Università degli Studi di Pisa
Dipartimento di Statistica e Matematica
Applicata all'Economia

Report n. 288

A computational comparison of some branch and bound methods for indefinite quadratic programs

Riccardo Cambini e Claudio Sodini

Pisa, Novembre 2006
- Stampato in Proprio –

Via Cosimo Ridolfi, 10 - 56124 PISA - Tel. Segr. Amm. 050 2216231 Segr. Stud. 050 2216317 Fax 050 2216375
Cod. Fisc. 80003670504 - P. IVA 00286820501 - Web http://statmat.ec.unipi.it/ - E-mail: dipstat@ec.unipi.it

A computational comparison of some branch and bound methods for indefinite quadratic programs

Riccardo Cambini and Claudio Sodini

Department of Statistics and Applied Mathematics
Faculty of Economics, University of Pisa
Via Cosimo Ridolfi 10, 56124 Pisa, ITALY
e-mail: cambric@ec.unipi.it, csodini@ec.unipi.it

Abstract

The aim of this paper is to propose different branch and bound methods for solving indefinite quadratic programs. In these methods the quadratic objective function is decomposed in a d.c. form and the relaxations are obtained by linearizing the concave part of the decomposition. In this light, various decomposition schemes have been considered and studied. The various branch and bound solution methods have been implemented and compared by means of a deep computational test.

Key words: Quadratic programming, branch and bound, d.c. decomposition

AMS - 2000 Math. Subj. Class. 90C20, 90C26, 90C31.
JEL - 1999 Class. Syst. C61, C63.

1 Introduction

The aim of this paper is to propose various solution methods for quadratic indefinite programs and ways they can be solved by means of branch and bound algorithms based on the partition of the feasible region and the relaxation of the objective function. These problems (see for example [2, 3, 4, 9, 16, 17, 18, 19, 20, 21, 26, 28, 29]) have been approached in the literature in several ways and in [1, 10, 15, 22, 27] they were solved with solution algorithms based on convex relaxations obtained by means of a transformation of the objective function in a d.c. form. In this light, various d.c. decompositions of the quadratic objective function, different from the ones proposed in [1, 10, 22, 27], have been studied.

In Section 2 we preliminarily define the considered quadratic problem. Then, in Section 3, we propose some componentwise relaxations of the objective function which allow us to state branch and bound schemes where the feasible region of the current subproblem is splitted with respect to a single variable. In Section 4 two more relaxations of the objective function are considered, thus obtaining branch and bound schemes where the feasible region of the current subproblem is splitted with respect to a linear function. Finally, in Section 5, the results of a deep computational test are provided and discussed.

2 Statement of the problem

In this paper we aim to study a generic quadratic problem having a polyhedral feasible region.

Definition 2.1 We define the following quadratic program:

$$P : \begin{cases} \min f(x) = \frac{1}{2}x^T Ax + c^T x \\ x \in X \subset \Re^n \end{cases}$$

where X is a compact polyhedron, $c \in \Re^n$ and $A \in \Re^{n\times n}$ is any symmetric matrix. Notice that X can be given by inequality constraints $Bx \leq b$ and/or box constraints $\tilde{l} \leq x \leq \tilde{u}$ and/or equality constraints $Mx = q$, where $B \in \Re^{m\times n}$, $b \in \Re^m$, $\tilde{l}, \tilde{u} \in \Re^n$, $M \in \Re^{h\times n}$, $q \in \Re^h$.

If A is positive semidefinite then f is convex and hence problem P can be solved by means of any of the known algorithms for convex quadratic programs. The aim of this paper is to describe some branch and bound schemes for solving problem P when A is not positive semidefinite and to provide detailed results of a computational comparison of the schemes themselves. The idea is to decompose $f(x)$ in a d.c. form, that is to decompose A in the form $A = Q - C$ where Q and C are symmetric positive semidefinite matrices. In this light, the objective function can be rewritten as:

$$f(x) = \frac{1}{2}x^T Qx + c^T x - \frac{1}{2}x^T Cx$$

A relaxation of function $f(x)$ can be obtained by linearizing its concave part $-\frac{1}{2}x^T Cx$. These relaxations can then be used in order to propose branch and bound schemes to determine an optimal solution of the problem.

3 Componentwise relaxations

This approach is based on the decomposition of matrix A in the form ([1]):

$$A = Q - diag(w) \tag{1}$$

where $Q \in \Re^{n\times n}$ is symmetric and positive semidefinite, $w \in \Re^n$, $w \geq 0$ and $diag(w)$ is the positive semidefinite diagonal matrix with diagonal elements

[1]Note that in [15, 22] a decomposition of this kind (with $w = k[1, \dots, 1]^T$, $k > 0$ large enough) was studied. Another decomposition of this kind, based on diagonal dominance, has been proposed in [22].

given by the components of vector w. Such a decomposition allows us to rewrite function f as follows:

$$f(x) = \frac{1}{2}x^T Qx + c^T x - \frac{1}{2}\sum_{i=1}^{n} w_i x_i^2$$

so that the concave part $-\frac{1}{2}\sum_{i=1}^{n} w_i x_i^2$ has separable variables thus allowing a componentwise linearization. In this light, we can consider branch and bound schemes where the feasible region of the current subproblem is splitted with respect to one of the components x_i such that $w_i \neq 0$.

