}$$

Notice that the over-all form of the dual problem to be solved has not changed. The new dual variables y_j corresponding to the additional linear constraints play a similar role as the dual variables y_{ij}, both must satisfy the negativity constraint. Even more, the dual problem (6.3) remains strictly feasible, this fact can be derived directly from Theorem 3.9:

Corollary 6.1. *Problem* (6.3) *is strictly feasible.*

Proof. Let $y^0 \in \mathbb{R}^{m+p+1}$ be defined as

$$y^0 := \begin{pmatrix} y_0 \\ (y_{ij})_{j \in \{l_i, \ldots, u_i\}, i \in \{1, \ldots, n\}} \\ (y_j)_{j \in \{1, \ldots, p\}} \end{pmatrix},$$

where y_0 and y_{ij} are defined as in the proof of Theorem 3.9 and y_j is set to zero for all $j = 1, \ldots, p$. We have that $y^0 \leq 0$ and $Q - \mathcal{A}^\top y^0 \succ 0$. Using the same reasoning as in the proof of Theorem 3.9, we know that there exists $\epsilon > 0$ such that $y^0 - \epsilon \mathbb{1} < 0$ and $Q - \mathcal{A}^\top (y^0 - \epsilon \mathbb{1}) \succ 0$. ∎

Due to the addition of linear constraints, primal strict feasibility might no longer be satisfied. However, as it was shown the dual problem is strictly feasible, by Theorem 2.8 this means strong duality holds. Therefore, if the primal problem is not feasible we know thus that the dual problem will be unbounded.

It is clear that the entire procedure described in Section 5.5 is still valid when there are additional linear constraints. Therefore the computation of primal solutions is done by properly adapting Algorithm 4. In the following sections, we detail the two main points where Algorithms CD and CD2D are changed, namely, the choice of the ascent direction and the closed-form formula for the step size.

6.1 Algorithm CD including linear constraints

The addition of p linear constraints in the primal problem implies that for the search of a coordinate direction there are p additional potential directions. As before, the entries of the gradient for the new coordinates can be explicitly computed as

$$\begin{aligned}\nabla_y f(y;\sigma)_j &= \beta_j - \sigma \langle W, A_j \rangle \\ &= \beta_j - \sigma((A_j)_{00} w_{00} + 2\sum_{k=1}^{n} (A_j)_{0k} w_{0k}).\end{aligned}$$

We then choose the coordinate of the gradient with largest absolute value, considering coordinates both corresponding to the lower bounding facets, the upper bounding facet and the new linear constraints. In Section 5.2, we observed that at most $1 + 4n$ candidates have to be considered to select the coordinate direction. Thus, in this case, we will have at most $1 + 4n + p$ candidates.

The computation of the step size follows an analogous procedure as in Section 5.2. This means, a problem similar to Problem (5.5) has to be solved. Therefore, if one of the new possible candidates for coordinate direction $e_j \in \mathbb{R}^{m+p+1}$ for $j \in \{1, \ldots, p\}$ has been chosen, we need to compute s such that either

$$\nabla_s f(y + se_j; \sigma) = 0 \quad \text{and} \quad s \leq -y_j$$

or

$$\nabla_s f(y + se_j; \sigma) > 0 \quad \text{and} \quad s = -y_j.$$

We have that

$$\nabla_s f(y + se_j; \sigma)_j = \beta_j - \sigma \left\langle A_j, (W^{-1} - sA_j)^{-1} \right\rangle. \tag{6.4}$$

The existence of an optimal step size now depends on primal feasibility. There is no guarantee that the level sets of the function are bounded, or as we already mentioned, if the primal problem is not feasible, the dual problem will be unbounded. Testing primal feasibility is a difficult task, however, from Lemma 5.2 we know that if there exists s in the correct direction of the line search that makes the gradient (6.4) zero, then there exists also one on the feasible region. This implies the following result.

Theorem 6.2.

(i) Let the coordinate j be such that $\nabla_y f(y;\sigma)_j > 0$ and $y_j < 0$. If the gradient (6.4) has a positive root, then for the smallest positive root s^+, either $y + s^+ e_j$ is

dual feasible and $\nabla_s f(y + s^+ e_j; \sigma) = 0$, *or* $y_j + s^+ > 0$, $y - y_j e_j$ *is dual feasible, and* $\nabla_s f(y - y_j e_j; \sigma) > 0$. *Otherwise,* $y + s e_{ij}$ *is dual feasible with* $\nabla_s f(y + s e_j; \sigma) > 0$ *for all* $s \in [0, -y_{ij}]$.

(ii) Let the coordinate j *be such that* $\nabla_y f(y; \sigma)_j < 0$. *If the gradient* (6.4) *has a negative root, then for the biggest negative root* s^-, *the point* $y + s^- e_j$ *is dual feasible and* $\nabla_s f(y + s^- e_j; \sigma) = 0$. *Otherwise,* $y + s e_{ij}$ *is dual feasible with* $\nabla_s f(y + s e_j; \sigma) > 0$ *for all* $s \leq 0$.

As before, in order to find the step size, it is necessary to compute the inverse of $W^{-1} - sA_j$. As it was mentioned, the constraint matrices A_j are rank-two matrices. They admit the following factorization

$$A_j = E_j I C_j,$$

where

$$E_j = \begin{pmatrix} \frac{1}{2}(A_j)_{00} & 1 \\ (A_j)_{01} & 0 \\ \vdots & \vdots \\ (A_j)_{0n} & 0 \end{pmatrix} \quad \text{and} \quad C_j = \begin{pmatrix} 1 & 0 & \dots & 0 \\ \frac{1}{2}(A_j)_{00} & (A_j)_{01} & \dots & (A_j)_{0n} \end{pmatrix}.$$

With the Woodbury formula and the factorization above, we have that the inner product of A_j and $(W^{-1} - sA_j)^{-1}$ reduces to the inner product of two 2×2 matrices:

$$\begin{aligned} \left\langle A_j, (W^{-1} - sA_j)^{-1} \right\rangle &= \left\langle E_j I C_j, W + W E_j (\tfrac{1}{s} I - C_j W E_j)^{-1} C_j W \right\rangle \\ &= \left\langle I, E_j^\top W C_j^\top + E_j^\top W E_j (\tfrac{1}{s} I - C_j W E_j)^{-1} C_j W C_j^\top \right\rangle. \end{aligned}$$

We obtain

$$\begin{aligned} E_j^\top W E_j &= \begin{pmatrix} d & f \\ f & w_{00} \end{pmatrix}, \quad C_j W C_j^\top = \begin{pmatrix} w_{00} & f \\ f & d \end{pmatrix}, \\ C_j W E_j &= \begin{pmatrix} f & w_{00} \\ d & f \end{pmatrix}, \quad E_j^\top W C_j^\top = \begin{pmatrix} f & d \\ w_{00} & f \end{pmatrix}, \end{aligned}$$

where

$$\begin{aligned} d &= \tfrac{1}{4} w_{00} (A_j)_{00}^2 + (A_j)_{00} \sum_{i=1}^n w_{0i} (A_j)_{0i} + \sum_{i=1}^n \sum_{k=1}^n w_{ik} (A_j)_{0i} (A_j)_{0k} \\ &= \left\langle W, (A_j)_{0\cdot} (A_j)_{0\cdot}^\top \right\rangle, \\ f &= \tfrac{1}{2} w_{00} (A_j)_{00} + \sum_{i=1}^n w_{0i} (A_j)_{0i} \\ &= W_{0\cdot}^\top (A_j)_{0\cdot}. \end{aligned}$$

Replacing the inner product in the gradient (6.4), we obtain a rational function of degree two

$$\nabla_s f(y + s e_j; \sigma)_j = \frac{\beta_j (d w_{00} - f^2) s^2 + (2 d \sigma w_{00} - 2 f^2 \sigma + 2 \beta_j f) s + 2 f \sigma - \beta_j}{(d w_{00} - f^2) s^2 + 2 f s - 1}.$$

Finally the step size is obtained setting the numerator to zero, yielding

$$s = \frac{-d\sigma w_{00} + f^2\sigma - \beta_j f \pm \sqrt{d^2\sigma^2 w_{00}^2 - 2df^2\sigma^2 w_{00} + f^4\sigma^2 + \beta_j^2 d w_{00}}}{\beta_j(dw_{00} - f^2)}.$$

In the implementation of the algorithm, if no root of the gradient (6.4) is found in the right direction, the step size has to be set to $-y_{ij}$ when the coordinate j is such that $\nabla_y f(y;\sigma)_j > 0$ and $y_j < 0$, or $s = M$, where $M \ll 0$, when the coordinate j is such that $\nabla_y f(y;\sigma)_j < 0$.

