Noname manuscript No.
(will be inserted by the editor)

SDP-based Branch-and-Bound for Non-convex Quadratic Integer Optimization

Christoph Buchheim · Maribel Montenegro · Angelika Wiegele

Received: date / Accepted: date

Abstract Semidefinite programming (SDP) relaxations have been intensively used for solving discrete quadratic optimization problems, in particular in the binary case. For the general non-convex integer case with box constraints, the branch-and-bound algorithm Q-MIST has been proposed [11], which is based on an extension of the well-known SDP-relaxation for max-cut. For solving the resulting SDPs, Q-MIST uses an off-the-shelf interior point algorithm.

In this paper, we present a tailored coordinate ascent algorithm for solving the dual problems of these SDPs. Building on related ideas of Dong [15], it exploits the particular structure of the SDPs, most importantly a small rank of the constraint matrices. The latter allows both an exact line search and a fast incremental update of the inverse matrices involved, so that the entire algorithm can be implemented to run in quadratic time per iteration. Moreover, we describe how to extend this approach to a certain two-dimensional coordinate update. Finally, we explain how to include arbitrary linear constraints into this framework, and evaluate our algorithm experimentally.

Keywords Quadratic integer programming · semidefinite programming · coordinate-wise optimization

This work was partially supported by the Marie Curie Initial Training Network MINO (Mixed-Integer Nonlinear Optimization) funded by the European Union. The first and the second author were partially supported by the DFG under grant BU 2313/4-2. This paper is based on the PhD thesis [23]; a preliminary version can be found in [10].

Christoph Buchheim
Fakultät für Mathematik, TU Dortmund, Vogelpothsweg 87, 44227 Dortmund, Germany
E-mail: christoph.buchheim@tu-dortmund.de

Maribel Montenegro
Fakultät für Mathematik, TU Dortmund, Vogelpothsweg 87, 44227 Dortmund, Germany
E-mail: maribel.montenegro@tu-dortmund.de

Angelika Wiegele
Alpen-Adria-Universität Klagenfurt, Universitätsstr. 65-67, 9020 Klagenfurt, Austria
E-mail: angelika.wiegele@aau.at

1 Introduction

We address integer quadratic optimization problem of the following form

$$\begin{aligned} \min \quad & x^\top \hat{Q} x + \hat{l}^\top x + \hat{c} \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^n, \end{aligned} \tag{IQP}$$

where $\hat{Q}$ is a symmetric $n \times n$ matrix, $\hat{l} \in \mathbb{R}^n$, $\hat{c} \in \mathbb{R}$, $A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$.

Even in the special case of a convex objective function, i.e., when $\hat{Q}$ is positive semidefinite, Problem (IQP) is NP-hard in general due to the presence of integrality constraints. In fact, in the unconstrained case it is equivalent to the NP-hard closest vector problem [4]. However, dual bounds can be computed by relaxing integrality and then solving the resulting convex QP-relaxations. These bounds can be used within a branch-and-bound algorithm [8] and improved in various ways exploiting integrality [7,9]. Dual bounds can also be derived from semidefinite relaxations [25]. More generally, convex discrete optimization problems can be addressed by solving convex non-linear relaxations or by other approaches such as outer approximation [5]. In the case of a non-convex objective, the problem remains NP-hard even if integrality constraints are dropped. If only box constraints are considered, the resulting problem is called Box-QP, it has attracted a lot of attention in the literature [14,12,6].

For integer variables subject to box constraints and a general quadratic objective function, a branch-and-bound algorithm called Q-MIST has been presented by Buchheim and Wiegele [11]. It is based on SDP formulations that generalize the well-known SDP relaxation for max-cut [27]. At each node of the branch-and-bound tree, Q-MIST calls a standard interior point method to solve a semidefinite relaxation obtained from Problem (IQP). 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., [29].

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 [19] and (low rank) reformulations as unconstrained non-convex optimization problems together with the use of non-linear methods to solve the resulting problems [21,13,16]. Recently, another algorithm has been proposed by Dong [15] for solving a class of semidefinite programs. The author also considers Problem (IQP) with box-constraints and reformulates it 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 Contribution. In this paper, we focus on improving Q-MIST by using an alternative method for solving the SDP relaxation. Our approach tries to exploit the specific problem structure, 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 by means of a coordinate ascent algorithm that adapts and generalizes the algorithm proposed in [15], based on a barrier model. In particular, we can efficiently find a coordinate with largest gradient entry, even if the number of constraints is exponentially large, and perform an exact line search using the Woodbury formula. We can guarantee the existence of an optimal step length by showing that the level sets of our barrier problem are always bounded.

Moreover, we can extend this idea and optimize over certain combinations of two coordinates simultaneously, which leads to a significant improvement of running times. Finally we explain how to extend this method in order to include arbitrary linear constraints instead of only box constraints. Experimentally, we show that this method not only improves Q-MIST with respect to using a general interior point algorithm, but also outperforms standard optimization software for most types of instances.

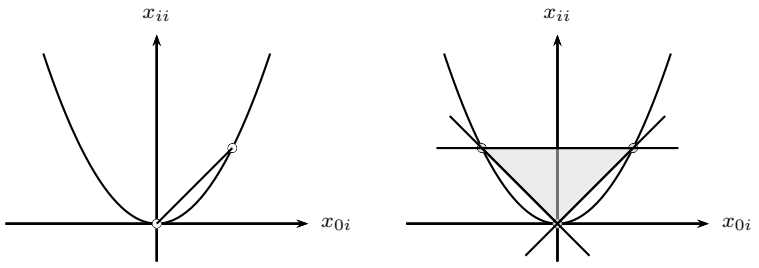
Outline. This paper is organized as follows. In Section 2 we recall the semidefinite relaxation of Problem (IQP) having box-constraints only, rewrite it in a matrix form, compute its dual and point out the properties of this problem that will be used later. In Section 3 we adapt and extend the coordinate descent algorithm presented in [15]. Then, we improve this 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, which is presented in Section 4. Finally, in Section 5 we evaluate this approach within the branch-and bound framework of Q-MIST. The experiments show that our approach produces lower bounds of the same quality but in significantly shorter computation time for instances of large size.