3.1 Main properties

Given a pair of vectors $l, u \in \Re^n$, such that $l \leq u$, we can denote with $B(l,u) = \{x \in \Re^n : l \leq x \leq u\}$ the box generated by l and u. The concave part $-\frac{1}{2}\sum_{i=1}^{n} w_i x_i^2$ of $f(x)$ can be linearized over $B(l,u)$ as follows:

$$f_B(x) = \frac{1}{2}x^T Qx + c^T x - \frac{1}{2}\sum_{i=1}^{n} w_i[x_i(l_i + u_i) - l_i u_i] = \frac{1}{2}x^T Qx + \bar{c}^T x + \bar{c}_0$$

with:

$$\bar{c}_0 = \frac{1}{2}\sum_{i=1}^{n} w_i l_i u_i \quad \text{and} \quad \bar{c} = (\bar{c}_i) \text{ where } \bar{c}_i = c_i - \frac{1}{2}w_i(l_i + u_i) \ \forall i = 1, \ldots, n.$$

Notice that $f_B(x)$ is a relaxation of $f(x)$ over the box $B(l,u)$, which allows to define the relaxed convex subproblem for the branch and bound schemes:

$$P_B(l,u) : \begin{cases} \min f_B(x) \\ x \in X \cap B(l,u) \end{cases}$$

The following result provides an estimation of the error done by solving the linearized problem instead of the original one. With this aim the next functions will be used:

$$\begin{aligned} Err_B(x,i) &= \frac{1}{2}w_i(u_i - x_i)(x_i - l_i) \ , \quad i = 1, \ldots, n \\ Err_B(x) &= f(x) - f_B(x) = \frac{1}{2}\sum_{i=1}^{n} w_i(u_i - x_i)(x_i - l_i) = \sum_{i=1}^{n} Err_B(x,i) \end{aligned}$$

Theorem 3.1 *Let us consider problems P and $P_B(l,u)$ and let*

$$x^* = \arg \min_{x \in X \cap B(l,u)} \{f(x)\} \quad \text{and} \quad y^* = \arg \min_{x \in X \cap B(l,u)} \{f_B(x)\} \ .$$

Then: $0 \leq f(x^*) - f_B(y^*) \leq Err_B(y^*)$.

Proof By means of the given definitions it is:

$$f(x^*) \leq f(y^*) \quad \text{and} \quad f_B(y^*) \leq f_B(x^*).$$

Noticing that $f_B(x^*) \leq f(x^*)$ we obtain:

$$0 \leq f_B(x^*) - f_B(y^*) \leq f(x^*) - f_B(y^*) \leq f(y^*) - f_B(y^*) = Err_B(y^*)$$

so that the result is proved. □

The previous result suggests to determine decompositions of the kind $A = Q - diag(w)$ having components of w as small as possible.

3.2 Branch and bound scheme

The following branch and bound scheme can then be given. With this aim, notice that $2n$ linear programs are preliminarly needed to determine $\tilde{l}$ and $\tilde{u}$ in the case they are not already given (actually, we just need to compute $\tilde{l}_i$ and $\tilde{u}_i$ for the indices i such that $w_i > 0$).

Procedure **Solve**$_1(P)$
  determine a decomposition $A = Q - diag(w)$;
  determine $\tilde{l}$ and $\tilde{u}$ (if needed);
  fix the positive value ϵ; $UB := +\infty$;
  Explore$_1(\tilde{l}, \tilde{u})$;
  x^* is an optimal solution and UB is its value;
end proc.

The core of the algorithm is the following recursive procedure "Explore$_1()$", where A_i denotes the i-th row of A:

Procedure **Explore**$_1(l, u)$
  if $X \cap B(l, u) \neq \emptyset$ *then*
    Let $\overline{x}$ be the optimal solution of $P_B(l, u)$;
    if $f(\overline{x}) < UB$ *then* $UB := f(\overline{x})$; $x^* := \overline{x}$ *end if*;
    if $f_B(\overline{x}) < UB$ *and* $Err_B(\overline{x}) > \epsilon$ *then*
      let $i = \arg\max_{j \in \{1,\ldots,n\}} \{Err_B(\overline{x}, j)\}$;
      define $l' := l$, $l'' := l$, $u' := u$, $u'' := u$;
      let $u'_i := \overline{x}_i$, $l''_i := \overline{x}_i$;
      if $A_i\overline{x} + c_i > 0$ *then*
        Explore$_1(l', u')$;
        Explore$_1(l'', u'')$;
      else
        Explore$_1(l'', u'')$;
        Explore$_1(l', u')$;
      end if;
    end if;
  end if;
end proc.

The proposed branching scheme follows the so called "rectangular method" (see for example [29]). In this scheme, problem $P_B(l,u)$ can be solved by means of any of the known algorithms for convex quadratic programs. Notice that the visit criterion $A_i\overline{x} + c_i > 0$ implies that we firstly solve the subproblem where the function $f(x)$ restricted to the single variable x_i is locally decreasing. Notice also that condition $Err_B(\overline{x}) > \epsilon > 0$ implies that for any index $i = \arg\max_{j\in\{1,\dots,n\}}\{Err_B(\overline{x},j)\}$ it results $Err_B(\overline{x},i) > 0$. This last condition implies that $w_i > 0$, which means that the only variables involved in the branching operations are the ones corresponding to the nonzero components of w. As a consequence, having zero components in vector w could improve the performance of the branch and bound algorithm.

3.3 Diagonal decomposition methods

In this subsection we aim to propose methods for decomposing matrix A in the form $A = Q - diag(w)$. In the light of the discussion given in Subsection 3.1, we also aim to determine vectors w having nonnegative components as small as possible. In order to manage numerical errors, we also aim to have integer components for w. We propose the following procedure `DiagDecomp`(A,v), which takes as inputs a symmetric matrix $A \in \Re^{n\times n}$ and a vector $v \in \Re^n$. The outputs are a nonnegative vector $w \in \Re^n$ and a corresponding matrix Q which results to be positive semidefinite.