It is clear that Algorithm CD can be easily extended to compute lower bounds for the optimal value of Problem 6.2. In the next section, we describe the steps that have to be changed in Algorithm CD2D. We will see that in this case, due to the structure of the constraint matrices A_j, Algorithm CD2D has some advantages over Algorithm CD.

6.2 Algorithm CD2D including linear constraints

A two-dimensional update is also possible for solving the dual of Problem (6.2), again in this case, any linear combination of a constraint matrix A_j with A_0 remains being a rank-two matrix. The optimal two-dimensional step size $(s_0(s), s)$ along the coordinate plane spanned by (e_0, e_j) can be computed following an analogous procedure to the one explained in Section 5.4. It turns out, in this case, that the computation of the step size is technically less complicated. Lemma 5.4 can be used to compute the step size $s_0(s)$ along the direction e_0, in terms of a given step size s along coordinate direction e_j. Recall that $W(s) = (W^{-1} - sA_j)^{-1}$, and thus

$$s_0(s) = \frac{1}{w(s)_{00}} - \sigma = -\frac{1}{w_{00}}((dw_{00} - f^2)s^2 + 2fs + \sigma w_{00} - 1),$$

with f, d defined as in the last section. We can define then the function

$$g_j(s) := f(y + s_0(s)e_0 + se_j; \sigma)$$

over the set $\{s \in \mathbb{R} \mid Q - \mathcal{A}^\top(y + s_0(s)e_0 + se_j) \succ 0\}$. We have to solve a similar problem to (5.13), namely, we need to find $s \in \mathbb{R}$ such that

$$(g_j'(s) = 0 \text{ and } s \leq -y_j) \quad \text{or} \quad (g_j'(s) > 0 \text{ and } s = -y_j).$$

We thus need to compute the derivative of $g_j(s)$

$$g_j'(s) = s_0'(s) + \beta_j - \sigma \left\langle s_0'(s)A_0 + A_j, (W^{-1} - s_0(s)A_0 - sA_j)^{-1} \right\rangle. \tag{6.5}$$

As we already pointed out, the existence of a step size is related with primal feasibility. We have the following theorem that, analogous to Theorem 6.2, is a direct consequence of Lemma 5.2.

Theorem 6.3.

(i) Let the coordinate j be such that $g_j'(0) > 0$ and $y_j < 0$. If the derivative (6.5) has a positive root, then for the smallest positive root s^+, either $y + s_0(s^+)e_0 + s^+e_j$ is dual feasible and $g_j'(s^+) = 0$, or $y_j + s^+ > 0$, $y + s_0(-y_j)e_0 - y_j e_j$ is dual feasible and $g_j'(-y_j) > 0$. Otherwise, $y + s_0(s)e_0 + se_j$ is dual feasible with $g_j'(s) > 0$ for all $s \in [0, -y_{ij}]$.

(ii) Let the coordinate j be such that $g_j'(0) < 0$. If the derivative (6.5) has a negative root, then for the biggest negative s^-, the point $y + s_0(s^-)e_0 + s^- e_j$ is dual feasible and $g_j'(s^-) = 0$. Otherwise, $y + s_0(s)e_0 + se_j$ is dual feasible with with $g_j'(s) > 0$ for all $s \leq 0$.

In order to compute the inner product in (6.5), we propose the following factorizations for the matrices $\bar{A}_j := s_0'(s)A_0 + A_j$ and $\tilde{A}_j := s_0(s)A_0 + sA_j$:

$$\bar{A}_j = \bar{E}_j I \bar{C}_j, \text{ and } \tilde{A}_j = \tilde{E}_j I \tilde{C}_j,$$

where

$$\bar{E}_j = \begin{pmatrix} \frac{1}{2}(s_0'(s) + (A_j)_{00}) & 1 \\ (A_j)_{01} & 0 \\ \vdots & \vdots \\ (A_j)_{0n} & 0 \end{pmatrix}, \quad \bar{C}_j = \begin{pmatrix} 1 & 0 & \dots & 0 \\ \frac{1}{2}(s_0'(s) + (A_j)_{00}) & (A_j)_{01} & \dots & (A_j)_{0n} \end{pmatrix},$$

$$\tilde{E}_j = \begin{pmatrix} \frac{1}{2}(s_0(s) + s(A_j)_{00}) & 1 \\ s(A_j)_{01} & 0 \\ \vdots & \vdots \\ s(A_j)_{0n} & 0 \end{pmatrix}, \quad \tilde{C}_j = \begin{pmatrix} 1 & 0 & \dots & 0 \\ \frac{1}{2}(s_0(s) + s(A_j)_{00}) & s(A_j)_{01} & \dots & s(A_j)_{0n} \end{pmatrix}.$$

In this way, the inner product of matrices in (6.5) can be rewritten as the inner product of two 2×2 matrices:

$$\begin{aligned} \left\langle \bar{A}_j, (W^{-1} - \tilde{A}_j)^{-1} \right\rangle &= \left\langle \bar{E}_j I \bar{C}_j, W + W\tilde{E}_j(I - \tilde{C}_j W \tilde{E}_j)\tilde{C}_j W \right\rangle \\ &= \left\langle I, \bar{E}_j^\top W \bar{C}_j^\top + \bar{E}_j^\top W \tilde{E}_j (I - \tilde{C}_j W \tilde{E}_j)\tilde{C}_j W \bar{C}_j^\top \right\rangle, \end{aligned}$$

where

$$\bar{E}_j^\top W \tilde{E}_j = \begin{pmatrix} d_1 & \bar{f} \\ \tilde{f} & w_{00} \end{pmatrix}, \quad \tilde{C}_j W \bar{C}_j^\top = \begin{pmatrix} w_{00} & \bar{f} \\ \tilde{f} & d_1 \end{pmatrix},$$
$$\tilde{C}_j W \tilde{E}_j = \begin{pmatrix} \tilde{f} & w_{00} \\ \tilde{d} & \tilde{f} \end{pmatrix}, \quad \bar{E}_j^\top W \bar{C}_j^\top = \begin{pmatrix} \bar{f} & \bar{d} \\ w_{00} & \bar{f} \end{pmatrix},$$

and

$$\begin{aligned} \bar{d} &= \left\langle W, (\bar{A}_j)_{0\cdot}(\bar{A}_j)_{0\cdot}^\top \right\rangle, \\ \tilde{d} &= \left\langle W, (\tilde{A}_j)_{0\cdot}(\tilde{A}_j)_{0\cdot}^\top \right\rangle, \\ \bar{f} &= W_{0\cdot}^\top (\bar{A}_j)_{0\cdot}, \\ \tilde{f} &= W_{0\cdot}^\top (\tilde{A}_j)_{0\cdot}, \\ d_1 &= \left\langle W, (\tilde{A}_j)_{0\cdot}(\bar{A}_j)_{0\cdot}^\top \right\rangle. \end{aligned}$$

By doing all calculations, one can verify that $\langle A_j, (W^{-1} - sA_j)^{-1} \rangle$ is actually zero. Replacing this into (6.5) we get $g_j'(s) = s_0'(s) + \beta_j$, where

$$s_0'(s) = -\frac{2}{w_{00}}((dw_{00} - f^2)s + f),$$

and setting $g'_j(s)$ to zero, we obtain a linear equation on the step size s, whose root is

$$s = \frac{2f - \beta_j w_{00}}{2(f^2 - dw_{00})}. \tag{6.6}$$

Observe that the step size s is independent on the value of σ, however the step s_0 is still dependent. From Theorem 6.3 it follows that:

(i) if the coordinate j is such that $g'_j(0) > 0$ and $y_j < 0$, and if the derivative (6.5) has a positive root, then the step size (6.6) must be positive. When there is no positive root s can be set to $-y_{ij}$.

(ii) if the coordinate j is such that $g'_j(0) < 0$, and if the derivative (6.5) has a negative root, then the step size (6.6) must be negative. When there is no negative root set $s = M$, with $M \ll 0$.