2 Preliminaries

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

$$\begin{aligned} \min \quad & x^\top \hat{Q} x + \hat{l}^\top x + \hat{c} \\ \text{s.t.} \quad & x \in D_1 \times \cdots \times D_n, \end{aligned} \tag{1}$$

where $\hat{Q} \in S_n$ is not necessarily positive semidefinite, $\hat{l} \in \mathbb{R}^n$, $\hat{c} \in \mathbb{R}$, and the feasible domain for variable x_i is a set $D_i = \{l_i, \ldots, u_i\}$ for $l_i, u_i \in \mathbb{Z}$; by S_n we denote the set of all symmetric $n \times n$-matrices. In [11], a more general class of problems has been considered, allowing arbitrary closed subsets $D_i \subseteq \mathbb{R}$.



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

Fig. 1 The set $P(D_i)$ and its polyhedral description

However, in many applications, the set D_i is finite, and for simplicity we may assume $D_i = \{l_i, \dots, u_i\}$ then. Moreover, the algorithm presented in the following is easily adapted to a mixed-integer setting. In Section 4, we will additionally allow arbitrary linear contraints.

2.1 Semidefinite relaxation

In [11] it has been proved that Problem (1) is equivalent to

$$\begin{aligned} \min \quad & \langle Q, X \rangle \\ \text{s.t.} \quad & (x_{0i}, x_{ii}) \in P(D_i) \quad \forall i = 1, \dots n \\ & x_{00} = 1 \\ & \operatorname{rank}(X) = 1 \\ & X \succeq 0 \,, \end{aligned} \tag{2}$$

where $P(D_i) := \operatorname{conv}\{(u, u^2) \mid u \in D_i\}$ and the matrix $Q \in S_{n+1}$ is defined as

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

As only the rank-constraint is non-convex in this formulation, by dropping it we obtain a semidefinite relaxation of (1).

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 1 shows two examples.

Lemma 1 *Let $D_i = \{l_i, \dots, u_i\}$ 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*

$$-x_{ii} + (2j+1)x_{0i} \le j(j+1), \quad j = l_i, l_i+1, \dots, u_i - 1,$$

and one upper bounding facet

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

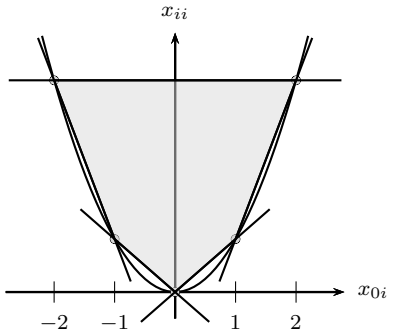


Fig. 2 The polytope $P(\{-2,-1,0,1,2\})$. Lower bounding facets are indexed, from left to right, by $j=-2,-1,0,1$, the upper bounding facet by 2.

Notice that in case $|D_i|=2$, i.e., when the variable is binary, there is only one lower bounding facet that together with the upper bounding facet results in a single equation. However, for sake of simplicity, we will not distinguish these cases in the following.

2.2 Matrix formulation

The relaxation of (2) contains the constraint $x_{00}=1$, this fact is exploited to rewrite the polyhedral description of $P(D_i)$ presented in Lemma 1 as

$$\begin{aligned}(\beta_{ij}-j(j+1))x_{00}-x_{ii}+(2j+1)x_{0i}&\le\beta_{ij},\quad j=l_i,l_i+1,\dots,u_i-1\\(\beta_{iu_i}+l_iu_i)x_{00}+x_{ii}-(l_i+u_i)x_{0i}&\le\beta_{iu_i}\end{aligned}$$

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, 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 as

$$\langle A_{ij},X\rangle\le\beta_{ij}\ ,$$

where, for each variable $i\in\{1,\dots,n\}$, the index ij represents the inequalities corresponding to lower bounding facets $j=l_i,l_i+1,\dots,u_i-1$ and $j=u_i$ corresponds to the upper bounding facet; see Figure 2 for an illustration.

Since each constraint links only the variables x_{00}, x_{0i} and x_{ii}, the constraint matrices $A_{ij}\in S_{n+1}$ are sparse, the only non-zero entries being

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

in the upper bound constraint and

$$(A_{ij})_{00}=\beta_{ij}-j(j+1),\quad (A_{ij})_{0i}=(A_{ij})_{i0}=j+\tfrac{1}{2},\quad (A_{ij})_{ii}=-1$$

in the case of a lower bound constraint. To be consistent, the constraint $x_{00} = 1$ is also written in matrix form as $\langle A_0, X\rangle = 1$, where $A_0 := e_0 e_0^\top \in S_{n+1}$. In summary, the SDP relaxation of (2) can now be written as

$$\begin{aligned} \min \quad & \langle Q, X\rangle \\ \text{s.t.} \quad & \langle A_0, X\rangle = 1 \\ & \langle A_{ij}, X\rangle \le \beta_{ij} \quad \forall j = l_i, \dots, u_i \quad \forall i = 1, \dots, n \\ & X \succeq 0. \end{aligned} \tag{3}$$

The following observation is crucial for the algorithm presented in this paper.

Lemma 2 *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) using a coordinate-wise approach, leading to a computationally cheap update at each iteration and an easy computation of the exact step size.

2.3 Dual problem

In order to derive the dual of Problem (3), we first introduce the linear operator $\mathcal{A}\colon S_{n+1} \longrightarrow \mathbb{R}^{m+1}$ as

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

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

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

We thus obtain the dual semidefinite program of Problem (3) as

$$\begin{aligned} \max \quad & \langle b, y\rangle \\ \text{s.t.} \quad & Q - \mathcal{A}^\top y \succeq 0 \\ & y_0 \in \mathbb{R} \\ & y_{ij} \le 0 \quad \forall j = l_i, \dots, u_i \quad \forall i = 1, \dots, n, \end{aligned} \tag{4}$$

the vector $b \in \mathbb{R}^{m+1}$ being defined as $b_0 = 1$ and $b_{ij} = \beta_{ij}$.