Procedure **DiagDecomp**(*inputs*: A, v; *outputs*: Q, w)
if A is positive semidefinite
then $Q := A$ and $w := 0$
else
let $mask$ be the vector such that:

$$mask(r) = \begin{cases} 1 & if\ row[A,r] \neq 0 \\ 0 & if\ row[A,r] = 0 \end{cases}$$

$v(r) := 0$ for all r such that $mask(r) = 0$;
$T = A + diag(v)$;
let $\tilde{A}$ be the submatrix of T made by its nonzero rows and columns;
let $\tilde{\alpha} \in \Re$ be the smallest eigenvalue of $\tilde{A}$;
let w be the vector such that $w(r) := max\{0, \lceil v(r) - \tilde{\alpha} * mask(r)\rceil\}$;
$Q := A + diag(w)$;
end if;
end proc.

To verify the positive semidefiniteness of Q, let

$$\Delta w = w - (v - \tilde{\alpha}\cdot mask) \geq 0,$$

so that:

$$Q = A + diag(w) = (A + diag(v - \tilde{\alpha}\cdot mask)) + diag(\Delta w).$$

By means of a known result on the perturbation of symmetric matrices ([2]), condition $\Delta\tilde{w}\geq 0$ implies that the smallest eigenvalue of Q is greater than or equal to the smallest eigenvalue of $(A + diag(v - \tilde{\alpha}\cdot mask))$. As a consequence, since $(A + diag(v - \tilde{\alpha}\cdot mask))$ is positive semidefinite (for the way $\tilde{\alpha}$ is found) then Q is positive semidefinite too.

Different decomposition can be obtained by starting from different vectors v. In our study we have considered the following vectors v:

- $v^1 = 0$;
- $v^2 = -diag(A)$;
- $v^3 = (v_i^3) \in \Re^n$ where $v_i^3 = 0$ if $a_{ii} \geq 0$, while $v_i^3 = -a_{ii}$ if $a_{ii} < 0$;
- $v^4 = (v_i^4) \in \Re^n$ where $v_i^4 = -a_{ii} + \sum_{j=1,j\neq i}^{n} |a_{ij}|$.

Vector v^1 represents the most trivial chance. Since a semidefinite matrix has nonnegative diagonal elements, also vectors v^2 (which vanishes the diagonal elements of the matrix) and v^3 (which vanishes just the negative diagonal elements of the matrix) can be proposed. On the other hand, vector v^4 is based on diagonal dominance properties. Notice that, given an indefinite matrix A, vectors v^1, v^2 and v^3 provide a nonpositive eigenvalue $\tilde{\alpha}$, while v^4 provides a nonnegative eigenvalue $\tilde{\alpha}$. Two more vectors v^5 and v^6 can be obtained by applying the same approach of vectors v^2 and v^4 to the matrix $A_- = -V diag(\lambda_-)V^T$, where $A = V diag(\lambda)V^T$ is the canonical form of A and λ_- is the vector obtained from λ by vanishing the positive elements.

Example 3.1 Let us consider the following square symmetric matrix:

$$A = \begin{bmatrix} 8 & 2 & 3 & 4 & 7 & -7 \\ 2 & -4 & -1 & 5 & -5 & 8 \\ 3 & -1 & 4 & 6 & -4 & -1 \\ 4 & 5 & 6 & 0 & 8 & -6 \\ 7 & -5 & -4 & 8 & 6 & -2 \\ -7 & 8 & -1 & -6 & -2 & -4 \end{bmatrix}$$

and let us decompose it by means of procedure `DiagDecomp`(A,v). The output vectors w^i, $i = 1,\ldots,6$, obtained from the previously described input vectors v^i, $i = 1,\ldots,6$, are the followings:

$$\begin{array}{lcl} w^1 = [20,20,20,20,20,20]^T & , & w^2 = [11,23,15,19,13,23]^T \\ w^3 = [18,22,18,18,18,22]^T & , & w^4 = [11,21,7,25,16,24]^T \\ w^5 = [15,21,16,19,17,22]^T & , & w^6 = [9,25,10,22,15,22]^T \end{array}$$

A computational experience made in testing procedure `DiagDecomp`(A,v) shows that vectors v^4 and v^6 seem to produce output vectors w with a smaller mean of the components (thus reducing the value of $Err_B(x)$). For this very reason we decided to use in the computational test of the branch and bound algorithm just vectors v^1 (the trivial one), v^4 and v^6. Let us

[2]Let $B, P, M \in \Re^{n\times n}$ be symmetric matrices such that $B = M + P$. Let also $\beta_1 \geq \beta_2 \geq \ldots \geq \beta_n$ be the eigenvalues of B, $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_n$ be the eigenvalues of M, $\pi_1 \geq \pi_2 \geq \ldots \geq \pi_n$ be the eigenvalues of P. Then, it is $\pi_n \leq \beta_k - \mu_k \leq \pi_1\ \forall k = 1,\ldots,n$.

note that none of the vectors w^i, $i = 1, \dots, 6$, is dominated by the others in the sense of Bomze [6].