The coordinate selection will be done in a similar way as in Section 5.4, i.e., we will choose the coordinate with the largest absolute value of $g'_j(0)$. Recall that from Section 5.4, we have $4n$ potential coordinates, after adding p linear constraints we will have that $4n + p$ candidates to be considered.

Chapter 7

Experiments

In Chapter 5 we have presented a coordinate-wise optimization approach that aims to improve the performance of Q-MIST, the branch-and-bound algorithm described in Section 3.2 and presented in [17]. As it was already mentioned, Q-MIST was implemented to use the CSDP library [11] for solving SDP-relaxations of type (3.2) at each node of the branch-and-bound tree. On the one side, the main motivation to consider a fast coordinate ascent method was to obtain quick and good lower bounds for the objective value of the quadratic integer problem (3.1). On the other side, our motivation was also to find approaches that allow us to solve larger size instances, since this is one of the main restrictions that SDP solvers, based on interior point methods, have.

We evaluate the performance of Algorithms CD and CD2D and compare their performance to CSDP, and to other non-linear solvers as COUENNE [7] and BARON [86]. We divided the experiments into two parts. In the first part, we concentrate on random instances of Problem (3.1). We are interested in understanding two main points: the behavior of the lower bounds at the root node and the influence of the stopping criterion on the branch-and-bound tree and running times. Finally, we evaluate the extension of our approach when linear constraints are added to the original problem. First of all, we begin by describing in the next section the general settings for all experiments.

7.1 General setup

Our experiments were carried out on Intel Xeon processors running at 2.60 GHz. For all the algorithms, the optimality tolerance OPTEPS was set to 10^{-6}. We have used as a base the code that already exists for Q-MIST. Algorithms CD and CD2D were implemented in C++, using routines from the LAPACK package [3] only in the initial phase for computing a starting point. Namely, to compute the smallest eigenvalue of $\hat{Q}$ needed to determine $y^{(0)}$, and the inverse matrix $W^{(0)} = (Q - \mathcal{A}^\top y^{(0)})^{-1}$. The updates in each iteration can be realized by elementary calculation, as explained in Chapter 5.

Recall that in Section 3.3, the parameter β_{ij} can be chosen arbitrarily. As it was pointed out, this parameter does not change the feasible region of the primal problem (3.3), however it does have an influence on its dual problem. We have tested several choices of β_{ij}, like setting it to zero for all the constraints, or according to Lemma 3.6, so that all constraint matrices have rank one. We have found out experimentally that when choosing the value of the parameter β_{ij} in such way that the constraint matrices A_{ij} have their first entry equal to zero, our approach has faster convergence. Hence,

we set $\beta_{iu_i} = -l_i u_i$ for the upper bounding facets and $\beta_{ij} = j(j+1)$ for lower bounding facets, with $j = l_i, \ldots, u_i$, see Section 3.3.

For our experiments, we have generated random instances in the same way as proposed in [17]: the objective matrix is $\hat{Q} = \sum_{i=1}^{n} \mu_i v_i v_i^\top$, where the n numbers μ_i are chosen as follows: for a given value of $p \in [0, 100]$, the first $pn/100$ μ_i's are generated uniformly from $[-1, 0]$ and the remaining ones from $[0, 1]$. Additionally, we generate n vectors of dimension n, with entries uniformly at random from $[-1, 1]$, and orthonormalize them to obtain the vectors v_i. The parameter p represents the percentage of negative eigenvalues, so that $\hat{Q}$ is positive semidefinite for $p = 0$, negative semidefinite for $p = 100$ and indefinite for any other value $p \in (0, 100)$. The entries of the vector $\hat{l}$ are generated uniformly at random from $[-1, 1]$, and $\hat{c} = 0$.

In the following we will consider mainly two types of variable domains. Whenever we refer to *ternary* instances, we have in mind sets D_i of the form $\{-1, 0, 1\}$, and with *integer* instances we mean instances with $D_i = \{-10, \ldots, 10\}$. In our implementation, we use the following rule to update the penalty parameter: whenever the entry of the gradient corresponding to the chosen coordinate has an absolute value below 0.1 in the case of ternary instances or below 0.001 for integer instances, we multiply σ by 0.25. As soon as σ falls below 10^{-8}, we fix it to this value. The initial σ is set to 1.

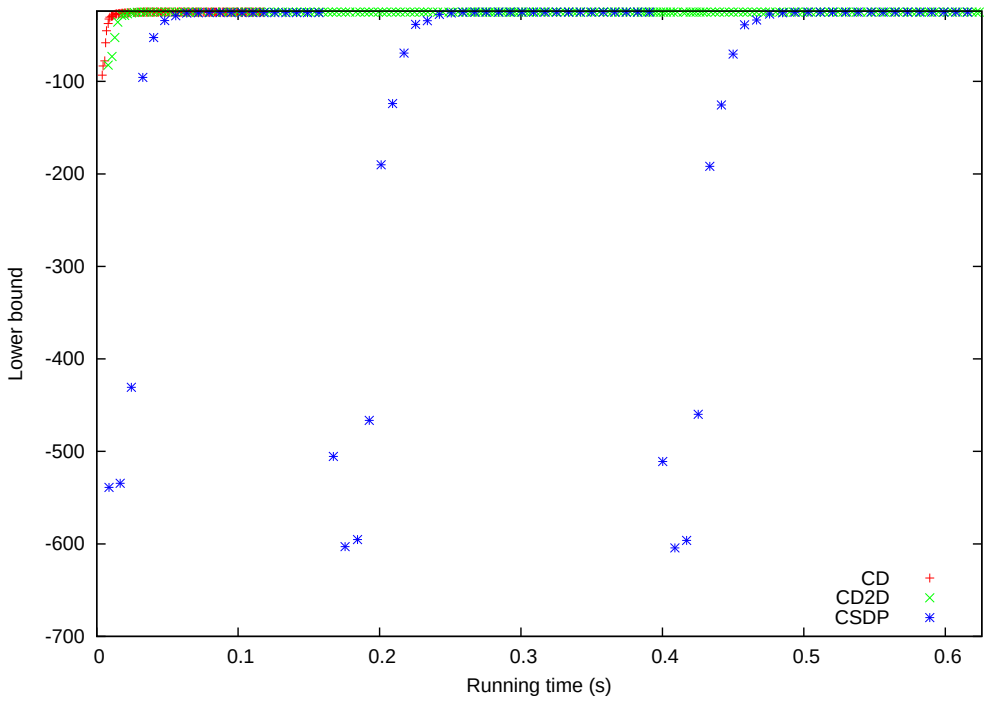
7.2 Root node behavior

We first evaluate the performance of both Algorithms CD and CD2D in the root node of the branch-and-bound tree and compare them with CSDP, the SDP solver used in [17]. We are interested in the improvement of the lower bound over time. In Figure 7.1 and 7.2 we plotted the lower bounds obtained by CSDP and the algorithms CD and CD2D in the root node, for a random instance with all sets D_i ternary and integer, respectively. We have chosen a random instance of size $n = 100$ and two values of p: 0 and 100.