We conclude this section by emphasizing some characteristics of any feasible solution of Problem (3).

Lemma 3 *Let X^* be a feasible solution of Problem* (3)*. For $i \in \{1,\dots,n\}$, consider the active set*

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

corresponding to variable i. Then

(i) for all $i \in \{1,\dots,n\}$, $|\mathcal{A}_i| \le 2$, and
(ii) if $|\mathcal{A}_i| = 2$, 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 $\langle A_{ij}, X\rangle \le \beta_{ij}$ are defined it is impossible to have more than two inequalities intersecting at one point; see for example Figures 1 and 2. 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 (4), Lemma 3 (i) means that in any optimal solution at most $2n+1$ out of the $m+1$ variables can be non-zero. Such a small number of non-zero variables motivates to consider a coordinate-wise optimization method to solve the dual problem (4). Moreover, by Lemma 3 (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.

2.4 Primal and dual strict feasibility

We next show that both Problem (3) and its dual, Problem (4), are strictly feasible. Using this we can conclude that strong duality holds and that both problems attain their optimal solutions.

Theorem 1 *Assume in the following that binary variables are modeled by one equation (instead of two inequalities). Then Problem* (3) *is strictly feasible.*

Proof Consider the functions $l_i(x)$ and $u_i(x)$ bounding x_{ii} in terms of x_{0i}, given by the upper and the lower bounding facets described in Lemma 1:

$$\begin{aligned} x_i \in [j, j+1] &\longmapsto l_i(x_i) := (2j+1)x_i - j(j+1), \quad j = l_i,\dots,u_i-1 \\ x_i \in [l_i, u_i] &\longmapsto u_i(x_i) := (l_i+u_i)x_i - l_iu_i. \end{aligned}$$

Notice that in case of binary variables, $l_i(x_i) = u_i(x_i)$. Now, define $x_0 := 1$ and $x_i := \frac{1}{2}(l_i+u_i)$ and let

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

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

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

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 linear inequalities. □

Theorem 2 *Problem* (4) *is strictly feasible.*

Proof If $Q \succ 0$, we have that $y^0 = 0$ is a feasible solution of Problem (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 $y^0_{ij} \le 0$ by construction, so it remains to show that $Q - \mathcal{A}^\top y^0 \succ 0$. To this end, first note that

$$\tilde{c} := \hat{c} - y_0 - \tilde{y}\sum_{i=1}^{n}(A_{iu_i})_{00} = 1 + (\tfrac{1}{2}\hat{l} - \tilde{y}a)^\top(\tfrac{1}{2}\hat{l} - \tilde{y}a) > 0\,. \tag{5}$$

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 (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 $\lambda_{max}(B) = (\frac{1}{2}\hat{l} - \tilde{y}a)^\top(\frac{1}{2}\hat{l} - \tilde{y}a) \ge 0$ and thus

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

by definition of $\tilde{y}$. We have found y^0 such that $y^0 \le 0$ and $Q - \mathcal{A}^\top y^0 \succ 0$, hence 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 $Q - \mathcal{A}^\top(y^0 - \epsilon\mathbb{1}) \succ 0$. □

Corollary 3 *Problem* (3) *and its dual, Problem* (4)*, both admit optimal solutions, and there is no duality gap.*

3 A coordinate ascent method

Our approach tries to exploit the specific structure of Problem (3), 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 (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 2 proposed by Dong [15]. Dong reformulates Problem (1) as a convex quadratically constrained problem and then produces convex relaxations 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 (4) has a similar structure to the semidefinite problems solved in [15], therefore similar ideas can be applied to solve it. However Problem (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.

As a first step, we introduce a barrier term in the objective function of Problem (4) to model the semidefinite constraint $Q - \mathcal{A}^\top y \succeq 0$. We obtain

$$\begin{aligned} \max \quad & f(y;\sigma) := \langle b, y\rangle + \sigma \log\det(Q - \mathcal{A}^\top y) \\ \text{s.t.} \quad & Q - \mathcal{A}^\top y \succ 0 \\ & y_0 \in \mathbb{R} \\ & y_{ij} \le 0 \quad \forall j = l_i, \dots, u_i \quad \forall i = 1, \dots, n \end{aligned} \tag{6}$$

for $\sigma > 0$. The barrier term tends to $-\infty$ if the smallest 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. Therefore, the role of the barrier 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 will see later that we do not need to introduce a barrier term for the non-negativity constraints $y_{ij} \le 0$, as they can be dealt with directly.

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 in the interior of the positive semidefinite cone; see e.g., [18].

Theorem 4 *For all $\sigma > 0$ and $z \in \mathbb{R}$, the level set*

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

of Problem (6) *is bounded.*

Proof We define the following set $\mathcal{N} := \{y \in \mathbb{R}^m \mid y_{ij} \le 0\}$ and first show that for all $y \in \mathcal{N} \setminus \{0\}$ with $\mathcal{A}^\top y = 0$, it holds that $\langle b, y\rangle \ne 0$. 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 $i' \in \{1, \dots, n\}$ and $j' \in \{l_{i'}, \dots, u_{i'}\}$ 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 1, we know that there exists a strictly feasible solution $X^0 \succ 0$ of Problem (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

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

but this is a contradiction. Secondly, observe that for all $y \in \mathcal{N}$, it holds that

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

The last inequality follows by Lemma 1.2.4 in [18]. We have that $\lambda_{\min}(X^0) > 0$ since $X^0 \succ 0$. Thus

$$\lambda_{\max}(Q - \mathcal{A}^\top y) \leq \frac{1}{\lambda_{\min}(X^0)} \big(\left\langle Q, X^0 \right\rangle - \langle b, y\rangle \big). \tag{7}$$

Since the level sets $\mathscr{L}_f(z)$ 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 (6), 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 of $f(\bar{y} + sy; \sigma)$ for $s \to \infty$: 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 either have $\lambda_{\min}(Q - \mathcal{A}^\top \bar{y} - s^* \mathcal{A}^\top y) = 0$ 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 (7) 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 $f(\bar{y} + sy) \to -\infty$ for $s \to \infty$. □

The boundedness of the upper level sets and the strict concavity of the objective function guarantee the convergence of a coordinate ascent method, when using the cyclical rule to select the coordinate direction and exact line search to compute the step length [24]. However, for practical performance reasons, we apply the Gauss-Southwell rule to choose the coordinate direction. Below we describe a general algorithm to solve Problem (6) in a coordinate-wise maximization manner.