4 General linear relaxations

This approach is based on the decomposition of matrix A in the form:

$$A = Q - \sum_{i=1}^{\nu_-(A)} d_i d_i^T \tag{2}$$

where $Q \in \Re^{n\times n}$ is symmetric and positive semidefinite, $d_1, \dots, d_{\nu_-(A)} \in \Re^n$ are linearly independent, and $\nu_-(A)$ is the number of negative eigenvalues of A. Such a decomposition allows us to rewrite function f as follows:

$$f(x) = \frac{1}{2}x^T Qx + c^T x - \frac{1}{2}\sum_{i=1}^{\nu_-(A)} (d_i^T x)^2$$

so that the concave part $-\sum_{i=1}^{\nu_-(A)}(d_i^T x)^2$ can be linearized with respect to the functions $d_i^T x$, $i = 1, \dots, \nu_-(A)$. In this light, we can consider branch and bound schemes where the feasible region of the current subproblem is splitted with respect to one of the functions $d_i^T x$.

4.1 Main properties

Given a pair of vectors $\alpha, \beta \in \Re^{\nu_-(A)}$, such that $\alpha \le \beta$, we can denote with $D(\alpha, \beta) = \{x \in \Re^n : \ \alpha_i \le d_i^T x \le \beta_i \ , \ i = 1, \dots, \nu_-(A)\}$ the set generated by α and β. The concave part $-\frac{1}{2}\sum_{i=1}^{\nu_-(A)}(d_i^T x)^2$ of $f(x)$ can be linearized over $D(\alpha, \beta)$ as follows:

$$f_D(x) = \frac{1}{2}x^T Qx - \frac{1}{2}\sum_{i=1}^{\nu_-(A)} [d_i^T x(\alpha_i + \beta_i) - \alpha_i\beta_i] + c^T x = \frac{1}{2}x^T Qx + \tilde{c}^T x + \tilde{c}_0$$

with:

$$\tilde{c} = c - \frac{1}{2}\sum_{i=1}^{\nu_-(A)} d_i(\alpha_i + \beta_i) \quad \text{and} \quad \tilde{c}_0 = \frac{1}{2}\sum_{i=1}^{\nu_-(A)} \alpha_i\beta_i$$

Notice that $f_D(x)$ is a relaxation of $f(x)$ over the set $D(\alpha, \beta)$, which allows to define the relaxed convex subproblem for the branch and bound schemes:

$$P_D(\alpha, \beta) : \begin{cases} \min f_D(x) \\ x \in X \cap D(\alpha, \beta) \end{cases}$$

The following result can be proved analogously to Theorem 3.1, where:

$$\begin{aligned} Err_D(x, i) &= \frac{1}{2}(\beta_i - d_i^T x)(d_i^T x - \alpha_i) \ , \quad i = 1, \dots, \nu_-(A) \\ Err_D(x) &= f(x) - f_D(x) = \frac{1}{2}\sum_{i=1}^{\nu_-(A)} (\beta_i - d_i^T x)(d_i^T x - \alpha_i) = \sum_{i=1}^{\nu_-(A)} Err_D(x, i) \end{aligned}$$

Theorem 4.1 *Let us consider problems P and $P_D(l,u)$ and let*

$$x^* = \arg \min_{x \in X \cap D(\alpha,\beta)} \{f(x)\} \quad \textit{and} \quad y^* = \arg \min_{x \in X \cap D(\alpha,\beta)} \{f_D(x)\}.$$

Then: $0 \leq f(x^*) - f_D(y^*) \leq Err_D(y^*)$.

4.2 Branch and bound scheme

First notice that $2\nu_-(A)$ linear programs are needed to determine the following values for $i = 1, \ldots, \nu_-(A)$:

$$\tilde{\alpha}_i = \min_{x \in X}\{d_i^T x\} \quad \text{and} \quad \tilde{\beta}_i = \max_{x \in X}\{d_i^T x\}$$

The following branch and bound scheme can then be given.

Procedure **Solve$_2$**(P)
  determine a decomposition $A = Q - \sum_{i=1}^{\nu_-(A)} d_i d_i^T$;
  determine $\tilde{\alpha}$ and $\tilde{\beta}$;
  fix the positive value ϵ; $UB := +\infty$;
  Explore$_2(\tilde{\alpha}, \tilde{\beta})$;
  x^* is an optimal solution and UB is its value;
end proc.

The core of the algorithm is the following recursive procedure "Explore$_2$()".

Procedure **Explore$_2$**(α, β)
  if $X \cap D(\alpha, \beta) \neq \emptyset$ *then*
    Let $\overline{x}$ be the optimal solution of $P_D(\alpha, \beta)$;
    if $f(\overline{x}) < UB$ *then* $UB := f(\overline{x})$; $x^* := \overline{x}$ *end if*;
    if $f_D(\overline{x}) < UB$ *and* $Err_D(\overline{x}) > \epsilon$ *then*
      let $i = \arg\max_{j \in \{1,\ldots,\nu_-(A)\}}\{Err_D(\overline{x}, j)\}$;
      define $\alpha' := \alpha$, $\alpha'' := \alpha$, $\beta' := \beta$, $\beta'' := \beta$;
      let $\beta_i' := d_i^T \overline{x}$, $\alpha_i'' := d_i^T \overline{x}$;
      if $d_i^T A\overline{x} + d_i^T c > 0$ *then*
        Explore$_2(\alpha', \beta')$;
        Explore$_2(\alpha'', \beta'')$;
      else
        Explore$_2(\alpha'', \beta'')$;
        Explore$_2(\alpha', \beta')$;
      end if;
    end if;
  end if;
end proc.

Problem $P_D(\alpha, \beta)$ can be solved by any of the known algorithms for convex quadratic programs. Notice that the visit criterion $d_i^T A\overline{x} + d_i^T c > 0$ implies that we firstly solve the subproblem where the function $f(x)$ restricted along the direction d_i is locally decreasing. Notice also that condition $Err_D(\overline{x}) > \epsilon > 0$ implies that for any index i such that $i = \arg\max_{j \in \{1,\ldots,n\}}\{Err_D(\overline{x}, j)\}$ it results $Err_D(\overline{x}, i) > 0$.