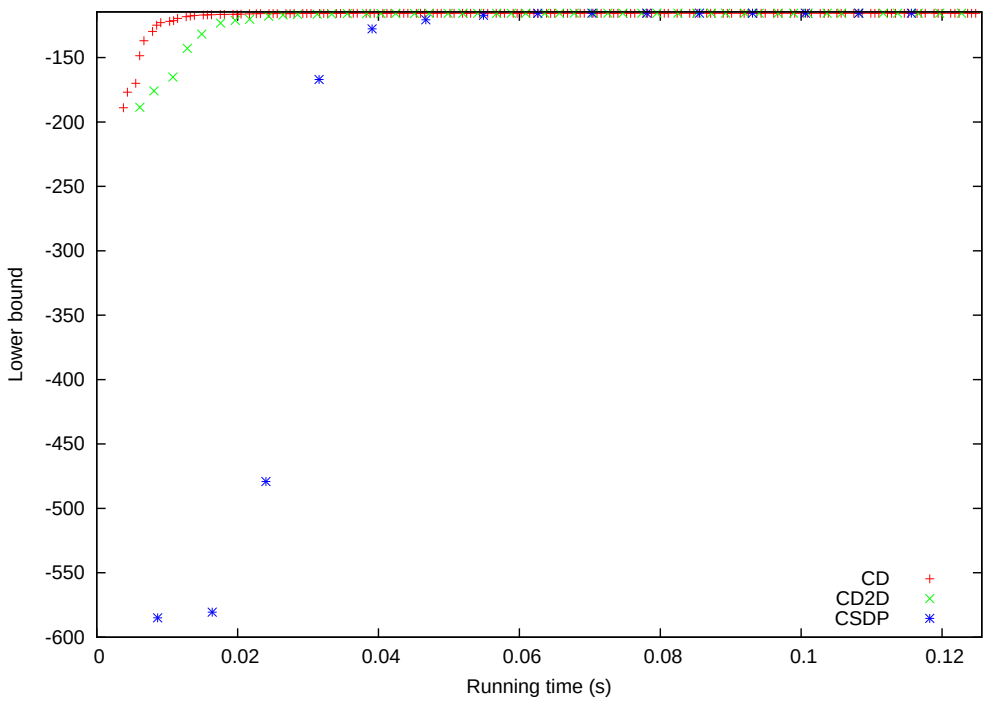
From Figure 7.1 and Figure 7.2, we see that both Algorithms CD and CD2D clearly dominate CSDP. Observe that for CSDP, the computation of the root bound for $p = 0$ involves one and two re-optimizations due to separation (see pictures shown in (a)). For this reason, the lower bound given by CSDP has to restart with a very weak value. In this particular instance, for both values of p, Algorithm CD is stronger than CD2D in case of ternary instances, see Figure 7.1. From the pictures in Figure 7.2, we can see that CD2D dominates the bounds of CD. This is true in particular for $p = 100$.

7.3 Primal solution

At the root node, we have also performed the evaluation of Algorithm 4, designed to compute an approximate primal solution of Problem (3.3) using the dual feasible solution y^* of Problem (3.4) (see the details in Section 5.5). Recall that we need to compute the eigenvalue decomposition of the matrix $Q - \mathcal{A}^\top y^*$, and set a tolerance to decide which other eigenvalues will be considered as zero. In the experiments we have taken into account that $Q - \mathcal{A}^\top y^*$ has always at least one zero eigenvalue, and considered as zero all the eigenvalues smaller equal than 0.01. We have run experiments to check the positive semidefiniteness of the matrix X^* at the root node of the branch-

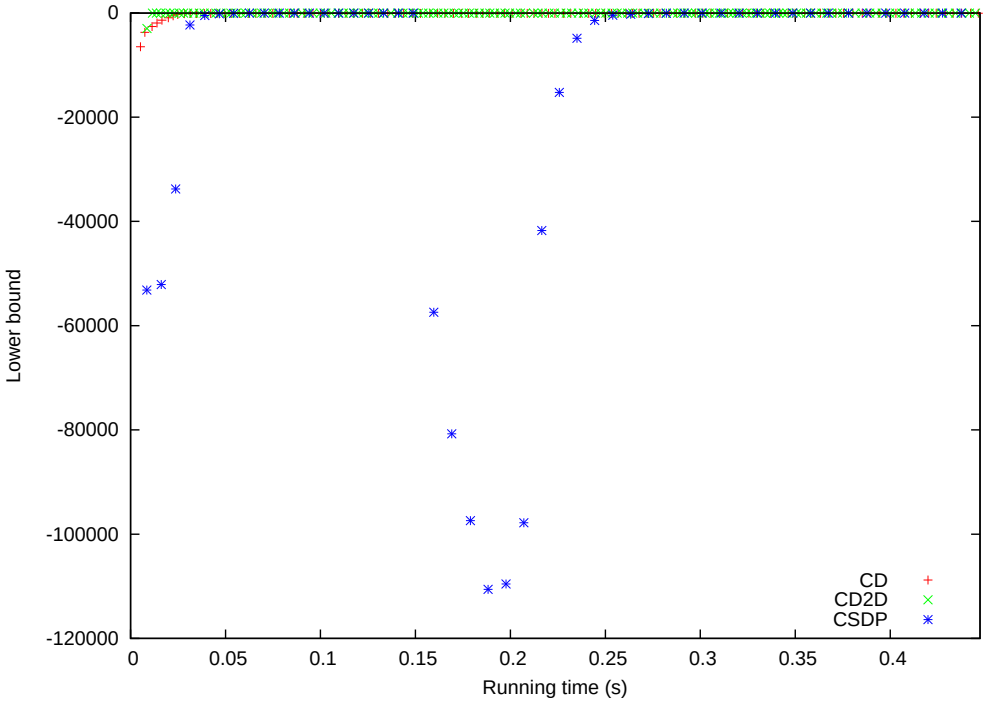


(a) Ternary instance, $p = 0$

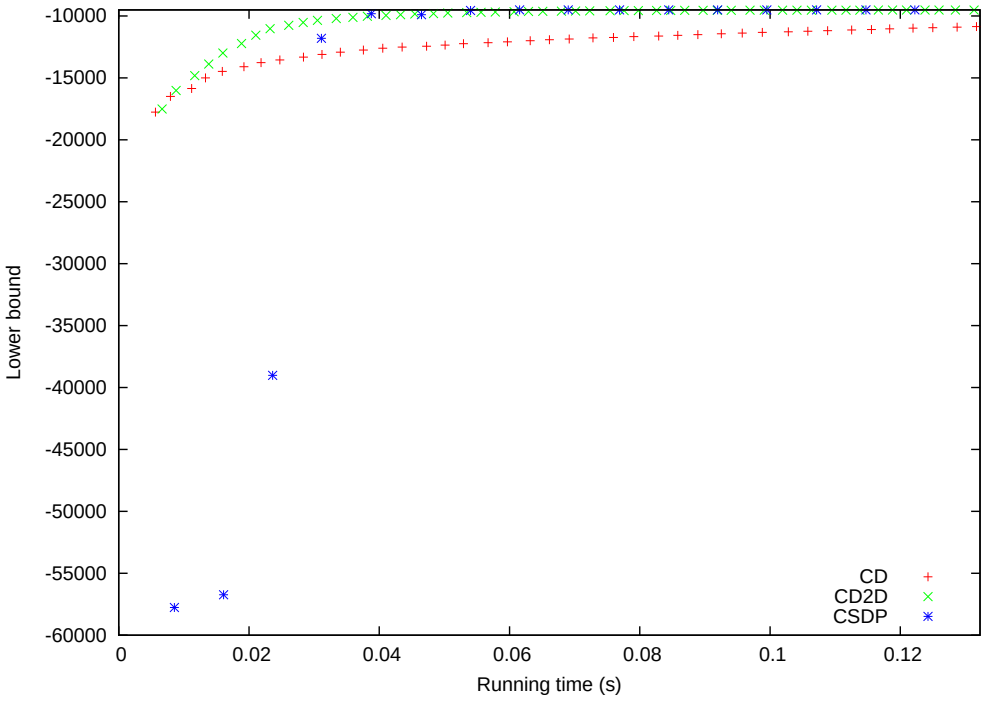


(b) Ternary instance, $p = 100$

Figure 7.1: Comparison of the lower bounds in the root node obtained by Q-MIST with CD, CD2D and CSDP, for a random ternary instance with two different values of p.



(a) Integer instance, $p = 0$



(b) Integer instance, $p = 100$

Figure 7.2: Comparison of the lower bounds in the root node obtained by Q-MIST with CD, CD2D and CSDP, for a random integer instance with two different values of p.

and-bound tree, with the dual variables obtained from Algorithms CD and CD2D. We did this test for all instances used in the experiments of the next sections. We have observed that in all the cases the smallest eigenvalue of X is always greater than -10^{-14}. Based on this fact we can conclude that the method works.

7.4 Stopping criterion

We next investigate the impact of our approach when used within the branch-and-bound scheme Q-MIST. For this it is important to find a good stopping criterion that either may allow an early pruning of the nodes or stops the algorithm when no further improvement of lower bounds is expected. Our approach has the advantage of producing feasible solutions of Problem (3.4) and thus a valid lower bound for Problem (3.3), at every iteration. This means that we can stop the iteration process and prune the node as soon as the current lower bound exceeds a known upper bound for Problem (3.3).

