Mathematical Programming manuscript No.
(will be inserted by the editor)

On the bridge between combinatorial optimization and nonlinear optimization: a family of semidefinite bounds for 0-1 quadratic problems leading to quasi-Newton methods

Jérôme Malick · **Frédéric Roupin**

the date of receipt and acceptance should be inserted later

This paper is dedicated to Claude Lemaréchal on the occasion of his 65th birthday. We take this opportunity to thank him deeply for the great moments we have had discussing with him (not only about math). His vision and his ability to put ideas into words has helped us deepen our understanding of optimization. This work builds on one of his lines of research: using convex analysis and nonlinear optimization for combinatorial optimization.

Abstract This article presents a family of semidefinite programming bounds, obtained by Lagrangian duality, for 0-1 quadratic optimization problems with linear or quadratic constraints. These bounds have useful computational properties: they have a good ratio of tightness to computing time, they can be optimized by a quasi-Newton method, and their final tightness level is controlled by a real parameter. These properties are illustrated on three standard combinatorial optimization problems: unconstrained 0-1 quadratic optimization, heaviest k-subgraph, and graph bisection.

Keywords Lagrangian duality, combinatorial optimization, 0-1 quadratic programming, nonlinear programming, semidefinite programming, branch-and-bound, quasi-Newton

1 Introduction, motivations

1.1 On the bridge between combinatorial optimization and nonlinear optimization

Along with cutting and branching, bounding is one of the fundamental paradigms of combinatorial optimization. As advocated by Claude Lemaréchal (see e.g. the review [Lem01]), Lagrangian duality (or Lagrangian relaxation) is the essential methodology to create bounds. The duality opens the door towards convex optimization, revealing a fundamental connection between nonlinear optimization and combinatorial optimization. This paper fits into these themes that have been some of Claude's recent lines of research. In this paper, we

J. Malick
CNRS, Lab. Jean Kuntzmann, Grenoble (France)
E-mail: jerome.malick@inria.fr

F. Roupin
LIPN - CNRS : UMR7030 - Université Paris 13 (France)
E-mail: Frederic.Roupin@lipn.univ-paris13.fr

study new bounds for combinatorial optimization, obtained by Lagrangian duality and combined with nonlinear optimization. To some extent, these bounds generalize and enhance the usual semidefinite programming bounds.

Semidefinite programming (SDP) is a prolific branch of convex optimization (see handbooks [SVW00] and [AL12]). Since the '90s, and in particular since the celebrated paper [GW95] about max-cut, semidefinite programming has become a fundamental bounding technique for combinatorial optimization. The recipe to formulate SDP relaxations of combinatorial problems [PRW95] has an illuminating interpretation by Lagrangian duality, as explained by Claude Lemaréchal in the pedagogic paper [LO99]. Recent research has spawned a focus on the power and the limitations of semidefinite programming for solving combinatorial optimization problems. The current sentiment is, roughly speaking, that semidefinite programming bounds have good tightness, but demand much time to be computed, which often prevents their direct use in branch-and-bound algorithms or other exact resolution schemes. To answer this challenge, a strategy is to use the problem structure, for example by exploiting sparsity (see e.g. [KKW09]), by reducing the size of problems by using hidden symmetries (see e.g. [dKS10]), by developing new algorithmic schemes (see e.g. [MPRW09]), or by adapting algorithms to special problem classes (see e.g. [RRW10]).

In this paper, we use a different approach: we do not consider the standard SDP bounds; rather we construct new SDP-based bounds that are less tight (they deteriorate the usual SDP bounds) but are easier to compute (they are amenable to efficient standard nonlinear optimization techniques). The influence of Claude Lemaréchal is visible from the basic approach (Lagrangian duality [Lem01]) to the practical resolution (limited memory quasi-Newton method [GL89]); we also extensively use convex analysis [HUL93].

1.2 A family of SDP bounds for 0-1 quadratic problems leading to a quasi-Newton method

This article presents a family of semidefinite bounds for binary quadratic problems subject to linear or quadratic constraints, that is, a general problem of the form

$$\begin{cases} \min \quad x^\top M_0 x + {m_0}^\top x \\ \quad x^\top M_j x + {m_j}^\top x + c_j = (\text{or} \leq) \, 0 \quad j = 1, \ldots, m \\ \quad x \in \{0,1\}^n \end{cases} \tag{1}$$

with $m_j \in \mathbb{R}^n$, $c_j \in \mathbb{R}$ and symmetric matrices M_j. This general form contains for example 0-1 quadratic problems with linear constraints (when $M_j = 0$), and 0-1 linear problems (when moreover $M_0 = 0$), that are in particular models for various problems in operations research. Notice that the equality and inequality constraints in (1) can be both constraints arising when modeling the problem, or redundant constraints added to strengthen the bounds. To illustrate our approach, we consider in this paper three combinatorial optimization problems of the form (1): unconstrained 0-1 quadratic programming problems, heaviest subgraph problems, and graph bisection problems (see details and references in Section 2.2).

The objective of this article is to develop bounding techniques for (1), trading computing time for a deterioration of the quality of the usual semidefinite bounds, in order to enhance the efficiency of exact resolution schemes. We introduce a new family of SDP-based bounds constructed by duality after handily rewriting the rank-one constraint appearing in the SDP reformulation of the problems (see Section 2.4). The computation of these bounds relies on robust tools of linear algebra and nonlinear optimization (more precisely, computing the bounds amounts to an eigendecomposition of a matrix, and improving the tightness can be done with Newton-like algorithms; see Section 4.3). The bounds turn out to have good ratios

of tightness to computing time (see Section 5), which is a key property in view of being used as a bounding procedure within branch-and-bound methods to solve 0-1 quadratic problems to optimality. Another important property is that we have a real parameter (α in the sequel) that controls the tightness level and the computing cost of these bounds.

Some of the underlying ideas in this work are not completely new: they are present in [Mal07], and used in the recent paper [MR11] to solve to optimality a special hard combinatorial optimization problem. The main goals of the present paper are the following:

- to clarify and simplify the approach by introducing a family of semidefinite bounds with direct interpretation from Lagrangian duality;
- to give a general methodology to construct and to use these semidefinite bounds for general 0-1 quadratic programming;
- to make a numerical study of these bounds: illustrating their features, comparing them with the usual bounds (in terms of tightness, computing time, and the balance of both), and advocating that they have interesting qualities in view of being embedded within branch-and-bound algorithms.

Here is the outline of the paper. In Section 2, we introduce the notation, three problems of the form (1), and transformations to come up with an equivalent SDP formulation of the problem (see forthcoming equation (15)). In Section 3, we apply the mechanism of Lagrangian duality to the reformulation, which gives us the new bounds. We study in Section 4 the properties of this family of bounds and discuss how to use them in practice. Section 5 reports numerical tests of these bounds on various standard test-problems.

2 Semidefinite reformulations of 0-1 quadratic problems

2.1 Notation, projections in symmetric matrix spaces

Let $\mathscr{S}_p$ be the space of $p \times p$ symmetric matrices, equipped with the norm $\|\cdot\|$ associated with its usual inner product

$$\langle A, B\rangle := \textstyle\sum_{i,j} A_{ij}B_{ij} = \operatorname{trace}(AB).$$

This inner product is relevant for our purposes through the relation $x^\top A x = \langle A, xx^\top\rangle$ for all $x \in \mathbb{R}^p$ and $A \in \mathscr{S}_p$. In other words, one can transform quadratic functions on $\mathbb{R}^p$ to linear functions on $\mathscr{S}_p$ with respect to the rank-one symmetric matrix $X = xx^\top$. This will be used to obtain the forthcoming SDP formulation (15). Note also that the binary constraint $x_i \in \{-1, 1\}$ reads $X_{ii} = 1$ on $X = xx^\top$. Introducing the operator $\operatorname{diag}\colon \mathscr{S}_p \to \mathbb{R}^p$ and the vector of all ones $e \in \mathbb{R}^p$, this gives us the constraint $\operatorname{diag}(X) = e$ that appears in (15). For more details on SDP formulations of combinatorial problems, see [PRW95] and [LO99].

The projections (with respect to $\|\cdot\|$) of a matrix $A \in \mathscr{S}_n$ onto the closed convex cone $\mathscr{S}_p^+$ and its polar cone $\mathscr{S}_p^-$ are denoted, respectively, by

$$A_+ := \operatorname*{argmin}_{X \succeq 0} \|X - A\| \qquad \text{and} \qquad A_- := \operatorname*{argmin}_{X \preceq 0} \|X - A\|.$$

Moreau's theorem (see [HUL93, Chap. III]) states that the projections are characterized by

$$\langle A_+, A_-\rangle = 0 \qquad \text{and} \qquad A = A_+ + A_- \,. \tag{2}$$

In fact, it is well-known (see e.g. [Hig88]) that we have an explicit formula for these projections as follows. Let $A = \sum_i \lambda_i p_i p_i^\top$ be a spectral decomposition of A with eigenvalues λ_i

and eigenvectors p_i, assumed to be pairwise orthogonal and of length one; then

$$A_+ = \sum_{\lambda_i>0} \lambda_i p_i p_i^\top \qquad \text{and} \qquad A_- = \sum_{\lambda_i<0} \lambda_i p_i p_i^\top.$$

Note finally that we have for $A \in \mathscr{S}_p$

$$(-A)_- = -A_+ \,. \tag{3}$$

2.2 Examples of binary quadratic optimization problems

The developments of this paper apply to general binary quadratic problems of the form (1). We illustrate them on the following three classical combinatorial optimization problems.