Outline of a barrier coordinate ascent algorithm for Problem (4)

1: **Starting point:** choose $\sigma > 0$ and any feasible solution y of (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 barrier 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 [15]. We thus need to compute the gradient of the objective function of Problem (6). See, e.g., [18] 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

$$\nabla_y f(y; \sigma) = b - \sigma\mathcal{A}(W)\ . \tag{8}$$

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 W, i.e., the inverse of an $(n+1)\times(n+1)$-matrix that only changes by a factor of one constraint matrix when changing the value of the dual variable. Thanks to the Woodbury formula and to the fact that our constraint matrices are rank-two matrices, the matrix W can be easily computed incrementally, 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. In fact, the special structure of Problem (3) 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 W in $O(n^2)$ time (see Section 3.4). Thus, the main ingredient of our algorithm is the computationally cheap update of W at each iteration and an easy computation of the optimal step size.

Before describing in detail how to choose an ascent direction and how to compute the step size, we address the choice of a feasible starting point. Compared to [15], the situation is more complex. We propose to choose as starting point the vector y^0 defined in the proof of Theorem 2. The construction described there can be directly implemented, however, it involves the computation of the smallest eigenvalue of $\hat{Q}$. Together with the computation of the inverse of $Q - \mathcal{A}^\top y^0$, this is the most expensive task in our algorithm, it requires $O(n^3)$ time.

3.1 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

$$ij^{(k)} \in \arg\max_{ij} |\nabla_y f(y;\sigma)_{ij}| \,. \tag{9}$$

However, moving a coordinate ij to a positive direction is allowed only in case $y_{ij} < 0$, so that the coordinate $ij^{(k)}$ in (9) has to be chosen among those satisfying either $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$, or $\nabla_y f(y;\sigma)_{ij} < 0$. The entries of the gradient depend on the type of inequality. By (8), 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 x_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 x_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 $\{l_i, \ldots, u_i - 1\}$. 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 $l_i, \ldots, u_i - 1$; 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 (9), independent of the sets D_i.

3.2 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

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

unless the chosen coordinate is zero, in which case the upper bound on s is dropped. Note that the function $s \mapsto f(y^{(k)} + s e_{ij^{(k)}}; \sigma)$ is strictly concave on $\{s \in \mathbb{R} \mid Q - \mathcal{A}^\top (y^{(k)} + s e_{ij^{(k)}}) \succ 0\}$. 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

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

Then, the gradient with respect to s is

$$\nabla_s f(y + s e_{ij}; \sigma) = \beta_{ij} - \sigma \left\langle A_{ij}, (W^{-1} - s A_{ij})^{-1} \right\rangle. \tag{10}$$

The next lemma states that if the coordinate direction is chosen as explained in the previous section, and the gradient (10) has at least one root in the right direction of the line search, then there exists a feasible step length.

Lemma 4

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

Proof We show (i), the proof of (ii) follows analogously. Let y be a feasible point of Problem (6) 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 infeasible, hence either $Q - \mathcal{A}^\top(y + s^+e_{ij})$ is not positive definite or $s^+ \geq -y_{ij}$. In the first case, there exists $0 < s' \leq s^+$ 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 $\nabla_y f(y;\sigma)_{ij} > 0$. □

If in addition we exploit that the level sets of the function are bounded, as shown by Theorem 4, then we can derive the following theorem:

Theorem 5

(i) Let the coordinate ij be chosen such that $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$. Then the gradient (10) *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 chosen such that $\nabla_y f(y;\sigma)_{ij} < 0$. Then the gradient (10) *has at least one negative root, and for the biggest negative root s^-, we have that $y + s^-e_{ij}$ is dual feasible and $\nabla_s f(y + s^-e_{ij};\sigma) = 0$.*

Proof We show (i), the proof of (ii) follows analogously. Let y be a feasible point of Problem (6) and ij such that $\nabla_y f(y;\sigma)_{ij} > 0$ and $y_{ij} < 0$.

From Theorem 4, we know that $\mathscr{L}_f(z)$, the level set of f at $z := f(y;\sigma)$, 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 $f(y;\sigma)$, or $y_{ij} + s > 0$.

In the first case, from continuous differentiability of $s \mapsto f(y + se_{ij};\sigma)$, 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 4, the smallest s^+ is feasible (which also directly follows from $y + s^+e_{ij} \in \mathscr{L}_f(z)$).

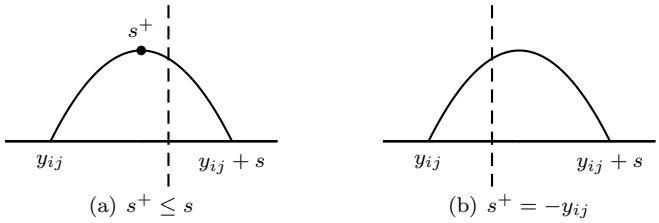


Fig. 3 Illustration of the existence of an optimal step size s^+, Theorem 5 (i)

Otherwise, if $y_{ij} + s > 0$, choose $s^+ = -y_{ij}$ and assume for a contradiction 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 3 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 2. Therefore, we will compute the inverse matrix $(W^{-1} - sA_{ij})^{-1}$ using the Woodbury formula for the rank-one or rank-two update. The computation is detailed in Appendix A.

3.3 Algorithm overview and running time

Our approach to solve Problem (4) is summarized in Algorithm CD.