4.3 Decomposition methods

In this subsection we aim to propose methods for decomposing matrix A in the form $A = Q - \sum_{i=1}^{\nu_-(A)} d_i d_i^T$. Two different methods will be proposed; the first one is the so called "Lagrange's decomposition method", while the second one uses the canonical form of symmetric matrices. The decomposition method of Lagrange (see [13]), based on the "Law of Inertia" ([3]), provides for any symmetric matrix A a decomposition of the kind $A = Q - \sum_{i=1}^{\nu_-(A)} d_i d_i^T$, where Q is positive semidefinite with $d_1, \ldots, d_{\nu_-(A)}$ linearly independent. Such a decomposition is described in procedure `DecompLagrange`(A), where row[T,r] denotes the r-th row of matrix T (see also [8]).

Procedure **DecompLagrange**(*inputs*: A; *outputs*: Q, k, d_1, ..., d_k)
T:=A; D:=0; k:=0;
while T$\neq$ 0 *do*
  if $T[i,i] = 0 \; \forall i \in \{1, \ldots, n\}$
    then select $r \in \{1, \ldots, n\}$ *such that* row[T,r]$\neq$ 0 *and set* T[r,r]:=-1;
    else select $r \in \{1, \ldots, n\}$ *such that* $r = \arg\max_{T[r,r]\neq 0}\{T[r,r]\}$;
  end if;
  v:=row[T,r]; α:=1/T[r,r];
  M:=-αvv^T; T:=T+M;
  if $\alpha < 0$ *then* k:=k+1; d_k :=v$\sqrt{-\alpha}$; D:=D+M *end if*;
end do;
Q=A+D;
end proc.

The following procedure `DecompEigen`(A) is based on the very well known properties of eigenvalues and eigenvectors of symmetric matrices.

Example 4.1 Applying the previous procedures to matrix A of Example 3.1 (having two negative eigenvalues) we obtain the next vectors d_1 and d_2:

$$\text{DecompLagrange} \; : \; \begin{cases} d_1 = [0, -2.7484, 0, 3.7902, -3.9232, 2.0059]^T \\ d_2 = [0, 2.4352, 0, 0, 0, -2.8724]^T \end{cases}$$
$$\text{DecompEigen} \; : \; \begin{cases} d_1 = [-0.9277, 2.5077, 0.8855, -2.0231, 1.3550, -2.3097]^T \\ d_2 = [-0.3627, 0.3596, -1.0896, 1.0410, -0.9956, -1.3920]^T \end{cases}$$

[3](The Law of Inertia for symmetric matrices [13]) Let $A \in \Re^{n\times n}$, $A \neq 0$, be a symmetric matrix and let $u_1, \ldots, u_r \in \Re^n \setminus \{0\}$ be $1 \leq r \leq n$ linearly independent vectors such that:

$$A = \sum_{i=1}^{r} \alpha_i u_i u_i^T, \text{ where } \alpha_i \in \{-1, 1\} \; \forall i = 1, \ldots, r.$$

Then the number of positive and the number of negative coefficients α_i are independent of the chosen set of linearly independent vectors $u_1, \ldots, u_r$.

Procedure DecompEigen(*inputs*: A; *outputs*: Q, k, $d_1, \ldots, d_k$)
D:=0; k:=0; determine the eigenvalues λ_i, $i = 1, \ldots, n$, of A and the corresponding eigenvectors v_i, $i = 1, \ldots, n$;
for i from 1 *to n do*
  if $\lambda_i < 0$ *then*
    k:=k+1; $d_k := v_i\sqrt{-\lambda_i}$; $M := -\lambda_i v_i v_i^T$; D:=D+M;
  end if;
end do;
Q=A+D;
end proc.

5 Computational results

The previously described branch and bound methods and decomposition procedures have been fully implemented with the software MatLab 7.2 R2006a on a computer having 2 Gb RAM and two Xeon dual core processors at 2.66 GHz. In the computational tests we considered 3 diagonal decompositions, namely Diag1, Diag4 and Diag6, obtained by using in `DiagDecomp`(A,v) vectors v^1, v^4 and v^6, respectively (see Subsection 3.3). We also considered both `DecompLagrange`(A) and `DecompEigen`(A) procedures. The cases of box feasible region and nonbox ones have been considered separately. The problems have been randomly created; in particular, matrices and vectors A, c, B, b, $\tilde{l}$, $\tilde{u}$, have been generated with components in the interval [-10,10] by using the "rand()" MatLab function (numbers generated with uniform distribution). We assumed also epsilon=0.1. Within the branch and bound procedures, the linear and convex quadratic problems have been solved with the "linprog()" and "quadprog()" MatLab functions. For each class of problems (dimension and type of region) 1500 randomly generated problems have been solved by means of the considered algorithms. The average number of iterations spent by the branch and bound methods to solve the problems is given as the result of the single test and as an index of the performance of the used decomposition.

5.1 Role of negative eigenvalues

In order to verify the impact of the numbers of negative eigenvalues of A in the performance of the considered methods, we solved problems of dimension $n = 8$ having a number of negative eigenvalues of matrix A from 1 to 8 ($num = 1, \ldots, 8$). The computational results regarding dense matrices A are provided in Table 1 and Table 2.