We propose the following stopping criterion. Every n iterations, we compare the gap at the current point (*new-gap*) with the previous one n iterations before (*old-gap*). If

$$(1 - \text{GAP})\ \textit{old-gap} < \textit{new-gap}$$

and the number of iterations is at least $|D_i|n$, or

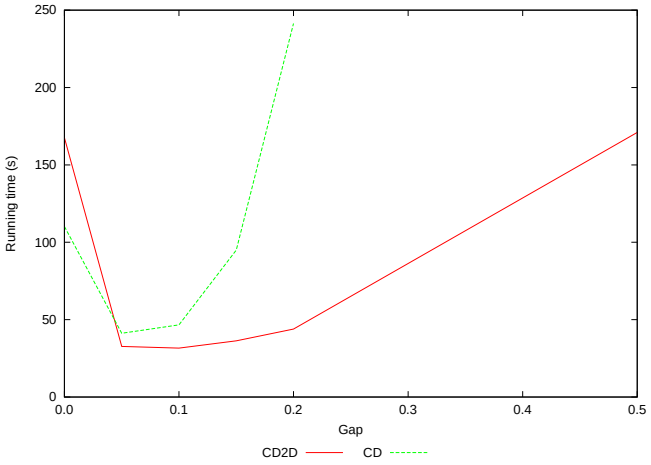
$$\textit{new-gap} < \text{OPTEPS}$$

we stop the algorithm. The gap is defined as the difference of the best upper bound known so far and the current lower bound. The value of GAP has to be taken in $[0, 1]$.

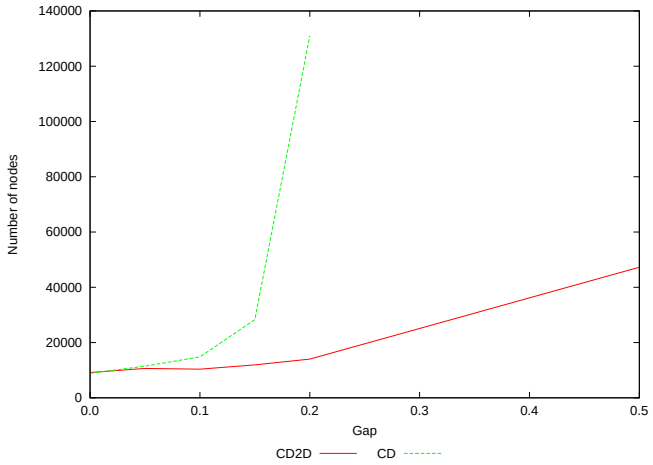
In Figure 7.3 we illustrate the influence of the parameter GAP on the running time and number of nodes needed in the entire branch-and-bound tree, for both Algorithm CD and CD2D. We have chosen 110 random ternary instances of size 50, 10 instances for each $p \in \{0, 10, \ldots, 100\}$. The x-axis corresponds to different values of GAP, while the y-axis to the average running time (Figure 7.3 (a)) and the average number of nodes (Figure 7.3 (b)), taken over the 110 instances. If GAP=0, then the algorithm will stop only when the new-gap reaches the absolute optimality tolerance. As expected, strong bounds are obtained, and thus the number of nodes is reduced and the time per node increases. When GAP = 1, the algorithm will stop immediately after $|D_i|n$ iterations, the lower bound produced may be too weak and therefore the number of nodes is large. A similar behavior of GAP is repeated for integer instances. We conclude that taking GAP=0.1 produces in the algorithm a good balance between the quality of the lower bounds and the number of nodes. We use the same stopping rules for both Algorithm CD and CD2D.

7.5 Total running time

We generated random instances for two types of domains: ternary $D_i = \{-1, 0, 1\}$ and integer $D_i = \{-10, \ldots, 10\}$. The matrix $\hat{Q}$, and the vector $\hat{l}$ defining the objective function $\hat{f}$ in Problem (3.1) are generated as explained above. Similar to [17], we generated 10 random instances for each $p \in \{0, 10, \ldots, 100\}$ and each size n, i.e., we



(a) Running time



(b) Number of nodes

Figure 7.3: Influence of the gap criterion on the number of nodes and the running time for ternary instances, the behavior for integer instances is similar.

generated 110 different instances for each n. On the one hand, we are interested in evaluating the performance of the branch-and-bound framework Q-MIST using the new Algorithms CD and CD2D, and compare them to CSDP. On the other hand, we compare to other non-convex integer programming software: COUENNE [7] and BARON [86].

In the following tables, n in the first column represents the number of variables. For each approach, we report the the number of solved instances (#), the average number of nodes explored in the branch-and-bound (*nodes*) and the average running time in seconds (*time*). All lines report average results over 110 random instances. We have set a time limit of one hour, and compute the averages considering only the instances solved to proven optimality within this period of time.

In Table 7.1 we present the results for ternary instances. As it can be seen, Q-MIST with all three approaches manages to solve all 110 instances for $n \leq 50$. Observe however that both Algorithms CD and CD2D require less time than CSDP even if the number of nodes enumerated is much larger. For $n > 50$, Q-MIST with the new approach solves much more instances than CSDP. Note that BARON and COUENNE solved all 110 instances only for $n \leq 20$ and $n \leq 30$, respectively.

Table 7.2 reports the results for integer instances, the results show that Algorithm CD2D outperforms all the other approaches. In this case, the lower bounds of Algorithm CD are too weak, leading to an excessive number of nodes and it is not able to solve all instances even of size 10 within the time limit. On contrary, Algorithm CD2D manages to solve much more instances than its competitors, also in this case of integer instances.

From the experiments reported in [17], it was already known that CSDP outperforms a previous version of COUENNE. The comparison of Q-MIST with BARON is new. We have used also ANTIGONE [65] for the comparison, but we do not report the results observed since they are not better than those obtained with COUENNE.

Summarizing we can state that Algorithm CD2D yields a significant improvement of the algorithm Q-MIST when compared with CSDP, and it is even capable to compete with other commercial and free software as BARON and COUENNE.

It is important to point out that the performance of BARON is almost not changed when considering ternary or integer variable domains, it solves more or less the same number of instances in both cases. On contrary, it is clear that the change of the domains affected the performance in our approach, specially in Algorithm CD. For the experiments in the next section, we will consider only Algorithm CD2D.

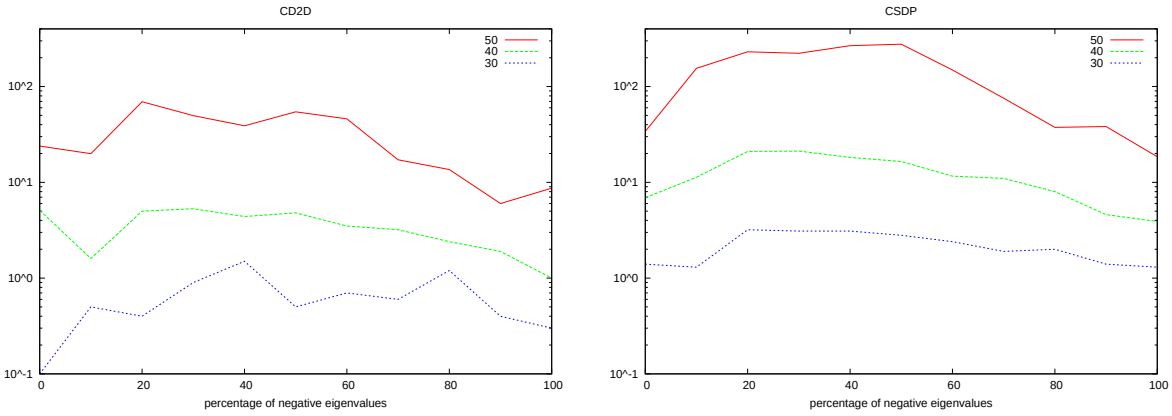
To conclude this section, we present pictures to illustrate how the percentage of negative eigenvalues influences the running time of Algorithm CD2D. Recall that, from the way we are producing the random instances, with the parameter p we can control the percentage of negative eigenvalues of the matrix $\hat{Q}$ in the objective function. We plotted in Figure 7.4 the average running time for 10 random instances for each value of p in $\{0, 10, \ldots, 100\}$. We consider ternary and integer instances. Each line represents a different number of variables. For this experiment we have set the time limit to 5400s. We chose only the dimensions for which all 110 instances were solved within the time limit. One can see that ternary instances of small size do not show any clear behavior. However, in general, one can say that both algorithms CSDP and CD2D keep the same tendency: instances with convex ($p = 0$) and concave ($p = 100$) objective function require less running time, while instances with indefinite $\hat{Q}$ are harder to solve.