Algorithm CD: Barrier coordinate ascent algorithm for Problem (4)

Input: $Q \in S_{n+1}$
Output: A lower bound on the optimal value of Problem (3)
Use Lemma 2 to compute $y^{(0)}$ such that $Q - \mathcal{A}^\top y^{(0)} \succ 0$
Compute $W^{(0)} \leftarrow (Q - \mathcal{A}^\top y^{(0)})^{-1}$
for $k = 0, 1, 2, \ldots$ **do**
  Choose a coordinate direction $e_{ij(k)}$ as described in Section 3.1
  Compute the step size $s^{(k)}$ as described in Section 3.2
  Update $y^{(k+1)} \leftarrow y^{(k)} + s^{(k)} e_{ij(k)}$
  Update $W^{(k)}$ using the Woodbury formula
  Update σ
  Terminate if some stopping criterion is met
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 3.1, 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 (4) at every iteration and hence a valid lower bound $\langle b, y^{(k)} \rangle$ for Problem (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). 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, this is further discussed in Section 5.

3.4 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) 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

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

To obtain a closed formula for the optimal step length s_0 in terms of a fixed step length s, 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:

$$\nabla_{s_0} f(y + s_0 e_0 + s e_{ij}; \sigma) = 1 - \sigma \left\langle A_0, (W^{-1} - s_0 A_0 - s A_{ij})^{-1} \right\rangle \,. \tag{11}$$

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

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

Substituting the last expression in the gradient (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 *Let s be a given step size along coordinate direction e_{ij}, then*

$$s_0 = \frac{1}{w(s)_{00}} - \sigma \tag{12}$$

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

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

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

$$\max_s \left\{ g_{ij}(s) \mid Q - \mathcal{A}^\top(y^{(k)} + s_0(s)e_0 + s e_{ij^{(k)}}) \succ 0, s \leq -y_{ij}^{(k)} \right\}. \tag{13}$$

Since the latter problem is uni-variate and differentiable, we need to find $s \in \mathbb{R}$ such that either $g'_{ij}(s) = 0$ and $s \leq -y_{ij}$ or $g'_{ij}(s) > 0$ and $s = -y_{ij}$. The derivative of $g_{ij}(s)$ is

$$g'_{ij}(s) = s'_0(s) + \beta_{ij} - \sigma \left\langle s'_0(s)A_0 + A_{ij}, (W^{-1} - s_0(s)A_0 - sA_{ij})^{-1} \right\rangle, \tag{14}$$

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 in the one dimensional approach, the proof is based on Theorem 4.

Theorem 6

(i) Let the coordinate ij be chosen such that $g'_{ij}(0) > 0$ and $y_{ij} < 0$. The expression (14) has at least one positive solution, and for the smallest such solution 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 (14) has at least one negative solution, and for the biggest such solution s^-, the point $y + s_0(s^-)e_0 + s^- e_{ij}$ is dual feasible and $g'_{ij}(s^-) = 0$.

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

$$ij \in \arg\max_{ij} |g'_{ij}(0)| \,, \tag{15}$$

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

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

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 again allows us to restrict the search to at most three candidates per variable. Thus only $4n$ potential coordinate directions need to 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 Section 5, we compare Algorithm CD and its improved version, Algorithm CD2D, experimentally.

3.5 Primal solutions

This section contains an algorithm to compute an approximate solution of Problem (3) using the information given by the dual optimal solution of Problem (4). We will prove that under some additional conditions the approximate primal solution produced is actually the optimal solution, provided that an optimal solution y^* for the dual problem (4) is given. First note that the primal optimal solution $X^* \in S^+_{n+1}$ must satisfy the complementarity condition

$$(Q - \mathcal{A}^\top y^*)X^* = 0 \tag{16}$$

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

$$\begin{cases} \langle A_0, X^* \rangle & = 1, \\ \langle A_{ij}, X^* \rangle & = \beta_{ij} \quad \forall i, j \in \mathcal{A}(y^*), \end{cases} \tag{17}$$

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

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 (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 (17), we have

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

This suggests, instead of solving the system (16) and (17) in order to compute X, to solve the system above 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

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

Then we can extend Y by zeros to obtain a matrix $Z \in S_{n+1}$, and finally compute $X = PZP^\top$. We formulate this procedure in Algorithm 2. In the implementation of the algorithm, we will always consider the smallest eigenvalue of $Q - \mathcal{A}^\top y$ as zero, this means that r is at least one, and there may be more zero eigenvalues considered as zero, depending on the allowed tolerance.

Algorithm 2: Compute approximate solution of (3) from dual solution

Input: optimal solution $y^* \in \mathbb{R}^{m+1}$ of Problem (4)
Output: $X \in S_{n+1}$
1 Compute $P \in \mathbb{R}^{(n+1)\times(n+1)}$ orthogonal and $\lambda \geq 0$ with $Q - \mathcal{A}^\top y^* = P\mathrm{Diag}(\lambda)P^\top$
2 Find a solution $Y \in S_{n+1-r}$ of the system of equations (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

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.

Theorem 7 *Let y^* be a feasible solution of* (4) *and $X^* \in S_{n+1}$ the corresponding matrix produced by Algorithm 2. If $X^* \succeq 0$, then (X^*, y^*) are primal-dual optimal solutions of Problems* (3) *and* (4).

Proof Let X^* be produced by Algorithm 2 such that it is positive semidefinite. We have that X^* is a feasible solution of Problem (3), since it satisfies the set of active constraints for the optimal dual solution y^*:

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

for all $ij \in \mathcal{A}(y^*)$, this holds since $Y \in S_{n+1-r}$ is the solution of the system of equations (18). It also satisfies complementarity slackness:

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

where the last equation holds since Z is computed as in Step 3 of Algorithm 2. Namely, if $\lambda_l = 0$, then the corresponding row l of $\mathrm{Diag}(\lambda)Z$ is equal to zero. The other rows of $\mathrm{Diag}(\lambda)Z$ are equal to zero from the definition of Z. □

Corollary 8 *Let y^* be a feasible solution of the dual problem (4). If the system*

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

has a unique solution, then Algorithm 2 produces that 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 [1], it was proved that dual non-degeneracy in semidefinite programming implies the existence of a unique optimal primal solution. Additionally, it was proved that dual non-degeneracy is a generic property. Putting these two facts together, it means that for randomly generated instances the probability of obtaining a unique optimal primal solution is one. From the practical point of view, we have implemented Algorithm 2 and run experiments to check the positive semidefiniteness of the computed matrix X. We will see that for the random instances considered in Section 5 this approach works very well in practice.