Taking into account that we considered just problems having dimension $n = 8$, we can observe that for matrices having a small number of negative eigenvalues the linear decomposition methods appears to have better performances than the diagonal ones. This behaviour reverses for matrices having a big number of negative eigenvalues. Such a kind of results can be justified recalling that in the linear decomposition methods we deal with a number of

num	Diag1	Diag4	Diag6	Lagrange	Eigen
1	197.96	24.437	89.551	7.5514	6.9653
2	170.75	39.091	68.069	22.496	22.293
3	80.092	27.953	36.052	47.863	54.368
4	41.039	19.481	19.968	91.107	108.39
5	19.237	12.014	11.217	192.67	186.8
6	12.173	8.541	8.1047	270.3	321.03
7	9.456	7.0853	6.9827	270.76	628.95
8	8.4787	6.584	6.584	175.91	1111.4

Table 1: Average Iterations - Box Region

num	Diag1	Diag4	Diag6	Lagrange	Eigen
1	430.79	40.923	204.12	8.86	6.356
2	567.15	148.01	278.05	33.121	20.979
3	413.39	186.95	231.66	86.509	50.737
4	238.78	150.48	152.16	178.61	106.39
5	154.81	116.03	111.12	353.68	201.59
6	114.28	88.684	85.897	516.27	383.69
7	89.769	70.451	69.323	568.54	778.02
8	78.454	64.765	64.765	572.85	1439.2

Table 2: Average Iterations - Non Box Region

vectors d_i equal to the number of negative eigenvalues. Notice also that for matrices having a big number of negative eigenvalues `DecompLagrange`(A) has a better performance than `DecompEigen`(A).

5.2 Performance for dense matrices

In the previous subsection we saw that the number of negative eigenvalues affects the performances of the various methods. For this reason, we carried on three computational experiences were we considered dense matrices A having the 25%, 50%, 75% of negative eigenvalues, respectively. Both box

n	Diag1	Diag4	Diag6	Lagrange	Eigen
7	89.962	22.047	38.785	17.276	20.625
8	170.75	39.091	68.069	22.496	22.293
9	347.19	65.517	123.1	24.915	24.564
10	300.76	80.323	109.79	75.783	66.807
11	471.39	130.88	172.61	97.603	71.129
12	865.76	201.96	300.92	115.02	76.049
13	1426.2	315.68	495.2	139.35	79.366
14	–	365.9	459.47	–	213.11
15	–	608.88	749.37	–	222.62
16	–	894.68	1249.9	–	233.28

Table 3: Average Iterations - Box Region - 25% eigenvalues

n	Diag1	Diag4	Diag6	Lagrange	Eigen
7	304.41	92.775	162.57	27.942	19.71
8	567.15	148.01	278.05	33.121	20.979
9	1132.6	251.05	516.11	37.11	22.465
10	1428.4	552.74	737.99	128.57	60.836
11	2358.1	786.84	1157.7	149.39	63.49
12	3263.6	1071.6	1605	169.84	67.13
13	4367.5	1274.5	2046	179.8	66.09
14	–	2581.6	3331.4	–	176.35
15	–	3242.1	4152.5	–	175.28
16	–	3673.6	5168.2	–	178.21

Table 4: Average Iterations - Non Box Region - 25% eigenvalues

n	Diag1	Diag4	Diag6	Lagrange	Eigen
5	23.326	8.0727	10.091	12.041	20.801
6	29.197	11.913	13.825	23.298	38.03
7	35.012	15.517	16.852	43.38	63.244
8	47.125	21.54	22.669	88.667	108.53
9	57.665	27.063	26.841	164.39	177.63
10	67.833	33.052	32.695	342.51	303.44
11	84.577	41.065	38.091	629.71	493.07
12	94.166	47.728	43.173	1093.7	798.89
13	105.6	55.263	49.219	1920.4	1251.1
14	–	70.56	61.062	–	2068.5
15	–	88.262	70.42	–	3039
16	–	97.498	78.03	–	4843.3

Table 5: Average Iterations - Box Region - 50% eigenvalues

n	Diag1	Diag4	Diag6	Lagrange	Eigen
5	61.95	30.985	37.331	20.88	20.872
6	100.36	54.472	60.607	39.916	37.843
7	162.59	96.832	103.83	75.367	62.353
8	305.77	181.41	189.38	168.62	109.03
9	543.67	340.41	331.77	324.44	178.62
10	717.15	433.56	430.98	579.59	295.29
11	1074.6	705.8	682.29	1099.4	473.33
12	1471.6	1011.8	930.4	1867.5	762.74
13	1969	1359.5	1238.4	3222.3	1238.3
14	–	1870.8	1725.1	–	1949
15	–	2587.4	2337.8	–	2953.5
16	–	2962.1	2625.4	–	4498.6

Table 6: Average Iterations - Non Box Region - 50% eigenvalues

n	Diag1	Diag4	Diag6	Lagrange	Eigen
7	11.918	7.7819	7.5083	119.57	139.2
8	12.173	8.541	8.1047	270.3	321.03
9	13.444	9.608	9.2467	611.27	763.81
10	14.376	10.726	10.212	1301.2	1832.4
11	19.926	14.369	13.373	2511.8	2210.5
12	20.335	15.144	14.193	4626.6	5070.6

Table 7: Average Iterations - Box Region - 75% eigenvalues

n	Diag1	Diag4	Diag6	Lagrange	Eigen
7	80.256	61.307	58.984	226.81	160.93
8	114.28	88.684	85.897	516.27	383.69
9	173.89	138.99	137.16	1080	916.11
10	265.66	197.06	195.19	2345	2282.6
11	430.95	344.07	330.01	3875.8	2512.8
12	521.98	396.98	383.05	6596.2	5765.1

Table 8: Average Iterations - Non Box Region - 75% eigenvalues

regions and non box regions have been considered. The results are given in Table 3 and Table 4 as regards to matrices A having the 25% of negative eigenvalues. In Table 5 and Table 6 we provides the results related to matrices A having the 50% of negative eigenvalues, while Table 7 and Table 8 summarizes the results given by matrices A having the 75% of negative eigenvalues.