Table 7.1: Results for ternary instances, $D_i = \{-1, 0, 1\}$

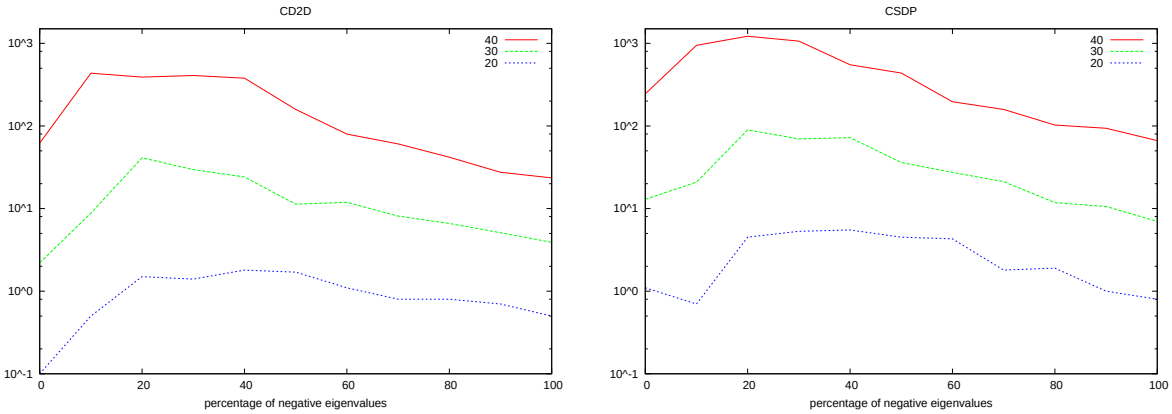
n	Q-MIST CD			Q-MIST CD2D			Q-MIST CSDP			COUENNE			BARON		
	#	nodes	time	#	nodes	time	#	nodes	time	#	nodes	time	#	nodes	time
10	110	49.31	0.03	110	28.05	0.02	110	10.11	0.07	110	11.91	0.10	110	1.42	0.07
20	110	250.31	0.16	110	174.24	0.06	110	67.95	0.32	110	2522.35	10.40	110	8.87	0.80
30	110	1531.29	1.25	110	668.47	0.65	110	247.24	2.17	85	150894.54	1225.72	110	8.67	27.59
40	110	3024.42	4.98	110	2342.75	3.47	110	1030.25	12.20	4	134864.75	2330.83	65	45.88	280.17
50	110	14847.49	46.61	110	10357.11	31.62	110	7284.09	136.81	0	–	–	21	29.14	222.93
60	107	34353.45	197.60	110	33780.15	155.84	109	17210.14	526.96	0	–	–	12	10.67	219.77
70	83	76774.30	515.98	98	94294.82	656.58	71	17754.41	887.17	0	–	–	3	2.33	257.51
80	63	98962.24	1151.22	65	126549.25	1150.02	34	19553.47	1542.38	0	–	–	0	–	–

Table 7.2: Results for integer instances, $D_i = \{-10, \ldots, 10\}$

n	Q-MIST CD			Q-MIST CD2D			Q-MIST CSDP			COUENNE			BARON		
	#	nodes	time	#	nodes	time	#	nodes	time	#	nodes	time	#	nodes	time
10	107	1085009.52	105.54	110	70.58	0.07	109	26.29	0.16	110	5817.25	7.51	110	45.43	0.49
20	10	296203.60	154.30	110	969.11	0.99	110	324.71	2.85	98	91473.86	489.05	109	140.43	6.44
30	4	179909.00	336.25	110	5653.71	13.89	110	2196.87	34.49	0	–	–	104	137.47	38.20
40	0	–	–	110	38458.96	187.76	108	13029.41	386.68	0	–	–	59	202.93	255.65
50	0	–	–	96	99205.07	944.79	67	24292.79	1247.10	0	–	–	15	17.87	279.82
60	0	–	–	53	84802.25	1329.92	26	30105.15	2088.00	0	–	–	8	11.25	282.82
70	0	–	–	2	48648.00	1218.50	1	2011.00	254.00	0	–	–	7	12.43	457.47



(a) Ternary $D_i = \{-1, 0, 1\}$



(b) Integer $D_i = \{-10, \ldots, 10\}$

Figure 7.4: Influence of the percentage of negative eigenvalues on the average running time.

7.6 Behavior with linear constraints

In Chapter 6 we have described how our approach can be extended when inequality constraints are added to Problem (3.1). For the experiments in this section we will consider ternary instances and two types of inequality constraints: $\sum_{i=1}^{n} x_i \leq 0$ and $a^\top x \leq b$, i.e., we solve two quadratic problems:

$$\begin{aligned} \min \quad & x^\top Q x + l^\top x + c \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i \leq 0 \\ & x \in \{-1, 0, 1\}^n, \end{aligned}$$

and

$$\begin{aligned} \min \quad & x^\top Q x + l^\top x + c \\ \text{s.t.} \quad & a^\top x \leq b \\ & x \in \{-1, 0, 1\}^n. \end{aligned}$$

The vector $a \in \mathbb{R}^n$ and the right hand side of $a^\top x \leq b$ are generated as follows: each entry a_i is chosen randomly distributed in $\{1, 2, \ldots, 5\}$ and b is randomly distributed in $\{1, \ldots, \sum_{i=1}^{n} a_i\}$. The objective function is generated as explained before. Tables 7.3 and 7.4 report the results of the performance of Algorithm Q-MIST with CD2D and CSDP, and compare with BARON. The dimension n of the problem is chosen from 10 to 50 and $p \in \{0, 10, \ldots, 100\}$, as before each line in the tables corresponds to the average computed over 110 instances solved within the time limit, 10 instances for each combination of n and p.

Comparing the results reported in Table 7.1 with those of Tables 7.3 and 7.4, one could say that the addition of a linear constraint does not change the over all behavior of our approach. As it can be seen, Q-MIST (with both approaches CD2D and CSDP) outperforms BARON. However Algorithm CD2D, as it was shown in Table 7.1, is much faster even if the number of nodes explored is larger.

Table 7.3: Results for ternary instances plus $\sum_{i}^{n} x_i \leq 0$

n	Q-MIST						BARON		
	CD2D			CSDP					
	#	nodes	time	#	nodes	time	#	nodes	time
10	110	35.85	0.01	110	12.73	0.02	110	1.29	0.09
20	110	195.56	0.35	110	74.18	0.34	110	6.70	1.10
30	110	993.21	1.08	110	332.38	2.65	110	17.31	43.86
40	110	3160.16	4.85	110	1199.55	16.47	48	13.44	233.40
50	110	13916.13	40.35	110	7235.00	159.66	20	61.20	174.96

Table 7.4: Results for ternary instances plus $a^\top x \leq b$

n	Q-MIST						BARON		
	CD2D			CSDP					
	#	nodes	time	#	nodes	time	#	nodes	time
10	110	29.36	0.01	110	11.15	0.05	110	1.41	0.08
20	110	185.78	0.24	110	70.75	0.29	110	9.15	1.04
30	110	685.64	0.74	110	247.80	2.16	110	16.04	38.17
40	110	2361.33	3.85	110	1035.29	14.95	56	37.23	289.56
50	110	9844.31	31.10	110	7140.91	165.15	21	67.48	191.01

Summary and outlook

In this thesis we have proposed a coordinate ascent algorithm to solve the dual problem of the semidefinite relaxation of non-convex quadratic programming problems where each variable is restricted to a finite sub-set of the integer numbers (Problem (3.1)). Our approach is an extension of an algorithm devised by Dong [26] for a similar (but simpler) class of semidefinite problems. We have embedded this approach into Q-MIST, the branch-and-bound scheme proposed by Buchheim and Wiegele in [17], originally designed for a larger class of quadratic problems and implemented to use interior point methods for solving the semidefinite relaxations.

The motivation of this research was two-fold. First of all, it aimed at obtaining quick and strong lower bounds for the objective function value of quadratic integer problems (3.1), that can be used later as a bounding procedure inside Q-MIST. The second motivation was the development of an algorithm that is capable to solve instances of larger size than allowed by interior point methods.