4 Adding linear constraints

Many optimization problems, such as the quadratic knapsack problem [26, 20], can be modeled as a quadratic problem with linear constraints. Linear constraints can be easily included into the current setting of our problem. Consider the following extension of Problem (IQP),

$$\begin{aligned}\min \quad & x^\top \hat{Q} x + \hat{l}^\top x + \hat{c}\\ \text{s.t.} \quad & a_j^\top x \le b_j \quad \forall j = 1,\dots,p\\ & x \in D_1 \times \cdots \times D_n\,.\end{aligned} \tag{18}$$

Notice that the linear constraint $a_j^\top x \leq b_j$ 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

$$A_j = \begin{pmatrix} \beta_j - b_j & \frac{a_{j_0}}{2} & \dots & \frac{a_{j_{n-1}}}{2} \\ \frac{a_{j_0}}{2} & 0 & \dots & 0 \\ \vdots & & \ddots & \\ \frac{a_{j_{n-1}}}{2} & 0 & \dots & 0 \end{pmatrix}.$$

Following a similar procedure as the one described in Section 2.1, we can formulate a semidefinite relaxation of Problem (18) as follows

$$\begin{array}{lll} \min & \langle Q, X \rangle & \\ \text{s.t.} & \langle A_0, X \rangle = 1 & \\ & \langle A_{ij}, X \rangle \leq \beta_{ij} & \forall j = l_i, \dots, u_i \quad \forall i = 1, \dots, n \\ & \langle A_j, X \rangle \leq \beta_j & \forall j = 1, \dots, p \\ & X \succeq 0. & \end{array} \tag{19}$$

The matrices Q, A_0 and A_{ij} are defined as in Section 2.2. Observe that the new constraint matrices A_j have rank two. The dual of Problem (19) can be calculated as

$$\begin{array}{lll} \max & \langle b, y \rangle & \\ \text{s.t.} & Q - \mathcal{A}^\top y \succeq 0 & \\ & y_0 \in \mathbb{R} & \\ & y_{ij} \leq 0 & \forall j = l_i, \dots, u_i \quad \forall i = 1, \dots, n \\ & y_j \leq 0 & \forall j = 1, \dots, p, \end{array} \tag{20}$$

where $\mathcal{A}$ and b are extended in the obvious way. Again, we want to solve the log-det form of Problem (20)

$$\begin{array}{lll} \max & f(y; \sigma) := \langle b, y \rangle + \sigma \log\det(Q - \mathcal{A}^\top y) & \\ \text{s.t.} & Q - \mathcal{A}^\top y \succ 0 & \\ & y_0 \in \mathbb{R} & \\ & y_{ij} \leq 0 & \forall j = l_i, \dots, u_i \quad \forall i = 1, \dots, n \\ & y_j \leq 0 & \forall j = 1, \dots, p. \end{array} \tag{21}$$

Notice that the overall 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 non-positivity constraint. Even more, the dual problem (20) remains strictly feasible, this fact can be easily derived from Theorem 2.

Corollary 9 *Problem* (20) *is strictly feasible.*

If also the primal problem (19) is strictly feasible, we can show as before that the level sets in our coordinate ascent method are bounded and that we can always find a feasible step length. However, due to the addition of linear constraints, primal strict feasibility might no longer be satisfied. However, by Corollary 9 strong duality holds. In particular, we obtain

Corollary 10 *If the primal problem* (19) *is infeasible, then Problem* (21) *is unbounded.*

Proof From Corollary 9 strong duality holds, hence Problem (20) is unbounded due to primal infeasibility. Thus, we can find an unbounded ray $y^0 + sy$, $s \geq 0$, starting at a strictly feasible solution y^0. Now consider the concave function $h(s) = \lambda_{\min}(Q - \mathcal{A}^\top(y^0 + sy))$. If there exists $s' > 0$ such that $h(s') < h(0)$, then by concavity $h(s) \to -\infty$ for $s \to \infty$ which is a contradiction to the feasibility of the ray. Thus $h(s) \geq h(0) = \lambda_{\min}(Q - \mathcal{A}^\top y^0) > 0$ for all $s \geq 0$. Hence, $\log\det(Q - \mathcal{A}^\top(y^0 + sy))$ is bounded from below so that the objective function of (21) goes to infinity. □

The proof of Corollary 10 shows how to adapt the coordinate search in this case: either an appropriate root such as in Theorem 5 or Theorem 6 exists, which can be used to determine the step length, or we have proven primal infeasibility.

In case Problem (19) is feasible but not strictly feasible, the barrier approach fails. In this case, Problem (21) may be unbounded and hence the algorithm wrongly concludes primal infeasibility.

5 Experiments

We now present the results of an experimental evaluation of our approach. 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 [2] 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 calculations, as explained in Section 3.

For our experiments, we have generated random instances in the same way as proposed in [11]. We will consider two types of variable domains: for *ternary* instances, we have $D_i = \{-1, 0, 1\}$, while for *integer* instances we set $D_i = \{-10, \ldots, 10\}$, for all i.

In our implementation, we use the following rule to update the barrier 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.

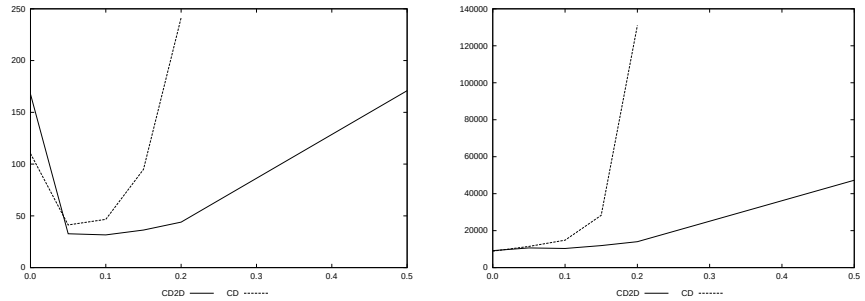
Recall that in Section 2.2, the parameter β_{ij} can be chosen arbitrarily. As it was pointed out, this parameter does not change the feasible region of the primal problem (3), however it does have an influence on its dual problem. We have tested several choices of β_{ij}, such as setting it to zero for all the constraints, or, according to Lemma 2, 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, see Section 2.2.