Example 1 (Unconstrained quadratic 0-1 problem) Consider the problem of minimizing a quadratic function with binary variables

$$\begin{cases} \min \quad x^\top M_0 x + m_0{}^\top x \\ \quad x \in \{0,1\}^n. \end{cases} \tag{4}$$

This problem is known to be NP-hard in general, and to have numerous applications (see e.g. an application in medical field in [PIS+04]). We also mention the celebrated max-cut problem on a (undirected) graph (see e.g. [GW95]) which can be formulated as

$$\begin{cases} \min \quad x^\top L x \\ \quad x \in \{-1,1\}^n \end{cases} \tag{5}$$

with L the Laplacian matrix of the graph. It is well-known that these two problems (4) and (5) are equivalent. In particular, (4) can be reformulated as a max-cut on a graph with unrestricted weights. □

Example 2 (Heaviest subgraph) Consider an undirected weighted graph $G = (V,E)$ with n vertices, and denote by $W = (w_{ij})_{ij}$ its weight-matrix (which is a symmetric $n \times n$ matrix). For an integer k in $\{1,\dots,n\}$, the heaviest k-subgraph problem (also called k-cluster problem; see references in [MR11]) consists in determining a subset S of k vertices such that the total edge weight of the subgraph induced by S is maximized. This problem is directly modeled in the form (1)

$$\begin{cases} \max \quad x^\top W x \\ \quad e^\top x = k, \quad x \in \{0,1\}^n. \end{cases} \tag{6}$$

In view of having better bounds when relaxing, we introduce the n so-called "product constraints" (standard reinforcement constraints, see e.g. [SA90], [LS91])

$$(e^\top x - k)\, x_i = 0, \quad \text{for } i = 1,\dots,n. \tag{7}$$

In some sense, adding these n equality constraints is optimal (see e.g. [LO99],[FR07]). This leads again to a problem of the form (1)

$$\begin{cases} \max \quad x^\top W x \\ \quad e^\top x = k, \quad x \in \{0,1\}^n, \\ \quad x^\top A_i x - k x_i = 0, \quad \text{for } i = 1,\dots,n \end{cases} \tag{8}$$

with the appropriate matrices $A_i \in \mathscr{S}_n$ (see [MR11, Eq. (3)] for more details). □

Example 3 (Graph bisection) With the notation of the previous example, the graph bisection problem consists in dividing the nodes of G into two subsets of k and $n-k$ vertices such that the total weights of edges that have end-points in different sets is minimal. This problem can be formulated as

$$\begin{cases} \min \quad x^\top W x \\ \quad e^\top x = 2k-n, \quad x \in \{-1,1\}^n. \end{cases} \tag{9}$$

Similarly to heaviest subgraph problem (8), the constraints can be strengthened as

$$\begin{cases} \min \quad x^\top W x \\ \quad e^\top x = 2k-n, \quad x \in \{-1,1\}^n, \\ \quad x^\top A_i x - (2k-n)x_i = 0, \quad \text{for } i = 1,\dots,n. \end{cases} \tag{10}$$

Moreover this problem can be obviously written as (1) after a change of variables. SDP techniques have been used for solving this problem to optimality (see [KRC00]). □

2.3 Reformulation with a rank-one constraint (semidefinite lifting)

We apply the standard lifting up to the symmetric matrix space by equivalently writing the 0-1 quadratic problem (1) as a linear SDP problem with a rank-one constraint (see more in [SVW00]). Though all the development can be made with $\{0,1\}$-variables, it is simpler to work with $\{-1,1\}$ variables when introducing the "spherical constraint" (see next section). There are two equivalent ways to proceed when changing variables and lifting to SDP:

1. *Lift and change*. First, apply the "recipe" (see e.g. [PRW95] or [LO99]) to transform the 0-1 quadratic problem (1) to a SDP problem with respect to

$$Y = \begin{bmatrix} 1 & x^\top \\ x & X \end{bmatrix} \in \mathscr{S}_{n+1}.$$

   Second, apply the change of variables $X \leftarrow UYU^\top$ given by the block-matrices

$$U := \begin{bmatrix} 1 & 0 \\ \frac{1}{2}e & \frac{1}{2}I_n \end{bmatrix} \qquad \text{and} \qquad U^{-1} = \begin{bmatrix} 1 & 0 \\ -e & 2I_n \end{bmatrix}.$$

2. *Change and lift*. We can also do it the other way around. First, apply (if necessary) the change of variable $\{0,1\} \to \{-1,1\}$ to (1). In fact, the transformation $[1;x] = U[1;y]$ with the above U can be used to make the change of variables and to homogenize the quadratic forms $q_j(x) = x^\top A_j x + b_j{}^\top x + c_j$ at the same time. Second, lift the problem to the matrix space with a semidefinite rank-one matrix of the form $X = xx^\top \in \mathscr{S}_{n+1}$.

The two above transformations are known to be equivalent (see e.g. [HPRW95], [Hel00]). Note that there are several degrees of freedom in the reformulation (although all the operations can be done automatically). For example, the constants c_j can be treated in different ways. The formulation choices may have an impact on the behavior of solvers.

Finally we get the following linear SDP problem with respect to $X \in \mathscr{S}_{n+1}$

$$\begin{cases} \min \quad \langle Q_0, X\rangle \\ \quad \langle Q_j, X\rangle = (\text{or} \le)\, q_j \quad j = 1,\dots,m \\ \quad \text{diag}(X) = e,\ X \succeq 0,\ \text{rank}(X) = 1 \end{cases} \tag{11}$$

with $\gamma_j, q_j \in \mathbb{R}$ such that $\gamma_j - q_j = c_j$, and with the $(n+1)\times(n+1)$-matrices Q_j defined by

$$Q_j := U^\top \begin{bmatrix} \gamma_j & \frac{1}{2}m_j{}^\top \\ \frac{1}{2}m_j & M_j \end{bmatrix} U.$$

For example, problem (4) has the equivalent SDP formulation

$$\left\{\begin{array}{l}\min \quad \langle Q,X\rangle \\ \quad \mathrm{diag}(X)=e \\ \quad X\succeq 0,\ \mathrm{rank}(X)=1\end{array}\right. \qquad \text{with} \quad Q:=\tfrac{1}{4}\begin{bmatrix} e^\top M_0 e+2m_0{}^\top e & e^\top M_0+m_0{}^\top \\ M_0e+m_0 & M_0\end{bmatrix}. \tag{12}$$

The combinatorial difficulty of the initial problem (1) is now concentrated in the rank-one constraint of (11). The SDP relaxation of this problem consists of dropping the rank-one constraint; the resulting problem is thus the linear SDP problem

$$\left\{\begin{array}{l}\min \quad \langle Q_0,X\rangle \\ \quad \langle Q_j,X\rangle = (\text{or}\le)\, q_j \quad j=1,\dots,m \\ \quad \mathrm{diag}(X)=e,\ X\succeq 0.\end{array}\right. \tag{13}$$

In this paper, we do not drop the rank constraint, and we follow instead the idea of [Mal07] to write this constraint in a more handy way, as a norm constraint.

2.4 Reformulation of the rank-one constraint (as a spherical constraint)

An interesting property [Mal07, Th. 1] is that, for $X\succeq 0$ with $\mathrm{diag}(X)=e$, we have

$$\|X\|\le n+1 \quad \text{and moreover} \quad \big(\|X\|=n+1 \iff \mathrm{rank}\,X=1\big). \tag{14}$$

Therefore we can replace the rank-one constraint in formulation (11) by the equivalent constraint $\|X\|^2=(n+1)^2$, called the "spherical constraint" in [Mal07]. Our problem (1) can thus be equivalently written as:

$$\left\{\begin{array}{l}\min \quad \langle Q_0,X\rangle \\ \quad \langle Q_j,X\rangle = (\text{or}\le)\, q_j \quad j=1,\dots,m \\ \quad \mathrm{diag}(X)=e,\ X\succeq 0,\ \|X\|^2=(n+1)^2.\end{array}\right. \tag{15}$$

In other words, we have transformed a quadratic problem in $\mathbb{R}^{n+1}$ with quadratic constraints into a linear SDP problem with one single quadratic constraint. This quadratic constraint carries the nonconvexity and the combinatorial difficulty of the initial problem. Obviously, dropping it take us back to the SDP relaxation (13).

3 Semidefinite bounds by Lagrangian duality

In this section, we attack the last reformulation (15) of the quadratic binary problem (1), by Lagrangian duality. We will apply, simply and directly, the usual duality mechanism (see e.g. [BV04, Chap. 5] and [HUL93, Chap. XII]). To do so, we separate the m affine constraints of (15) into m_I inequalities and m_E equalities (with $m_I+m_E=m$, and $m_E\ge n$ since there are already the n constraints $X_{ii}=1$). So we work from now on with

$$\left\{\begin{array}{l}\min \quad \langle Q,X\rangle \\ \quad \langle A_i,X\rangle \le a_i, \quad i=1,\dots,m_I \\ \quad \langle B_i,X\rangle = b_i, \quad i=1,\dots,m_E \\ \quad X\succeq 0,\ \|X\|^2=(n+1)^2\end{array}\right. \tag{16}$$

which is (15) with new notation. We emphasize that there has not been any relaxation so far; in other words, the optimal value of (16) is equal to the one of the original problem (1):

$$\mathrm{val}(16) = \mathrm{val}(1).$$

We introduce the dual space $\mathscr{D} := \mathbb{R}^{m_I}_+ \times \mathbb{R}^{m_E} \times \mathscr{S}^-_{n+1} \times \mathbb{R}$, and define the Lagrangian, a function of the primal variable $X \in \mathscr{S}_{n+1}$ and the dual variables $(\lambda, \mu, Z, \alpha) \in \mathscr{D}$

$$\begin{aligned} L(X;\lambda,\mu,Z,\alpha) = \langle Q, X\rangle &+ \sum_{i=1}^{m_I} \lambda_i(\langle A_i, X\rangle - a_i) + \sum_{i=1}^{m_E} \mu_i(\langle B_i, X\rangle - b_i) \\ &+ \frac{\alpha}{2}(\|X\|^2 - (n+1)^2) + \langle Z, X\rangle. \end{aligned} \tag{17}$$

The associated dual function is the concave function defined as

$$\theta(\lambda,\mu,Z,\alpha) := \inf_{X \in \mathscr{S}_{n+1}} L(X;\lambda,\mu,Z,\alpha). \tag{18}$$

We already know by the weak duality result ([HUL93, XII.2.1.5]), that each value of θ gives a lower bound on the optimal value of (16); in other words, we have for all $(\lambda, \mu, Z, \alpha) \in \mathscr{D}$

$$\theta(\lambda,\mu,Z,\alpha) \leq \mathrm{val}(16). \tag{19}$$

The best of these bounds is the optimal value of the associated dual problem

$$\sup_{(\lambda,\mu,Z,\alpha)\in\mathscr{D}} \theta(\lambda,\mu,Z,\alpha). \tag{20}$$

So far, we have just given definitions and general properties; the approach has a real interest only if we can compute (some of) these bounds. In general, solving the optimization problem (18) - and therefore (20) a fortiori - is not easy as it has a nonlinear objective function and a semidefinite conic constraint. In fact, the computability of this problem essentially relies on the sign of α, as follows.