In our approach we have cleverly exploited the structure of the problem. Due to the finite variable domains, the semidefinite relaxation of Problem (3.1) could be completely described by a finite number of linear inequality constraints per variable plus the semidefinite constraint $X \succeq 0$ where each set of constraint involves only the variables x_{00}, x_{0i} and x_{ii}. We reformulated the inequality constraints using matrix notation. The matrices associated with the inequalities are characterized by sparsity and a low rank. In fact, they are symmetric matrices with at most three entries different from zero and have rank equal to one or two. In addition, we proved that for each variable at most two constraints can be active. This means that, in an optimal solution, only a few dual variables can be non-zero among the exponentially many. All these properties together motivated the use of coordinate-wise optimization methods to solve the dual problem (3.4). To be precise, we extended the ideas of a coordinate ascent method proposed by Dong [26]. The author has studied the same class of quadratic problems as in [17], and proposed a convex quadratically constrained reformulation for this class of problems. Convex relaxations for the resulting problem are obtained via a semi-infinite relaxation. In order to produce valid cutting surfaces, a semidefinite problem has to be solved. We have observed that this problem has a similar structure to Problem (3.4). The problems we are solving are however more general, the semidefinite constraint includes matrices of rank one or two, instead of diagonal (rank-one) matrices. We deal with an exponential number of dual of variables restricted to be non-positive. The objective function in our case is linear instead of scalar. The algorithm devised by Dong for this class of semidefinite problems consists in the introduction of a barrier function to model the semidefinite constraint, and the use of coordinate descent methods with exact line search.

Summarizing, our approach consists in the following: similar as in [26], we have

lifted the semidefinite constraint into the objective function and introduce a penalty parameter. We deal with the non-positivity constraint on the dual variables explicitly in the line search. The choice of the ascent direction is performed using the Gauss-Southwell rule. The regular structure of the problem leads to consider only a few coordinates among the exponentially many potential candidates. In the chosen direction, we performed an exact line search which turns out in a closed-form formula for computing the step length. We have provided the theoretical requirements that guarantee the existence of optimal step sizes. Each iteration of the algorithm requires the computation of an inverse matrix of order $n + 1$, that is the perturbation of one previously known matrix by a rank one or two constraint matrix. This computation has been done efficiently using the Woodbury formula, in $O(n^2)$ time. Even more, we have improved the convergence of our algorithm by taking into account the special structure of one the constraint matrices, whose rank is one and has the property that every linear combination with any other constraint matrix still has rank at most two. In other words, to improve the original approach, we perform a plane-search rather than a line search, and simultaneously update two dual variables and still recompute the inverse matrix in $O(n^2)$ time.

We have implemented two algorithms: Algorithm CD and Algorithm CD2D, one-dimension and two-dimension update, respectively. As a next step, we have integrated both algorithms into the branch-and-bound framework Q-MIST. We have evaluated the performance of both algorithms in the root node of the branch-and-bound tree and compared them with CSDP, the SDP solver used in [17]. The experiments performed show that the two main objectives of our research were successfully accomplished for randomly generated instances. We also showed experimentally that Algorithm CD2D significantly outperforms Algorithm CD, it produces lower bounds as strong as the ones provided by CSDP and it runs much faster for instances of large size, with ternary and integer domains. Additionally, Algorithm CD2D has been able to solve larger size instances than other more general non-convex integer non-linear programing solvers like BARON and COUENNE.

Nevertheless, we consider that there is an extensive path of research that could be considered to improve our approach and extend its range of application. From the theoretical point of view, one of the main open questions we were not able to precisely answer is related to the influence of the parameter β_{ij} on the dual problem (3.4). Therefore, having a better understanding of this parameter may lead to better convergence results of the algorithm. Another possible extension, that goes in the same direction of the application explained in Chapter 6, could be to consider tighter formulation of the linear constraints that can be still formulated as low-rank matrices. Form the implementation side, further experiments could be performed to see the influence of other parameters in the convergence of the algorithm, such as the update rule of the penalty parameter σ, the starting value of σ, or the selection of a different starting point that does not require the expensive computation of the smallest eigenvalue. Finally, it would be interesting to evaluate the performance of our approach with instances that appear in real-world applications.

References

[1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5:13–51, 1993.

[2] F. Alizadeh, J.-P.A. Haeberly, and M.L. Overton. Complementarity and nondegeneracy in semidefinite programming. *Mathematical Programming*, 77(1):111–128, 1997.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[4] O. Banerjee, E. Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, June 2008.

[5] F. Barahona, M. Jünger, and G. Reinelt. Experiments in quadratic 0–1 programming. *Mathematical Programming*, 44(1):127–137, 1989.

[6] A.I. Barvinok. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13:189–202, 1995.

[7] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4-5):597–634, 2009.

[8] D.P. Bertsekas. *Nonlinear programming*. Athena Scientific, Belmont (Mass.), 1999.

[9] L. Bo and C. Sminchisescu. Greedy block coordinate descent for large scale gaussian process regression. *CoRR*, abs/1206.3238, 2012.

[10] P. Van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical report, University of Amsterdam, Department of Mathematics, Amsterdam, 1981.

[11] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11(1-4):613–623, 1999.

[12] C.A. Bouman and K. Sauer. A unified approach to statistical tomography using coordinate descent optimization. *Transactions on Image Processing*, 5(3):480–492, March 1996.

[13] C. Buchheim, A. Caprara, and A. Lodi. An effective branch-and-bound algorithm for convex quadratic integer programming. *Mathematical Programming*, 135(1-2):369–395, 2012.

[14] C. Buchheim, R. Hübner, and A. Schöbel. Ellipsoid bounds for convex quadratic integer programming. *SIAM Journal on Optimization*, 25(2):741–769, 2015.

[15] C. Buchheim, M. Montenegro, and A. Wiegele. A coordinate ascent method for solving semidefinite relaxations of non-convex quadratic integer programs. In *ISCO*, volume 9849 of *Lecture Notes in Computer Science*, pages 110–122. Springer, 2016.

[16] C. Buchheim and E. Traversi. On the separation of split inequalities for non-convex quadratic integer programming. *Discrete Optimization*, 15(C):1–14, 2015.

[17] C. Buchheim and A. Wiegele. Semidefinite relaxations for non-convex quadratic mixed-integer programming. *Mathematical Programming*, 141(1-2): 435–452, 2013.

[18] S. Burer and A. Letchford. On nonconvex quadratic programming with box constraints. *SIAM Journal on Optimization*, 20(2):1073–1089, 2009.

[19] S. Burer and A. Letchford. Non-convex mixed-integer nonlinear programming: a survey. *Surveys in Operations Research and Management Science*, 17(2):97–106, 2012.

[20] S. Burer and R.D. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming (series B)*, 95:2003, 2001.

[21] S. Burer and D. Vandenbussche. A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. *Mathematical Programming*, 113(2):259–282, 2008.

[22] S. Burer and D. Vandenbussche. Globally solving box-constrained nonconvex quadratic programs with semidefinite-based finite branch-and-bound. *Computational Optimization and Applications*, 43(2):181–195, 2009.

[23] K.W. Chang, C.J. Hsieh, and C.J. Lin. Coordinate descent method for large-scale l2-loss linear support vector machines. *Journal of Machine Learning Research*, 9:1369–1398, June 2008.

[24] S.J. Chung and K.G. Murty. Polynomially bounded ellipsoid algorithms for convex quadratic programming. In *Nonlinear Programming 4*, pages 439 – 485. Academic Press, 1981.

[25] R.W. Cottle. Manifestations of the schur complement. *Linear Algebra and its Applications*, 8(3):189 – 211, 1974.

[26] H. Dong. Relaxing nonconvex quadratic functions by multiple adaptive diagonal perturbations. *SIAM Journal on Optimization*, 26(3):1962–1985, 2016.

[27] J.A. Fessler. Grouped coordinate descent algorithms for robust edge-preserving image restoration. volume 3170, pages 184–194, 1997.

[28] U. Fincke and M. Pohst. Improved Methods for Calculating Vectors of Short Length in a Lattice, Including a Complexity Analysis. *Mathematics of Computation*, 44(170):463–471, 1985.

[29] C.A. Floudas and V. Visweswaran. *Quadratic Optimization*, pages 217–269. Springer US, Boston, MA, 1995.

[30] M.R. Garey and D.S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[31] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237 – 267, 1976.

[32] P.E. Gill and E. Wong. Sequential quadratic programming methods. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 147–224. Springer New York, 2012.

[33] M.X. Goemans. Semidefinite programming in combinatorial optimization. *Mathematical Programming*, 79:143–161, 1997.