5.1 Stopping criterion

It is important to find a good stopping criterion that either may allow an early pruning of the nodes as soon as the current upper bound is reached, or stops the algorithm when it cannot be expected any more to reach this bound. Our approach has the advantage of producing feasible solutions of Problem (4) and thus a valid lower bound for Problem (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).

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 $(1 - \text{GAP})$*old-gap* $<$ *new-gap* and the number of iterations is at least $|D_i| \cdot n$, or *new-gap* $<$ 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 4 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 horizontal axis corresponds to different values of GAP, while the vertical axis corresponds to the average running time (Figure 4 (a)) and the average number of nodes (Figure 4 (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 choosing GAP=0.1 produces 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.



(a) Running time (b) Number of nodes

Fig. 4 Influence of the gap criterion on the running time and the number of nodes for ternary instances, the behavior for integer instances is similar.

5.2 Total running time

Next, 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. Furthermore, we compare to other non-convex integer programming software: COUENNE [3] and BARON [28].

In the following tables, n in the first column represents the number of variables. For each approach, we report the number of solved instances ($\#$), the average number of nodes explored in the branch-and-bound scheme (*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 1 we present the results for ternary instances. As it can be observed, Q-MIST manages to solve all 110 instances for $n \leq 50$ with all three approaches. 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 with CSDP. Note that BARON and COUENNE solved all 110 instances only for $n \leq 20$ and $n \leq 30$, respectively.

Table 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 [11], 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 [22] for the comparison, but we do not report the results observed since they are not better than those obtained with COUENNE.

Table 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 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

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. However, 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 obvious that the change of the domains affected the performance of our approach significantly, especially in Algorithm CD.

5.3 Primal solution

At the root node, we have performed the evaluation of Algorithm 2, designed to compute an approximate primal solution of Problem (3) using the dual feasible solution y^* of Problem (4); see the details in Section 3.5. 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 or equal to 0.01. We have run experiments to check the positive semidefiniteness of the matrix X^* at the root node of the branch-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 previous section. 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.

5.4 Behavior with linear constraints

In Section 4 we have described how our approach can be extended when inequality constraints are added to Problem (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 $a^\top x \leq b$. 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 3 and 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 1 with those of Tables 3 and 4, one can conclude that the addition of a linear constraint does not change the overall behavior of our approach. As it can be seen, Q-MIST – with both approaches CD2D and CSDP – outperforms BARON. However, Algorithm CD2D, as shown in Table 1, is much faster even if the number of nodes explored is larger.

Table 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 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

6 Conclusion

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.

References

1. Alizadeh, F., Haeberly, J.P., Overton, M.: Complementarity and nondegeneracy in semidefinite programming. Mathematical Programming **77**(1), 111–128 (1997)
2. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, third edn. Society for Industrial and Applied Mathematics, Philadelphia, PA (1999)
3. Belotti, P., Lee, J., Liberti, L., Margot, F., Wächter, A.: Branching and bounds tightening techniques for non-convex MINLP. Optimization Methods and Software **24**(4-5), 597–634 (2009)
4. Boas, P.V.E.: Another NP-complete problem and the complexity of computing short vectors in a lattice. Tech. rep., University of Amsterdam, Department of Mathematics, Amsterdam (1981)
5. Bonami, P., Biegler, L.T., Conn, A.R., Cornuéjols, G., Grossmann, I.E., Laird, C.D., Lee, J., Lodi, A., Margot, F., Sawaya, N., Wächter, A.: An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optimization **5**, 186–204 (2008)
6. Bonami, P., Gunluk, O., Linderoth, J.: Solving box-constrained nonconvex quadratic programs. Tech. rep., Optimization Online (2016)
7. Buchheim, C., Caprara, A., Lodi, A.: An effective branch-and-bound algorithm for convex quadratic integer programming. Mathematical Programming **135**(1-2), 369–395 (2012)
8. Buchheim, C., De Santis, M., Lucidi, S., Rinaldi, F., Trieu, L.: A feasible active set method with reoptimization for convex quadratic mixed-integer programming. SIAM Journal on Optimization **26**(3), 1695–1714 (2016)
9. Buchheim, C., Hübner, R., Schöbel, A.: Ellipsoid bounds for convex quadratic integer programming. SIAM Journal on Optimization **25**(2), 741–769 (2015)
10. Buchheim, C., Montenegro, M., Wiegele, A.: A coordinate ascent method for solving semidefinite relaxations of non-convex quadratic integer programs. In: ISCO, *Lecture Notes in Computer Science*, vol. 9849, pp. 110–122. Springer (2016)
11. Buchheim, C., Wiegele, A.: Semidefinite relaxations for non-convex quadratic mixed-integer programming. Mathematical Programming **141**(1-2), 435–452 (2013)
12. Burer, S., Letchford, A.: On nonconvex quadratic programming with box constraints. SIAM Journal on Optimization **20**(2), 1073–1089 (2009)
13. Burer, S., Monteiro, R.: A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. Mathematical Programming (Series B) **95**, 2003 (2001)
14. Burer, S., Vandenbussche, D.: Globally solving box-constrained nonconvex quadratic programs with semidefinite-based finite branch-and-bound. Computational Optimization and Applications **43**(2), 181–195 (2009)
15. Dong, H.: Relaxing nonconvex quadratic functions by multiple adaptive diagonal perturbations. SIAM Journal on Optimization **26**(3), 1962–1985 (2016)
16. Grippo, L., Palagi, L., Piccialli, V.: An unconstrained minimization method for solving low-rank SDP relaxations of the maxcut problem. Mathematical Programming **126**(1), 119–146 (2011)
17. Hager, W.: Updating the inverse of a matrix. SIAM Review **31**(2), 221–239 (1989)
18. Helmberg, C.: Semidefinite Programming for Combinatorial Optimization. Professorial dissertation, Technische Univertität Berlin, Berlin (2000)
19. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. SIAM Journal on Optimization **10**(3), 673–696 (2000)
20. Helmberg, C., Rendl, F., Weismantel, R.: A semidefinite programming approach to the quadratic knapsack problem. Journal of Combinatorial Optimization **4**(2), 197–215 (2000)
21. Homer, S., Peinado, M.: Design and performance of parallel and distributed approximation algorithms for maxcut. Journal of Parallel Distributed Computing **46**(1), 48–61 (1997)
22. Misener, R., Floudas, C.A.: ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations. Journal of Global Optimization (2014)