First of all, when $\alpha < 0$, observe that $\theta(\lambda, \mu, Z, \alpha) = -\infty$ so that this "bound" is useless. Second, the case when $\alpha = 0$ deserves a special treatment. Indeed note that the Lagrangian $L(X;\lambda,\mu,Z,0)$ corresponds to the standard Lagrangian for linear SDP programming. Thus we have an explicit expression of $\theta(\lambda,\mu,Z,0)$ that gives us the usual linear SDP dual problem (which corresponds to the dual of (13))

$$\begin{cases} \max & -a^\top\lambda - b^\top\mu \\ & A(\lambda) + B(\mu) + Q + Z = 0 \\ & Z \preceq 0 \end{cases} \tag{21}$$

where we set

$$A(\lambda) = \sum_{i=1}^{m_I} \lambda_i A_i \qquad \text{and} \qquad B(\mu) = \sum_{i=1}^{m_E} \mu_i B_i.$$

In other words, the best of the bounds $\theta(\lambda,\mu,Z,0)$ for the dual variables (λ,μ,Z) is simply the usual SDP bound of (1). There exist many efficient algorithms and software to compute this SDP bound (see e.g. [HRVW96], [HR00]). In our numerical experiments, we use the two software packages [Bor99] and [Hel04] known to be among the best ones.

Finally, problem (18) with $\alpha > 0$ can be solved explicitly, as shown by the next two results. We denote the positive real numbers by $\mathbb{R}_{++}$.

Theorem 1 (Dual function) *Given dual variables $(\lambda,\mu,Z,\alpha)\in\mathscr{D}$ with $\alpha>0$, the dual function can be written*

$$\theta(\lambda,\mu,Z,\alpha)=-a^\top\lambda-b^\top\mu-\frac{1}{2\alpha}\|Q+A(\lambda)+B(\mu)+Z\|^2-\frac{\alpha}{2}(n+1)^2.$$

This expression is differentiable with respect to $(\lambda,\mu,Z,\alpha)\in\mathbb{R}^{m_I}\times\mathbb{R}^{m_E}\times\mathscr{S}_{n+1}\times\mathbb{R}_{++}$, and we know the partial differentials; in particular:

$$\begin{aligned}\partial_{\lambda_i}\theta(\lambda,\mu,Z,\alpha)&=-\frac{1}{\alpha}\langle A_i,Q+A(\lambda)+B(\mu)+Z\rangle-a_i\\ \partial_{\mu_i}\theta(\lambda,\mu,Z,\alpha)&=-\frac{1}{\alpha}\langle B_i,Q+A(\lambda)+B(\mu)+Z\rangle-b_i.\end{aligned}$$

Proof Note that the Lagrangian (17) can be written as

$$L(X;\lambda,\mu,Z,\alpha)=\alpha\|X\|^2/2+\langle X,Q_1(\lambda,\mu,Z)\rangle+q(\lambda,\mu,\alpha)$$

with the help of

$$Q_1(\lambda,\mu,Z):=Q+A(\lambda)+B(\mu)+Z,\quad\text{and}\quad q(\lambda,\mu,\alpha):=-\frac{\alpha}{2}(n+1)^2-a^\top\lambda-b^\top\mu.$$

As a function of X, the Lagrangian is obviously strongly convex and differentiable; thus it admits a unique minimizer characterized by $0=\nabla_X L(X;\lambda,\mu,Z,\alpha)=\alpha X+Q_1(\lambda,\mu,Z)$. This gives

$$\theta(\lambda,\mu,Z,\alpha)=-\frac{1}{2\alpha}\|Q_1(\lambda,\mu,Z)\|^2+q(\lambda,\mu,\alpha)\tag{22}$$

which is the form we wanted. This function is clearly differentiable with respect to all the dual variables; the expressions of the differentials follow from standard calculus rules. □

Theorem 2 (Simplified dual function) *Given dual variables (λ,μ,α) with $\alpha>0$, the dual function can be maximized over Z: the simplified dual function is*

$$\begin{aligned}\Theta(\lambda,\mu,\alpha)&:=\max_{Z\preceq 0}\ \theta(\lambda,\mu,Z,\alpha)\\ &=-a^\top\lambda-b^\top\mu-\frac{1}{2\alpha}\left\|(Q+A(\lambda)+B(\mu))_-\right\|^2-\frac{\alpha}{2}(n+1)^2.\end{aligned}$$

This expression is differentiable at any $(\lambda,\mu,\alpha)\in\mathbb{R}^{m_I}\times\mathbb{R}^{m_E}\times\mathbb{R}_{++}$, and we have

$$\begin{aligned}\partial_{\lambda_i}\Theta(\lambda,\mu,\alpha)&=-\frac{1}{\alpha}\langle A_i,(Q+A(\lambda)+B(\mu))_-\rangle-a_i\\ \partial_{\mu_i}\Theta(\lambda,\mu,\alpha)&=-\frac{1}{\alpha}\langle B_i,(Q+A(\lambda)+B(\mu))_-\rangle-b_i.\end{aligned}$$

Proof Isolate matrix $Q_2(\lambda,\mu)=Q+A(\lambda)+B(\mu)$ in the expression of $\theta(\lambda,\mu,Z,\alpha)$ of Theorem 1. Observe that the solution of

$$\min_{Z\preceq 0}\|Q_2(\lambda,\mu)+Z\|$$

is the projection $(-Q_2(\lambda,\mu))_-=-Q_2(\lambda,\mu)_+$ by (3). Therefore the minimum value is then $\|-Q_2(\lambda,\mu)_+ +Q_2(\lambda,\mu)\|=\|Q_2(\lambda,\mu)_-\|$ by (2). With the expression of $\theta(\lambda,\mu,Z,\alpha)$ of Theorem 1, this gives the expression of $\Theta(\lambda,\mu,\alpha)$.

To differentiate Θ, we want to use the theorem of differentiability of a max (or min) function (see e.g. [HUL93, VI.4.4.5]) which asks for compactness of the "index set" on which the maximum is taken. Fix any $(\bar\lambda,\bar\mu,\bar\alpha)\in\mathbb{R}^{m_I}\times\mathbb{R}^{m_E}\times\mathbb{R}_{++}$ and consider a closed ball V around them. Observe now that the set $U=-(Q_2(V))_+$ is compact as well (since Q_2 and the projection are both continuous) and that we can write

$$\Theta(\lambda,\mu,\alpha)=\max_{Z\in U\cap\mathscr{S}^-_{n+1}}\theta(\lambda,\mu,Z,\alpha)\qquad\text{for all }(\lambda,\mu,\alpha)\in V.$$

Since the maximum is now taken on a compact set, [HUL93, VI.4.4.5] gives the differentiability around $(\bar\lambda,\bar\mu,\bar\alpha)$ together with the expression of partial differentials.

Let us make the calculation of the partial differential with respect to λ_i explicit. By [HUL93, VI.4.4.5], we have $\partial_{\lambda_i}\Theta(\lambda,\mu,\alpha)=\partial_{\lambda_i}\theta(\lambda,\mu,-Q_2(\lambda,\mu)_+,\alpha)$. The expression of $\partial_{\lambda_i}\theta$ of Theorem 1 yields, with the help of (2)

$$\partial_{\lambda_i}\Theta(\lambda,\mu,\alpha)=-\frac{1}{\alpha}\langle A_i,Q_2(\lambda,\mu)-Q_2(\lambda,\mu)_+\rangle-a_i=\frac{1}{\alpha}\langle A_i,Q_2(\lambda,\mu)_-\rangle-a_i$$

We obtain the other partial differentials the same way. □

This theorem shows that for any dual variables (λ,μ,α) with $\alpha>0$ we have an explicit expression of the dual lower bound $\Theta(\lambda,\mu,\alpha)$. In the remainder of this paper, we study these new bounds – theoretically and numerically.

4 Study of the family of semidefinite bounds

Consider the bounds for the 0-1 quadratic problem (1) formulated as (16), introduced by duality in the previous section; we study in this section the family of bounds

$$\Theta(\lambda,\mu,\alpha)=-a^\top\lambda-b^\top\mu-\frac{1}{2\alpha}\left\|(Q+A(\lambda)+B(\mu))_-\right\|^2-\frac{\alpha}{2}(n+1)^2\qquad(23)$$

with respect to $(\lambda,\mu,\alpha)\in\mathscr{D}_+$ where

$$\mathscr{D}_+:=\mathbb{R}^{m_I}_+\times\mathbb{R}^{m_E}\times\mathbb{R}_{++}.$$

Obviously, these are indeed lower bounds of the optimal value of (1), since in view of (19) and Theorem 2 we have

$$\Theta(\lambda,\mu,\alpha)\le\text{val}(16)=\text{val}(1).\qquad(24)$$

We study in this section some important properties of the bound $\Theta(\lambda,\mu,\alpha)$ (computation cost in Section 4.1, tightness in Section 4.2) and we discuss how to use them in practice within a branch-and-bound procedure (Section 4.3). A numerical study on our three test-problems is done afterwards in Section 5.