[34] M.X. Goemans and D.P. Williamson. 0.878-approximation algorithms for MAX CUT and MAX 2SAT. In *Proc. 26th STOC*, pages 422–431, New York, 1994. ACM Press.

[35] M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[36] L. Grippo, L. Palagi, M. Piacentini, V. Piccialli, and G. Rinaldi. Speedp: an algorithm to compute SDP bounds for very large max-cut instances. *Mathematical Programming*, 136(2):353–373, 2012.

[37] L. Grippo, L. Palagi, and V. Piccialli. An unconstrained minimization method for solving low-rank SDP relaxations of the maxcut problem. *Mathematical Programming*, 126(1):119–146, 2011.

[38] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[39] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Algorithms and combinatorics. Springer-Verlag, Berlin, New York, 1988.

[40] W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239, 1989.

[41] P.L. Hammer and A.A. Rubin. Some remarks on quadratic programming with 0-1 variables. *RAIRO - Operations Research - Recherche Opérationnelle*, 4(V3):67–79, 1970.

[42] C. Helmberg. Semidefinite programming. Technical Report SC-99-49, ZIB, Takustr.7, 14195 Berlin, 1999.

[43] C. Helmberg. *Semidefinite Programming for Combinatorial Optimization*. Professorial dissertation, Technische Univertität Berlin, Berlin, 2000.

[44] C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696, 2000.

[45] C. Helmberg, F. Rendl, R.J. Vanderbei, and H. Wolkowicz. An interior–point method for semidefinite programming. *SIAM Journal on Optimization*, 1994.

[46] C. Helmberg, F. Rendl, and R. Weismantel. A semidefinite programming approach to the quadratic knapsack problem. *Journal of Combinatorial Optimization*, 4(2):197–215, 2000.

[47] S. Homer and M. Peinado. Design and performance of parallel and distributed approximation algorithms for maxcut. *Journal of Parallel Distributed Computing*, 46(1):48–61, 1997.

[48] R.A. Horn and C.R. Johnson, editors. *Matrix Analysis*. Cambridge University Press, New York, NY, USA, 1986.

[49] F.L. Huang, C.J. Hsieh, K.W. Chan, and C.J. Lin. Iterative scaling and coordinate descent methods for maximum entropy models. *Journal of Machine Learning Research*, 11:815–848, March 2010.

[50] M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

[51] S. Kapoor and P.M. Vaidya. Fast algorithms for convex quadratic programming and multicommodity flows. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 147–159, 1986.

[52] R.M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.

[53] E. de Klerk. *Aspects of semidefinite programming: interior point algorithms and selected applications*. Applied optimization. Kluwer Academic Publishers, Dordrecht, Boston, London, 2002.

[54] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang. The unconstrained binary quadratic programming problem: a survey. *Journal of Combinatorial Optimization*, 28(1):58–81, 2014.

[55] M. Kojima, S. Shindoh, and S. Hara. Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM Journal on Optimization*, 7(1):86–125, January 1997.

[56] M.K. Kozlov, S.P. Tarasov, and L.G. Hačijan. The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228, 1980.

[57] M. Laurent and S. Poljak. On the facial structure of the set of correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, 17(3):530–547, 1996.

[58] J. W. Lawrence. Reduction of integer polynomial programming problems to zero-one linear programming problems. *Operations Research*, 15(6):1171–1174, 1967.

[59] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1):1–7, 1979.

[60] Z.-Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, 1992.

[61] Z.-Q. Luo and P. Tseng. Error bounds and convergence analysis of feasible descent methods: a general approach. *Annals of Operations Research*, 46(1):157–178, 1993.

[62] G.P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I – Convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[63] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, 2002.

[64] K.S. Miller. On the inverse of the sum of matrices. *Mathematics Magazine*, 54(2):67–72, 1981.

[65] R. Misener and C. A. Floudas. ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 2014. DOI: 10.1007/s10898-014-0166-2.

[66] B. Mohar and S. Poljak. *Eigenvalues in Combinatorial Optimization*, pages 107–151. Springer New York, New York, NY, 1993.

[67] R.D. Monteiro and I. Adler. Interior path following primal-dual algorithms. part ii: Convex quadratic programming. *Mathematical Programming*, 44(1):43–66, 1989.

[68] R.D. Monteiro and M.J. Todd. Path-following methods for semidefinite programming. *In: Saigal, R. Vandenberghe, L. Wolkowicz, H. (eds) Handbook of Semidefinite Programming. Kluwer Academic Publishers, Boston-Dordrecht-London*, 2000.

[69] J. Ortega and W. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

[70] M. Padberg. The boolean quadric polytope: Some characteristics, facets and relatives. *Mathematical Programming*, 45(1):139–172, 1989.

[71] P.M. Pardalos. Global optimization algorithms for linearly constrained indefinite quadratic problems. *Computers & Mathematics with Applications*, 21(6):87 – 97, 1991.

[72] P.M. Pardalos and S.A. Vavasis. Quadratic programming with one negative eigenvalue is NP-hard. *Journal of Global Optimization*, 1:15–22, 1991.

[73] G. Pataki. On the rank of extreme matrices on semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematical Methods of Operations Research*, 23:339–358, 1998.

[74] D. Pisinger. The quadratic knapsack problem a survey. *Discrete Applied Mathematics*, 155(5):623 – 648, 2007.

[75] E. Polak. *Computational methods in optimization : a unified approach*, volume 77 of *Mathematics in science and engineering*. Academic press, New York, London, 1971.

[76] Svatopluk Poljak and Franz Rendl. Solving the max-cut problem using eigenvalues. *Discrete Applied Mathematics*, 62(1):249 – 278, 1995.

[77] S. Puntanen and G.P. Styan. *Historical Introduction: Issai Schur and the Early Development of the Schur Complement*, pages 1–16. Springer US, 2005.

[78] F. Rendl, G. Rinaldi, and A. Wiegele. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Mathematical Programming*, 121(2):307–335, 2010.

[79] I.G. Rosenberg. Brèves communications. 0-1 optimization and non-linear programming. *RAIRO - Operations Research - Recherche Opérationnelle*, 6(V2):95–97, 1972.

[80] N. V. Sahinidis. *BARON 16.3.4: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual, 2016.

[81] S. Sahni. Computationally related problems. *SIAM Journal on Computing*, 3(4):262–279, 1974.

[82] R.W.H. Sargent and D.J. Sebastian. On the convergence of sequential minimization algorithms. *Journal of Optimization Theory and Applications*, 12(6):567–575, 1973.

[83] A. Shapiro. Extremal problems on the set of nonnegative definite matrices. *Linear Algebra and its Applications*, 67:7 – 18, 1985.

[84] H.D. Sherali and W.P. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*. Springer, 1998.

[85] C. De Simone. The cut polytope and the boolean quadric polytope. *Discrete Mathematics*, 79(1):71 – 75, 1990.

[86] M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103:225–249, 2005.

[87] M.J. Todd. A study of search directions in primal-dual interior-point methods for semidefinite programming. *Operations methods and Software*, 11:1–46, 1999.

[88] P. Tseng and S. Yun. A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training. *Computational Optimization and Applications*, 47(2):179–206, 2010.

[89] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, March 1996.

[90] S.A. Vavasis. Quadratic programming is in np. *Information Processing Letters*, 36(2):73 – 77, 1990.

[91] S.A. Vavasis. *Nonlinear Optimization: Complexity Issues*. Oxford University Press, Inc., New York, NY, USA, 1991.

[92] H. Wolkowicz. Some applications of optimization in matrix theory. *Linear Algebra and its Applications*, 40:101 – 118, 1981.

[93] H. Wolkowicz, R. Saigal, and L. Vandenberghe. *Handbook of semidefinite programming: theory, algorithms, and applications*. International series in operations research & management science. Kluwer Academic, Boston, London, 2000.

[94] S.J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, June 2015.

[95] J.C. Ye, K.J. Webb, C.A. Bouman, and R.P. Millane. Optical diffusion tomography by iterative-coordinate-descent optimization in a bayesian framework. *Journal of the Optical Society of America A*, 16(10):2400–2412, Oct 1999.

[96] Y. Ye and E. Tse. An extension of Karmarkar's projective algorithm for convex quadratic programming. *Mathematical Programming*, 44:157–179, 1989.