23. Montenegro, M.: A coordinate ascent method for solving semidefinite relaxations of non-convex quadratic integer programs. Phd thesis, Technische Universität Dortmund (2017)
24. Ortega, J., Rheinboldt, W.: Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York (1970)
25. Park, J., Boyd, S.: A semidefinite programming method for integer convex quadratic minimization. Optimization Letters (2017)
26. Pisinger, D.: The quadratic knapsack problem a survey. Discrete Applied Mathematics **155**(5), 623 – 648 (2007)
27. Poljak, S., Rendl, F.: Solving the max-cut problem using eigenvalues. Discrete Applied Mathematics **62**(1), 249 – 278 (1995)
28. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Mathematical Programming **103**, 225–249 (2005)
29. Wolkowicz, H., Saigal, R., Vandenberghe, L.: Handbook of semidefinite programming: theory, algorithms, and applications. International series in operations research & management science. Kluwer Academic, Boston, London (2000)

A Step size for CD

Each constraint matrix A_{ij} can be factored as follows:

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

where $E_{ij} \in \mathbb{R}^{(n+1)\times 2}$ is defined by $E_{ij} := (e_0\ e_i)$, $e_0, e_i \in \mathbb{R}^{n+1}$, $C_{ij} \in \mathbb{R}^{2\times(n+1)}$ is defined by $C := (A_{ij})_{\{0,i\},\{0,\dots,n\}}$, and I is the 2×2-identity matrix. By the Woodbury formula [17]

$$(W^{-1} - sA_{ij})^{-1} = (W^{-1} - sE_{ij}IC_{ij})^{-1} = W + WE_{ij}(\tfrac{1}{s}I - C_{ij}WE_{ij})^{-1}C_{ij}W\ . \quad (22)$$

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 2, we know under which conditions a constraint matrix A_{ij} has rank-one. In that case, we obtain the following factorization:

$$A_{ij} = (A_{ij})_{ii} vv^\top, \quad (23)$$

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,

$$(W^{-1} - sA_{ij})^{-1} = (W^{-1} - s(A_{ij})_{ii}vv^\top)^{-1} = W + \frac{(A_{ij})_{ii}s}{1 - (A_{ij})_{ii}sv^\top Wv} Wvv^\top W. \quad (24)$$

Now, we need to find the value of s that makes the gradient in (10) 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 (22) in the gradient (10) 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 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 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 (24) in the gradient (10) and setting it to zero:

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

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

$$\beta_{ij} - \sigma (A_{ij})_{ii} t - \sigma t^2 \frac{(A_{ij})_{ii}^2 s}{1 - (A_{ij})_{ii} t s} = 0. \tag{25}$$

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 (25) 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

$$\nabla_s f(y + s e_0; \sigma) = 1 - \sigma \left\langle A_0, (W^{-1} - s A_0)^{-1} \right\rangle.$$

As before, the inverse of $W^{-1} - s A_0$ is computed using the Woodbury formula for rank-one update

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

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

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

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 (23) 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

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

B Adding linear constraints

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.

B.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 3.2, 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 3.2. Therefore, if one of the new possible candidates for coordinate direction $e_j \in \mathbb{R}^{m+p+1}$ for $j \in \{1,\dots,p\}$ has been chosen, we need to compute s such that either

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

or

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

We have that

$$\nabla_s f(y + se_j;\sigma)_j = \beta_j - \sigma \left\langle A_j, (W^{-1} - sA_j)^{-1} \right\rangle. \tag{26}$$

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 4 we know that if there exists s in the correct direction of the line search that makes the gradient (26) zero, then there exists also one on the feasible region. This implies the following result.

Theorem 11

(i) Let the coordinate j be such that $\nabla_y f(y;\sigma)_j > 0$ and $y_j < 0$. If the gradient (26) 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, $y + se_{ij}$ is dual feasible with $\nabla_s f(y + se_j;\sigma) > 0$ 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 (26) 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, $y + se_{ij}$ is dual feasible with $\nabla_s f(y + se_j;\sigma) > 0$ for all $s \le 0$.

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:

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

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 (26), we obtain a rational function of degree two

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

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 (26) 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 19. 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.

B.2 Algorithm CD2D including linear constraints

A two-dimensional update is also possible for solving the dual of Problem (19), 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 3.4. It turns out, in this case, that the computation of the step size is technically less complicated. Lemma 5 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 (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 $g_j(s)$

$$g_j'(s) = s_0'(s) + \beta_j - \sigma \left\langle s_0'(s)A_0 + A_j, (W^{-1} - s_0(s)A_0 - sA_j)^{-1} \right\rangle. \tag{27}$$

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 11, is a direct consequence of Lemma 4.

Theorem 12

(i) Let the coordinate j be such that $g_j'(0) > 0$ and $y_j < 0$. If the derivative (27) 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 $y_j + s^+ > 0$, $y + s_0(-y_j)e_0 - y_je_j$ 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 (27) 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 (27), 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 (27) 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 (27) 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

$$s = \frac{2f - \beta_j w_{00}}{2(f^2 - dw_{00})}. \tag{28}$$

Observe that the step size s is independent on the value of σ, however the step s_0 is still dependent. From Theorem 12 it follows that:

(i) if the coordinate j is such that $g_j'(0) > 0$ and $y_j < 0$, and if the derivative (27) has a positive root, then the step size (28) 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 (27) has a negative root, then the step size (28) 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 3.4, i.e., we will choose the coordinate with the largest absolute value of $g_j'(0)$. Recall that from Section 3.4, we have $4n$ potential coordinates, after adding p linear constraints we will have that $4n + p$ candidates to be considered.