4.1 Computing of the bounds

The bounds $\Theta(\lambda,\mu,\alpha)$ are computed by standard linear algebra:

1. The initial quadratic problem is reformulated as (16) by a sequence of elementary operations that could be fully automatized (to be implemented in SDP_S [Rou04]).
2. The main operation is then to assemble the matrix $Q+A(\lambda)+B(\mu)$ and to project it onto the cone of semidefinite negative matrices $\mathscr{S}^-_{n+1}$.

As recalled in Section 2.1, projecting onto $\mathscr{S}^-_{n+1}$ is done by computing an eigendecomposition of the matrix to project. The dominant cost of the computation of $\Theta(\lambda,\mu,\alpha)$ is thus of order $O(n^3)$. This may prevent the use of these bounds for large graphs. Since we aim at developing a bouding technique to be embedded within a branch-and-bound algorithm, this is not a limitation: regarding exact resolution of our three test-problems, medium-size problems ($n \leq 500$) are already challenging instances, to say the least. In this case, the eigendecomposition can be performed quickly by efficient linear algebra routines. Thus, the new bound are not too expensive in view of our targeted use; but, of course, they are of relevance only if they are not too weak. This is studied in the next section.

Before moving to tigthness, let us make the bounds for one of our test-problems explicit for illustration. This will also highlight the connection with [MR11].

Example 4 (Bounds for the heaviest subgraph problem) Particularizing bound (23) for problem (8) yields, for $\mu = (\mu_1,\mu_2) \in \mathbb{R}^{n+1}\times\mathbb{R}^{n+1}$ and $\alpha > 0$,

$$\Theta_{\text{heav}}(\mu,\alpha) := -b^\top\mu - \frac{1}{2\alpha}\|(Q+A_{\text{heav}}(\mu))_-\|^2 - \frac{\alpha}{2}(n+1)^2,$$

where we set

$$Q = U^\top \begin{bmatrix} 0 & 0 \\ 0 & -W \end{bmatrix} U \qquad \text{and} \qquad A_{\text{heav}}(\mu) := \text{Diag}(\mu_1) + U^\top \begin{bmatrix} 0 & \frac{k}{2}\mu_2{}^\top \\ \frac{k}{2}\mu_2 & -\sum_{i=1}^{n+1}(\mu_2)_i A_i \end{bmatrix} U.$$

We note that the above bounds have already appeared in [MR11]: more specifically, these are not the bounds introduced in [MR11, Sec. 3], but they are essentially the same as the ones used in practice, implicitly given by [MR11, Lemma 3] as

$$\Theta'_{\text{heav}}(\mu,\alpha) := \alpha\Big(-b^\top\mu + \frac{1}{2}\|(-Q/\alpha + A_{\text{heav}}(\mu))_+\|^2 - \frac{1}{2}(n+1)^2\Big).$$

Indeed we have $\Theta_{\text{heav}}(\mu,\alpha) = \Theta'_{\text{heav}}(-\mu/\alpha,\alpha)$. In addition to the direct interpretation by duality, one could see a numerical interest in using the formulation Θ_{heav} instead of Θ'_{heav} to compute the bound. To get Θ_{heav} indeed, we compute the eigendecomposition of a matrix whose entries are of the same scale as the data of the problem, whereas the entries of the matrix in Θ'_{heav} are badly scaled when α gets small. □

4.2 How tight are the bounds?

This section shows that the new SDP bounds (23) are always less tight than the usual SDP bounds, but that one can approximate them in arbitrary precision. The first theorem goes one step further than (24) by showing that the new SDP bounds are in fact lower bounds on the usual SDP bound of the initial problem.

Theorem 3 (Less tight bounds) *For any $(\lambda,\mu,\alpha) \in \mathscr{D}_+$, the bound $\Theta(\lambda,\mu,\alpha)$ is less tight than the usual SDP bound:*

$$\Theta(\lambda,\mu,\alpha) \leq \text{val}(13) \leq \text{val}(1).$$

Proof Denote by $\mathscr{C}$ the constraint set of the SDP relaxation (13), that then becomes

$$\begin{cases} \min & \langle Q, X\rangle \\ & X \in \mathscr{C}. \end{cases} \tag{25}$$

Recalling that (16) is just (15) written with other notation, observe that the feasible set of (16) is included in $\mathscr{C}$, so that

$$\mathrm{val}\,(25) \leq \mathrm{val}\,(16) = \mathrm{val}\,(1). \tag{26}$$

Consider now $\alpha > 0$ fixed, and gather together the terms $\langle Q, X\rangle$ and $\alpha(\|X\|^2 - (n+1)^2)/2$ in (17) and (18): we then observe that

$$\begin{cases} \max & \theta(\lambda, \mu, Z, \alpha) \\ & (\lambda, \mu, Z) \in \mathbb{R}^{m_I}_+ \times \mathbb{R}^{m_E} \times \mathscr{S}^-_{n+1}, \end{cases} \tag{27}$$

is the dual problem of

$$\begin{cases} \min & \langle Q, X\rangle + \frac{\alpha}{2}(\|X\|^2 - (n+1)^2) \\ & X \in \mathscr{C}. \end{cases} \tag{28}$$

By weak duality we have $\mathrm{val}\,(27) \leq \mathrm{val}\,(28)$. Since any $X \in \mathscr{C}$ is positive semidefinite with ones on the diagonal, (14) gives $\alpha(\|X\|^2 - (n+1)^2) \leq 0$, so that we have furthermore $\mathrm{val}\,(28) \leq \mathrm{val}\,(25)$. Putting together the inequalities, we have

$$\Theta(\lambda, \mu, \alpha) \leq \mathrm{val}\,(27) \leq \mathrm{val}\,(28) \leq \mathrm{val}\,(25)\big(= \mathrm{val}\,(13)\big),$$

which allows us to conclude with (26). □

The next result somehow compensates the bad news of the previous one: it states that we can get close to the SDP bound with $\Theta(\lambda, \mu, \alpha)$. To prove the result, we need existence of solutions to (21), so we make a Slater-type assumption.

Theorem 4 (Arbitrarily tight bounds) *Set $\Lambda := \mathbb{R}^{m_I}_+ \times \mathbb{R}^{m_E}$, and assume that there exists a matrix $\bar{X} \in \mathscr{S}_{n+1}$ such that*

$$\bar{X} \succ 0, \qquad \langle A_i, \bar{X}\rangle \leq a_i,\ i = 1, \ldots, m_I \qquad \text{and} \qquad \langle B_j, \bar{X}\rangle = b_j,\ j = 1, \ldots, m_E \tag{29}$$

Then the bounds $\Theta(\lambda, \mu, \alpha)$ satisfy the following two properties:

(i) for $0 < \alpha \leq \beta$,

$$\max_{(\lambda,\mu)\in\Lambda} \Theta(\lambda, \mu, \beta) \leq \max_{(\lambda,\mu)\in\Lambda} \Theta(\lambda, \mu, \alpha) \leq \mathrm{val}\,(21);$$

(ii) the upper bound is reached when $\alpha > 0$ vanishes:

$$\lim_{\alpha\to 0} \max_{(\lambda,\mu)\in\Lambda} \Theta(\lambda, \mu, \alpha) = \mathrm{val}\,(21).$$

Proof Using a standard result of convex optimization (apply e.g. [HUL93, VII.2.2.5] with $c_0(X) = \lambda_{\max}(-X)$), the weak Slater assumption (29) gives the existence of dual solutions (i.e. solutions to (21) in our case), and together with existence of primal solution (the primal feasible set is compact), it also gives no duality gap (i.e. $\mathrm{val}\,(21) = \mathrm{val}\,(13)$). We use now the notation of the proof of Theorem 3 and we introduce the function

$$T(\alpha) := \max_{(\lambda,\mu)\in\Lambda} \Theta(\lambda, \mu, \alpha) = \max_{(\lambda,\mu,Z)\in\mathbb{R}^{m_I}_+ \times \mathbb{R}^{m_E} \times \mathscr{S}^-_{n+1}} \theta(\lambda, \mu, Z, \alpha)$$

Note first that the Slater point of (13) is also a Slater point of (28) (that has the same constraint set, denoted $\mathscr{C}$). Therefore, the optimal values of (27) and (28) coincide, in other words, $T(\alpha) = \text{val}(28)$. Look now at (28): the constraint set $\mathscr{C}$ is compact (it is closed by definition and bounded by (14)), and the objective function is strongly convex (since $\alpha > 0$). Therefore, there exists a unique solution $X^* \in \mathscr{C}$, and we can use the theorem of differentiability of a max function (see e.g. [HUL93, VI.4.4.5]): T is differentiable and

$$T'(\alpha) \;=\; (\|X^*\|^2 - (n+1)^2)/2 \;\leq\; 0.$$

The above negativity comes from (14) since matrices lying in $\mathscr{C}$ are in particular positive semidefinite with ones on the diagonal. Thus we have that T is nonincreasing on $\mathbb{R}_{++}$, and this proves (i) (since the last inequality in (i) comes from Theorem 3).