The obtained computational results show the following behaviours of the considered methods:

- in the case of 25% of negative eigenvalues, the linear decomposition methods have better performances than the diagonal ones, for both box regions and non box ones;
- for matrices A having 50% of negative eigenvalues, the diagonal decomposition methods have better performances than the linear ones for any dimension in the case of box regions, for $n > 13$ in the case of non box ones;
- in the case of 75% of negative eigenvalues, the diagonal decomposition methods have better performances than the linear ones, for both box regions and non box ones;
- among the diagonal decomposition methods, the performance of procedure `DiagDecomp`(A,v^1) is worst than `DiagDecomp`(A,v^4) and than `DiagDecomp`(A,v^6) ones;
- among the linear decomposition methods, the performance of procedure `DecompLagrange`(A) is worst than `DecompEigen`(A) one; just in

the case of box regions and small dimensions n the performances of the two methods are comparable;

- the performance of procedure DecompEigen(A) is comparable for box regions and non box ones; the other methods have the best performance in the box case.

These results confirm that in the case of matrices A having a small number of negative eigenvalues the linear decomposition methods (in particular, DecompEigen(A) procedure) have better performances than the diagonal ones. The situation reverses when the number of negative eigenvalues increases (in particular, DiagDecomp(A,v^4) and DiagDecomp(A,v^6) procedures provide the best performances).

5.3 Performance for sparse matrices

In Table 9 and Table 10 we now provide some results concerning sparse matrices A (at least 66% of zero elements).

n	Diag1	Diag4	Diag6	Lagrange	Eigen
7	33.572	8.5868	11.168	32.1	29.528
8	45.628	11.62	14.685	79.121	58.833
9	70.964	15.199	18.891	127.27	106.48
10	87.14	17.836	21.82	233.84	176.07
11	117.55	22.857	27.499	494.46	317.01
12	155.91	29.862	36.064	900.58	544.18
13	186.6	34.132	38.693	1328.2	867.11
14	–	44.071	46.205	–	1537.3
15	–	51.322	52.389	–	2399.6
16	–	61.675	72.889	–	3653.8

Table 9: Average Iterations - Box Region - Sparse Matrix A

n	Diag1	Diag4	Diag6	Lagrange	Eigen
7	149.29	52.031	65.806	62.444	35.508
8	239.29	87.458	109.55	129.01	63.408
9	383.33	144.24	173.99	268.74	111.95
10	676.52	239.17	282.1	481.29	195.6
11	1033.1	352.94	420.19	934.05	334.63
12	1503.4	521.95	597.47	1549	553.18
13	2152.6	748.66	848.13	2364.3	899.92
14	–	1278.8	1395.7	–	1587
15	–	1768.3	1900.6	–	2414.9
16	–	2199	2350	–	3753.9

Table 10: Average Iterations - Non Box Region - Sparse Matrix A

The behaviour of the considered methods is similar to the case of dense matrices with 50% of negative eigenvalues, with a reduction (for all the methods) of the number of iterations.

5.4 Performance for higher dimensions problems

To complete the overall computational test we solved some problems with dimension from $n = 20$, by using the methods which resulted to have the best performance. The obtained results are given in Table 11. Both box regions and non box regions have been considered, as well as matrices A having 25%, 50% and 75% of negative eigenvalues. For each category 100 random problems have been solved.

n	25% - Eigen		50% - Diag6		75% - Diag6	
	Box	Non Box	Box	Non Box	Box	Non Box
20	674.56	583.58	178.94	5460	35.76	3709.6
25	1900.8	1422.2	337.6	6635	52.9	7021.9
30	9739.3	8982.3	491.48	7206	111.44	7049.7
35	14926	15262	1282.2	11694	233.84	9113
40	–	–	2394.3	–	286	–
45	–	–	3016	–	742	–
50	–	–	5448.1	–	1108.1	–
55	–	–	6154.2	–	1456	–
60	–	–	6345.9	–	2182.4	–
65	–	–	8053	–	3967.4	–

Table 11: Average Iterations - Dense Matrix A

It is worth noticing that the number of iterations needed by the used methods to solve problems with non box region quickly augments when the dimension of the problems increases. This does not happen when problems with box region and a matrix A with at least 50% of negative eigenvalues are solved by procedure DiagDecomp(A,v^6).

6 Conclusions

In this paper we propose various methods to solve indefinite quadratic programs with polyhedral region. In the studied methods no variable transformations are needed, so that peculiarities of the feasible region are maintained. In particular, the performances of the various methods have been analyzed separately for problems having box region and for problems having non box one. Procedure DecompEigen(A) provides a similar computational behaviour for box regions and for non box ones; all the other methods show worst performances for the non box regions case.

The performances of the methods change with respect to the number of negative eigenvalues of the quadratic form of the objective function. In particular, for few negative eigenvalues procedure DecompEigen(A) seems to have the best performance, even if the number of iterations needed to solve the problems increases quickly with the dimension of the problems themselves. In the case of at least 50% of negative eigenvalues, the best performance is given by the DiagDecomp(A,v) procedures which also result to be stable in the case of box regions.

Finally, notice that in [10] a solution method has been proposed for box quadratic problems. Such a method is based on a variables transformation performed by means of the eigenvectors of matrix A. It is worth noticing that procedure DecompEigen(A) provides the same iterations than the method proposed in [10], without the need of any variables transformations.