We now turn to (ii). We denote by (λ^*, μ^*, Z^*) a solution to (21) (that exists by assumption); there holds $\theta(\lambda^*, \mu^*, Z^*, 0) = \text{val}(21)$. On the other hand, we have for $\alpha > 0$

$$\theta(\lambda^*, \mu^*, Z^*, \alpha) \leq \Theta(\lambda^*, \mu^*, \alpha) \leq \max_{(\lambda,\mu)\in\Lambda} \Theta(\lambda, \mu, \alpha) \leq \text{val}(21). \tag{30}$$

by definition of Θ and Theorem 3. Recall that θ is concave and upper-semicontinuous since it is defined in (18) as an infinum of linear functions $L(X;\cdot,\cdot,\cdot)$ (see e.g. [HUL93, IV.2.1.2]). Moreover, the domain of θ contains $\mathscr{D}_+$ as well as the point $(\lambda^*, \mu^*, Z^*, 0)$. We deduce that θ is continuous when $\alpha > 0$ tends to zero; more specifically, we have

$$\lim_{\alpha\to 0} \theta(\lambda^*, \mu^*, Z^*, \alpha) = \theta(\lambda^*, \mu^*, Z^*, 0) = \text{val}(21).$$

This limit together with (30) allows us to conclude the proof. □

Experiments in Section 5 illustrate this result: the optimal value of (21) is well approximated by taking α small (between 10^{-2} and 10^{-4} depending on the problem). Let us also mention that this convergence is accentuated in practice by rounding. The optimal value of 0-1 problem (1) is integer if the entries of the M_j and m_j are integers as well, so that rounding up $\Theta(\lambda, \mu, \alpha)$ still gives a lower bound to (1). Rounding up the SDP bound and $\Theta(\lambda, \mu, \alpha)$ for very small α would give the same bound (except in pathological cases).

Individually, the lower bound $\Theta(\lambda, \mu, \alpha)$ could have good tightness, but for a given triple (λ, μ, α), we still have no control on the quality of the bound. We explain in the next section how to use these bounds in practice.

4.3 How to use the bounds in practice?

Maximizing bounds. We note first that Theorem 4 yields another way to compute the SDP bound, specifically by solving

$$\max_{(\lambda,\mu,\alpha)\in\mathscr{D}_+} \Theta(\lambda, \mu, \alpha) = \text{val}(21).$$

However we do not use directly this property because the constraint $\alpha > 0$ in $\mathscr{D}_+$ complicates the picture: the constraint set is not closed, and handling it would require extra technicalities. We proceed otherwise, as follows.

Property (i) of Theorem 4 suggests that α acts like a control parameter to set the level of tightness. Fixing $\alpha > 0$ yields a simpler optimization problem

$$\max_{\lambda\in\mathbb{R}^{m_I}_+,\, \mu\in\mathbb{R}^{m_E}} \Theta(\lambda, \mu, \alpha) \tag{31}$$

with a differentiable objective function (by Theorem 2) and just nonnegativity constraints on the dual variable λ. This problem can be solved by any algorithm of nonlinear optimization (handling box constraints).

Newton-like algorithm. Though Θ is differentiable, it is not twice differentiable due to the projection onto the cone of negative semidefinite matrices in (23). Direct application of the Newton method is not possible, so we turn to Newton-like methods: quasi-Newton methods (see e.g. [BGLS03]), or (inexact) semismooth Newton methods (see [QS93] and [NW99]).

In our numerical experiments, we use the limited memory quasi-Newton of [GL89] to solve (31). It proves to have good numerical behavior, as shown in the next section. Using a limited-memory version allows us to deal with problems (1) with possibly many constraints. In particular, one could add without concern redundant constraints to sharpen the bounds, as is done with the product-constraints in (8).

The parameter α controls the tightness level of the bounds. We also notice that (31) can be solved to various levels of accuracy, so that the stopping criterion of the quasi-Newton solver is a secondary practical parameter impacting the quality of the bound. In Section 5, the stopping criterion is set from `1e-4` to `1e-7` depending on the type of problem and on the value of α. We also mention that we scale the data initially to have better control on the stopping criterion: the constraints $\langle B_i, X\rangle = b_i$ are scaled so that B_i are of norm 1.

We finally underline an important point. Throughout the run of the chosen maximizing algorithm, the current objective value $\Theta(\lambda_k, \mu_k, \alpha)$ is obviously a bound. So in practice within branch-and-bound, the run is stopped when the current value gets above the threshold given by the best known solution of (1). Only few iterations are often necessary to be able to prune the branch-and-bound tree (see the experiments with branch-and-bound in [MR11] using Θ'_{heav} of example 4).

Choosing the tightness parameter α. The question reduces now to how to choose the tightness level α. On the one hand, reducing α enhances the final gap obtained by the maximization (Theorem 4 (i)). On the other hand, reducing α also yields numerical difficulties. We can guess by the expression of Θ and its gradient (Theorem 2) that very small values of α can cause ill-conditioning. Numerical experiments confirm that solving the problem becomes more difficult as α decreases (see Section 5.2).

The goal now is to adapt α to deteriorate the tightness of the SDP bound in order to save computing time. Thus a balance has to be found when reducing α. In practice, one could do preliminary testing (as in Section 5.2) to choose an appropriate α for each problem.

5 Computational study and comparison

We illustrate the properties of the family of bounds $\Theta(\lambda, \mu, \alpha)$ on the three examples of Section 2.2. We denote the bounds (23) respectively by

- Θ_{uqp} for the unconstrained 0-1 quadratic programming problem (4),
- Θ_{heav} for the heaviest subgraph problem (2),
- Θ_{bis} for the graph bisection problem (3).

Section 5.2 shows the profiles of the improvement in tightness when using α to control tightness. Afterwards, numerical experiments of Section 5.3 compare these new bounds with the standard semidefinite programming ones.

5.1 Machine, criterion and data

The numerical experiments have been carried out on a Pentium IV 2.2 GHz with 1 GB of RAM under Linux. The reported computing times are in seconds. In section 5.2, we evaluate the tightness of the bounds by the relative gap between the bound rounded up to the next integer, and the best known value of the initial problem. The best known solution can be either the optimal (computed by an exact resolution scheme), or suboptimal (computed by heuristics, when all exact schemes failed); more details are given for each case.
Unconstrained 0-1 quadratic programming. We consider some of the biggest and most difficult problems of [BE07]: randomly generated instances created with the generator of [PR90] where n is the number of variables and d is the density of matrix M_0 in (4). For each couple of parameters among

$$n = 100, 150, 200 \qquad \text{and} \qquad d = 30\%, 80\%, 100\%$$

we have 10 instances of those problems. In tables 1 and 4, each reported result is thus the mean of the 10 instances for given n and d. The solution of (4) used to evaluate the gap for different bounds is as in [BE07]: it is the exact solution for $n = 100$, and the best feasible available feasible solution for $n = 150, 200$.

Heaviest subgraph problems. We consider randomly generated instances used by several papers [Bil05], [BEP09] and [MR11]. The parameters of these instances are: the size of the graph n, the graph density d, and the value of k. For each combination of parameters among:

$$n = 80, 100, 300 \qquad d = 25\%, 50\%, 75\% \qquad \text{and} \qquad k = n/4, n/2, 3n/4,$$

we have 5 instances of problems. In tables 2 and 5, each reported result is the mean of the 15 instances for given n and d. For $n = 80$ and $n = 100$ the best known solutions are optimal (taken from [BEP09]). For $n = 300$, we used the heuristic of [MR11] (a greedy algorithm followed by a "two-opt" procedure) to generate a good feasible solution. Thus the real gaps between the bounds Θ_{heav} and the optimal value are certainly smaller.

Graph bisection problem. We consider some of the problems of [KRC00]. We have 8 graphs of size n between 84 to 204; two of them are the weighted graphs ex84f and ex132f, the others (denoted ex*a) are unweighted. We get 32 instances of the bisection problem taking

$$k = \lfloor n/2 \rfloor, \lfloor 3n/4 \rfloor, \lfloor 7n/12 \rfloor, \lfloor 13n/24 + 1/2 \rfloor.$$

Note that for $k = \lfloor n/2 \rfloor$, we obtain the well-studied equicut problem. In the forthcoming tables 3 and 6, each reported result corresponds to one instance. The best known solutions are taken from [KRC00], except for $n \geq 156$. These solutions are optimal when $n = 84$ and are at most at 1% for $n = 108$ and $n = 132$. For $n \geq 156$, we compute a suboptimal solution with a simple heuristic ("two-opt" procedure from randomly generated bisections).

5.2 Comparison of the bounds; role of α as a control parameter

We test and analyze the impact in terms of quality of the bound and computing time required for different choices of α. Such preliminary testing allows one to adjust α for each problem type. The three tables of this section report the time to optimize the bounds for a given value of α and the final gap between the bound and the best known solution of the initial combinatorial problem.

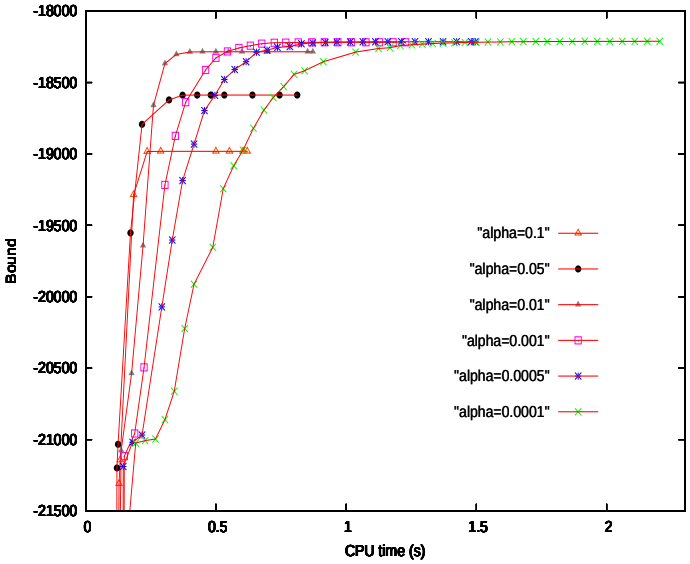

Fig. 1 Comparison (axis: value of the bound vs cputime), for different values of α, of the bounds $\Theta_{\text{uqp}}(\mu,\alpha)$ for a problem (4) of size $n = 150$.

Problem		Θ_{uqp} with $\alpha = 10^{-1}$		Θ_{uqp} with $\alpha = 5.10^{-2}$		Θ_{uqp} $\alpha = 10^{-2}$	
n	d	time (s)	gap(%)	time (s)	gap(%)	time (s)	gap(%)
100	100%	0.14”	9.53%	0.15”	8.57%	0.22”	7.82%
150	30%	0.33”	12.92%	0.35”	10.62%	0.43”	8.86%
150	80%	0.31”	11.67%	0.36”	10.28%	0.46”	9.21%
200	30%	0.57”	14.32%	0.70”	11.59%	1.03”	9.48%
200	80%	0.62”	11.76%	0.84”	10.19%	1.10”	8.96%
mean		**0.39”**	**12.04%**	**0.48”**	**10.25%**	**0.65”**	**8.87%**
Problem		Θ_{uqp} with $\alpha = 10^{-3}$		Θ_{uqp} with $\alpha = 5.10^{-3}$		Θ_{uqp} with $\alpha = 10^{-4}$	
n	d	time (s)	gap(%)	time (s)	gap(%)	time (s)	gap(%)
100	100%	0.44”	7.65%	0.69”	7.64%	1.06”	7.64%
150	30%	0.92”	8.46%	1.47”	8.45%	2.23”	8.43%
150	80%	0.99”	8.95%	1.60”	8.94%	2.57”	8.93%
200	30%	1.76”	9.02%	2.84”	9.00%	4.52”	8.99%
200	80%	2.01”	8.70%	3.29”	8.68%	5.11”	8.68%
mean		**1.22”**	**8.56%**	**1.98”**	**8.55%**	**3.10”**	**8.53%**

Table 1 Numerical results for the unconstrained quadratic problem (4) (each entry is the mean of 10 results). The table reports the computing time of Θ_{uqp} and gap(%) the averaged relative difference between the computed bounds and the best known solution of (4).

For unconstrained 0-1 quadratic programming, six different values of α are compared in table 1. Figure 1 gives an illustration of the convergence for a particular instance with $n = 150$ and $d = 0.3$. We note that using $\alpha = 10^{-4}$ is more expensive than $\alpha = 10^{-3}$ for only a small improvement of the gap.

For heaviest subgraph problems and for bisection problems, the picture is different. Tables 2 and 3 shows that the tightness greatly increases when α vanishes. In particular, taking $\alpha = 10^{-4}$ provides a SDP-like bound in both cases. The computing cost differs though: for the heaviest subgraph problems, this is obtained within a reasonable computing time (when $n \leq 100$), whereas it is more costly for graph bisection problems of the same sizes. More

generally, we shall see in Section 5.3, that our best bounds are SDP-like for the three problems. Thus the corresponding gaps in Tables 1, 2, and 3 are almost equal to the SDP ones.

Problem		Θ_{heav} with $\alpha = 10^{-2}$		Θ_{heav} with 5.10^{-3}		Θ_{heav} with 10^{-3}		Θ_{heav} with 10^{-4}	
n	k	time (s)	gap(%)	time (s)	gap(%)	time (s)	gap(%)	time (s)	gap(%)
80	20	0.04"	29.77%	0.07"	17.00%	0.10"	9.64%	0.18"	8.07%
	40	0.03"	18.16%	0.05"	4.65%	0.07"	2.45%	0.11"	2.00%
	60	0.07"	3.13%	0.09"	1.73%	0.10"	0.83%	0.15"	0.65%
100	25	0.05"	32.49%	0.09"	17.38%	0.14"	9.46%	0.24"	7.77%
	50	0.04"	13.55%	0.07"	5.67%	0.09"	2.47%	0.16"	1.98%
	75	0.07"	3.40%	0.09"	1.79%	0.15"	0.82%	0.17"	0.63%
300	75	0.43"	46.48%	1.12"	19.59%	1.85"	9.63%	3.62"	7.43%
	150	0.49"	24.22%	0.68"	4.23%	0.85"	2.40%	1.11"	1.67%
	225	0.43"	6.02%	0.55"	4.04%	1.09"	2.28%	3.05"	0.50%
mean		**0.18"**	**19.69%**	**0.31"**	**8.46%**	**0.49"**	**4.44%**	**0.97"**	**3.41%**

Table 2 Numerical results for the heaviest subgraph problem (8) (each entry is the mean of 15 results). The table reports the computing time of Θ_{heav} and gap(%), the averaged relative difference between the computed bounds and the best known solution of (8).

Problem		Θ_{bis} with $\alpha = 10^{-2}$		Θ_{bis} with $\alpha = 5.10^{-3}$		Θ_{bis} with $\alpha = 10^{-3}$		Θ_{bis} with 10^{-4}	
graph	k	time (s)	gap(%)	time (s)	gap(%)	time (s)	gap(%)	time (s)	gap(%)
ex84a	42	0.09"	5.26%	0.11"	4.31%	0.20"	3.50%	0.29"	3.37%
	63	0.08"	7.12%	0.08"	6.02%	0.13"	4.93%	0.22"	4.74%
	49	0.11"	5.83%	0.14"	4.99%	0.15"	4.16%	0.26"	4.02%
	46	0.10"	5.58%	0.13"	4.63%	0.17"	3.81%	0.28"	3.67%
ex84f	42	0.18"	1.33%	0.18"	1.26%	0.28"	1.20%	0.47"	1.20%
	63	0.15"	1.99%	0.18"	1.90%	0.29"	1.81%	0.45"	1.80%
	49	0.18"	1.54%	0.17"	1.47%	0.27"	1.39%	0.54"	1.38%
	46	0.18"	1.43%	0.21"	1.36%	0.35"	1.28%	0.49"	1.27%
ex108a	54	0.18"	5.13%	0.18"	4.17%	0.24"	3.45%	0.40"	3.29%
	81	0.17"	6.23%	0.18"	5.03%	0.23"	4.04%	0.53"	3.29%
	63	0.16"	5.39%	0.18"	4.39%	0.27"	3.65%	0.52"	3.83%
	59	0.17"	5.34%	0.19"	4.37%	0.26"	3.64%	0.40"	3.48%
ex132a	66	0.28"	4.67%	0.33"	3.66%	0.47"	2.92%	0.71"	2.76%
	99	0.24"	5.44%	0.27"	4.21%	0.40"	3.26%	0.81"	3.05%
	77	0.32"	4.88%	0.33"	3.89%	0.36"	3.12%	0.83"	2.96%
	72	0.28"	4.92%	0.36"	3.96%	0.42"	3.21%	0.57"	3.05%
ex132f	66	0.51"	1.27%	0.58"	1.14%	1.07"	1.07%	1.34"	1.07%
	99	0.39"	1.65%	0.43"	1.54%	0.63"	1.45%	1.25"	1.43%
	77	0.41"	1.19%	0.52"	1.10%	0.68"	1.03%	1.22"	1.02%
	72	0.47"	1.23%	0.50"	1.08%	0.84"	1.01%	1.12"	1.00%
ex156a	78	0.62"	5.06%	0.66"	4.10%	0.83"	3.36%	0.82"	3.22%
	117	0.34"	6.04%	0.47"	4.84%	0.61"	3.89%	1.23"	3.64%
	91	0.40"	5.15%	0.41"	4.19%	0.61"	3.43%	0.97"	3.28%
	85	0.46"	4.79%	0.52"	3.85%	0.58"	3.11%	1.00"	2.96%
ex180a	90	0.82"	4.72%	1.04"	3.78%	1.27"	3.00%	1.71"	2.83%
	135	0.46"	5.36%	0.53"	4.19%	0.74"	3.25%	1.52"	3.02%
	105	0.54"	4.79%	0.69"	3.79%	0.81"	3.01%	1.93"	2.84%
	98	0.62"	4.71%	0.86"	3.73%	1.17"	2.97%	2.20"	2.81%
ex204a	102	1.16"	4.06%	1.50"	3.11%	1.58"	2.36%	2.07"	2.21%
	153	0.96"	5.32%	0.72"	4.13%	1.19"	3.21%	2.55"	3.03%
	119	0.90"	4.50%	1.10"	3.51%	1.36"	2.74%	2.46"	2.60%
	111	0.95"	4.10%	1.07"	3.12%	1.68"	2.36%	2.61"	2.23%
mean		**0.40"**	**4.28%**	**0.46"**	**3.49%**	**0.62"**	**2.87%**	**1.06"**	**2.67%**

Table 3 Numerical results for graph bisection problems. The table reports the computing time of Θ_{bis} and gap(%), the averaged relative difference between the computed bounds and the best known solution of (8).

5.3 Comparison with standard semidefinite bounds

This section gives numerical comparisons of the new bounds and the SDP bound (21), in terms of tightness, computation time and the balance between both. To compute the SDP bound, we use two SDP solvers known to be efficient:

1. SB [Hel04], a SDP solver based on the spectral bundle method [HR00]. This software can handle large-scale problems and is known to be efficient in the context of combinatorial optimization. We use it with the default settings, except that: we set the stopping criterion to `1e-4` (instead of `1e-5`), we activate (with `-sh 1` and `-si 1`) the heuristic for finding a good starting point and the initial rescaling of the constraints.
2. CSDP [Bor99], a robust and efficient interior point solver. We use it with the default settings, except that we activate "Fastmode" and we set the stopping criterion `objtol` to `1e-5` (instead of `1e-8`).