References

[1] Barrientos O. and R. Correa (2000), "An algorithm for global minimization of linearly constrained quadratic functions", *Journal of Global Optimization*, vol.16, pp.77-93.

[2] Best M.J. and B. Ding (1997), "Global and local quadratic minimization", *Journal of Global Optimization*, vol.10, pp.77-90.

[3] Best M.J. and B. Ding (2000), "A decomposition method for global and local quadratic minimization", *Journal of Global Optimization*, vol.16, pp.133-151.

[4] Bomze I.M. and G. Danninger (1994), "A finite algorithm for solving general quadratic problems", *Journal of Global Optimization*, vol.4, pp.1-16.

[5] Bomze I.M. (2002), "Branch-and-bound approaches to standard quadratic optimization problems", *Journal of Global Optimization*, vol.22, pp.17-37.

[6] Bomze I.M. and M. Locatelli (2004), "Undominated d.c. decompositions of quadratic functions and applications to branch-and-bound approaches", *Computational Optimization and Applications*, vol.28, pp.227-245.

[7] Cambini R. and C. Sodini (2002), "A finite algorithm for particular d.c. quadratic programming problem", *Annals of Operations Research*, vol.117, pp.33-49.

[8] Cambini R. and C. Sodini (2005), "Decomposition methods for solving nonconvex quadratic programs via branch and bound", *Journal of Global Optimization*, vol.33, n.3, pp.313-336.

[9] Churilov L. and M. Sniedovich (1999), "A concave composite programming perspective on D.C. programming", in *Progress in Optimization*, edited by A. Eberhard, R. Hill, D. Ralph and B.M. Glover, *Applied Optimization*, vol.30, Kluwer Academic Publishers, Dordrecht.

[10] De Angelis P.L., Pardalos P.M. and G. Toraldo (1997), "Quadratic Programming with Box Constraints", in *Developments in Global Optimization*, edited by I.M. Bomze et. al., Kluwer Academic Publishers, Dordrecht, pp.73-93.

[11] Ellero A. (1996), "The optimal level solutions method", *Journal of Information & Optimization Sciences*, vol.17, no.2, pp.355-372.

[12] Falk J.E. and R.M. Soland (1969), "An algorithm for separable nonconvex programming problems", *Management Science*, vol.15, no.9, pp.550-569.

[13] Gantmacher F.R. (1960), "The theory of matrices", vol.1, Chelsea Publishing Company, New York.

[14] Hansen P., Jaumard B., Ruiz M. and J. Xiong (1993), "Global minimization of indefinite quadratic functions subject to box constraints", *Naval Research Logistics*, vol.40, pp.373-392.

[15] Hiriart-Urruty J.B. (1985), "Generalized differentiability, duality and optimization for problems dealing with differences of convex functions", in *Convexity and Duality in Optimization*, Lecture Notes in Economics and Mathematical Systems, vol.256, Springer-Verlag.

[16] Horst R. (1976), "An algorithm for nonconvex programming problems", *Mathematical Programming*, vol.10, no.3, pp.312-321.

[17] Horst R. (1980), "A note on the convergence of an algorithm for nonconvex programming problems", *Mathematical Programming*, vol.19, no.2, pp.237-238.

[18] Horst R. and H. Tuy (1990), "Global Optimization", Springer-Verlag, Berlin.

[19] Horst R., Pardalos P.M. and N.V. Thoai (1995), "Introduction to Global Optimization", *Nonconvex Optimization and Its Applications*, vol.3, Kluwer Academic Publishers, Dordrecht.

[20] Horst R. and Nguyen Van Thoai (1996), "A new algorithm for solving the general quadratic programming problem", *Computational Optimization and Applications*, vol.5, no.1, pp.39-48.

[21] Konno H. (1976), "Maximization of a convex quadratic function under linear constraints", *Mathematical Programming*, vol.11, no.2, pp.117-127.

[22] Le Thi Hoai An and Pham Dinh Tao (1997), "Solving a class of linearly constrained indefinite quadratic problems by D.C. algorithms", *Journal of Global Optimization*, vol.11, pp.253-285.

[23] Muu L.D. and W. Oettli (1991), "An algorithm for indefinite quadratic programming with convex constraints", *Operations Research Letters*, vol.10, no.6, pp.323-327.

[24] Pardalos P.M., Glick J.H. and J.B. Rosen (1987), "Global minimization of indefinite quadratic problems", *Computing*, vol.39, no.4, pp.281-291.

[25] Pardalos P.M. (1991), "Global optimization algorithms for linearly constrained indefinite quadratic problems", *Computers & Mathematics with Applications*, vol.21, pp.87-97.

[26] Phong Thai Quynh, An Le Thi Hoai and Tao Pham Dinh (1995), "Decomposition branch and bound method for globally solving linearly constrained indefinite quadratic minimization problems", *Operations Research Letters*, vol.17, no.5, pp.215-220.

[27] Rosen J.B. and P.M. Pardalos (1986), "Global minimization of large scale constrained concave quadratic problems by separable programming", *Mathematical Programming*, vol.34, pp.163-174.

[28] Tuy H. (1995), "D.C. Optimization: theory, methods and algorithms", in *Handbook of Global Optimization*, edited by R. Horst and P.M. Pardalos, *Nonconvex Optimization and Its Applications*, vol.2, Kluwer Academic Publishers, Dordrecht, pp.149-216.

[29] Tuy H. (1998), "Convex Analysis and Global Optimization", *Nonconvex Optimization and Its Applications*, vol.22, Kluwer Academic Publishers, Dordrecht.