Remark 1 (Different formulations of the same SDP bound) Computing times depend obviously on the solvers, and in turn the performances of the solvers depend on the formulation of the bounds. There is no ambiguity in the SDP relaxation of (4), but there is some ambiguity in those of (8) and (10). It turns out that adding the n product constraints (7) is equivalent to adding the squared constraint $(e^\top x - k)^2 = 0$ (see e.g. [LO99]); this leads to two equivalent SDP relaxations. For the k-heaviest problem, preliminary tests of [MR10] showed that the formulation with the product constraints is best solved by SB, while the one with the squared constraint is best solved by CSDP. We have observed a similar behavior for the graph bisection problems. In the numerical experiments below, we take the most advantageous formulation for each solver. We emphasize this idea which is unusual (as far as we are aware) but essential to make fair comparisons. We see this idea as a secondary but important contribution of this paper. □

Problem		Θ_{uqp} with $\alpha = 10^{-3}$		SDP with SB		SDP with CSDP	
n	d	time (s)	gap(%)	time	time (s) to achieve Θ_{uqp}	time (s)	time (s) to achieve Θ_{uqp}
100	100%	0.44"	0.02%	2.22"	0.94"	1.48"	0.84"
150	30%	0.92"	0.04%	3.96"	2.64"	2.49"	2.35"
150	80%	0.99"	0.03%	4.93"	2.49"	3.33"	2.21"
200	30%	1.76"	0.05%	7.54"	3.76"	4.65"	3.34"
200	80%	2.01"	0.04%	10.94"	3.68"	7.14"	3.27"
mean		**1.22"**	**0.04%**	**5.92"**	**3.82"**	**2.70"**	**2.40"**

Table 4 Numerical tests for the unconstrained quadratic problem (each entry is the mean of 15 results).

Problem		Θ_{heav} with $\alpha = 10^{-4}$		SDP with SB		SDP with CSDP	
n	k	time (s)	gap(%)	time	time (s) to achieve Θ_{heav}	time (s)	time (s) to achieve Θ_{heav}
80	20	0.18"	0.16%	0.93"	0.44"	0.33"	0.26"
	40	0.11"	0.05%	0.97"	0.66"	0.36"	0.30"
	60	0.15"	0.01%	1.38"	1.19"	0.34"	0.29"
100	25	0.24"	0.17%	2.07"	0.84"	0.58"	0.45"
	50	0.16"	0.06%	1.98"	0.93"	0.56"	0.46"
	75	0.17"	0.01%	4.66"	3.12"	0.59"	0.50"
300	75	3.62"	0.11%	21.49"	8.18"	9.73"	7.79"
	150	1.11"	0.04%	11.15"	10.13"	11.86"	9.52"
	225	3.05"	0.01%	34.02"	11.45"	8.78"	6.82"
mean		**0.97"**	**0.07%**	**8.74"**	**4.10"**	**3.68"**	**2.93"**

Table 5 Numerical tests for the heaviest subgraph problem (each entry is the mean of 15 results).

Tables 4, 5 and 6 report the computation times of the three solvers. Recall that the solvers do not compute the same bounds: to compare the SDP bound (21) and the maximum of $\Theta(\mu, \alpha)$ for fixed α, we report:

- gap% which is the relative difference between the two bounds,
- the computing times for SB and CSDP to converge,
- the computing times for SB and CSDP to achieve the final Θ.

We emphasize that "gap" does not have the same meaning as in the previous tables: it is now the difference between $\Theta(\mu_k, \alpha)$ and the SDP bound (21). We also note that to have a fair comparison, the bounds are not rounded up to the next integer as was done previously. Finally we mention that table 5 aggregates the results of [MR11].

The three tables show that the new bounds provide SDP-quality bounds more quickly. More specifically, the difference between the final Θ and the SDP bound is always smaller than 0.1% (recall moreover that in practice it would be reduced again by rounding). Though SB and CSDP eventually converge to the SDP bound, which is more tight, they need more time to do so, and they also need more time to achieve the best Θ (see the columns "time to achieve Θ"). On the unconstrained quadratic problems and heaviest subgraph problems, our solver clearly outperforms SB and CSDP: in table 4, the gain in time is by a factor of 3 compared to SB and a factor of 2 compared to CSDP. Note that the gain is less important for graph bisection problems: the average time is still better, but CSDP performs better for some graphs (ex84f in particular).

Problem		Θ_{bis} with $\alpha = 10^{-4}$		SDP with SB		SDP with CSDP	
graph	k	time (s)	gap(%)	time (s)	time (s) to achieve Θ_{bis}	time (s)	time (s) to achieve Θ_{bis}
ex84a	42	0.29"	0.04%	0.82"	0.39"	0.41"	0.33"
	63	0.22"	0.03%	0.78"	0.41"	0.57"	0.49"
	49	0.26"	0.03%	1.03"	0.43"	0.42"	0.33"
	46	0.28"	0.03%	0.66"	0.36"	0.42"	0.33"
ex84f	42	0.47"	0.02%	1.11"	0.58"	0.39"	0.33"
	63	0.45"	0.01%	1.50"	1.18"	0.47"	0.37"
	49	0.54"	0.01%	1.60"	0.71"	0.53"	0.45"
	46	0.49"	0.01%	0.80"	0.63"	0.43"	0.36"
ex108a	54	0.40"	0.03%	1.66"	0.84"	0.77"	0.61"
	81	0.53"	0.02%	4.27"	2.32"	0.99"	0.80"
	63	0.52"	0.03%	1.30"	0.75"	0.77"	0.60"
	59	0.40"	0.04%	1.36"	0.56"	0.78"	0.57"
ex132a	66	0.71"	0.02%	2.07"	1.22"	1.39"	1.11"
	99	0.81"	0.04%	4.09"	1.71"	1.52"	1.24"
	77	0.83"	0.02%	2.63"	1.44"	1.95"	1.56"
	72	0.57"	0.04%	2.21"	0.94"	1.56"	1.23"
ex132f	66	1.34"	0.01%	2.16"	1.52"	1.39"	1.19"
	99	1.25"	0.01%	2.62"	2.11"	1.50"	1.40"
	77	1.22"	0.01%	2.87"	2.00"	1.76"	1.51"
	72	1.12"	0.01%	5.54"	3.16"	2.57"	2.20"
ex156a	78	0.82"	0.04%	2.48"	1.11"	2.35"	1.73"
	117	1.23"	0.03%	6.35"	3.35"	3.27"	2.66"
	91	0.97"	0.03%	3.17"	1.88"	2.08"	1.79"
	85	1.00"	0.05%	2.52"	1.20"	2.08"	1.64"
ex180a	90	1.71"	0.02%	2.70"	1.88"	3.13"	2.51"
	135	1.52"	0.03%	10.41"	5.16"	2.83"	2.33"
	105	1.93"	0.02%	3.98"	1.91"	2.49"	2.00"
	98	2.20"	0.03%	3.70"	2.14	2.61"	1.96"
ex204a	102	2.07"	0.02%	6.23"	2.85"	3.98"	2.81"
	153	2.55"	0.03%	30.94"	20.21"	3.89"	3.20"
	119	2.46"	0.03%	6.88"	4.34"	4.08"	3.06"
	111	2.61"	0.04%	5.75"	3.51"	3.52"	2.81"
mean		**1.06"**	**0.03%**	**3.94"**	**2.28"**	**1.78"**	**1.42"**

Table 6 Numerical results for graph bisection problems.

We highlight the quick convergence of our solver by plotting the run for a characteristic instance of each problem: in Figure 2 for a quadratic problem, in Figure 3 for a bisection problem, and similar plots for heaviest subgraph problems are in [MR11] and [MR10].

The figures illustrate that our solver has a very strong initial increase effect. In particular, the improvement of the bound is very large in the first iterations. Moreover, we can observe in the tables that the running times of our solver are similar for instances of the same size. These two properties are highly desirable in the context of branch-and-bound.

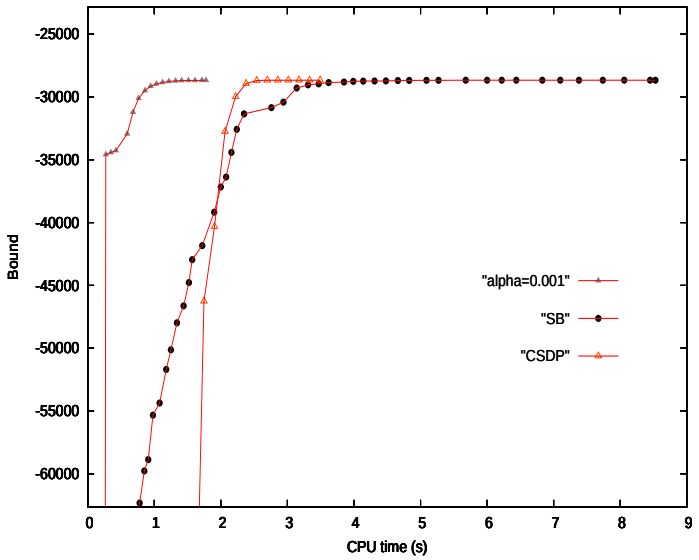

Fig. 2 An instance of the unconstrained quadratic problem (4) of size $n = 100$: comparison of the maximization of Θ_{uqp} with the runs of SB and CSDP computing the SDP relaxation.

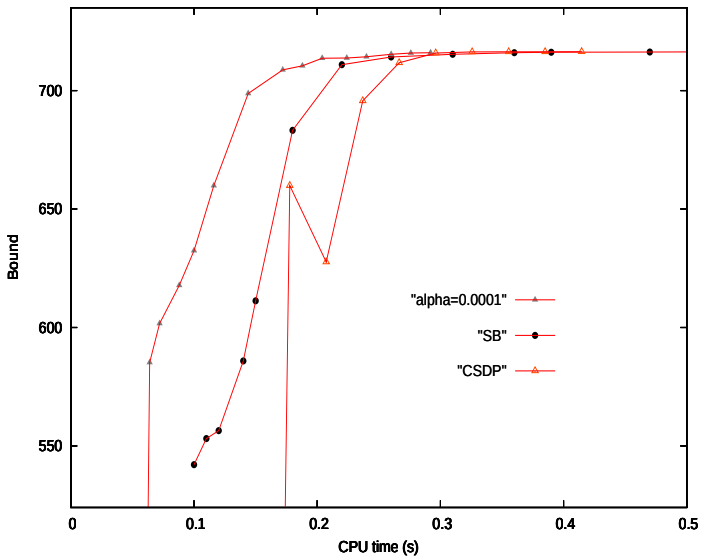

Fig. 3 The instance ex84a of the graph bisection problem (10) with $k = 42$: comparison of the maximization of Θ_{bis} with the runs of SB and CSDP computing the SDP relaxation.

Each of the two SDP solvers has one of the two properties mentioned above. CSDP is also quite robust with respect to computing time (in contrast with SB for which "accidents" occurred, see for example ex204a in table 6). On the other hand, SB also shows a nice initial increase, although the computing times of SB are in general larger than those of CSDP. We notice in figure 3 that the initial heuristic of SB for finding a starting point is very efficient.

We point out that during the solving process, current primal and dual solutions of CSDP are not feasible and thus the dual value is not necessarily a valid lower bound. Nevertheless, we have observed during all our experiments that CSDP gives in fact lower bounds. We see also on Figure 3 that CSDP deteriorates the dual bound in order to make a big improvement in the feasibility of the current dual solution in the middle of the run.

5.4 Conclusions, perspectives

This paper introduces a family of semidefinite bounds for 0-1 quadratic programming. It presents a methodology to construct these bounds and to use them in practice. Numerical experiments illustrate the properties of these bounds on three combinatorial optimization problems.

The approach could be applied to any 0-1 problem of the form (1), but it is certainly not pertinent in any situation. Branch-and-bound algorithms using linear programming have reached a high level of sophistication and performance; they can solve efficiently many problems of the type (1). They have nevertheless shown limitations on some problems (such as the three problems considered here), and consequently, alternative approaches have been recently developed using quadratic programming (in particular the so-called QCR method, see [BE07], [BEP09]) or semidefinite programming (see [RRW10]).

Our approach is relevant in these latter situations. We have a parameter α to set the targeted level of tightness below the usual SDP bound. Experiments with α small show that we can (almost) attain SDP bounds quicker than state-of-the-art SDP solvers. In addition, preliminary computational experiments with α large show that we can attain linear programming bounds and quadratic programming bounds quicker than CPLEX. Although this computational study has yet to be completed, we can say that this new family of bounds have some universality: whatever type of bound one looks for (tight, cheap, or balanced), one may be able to find it in this family.

Tests such as those of Section 5.2 are useful for setting the control parameter α to the desired tightness level. Once α is set, the bounds are computed by a quasi-Newton algorithm which combines several of the advantages of the SDP solvers: it is reliable (like CSDP) and has a strong initial increase effect (like SB). Those are properties of paramount importance for this method to be used as bounding procedure within branch-and-bound algorithms. A version of these bounds is used with success in [MR11] in a branch-and-bound method to solve heaviest k-subgraph problems to optimality.

Finally, we mention that it is useful to combine these bounds with good heuristics for computing feasible solutions. Though cheaper than the usual SDP bounds, computing the bounds still demands sophisticated numerical linear algebra and optimization techniques. The bounds give better pruning in the search tree when combined with good feasible solutions, allowing us to fully benefit from the computing time investment in the bound quality.

Acknowledgement

We are grateful to Sourour Elloumi who provided us a lot of material for the numerical tests, and to Sofia Zaourar who helped us to develop parts of the solver for bisection problems.

References

[AL12] M. Anjos and J.B. Lasserre. Handbook of semidefinite, conic and polynomial optimization. Springer, 2012.

[BE07] A. Billionnet and S. Elloumi. Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. Mathematical Programming, 109(1):55–68, 2007.

[BEP09] A. Billionnet, S. Elloumi, and M.-C. Plateau. Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method. Discrete Applied Mathematics, 157(6):1185–1197, 2009.

[BGLS03] J.F. Bonnans, J.Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal. Numerical Optimization. Springer Verlag, 2003.

[Bil05] A. Billionnet. Different formulations for solving the heaviest k-subgraph problem. Information Systems and Operational Res., 43(3):171–186, 2005.

[Bor99] B. Borchers. CSDP, a C library for semidefinite programming. Optimization Methods and Software, 11(1):613–623, 1999.

[BV04] S. Boyd and L. Vandenberghe. Convex optimization. Cambridge University Press, 2004.

[dKS10] E. de Klerk and R. Sotirov. Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. Math. Prog., 122(2), 2010.

[FR07] A. Faye and F. Roupin. Partial lagrangian for general quadratic programming. 4'OR, A Quarterly Journal of Operations Research, 5(1):75–88, 2007.

[GL89] J.Ch. Gilbert and C. Lemaréchal. Some numerical experiments with variable-storage quasi-Newton algorithms. Mathematical Programming, 45:407–435, 1989.

[GW95] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM, 6:1115–1145, 1995.

[Hel00] C. Helmberg. Semidefinite Programming for Combinatorial Optimization. PhD thesis, Habilitationsschrift, TU Berlin, 2000.

[Hel04] C. Helmberg. A C++ implementation of the Spectral Bundle Method. http://www-user.tu-chemnitz.de/~helmberg/SBmethod/, 2004. Version 1.1.3.

[Hig88] N. Higham. Computing a nearest symmetric positive semidefinite matrix. Linear Algebra and its Applications, 103:103–118, 1988.

[HPRW95] Christoph Helmberg, Svatopluk Poljak, Franz Rendl, and Henry Wolkowicz. Combining semidefinite and polyhedral relaxations for integer programs. In Egon Balas and Jens Clausen, editors, IPCO, volume 920 of Lecture Notes in Computer Science, pages 124–134. Springer, 1995.

[HR00] C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. SIAM Journal on Optimization, 10(3):673–696, 2000.

[HRVW96] C. Helmberg, F. Rendl, R.J. Vanderbei, and H. Wolkowicz. An interior point method for semidefinite programming. SIAM Journal on Optimization, 6:342–361, 1996.

[HUL93] J.-B. Hiriart-Urruty and C. Lemaréchal. Convex Analysis and Minimization Algorithms. Springer Verlag, Heidelberg, 1993. Two volumes.

[KKW09] S. Kim, M. Kojima, and H. Waki. Exploiting sparsity in sdp relaxation for sensor network localization. SIAM J. on Optimization, 20(1):192–215, 2009.

[KRC00] S. Karisch, F. Rendl, and J. Clausen. Solving graph bisection problems with semidefinite programming. INFORMS J. on Computing, 12(3):177–191, 2000.

[Lem01] C. Lemaréchal. Lagrangian relaxation. In M. Jünger and D. Naddef, editors, Computational Combinatorial Optimization, pages 112–156. Springer Verlag, Heidelberg, 2001.

[LO99] C. Lemaréchal and F. Oustry. Semidefinite relaxations and Lagrangian duality with application to combinatorial optimization. Rapport de Recherche 3710, INRIA, 1999.

[LS91] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. SIAM Journal on Optimization, 1(2):166–190, 1991.

[Mal07] J. Malick. Spherical constraint in Boolean quadratic programming. Journal of Global Optimization, 39(4), 2007.

[MPRW09] J. Malick, J. Povh, F. Rendl, and A. Wiegele. Regularization methods for semidefinite programming. SIAM Journal on Optimization, 20(1):336–356, 2009.

[MR10] J. Malick and F. Roupin. Numerical study of semidefinite bounds for the k-cluster problem. In Electronics Notes of Discrete Mathematics, pages 399–406. Elsevier, 2010. n ISCO'10, International Symposium on Combinatorial Optimization.

[MR11] J. Malick and F. Roupin. Solving k-cluster problems to optimality using adjustable semidefinite programming bounds. To appear in Math. Programming, 2011.

[NW99] J. Nocedal and S.J. Wright. Numerical Optimization. Springer Verlag, New York, 1999.

[PIS+04] P. Pardalos, L. D. Iasemidis, J. C. Sackellares, W. Chaovalitwongse, P. Carney, O. Prokopyev, V. Yatsenko, and D-S Shiau. Seizure warning algorithm based on optimization and nonlinear dynamics. Mathematical Programming, 101:365–385, 2004.

[PR90] P. Pardalos and G.P. Rodgers. Computational aspects of a branch and bound algorithm for quadratic zero-one programming. Computing, 45:134–144, 1990.

[PRW95] S. Poljak, F. Rendl, and H. Wolkowicz. A recipe for semidefinite relaxation for (0,1)-quadratic programming. Journal of Global Optimization, 7:51–73, 1995.

[QS93] L.Q. Qi and J. Sun. A nonsmooth version of Newton's method. Mathematical Programming, 58(3):353–367, 1993.

[Rou04] F. Roupin. From linear to semidefinite programming: an algorithm to obtain semidefinite relaxations for bivalent quadratic problems. Journal of Combinatorial Optimization, 8(4):469–493, 2004.

[RRW10] F. Rendl, G. Rinaldi, and A. Wiegele. Solving max-cut to optimality by intersecting semidefinite and polyedral relaxations. Math. Programming, 121:307–335, 2010.

[SA90] Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. SIAM Journal on Discrete Mathematics, 3(3):411–430, 1990.

[SVW00] R. Saigal, L. Vandenberghe, and H. Wolkowicz. Handbook of Semidefinite Programming. Kluwer, 2000.